MULTI-QUERY ASSOCIATIVE CORRECTION: ENRICHING THE READOUT INTERFACE FOR HISTORY-CONDITIONED LOW-RANK ADAPTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Standard LoRA learns a fixed low-rank correction map, while results from explicitly partitioned adapters suggest that how a fixed adaptation budget is organized can matter. We ask whether such differentiation can instead be learned from a common training stream, without predefined subset identities. We study this through a readout-centric view of low-rank adaptation and identify an *allocation–expression gap*: allocating multiple or wider readouts does not by itself produce differentiated corrections. Such differentiation requires data-dependent variation to occupy the exposed coordinates and remain distinct after decoding. This analysis highlights the joint importance of data-dependent evidence exposure and differentiated decoding. Under a compact recurrent-state design, these motivate multiple conditioned reads from a shared state and group-specific decoding, while previous-state retrieval additionally motivates a direct path for current evidence. MQ-ACM instantiates this design with one shared associative state, while joint task-gradient initialization provides a coordinated starting geometry for the paired input–output coordinates. On our main Llama-3.1-8B generation setting, the default $r = 8$, $K = 4$ configuration improves rank-8 LoRA by 0.50 on MT-Bench, 5.96 points on GSM8K, and 6.30 points on HumanEval. A complementary commonsense evaluation under a distinct adaptation setting provides additional validation. Effective-rank and alignment diagnostics characterize realized correction structure; interventions support functional use of associative and query-specific decoded terms. These results support readout organization as a distinct design dimension for history-conditioned low-rank adaptation, with additional computation despite fixed state storage.

1 INTRODUCTION

From prescribed splits to learned organization. Low-Rank Adaptation (LoRA) learns a compact correction map for a frozen backbone (Hu et al., 2022). After training, the map is fixed, although its output varies with the input. Figure 1(a) reproduces a split-LoRA experiment from HydraLoRA (Tian et al., 2024): assigning data subsets to separate components improves MMLU over a monolithic adapter at the same total rank $r \times n = 32$ and parameter budget. This suggests that organization can matter, but leaves open how to learn it. We ask whether an adapter can learn differentiated corrections from a common training stream, without predefined subset identities, while adapting their readout to the current input and sequence context.

When learned organization becomes expressed correction. Multiple components or a wider readout interface do not by themselves establish useful organization: allocated coordinates must be occupied by data-dependent variation, and those variations must remain distinct after decoding. Our readout-centric lens characterizes this *allocation–expression gap* by distinguishing local width r, aggregate width R, and effective correction rank r_{eff} (Figure 1(b,c)). For multiple reads, query-specific output effects couple read deviations with decoder deviations; differences on both sides are necessary but can still cancel. Characterizing when differentiation survives to the output therefore necessitates joint analysis of data-dependent readout and differentiated decoding. The analysis characterizes when allocated coordinates can become distinct output effects; whether those effects are useful for a task remains an empirical question.

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.

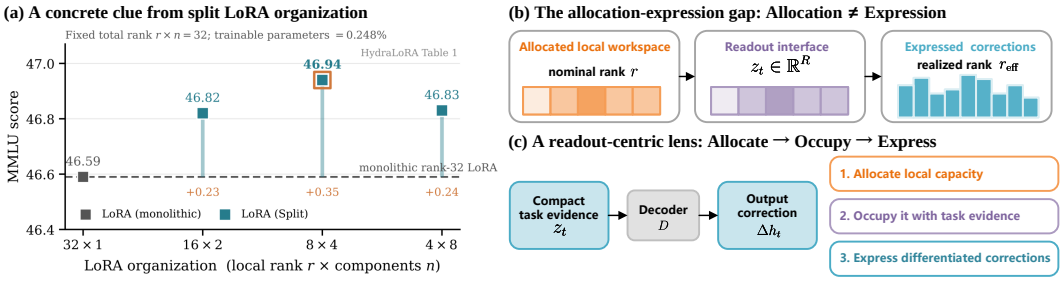


Figure 1: The organization problem. (a) Subset-split LoRA results from Tian et al. (2024) at fixed total rank and parameter budget. (b,c) Allocated coordinates, data-dependent readout variation ("occupy"), and decoded expression.

From analysis to recurrent interface design. A compact evolving state offers one way to make organization sequence-dependent: preceding tokens shape the state, while current-input-conditioned reads determine which stored evidence enters the correction. Under this compact-storage choice, data-dependent readout motivates **(1) multiple conditioned reads without replicating the state**, while read–decoder coupling motivates **(2) group-specific decoding** that allows their differences to survive. Because retrieval uses the previous state, **(3) a state-independent route for current evidence** keeps current evidence available even when the previous state is empty. The first expands access under the storage constraint; the second supports the expression of readout differentiation after decoding; the third complements history-based retrieval with direct current evidence. Together, these choices connect dynamic organization to the correction that can actually be expressed.

MQ-ACM as an instance. Following δ-mem's correction route on a frozen full-attention backbone (Lei et al., 2026), **MQ-ACM**—a **M**ulti-**Q**uery **A**ssociative **C**orrection **M**echanism—writes one shared state, queries it multiple times, and decodes each read through a group-specific map. A direct path supplies current evidence alongside retrieval. Training learns the write, query, direct, and decoder maps jointly from the same examples. At inference, these maps remain fixed while their reads and corrections depend on the current input and evolving state, so training learns organization rules whose application varies with sequence context. Increasing K at fixed r expands the interface to $R = Kr$ with added parameters and computation, preserving the $r \times r$ state and write width.

To support optimization of this paired interface, we build on gradient-guided low-rank initialization (Wang et al., 2024) and jointly initialize the shared direct-input map and projection-specific decoders. Subsequent training determines their use and task benefit.

Contributions and evidence. We make three connected contributions: (1) an analysis of how data-dependent readouts and decoding turn allocated coordinates into expressed correction; (2) MQ-ACM, which realizes history-conditioned organization through multiple reads of a shared evolving state, differentiated decoding, and direct access to current evidence; and (3) experiments that test task value, realized correction structure, functional use, component dependence, and cost. Generation and commonsense evaluations assess task value; checkpoint diagnostics characterize correction structure, interventions test functional use, and retrained ablations and timing measurements examine component dependence and cost. On Llama-family 8B models, the results show that readout organization can improve low-rank adaptation while preserving compact state storage, with task-dependent preferences over the expanded interface and additional computation.

2 Related Work

Low-rank capacity and organization. LoRA learns $\Delta W = BA$ (Hu et al., 2022). Variants reallocate or compose rank (Zhang et al., 2023; Ye et al., 2026), improve optimization/update geometry (Kalajdzievski, 2023; Hayou et al., 2024; Liu et al., 2024; Wang et al., 2025), or share parameters (Li et al., 2026). MELoRA composes mini-adapters; HydraLoRA learns routed output experts with a shared input factor (Ren et al., 2024; Tian et al., 2024). Its subset-split experiment motivates Figure 1(a). MQ-ACM studies differentiation through reads of evolving shared memory. Static parallel branches alone equal a wider factorization (§ A.9).

Initialization and task-gradient guidance. LoRA-GA shows that initialization can improve low-rank learning by aligning initial update gradients (Wang et al., 2024). GoRA guides allocation and initialization, while RaLoRA uses gradient intrinsic dimensionality (He et al., 2025; Ye et al., 2026). MQ-ACM uses joint task gradients to pair a shared input basis with projection-specific decoder coordinates. This supports optimization of the readout interface; the spectrum selects coordinates, and a separate scale sets the initial correction magnitude.

Recurrent state and correction interfaces. Memory-augmented LMs retain history in external or recurrent stores (Wu et al., 2022; Wang et al., 2023); linear attention uses fixed-size states (Katharopoulos et al., 2020), and delta-rule variants modify writing/retention (Yang et al., 2024; 2025). Attention-linearization replaces or approximates softmax attention (Zhang et al., 2025). The closest structural precursor, δ-mem, adds compact online memory to a frozen full-attention backbone (Lei et al., 2026). MQ-ACM retains this route and studies how multiple reads and decoders use one state; task comparisons use LoRA-family adapters.

3 A Readout-Centric Lens on Low-Rank Correction

Overview. The organization problem asks which allocated coordinates vary with the data and which variations survive decoding. We analyze both to guide recurrent interface design (§ 4).

3.1 Allocated Width vs. Realized Variation

For a frozen projection W_h with input x_t, write

$$y_{t,h} = W_h x_t + \Delta h_t, \qquad \Delta h_t = s_R D_h z_t, \tag{1}$$

where $z_t \in \mathbb{R}^R$ is the exposed readout, $D_h \in \mathbb{R}^{d_h \times R}$ is the correction decoder, and $s_R > 0$ is a scalar. We call R the aggregate readout width. For standard LoRA, $z_t = Ax_t$ and R equals its rank hyperparameter. In MQ-ACM, r is the local state/read width and $R = Kr$ the aggregate readout width; Eq. (2) characterizes its recurrent history-conditioned correction at output level.

The factorization separates available evidence from its output effect. Individual latent coordinates are not identifiable: for any invertible $T \in \mathbb{R}^{R\times R}$, replacing z_t by $T^{-1}z_t$ and D_h by D_hT leaves Δh_t unchanged. We therefore analyze output-level variation. Under a specified evaluation distribution over tokens and histories, define

$$\Sigma_z = \mathrm{Cov}(z_t), \qquad \Sigma_{\Delta,h} = s_R^2 D_h \Sigma_z D_h^\top. \tag{2}$$

Let $r_{\mathrm{alg},h} = \mathrm{rank}(\Sigma_{\Delta,h})$ denote the algebraic dimension of correction variation. When $\mathrm{tr}(\Sigma_{\Delta,h}) > 0$, let $\lambda_1, \ldots, \lambda_m$ be its positive eigenvalues and $\pi_i = \lambda_i / \sum_j \lambda_j$. The effective correction rank is

$$r_{\mathrm{eff},h} = \exp\left(-\sum_{i=1}^{m} \pi_i \log \pi_i\right). \tag{3}$$

We set $r_{\mathrm{eff},h} = 0$ when $\Sigma_{\Delta,h} = 0$; otherwise it is generally non-integer and summarizes spectral spread across active correction directions.

Lemma 1 (Width bounds correction variation)**.** *For the interface in Eq. (1),*

$$0 \le r_{\mathrm{eff},h} \le r_{\mathrm{alg},h} \le \min\{\mathrm{rank}(D_h), \mathrm{rank}(\Sigma_z), R\}. \tag{4}$$

If $\Sigma_{\Delta,h} \neq 0$, then $r_{\mathrm{eff},h} \ge 1$.

This formalizes Figure 1(c): *allocate* refers to width R, *occupy* to data-dependent variation Σ_z, and *express* to decoded variation $\Sigma_{\Delta,h}$. Here, occupancy denotes the distribution-dependent variation of the exposed readout coordinates. The bound limits variation; task value is evaluated separately.

3.2 Why Multiple Reads Need Differentiated Decoding

Consider the associative contribution of K local reads to projection h:

$$\Delta h_t^{\mathrm{assoc}} = s_R \sum_{k=1}^{K} D_{h,k} S_{t-1} q_{t,k}^m, \tag{5}$$

where $S_{t-1} \in \mathbb{R}^{r\times r}$ is a shared state and $D_{h,k} \in \mathbb{R}^{d_h\times r}$ is the decoder block for group k. Multiple queries view the same state, but distinct reads need not yield distinct output directions.

Proposition 1 (Attainable multi-query correction span)**.** *For fixed S, define $M_h(S) = [D_{h,1}S, \dots, D_{h,K}S]$. If the queries vary freely, attainable corrections form* $\mathrm{Col}(M_h(S))$*, with*

$$\mathrm{rank}(M_h(S)) \le \min\{d_h, K\,\mathrm{rank}(S)\} \le \min\{d_h, Kr\}. \tag{6}$$

If $D_{h,1} = \cdots = D_{h,K} = D_h^\star$, where $D_h^\star \in \mathbb{R}^{d_h\times r}$ is the shared block, the span reduces to $\mathrm{Col}(D_h^\star S)$ *and has dimension at most r.*

Separate decoders can expand the attainable output span of one $r \times r$ state (Proposition 1). Query generation, joint normalization, and data distribution determine the realized subset (§ A.4).

To identify what produces a query-specific output, let $a_{t,k} = S_{t-1}q_{t,k}^m$ and define

$$\bar{a}_t = \frac{1}{K}\sum_{k=1}^{K} a_{t,k}, \quad \epsilon_{t,k} = a_{t,k} - \bar{a}_t, \quad \bar{D}_h = \frac{1}{K}\sum_{k=1}^{K} D_{h,k}. \tag{7}$$

Because $\sum_k \epsilon_{t,k} = 0$,

$$\sum_{k=1}^{K} D_{h,k}a_{t,k} = K\bar{D}_h\bar{a}_t + \sum_{k=1}^{K}(D_{h,k} - \bar{D}_h)\epsilon_{t,k}. \tag{8}$$

The first term maps common evidence; the second couples read deviations with decoder deviations. Its norm is bounded by the product of their dispersions. If either side is identical across groups, the differential term vanishes; differences on both sides remain insufficient if their contributions cancel. This identifies the output term whose functional use we test in § 5.3.

Attention-conditioned visibility. Projection-space variation must also pass through the backbone computation. For one attention head of dimension d_a, hold keys $K_{\le t}$ fixed and let $p_t = \mathrm{softmax}(K_{\le t}^\top q_t/\sqrt{d_a})$, with D_q restricted to this head. For a sufficiently small query correction $\delta q_t = s_R D_q z_t$,

$$\delta p_t = \frac{s_R}{\sqrt{d_a}} J(p_t) K_{\le t}^\top D_q z_t + \mathcal{O}\big(\|\delta q_t\|_2^2\big)\,,$$

where $J(p) = \mathrm{Diag}(p) - pp^\top$ and $J(p)\mathbf{1} = 0$. Thus R bounds the dimension of local first-order probability variation; attention geometry and correction scale determine its magnitude. This fixed-key query result complements empirical Q/K/V/O correction tests at model level (§ A.6).

Design implications. Under a compact storage constraint, multiple reads can expand access to one state; Eq. (8) requires differentiated decoding for a query-specific output term. Previous-state retrieval additionally motivates a direct route for current evidence. These requirements guide the architecture. Joint initialization aids paired coordinate learning; the algebra does not require it.

4 Multi-Query Associative Correction

Method overview. Figure 2 implements *store once, query many, and decode separately*. The shared state and group-specific decoders form the core interface; a direct path supplies current evidence, while joint initialization and query regularization support learning. We assume causal token order, additive attention-projection corrections, and a persistent per-layer state.

4.1 One State, Multiple Query-Conditioned Reads

Each selected attention block conditions reads on a shared history state $S_t \in \mathbb{R}^{r\times r}$. From its input $x_t \in \mathbb{R}^d$, it forms K read queries and one shared write pair:

$$\begin{aligned}
\widetilde{Q}_t^m &= \mathrm{Norm}\big(\tanh(W_q^m x_t)\big)\,,\\
Q_t^m &= \mathrm{reshape}(\widetilde{Q}_t^m) = \{q_{t,k}^m\}_{k=1}^K,\\
k_t^m &= \mathrm{Norm}(\tanh(W_k^m x_t))\,, \qquad v_t^m = W_v^m x_t,\\
\beta_t &= \sigma(w_\beta^\top x_t + b_\beta) \in (0,1), \qquad \lambda_t = 1 - \beta_t,
\end{aligned} \tag{9}$$

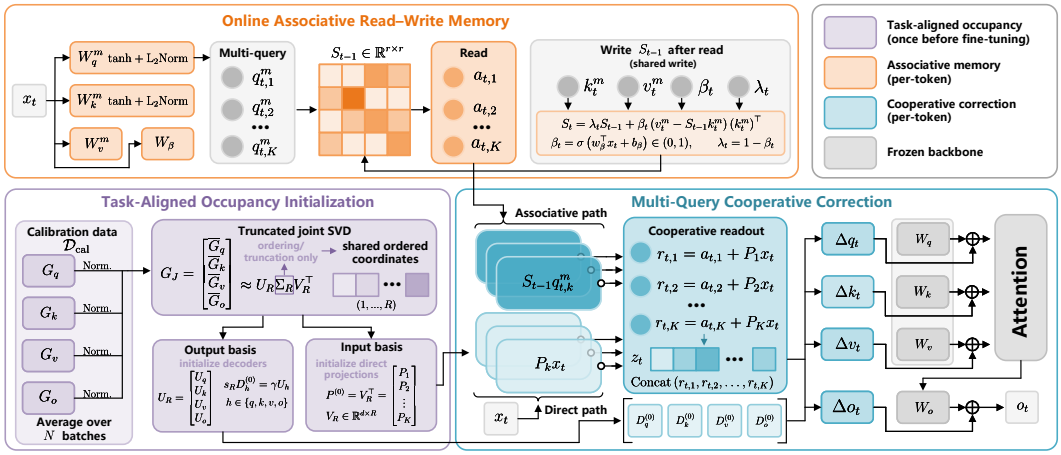


Figure 2: MQ-ACM: shared-state reads and direct evidence feed group-specific decoders. Initialization supplies paired coordinates; training determines their occupancy.

where $\mathrm{Norm}(u) = u/\max(\|u\|_2, \epsilon)$ acts on the full Kr-dimensional query vector before reshaping. Queries learn to address shared memory without predefined subset labels. Reading precedes writing:

$$a_{t,k} = S_{t-1} q_{t,k}^m, \qquad k = 1, \ldots, K, \tag{10}$$

The shared input-conditioned scalar-gated delta update is

$$S_t = \lambda_t S_{t-1} + \beta_t \left(v_t^m - S_{t-1} k_t^m\right) \left(k_t^m\right)^\top. \tag{11}$$

The residual writes value-prediction error; reading the old state prevents same-position retrieval. At fixed r, increasing K preserves the $r \times r$ state and r-wide write path while expanding the query/direct/decoder interface to $R = Kr$. Local width, aggregate width, and parameter count therefore define distinct comparisons.

4.2 Cooperative Readout and Differentiated Decoding

Although queries depend on the current input, retrieval is mediated by the old state: when $S_0 = 0$, the first associative read is zero. The direct path provides current evidence independently of stored history, complementing the associative route. Let $P \in \mathbb{R}^{R \times d}$ be the shared direct-input map, partitioned as $P = [P_1^\top, \ldots, P_K^\top]^\top$, with $P_k \in \mathbb{R}^{r \times d}$. Each group combines both sources:

$$\begin{aligned} r_{t,k} &= a_{t,k} + P_k x_t, \\ z_t &= \mathrm{Concat}(r_{t,1}, \ldots, r_{t,K}) \in \mathbb{R}^R, \qquad R = Kr. \end{aligned} \tag{12}$$

For each selected projection h, the decoder $D_h \in \mathbb{R}^{d_h \times R}$ produces

$$\Delta h_t = s_R D_h z_t, \qquad s_R = \frac{\alpha}{\sqrt{R}}, \qquad D_h = [D_{h,1}, \ldots, D_{h,K}]. \tag{13}$$

The generation setting uses $h \in \{q, k, v, o\}$; commonsense uses $h \in \{q, k, v\}$. Each $D_{h,k}$ maps into the full target space rather than a disjoint output slice. Expanding the correction gives

$$\Delta h_t = \underbrace{s_R \sum_{k=1}^{K} D_{h,k} a_{t,k}}_{\Delta h_t^{\text{assoc}}} + \underbrace{s_R \sum_{k=1}^{K} D_{h,k} P_k x_t}_{\Delta h_t^{\text{dir}}}. \tag{14}$$

Both paths train through their sum, with group-specific decoding enabling Eq. (8). The direct term is a fixed map $s_R D_h P$ of rank at most R. With module input x_t fixed, different histories can change the associative correction through S_{t-1}; this term generally cannot be folded into one fixed matrix across histories (§ A.8). Here x_t is the correction module's block input; Q/K/V and O share channel indexing for coordinate construction while retaining projection-specific backbone activations.

4.3 Task-Aligned Coordinate Initialization

To support optimization of the paired interface (Wang et al., 2024), we initialize the shared P and projection decoders D_h jointly. Using 32 training-only calibration batches with corrections disabled and backbone weights fixed, we accumulate:

$$G_h = \frac{1}{N}\sum_{i=1}^{N} \nabla_{W_h} \mathcal{L}_i, \qquad h \in \{q, k, v, o\}. \tag{15}$$

The following equations show Q/K/V/O; the commonsense setting omits O. To avoid domination by a high-norm branch, set $\bar{G}_h = G_h / \max(\|G_h\|_F, \epsilon)$ and compute a rank-R truncated joint SVD:

$$\begin{aligned} G_J &= [\bar{G}_q^\top, \bar{G}_k^\top, \bar{G}_v^\top, \bar{G}_o^\top]^\top \approx U_R \Sigma_R V_R^\top, \\ U_R &= [U_q^\top, U_k^\top, U_v^\top, U_o^\top]^\top. \end{aligned} \tag{16}$$

We initialize the previously defined maps as

$$P^{(0)} = V_R^\top, \qquad s_R D_h^{(0)} = \gamma U_h. \tag{17}$$

Here $\gamma = 10^{-3}$ sets correction scale; the spectrum orders and truncates coordinates. Joint SVD pairs P's shared input basis with projection-specific left blocks for D_h, giving task-related starting geometry. Fine-tuning determines coordinate use and task benefit (O-projection convention: § A.7).

4.4 Learning and State Evolution

The memory projections, scalar gate, direct map, and decoders learn jointly from the same examples. The evolving state supplies sequence-specific content. As an auxiliary training regularizer, we discourage collinear query directions with

$$\mathcal{L}_{\text{div}} = \mathbb{E}_t \left[\frac{1}{K(K-1)} \sum_{i \neq j} \cos^2(q_{t,i}^m, q_{t,j}^m) \right], \tag{18}$$

with $\mathcal{L}_{\text{div}} = 0$ for $K = 1$. Cosine uses denominator $\max(\|u\|_2, \epsilon_{\cos}) \max(\|v\|_2, \epsilon_{\cos})$, with $\epsilon_{\cos} > 0$. This auxiliary term acts on query directions; read- and output-level differentiation are evaluated separately by the diagnostics in § 5.3. The objective is

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_{\text{div}} \mathcal{L}_{\text{div}}. \tag{19}$$

with $\lambda_{\text{div}} = 10^{-4}$ in all experiments.

At inference, fixed parameters produce reads from the input and evolving state S_t. States reset to zero for each training forward call and independent evaluation example, and persist from prefill through generation. The per-layer state augments the KV cache (algorithms and lifecycle: § B).

5 Experiments

We evaluate three questions: does learned readout organization have task value, which decoded terms do trained models use, and how do component choices and computation affect the result?

5.1 Experimental Setup

Tasks and protocols. We fine-tune Llama-3.1-8B-Base (Grattafiori et al., 2024) on 100K MetaMathQA (Yu et al., 2024), 100K code-only Code-Feedback (Zheng et al., 2024), or 52K WizardLM examples (Xu et al., 2024), and evaluate GSM8K accuracy (Cobbe et al., 2021), HumanEval pass@1 (Chen et al., 2021), or GPT-4o first-turn MT-Bench (Zheng et al., 2023). Following GoRA (He et al., 2025), we train Q/K/V/O for one epoch at default learning rate 5×10^{-5} (overrides: § C.3).

For commonsense reasoning, we follow LLM-Adapters (Hu et al., 2023) on Llama-3-8B and evaluate BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2020), ARC-C/E (Clark et al., 2018), and OpenBookQA

Table 1: Llama-3.1-8B results: baseline rank 8 (reference rank for adaptive methods); MQ-ACM local width 8. Bold/underline: best/second-best low-rank results.

Method	MT-Bench	GSM8K	HumanEval
Full fine-tuning	5.95 ± 0.08	73.69 ± 0.28	51.63 ± 1.27
LoRA (Hu et al., 2022)	6.08 ± 0.04	67.78 ± 1.25	43.09 ± 0.35
rsLoRA (Kalajdzievski, 2023)	6.08 ± 0.09	68.36 ± 0.74	45.78 ± 2.80
DoRA (Liu et al., 2024)	6.24 ± 0.12	69.17 ± 1.00	43.70 ± 1.54
LoRA+ (Hayou et al., 2024)	6.38 ± 0.10	71.29 ± 0.93	44.51 ± 2.11
OLoRA (Büyükakyüz, 2024)	6.13 ± 0.04	68.54 ± 0.42	43.29 ± 2.44
PiSSA (Meng et al., 2024)	5.76 ± 0.23	68.56 ± 1.03	44.10 ± 1.54
LoRA-GA (Wang et al., 2024)	6.00 ± 0.06	71.39 ± 0.90	43.29 ± 0.61
AdaLoRA (Zhang et al., 2023)	6.13 ± 0.12	70.63 ± 0.77	41.46 ± 3.66
MELoRA (Ren et al., 2024)	6.11 ± 0.07	71.27 ± 1.10	45.73 ± 2.80
MoRA (Jiang et al., 2024)	6.32 ± 0.12	68.12 ± 0.91	29.68 ± 5.73
GoRA (He et al., 2025)	6.25 ± 0.05	72.91 ± 0.76	$\underline{48.98 \pm 2.14}$
RaLoRA (Ye et al., 2026)	6.39 ± 0.08	72.25 ± 0.59	48.78 ± 1.61
MQ-ACM ($K = 1$)	$\mathbf{6.75 \pm 0.07}$	73.24 ± 1.53	44.31 ± 2.88
MQ-ACM ($K = 2$)	6.28 ± 0.14	73.21 ± 0.62	48.37 ± 1.41
MQ-ACM ($K = 4$)	$\underline{6.58 \pm 0.05}$	$\underline{73.74 \pm 0.34}$	$\mathbf{49.39 \pm 1.22}$
MQ-ACM ($K = 8$)	6.39 ± 0.14	$\mathbf{74.05 \pm 0.82}$	46.95 ± 0.61

Table 2: Commonsense accuracy on Llama-3-8B; MQ-ACM: $K = 4, r = 8$ ($R = 32$). Task columns: three-seed means; deviations: Table E.1. HellaS: HellaSwag; WinoG: WinoGrande.

Method	Params	BoolQ	PIQA	SIQA	HellaS	WinoG	ARC-C	ARC-E	OBQA	Average
LoRA ($r = 8$)	4.72M	71.58	86.38	78.27	<u>93.11</u>	82.72	77.84	89.77	81.80	82.68 ± 0.09
LoRA ($r = 32$)	18.87M	**73.49**	<u>87.83</u>	<u>79.63</u>	92.71	<u>85.32</u>	**78.61**	**90.10**	<u>84.60</u>	84.04 ± 0.45
MQ-ACM	16.91M	<u>73.03</u>	**87.96**	**80.14**	**94.77**	**86.85**	<u>78.58</u>	<u>89.94</u>	**84.87**	$\mathbf{84.52 \pm 0.18}$

(Mihaylov et al., 2018). This separate setting uses one epoch, learning rate 10^{-4}, 100 warm-up steps, and Q/K/V targets. § C gives full configurations.

Scope, questions, and provenance. The formulation targets causal decoder-only Transformers; experiments cover only Llama-family 8B models. Table 1 asks what richer readout provides at fixed $r = 8$; Table 2 compares $R = 32$ MQ-ACM with static rank-32 LoRA in performance and parameters; Figure 3(a) controls $R = 32$ while varying (K, r). Protocols match within each benchmark. GSM8K/HumanEval baselines follow GoRA/RaLoRA (He et al., 2025; Ye et al., 2026); we evaluate MT-Bench baselines and all MQ-ACM results. Results are three-seed mean $\pm$ sample std.

5.2 Main Results

Fixed local width, richer readout interface. Table 1 keeps $r = 8$ while varying learned reads. Default $K = 4$ improves LoRA by 0.50, 5.96, and 6.30 on MT-Bench, GSM8K, and HumanEval. The best MQ-ACM means occur at $K = 1, 8, 4$, respectively: larger K exposes more coordinates but does not guarantee a gain. Full fine-tuning remains higher on HumanEval.

Aggregate width versus rank-32 LoRA. At $R = 32$, MQ-ACM averages 84.52 with 16.91M parameters versus 84.04 and 18.87M for rank-32 LoRA (Table 2). MQ-ACM's shared write path yields a different parameterization (§ B.7); rank-8 LoRA is the fixed-local-width reference.

5.3 Mechanistic Diagnostics

We separate expressed variation from functional use. Figure 3 reports checkpoint statistics; Figure 4 intervenes on the analyzed associative and differential terms to test their use in trained models.

Diagnostic definitions. For a nonzero held-out projection gradient G_h^{ho}, let $V = \mathrm{orth}(P^\top)$ and $U_h = \mathrm{orth}(D_h)$. Their input, output, and bilateral gradient coverage is

$$C_{\mathrm{in}} = \frac{\|G_h^{\mathrm{ho}} V\|_F^2}{\|G_h^{\mathrm{ho}}\|_F^2}, \quad C_{\mathrm{out}} = \frac{\|U_h^\top G_h^{\mathrm{ho}}\|_F^2}{\|G_h^{\mathrm{ho}}\|_F^2}, \quad C_{\mathrm{bi}} = \frac{\|U_h^\top G_h^{\mathrm{ho}} V\|_F^2}{\|G_h^{\mathrm{ho}}\|_F^2}. \tag{20}$$

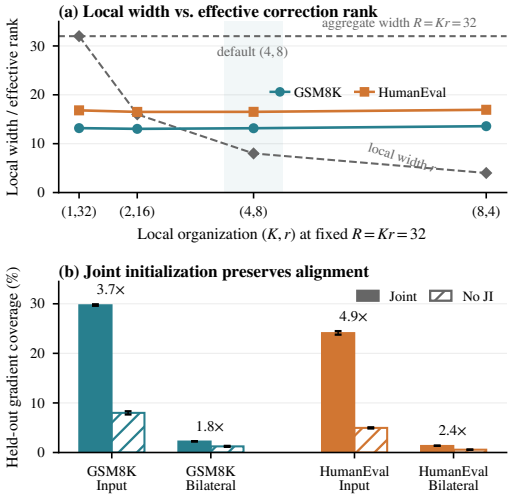


Figure 3: (a) Total-correction effective rank at $R = 32$; dashed: local r; shading: $(K, r) = (4, 8)$. (b) Gradient coverage, macro-averaged over layers and Q/K/V/O. Bars: three-seed std.

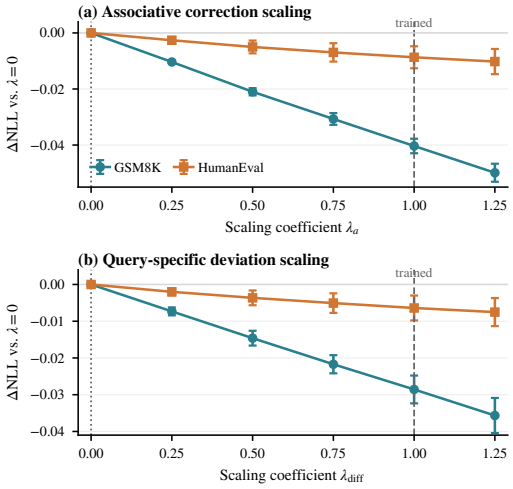


Figure 4: Scaling (a) associative and (b) query-specific decoded terms. ΔNLL is relative to removal ($\lambda = 0$); $\lambda = 1$ recovers the trained model. Bars: three-seed std.

Table 3: Retrained ablations ($K = 4, r = 8$). JI: joint initialization; DP: direct path.

Variant	JI	DP	MT-Bench	GSM8K	HumanEval
Full	✓	✓	6.58 ± 0.05	73.74 ± 0.34	$\mathbf{49.39 \pm 1.22}$
No DP	✓	–	6.59 ± 0.20	72.81 ± 0.94	46.34 ± 1.22
No JI	–	✓	6.53 ± 0.07	$\mathbf{74.60 \pm 1.52}$	44.92 ± 1.96
Neither	–	–	$\mathbf{6.62 \pm 0.18}$	72.53 ± 0.90	47.36 ± 1.86

Table 4: Batch-1 generation (Q/K/V/O, H100); prefill excluded. State: extra recurrent memory.

Method	Params	State (KiB)	Peak mem. (GiB)	Gen. ms/token
LoRA $r = 8$	6.82M	none	15.34	32.52 ± 4.32
MQ-ACM $K = 1$	6.95M	8	15.34	79.60 ± 1.11
MQ-ACM $K = 4$	21.10M	8	15.36	98.89 ± 1.09

Figure 3 macro-averages these quantities over layers and Q/K/V/O. Teacher-forced interventions fix parameters and reference tokens while recomputing activations and states (protocols: § D).

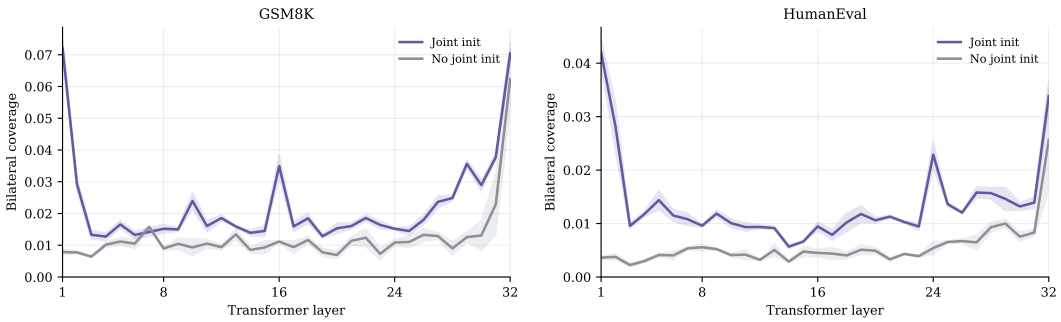
From allocated width to realized variation. Default $K = 4, r = 8$ yields total-correction $r_{\text{eff}} \approx 13.2/16.5$ on GSM8K/HumanEval, between local $r = 8$ and $R = 32$ (Figure 3(a)). At fixed R, effective rank and task scores vary differently across (K, r); changes in state width, storage, parameters, and diversity geometry make this an interface-factorization control (§ E.2).

Use of history-conditioned corrections. To test whether the state-mediated route contributes, Figure 4(a) scales the associative term while retaining the direct path. Every tested positive scale lowers teacher-forced NLL relative to $\lambda_a = 0$. Restoring $\lambda_a = 1$ raises GSM8K/HumanEval scores from $71.22/47.97$ to $73.74/49.39$; 1.25 further lowers NLL without improving task scores. This intervention measures functional use of the learned associative route within the evaluated sequence context.

Query-specific differences contribute beyond the common read. Eq. (8) isolates a decoded differential term. Figure 4(b) tests its use by substituting $\bar{a}_t + \lambda_{\text{diff}} \epsilon_{t,k}$ before decoding. Removal retains the direct and common associative terms; every tested positive scale lowers NLL. Absolute cosine falls from $0.914/0.934$ for raw reads to $0.085/0.065$ after decoding on GSM8K/HumanEval. These results support functional use of decoded differences without identifying semantic partitions.

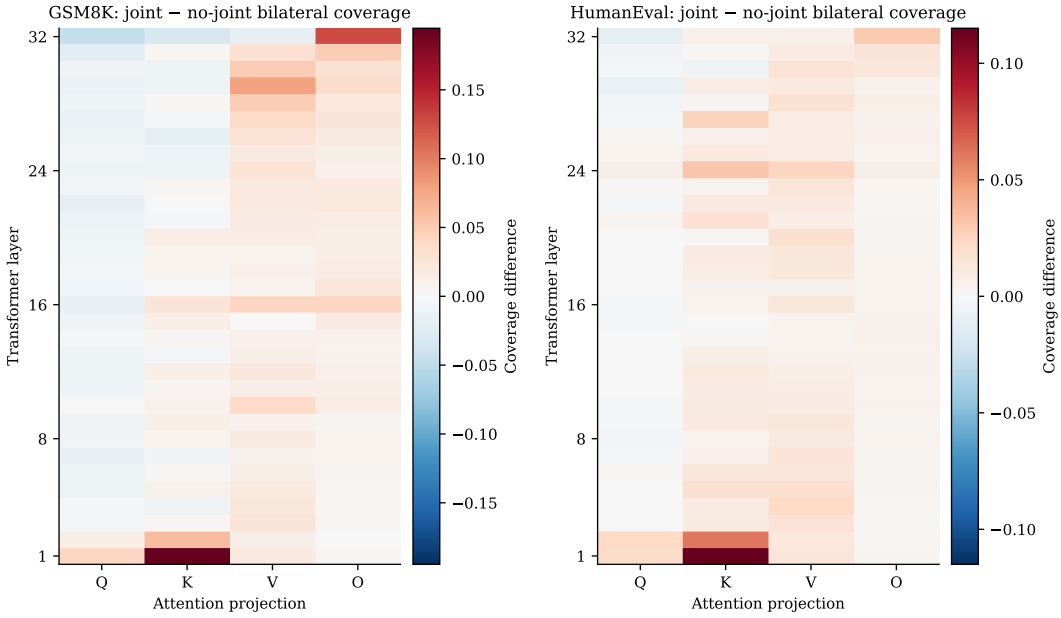
Alignment after joint initialization. To examine the paired starting geometry after training, we compare held-out gradient coverage. Joint initialization gives $3.7\times / 4.9\times$ higher input coverage and raises bilateral coverage from 1.24% to 2.26% and 0.57% to 1.36% on GSM8K/HumanEval, while output-only coverage decreases. This trained-checkpoint pattern is consistent with coordinated input–output geometry and complements Table 3.

Figures 5 and 6 locate this alignment signature across depth and projections: the bilateral-coverage gain persists through most layers and spans multiple projections, with some negative cells.



(a) GSM8K **(b)** HumanEval

Figure 5: Layer-wise bilateral gradient coverage, averaged over Q/K/V/O.



(a) GSM8K **(b)** HumanEval

Figure 6: Layer–projection differences in bilateral coverage, $C_{\text{bi}}^{\text{joint}} - C_{\text{bi}}^{\text{no joint}}$.

5.4 Retrained Ablations and Inference Cost

Component benefits depend on the task. Retraining favors Full on HumanEval, No JI on GSM8K, and Neither on MT-Bench (Table 3). Neither joint initialization nor the direct path benefits every task, despite functional use in Full. Stronger alignment also need not improve scores: Full has higher bilateral coverage but a lower GSM8K mean than No JI (Figure 3(b)).

Fixed storage, added computation. At $r = 8$, $K = 1$ and $K = 4$ each use an 8 KiB state over 32 layers. Increasing K from 1 to 4 raises parameters from 6.95M to 21.10M and measured H100 latency from 79.60 to 98.89 ms/token ($+24\%$); $K = 4$ remains $3.04\times$ slower than LoRA (Table 4).

6 Conclusion

On the evaluated Llama-family 8B models, MQ-ACM learns differentiated corrections from a common stream without predefined subset identities. Interventions support functional use of associative and query-specific terms. Data-dependent occupancy and decoding determine how allocated width becomes expressed variation. Query count and component benefits remain task-dependent, with additional computation at fixed state storage. Readout organization offers a design dimension whose task value requires empirical evaluation.

AI USE STATEMENT

ChatGPT was used to assist with manuscript organization, language revision, and checks of consistency between the main text and appendices. Its use did not generate new experimental results. The use of GPT-4o as the MT-Bench evaluator is described in §§ 5 and C.

REPRODUCIBILITY STATEMENT

§ A provides definitions, assumptions, and proofs for the theoretical claims. § B details initialization, the forward computation, training, inference, and state handling. § C records the datasets, training configurations, baseline settings, and computing environment. §§ D and E describe the diagnostic protocols and report additional results and controls.

REFERENCES

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 (AAAI)*, volume 34, pp. 7432–7439, 2020.

Kerim Büyükakyüz. OLoRA: Orthonormal Low-Rank Adaptation of Large Language Models. *arXiv preprint arXiv:2406.01775*, 2024.

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

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 (NAACL-HLT), Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

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.

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

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, 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, 2024.

Haonan He, Peng Ye, Yuchen Ren, Yuan Yuan, Luyang Zhou, Shucun Ju, and Lei Chen. GoRA: Gradient-driven Adaptive Low Rank Adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 114074–114102, 2025.

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.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5254–5276, 2023.

Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, and Deqing Wang. MoRA: High-Rank Updating for Parameter-Efficient Fine-Tuning. *arXiv preprint arXiv:2405.12130*, 2024.

Damjan Kalajdzievski. A Rank Stabilization Scaling Factor for Fine-Tuning with LoRA. *arXiv preprint arXiv:2312.03732*, 2023.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers Are RNNs: Fast Autoregressive Transformers with Linear Attention. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165, 2020.

Jingdi Lei, Di Zhang, Junxian Li, Weida Wang, Kaixuan Fan, Xiang Liu, Qihan Liu, Xiaoteng Ma, Baian Chen, and Soujanya Poria. δ-mem: Efficient Online Memory for Large Language Models. *arXiv preprint arXiv:2605.12357*, 2026.

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.

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, 2024.

Fanxu Meng, Zhaohui Wang, and Muhan Zhang. PiSSA: Principal Singular Values and Singular Vectors Adaptation of Large Language Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 121038–121072, 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.

Pengjie Ren, Chengshun Shi, Shiguang Wu, Mengqi Zhang, Zhaochun Ren, Maarten de Rijke, Zhumin Chen, and Jiahuan Pei. MELoRA: Mini-Ensemble Low-Rank Adapters for Parameter-Efficient Fine-Tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL), Volume 1: Long Papers*, pp. 3052–3064, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pp. 8732–8740, 2020.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense Reasoning about Social Interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473, 2019.

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, and Chengzhong Xu. HydraLoRA: An Asymmetric LoRA Architecture for Efficient Fine-Tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 9565–9584, 2024.

Shaowen Wang, Linxi Yu, and Jian Li. LoRA-GA: Low-Rank Adaptation with Gradient Approximation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 54905–54931, 2024.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting Language Models with Long-Term Memory. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 74530–74543, 2023.

Zhengbo Wang, Jian Liang, Ran He, Zilei Wang, and Tieniu Tan. LoRA-Pro: Are Low-Rank Adapters Properly Optimized? In *International Conference on Learning Representations (ICLR)*, pp. 93787–93808, 2025.

Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing Transformers. In *International Conference on Learning Representations (ICLR)*, 2022.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering Large Pre-Trained Language Models to Follow Complex Instructions. In *International Conference on Learning Representations (ICLR)*, pp. 30745–30766, 2024.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing Linear Transformers with the Delta Rule over Sequence Length. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 115491–115522, 2024.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated Delta Networks: Improving Mamba2 with Delta Rule. In *International Conference on Learning Representations (ICLR)*, pp. 29687–29707, 2025.

Jingqi Ye, Haonan He, Minglei Li, Fujun Han, Tao Chen, and Peng Ye. Gradient Intrinsic Dimensionality Alignment: Narrowing the Gap Between Low-Rank Adaptation and Full Fine-Tuning. In *International Conference on Learning Representations (ICLR)*, pp. 55765–55790, 2026.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. MetaMath: Bootstrap Your Own Mathematical Questions for Large Language Models. In *International Conference on Learning Representations (ICLR)*, pp. 45040–45061, 2024.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 4791–4800, 2019.

Michael Zhang, Simran Arora, Rahul Chalamala, Alan Wu, Benjamin Spector, Aaryan Singhal, Krithik Ramesh, and Christopher Ré. LoLCATs: On Low-Rank Linearizing of Large Language Models. In *International Conference on Learning Representations (ICLR)*, pp. 46200–46253, 2025.

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.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 46595–46623, 2023.

Tianyu Zheng, Ge Zhang, Tianhao Shen, Xueling Liu, Bill Yuchen Lin, Jie Fu, Wenhu Chen, and Xiang Yue. OpenCodeInterpreter: Integrating Code Generation with Execution and Refinement. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12834–12859, 2024.

APPENDIX CONTENTS

A Notation and Theoretical Foundations **14**
- A.1 Notation and Dimensional Conventions . . . 14
- A.2 Coordinate Invariance and Output-Level Variation . . . 16
- A.3 Proof of the Width Bound on Correction Variation . . . 17
- A.4 Attainable Multi-Query Correction Span . . . 18
- A.5 Common–Differential Decomposition . . . 20
- A.6 Attention-Conditioned Visibility . . . 21
- A.7 Geometry of Task-Aligned Coordinate Initialization . . . 22
- A.8 Static Direct Capacity and History-Dependent Associative Capacity . . . 24
- A.9 Why Parallel Low-Rank Branches Alone Are One Wider Factorization . . . 24
- A.10 Query-Diversity Regularization . . . 25
- A.11 Scope of the Analysis . . . 25

B Algorithm and Implementation Details **26**
- B.1 Module Placement and Runtime Objects . . . 26
- B.2 Joint Task-Gradient Initialization . . . 27
- B.3 MQ-ACM Forward Computation . . . 28
- B.4 Training Procedure . . . 30
- B.5 Autoregressive Inference . . . 31
- B.6 Numerical Precision, Checkpoints, and State Storage . . . 31
- B.7 Parameter and Computational Scaling . . . 31

C Experimental Details **32**
- C.1 Natural-Language Generation . . . 32
- C.2 Commonsense Reasoning . . . 33
- C.3 Baseline Hyperparameters . . . 34
- C.4 Training Environment . . . 34

D Mechanistic Diagnostic Protocols **35**
- D.1 Diagnostic Overview . . . 35
- D.2 Effective Correction Rank . . . 35
- D.3 Held-Out Gradient-Subspace Coverage . . . 36
- D.4 Shared Intervention Design . . . 37
- D.5 Associative-Path Scaling . . . 37
- D.6 Query-Specific Differential Scaling . . . 38
- D.7 Query-Pair Similarity Before and After Decoding . . . 39
- D.8 Scope of the Diagnostics . . . 39

E Supplementary Experimental Results **39**
- E.1 Complete Commonsense Deviations . . . 39
- E.2 Fixed-R Structural Control . . . 40
- E.3 Full Gradient-Subspace Coverage Analysis . . . 40
- E.4 Associative and Query-Differential Interventions . . . 41
- E.5 Raw-Read and Decoded-Correction Similarity . . . 42
- E.6 Summary of Diagnostic Evidence . . . 42

A Notation and Theoretical Foundations

The appendix uses the same notation as the main paper: r denotes the local state/read width, K the number of query groups, and $R = Kr$ the aggregate exposed readout width. Algebraic and effective correction variation are defined separately from this allocated interface width.

This appendix supplies the definitions, dimensional conventions, and proofs underlying the readout-centric analysis in the main paper. The derivations use the exact MQ-ACM computation described in the method section: one shared $r \times r$ associative state, a full-Kr-dimensional normalized query projection reshaped into K local queries, one shared memory key–value write pair, a direct map partitioned into K groups, and projection-specific group decoders. We distinguish throughout among the local width r, the aggregate readout width $R = Kr$, the rank of the shared state, and the realized output-space variation of a trained correction.

Clarification of schematic terminology. The phrase *realized rank* in Figure 1(b) of the main paper is schematic shorthand for the distribution-dependent effective correction rank r_{eff}. In Figure 1(c), *occupy* refers to data-dependent readout variation Σ_z. The initialization panel in Figure 2 supplies paired task-gradient-informed coordinates; its "task-aligned occupancy" label denotes a design aim, while fine-tuning determines coordinate use. The write operation in Figure 2 reads from the old state S_{t-1} and produces S_t; runtime corrections use the trained decoders D_h initialized from their displayed initial values.

A.1 Notation and Dimensional Conventions

Table A.1: Notation and dimensions.

Symbol	Shape	Meaning
d	scalar	Input hidden dimension
d_h	scalar	Output dimension of target projection h
r	scalar	Local memory-state and local-read width
K	scalar	Number of query groups
$R = Kr$	scalar	Aggregate exposed readout width
S_t	$r \times r$	Shared associative state after position t
$q^m_{t,k}$	r	Query for local read k
k^m_t, v^m_t	r	Shared memory write key and value
$a_{t,k}$	r	Associative read $S_{t-1}q^m_{t,k}$
$P_k x_t$	r	Group-k direct signal
$r_{t,k}$	r	Combined group readout
z_t	R	Concatenated readout
$D_{h,k}$	$d_h \times r$	Group-specific decoder block
D_h	$d_h \times R$	Full decoder for projection h
s_R	scalar	Width-dependent correction scale
Σ_z	$R \times R$	Population covariance of z_t
$\Sigma_{\Delta,h}$	$d_h \times d_h$	Population covariance of decoded corrections
$r_{\text{alg},h}$	scalar	Algebraic dimension of correction variation
$r_{\text{eff},h}$	scalar	Entropy effective correction rank

For a selected frozen attention projection W_h, where $h \in \{q, k, v, o\}$, let

$$W_h \in \mathbb{R}^{d_h \times d}, \qquad x_t \in \mathbb{R}^d, \qquad y_{t,h} \in \mathbb{R}^{d_h}.$$

The generic adaptation interface used in the main-paper analysis is

$$y_{t,h} = W_h x_t + \Delta h_t, \qquad \Delta h_t = s_R D_h z_t, \tag{A.1}$$

where $z_t \in \mathbb{R}^R$ is the exposed readout, $D_h \in \mathbb{R}^{d_h \times R}$ is the correction decoder, and $s_R > 0$ is a scalar correction scale. In MQ-ACM,

$$R = Kr, \qquad s_R = \frac{\alpha}{\sqrt{R}}, \tag{A.2}$$

where r is the local state/read width and K is the number of query groups. The quantity R is the dimension of the concatenated exposed readout; algebraic correction variation is defined separately through $\Sigma_{\Delta,h}$.

EXACT MQ-ACM STATE AND READOUT DEFINITIONS

For each selected attention block, MQ-ACM maintains one state

$$S_t \in \mathbb{R}^{r\times r}.$$

Given x_t, the memory projections are

$$\widetilde{Q}_t^m = \mathrm{Norm}\big(\tanh(W_q^m x_t)\big) \in \mathbb{R}^{Kr}, \tag{A.3}$$

$$Q_t^m = \mathrm{reshape}(\widetilde{Q}_t^m) = \{q_{t,k}^m\}_{k=1}^K, \qquad q_{t,k}^m \in \mathbb{R}^r, \tag{A.4}$$

$$k_t^m = \mathrm{Norm}(\tanh(W_k^m x_t)) \in \mathbb{R}^r,$$

$$v_t^m = W_v^m x_t \in \mathbb{R}^r, \tag{A.5}$$

and

$$\beta_t = \sigma(w_\beta^\top x_t + b_\beta) \in (0,1), \qquad \lambda_t = 1 - \beta_t. \tag{A.6}$$

Here

$$\mathrm{Norm}(u) = \frac{u}{\max(\|u\|_2, \epsilon)}. \tag{A.7}$$

Importantly, the normalization in Equation (A.3) is applied once to the complete Kr-dimensional query projection before it is reshaped into K local queries. The individual $q_{t,k}^m$ are therefore not independently normalized.

MQ-ACM uses a read-before-write order:

$$a_{t,k} = S_{t-1} q_{t,k}^m, \qquad a_{t,k} \in \mathbb{R}^r, \tag{A.8}$$

followed by the scalar-gated delta update

$$S_t = \lambda_t S_{t-1} + \beta_t \left(v_t^m - S_{t-1} k_t^m\right) (k_t^m)^\top. \tag{A.9}$$

Thus, the read at position t depends on S_{t-1} and cannot directly retrieve the key–value pair written at the same position. The residual $v_t^m - S_{t-1}k_t^m$ is the value-prediction error under the previous state.

The direct map is

$$P \in \mathbb{R}^{R\times d}, \qquad P = \begin{bmatrix} P_1 \\ \vdots \\ P_K \end{bmatrix}, \qquad P_k \in \mathbb{R}^{r\times d}. \tag{A.10}$$

Each group combines its associative read with the corresponding direct signal:

$$r_{t,k} = a_{t,k} + P_k x_t, \tag{A.11}$$

$$z_t = \mathrm{Concat}(r_{t,1}, \ldots, r_{t,K}) \in \mathbb{R}^R. \tag{A.12}$$

For projection h, the decoder is partitioned as

$$D_h = [D_{h,1}, \ldots, D_{h,K}], \qquad D_{h,k} \in \mathbb{R}^{d_h \times r}. \tag{A.13}$$

The total correction is

$$\Delta h_t = s_R D_h z_t = \Delta h_t^{\mathrm{assoc}} + \Delta h_t^{\mathrm{dir}}, \tag{A.14}$$

with

$$\Delta h_t^{\mathrm{assoc}} = s_R \sum_{k=1}^{K} D_{h,k} a_{t,k}, \qquad \Delta h_t^{\mathrm{dir}} = s_R \sum_{k=1}^{K} D_{h,k} P_k x_t. \tag{A.15}$$

A.2 Coordinate Invariance and Output-Level Variation

The coordinates of z_t are not identifiable by themselves. Let $T \in \mathbb{R}^{R \times R}$ be invertible and define

$$\widetilde{z}_t = T^{-1} z_t, \qquad \widetilde{D}_h = D_h T. \tag{A.16}$$

Then

$$s_R \widetilde{D}_h \widetilde{z}_t = s_R D_h T T^{-1} z_t = s_R D_h z_t. \tag{A.17}$$

Therefore, any invertible change of latent basis leaves the decoded correction unchanged. This is why the main paper measures output-space variation rather than assigning semantic meaning to individual latent coordinates.

Under a specified evaluation distribution over tokens and histories, define

$$\Sigma_z = \mathrm{Cov}(z_t), \qquad \Sigma_{\Delta,h} = \mathrm{Cov}(\Delta h_t) = s_R^2 D_h \Sigma_z D_h^\top. \tag{A.18}$$

Under the transformed coordinates,

$$\mathrm{Cov}(\widetilde{z}_t) = T^{-1} \Sigma_z T^{-\top}, \tag{A.19}$$

and consequently

$$s_R^2 \widetilde{D}_h \, \mathrm{Cov}(\widetilde{z}_t) \widetilde{D}_h^\top = s_R^2 D_h \Sigma_z D_h^\top. \tag{A.20}$$

Thus, $\Sigma_{\Delta,h}$, its algebraic rank, and its effective rank are invariant to any invertible reparameterization of the exposed coordinates.

Scope of the invariance. Equations (A.16)–(A.20) establish an output-level equivalence of the exposed readout–decoder representation. The MQ-ACM grouping structure, recurrent state construction, and query-diversity parameterization remain fixed under this analysis. The invariant object used below is the decoded correction covariance $\Sigma_{\Delta,h}$, which is independent of the particular latent basis chosen for the exposed coordinates.

Let

$$r_{\mathrm{alg},h} = \mathrm{rank}(\Sigma_{\Delta,h}). \tag{A.21}$$

When $\Sigma_{\Delta,h} \neq 0$, let $\lambda_1, \ldots, \lambda_m > 0$ be its positive eigenvalues, where $m = r_{\mathrm{alg},h}$, and define

$$\pi_i = \frac{\lambda_i}{\sum_{j=1}^{m} \lambda_j}. \tag{A.22}$$

The effective correction rank is

$$r_{\mathrm{eff},h} = \exp\left(-\sum_{i=1}^{m} \pi_i \log \pi_i \right). \tag{A.23}$$

When $\Sigma_{\Delta,h} = 0$, we set $r_{\mathrm{eff},h} = 0$.

Because multiplying $\Sigma_{\Delta,h}$ by any positive scalar leaves the normalized eigenvalue distribution $\{\pi_i\}$ unchanged, $r_{\mathrm{eff},h}$ measures the spread of variation across active output directions independently of the overall correction magnitude.

A.3 Proof of the Width Bound on Correction Variation

Lemma A.1 (Width bounds correction variation)**.** *For the interface in Equation (A.1),*

$$0 \le r_{\mathrm{eff},h} \le r_{\mathrm{alg},h} \le \min\{\mathrm{rank}(D_h), \mathrm{rank}(\Sigma_z), R\}. \tag{A.24}$$

If $\Sigma_{\Delta,h} \neq 0$*, then* $r_{\mathrm{eff},h} \ge 1$.

Proof. The covariance Σ_z is positive semidefinite, so it has a positive-semidefinite square root $\Sigma_z^{1/2}$. Equation (A.18) can be written as

$$\Sigma_{\Delta,h} = s_R^2 \left(D_h \Sigma_z^{1/2} \right) \left(D_h \Sigma_z^{1/2} \right)^{\top}. \tag{A.25}$$

For any matrix B,

$$\mathrm{rank}(BB^{\top}) = \mathrm{rank}(B).$$

Since $s_R > 0$,

$$r_{\mathrm{alg},h} = \mathrm{rank}(D_h \Sigma_z^{1/2}). \tag{A.26}$$

Using the standard rank inequality for a matrix product,

$$\mathrm{rank}(D_h \Sigma_z^{1/2}) \le \min\{\mathrm{rank}(D_h), \mathrm{rank}(\Sigma_z^{1/2})\}. \tag{A.27}$$

A positive-semidefinite matrix and its square root have the same rank, hence

$$r_{\mathrm{alg},h} \le \min\{\mathrm{rank}(D_h), \mathrm{rank}(\Sigma_z)\}. \tag{A.28}$$

Because D_h has R columns and $\Sigma_z \in \mathbb{R}^{R \times R}$,

$$\mathrm{rank}(D_h) \le R, \qquad \mathrm{rank}(\Sigma_z) \le R, \tag{A.29}$$

which yields the final algebraic-rank bound in Equation (A.24).

It remains to bound the entropy effective rank. If $\Sigma_{\Delta,h} = 0$, then $r_{\mathrm{alg},h} = 0$, and the definition $r_{\mathrm{eff},h} = 0$ gives the result directly.

Now suppose $\Sigma_{\Delta,h} \neq 0$. Because it is positive semidefinite, the number m of positive eigenvalues equals its algebraic rank:

$$m = r_{\mathrm{alg},h} \geq 1. \tag{A.30}$$

The normalized eigenvalues $\pi_1, \ldots, \pi_m$ form a probability distribution. Its Shannon entropy satisfies

$$0 \leq -\sum_{i=1}^{m} \pi_i \log \pi_i \leq \log m. \tag{A.31}$$

The upper bound follows, for example, from the nonnegativity of the Kullback–Leibler divergence between $\{\pi_i\}$ and the uniform distribution on m elements. Exponentiating Equation (A.31) gives

$$1 \leq r_{\mathrm{eff},h} \leq m = r_{\mathrm{alg},h}. \tag{A.32}$$

Combining Equations (A.28)–(A.32) proves the lemma. □

Interpretation The three factors in Equation (A.25) play different roles:

- R specifies how many coordinates the interface allocates;
- Σ_z records which combinations of those coordinates actually vary under the evaluation distribution; and
- D_h determines which varying combinations survive after decoding.

The bound characterizes correction variation: R specifies the allocated coordinates, while the active distribution over those coordinates is determined by the trained readout, decoder, and evaluation data. The relationship between $r_{\mathrm{eff},h}$ and downstream accuracy remains empirical.

A.4 Attainable Multi-Query Correction Span

For the associative contribution, fix a state $S \in \mathbb{R}^{r\times r}$ and suppress the token index. Define the stacked query

$$q = \begin{bmatrix} q_1 \\ \vdots \\ q_K \end{bmatrix} \in \mathbb{R}^{Kr} \tag{A.33}$$

and

$$M_h(S) = [D_{h,1}S, \ldots, D_{h,K}S] \in \mathbb{R}^{d_h \times Kr}. \tag{A.34}$$

Then the associative correction is

$$\Delta h^{\mathrm{assoc}} = s_R M_h(S) q. \tag{A.35}$$

Tied-block notation. The full decoder introduced in Equation (A.13) has shape $D_h \in \mathbb{R}^{d_h \times R}$, whereas each group block has shape $D_{h,k} \in \mathbb{R}^{d_h \times r}$. In the tied-decoder special case, we therefore denote the common block by $D_h^\star \in \mathbb{R}^{d_h \times r}$. The same notation is used in Proposition 1 of the main paper, keeping the shared block distinct from the full decoder.

Proposition A.1 (Attainable multi-query correction span)**.** *If the K query vectors vary freely, the set of attainable associative corrections for fixed S is* $\mathrm{Col}(M_h(S))$*, and*

$$\mathrm{rank}(M_h(S)) \leq \min\{d_h, K\,\mathrm{rank}(S)\} \leq \min\{d_h, Kr\}. \tag{A.36}$$

If $D_{h,1} = \cdots = D_{h,K} = D_h^\star$, then the span reduces to $\mathrm{Col}(D_h^\star S)$*, whose dimension is at most r.*

Proof. Because $s_R > 0$, scalar multiplication by s_R does not change a column space. As q varies over $\mathbb{R}^{Kr}$, the set

$$\{s_R M_h(S)q : q \in \mathbb{R}^{Kr}\}$$

is exactly $\mathrm{Col}(M_h(S))$.

The matrix $M_h(S)$ has d_h rows, so

$$\mathrm{rank}(M_h(S)) \leq d_h. \tag{A.37}$$

For a horizontal concatenation,

$$\mathrm{rank}([A_1, \ldots, A_K]) \leq \sum_{k=1}^{K} \mathrm{rank}(A_k). \tag{A.38}$$

Applying this with $A_k = D_{h,k}S$ gives

$$\mathrm{rank}(M_h(S)) \leq \sum_{k=1}^{K} \mathrm{rank}(D_{h,k}S). \tag{A.39}$$

Each product satisfies

$$\mathrm{rank}(D_{h,k}S) \leq \mathrm{rank}(S), \tag{A.40}$$

so

$$\mathrm{rank}(M_h(S)) \leq K\,\mathrm{rank}(S). \tag{A.41}$$

Combining Equations (A.37) and (A.41) yields the first bound in Equation (A.36). Since $S \in \mathbb{R}^{r\times r}$,

$$\mathrm{rank}(S) \leq r, \tag{A.42}$$

which yields the second bound.

If all decoder blocks are tied,

$$D_{h,1} = \cdots = D_{h,K} = D_h^\star,$$

then

$$M_h(S) = [D_h^\star S, \ldots, D_h^\star S]. \tag{A.43}$$

Repeating the same block does not enlarge its column space, hence

$$\mathrm{Col}(M_h(S)) = \mathrm{Col}(D_h^\star S), \tag{A.44}$$

and

$$\mathrm{rank}(M_h(S)) = \mathrm{rank}(D_h^\star S) \leq \mathrm{rank}(S) \leq r. \tag{A.45}$$

This proves the proposition. □

STATE EXPOSURE VERSUS STATE DIMENSION

The proposition distinguishes the state from its exposed interface. Before decoding, the stacked associative reads satisfy

$$\begin{bmatrix} a_1 \\ \vdots \\ a_K \end{bmatrix} = (I_K \otimes S) \begin{bmatrix} q_1 \\ \vdots \\ q_K \end{bmatrix}. \tag{A.46}$$

Because

$$\operatorname{rank}(I_K \otimes S) = K \operatorname{rank}(S), \tag{A.47}$$

placing K reads in separate coordinate blocks can expose up to $K \operatorname{rank}(S)$ stacked directions even though the stored state remains one $r \times r$ matrix. Group-specific decoder blocks can map the separate read blocks to different output directions. Tying the decoder blocks eliminates the possibility of expanding the associative output span beyond $\operatorname{Col}(D_h^\star S)$ through group-specific decoder orientation; it does not prevent multiple reads from changing coefficients within that common span.

The freely varying-query condition is an algebraic envelope. In the implemented model, the queries are generated from the same x_t, jointly normalized in $\mathbb{R}^{Kr}$, and restricted by learned parameters and the data distribution. The realized correction covariance can therefore occupy a strict subset of the span in Equation (A.36).

A.5 COMMON–DIFFERENTIAL DECOMPOSITION

Let

$$a_{t,k} = S_{t-1} q_{t,k}^m, \qquad \bar{a}_t = \frac{1}{K} \sum_{k=1}^{K} a_{t,k}, \qquad \epsilon_{t,k} = a_{t,k} - \bar{a}_t, \tag{A.48}$$

and define

$$\bar{D}_h = \frac{1}{K} \sum_{k=1}^{K} D_{h,k}, \qquad \Delta D_{h,k} = D_{h,k} - \bar{D}_h. \tag{A.49}$$

By construction,

$$\sum_{k=1}^{K} \epsilon_{t,k} = 0, \qquad \sum_{k=1}^{K} \Delta D_{h,k} = 0. \tag{A.50}$$

Expanding the unscaled associative decoder output gives

$$\begin{aligned} \sum_{k=1}^{K} D_{h,k} a_{t,k} &= \sum_{k=1}^{K} (\bar{D}_h + \Delta D_{h,k})(\bar{a}_t + \epsilon_{t,k}) \\ &= K \bar{D}_h \bar{a}_t + \bar{D}_h \sum_{k=1}^{K} \epsilon_{t,k} + \left(\sum_{k=1}^{K} \Delta D_{h,k} \right) \bar{a}_t + \sum_{k=1}^{K} \Delta D_{h,k} \epsilon_{t,k} \\ &= K \bar{D}_h \bar{a}_t + \sum_{k=1}^{K} (D_{h,k} - \bar{D}_h) \epsilon_{t,k}. \end{aligned} \tag{A.51}$$

Multiplying by s_R yields the corresponding decomposition of $\Delta h_t^{\mathrm{assoc}}$.

The first term in Equation (A.51) is the common component: it maps the mean read through the mean decoder. The second is the query-specific differential component: it couples deviations among reads with deviations among decoder blocks.

DISPERSION BOUND

Let

$$e_{t,h}^{\text{diff}} = \sum_{k=1}^{K} (D_{h,k} - \bar{D}_h)\epsilon_{t,k}. \tag{A.52}$$

Using the triangle inequality and the matrix operator norm,

$$\begin{aligned} \|e_{t,h}^{\text{diff}}\|_2 &\le \sum_{k=1}^{K} \|(D_{h,k} - \bar{D}_h)\epsilon_{t,k}\|_2 \\ &\le \sum_{k=1}^{K} \|D_{h,k} - \bar{D}_h\|_2 \|\epsilon_{t,k}\|_2. \end{aligned} \tag{A.53}$$

Applying Cauchy–Schwarz to the final sum gives

$$\|e_{t,h}^{\text{diff}}\|_2 \le \left(\sum_{k=1}^{K} \|D_{h,k} - \bar{D}_h\|_2^2 \right)^{1/2} \left(\sum_{k=1}^{K} \|\epsilon_{t,k}\|_2^2 \right)^{1/2}. \tag{A.54}$$

The scaled differential correction is bounded by s_R times the right-hand side.

Equation (A.54) shows two necessary conditions for a nonzero query-specific decoded term:

1. the associative reads must differ, so at least one $\epsilon_{t,k} \neq 0$; and
2. the decoder blocks must differ, so at least one $D_{h,k} - \bar{D}_h \neq 0$.

Both conditions are necessary, while their relative orientation controls the surviving differential signal. The term can still vanish through cancellation when read deviations fall in decoder null spaces or when group contributions cancel after decoding.

RELATION TO THE QUERY-DIFFERENCE INTERVENTION

The intervention in § D rescales the second term of Equation (A.51) after group-specific decoding while keeping the common term, direct path, query construction, decoder parameters, and state-update rule unchanged at the intervention site. Because the modified correction changes downstream hidden activations, the complete network is recomputed and subsequent queries and numerical state trajectories may differ. This protocol measures functional use of the decoded query-specific deviation within a fixed trained checkpoint; separately retrained architectural controls are reported as distinct experiments.

A.6 ATTENTION-CONDITIONED VISIBILITY

A decoded query correction can vary without producing an equally large change in the frozen attention probabilities. To make the main paper's query-side visibility statement dimensionally explicit, consider one attention head g. Let the full query-projection output have dimension $d_q = H_q d_a$, where H_q is the number of query heads and d_a is the per-head query/key dimension. Let $E_g \in \mathbb{R}^{d_a \times d_q}$ select the coordinates of head g, and define

$$D_{q,g} = E_g D_q \in \mathbb{R}^{d_a \times R}.$$

Let $q_{t,g} \in \mathbb{R}^{d_a}$ be the baseline query of head g, and let $K_{\le t,g} \in \mathbb{R}^{d_a \times n_t}$ contain the frozen keys used by that head at position t. Define

$$\ell_{t,g} = \frac{K_{\le t,g}^\top q_{t,g}}{\sqrt{d_a}}, \qquad p_{t,g} = \mathrm{softmax}(\ell_{t,g}). \tag{A.55}$$

The MQ-ACM query correction restricted to head g is

$$\delta q_{t,g} = E_g(s_R D_q z_t) = s_R D_{q,g} z_t. \tag{A.56}$$

The induced logit perturbation is

$$\delta \ell_{t,g} = \frac{K_{\le t,g}^\top \delta q_{t,g}}{\sqrt{d_a}}. \tag{A.57}$$

For $p = \mathrm{softmax}(\ell)$, the Jacobian is

$$J(p) = \frac{\partial p}{\partial \ell} = \mathrm{Diag}(p) - pp^\top, \tag{A.58}$$

and componentwise

$$\frac{\partial p_i}{\partial \ell_j} = p_i(\mathbf{1}_{i=j} - p_j). \tag{A.59}$$

For sufficiently small $\delta q_{t,g}$, a first-order Taylor expansion gives

$$\begin{aligned} \delta p_{t,g} &= J(p_{t,g})\delta \ell_{t,g} + \mathcal{O}\big(\|\delta \ell_{t,g}\|_2^2\big) \\ &= \frac{1}{\sqrt{d_a}} J(p_{t,g}) K_{\le t,g}^\top \delta q_{t,g} + \mathcal{O}\big(\|\delta q_{t,g}\|_2^2\big) \\ &= \frac{s_R}{\sqrt{d_a}} J(p_{t,g}) K_{\le t,g}^\top D_{q,g} z_t + \mathcal{O}\big(\|s_R D_{q,g} z_t\|_2^2\big), \end{aligned} \tag{A.60}$$

where the second-order constant absorbs the fixed key matrix within the local neighborhood.

Because

$$J(p_{t,g})\mathbf{1} = 0, \tag{A.61}$$

a logit perturbation that is constant across the available keys produces no first-order change in the normalized attention probabilities. More generally, the first-order probability variation lies in

$$\mathcal{V}_{t,g} = \mathrm{Col}\left(J(p_{t,g}) K_{\le t,g}^\top D_{q,g}\right), \tag{A.62}$$

and hence

$$\dim(\mathcal{V}_{t,g}) \le \mathrm{rank}(D_{q,g}) \le \min\{d_a, R\}.$$

Thus, R upper-bounds the dimension of the local first-order attention-probability variation, while its magnitude and realized dimension depend on the local attention geometry and correction scale.

This per-head calculation supports the query-side visibility statement in the main paper under fixed keys and a sufficiently small query correction. Simultaneous Q/K/V/O corrections, changes in attention values, finite nonlinear perturbations, and complete recurrent evolution are evaluated at the model level.

A.7 Geometry of Task-Aligned Coordinate Initialization

Let $\mathcal{H} = \{h_1, \ldots, h_{|\mathcal{H}|}\}$ denote the selected target projections. The generation experiments use $\mathcal{H} = \{q, k, v, o\}$, whereas the commonsense experiments use $\mathcal{H} = \{q, k, v\}$.

With MQ-ACM corrections disabled, the calibration gradient for projection $h \in \mathcal{H}$ is

$$G_h = \frac{1}{N} \sum_{i=1}^{N} \nabla_{W_h} \mathcal{L}_i \in \mathbb{R}^{d_h \times d}. \tag{A.63}$$

Each branch is stabilized and normalized independently:

$$\bar{G}_h = \frac{G_h}{\max(\|G_h\|_F, \epsilon)}. \tag{A.64}$$

If $\|G_h\|_F \geq \epsilon$, then $\|\bar{G}_h\|_F = 1$. A zero or numerically negligible branch is not assigned unit weight by this stabilized definition.

The normalized branches are stacked by output rows:

$$G_J = \begin{bmatrix} \bar{G}_{h_1} \\ \vdots \\ \bar{G}_{h_{|\mathcal{H}|}} \end{bmatrix} \in \mathbb{R}^{(\sum_{h\in\mathcal{H}} d_h)\times d}. \tag{A.65}$$

This construction identifies the d input-coordinate axes of the selected projections. For Q/K/V this identification is literal because they share the same block input. When O is included, the construction uses the same d-dimensional channel indexing, while the underlying Q/K/V and O input activations remain projection-specific.

Assuming

$$R \leq \min\left\{d, \sum_{h\in\mathcal{H}} d_h\right\},$$

a rank-R truncated SVD gives

$$G_J \approx U_R \Sigma_R V_R^\top, \tag{A.66}$$

where

$$U_R = \begin{bmatrix} U_{h_1} \\ \vdots \\ U_{h_{|\mathcal{H}|}} \end{bmatrix}, \qquad U_h \in \mathbb{R}^{d_h\times R}, \qquad V_R \in \mathbb{R}^{d\times R}. \tag{A.67}$$

If R exceeds the numerical rank of G_J, only the directions associated with nonzero retained singular values are identified by the calibration gradients; any remaining basis directions are a numerical orthonormal completion.

The initialization is

$$P^{(0)} = V_R^\top, \qquad s_R D_h^{(0)} = \gamma U_h. \tag{A.68}$$

PAIRED INPUT AND OUTPUT COORDINATES

The columns of V_R define shared input coordinates, while the corresponding columns of each row block U_h define projection-specific output coordinates. Since V_R has orthonormal columns,

$$P^{(0)} P^{(0)\top} = V_R^\top V_R = I_R. \tag{A.69}$$

The complete left factor also has orthonormal columns:

$$U_R^\top U_R = \sum_{h\in\mathcal{H}} U_h^\top U_h = I_R. \tag{A.70}$$

Individual row blocks U_h need not have orthonormal columns. Orthogonality holds only for the complete stacked factor.

The retained subspaces are ordered by singular value. Individual singular vectors remain sign-ambiguous and, when singular values are repeated, are defined only up to an orthogonal rotation within the repeated-singular-value subspace. Consequently, the selected subspaces are the invariant objects; individual coordinate identities need not be unique.

Partitioning $P^{(0)}$ by rows and $D_h^{(0)}$ by columns assigns contiguous retained coordinates to P_k and $D_{h,k}$. The singular values determine selection and ordering, but are not inserted into the forward correction amplitude.

INITIAL DIRECT CORRECTION

The direct correction is a static matrix applied to x_t:

$$\Delta h_t^{\mathrm{dir}} = s_R D_h P x_t = \Delta W_h^{\mathrm{dir}} x_t, \qquad \Delta W_h^{\mathrm{dir}} = s_R D_h P. \tag{A.71}$$

At initialization,

$$\Delta W_h^{\mathrm{dir},(0)} = s_R D_h^{(0)} P^{(0)} = \gamma U_h V_R^\top . \tag{A.72}$$

Equation (A.72) uses the retained singular vectors to pair input and output coordinates, while γ sets the initialization magnitude. For branches above the stabilization threshold, Equation (A.64) equalizes their Frobenius scale before constructing the joint basis. The joint SVD is a task-related optimization choice for the paired interface; it is not uniquely prescribed by the readout analysis. Subsequent fine-tuning learns the memory queries and the use and data-dependent variation of these coordinates.

A.8 Static Direct Capacity and History-Dependent Associative Capacity

From Equation (A.71),

$$\mathrm{rank}(\Delta W_h^{\mathrm{dir}}) = \mathrm{rank}(s_R D_h P) \le \min\{\mathrm{rank}(D_h), \mathrm{rank}(P), R\} \le R. \tag{A.73}$$

Thus, the direct path is itself a history-independent linear correction of rank at most R. In grouped form,

$$D_h P = [D_{h,1}, \ldots, D_{h,K}] \begin{bmatrix} P_1 \\ \vdots \\ P_K \end{bmatrix} = \sum_{k=1}^{K} D_{h,k} P_k. \tag{A.74}$$

The associative path differs because its input is generated through the evolving state:

$$\Delta h_t^{\mathrm{assoc}} = s_R \sum_{k=1}^{K} D_{h,k} S_{t-1} q_{t,k}^m. \tag{A.75}$$

It is therefore history-conditioned and generally cannot be represented by one fixed matrix multiplying x_t across all sequences and positions.

The total readout can be decomposed as

$$z_t = z_t^{\mathrm{assoc}} + z_t^{\mathrm{dir}}. \tag{A.76}$$

Its covariance contains cross terms:

$$\Sigma_z = \Sigma_{\mathrm{assoc}} + \Sigma_{\mathrm{dir}} + \Sigma_{\mathrm{assoc,dir}} + \Sigma_{\mathrm{dir,assoc}}. \tag{A.77}$$

Therefore, algebraic rank and entropy effective rank depend on the covariance of the combined readout, including cross terms between the associative and direct paths. Figure 3(a) accordingly reports the effective rank of the complete decoded correction. Because the two paths are optimized through their sum, full-model performance reflects the jointly trained interface.

A.9 Why Parallel Low-Rank Branches Alone Are One Wider Factorization

Suppose K static low-rank branches receive the same input and their outputs are summed:

$$\sum_{k=1}^{K} B_k A_k x, \qquad A_k \in \mathbb{R}^{r \times d}, \quad B_k \in \mathbb{R}^{d_h \times r}. \tag{A.78}$$

Define

$$B_{\mathrm{cat}} = [B_1, \ldots, B_K] \in \mathbb{R}^{d_h \times Kr}, \tag{A.79}$$

and

$$A_{\text{cat}} = \begin{bmatrix} A_1 \\ \vdots \\ A_K \end{bmatrix} \in \mathbb{R}^{Kr \times d}. \tag{A.80}$$

Then

$$\sum_{k=1}^{K} B_k A_k x = B_{\text{cat}} A_{\text{cat}} x. \tag{A.81}$$

Accordingly, K static low-rank branches receiving the same input are algebraically equivalent to one factorization with width at most Kr. MQ-ACM adds a different computational structure: one shared state is written through an r-dimensional key–value pathway, queried through K input-conditioned reads, combined with a direct signal, and decoded through group-specific blocks. The span proposition and common–differential decomposition characterize when those queried reads remain distinct after decoding.

A.10 Query-Diversity Regularization

The training objective is

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_{\text{div}} \mathcal{L}_{\text{div}}. \tag{A.82}$$

For mathematical completeness, define the stabilized cosine

$$\cos_{\epsilon_{\cos}}(u, v) = \frac{u^\top v}{\max(\|u\|_2, \epsilon_{\cos}) \max(\|v\|_2, \epsilon_{\cos})},$$

where $\epsilon_{\cos} > 0$ denotes the denominator stabilization used by the implementation. For $K > 1$,

$$\mathcal{L}_{\text{div}} = \mathbb{E}_t \left[\frac{1}{K(K-1)} \sum_{i \neq j} \cos^2_{\epsilon_{\cos}}(q^m_{t,i}, q^m_{t,j}) \right], \tag{A.83}$$

and $\mathcal{L}_{\text{div}} = 0$ for $K = 1$.

The squared stabilized cosine penalizes both positively and negatively collinear query directions and acts directly on the query vectors $q^m_{t,k}$. Multiplication by a low-rank or anisotropic state can change their relationships, and group-specific decoders can further reshape the resulting reads. We therefore use the regularizer as an auxiliary pressure against query-direction collapse and evaluate read- and output-level differentiation separately.

A.11 Scope of the Analysis

The theoretical statements above define the objects used by the readout-centric interpretation and specify the level at which each result applies. The motivating split-LoRA observation uses external data organization, while MQ-ACM learns input- and history-conditioned reads from a common training stream. Query groups have no predefined subset or semantic identities. The diagnostics evaluate output variation and functional use; they do not assign semantic labels to groups.

Interface dimensions and realized variation. The local width r specifies the shared-state dimension, the memory write width, and each local read. The aggregate exposed readout width is $R = Kr$. Increasing K therefore leaves the $r \times r$ state and r-dimensional key–value write bottleneck unchanged while enlarging the query/direct/decoder interface. Realized correction variation is characterized separately through $\Sigma_{\Delta,h}$, $r_{\text{alg},h}$, and $r_{\text{eff},h}$, which depend on the evaluation distribution, trained readout covariance, and decoder geometry.

Multi-query expression. Proposition A.1 describes an attainable algebraic envelope for fixed state S under freely varying query coefficients. The implemented query generator, shared input, full-vector normalization, and data distribution determine the realized subset of this envelope. Equation (A.51) further shows that query-specific decoded effects are controlled jointly by read dispersion and decoder dispersion, with their relative orientation determining the surviving differential signal.

Coordinate initialization and the complete correction. Joint task-gradient initialization selects paired input/output coordinates: right singular vectors initialize the direct-input basis, projection-specific left blocks initialize decoder outputs, and γ sets the initial correction scale. Fine-tuning subsequently determines coordinate use. The complete correction combines a history-conditioned associative contribution with the static direct map $s_R D_h P$, so full-model results characterize their jointly optimized interface.

Analytical and structural controls. The attention-conditioned visibility result applies locally to sufficiently small query corrections with fixed keys. Model-level Q/K/V/O behavior is evaluated through the trained system. The fixed-R sweep in the diagnostics compares alternative interface factorizations; changing (K, r) at constant R also changes local/state width, state storage, parameter count, and diversity geometry.

Runtime state and diagnostics. At inference, learned parameters remain fixed and the $r \times r$ state evolves through the forward recurrence in Equation (A.9). Effective rank and gradient coverage are computed offline from trained checkpoints. The backbone retains its standard KV cache, while MQ-ACM adds the recurrent state and readout computation described in § B.7.

B Algorithm and Implementation Details

This appendix gives the executable MQ-ACM procedure corresponding to the definitions in the main paper and § A. We separate one-time task-gradient initialization, differentiable sequence processing during fine-tuning, and stateful autoregressive inference. The pretrained backbone remains frozen throughout MQ-ACM training and inference.

B.1 Module Placement and Runtime Objects

MQ-ACM is attached to each selected Transformer attention block. For a block with hidden width d, the module receives the block input $x_t \in \mathbb{R}^d$ and produces additive corrections for a set of frozen attention projections

$$\mathcal{H} \subseteq \{q, k, v, o\}.$$

The natural-language generation experiments use $\mathcal{H} = \{q, k, v, o\}$, whereas the commonsense experiments use $\mathcal{H} = \{q, k, v\}$.

For one attention block, MQ-ACM contains a Kr-dimensional memory-query projection W_q^m, shared r-dimensional memory-key and memory-value projections W_k^m, W_v^m, a scalar write gate, a direct map $P \in \mathbb{R}^{R \times d}$, and one decoder $D_h \in \mathbb{R}^{d_h \times R}$ for every $h \in \mathcal{H}$, where

$$R = Kr.$$

The K query groups share the same $r \times r$ associative state and the same key–value write pathway. The target projections also share the exposed readout z_t, but use projection-specific decoders D_h.

Architectural scope. The construction assumes causal token processing, additive access to selected attention-projection outputs, and a persistent per-layer recurrent state. Projection-specific dimensions d_h allow the same formulation to cover attention implementations with different Q/K/V/O output widths. The derivation uses these architectural interfaces rather than Llama-specific normalization or positional-encoding components. The empirical study in the main paper is restricted to Llama-family 8B models.

The state is a runtime activation rather than a learned parameter. A separate state is maintained for every active sequence and every adapted layer. It is initialized to zero for a new sequence, updated during the forward computation, and excluded from the adapter checkpoint.

B.2 Joint Task-Gradient Initialization

Joint initialization is performed once before formal fine-tuning using calibration batches drawn from the training split. Downstream evaluation examples are reserved for evaluation and diagnostics.

Calibration-gradient collection

For layer ℓ and target projection h, we accumulate

$$G_{\ell,h} = \frac{1}{N_{\text{cal}}} \sum_{i=1}^{N_{\text{cal}}} \nabla_{W_{\ell,h}} \mathcal{L}_i. \tag{B.1}$$

The experiments use $N_{\text{cal}} = 32$ calibration batches. During this stage, the backbone remains frozen, MQ-ACM corrections are disabled when collecting the target gradients, and no optimizer step is taken. Target-projection gradients are accumulated in FP32 and averaged across processes.

Each nonzero branch is normalized independently:

$$\bar{G}_{\ell,h} = \frac{G_{\ell,h}}{\max(\|G_{\ell,h}\|_F, \epsilon)}, \qquad \epsilon = 10^{-6}. \tag{B.2}$$

This normalization gives the selected Q/K/V/O or Q/K/V branches equal weighting in the joint matrix rather than preserving their original Frobenius-norm ratios.

Layer-wise joint decomposition

For each attention block, the normalized gradients are stacked by output rows:

$$G_{\ell,J} = \begin{bmatrix} \bar{G}_{\ell,h_1} \\ \bar{G}_{\ell,h_2} \\ \vdots \\ \bar{G}_{\ell,h_{|\mathcal{H}|}} \end{bmatrix}. \tag{B.3}$$

A randomized rank-R SVD is then computed:

$$G_{\ell,J} \approx U_{\ell,R} \Sigma_{\ell,R} V_{\ell,R}^{\top}. \tag{B.4}$$

The decomposition uses oversampling 16, two power iterations, CPU FP32 arithmetic, and layer seed `training_seed + layer_index`. Partitioning the left singular vectors by the row ranges of the selected projections gives

$$U_{\ell,R} = \begin{bmatrix} U_{\ell,h_1} \\ \vdots \\ U_{\ell,h_{|\mathcal{H}|}} \end{bmatrix}. \tag{B.5}$$

The direct map and decoders are initialized as

$$P_{\ell}^{(0)} = V_{\ell,R}^{\top}, \tag{B.6}$$

$$s_R D_{\ell,h}^{(0)} = \gamma U_{\ell,h}, \qquad s_R = \frac{\alpha}{\sqrt{R}}, \qquad \gamma = 10^{-3}. \tag{B.7}$$

The singular values determine coordinate ordering and truncation, while γ sets the initialization amplitude.

Algorithm B.1 Joint task-gradient coordinate initialization.

Input: Frozen backbone; selected projections $\mathcal{H}$; local width r; query count K; $R = Kr$; calibration loader; $N_{\text{cal}} = 32$
Output: Initialized direct maps $\{P_\ell\}$ and decoders $\{D_{\ell,h}\}$

```
Disable MQ-ACM correction outputs
Initialize FP32 gradient accumulators G_{ℓ,h} ← 0
for i = 1, ..., N_cal do
    Compute the task loss on calibration batch i
    Accumulate gradients of each frozen target projection W_{ℓ,h}
end for
Average the accumulators across distributed processes
for all adapted layers ℓ do
    for all h ∈ H do
        Ḡ_{ℓ,h} ← G_{ℓ,h} / max(‖G_{ℓ,h}‖_F, ε)
    end for
    Stack {Ḡ_{ℓ,h}}_{h∈H} by output rows to obtain G_{ℓ,J}
    Compute the randomized rank-R SVD G_{ℓ,J} ≈ U_{ℓ,R} Σ_{ℓ,R} V_{ℓ,R}^⊤
    P_ℓ ← V_{ℓ,R}^⊤
    for all h ∈ H do
        Extract row block U_{ℓ,h} and set D_{ℓ,h} ← (γ/s_R) U_{ℓ,h}
    end for
end for
Re-enable MQ-ACM corrections
```

The no-joint-initialization ablation bypasses Algorithm B.1 and uses the corresponding zero-start initialization.

B.3 MQ-ACM Forward Computation

Batched state and memory projections

For batch size B, the implementation stores

$$S_t \in \mathbb{R}^{B\times r\times r}, \tag{B.8}$$

with $S_0 = 0$ for a new sequence. The equations below are written for one sequence and vectorized over the batch in the implementation.

Given x_t, the complete memory-query vector is

$$\widetilde{Q}_t^m = \mathrm{Norm}\big(\tanh(W_q^m x_t)\big) \in \mathbb{R}^R, \tag{B.9}$$

and is then reshaped into K contiguous groups:

$$Q_t^m = \mathrm{reshape}(\widetilde{Q}_t^m) = \{q_{t,k}^m\}_{k=1}^K, \qquad q_{t,k}^m \in \mathbb{R}^r. \tag{B.10}$$

Normalization is applied once in the full $R = Kr$ dimensional space before reshaping; the local queries are not normalized independently.

The shared write key, value, and gate are

$$k_t^m = \mathrm{Norm}(\tanh(W_k^m x_t))\,, \tag{B.11}$$

$$v_t^m = W_v^m x_t, \tag{B.12}$$

$$\beta_t = \sigma(w_\beta^\top x_t + b_\beta), \qquad \lambda_t = 1 - \beta_t. \tag{B.13}$$

The normalization function is

$$\mathrm{Norm}(u) = \frac{u}{\max(\|u\|_2, 10^{-6})}. \tag{B.14}$$

READ, DIRECT COOPERATION, AND DECODING

MQ-ACM first reads the old state:

$$a_{t,k} = S_{t-1} q_{t,k}^m. \tag{B.15}$$

Partition the direct map by rows,

$$P = \begin{bmatrix} P_1 \\ \vdots \\ P_K \end{bmatrix}, \qquad P_k \in \mathbb{R}^{r \times d}. \tag{B.16}$$

Each group combines associative and direct evidence:

$$r_{t,k} = a_{t,k} + P_k x_t, \tag{B.17}$$

and the complete exposed readout is

$$z_t = \mathrm{Concat}(r_{t,1}, \ldots, r_{t,K}) \in \mathbb{R}^R. \tag{B.18}$$

For every selected target projection,

$$\Delta h_t = s_R D_h z_t, \qquad s_R = \frac{\alpha}{\sqrt{R}}. \tag{B.19}$$

The experiments use $\alpha = 16$. The same z_t is decoded into different target spaces by projection-specific D_h.

READ-BEFORE-WRITE STATE UPDATE

After all reads at position t are formed, the previous state predicts the current memory value:

$$\widehat{v}_t^m = S_{t-1} k_t^m. \tag{B.20}$$

The state is then updated as

$$S_t = \lambda_t S_{t-1} + \beta_t \left(v_t^m - \widehat{v}_t^m\right) \left(k_t^m\right)^\top. \tag{B.21}$$

The scalar gate couples retention and writing through $\lambda_t = 1 - \beta_t$. Because the read uses S_{t-1}, the current token cannot retrieve the association written at the same position.

Algorithm B.2 One-token MQ-ACM forward pass for one attention block.

Input: Token representation x_t; previous state S_{t-1}; learned MQ-ACM parameters
Output: Projection corrections $\{\Delta h_t\}_{h\in\mathcal{H}}$ and updated state S_t

$\widetilde{Q}_t^m \leftarrow \mathrm{Norm}(\tanh(W_q^m x_t))$
$\{q_{t,k}^m\}_{k=1}^K \leftarrow \mathrm{reshape}(\widetilde{Q}_t^m, K, r)$
$k_t^m \leftarrow \mathrm{Norm}(\tanh(W_k^m x_t))$; $v_t^m \leftarrow W_v^m x_t$
$\beta_t \leftarrow \sigma(w_\beta^\top x_t + b_\beta)$; $\lambda_t \leftarrow 1 - \beta_t$
for $k = 1, \ldots, K$ **do**
    $a_{t,k} \leftarrow S_{t-1} q_{t,k}^m$
    $r_{t,k} \leftarrow a_{t,k} + P_k x_t$
end for
$z_t \leftarrow \mathrm{Concat}(r_{t,1}, \ldots, r_{t,K})$
for all $h \in \mathcal{H}$ **do**
    $\Delta h_t \leftarrow s_R D_h z_t$
end for
$S_t \leftarrow \lambda_t S_{t-1} + \beta_t (v_t^m - S_{t-1} k_t^m)(k_t^m)^\top$

Padding positions are masked from the recurrent pathway: their associative reads are set to zero and their update uses $\beta_t = 0$ and $\lambda_t = 1$, so $S_t = S_{t-1}$.

B.4 Training Procedure

The pretrained backbone is frozen. The memory query/key/value projections, scalar gate, direct map, and target decoders are trainable. All query groups are optimized on the same training examples. The dynamic state is a forward-pass runtime activation that evolves through the recurrence.

The training objective is

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda_{\text{div}} \mathcal{L}_{\text{div}}, \tag{B.22}$$

where

$$\mathcal{L}_{\text{div}} = \mathbb{E}_{t\in\mathcal{V}} \left[\frac{1}{K(K-1)} \sum_{i\neq j} \cos^2(q_{t,i}^m, q_{t,j}^m) \right] \tag{B.23}$$

for $K > 1$, and $\mathcal{L}_{\text{div}} = 0$ for $K = 1$. Here $\mathcal{V}$ contains valid, non-padding tokens. The experiments use

$$\lambda_{\text{div}} = 10^{-4}. \tag{B.24}$$

The cosine in Equation (B.23) uses the stabilized convention of § A.10, including zero-norm queries. The regularizer acts on the local queries after the full Kr-dimensional vector has been normalized and reshaped; state reads and decoded corrections are evaluated separately in the diagnostics.

A zero state is created at every training forward call, so each mini-batch begins a fresh state trajectory. Sequence packing is disabled in the natural-language generation experiments, keeping unrelated examples in separate trajectories.

The NLG experiments use query dropout 0; the commonsense experiments use query dropout 0.05. Dropout is disabled in evaluation mode.

Algorithm B.3 MQ-ACM fine-tuning.

Input: Frozen backbone; initialized MQ-ACM parameters; training loader; diversity weight λ_{div}

for all optimizer steps **do**
  Clear parameter gradients
  for all gradient-accumulation micro-batches **do**
    Reset every layer state to zero
    Run Algorithm B.2 over the sequence
    Compute $\mathcal{L}_{\text{task}} + \lambda_{\text{div}}\mathcal{L}_{\text{div}}$
    Backpropagate the scaled micro-batch loss
  end for
  Clip gradients and update only MQ-ACM parameters
  Advance the learning-rate scheduler
end for

B.5 Autoregressive Inference

At inference, learned parameters are fixed and only the state evolves. No backward pass, optimizer step, or test-time parameter update is used.

For each new prompt or evaluation example, every layer begins from a zero state. The prompt is processed in causal order, and the resulting state is retained during generation. Prompt tokens and generated tokens therefore form one continuous state trajectory. The state is reset between independent GSM8K problems, HumanEval tasks, MT-Bench questions, commonsense examples, and unrelated conversations.

MQ-ACM operates alongside the frozen Transformer's ordinary KV cache. The KV cache stores explicit attention keys and values, whereas the MQ-ACM state stores an auxiliary associative summary. Both structures persist during generation.

When beam search is used, each beam maintains its own MQ-ACM state. Beam forking and reordering apply the same beam indices to the state tensors.

B.6 Numerical Precision, Checkpoints, and State Storage

The associative state is maintained in FP32 even when the surrounding model uses BF16 or FP16 mixed precision. State prediction, retention, and the delta outer product are therefore computed in FP32.

MQ-ACM checkpoints store learned adapter tensors and configuration metadata. Dynamic sequence states, transient corrections, and the backbone KV cache are runtime objects. Loading a checkpoint therefore initializes a fresh zero state for a new sample.

For L adapted layers, B_{state} active sequences or beams, local width r, and b_{state} bytes per state element, logical recurrent-state storage is

$$M_{\text{state}} = LB_{\text{state}}r^2 b_{\text{state}}. \tag{B.25}$$

With $L = 32$, $B_{\text{state}} = 1$, $r = 8$, and FP32 state,

$$M_{\text{state}} = 32 \cdot 1 \cdot 8^2 \cdot 4 = 8192 \text{ bytes} = 8 \text{ KiB}. \tag{B.26}$$

This storage is independent of K, because all queries share one state.

B.7 Parameter and Computational Scaling

Let

$$D_{\mathcal{H}} = \sum_{h \in \mathcal{H}} d_h.$$

With the direct path enabled, the active MQ-ACM parameters per adapted layer are

$$N_{\text{layer}} = (2d + D_{\mathcal{H}})R + 2dr + d + 1. \tag{B.27}$$

The terms correspond to the full memory-query projection, direct map, target decoders, shared memory key/value projections, and scalar gate. Disabling the direct path removes dR parameters per layer. For standard LoRA of rank r_L over the same target set $\mathcal{H}$, the per-layer parameter count is

$$N_{\text{LoRA,layer}} = r_L \left(|\mathcal{H}|d + D_{\mathcal{H}}\right). \tag{B.28}$$

For the reported NLG configuration, $d = 4096$, $D_{\mathcal{H}} = 10240$, $r = 8$, and 32 layers give 6,946,848 trainable parameters for $K = 1$ and 21,102,624 for $K = 4$, as reported in Table 4. In the commonsense Q/K/V setting, $D_{\mathcal{H}} = 6144$; Equation B.27 gives 16.91M parameters for MQ-ACM with $K = 4, r = 8, R = 32$, while Equation (B.28) gives 18.87M for rank-32 LoRA. The difference follows from the two parameterizations and their sharing structure.

Ignoring elementwise nonlinearities and constant factors, the additional per-token computation per adapted layer is

$$O\big((2d + D_{\mathcal{H}})R + 2dr + Rr + r^2\big). \tag{B.29}$$

The recurrent-state storage remains independent of K, while the query, direct, and decoding terms grow with $R = Kr$. Additional reads therefore leave the stored state fixed but increase trainable parameters and generation computation. MQ-ACM also retains the asymptotic cost and storage of the frozen full-attention backbone, including the standard KV cache.

C Experimental Details

Following the organization commonly used in recent LoRA studies, we group the settings by benchmark family and list method-specific hyperparameters separately. Data, training duration, target modules, and evaluation are matched within each benchmark family. All reported results are averaged over training seeds 0, 1, and 2 and are presented as mean $\pm$ sample standard deviation.

Benchmark rationale. We select benchmarks with qualitatively different correction demands: MT-Bench (Zheng et al., 2023) measures open-ended conversational instruction following, GSM8K (Cobbe et al., 2021) requires exact-answer multi-step reasoning, HumanEval (Chen et al., 2021) tests executable code synthesis by functional correctness, and the commonsense suite spans reading comprehension, physical and social commonsense, event completion, coreference, and science QA. Together, they test whether MQ-ACM's gains persist across conversation, mathematics, code, and commonsense reasoning, as well as judge-based, exact-match, execution-based, and accuracy-based evaluation protocols.

C.1 Natural-Language Generation

Models and Datasets

We fine-tune Llama-3.1-8B-Base on three representative generation settings:

1. **Mathematical reasoning:** MetaMathQA is used for fine-tuning and GSM8K for evaluation. We report exact-answer accuracy.
2. **Code generation:** Code-Feedback with code-only supervision is used for fine-tuning and HumanEval for evaluation. We report pass@1.
3. **Conversational instruction following:** WizardLM is used for fine-tuning and MT-Bench for evaluation. We report the GPT-4o first-turn score.

Implementation Details

All models are trained for one epoch. The peak learning rate is 5×10^{-5} for full fine-tuning and all low-rank methods except AdaLoRA, for which it is 5×10^{-4}. Unless a method reallocates rank internally, the reference LoRA rank is 8 and the scaling factor is 16. All low-rank methods adapt the Q/K/V/O projections.

Table C.1: Training configuration for natural-language generation.

Setting	Value
Optimizer	AdamW
Adam coefficients	$(0.9, 0.999)$
Weight decay	5×10^{-4}
Effective batch size	64
Learning-rate schedule	Cosine, final ratio 0.1
Warm-up ratio	0.03
Maximum sequence length	1,024
Precision	BF16
Gradient clipping	1.0
Training runtime	DeepSpeed ZeRO-2

MQ-ACM Configurations

The main comparison fixes local width $r = 8$ and evaluates $K \in \{1, 2, 4, 8\}$, giving aggregate widths $R \in \{8, 16, 32, 64\}$. The structural control fixes $R = 32$ and compares $(K, r) \in \{(1, 32), (2, 16), (4, 8), (8, 4)\}$. The latter comparison changes local state width and parameterization and is therefore interpreted as a factorization control rather than an isolated query-count ablation.

Table C.2: Shared MQ-ACM hyperparameters.

Setting	Value
Correction scale	$\alpha = 16$, $s_R = \alpha/\sqrt{R}$
Direct path	Enabled
Calibration batches	32
Joint-initialization scale	10^{-3}
Randomized-SVD oversampling / iterations	16 / 2
Query-diversity coefficient	10^{-4}
Query dropout	0
State precision	FP32

The state is reset for every training forward call and every independent evaluation example. During autoregressive inference, it remains continuous from prompt prefill through generated tokens.

C.2 Commonsense Reasoning

Models and Datasets

We fine-tune Llama-3-8B and evaluate BoolQ, PIQA, SIQA, HellaSwag, WinoGrande, ARC-Challenge, ARC-Easy, and OpenBookQA following the LLM-Adapters protocol. We report task accuracy and the unweighted macro-average over the eight benchmarks. All methods adapt Q/K/V projections.

Implementation Details

Table C.3: Training configuration for commonsense reasoning.

Setting	Value
Epochs	1
Optimizer	AdamW
Learning rate	10^{-4}
Warm-up steps	100
Effective batch size	16
Maximum sequence length	256
Precision	FP16
Scheduler	Linear
Activation checkpointing	Enabled
Training runtime	DeepSpeed ZeRO-2

The compared operating points are LoRA with $(r, \alpha) = (8, 16)$, rank-32 LoRA with $(r, \alpha) = (32, 64)$, and MQ-ACM with $(K, r, R, \alpha) = (4, 8, 32, 16)$. All use dropout 0.05; MQ-ACM additionally uses 32 calibration batches and diversity coefficient 10^{-4}.

The two benchmark families demonstrate compatibility with both Q/K/V/O and Q/K/V target sets. They use different backbones, tasks, and training recipes and are therefore reported as separate evaluation settings.

C.3 Baseline Hyperparameters

For GSM8K and HumanEval, published baseline values follow the matched Llama-3.1-8B protocols used by GoRA and Gradient Intrinsic Dimensionality Alignment. MT-Bench baselines and all MQ-ACM results are evaluated in this work. Method-specific settings follow the recommendations of the corresponding papers whenever applicable.

Table C.4: Method-specific baseline settings.

Method	Configuration
LoRA	Rank 8, scale 16, Kaiming down-projection and zero up-projection
LoRA+	Learning-rate ratio $\eta_B/\eta_A = 16$
PiSSA	Fast SVD with 64 iterations
AdaLoRA	Initial rank 12, target rank 8, $t_i = 150$, $t_f = 900$
MELoRA	Two structured mini-adapters
LoRA-GA	Stable scaling with $\gamma = 16$ and pretrained-weight compensation during gradient-based initialization
MoRA	Rotation-based compression and reconstruction under the rank-8-equivalent parameter budget
GoRA	Reference rank 8, $r_{\min} = 4$, $r_{\max} = 32$, source gradient-based initialization
RaLoRA	Reference rank 8 with $n_{\max} \in \{16, 32, 64\}$ searched as in the source protocol

rsLoRA, DoRA, and OLoRA use their recommended rank-8 implementations under the common data and training protocol.

C.4 Training Environment

The implementation uses Python 3.9 and PyTorch 2.8, with Hugging Face Transformers 4.x and DeepSpeed 0.15. Training uses a single GPU process with ZeRO-2. The dedicated efficiency measurements in the main paper are performed on an NVIDIA H100 PCIe GPU in BF16.

D MECHANISTIC DIAGNOSTIC PROTOCOLS

This appendix specifies the procedures used to produce Figures 3–6 and the supplementary analyses in § E, separating learned structure from the functional use of historical and query-specific corrections. Unless otherwise stated, every diagnostic uses a final trained checkpoint in evaluation mode, performs no optimizer step, and reports the mean ± sample standard deviation over seeds 0, 1, and 2.

D.1 DIAGNOSTIC OVERVIEW

Diagnostic	Configuration	Analysis unit	Quantity
Effective correction rank	Fixed-R sweep	Valid token, layer, Q/K/V/O	Entropy effective rank of decoded total correction
Held-out gradient coverage	Joint-init vs. zero-start, $K = 4, r = 8$	Layer–projection cell	Input, output, and bilateral coverage
Associative-path scaling	Full $K = 4, r = 8$ model	Teacher-forced tokens and generated examples	NLL and task score
Query-difference scaling	Full $K = 4, r = 8$ model	Teacher-forced tokens	NLL
Query-pair similarity	Full $K = 4, r = 8$ model	Query pair, token, layer, projection	Signed and absolute cosine before and after decoding

The fixed-R analysis compares alternative factorizations of an aggregate interface width $R = 32$; local state width, state storage, parameter count, and diversity geometry change jointly across these factorizations. The intervention analyses instead hold one trained architecture fixed and modify one decoded term at inference time.

D.2 EFFECTIVE CORRECTION RANK

CONFIGURATIONS AND TRAJECTORIES

We evaluate

$$(K, r) \in \{(1, 32), (2, 16), (4, 8), (8, 4)\}, \qquad R = Kr = 32.$$

For GSM8K, the first 256 official test prompts are processed in dataset order. For HumanEval, all 164 official prompts are processed. The maximum analyzed length is 768 tokens. Only valid question or code-prompt tokens are retained; template wrappers, special tokens, and padding are excluded. State is reset for each prompt and evolves over the valid sequence.

COLLECTED CORRECTION

For token/history observation i, layer ℓ, and target projection h, the diagnostic records

$$\delta^{\mathrm{tot}}_{i,\ell,h} = \delta^{\mathrm{assoc}}_{i,\ell,h} + \delta^{\mathrm{dir}}_{i,\ell,h} = s_R D_{\ell,h} z_i. \tag{D.1}$$

The reported statistic is therefore computed from the complete decoded correction, including both associative and direct contributions.

For one layer–projection cell with n retained observations,

$$\bar{\delta}_{\ell,h} = \frac{1}{n} \sum_{i=1}^{n} \delta^{\mathrm{tot}}_{i,\ell,h}, \tag{D.2}$$

$$\widehat{\Sigma}_{\Delta,\ell,h} = \frac{1}{n-1} \sum_{i=1}^{n} (\delta^{\mathrm{tot}}_{i,\ell,h} - \bar{\delta}_{\ell,h})(\delta^{\mathrm{tot}}_{i,\ell,h} - \bar{\delta}_{\ell,h})^{\top}. \tag{D.3}$$

Count, sum, and cross-product statistics are accumulated on CPU in float64. The covariance is symmetrized before eigendecomposition, and small negative eigenvalues caused by numerical error are clamped to zero.

Let $\lambda_1, \ldots, \lambda_m$ be the positive eigenvalues and

$$p_j = \frac{\lambda_j}{\sum_a \lambda_a}.$$

The effective rank is

$$r_{\mathrm{eff},\ell,h} = \exp\left(-\sum_{j=1}^{m} p_j \log p_j \right). \tag{D.4}$$

A floor of 10^{-12} is used only when evaluating $p_j \log p_j$. If the covariance is zero, effective rank is defined as zero.

AGGREGATION

We first compute one effective rank for every layer and projection, then macro-average Q/K/V/O within each layer, macro-average the 32 layers, and finally aggregate the three seeds. Each layer–projection cell receives equal weight.

The statistic is distribution-dependent and descriptive. We compute it offline from trained checkpoints and compare it with task scores empirically.

D.3 HELD-OUT GRADIENT-SUBSPACE COVERAGE

COMPARED CHECKPOINTS AND DATA

We compare full MQ-ACM checkpoints with joint initialization to otherwise matched zero-start checkpoints. Both use $K = 4, r = 8$, the direct path, Q/K/V/O adaptation, and the same task training protocol.

For each task, variant, and seed, the diagnostic uses a fixed ordered subset of 128 downstream evaluation examples. These examples are outside the MetaMathQA and Code-Feedback fine-tuning sets. Target-projection gradients are accumulated in FP32 without updating model parameters.

SUBSPACES AND COVERAGE

For a held-out gradient matrix

$$G^{\mathrm{ho}}_{\ell,h} \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}},$$

define

$$V_\ell = \mathrm{orth}(P_\ell^\top), \qquad U_{\ell,h} = \mathrm{orth}(D_{\ell,h}). \tag{D.5}$$

Here, V_ℓ is shared across the Q/K/V/O projections within layer ℓ, whereas $U_{\ell,h}$ is specific to projection h. Orthonormal bases are computed by SVD using the relative numerical-rank threshold

$$\sigma_j > 10^{-6}\sigma_{\max}. \tag{D.6}$$

The three coverage measures are

$$C^{\ell,h}_{\mathrm{in}} = \frac{\|G^{\mathrm{ho}}_{\ell,h} V_\ell\|_F^2}{\|G^{\mathrm{ho}}_{\ell,h}\|_F^2}, \tag{D.7}$$

$$C_{\text{out}}^{\ell,h} = \frac{\|U_{\ell,h}^{\top} G_{\ell,h}^{\text{ho}}\|_F^2}{\|G_{\ell,h}^{\text{ho}}\|_F^2}, \tag{D.8}$$

$$C_{\text{bi}}^{\ell,h} = \frac{\|U_{\ell,h}^{\top} G_{\ell,h}^{\text{ho}} V_{\ell}\|_F^2}{\|G_{\ell,h}^{\text{ho}}\|_F^2}. \tag{D.9}$$

C_{in} measures the gradient energy visible through the learned input coordinates, C_{out} measures the gradient energy contained in the projection-specific decoder output span, and C_{bi} measures the gradient energy jointly represented by the shared input and projection-specific output subspaces.

For each run, the 128 cells formed by 32 layers and the Q/K/V/O projections are macro-averaged. The resulting three run-level values are summarized by their mean and sample standard deviation. Layer-wise curves and layer–projection heatmaps use the same cell-level statistics.

Because only final checkpoints are analyzed, this experiment measures post-training gradient-subspace alignment or coverage, rather than the instantaneous effect of initialization before optimization.

D.4 Shared Intervention Design

The interventions test whether a learned interface uses the historical and query-specific contributions that motivate its design. Both operate on fixed trained checkpoints. The model remains in evaluation mode and receives no parameter updates. For each scaling condition, the complete forward pass is recomputed on the same teacher-forced token sequence. Only the multiplier applied to the selected decoded term is changed; the architecture, learned parameters, and state-update rule remain fixed. Because the modified correction changes downstream hidden activations, the numerical state trajectory may differ across scaling conditions. Under free generation, each condition induces an independent autoregressive rollout, so generated tokens, hidden activations, and states may also diverge.

The implementation verifies that $\lambda = 1$ reproduces the unmodified logits within numerical tolerance. These interventions operate on a jointly trained checkpoint at inference time; separately retrained component controls are reported in the main-paper ablation table.

D.5 Associative-Path Scaling

The full correction is decomposed as

$$\Delta h_t = \Delta h_t^{\text{dir}} + \Delta h_t^{\text{assoc}}. \tag{D.10}$$

The intervention uses

$$\Delta h_t(\lambda_{\text{assoc}}) = \Delta h_t + (\lambda_{\text{assoc}} - 1)\Delta h_t^{\text{assoc}}. \tag{D.11}$$

Thus, $\lambda_{\text{assoc}} = 0$ removes the decoded associative contribution, $\lambda_{\text{assoc}} = 1$ recovers the trained model, and values above one amplify the same contribution.

Teacher-forced NLL

The NLL sweep uses

$$\lambda_{\text{assoc}} \in \{0, 0.25, 0.5, 0.75, 1, 1.25\}.$$

For each seed and task, 128 complete prompt–target trajectories are evaluated with batch size 1 and maximum length 1,024. Loss is computed only on target tokens; prompt, wrapper, and padding tokens are excluded.

For GSM8K, the target is the official answer continuation. For HumanEval, the trajectory contains the official prompt followed by the canonical solution inside the HumanEval instruction template.

For plotting, the difference is computed within each seed relative to $\lambda_{\text{assoc}} = 0$:

$$\Delta\text{NLL}_s(\lambda) = \text{NLL}_s(\lambda) - \text{NLL}_s(0). \tag{D.12}$$

FREE-GENERATION SCORE

The generation control uses

$$\lambda_{\text{assoc}} \in \{0, 0.5, 1, 1.25\}.$$

GSM8K is evaluated on all 1,319 test examples and HumanEval is evaluated on all 164 tasks with the same generation protocol as the main experiment.

Teacher-forced NLL and free-generation score are reported separately because a lower reference-trajectory NLL need not produce a monotonic improvement in autoregressive task success.

D.6 QUERY-SPECIFIC DIFFERENTIAL SCALING

For associative reads $a_{t,k}$, define

$$\bar{a}_t = \frac{1}{K}\sum_{k=1}^{K} a_{t,k}, \qquad \epsilon_{t,k} = a_{t,k} - \bar{a}_t. \tag{D.13}$$

The decoded associative correction is separated into common and query-specific terms:

$$\Delta h_{t,h}^{\text{assoc}} = s_R \sum_{k=1}^{K} D_{h,k}\bar{a}_t + s_R \sum_{k=1}^{K} D_{h,k}\epsilon_{t,k}. \tag{D.14}$$

Let the second term be $\Delta h_{t,h}^{\text{diff}}$. The intervention is

$$\Delta h_{t,h}(\lambda_{\text{diff}}) = \Delta h_{t,h} + (\lambda_{\text{diff}} - 1)\Delta h_{t,h}^{\text{diff}}. \tag{D.15}$$

By linearity, this is equivalent to replacing

$$a_{t,k} \longmapsto \bar{a}_t + \lambda_{\text{diff}}\epsilon_{t,k} \tag{D.16}$$

before group-specific decoding. At $\lambda_{\text{diff}} = 0$, the direct path and common associative term remain, while the decoded query-specific deviation is removed.

The NLL sweep uses

$$\lambda_{\text{diff}} \in \{0, 0.25, 0.5, 0.75, 1, 1.25\}$$

and follows the same examples, target masks, state lifecycle, and aggregation as the teacher-forced NLL protocol in § D.5. This intervention asks whether learned query differences contribute beyond the common read within the jointly trained interface. Table 3 instead reports separately retrained initialization and direct-path removals.

D.7 Query-Pair Similarity Before and After Decoding

For every valid token, layer, projection, and unordered query pair (k, k'), the raw-read cosine is

$$\cos^{\text{raw}}_{k,k'} = \frac{a_{t,k}^{\top} a_{t,k'}}{\max(\|a_{t,k}\|_2, 10^{-6}) \max(\|a_{t,k'}\|_2, 10^{-6})}. \tag{D.17}$$

The decoded group contribution is

$$c_{t,h,k} = D_{h,k} a_{t,k}, \tag{D.18}$$

with cosine

$$\cos^{\text{dec}}_{k,k'} = \frac{c_{t,h,k}^{\top} c_{t,h,k'}}{\max(\|c_{t,h,k}\|_2 \|c_{t,h,k'}\|_2, 10^{-12})}. \tag{D.19}$$

Padding is excluded, and zero-norm pairs are assigned cosine zero. Statistics are first computed within each layer–projection cell, then macro-averaged over layers and Q/K/V/O, and finally aggregated over seeds.

Every decoder block uses a dense parameterization over the full target coordinate system. Low decoded cosine therefore reflects learned decoder orientation.

Cosine similarity describes geometric relatedness before and after decoding. We interpret it together with the differential-scaling intervention when assessing output differentiation.

D.8 Scope of the Diagnostics

The diagnostics cover three levels of trained-model behavior: realized correction variation, post-training input/output coordinate alignment, and functional use of associative and query-specific decoded terms. Effective rank and coverage are analyzed as checkpoint statistics, the fixed-R sweep as an interface-factorization control, and the scaling experiments as interventions within fixed trained checkpoints. Separately retrained component removals are reported in Table 3.

E Supplementary Experimental Results

This appendix reports the additional aggregate results needed to support the main-paper conclusions. All values are arithmetic mean ± sample standard deviation over three independent training seeds. The tables and figures report aggregate statistics; the diagnostic procedures are specified in § D.

E.1 Complete Commonsense Deviations

Table E.1 complements Table 2 in the main paper by reporting the complete task-level means and standard deviations for the eight commonsense benchmarks. The macro-average is already reported in the main paper and is therefore omitted here.

Table E.1: Commonsense accuracy (mean ± sample std., three seeds).

Task	LoRA ($r = 8$)	LoRA ($r = 32$)	MQ-ACM ($K = 4, r = 8$)
BoolQ	71.58 ± 0.92	73.49 ± 0.32	73.03 ± 0.46
PIQA	86.38 ± 0.21	87.83 ± 0.25	87.96 ± 0.08
SIQA	78.27 ± 0.59	79.63 ± 0.41	80.14 ± 0.18
HellaSwag	93.11 ± 0.05	92.71 ± 3.16	94.77 ± 0.21
WinoGrande	82.72 ± 0.57	85.32 ± 0.63	86.85 ± 0.18
ARC-C	77.84 ± 0.20	78.61 ± 0.38	78.58 ± 0.48
ARC-E	89.77 ± 0.39	90.10 ± 0.17	89.94 ± 0.73
OBQA	81.80 ± 0.40	84.60 ± 1.31	84.87 ± 0.99

As reported in Table 2 of the main paper, MQ-ACM improves the macro-average by 1.84 points over rank-8 LoRA and by 0.48 points over rank-32 LoRA. At the task level, it obtains the highest mean on PIQA, SIQA, HellaSwag, WinoGrande, and OpenBookQA. Rank-32 LoRA shows relatively high HellaSwag variance because one run is substantially lower than the other two; the run is retained in the aggregate.

Only the formal official commonsense protocol is included. A separate exploratory configuration attains a higher average but differs in supervision, sequence length, precision, effective batch size, and calibration settings, and is therefore excluded from all formal comparisons.

E.2 Fixed-R Structural Control

We fix the aggregate interface width at

$$R = Kr = 32$$

and compare

$$(K, r) \in \{(1, 32), (2, 16), (4, 8), (8, 4)\}.$$

This control fixes the exposed width while local state width, state storage, trainable parameter count, and diversity geometry vary with the factorization. We therefore interpret it as a comparison of interface factorizations.

Table E.2: Performance and correction structure at $R = 32$.

Task	(K, r)	Task score	Effective rank	Raw read $\lvert\cos\rvert$
GSM8K	$(1, 32)$	72.91 ± 0.32	13.18 ± 0.04	–
GSM8K	$(2, 16)$	73.87 ± 0.82	13.03 ± 0.04	0.941 ± 0.002
GSM8K	$(4, 8)$	73.74 ± 0.34	13.16 ± 0.05	0.908 ± 0.006
GSM8K	$(8, 4)$	73.59 ± 1.19	13.58 ± 0.06	0.904 ± 0.005
HumanEval	$(1, 32)$	47.76 ± 0.35	16.82 ± 0.01	–
HumanEval	$(2, 16)$	46.34 ± 1.22	16.50 ± 0.04	0.947 ± 0.002
HumanEval	$(4, 8)$	49.39 ± 1.22	16.51 ± 0.06	0.929 ± 0.004
HumanEval	$(8, 4)$	47.56 ± 1.22	16.94 ± 0.04	0.929 ± 0.001

At the default $(K, r) = (4, 8)$,

$$r = 8 < r_{\text{eff}} < R = 32.$$

The entropy-based r_{eff} is generally noninteger and is macro-averaged over layer–projection cells in Table E.2. At the default factorization, the effective number of output-variation directions exceeds the local width while remaining substantially below the aggregate width. Effective rank changes only modestly across the four factorizations while task scores vary differently, supporting the allocation–expression distinction. The tested settings therefore show task-dependent relationships among K, local grouping, effective rank, and downstream score.

E.3 Full Gradient-Subspace Coverage Analysis

The main paper reports input and bilateral gradient coverage. Table E.3 adds the output-only control.

Table E.3: Gradient coverage (%), macro-averaged over 32 layers and Q/K/V/O. Mean ± sample std., three seeds.

Task	Variant	Input coverage C_{in}	Output coverage C_{out}	Bilateral coverage C_{bi}
GSM8K	Joint initialization	29.745 ± 0.154	9.772 ± 0.131	2.256 ± 0.036
GSM8K	No joint initialization	7.993 ± 0.324	12.125 ± 0.361	1.236 ± 0.100
HumanEval	Joint initialization	24.128 ± 0.358	7.035 ± 0.071	1.364 ± 0.019
HumanEval	No joint initialization	4.961 ± 0.121	9.521 ± 0.170	0.567 ± 0.030

Joint initialization yields substantially higher input and bilateral coverage on both tasks, while output-only coverage is lower. This combined pattern is consistent with coordinated input–output geometry at the trained checkpoints.

LAYER-WISE BILATERAL COVERAGE

Figure 5 in the main paper shows that joint initialization remains above the no-joint condition through most of the network depth on both tasks, although the gain is not uniform across layers.

LAYER–PROJECTION COVERAGE MAPS

Figure 6 in the main paper shows that the gains are distributed across multiple layers and projections rather than being generated by one isolated cell. Some cells are negative, so the effect is not uniformly positive for every projection in every layer.

All coverage statistics are measured at trained final checkpoints. The analysis therefore characterizes the gradient-subspace alignment signature retained after training.

E.4 ASSOCIATIVE AND QUERY-DIFFERENTIAL INTERVENTIONS

The interventions operate on fixed trained checkpoints with learned parameters, the state-update rule, direct path, decoders, and frozen backbone held fixed. For each scaling condition, the complete forward computation is rerun. Because modified corrections change downstream hidden activations, subsequent queries and numerical state trajectories may differ across conditions.

ASSOCIATIVE-PATH SCALING

The decoded correction is modified as

$$\Delta h'_t = \Delta h_t^{\text{full}} + (\lambda_{\text{assoc}} - 1)\Delta h_t^{\text{assoc}}.$$

Table E.4: Associative-term scaling: teacher-forced NLL and free-generation task scores.

Task	λ_{assoc}	Teacher-forced NLL	Task score
GSM8K	0.00	0.8933 ± 0.0041	71.22 ± 0.50
GSM8K	0.50	0.8723 ± 0.0028	73.16 ± 0.80
GSM8K	1.00	0.8529 ± 0.0018	73.74 ± 0.34
GSM8K	1.25	0.8434 ± 0.0013	73.64 ± 0.29
HumanEval	0.00	0.3423 ± 0.0033	47.97 ± 0.70
HumanEval	0.50	0.3373 ± 0.0014	48.37 ± 1.86
HumanEval	1.00	0.3336 ± 0.0015	49.39 ± 1.22
HumanEval	1.25	0.3321 ± 0.0022	47.76 ± 0.70

Restoring the learned associative contribution improves both NLL and mean task score relative to $\lambda_{\text{assoc}} = 0$. Amplifying it to 1.25 further improves teacher-forced NLL but not free-generation performance, showing that reference-token likelihood and autoregressive task success are related but not interchangeable.

QUERY-SPECIFIC DIFFERENTIAL SCALING

Let

$$a_{t,k} = \bar{a}_t + \epsilon_{t,k}.$$

The intervention scales only the decoded query-specific differential term:

$$\Delta h'_t = \Delta h_t^{\text{full}} + (\lambda_{\text{diff}} - 1)\Delta h_t^{\text{diff}}.$$

Table E.5: Query-specific differential scaling: teacher-forced NLL.

Task	λ_{diff}	Teacher-forced NLL
GSM8K	0.00	0.8815 ± 0.0035
GSM8K	0.50	0.8669 ± 0.0023
GSM8K	1.00	0.8529 ± 0.0018
GSM8K	1.25	0.8458 ± 0.0022
HumanEval	0.00	0.3400 ± 0.0035
HumanEval	0.50	0.3364 ± 0.0020
HumanEval	1.00	0.3336 ± 0.0015
HumanEval	1.25	0.3325 ± 0.0016

Every tested positive differential scale lowers NLL relative to the common-only condition, supporting functional use of decoded group-specific deviations within the tested range.

E.5 RAW-READ AND DECODED-CORRECTION SIMILARITY

Table E.6: Pairwise cosine statistics for the $K = 4$ associative groups.

Task	Space	Signed cosine	Absolute cosine
GSM8K	Raw associative read	0.035 ± 0.018	0.914 ± 0.006
GSM8K	Decoded correction	0.008 ± 0.001	0.085 ± 0.002
HumanEval	Raw associative read	0.028 ± 0.029	0.934 ± 0.003
HumanEval	Decoded correction	0.003 ± 0.000	0.065 ± 0.001

The raw reads have high absolute cosine but near-zero signed cosine, indicating a mixture of strongly aligned and strongly anti-aligned pairs rather than simple positive duplication. After group-specific decoding, absolute cosine falls substantially. This is consistent with the common–differential decomposition in § A: related reads can still be mapped into distinct output directions when decoder blocks differ.

Together with the differential-scaling intervention, these statistics support functional use of learned query differences after decoding. They characterize data-driven organization in the trained interface without identifying semantic partitions. Separately retrained decoder-sharing alternatives remain a distinct experimental question.

E.6 SUMMARY OF DIAGNOSTIC EVIDENCE

The supplementary results address distinct parts of the organization question. Effective rank characterizes realized output variation within allocated width R; cosine statistics describe how decoding changes read geometry. Scaling interventions test functional use of associative and query-specific terms in trained models. Gradient coverage characterizes post-training alignment, whose task value is assessed separately by retrained ablations. These measurements support learned differentiation without identifying semantic partitions or implying that greater width, effective rank, or alignment guarantees higher task scores.