STRUCTURED ALGEBRAIC ADAPTERS FOR FINE-TUNING OF LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Parameter-efficient fine-tuning restricts the trainable update space of a pretrained model, commonly through low-rank factorization. We study Structured Algebraic Adapters (SAA), which instead learn coefficients over a fixed family of block-wise signed-permutation operators. This parameterization supports residual-state and weight-space adaptation without imposing an explicit low-rank bottleneck, while keeping the pretrained backbone frozen. We evaluate SAA against LoRA at approximately matched trainable-parameter budgets on mathematical supervised fine-tuning, data-to-text generation, and reinforcement learning with verifiable rewards. On Ministral-3B with canonical-boxed training targets, weight-space SAA reaches 76.6% GSM8K accuracy under a MetaMathQA-aligned evaluation prompt, compared with 74.2% for projection-level LoRA. However, improvements in answer formatting do not imply uniform gains in mathematical correctness, and neither method improves MATH performance over the frozen backbone in this setting. On GPT-2 Medium, selected SAA and LoRA configurations achieve comparable E2E generation scores. In our Qwen3-0.6B RL experiments, SAA obtains higher answer accuracy than LoRA and the frozen backbone on all four evaluations. These single-seed results motivate structured operator parameterizations as an alternative to low-rank adaptation, while revealing task- and placement-dependent trade-offs.

1 INTRODUCTION

Parameter-efficient fine-tuning (PEFT) adapts pretrained models while updating only a small set of parameters. LoRA (Hu et al., 2022) achieves this through an additive weight update

$$\Delta \boldsymbol{W} = \boldsymbol{B}\boldsymbol{A}, \qquad \operatorname{rank}(\Delta W) \le r, \tag{1}$$

where $\boldsymbol{B} \in \mathbb{R}^{d_{\text{out}} \times r}$ and $\boldsymbol{A} \in \mathbb{R}^{r \times d_{\text{in}}}$. Its trainable parameter count is $r(d_{\text{out}} + d_{\text{in}})$, linking the budget to an explicit bound on update rank.

A compact parameterization need not impose this rank constraint. FourierFT (Gao et al., 2024) learns selected coefficients in a fixed Fourier representation, while Quantum-PEFT (Koike-Akino et al., 2025) constructs updates through compact quantum-inspired matrix parameterizations. These approaches establish alternatives to low-rank factorization. The question we investigate is therefore not whether such alternatives exist, but whether a particular algebraically structured operator family provides a useful space for adaptation across placements, tasks, and training objectives.

We introduce Structured Algebraic Adapters (SAA), which represent a correction as

$$\boldsymbol{Q}_\alpha = \sum_{k=1}^{K} \alpha_k \boldsymbol{Q}_k, \tag{2}$$

where the operators $\boldsymbol{Q}_k$ are fixed and only their coefficients are trained. The operators are constructed blockwise from XOR-shift permutations and Walsh sign patterns, yielding real signed-permutation matrices within each block. Their connection to qubit-operator algebra motivates the construction, but neither quantum hardware nor a quantum computational advantage is assumed.

SAA restricts the update to a low-dimensional parameterized space without imposing a corresponding low-rank factorization. For example, a nonzero multiple of a signed-permutation matrix on an

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.

N-dimensional block has rank N despite requiring only one coefficient. This does not guarantee that every learned correction is full rank; rather, it separates coefficient count from an explicit rank bottleneck.

We consider two applications of this operator family: residual-state adaptation, $\boldsymbol{h}' = \boldsymbol{h} + \boldsymbol{Q}_\alpha \boldsymbol{h}$, and weight-space adaptation, $\boldsymbol{W}' = \boldsymbol{W} + \boldsymbol{Q}_\alpha$, with compatible operator dimensions. We compare these with residual LoRA and projection-level LoRA, respectively, while keeping the pretrained backbone frozen. This distinction matters because adapter placement changes which computations can be modified; matched parameter counts alone do not make different placements equivalent.

Our experiments examine supervised mathematical adaptation on Ministral-3B, data-to-text generation on GPT-2 Medium, and reward-based mathematical adaptation on Qwen3-0.6B. We distinguish teacher-forced validation loss, answer-format compliance, and generated-answer correctness. The results are not uniformly ordered across these measures. Weight-space SAA performs favorably in the canonical-boxed math comparison, but improvements over the frozen backbone are task-dependent. On E2E, the selected configurations are comparable to LoRA. Under reward-based optimization, SAA achieves higher answer accuracy across the four reported evaluations. Because the supervised and RL experiments differ in backbone and training data, this contrast does not isolate the training objective as the cause. Our contributions are:

- We formulate a blockwise algebraic operator parameterization for residual-state and weight-space adaptation without an explicit low-rank bottleneck.
- We evaluate SAA against LoRA at approximately matched parameter budgets on mathematical SFT, data-to-text generation, and reward-based mathematical adaptation.
- We distinguish supervised fitting, output-format compliance, and answer correctness, showing that improvements in one measure need not produce improvements in the others.

Our single-seed comparisons characterize the behavior of these configurations rather than establish universal superiority. Isolating the benefit of the specific operator family from other differences in the update space requires additional basis-controlled comparisons.

2 Related Work

Structured and high-rank PEFT. LoRA (Hu et al., 2022) and variants such as DoRA (Liu et al., 2024a) and VeRA (Kopiczko et al., 2024) retain low-rank factorization as a central component of the adaptation. A growing line of work instead explores parameter-efficient updates that need not remain low rank. MoRA (Jiang et al., 2024) uses a trainable square matrix together with fixed compression and expansion operators to obtain high-rank updates at a LoRA-like parameter budget, while HiRA (Huang et al., 2025) uses a Hadamard product with the frozen pretrained weights to increase the attainable update rank. SHiRA (Bhardwaj et al., 2024) takes a different approach by directly training a sparse subset of the pretrained weights.

Among fixed-representation methods, FourierFT (Gao et al., 2024) is particularly related to SAA: it learns a small set of coefficients in a fixed Fourier representation and reconstructs the weight update through an inverse discrete Fourier transform. SAA likewise optimizes coefficients over fixed structured directions, but uses blockwise real signed-permutation operators derived from XOR shifts and Walsh sign patterns rather than Fourier components. Other structured alternatives include butterfly-factorized orthogonal fine-tuning (BOFT) (Liu et al., 2024b), quantum-informed tensor adaptation (QuanTA) (Chen et al., 2024), and quantum-inspired unitary parameterizations such as Quantum-PEFT (Koike-Akino et al., 2025). Finally, representation fine-tuning methods such as ReFT (Wu et al., 2024) modify hidden representations rather than pretrained weight matrices, making them conceptually relevant to the residual-state form of SAA.

Parameter-efficient adaptation. LoRA (Hu et al., 2022) learns additive weight updates through a low-rank factorization while keeping the pretrained backbone frozen. Our approach also restricts the trainable update space, but learns coefficients of fixed algebraic operators rather than two matrix factors. Consequently, its parameter budget controls the number of coefficients rather than an explicit low-rank bottleneck.

Structured update parameterizations. FourierFT (Gao et al., 2024) learns a subset of spectral coefficients and reconstructs additive weight updates through an inverse discrete Fourier transform. It is closely related to SAA in using a fixed operator family with trainable coefficients, but differs in the underlying representation: SAA uses blockwise signed-permutation operators rather than Fourier components. BOFT (Liu et al., 2024b) instead composes sparse orthogonal factors in a butterfly structure to transform pretrained weights. Unlike this multiplicative orthogonal parameterization, SAA constructs additive corrections without requiring the resulting operator to remain orthogonal. PSOFT (Wu et al., 2026) confines adaptation to SVD-derived principal subspaces, combining orthogonal transformations with trainable scaling vectors that relax orthogonality. Unlike PSOFT, SAA does not require a decomposition of the pretrained weights and learns corrections in a fixed algebraic operator family rather than restricting adaptation to their principal subspaces.

Quantum-inspired adaptation. Quantum-PEFT (Koike-Akino et al., 2025) constructs additive updates using compact quantum-inspired parameterizations of matrix factors, including Pauli rotations and Taylor approximations to exponential mappings. QuIC (Raj & Coyle, 2025) builds multiplicative adapters from compound matrices derived from small trainable matrices, with orthogonality inherited from the base parameterization. SAA instead directly optimizes a linear combination of fixed real algebraic operators, without circuit composition or orthogonality constraints on the learned correction. We study this parameterization in both residual-state and weight-space adaptation, under supervised fine-tuning and reward-based optimization.

3 METHOD

3.1 BLOCK-STRUCTURED RESIDUAL ADAPTER

We consider a parameter-efficient adapter acting either on the hidden representation or weights of a frozen model:

$$\boldsymbol{h}' = (\boldsymbol{I} + \boldsymbol{Q}_\alpha)\boldsymbol{h}, \qquad \boldsymbol{Q}_\alpha = \sum_{k=1}^{K} \alpha_k \boldsymbol{Q}_k, \tag{3}$$

where the $\boldsymbol{Q}_k$ are fixed structured operators and only their coefficients are trained. The operators are real signed permutations constructed from XOR shifts and Walsh sign patterns.

3.2 INSERTION SITES

In the two-site residual configuration, both adapters are placed after the final decoder block. Let $\boldsymbol{h}_L$ denote the output of the final block, let

$$\mathrm{RMSNorm}(\boldsymbol{u}) = \boldsymbol{g} \odot \frac{\boldsymbol{u}}{\sqrt{\frac{1}{D}\|\boldsymbol{u}\|_2^2 + \varepsilon}}, \tag{4}$$

denote the frozen final normalization, and let $\boldsymbol{W}_{\mathrm{LM}}$ denote the frozen language-model head. The logits are

$$\boldsymbol{\ell} = \boldsymbol{W}_{\mathrm{LM}} \left(\boldsymbol{I} + \boldsymbol{Q}^{(2)}_{\alpha^{(2)}}\right) \mathrm{RMSNorm}\Big(\left(\boldsymbol{I} + \boldsymbol{Q}^{(1)}_{\alpha^{(1)}}\right) \boldsymbol{h}_L\Big). \tag{5}$$

The first adapter acts between the final decoder block and the final normalization, while the second acts between the normalization and the language-model head.

For teacher-forced evaluation, $\boldsymbol{h}_L$ is independent of the adapter coefficients because both adapters occur after the final decoder block. During autoregressive generation, the adapters can affect later hidden states indirectly through their effect on previously generated tokens.

The second site is algebraically equivalent to an additive modification of the language-model head,

$$\boldsymbol{W}'_{\mathrm{LM}} = \boldsymbol{W}_{\mathrm{LM}} + \boldsymbol{W}_{\mathrm{LM}} \boldsymbol{Q}^{(2)}_{\alpha^{(2)}}. \tag{6}$$

For a fixed $\boldsymbol{h}_L$, the only nonlinear dependence on the residual adapters arises through the normalization denominator following site 1; the remaining operations are linear in the adapted representation.

Accordingly, this configuration modifies the post-decoder readout pathway rather than the internal Transformer computation for the current token.

SAA can also be attached to weight projections inside the network. We distinguish this weight-space configuration from the residual configuration and report its projections, layers, and parameter budgets alongside the corresponding experiments.

3.3 Adapter Optimization

The backbone parameters θ_0 are frozen and only α is trained; $p_{\theta_0,\alpha}$ denotes the adapted model. The structured adapter defines where and how the model can change, whereas the training objective defines which behavior is optimized.

Supervised fine-tuning. Given pairs $(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)})$, we minimize the token-averaged negative log-likelihood of the target tokens:

$$\mathcal{L}_{\mathrm{SFT}}(\alpha) = -\frac{1}{\sum_i T_i} \sum_i \sum_{t=1}^{T_i} \log p_{\theta_0,\alpha}\left(y_t^{(i)} \mid \boldsymbol{x}^{(i)}, \boldsymbol{y}_{<t}^{(i)}\right). \tag{7}$$

Where enabled, an additional L2 penalty discourages deviation from initialization. Regularization settings are specified per experiment. Held-out loss is the token-weighted negative log-likelihood of the target tokens, without the penalty.

Reinforcement learning. For each prompt x, we sample G completions and assign rewards R_g based on answer correctness and the specified format bonus. No reference solution tokens are used as teacher-forced targets. We optimize

$$\begin{aligned} A_g &= \frac{R_g - \bar{R}}{\sigma_R + 10^{-6}}, \\ \mathcal{L}_{\mathrm{RL}} &= -\frac{1}{G} \sum_{g=1}^{G} \frac{A_g}{T_g} \sum_{t=1}^{T_g} \log \pi_\alpha\big(y_t^{(g)} \mid x, y_{<t}^{(g)}\big). \end{aligned} \tag{8}$$

Here, π_α is the adapted model's autoregressive policy, $\bar{R}$ and σ_R are the group reward mean and population standard deviation, and T_g is the completion length. Advantages are held fixed during differentiation; groups with identical rewards are skipped. We use neither policy-ratio clipping nor a KL penalty. Training settings are given in Section 4.1.

4 Experiments

4.1 Setup

Objectives. We evaluate structured algebraic adapters (SAA) under supervised fine-tuning (SFT) and reinforcement learning (RL). Our experiments examine whether these adapters can learn task-specific output conventions while maintaining or improving answer correctness. SFT experiments cover mathematical problem solving and data-to-text generation, while RL experiments study mathematical problem solving with correctness-based rewards. We measure output-format compliance separately from answer correctness where applicable.

Models and datasets. Our SFT experiments include Ministral-3B Base on MetaMathQA-40K and GPT-2 Medium on the E2E natural language generation benchmark. Our RL experiments use Qwen3-0.6B Base trained on Math10K. We evaluate mathematical performance on GSM8K, MATH, MATH-500, and held-out mathematical examples. For E2E, we evaluate generated descriptions against the benchmark's reference texts.

Baselines and controlled comparisons. We compare SAA with LoRA at approximately matched trainable-parameter budgets and include the frozen pretrained model as a reference. Within each controlled comparison, we use the same backbone, data split, prompt, optimization settings, training

duration, and evaluation procedure. We also match the random seed. For RL comparisons, we additionally hold the reward function, rollout sampling, and generation budget fixed.

We distinguish adapters applied to internal weight projections from adapters applied to residual representations, and report their insertion sites explicitly. Parameter counts are reported for each configuration; matching parameter counts alone does not imply identical placement or functional form.

Training and implementation. All experiments are run on NVIDIA RTX 5090 GPUs. Only adapter parameters are updated during adapter fine-tuning; the pretrained backbone remains frozen. We report training budgets, adapter placement, initialization, and output scaling alongside the corresponding experiments.

RL protocol. We train for 1,200 steps with one problem per step and four sampled completions per problem, using temperature 0.8, top-p 0.95, and a maximum of 512 generated tokens. Correct answers receive reward 1, incorrect answers 0, with an additional 0.02 bonus for a boxed answer. We standardize rewards within each group and optimize advantage-weighted, token-averaged completion log-probabilities, without policy-ratio clipping or a KL penalty. Optimization uses AdamW with learning rate 5×10^{-6}, zero weight decay, and gradient clipping. We evaluate the step-1200 checkpoints using greedy decoding.

Checkpoint selection. For each training run, we select the checkpoint with the lowest held-out validation loss. Any exceptions are stated alongside the corresponding results.

Comparison protocol and limitations. Each comparison uses a single training seed. Shared optimization settings provide a controlled comparison, but do not establish the best achievable performance of each method. Checkpoint and configuration selection procedures are reported per experiment. We assess teacher-forced loss, format compliance, and answer correctness separately, since improvements in one need not imply improvements in the others.

4.2 Supervised adaptation on mathematical tasks

We investigate how structured algebraic adapters affect three aspects of supervised adaptation: held-out teacher-forced loss, compliance with the requested answer format, and correctness of generated answers. Our main experiments use Ministral-3B Base fine-tuned on MetaMathQA-40K. We evaluate both residual-state SAA (Eq. 3) and weight-space SAA, comparing them with residual LoRA and projection-level PEFT LoRA, respectively. The pretrained backbone remains frozen.

Generation at the 1.5M budget. Table 1 compares four adapter configurations at approximately matched parameter budgets. SAA-weights achieves the highest verified MetaMathQA, MATH, and MATH-500 scores among the fine-tuned configurations. SAA-weights achieves the highest strict and flexible GSM8K accuracy among the evaluated configurations. The frozen backbone remains stronger on MATH and MATH-500, however, and the lower supervised loss of residual SAA does not translate into uniformly better generation performance than residual LoRA.

Table 1: Ministral-3B math SFT results with a 1.5M parameter budget.

	GSM8K		MMQA holdout					
Method	Strict	Flexible	Strict	Flexible	Verify	Boxed	MATH	MATH-500
Base	0.0	75.2	0.0	40.6	77.8	28.4	41.3	39.4
Residual LoRA	62.2	70.1	43.4	64.0	73.2	91.2	34.4	31.6
PEFT LoRA	72.5	74.7	**47.6**	70.0	76.6	**95.2**	35.5	36.0
Residual SAA	66.1	72.9	39.2	69.2	74.8	79.4	33.3	30.0
SAA-weights	**73.8**	**77.0**	43.8	**71.6**	**77.2**	91.2	**38.9**	**36.4**

Format compliance at the 3M budget. Table 2 reports results with canonical two-line answer tails and appended boxed answers. Target preprocessing differs from the 1.5M experiment, so these tables do not isolate the effect of the parameter budget.

SAA-weights achieves 75.4% strict accuracy and 96.0% boxed-output compliance on the MetaMathQA holdout, compared with 0.0% and 28.4% for the frozen backbone. Verified accuracy remains close to the backbone (77.4% versus 77.8%). Thus, the clearest gain is adherence to the requested answer format while approximately preserving correctness on this holdout. Strict accuracy combines correctness and formatting, whereas boxed-output compliance measures the presence of a boxed answer without requiring it to be correct. The gains are not uniform across tasks or adapter placements. SAA-weights outperforms the other fine-tuned configurations on all reported metrics, but remains below the frozen backbone on MATH and MATH-500. Differences between the fine-tuning and evaluation distributions may contribute to this limited transfer, although we do not isolate their effect. Residual SAA also has substantially lower verified accuracy than SAA-weights in this experiment. All table entries are percentages. GSM8K uses the MetaMathQA-aligned evaluation

Table 2: Ministral-3B math SFT with canonical boxed targets (3M budget).

	GSM8K		MMQA holdout					
Method	Strict	Flexible	Strict	Flexible	Verify	Boxed	MATH	MATH-500
Base	0.0	75.2	0.0	40.6	77.8	28.4	41.3	39.4
Residual LoRA	58.8	62.4	42.4	44.6	61.2	83.0	26.2	24.8
PEFT LoRA	74.5	74.2	71.6	67.8	72.8	95.6	30.4	29.0
Residual SAA	56.2	66.2	42.4	59.2	67.2	90.4	23.3	20.2
SAA-weights	**76.9**	**76.6**	**75.4**	**70.6**	**77.4**	**96.0**	**35.7**	**34.8**

prompt; we report strict and flexible answer-extraction accuracy. MATH and MATH-500 use mathematical answer verification. For the MetaMathQA holdout, we report strict and flexible extraction accuracy, verified accuracy, and boxed-output compliance (the presence of a boxed answer, regardless of correctness). Bold denotes the best fine-tuned result within each table. The held-out loss curves in Figure 1 complement these results: SAA-weights attains the lowest loss late in training among the four configurations.

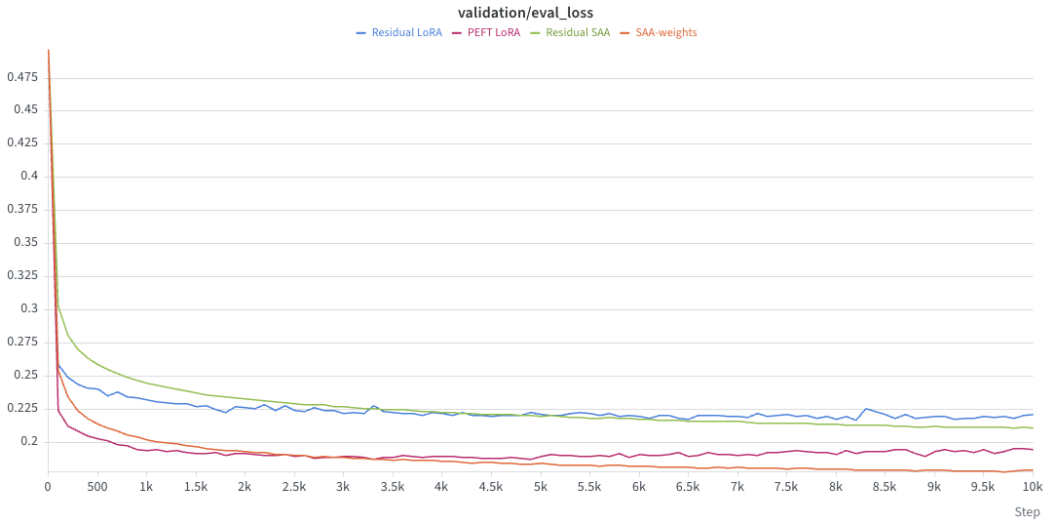

Figure 1: Held-out teacher-forced loss on MetaMathQA-40K for Ministral-3B adapters with a 3M parameter budget and canonical boxed targets.

4.3 SUPERVISED ADAPTATION BEYOND MATHEMATICS

We additionally evaluate SAA on data-to-text generation by fine-tuning GPT-2 Medium on the E2E benchmark. Both SAA-weights and PEFT LoRA adapt the fused attention projections across all 24 transformer layers, with exactly 393,216 trainable parameters. We evaluate generated descriptions using the official E2E metrics: BLEU, NIST, METEOR, ROUGE-L, and CIDEr.

Table 3 presents the configurations with the highest observed test BLEU for each method at this budget. Both use Xavier initialization; SAA uses output scale 8, while LoRA uses rank 4 and $\alpha = 32$. These configurations are drawn from an exploratory initialization and scaling sweep, whose full results are reported in the appendix. SAA-weights achieves performance close to PEFT LoRA across all five metrics, with slightly higher BLEU, ROUGE-L, and CIDEr, and slightly lower NIST and METEOR. The differences are small and do not establish a consistent advantage from these single-seed experiments. Figure 2 shows the corresponding held-out loss curves, allowing supervised fitting to be compared with generation quality. One possible contributor to the different behavior across backbones is the geometry of the adapted projections. GPT-2 Medium uses a fused projection from 1,024 to 3,072 dimensions, whereas our Ministral-3B experiments adapt query and value projections from 3,072 to 4,096 and 1,024 dimensions, respectively. The larger input dimension may provide a setting in which the structured parameterization is more effective. However, backbone, task, and placement also differ between these experiments; a controlled matrix-size ablation is required to test this explanation.

Table 3: GPT-2 Medium on E2E with a 0.39M parameter budget.

Method	BLEU	NIST	METEOR	ROUGE-L	CIDEr
Base	4.57	1.037	12.63	18.62	0.000
PEFT LoRA	66.60	**8.631**	**45.39**	68.76	2.272
SAA-weights	**66.77**	8.548	45.11	**69.00**	**2.282**

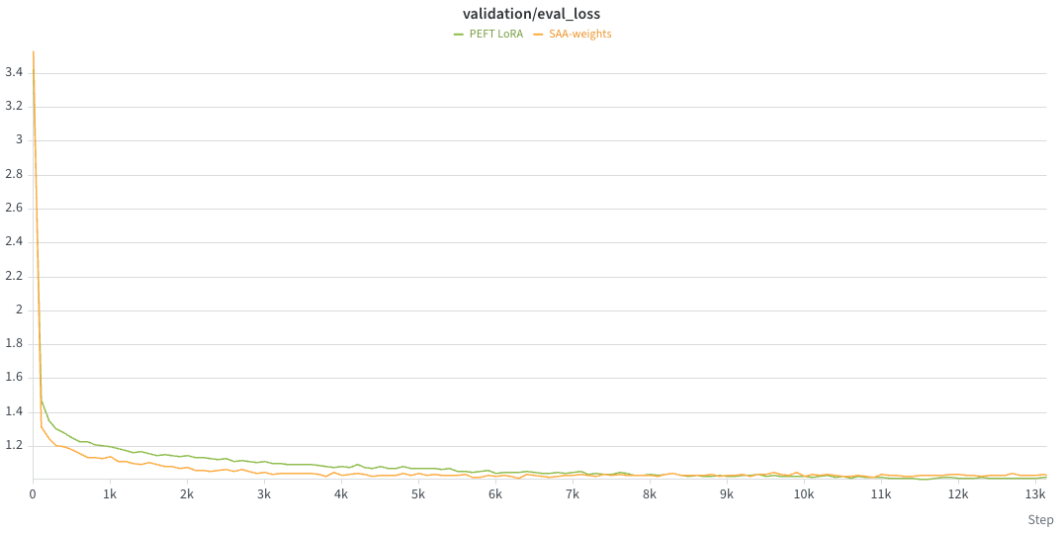

Figure 2: Held-out loss for the GPT-2 Medium configurations reported in Table 3.

4.4 REINFORCEMENT LEARNING ON MATHEMATICAL TASKS

We compare parameter-matched SAA and LoRA adapters on Qwen3-0.6B Base using GRPO-style optimization on the same Math10K training split, with identical rewards, rollout settings, and optimization hyperparameters. The pretrained backbone remains frozen.

Table 4: Qwen3-0.6B accuracy after RL on Math10K.

Method	Math10K	GSM8K	MATH-500	SVAMP
Frozen backbone	75.44	51.78	30.40	70.80
Residual LoRA	76.39	52.84	31.00	66.20
Residual SAA	**79.22**	**54.59**	**32.40**	**72.90**
SAA − LoRA (pp)	+2.83	+1.75	+1.40	+6.70

Accuracy is reported in percentages. Math10K uses the 1,800-example held-out split; GSM8K, MATH-500, and SVAMP contain 1,319, 500, and 1,000 evaluation examples, respectively.

SAA outperforms both parameter-matched LoRA and the frozen backbone on all four evaluations (Table 4). Its gains over LoRA range from 1.40 percentage points on MATH-500 to 6.70 points on SVAMP. On SVAMP, LoRA reduces accuracy from the frozen backbone's 70.80% to 66.20%, whereas SAA improves it to 72.90%.

RL-trained SAA outperforms both LoRA and the frozen backbone on Math10K, GSM8K, MATH-500, and SVAMP. These gains in answer correctness complement the SFT results, where improvements in format compliance did not consistently improve accuracy over the frozen backbone. Given limited GPU resources, we evaluated SFT and RL in separate experimental settings rather than conducting a matched comparison of the two objectives.

5 Conclusions

We introduced Structured Algebraic Adapters (SAA), which adapt frozen language models through trainable coefficients over a fixed family of algebraically structured operators. This parameterization supports residual-state and weight-space adaptation without imposing an explicit low-rank constraint.

Our experiments compare SAA with LoRA at approximately matched trainable-parameter budgets under supervised fine-tuning and reinforcement learning. On Ministral-3B, weight-space SAA outperforms the other fine-tuned configurations on all reported metrics in the canonical-boxed 3M comparison. However, neither SAA nor LoRA improves MATH or MATH-500 accuracy over the frozen backbone in this setting. These results distinguish improved output-format compliance from gains in answer correctness. On GPT-2 Medium, the selected SAA and LoRA configurations achieve comparable E2E generation performance.

Under reinforcement learning on Qwen3-0.6B, SAA achieves higher answer accuracy than both LoRA and the frozen backbone on all four evaluations. Its gains over LoRA are 2.83 percentage points on the Math10K evaluation split and 1.40–6.70 points on GSM8K, MATH-500, and SVAMP.

The comparisons use single training seeds, and the E2E configuration comparison is exploratory. Additional seeds and basis-controlled comparisons would help establish the robustness of the results and isolate the contribution of the specific algebraic operator family. Cross-dataset overlap with the RL training data remains to be assessed.

Together, these results support structured operator parameterizations as an alternative to low-rank factorization, with performance depending on task, adapter placement, and training objective.

AI Use Statement

Generative AI assistants, including ChatGPT (OpenAI) and Claude (Anthropic), were used during manuscript preparation for language editing, proofreading, LaTeX formatting, and checks of textual consistency and arithmetic. All AI-assisted text and suggested edits were reviewed by the authors. The authors take full responsibility for the scientific content, reported results, interpretations, and final form of the manuscript.

Reproducibility Statement

The adapter construction and optimization objectives are described in Sections 3 and 3.3. Models, datasets, adapter placements, parameter budgets, training settings, and evaluation procedures are reported in Section 4.1 and in the corresponding experimental subsections.

References

Kartikeya Bhardwaj, Nilesh Prasad Pandey, Sweta Priyadarshi, Viswanath Ganapathy, Shreya Kadambi, Rafael Esteves, Shubhankar Borse, Paul Whatmough, Risheek Garrepalli, Mart Van Baalen, Harris Teague, and Markus Nagel. Sparse high rank adapters. *arXiv preprint arXiv:2406.13175*, 2024. URL https://arxiv.org/abs/2406.13175.

Zhuo Chen, Rumen Dangovski, Charlotte Loh, Owen Dugan, Di Luo, and Marin Soljačić. QuanTA: Efficient high-rank fine-tuning of llms with quantum-informed tensor adaptation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 92210–92245, 2024. doi: 10.52202/079017-2928. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/a7c17115db36193f6b83b71b0fe1d416-Abstract-Conference.html.

Ziqi Gao, Qichao Wang, Aochuan Chen, Zijing Liu, Bingzhe Wu, Liang Chen, and Jia Li. Parameter-efficient fine-tuning with discrete Fourier transform. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 14884–14901. PMLR, 2024. URL https://proceedings.mlr.press/v235/gao24o.html.

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*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Qiushi Huang, Tom Ko, Zhan Zhuang, Lilian Tang, and Yu Zhang. HiRA: Parameter-efficient hadamard high-rank adaptation for large language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/48c368f105e8145b945227b73255635a-Abstract-Conference.html.

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.

Toshiaki Koike-Akino, Francesco Tonin, Yongtao Wu, Frank Zhengqing Wu, Leyla Naz Candogan, and Volkan Cevher. Quantum-PEFT: Ultra parameter-efficient fine-tuning. *arXiv preprint arXiv:2503.05431*, 2025. URL https://arxiv.org/abs/2503.05431.

Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M. Asano. VeRA: Vector-based random matrix adaptation. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/1b53ad08de383a049e9668a9d0b6a053-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*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32100–32121. PMLR, 2024a. URL https://proceedings.mlr.press/v235/liu24bn.html.

Weiyang Liu, Zeju Qiu, Yao Feng, Yuliang Xiu, Yuxuan Xue, Longhui Yu, Haiwen Feng, Zhen Liu, Juyeon Heo, Songyou Peng, Yandong Wen, Michael J. Black, Adrian Weller, and David Ha. Parameter-efficient orthogonal finetuning via butterfly factorization. In *International Conference on Learning Representations*, 2024b.

URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/a63ce8e6867a1bf4b4ca62e5077814d9-Abstract-Conference.html.

Snehal Raj and Brian Coyle. QuIC: Quantum-inspired compound adapters for parameter efficient fine-tuning. *arXiv preprint arXiv:2502.06916*, 2025. URL https://arxiv.org/abs/2502.06916.

Fei Wu, Jia Hu, Geyong Min, and Shiqiang Wang. Efficient orthogonal fine-tuning with principal subspace adaptation. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/2b5af479527167d4af78847a9b9b645f-Abstract-Conference.html.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. ReFT: Representation finetuning for language models. *arXiv preprint arXiv:2404.03592*, 2024. URL https://arxiv.org/abs/2404.03592.