LIVE-MLA: LATENT INFERENCE WITH VALUE ERROR FEEDBACK FOR MULTI-HEAD ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Most pretrained models use Multi-Head Attention (MHA) or Grouped-Query Attention (GQA). Recent methods convert these models to Multi-Head Latent Attention (MLA) through low-rank decomposition, reducing KV-cache storage without training comparable MLA models from scratch. The resulting low-dimensional bottleneck leaves reconstruction residuals that fixed low-rank mappings cannot adapt to during inference. We consider both how to correct these residuals during decoding and how to account for their effect when fitting the low-rank mappings. First, when a token leaves a small exact recent region, its latent representation and value residual are available together and can guide corrections for older compressed tokens. Second, KV reconstruction error alone is insufficient to assess compression, since keys and values matter through the attention outputs they produce. These observations lead to LIVE-MLA, which combines Residual Feedback with attention-weighted conversion. Residual Feedback fits sequence-specific value corrections from the observed pairs in closed form. Attention-weighted conversion weights each token's joint key-value reconstruction objective by the attention mass it receives. All persistent states are included in the same fixed cache budget. Experiments show that LIVE-MLA narrows the perplexity gap left by training-free conversion. Shorter recovery training further narrows the gap to the original model while using $4\times$ to $60\times$ fewer recovery-training tokens than the evaluated baseline adaptations.

1 INTRODUCTION

Large language models cache the keys and values of previous tokens during autoregressive decoding. Because this KV cache grows linearly with context length, it becomes a major memory constraint in long-context inference (Pope et al., 2023). Multi-Head Latent Attention (MLA) reduces this cost by caching low-dimensional latent representations from which keys and values are reconstructed (Liu et al., 2024a).

Most pretrained models, however, use Multi-Head Attention (MHA) (Vaswani et al., 2017) or Grouped-Query Attention (GQA) (Ainslie et al., 2023). Recent methods therefore convert these models to MLA through low-rank decomposition, avoiding the cost of training comparable MLA models from scratch (Ji et al., 2025; Meng et al., 2026; Zhou et al., 2026). These approaches compress each token's hidden state to a shared low-dimensional latent representation, from which its original key and value representations are approximately reconstructed.

The resulting low-dimensional bottleneck leaves reconstruction residuals $K - \widehat{K}$ and $V - \widehat{V}$. Prior methods reduce these residuals either by factorizing pretrained weights or fitting the low-rank mappings to KV activations collected from calibration data (Ji et al., 2025; Li et al., 2025; Meng et al., 2026; Zhou et al., 2026), but keep these mappings fixed during inference. They therefore cannot adapt to the sequence-specific residuals that arise on the sequence being decoded.

Two empirical observations motivate our approach. First, Figure 1(a) shows that replacing exact KV representations with their low-rank reconstructions causes a measurable drop in token-level next-token accuracy on text datasets. This result confirms that reconstruction residuals affect model predictions and motivates retaining a small region of recent KV representations in exact form.

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.

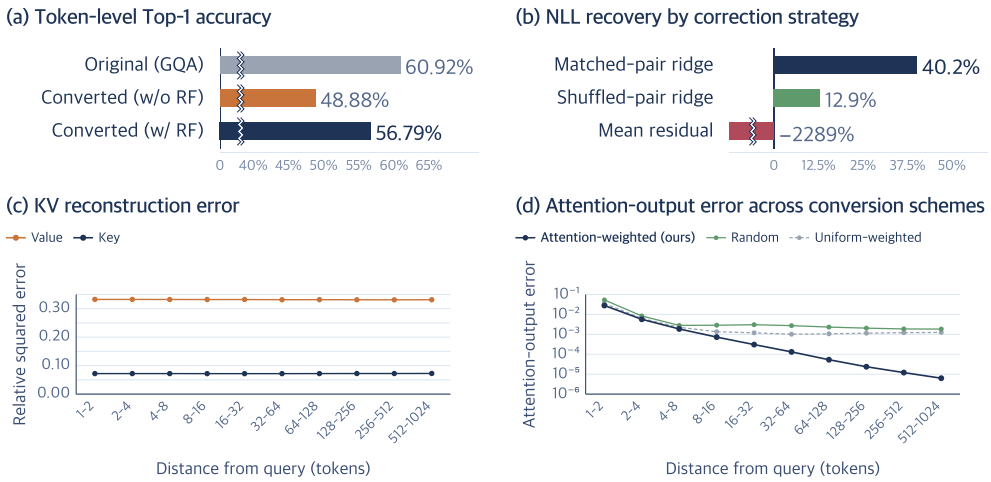

Figure 1: Evidence motivating LIVE-MLA. (a) On WikiText-2, Residual Feedback recovers much of the token-level Top-1 next-token accuracy lost after conversion. (b) On older compressed tokens, corrections learned from matched latent–residual pairs (c_i, e_i) yield greater NLL recovery than those learned from shuffled pairs $(c_i, e_j; i \neq j)$ or the mean residual $\bar{e}$. NLL recovery is the fraction of the NLL reduction obtained by directly applying the measured residuals that is recovered by each estimated correction; negative values indicate degradation relative to no correction. (c) Key and value reconstruction errors change little with source-token distance. (d) Attention-output error decreases with distance under all three conversion schemes, most sharply with attention weighting.

Several KV-cache compression methods already adopt this strategy by retaining a small recent region in exact form (Liu et al., 2024c; Staniszewski & Lancucki, 2026). We observe, however, that this region can serve an additional, previously underused role. Within the region, each token's exact and reconstructed KV representations coexist, making its reconstruction residuals directly observable on the current inference sequence. This raises a natural question. *Why leave these inference-time residuals unused when they can be fed back to correct older compressed tokens?*

To do so, Residual Feedback fits a sequence-specific ridge mapping from latent representations to observed value residuals and applies the predicted residuals to correct older compressed tokens. Figure 1(a) shows that Residual Feedback measurably narrows the token-level accuracy gap introduced by conversion, while Figure 1(b) shows that effective correction depends on preserving the correspondence between latent representations and their residuals.

Our second observation is that KV reconstruction error alone is insufficient to assess the effect of compression, since keys and values ultimately matter through the attention outputs they produce. Figures 1(c) and 1(d) show that recent and distant tokens have similar KV reconstruction errors, whereas attention-output error decreases sharply with distance from the query. This contrast motivates our attention-weighted conversion, which weights each token's joint key-value reconstruction objective by the attention mass it receives.

These observations lead to LIVE-MLA, a training-efficient framework for MHA/GQA-to-MLA conversion that provides strong training-free conversion with attention-weighted objective and inference-time residuals and supports resource-efficient recovery training. Our contributions are threefold.

- **Residual Feedback.** Using paired latent–residual observations from a small exact recent-token region, Residual Feedback fits sequence-specific value corrections in closed form and applies them to older compressed tokens without gradient-based parameter updates.
- **Attention-weighted Conversion.** We weight each token's joint key–value reconstruction objective by the attention mass it receives, prioritizing errors that are more likely to affect subsequent attention outputs.

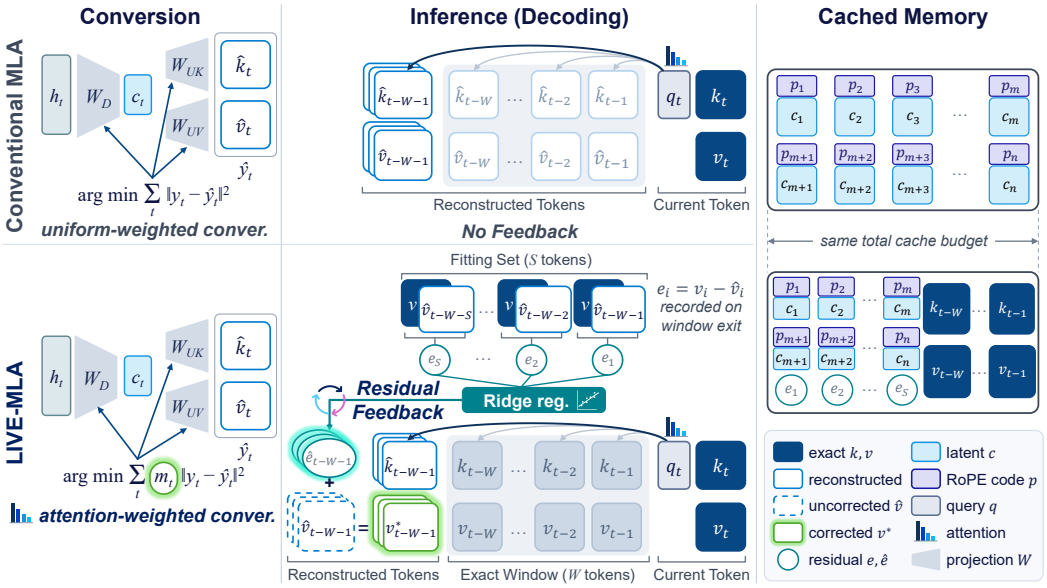

Figure 2: Overview of LIVE-MLA under a fixed total cache budget. Attention-weighted conversion produces the base MLA mappings, while a small exact recent region makes the resulting reconstruction residuals observable. For each query, Residual Feedback fits a sequence-specific value correction from the residuals of the tokens that most recently left the window and applies it to all older compressed tokens. All persistent states are included in the budget.

- **Long-context Performance and Efficient Recovery.** Under matched cache budgets that include all persistent states, LIVE-MLA outperforms or remains competitive with existing conversions on perplexity and long-context benchmarks without recovery training. On Qwen3-8B-Base, its stronger starting point enables optional recovery training to match or exceed baselines on most metrics using 4–60× fewer recovery-training tokens.

2 Preliminaries

Attention. Let $h_t \in \mathbb{R}^{d_{\text{model}}}$ be the hidden state of token t. An attention layer has H query heads and G key-value heads (Vaswani et al., 2017; Ainslie et al., 2023) with key and value widths d_k and d_v, and query head $u \in \{1, \dots, H\}$ uses key-value head $\pi(u) \in \{1, \dots, G\}$. A dimension concatenated across the key-value heads is written as a product (e.g., Gd_k), and for such a vector or matrix the superscript $(\pi(u))$ denotes the block of key-value head $\pi(u)$. Query head u of token t attends to the keys and values of tokens $i \le t$ with attention weights $\alpha_{ti}^{(u)}$ and outputs $o_t^{(u)} = \sum_{i \le t} \alpha_{ti}^{(u)} v_i^{(\pi(u))}$, where $v_i \in \mathbb{R}^{Gd_v}$ is the value of token i. All quantities are defined per layer. Stacking the per-token keys and values of a sequence gives K and V.

Latent representation. Conversion to MLA compresses each token's hidden state to a shared low-dimensional latent representation, from which its key and value are approximately reconstructed. Following partial RoPE (Ji et al., 2025), each key head keeps rotary position embedding (RoPE) (Su et al., 2024) on d_R rotary coordinates, formed by its first $d_R/2$ RoPE pairs, and treats the remaining $d_k - d_R$ as content coordinates. Let $k_t^{\text{C}} \in \mathbb{R}^{G(d_k - d_R)}$ be the content coordinates of the exact key before key normalization and RoPE, and let $v_t \in \mathbb{R}^{Gd_v}$ be the exact value. The converted layer applies the low-rank mappings

$$c_t = h_t W_D, \qquad p_t = h_t W_P, \qquad \hat{k}_t = c_t W_{UK}, \qquad \hat{v}_t = c_t W_{UV},$$

where the latent representation $c_t \in \mathbb{R}^r$ is shared by the content-key and value paths and $\hat{k}_t$ reconstructs k_t^{C}. The RoPE code $p_t \in \mathbb{R}^{d_R}$ is shared across key-value heads, as is the decoupled

rotary key of MLA (Liu et al., 2024a). One complex coefficient per head is obtained by a rank-one fit (Meng et al., 2026) and folded into the query (Appendix B.1). We use $d_R = 64$. The converted layer also keeps the original key and value projections, which produce the exact keys and values of the exact window (Section 3.2).

Cached memory. An older compressed token stores its latent representation c_t, its RoPE code p_t, and, for models that normalize each key head before RoPE, the per-head normalization factors $\gamma_t \in \mathbb{R}^G$ computed from the exact key. Its stored width is $\kappa = r + d_R + d_N$ elements, with $d_N = G$ when γ_t is stored and $d_N = 0$ otherwise. Its key and value are reconstructed from this state. As in MLA (Liu et al., 2024a; Meng et al., 2026), W_{UK} is absorbed into the query, RoPE is applied only to p_t, γ_t rescales the score, and W_{UV} is absorbed into the output projection, so the reconstructed keys and values are never materialized (Appendix B.1). Scores for exact and reconstructed tokens are normalized by one softmax.

3 METHOD

LIVE-MLA builds the base MLA mappings with attention-weighted conversion, then keeps a small exact recent region, the exact window, during decoding. When a token leaves this window, its exact and reconstructed values are both available, so its value residual can be measured directly. Residual Feedback fits a sequence-specific value correction from the residuals of the tokens that most recently left the window and applies it to older compressed tokens. All persistent states are included in the same fixed cache budget.

3.1 BASE CONVERSION AND RESIDUAL OBSERVATION

Exact window and residual observation. Let the exact window retain $W = 15$ preceding tokens with their exact keys and values. Including the current token, at most $W + 1 = 16$ exact tokens participate in attention for each query. Residual Feedback records and corrects only value residuals. Value errors enter the output through their attention weights, while inaccurate key corrections also change the logits and total attention mass. In a diagnostic configuration, value-only correction lowered perplexity on both models, while adding key correction substantially degraded Qwen3-8B-Base (Appendix C.8). Appendix C.7 analyzes this difference.

Indexing tokens from zero, before processing token t let

$$\mathcal{R}_t = \{\, i \mid \max(0, t - W) \le i < t \,\}, \qquad \mathcal{O}_t = \{\, i \mid 0 \le i < t - W \,\}.$$

Tokens in $\mathcal{R}_t$ retain their exact keys and values, while tokens in $\mathcal{O}_t$ retain only their compressed state. Token t uses its exact key and value and enters the cache after its output is computed. Thus, at most $W + 1$ tokens use exact representations for each query.

Once the output for token t is computed, token $i = t - W$ leaves the window for $t \ge W$. Before discarding its exact value, we record its latent representation and value residual as the pair (c_i, e_i), where

$$e_i = v_i - \hat{v}_i.$$

Because this pair is recorded after query t has been processed, Residual Feedback first uses it for query $t + 1$. Residual rows are stored in BF16. Appendix C.8 evaluates this format.

Attention-weighted conversion. The latent representation cannot preserve all information in the original keys and values. Errors on tokens that receive more attention are more likely to affect later outputs, so we weight each key-value reconstruction objective by the attention mass it receives.

Let a index sequences of length T_a, and let $\mathcal{I}_a$ denote the first $T_a/2$ positions used for fitting. The attention weight from query head u at position τ to token t in sequence a is denoted by $\alpha^{(u)}_{a,\tau t}$. This choice leaves later queries for measuring the attention received by every fitted token.

Keys and values can differ in scale, so we choose s such that sk^{C} and v have the same average squared magnitude per coordinate over the calibration targets. From the attention weights of the

pretrained model, we define

$$m_{a,t} \propto \sum_{u=1}^{H} \sum_{\tau=t+W+1}^{T_a-1} \alpha_{a,\tau t}^{(u)}, \qquad \frac{1}{|\mathcal{I}_a|} \sum_{t \in \mathcal{I}_a} m_{a,t} = 1.$$

The inner sum includes only queries that read token t from the compressed cache after it leaves the exact window. Normalization is performed separately for each layer and sequence, so $m_{a,t} = 1$ marks an average calibration token.

Let M denote the joint linear map from the hidden state to the scaled content-key and value representations, so that the target $y_{a,t} = [\, s k_{a,t}^{\mathrm{C}}, v_{a,t} \,]$ is reconstructed as $\hat{y}_{a,t} = h_{a,t} M$. We then solve

$$M^\star = \arg \min_{\mathrm{rank}(M) \le r} \sum_a \sum_{t \in \mathcal{I}_a} m_{a,t} \, \| y_{a,t} - \hat{y}_{a,t} \|_2^2$$

by accumulating the weighted second and cross moments and applying a reduced-rank factorization. Writing the solution as $M^\star = W_D \, [\, s W_{UK}, W_{UV} \,]$ defines W_D, W_{UK}, and W_{UV}, so that $\hat{y}_{a,t} = [\, s\hat{k}_{a,t}, \hat{v}_{a,t} \,]$. The factor s is used only during fitting, so W_{UK} reconstructs the content key at its original scale. Setting $m_{a,t} = 1$ for every token recovers the uniform-weighted conversion shown in Figure 2.

3.2 Residual Feedback

Let $S = 16$ denote the maximum number of latent–residual pairs used for each query. For query t, Residual Feedback (RF) uses the fitting set of tokens that most recently left the window,

$$\mathcal{F}_t = \{\, i \in \mathcal{O}_t \mid i \ge t - W - S \,\}.$$

We append a constant feature to each latent representation, $\phi_i = [\, c_i, 1 \,]$, allowing the fit to include an offset. Let $\Phi_t \in \mathbb{R}^{|\mathcal{F}_t| \times (r+1)}$ and $E_t \in \mathbb{R}^{|\mathcal{F}_t| \times G d_v}$ stack these features and their corresponding residuals. When $\mathcal{F}_t$ is nonempty, we fit a ridge mapping $B \in \mathbb{R}^{(r+1) \times G d_v}$ by

$$B_t^\star = \arg\min_B \, \| E_t - \Phi_t B \|_F^2 + \lambda_t \| B \|_F^2, \qquad \lambda_t = \rho \, \frac{\| \Phi_t \|_F^2}{|\mathcal{F}_t|},$$

where $\rho = 10^{-3}$ is the regularization coefficient. Since $|\mathcal{F}_t| \le S$, we compute $B_t^\star = \Phi_t^\top \big(\Phi_t \Phi_t^\top + \lambda_t I_{|\mathcal{F}_t|} \big)^{-1} E_t$, where $I_{|\mathcal{F}_t|}$ is the identity matrix of size $|\mathcal{F}_t|$. The predicted residual of an older compressed token $j \in \mathcal{O}_t$ is $\hat{e}_j = \phi_j B_t^\star$. Feedback is disabled when $\mathcal{F}_t$ is empty. It first applies at $t = W + 1$, after which the fitting set grows to S rows and slides by one token per query.

Corrected attention output. Attention weights use exact keys for $\mathcal{R}_t \cup \{t\}$ and the compressed-token scores of Section 2 (Appendix B.1) for $\mathcal{O}_t$. Adding the predicted residual to the reconstructed value gives the corrected value $v_j^* = \hat{v}_j + \hat{e}_j$, and the corrected output of query head u is

$$\tilde{o}_t^{(u)} = \sum_{i \in \mathcal{R}_t \cup \{t\}} \alpha_{ti}^{(u)} \, v_i^{(\pi(u))} + \sum_{j \in \mathcal{O}_t} \alpha_{tj}^{(u)} \, v_j^{*(\pi(u))}.$$

Since the ridge mapping is linear, its attention-weighted contribution can be computed as

$$\sum_{j \in \mathcal{O}_t} \alpha_{tj}^{(u)} \, \hat{e}_j^{(\pi(u))} = \Big(\sum_{j \in \mathcal{O}_t} \alpha_{tj}^{(u)} \phi_j \Big) \big(B_t^\star \big)^{(\pi(u))}.$$

This avoids constructing a separate predicted residual for every older token.

3.3 Fixed Cache Budget

All state retained between decoding steps is included in the same fixed cache budget. Let L denote the number of attention layers and ℓ index a layer. We use a common latent width r across layers and define the compressed width as $\kappa_\ell = r + d_R + d_{N,\ell}$. For a context of length $T \ge W + S$, the BF16 cache storage of layer ℓ is $\mathcal{M}_\ell(T) = 2\big(T\kappa_\ell + W G_\ell (d_{k,\ell} + d_{v,\ell}) + S G_\ell d_{v,\ell} \big)$ bytes. The three terms account for the compressed state, exact window, and residual rows. We choose the largest common latent width r for which the total storage $\sum_\ell \mathcal{M}_\ell$ at the reference length $T_{\mathrm{ref}} = 2048$ does not exceed a target fraction β of the original BF16 KV-cache storage, so that $1 - \beta$ is the memory reduction. Appendix B.2 gives the exact constraint, the byte-level accounting, and the resulting latent widths.

Table 1: Training-free conversion. Cache budgets are defined at a 2K reference length. Bold marks the best converted result for each model, memory reduction, and metric.

Method	Mem.	PPL ↓	LongBench-2K ↑	RULER-2K ↑	RULER-4K ↑
Llama-3.1-8B-Instruct					
Original	-	*7.22*	*14.58*	*94.73*	*94.93*
TransMLA	−50%	12.32	12.45	80.15	74.80
MHA2MLA	−50%	1029.8	1.07	0.05	0.06
CARE	−50%	9.64	**14.59**	89.16	86.13
Ours	−50%	**7.62**	14.02	**91.76**	**90.82**
TransMLA	−75%	26.28	7.19	**52.21**	**41.60**
MHA2MLA	−75%	1430.9	1.00	0.00	0.02
CARE	−75%	48.31	7.22	29.25	23.52
Ours	−75%	**10.11**	**11.53**	31.34	23.71
TransMLA	−87.5%	127.9	3.33	0.78	0.49
MHA2MLA	−87.5%	4878.3	1.55	0.00	0.00
CARE	−87.5%	398.3	2.75	0.52	0.53
Ours	−87.5%	**15.29**	**7.84**	**2.80**	**3.05**
Qwen3-8B-Base					
Original	-	*7.00*	*22.63*	*94.39*	*93.69*
TransMLA	−50%	86.62	1.00	2.14	0.70
MHA2MLA	−50%	635.8	1.06	0.03	0.02
CARE	−50%	10.72	15.95	**84.53**	**77.85**
Ours	−50%	**7.59**	**20.48**	84.40	63.95
TransMLA	−75%	1213.6	1.34	0.06	0.09
MHA2MLA	−75%	1729.9	0.82	0.02	0.03
CARE	−75%	20.44	9.13	**37.41**	**27.81**
Ours	−75%	**9.37**	**14.14**	27.79	16.14
TransMLA	−87.5%	2217.4	1.86	0.05	0.06
MHA2MLA	−87.5%	3759.4	1.03	0.00	0.00
CARE	−87.5%	103.5	5.51	**6.39**	**3.55**
Ours	−87.5%	**13.90**	**9.99**	3.59	2.80

3.4 Recovery Training and Inference

Recovery training updates only the attention parameters affected by the conversion, including the query, key, value, and output projections, the query and key normalizers, the low-rank mappings, and the RoPE code projection W_P with its head-specific coefficients, and optimizes the next-token cross-entropy loss of the converted model. Residual Feedback is refitted in each forward pass, but gradients do not pass through the observed pairs or the ridge solve; the fitted ridge mapping is held fixed while gradients flow through its application to older compressed tokens. We compute the query-specific ridge mappings in parallel, with each query using only the pairs available at that position. Inference follows the same procedure, so it uses no future information and does not update the model parameters.

4 Experiments

4.1 Setup

We evaluate Llama-3.1-8B-Instruct (Grattafiori et al., 2024) and Qwen3-8B-Base (Yang et al., 2025). Both models use GQA with eight key-value heads. Qwen3-8B-Base additionally applies RMS normalization to each query and key head before RoPE. We define KV-cache memory reduction as the percentage decrease in persistent decoding-cache storage relative to the original model at a reference length of 2,048 tokens. We evaluate reductions of 50%, 75%, and 87.5% and choose the cache dimensions of each method according to its storage layout. The accounted storage for LIVE-MLA includes the compressed token states, stored normalization values when required, the exact recent-token window, and the Residual Feedback state. Appendices B.2 and B.5 provide the complete cache accounting and configuration details.

Table 2: Recovery-training results on Qwen3-8B-Base. Bold marks the best converted result for each memory-reduction level and column.

Method	Mem.	Tokens	vs. Ours	PPL ↓	LongBench-2K ↑	RULER-2K ↑	RULER-4K ↑
Original	-	-	-	*7.00*	*22.63*	*94.39*	*93.69*
TransMLA	−50%	3B	30×	7.22	19.12	93.32	**89.64**
MHA2MLA	−50%	6B	60×	7.80	17.22	86.70	85.43
Ours	−50%	**100M**	-	**6.99**	**21.55**	**93.90**	88.18
TransMLA	−75%	6B	20×	7.39	15.67	92.41	**89.20**
MHA2MLA	−75%	6B	20×	9.31	10.39	71.15	62.27
Ours	−75%	**300M**	-	**7.16**	**19.06**	**93.28**	88.02
TransMLA	−87.5%	6B	4×	7.86	10.75	**85.51**	76.61
MHA2MLA	−87.5%	6B	4×	10.11	9.33	65.47	54.18
Ours	−87.5%	**1.5B**	-	**7.55**	**15.89**	82.91	**76.90**

We report WikiText-2 (Merity et al., 2016) perplexity, the unweighted mean over nine LongBench (Bai et al., 2024) tasks with a 2,048-token input limit, and the unweighted mean over thirteen RULER (Hsieh et al., 2024) tasks at 2K and 4K. Appendix B.4 provides the full evaluation protocol.

4.2 Training-Free Conversion

We compare LIVE-MLA with TransMLA (Meng et al., 2026), MHA2MLA (Ji et al., 2025), and CARE (Zhou et al., 2026). No gradient-based recovery training is applied in this setting.

Table 1 reports the training-free results. LIVE-MLA achieves the lowest perplexity on both models at every memory reduction level. Under 50% reduction, its perplexity is 0.40 points above the original Llama-3.1-8B-Instruct model and 0.59 points above the original Qwen3-8B-Base model. Its advantage over the other conversions widens sharply as the reduction becomes more aggressive. Under 87.5% reduction, LIVE-MLA obtains 15.29 on Llama-3.1-8B-Instruct and 13.90 on Qwen3-8B-Base, whereas the next-best converted perplexities are 127.9 and 103.5.

The large performance gaps between several converted baselines and their original models under aggressive memory reduction are consistent with prior reports that preserving model quality often requires additional recovery training. MHA2MLA addresses this gap through recovery fine-tuning, TransMLA benefits from additional training as compression becomes more aggressive, and CARE introduces a subsequent healing stage to restore performance (Ji et al., 2025; Meng et al., 2026; Zhou et al., 2026). In contrast, LIVE-MLA narrows the conversion gap before any gradient-based recovery training.

The long-context results are more mixed. Under 50% memory reduction, LIVE-MLA gives the highest LongBench-2K score on Qwen3-8B-Base and the highest RULER scores on Llama-3.1-8B-Instruct, whereas CARE leads on Llama-3.1-8B-Instruct LongBench-2K and the Qwen3-8B-Base RULER metrics. Under 75% reduction, LIVE-MLA gives the highest LongBench-2K score on both models, while TransMLA and CARE lead on the Llama-3.1-8B-Instruct and Qwen3-8B-Base RULER metrics, respectively. Under 87.5% reduction, LIVE-MLA gives the best converted result on every long-context metric for Llama-3.1-8B-Instruct and the highest LongBench-2K score for Qwen3-8B-Base, while CARE retains the highest Qwen3-8B-Base RULER scores. These RULER advantages should be considered alongside CARE's decoding path, which reconstructs full keys online and incurs substantially higher latency at longer contexts and larger batch sizes than methods using absorbed decoding paths (Appendix B.6). Overall, LIVE-MLA provides a more favorable balance among perplexity, long-context accuracy, and decoding efficiency.

4.3 Recovery Training

We further evaluate recovery training on Qwen3-8B-Base under all three memory reduction levels. LIVE-MLA uses 100M, 300M, and 1.5B recovery tokens under 50%, 75%, and 87.5% reductions, respectively. The baseline runs use between 3B and 6B tokens. We therefore examine whether LIVE-MLA can reach a comparable recovered performance range with substantially less training.

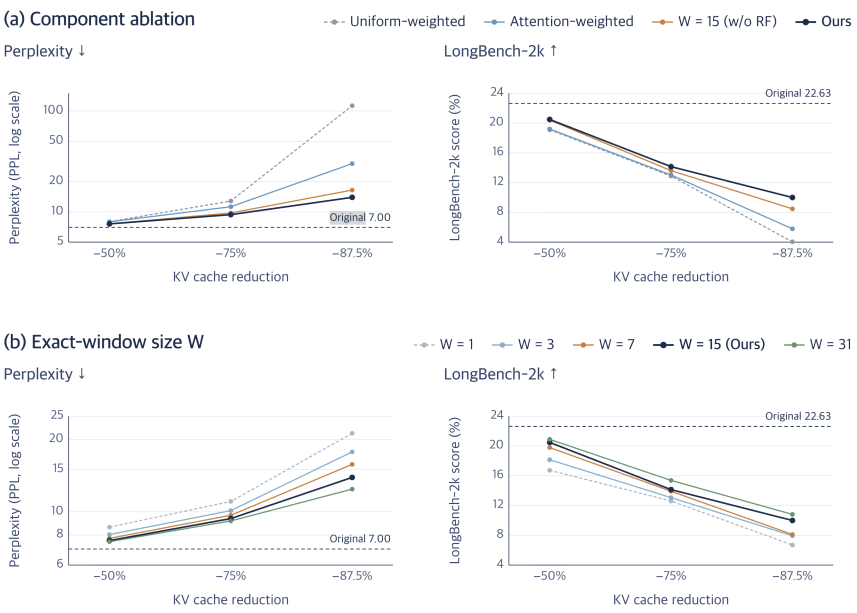

Figure 3: Ablations on Qwen3-8B-Base under matched cache budgets. We report WikiText-2 perplexity and LongBench-2K. Panel (a) cumulatively adds attention-weighted conversion, an exact window retaining $W = 15$ preceding tokens, and Residual Feedback. Panel (b) varies the number W of preceding exact tokens while keeping the other components fixed.

Table 2 shows that LIVE-MLA achieves the lowest perplexity and the highest LongBench-2K score among the converted models at every memory reduction level, while using substantially fewer recovery tokens. Under 50% memory reduction, 100M tokens reduce its perplexity to 6.99, essentially matching the original model and outperforming TransMLA after 3B tokens and MHA2MLA after 6B tokens. LIVE-MLA also achieves the best converted LongBench-2K and RULER-2K scores, while TransMLA leads on RULER-4K. Under 75% reduction, LIVE-MLA reaches a perplexity of 7.16 after 300M tokens, compared with 7.39 for TransMLA and 9.31 for MHA2MLA after 6B tokens. It again achieves the best converted LongBench-2K and RULER-2K scores, while TransMLA remains slightly higher on RULER-4K. Under 87.5% memory reduction, LIVE-MLA achieves the best converted results in perplexity, LongBench-2K, and RULER-4K after 1.5B tokens, while TransMLA leads on RULER-2K.

Overall, LIVE-MLA uses $4\times$–$30\times$ fewer recovery tokens than TransMLA and $4\times$–$60\times$ fewer than MHA2MLA. These results show that LIVE-MLA provides a stronger training-free starting point, enabling shorter recovery training to match or outperform longer baseline recovery runs.

4.4 Component Ablations

We evaluate the components of LIVE-MLA on Qwen3-8B-Base. The latent width is adjusted so that every configuration satisfies the same total cache budget.

Figure 3(a) shows the cumulative effect of each component. Attention-weighted conversion improves over uniform conversion at every memory-reduction level. Retaining $W = 15$ preceding tokens in exact form provides a further gain, and adding Residual Feedback gives the best overall results. The differences become larger as the memory reduction becomes more aggressive, showing that the observed residuals are particularly useful when the latent representation is narrow. The exact window and Residual Feedback state are included in the cache accounting at every reduction level.

Figure 3(b) varies W while keeping Residual Feedback enabled. Under a fixed budget, a larger exact window requires a smaller latent width. For both evaluated models, one exact token stores $G(d_k+d_v) = 2{,}048$ key and value elements per layer, equal to the storage required for one additional latent dimension across the $2{,}048$-token reference length. Thus, each additional exact token requires

reducing the latent width by one dimension. Performance improves up to $W = 31$, the largest value tested. We nevertheless use the fixed default $W = 15$ across all models and memory-reduction levels, without tuning it on the evaluation data. This result suggests that the reported main results are conservative with respect to the window size. Finally, the 16 residual rows used by Residual Feedback require the same storage as eight exact tokens or eight latent dimensions. Reallocating this storage to either alternative does not yield lower perplexity than Residual Feedback under 75% and 87.5% memory reduction.

5 Related Work

Converting pretrained attention to MLA. MLA, introduced in DeepSeek-V2 and retained in DeepSeek-V3 (Liu et al., 2024a;b), reduces the KV cache by caching a low-dimensional latent representation shared by keys and values. Since MLA is normally adopted during pretraining, recent work converts pretrained MHA and GQA models into MLA instead of training MLA models from scratch (Meng et al., 2026; Li et al., 2025; Zhou et al., 2026). MHA2MLA and X-EcoMLA initialize the latent projections by SVD of the pretrained key and value weights (Ji et al., 2025; Li et al., 2025), and TransMLA applies a balanced joint PCA to key and value activations (Meng et al., 2026). CARE, LatentLLM, and MHA2MLA-VLM instead fit the factorization to the activation covariance: CARE additionally allocates rank non-uniformly across layers (Zhou et al., 2026), LatentLLM jointly decomposes paired projections (Koike-Akino et al., 2026), and MHA2MLA-VLM decouples the visual and textual KV spaces (Fan et al., 2026). In all of these methods, the latent mappings are determined once at conversion time and remain fixed during inference, and accuracy lost under aggressive compression is typically recovered through fine-tuning or distillation. In contrast, we adapt the reconstruction to each input sequence at inference time.

Low-rank conversion objectives. Post-hoc low-rank compression converts the pretrained weights into compression and reconstruction projections, and the choice of approximation objective can substantially affect how well the original attention computation is preserved. Rather than minimizing the weight approximation error of a truncated SVD (Chang et al., 2025), activation-aware methods minimize the error in activation space through covariance whitening (Wang et al., 2025; Zhou et al., 2026; Fan et al., 2026), while attention-aware objectives directly approximate the query–key and value–output interactions (Lesens et al., 2025; Koike-Akino et al., 2026).

Residual compensation in compressed KV caches. Many KV-cache compression methods keep a window of recent tokens in exact form, whether as a full-precision residual window under quantization (Liu et al., 2024c; Duanmu et al., 2024), as a retained window under eviction (Xiao et al., 2024), or as full-rank anchors under low-rank projection (Zhu et al., 2026), and DAR (Wang et al., 2026) shows that, when trained under such a split, distant tokens tolerate lower-dimensional representations than local ones. Other methods compensate for compression error through stored low-rank and sparse approximations of the quantization residual (Kang et al., 2024) or through online updates of the projection basis (Zhu et al., 2026). Rather than using the recent window only to preserve local context, we treat it as an online source of supervision, fitting the observed value residuals in closed form to correct older compressed tokens without per-token residual storage or gradient updates.

6 Conclusion

We introduced LIVE-MLA, a training-efficient framework for converting pretrained MHA/GQA models to MLA under fixed cache budgets. Attention-weighted conversion prioritizes reconstruction errors that are more likely to affect later attention outputs. During inference, a small exact window exposes paired latent representations and value residuals, which Residual Feedback uses to fit sequence-specific corrections for older compressed tokens in closed form. The compressed token state, exact window, and residual rows are included in the same cache budget.

Experiments on language modeling and long-context benchmarks show that LIVE-MLA narrows the gap left by training-free conversion. Limited recovery training provides further improvement while using $4\times$ to $60\times$ fewer recovery-training tokens than the evaluated baseline adaptations. These results show that residuals observed during inference provide useful sequence-specific information that fixed conversion mappings leave unused.

AI USE STATEMENT

Generative AI tools were used for language polishing and for assistance with implementation, debugging, and the presentation of mathematical arguments. The authors reviewed all AI-assisted text, code, and mathematical suggestions. All code used in the experiments was tested, and all reported results were verified by the authors. Generative AI tools were not used as a source of factual evidence. The authors take responsibility for the final manuscript, implementation, claims, and results.

ETHICS STATEMENT

This work studies KV-cache compression for language-model inference. It does not involve human subjects, private data, or a new dataset. The experiments use publicly available models and datasets under their respective licenses. LIVE-MLA reduces inference memory but does not address biases or harmful outputs inherited from the original models.

REPRODUCIBILITY STATEMENT

Section 2 defines the converted attention computation and cached state. Section 3 describes attention-weighted conversion, Residual Feedback, the fixed cache budget, recovery training, and inference. Appendix B provides the conversion details, cache accounting, training configuration, evaluation protocols, baseline configurations, and decoding-latency setup. Appendix C provides the assumptions, proofs, and diagnostic experiments used in the analysis. These sections describe the configurations and procedures used for the reported results.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 4895–4901, 2023.

Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on learning theory*, pp. 185–209. PMLR, 2013.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. Longbench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 3119–3137, 2024.

Chi-Chih Chang, Wei-Cheng Lin, Chien-Yu Lin, Chong-Yan Chen, Yu-Fang Hu, Pei-Shuo Wang, Ning-Chi Huang, Luis Ceze, Mohamed Abdelfattah, and Kai-Chiang Wu. Palu: Kv-cache compression with low-rank projection. In *International Conference on Learning Representations*, volume 2025, pp. 50222–50249, 2025.

Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations*, volume 2024, pp. 35549–35562, 2024.

Haojie Duanmu, Zhihang Yuan, Xiuhong Li, Jiangfei Duan, Xingcheng Zhang, and Dahua Lin. Skvq: Sliding-window key and value cache quantization for large language models. *arXiv preprint arXiv:2405.06219*, 2024.

Xiaoran Fan, Zhichao Sun, Tao Ji, Lixing Shen, and Tao Gui. Mha2mla-vlm: Enabling deepseek's economical multi-head latent attention across vision-language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30638–30646, 2026.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Daniel Hsu, Sham M Kakade, and Tong Zhang. Random design analysis of ridge regression. *Foundations of Computational Mathematics*, 14(3):569–600, 2014.

Tao Ji, Bin Guo, Yuanbin Wu, Qipeng Guo, Lixing Shen, Zhan Chen, Xipeng Qiu, Qi Zhang, and Tao Gui. Towards economical inference: Enabling deepseek's multi-head latent attention in any transformer-based llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 33313–33328, 2025.

Hao Kang, Qingru Zhang, Souvik Kundu, Geonhwa Jeong, Zaoxing Liu, Tushar Krishna, and Tuo Zhao. Gear: An efficient error reduction framework for kv cache compression in llm inference. In *ENLSP*, pp. 305–321, 2024.

Toshiaki Koike-Akino, Xiangyu Chen, Jing Liu, Ye Wang, Pu Perry Wang, and Matthew Brand. Latentllm: Activation-aware transform to multi-head latent attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 22644–22652, 2026.

Hannah Laus, Claudio Mayrink Verdun, Hao Wang, Flavio du Pin Calmon, and Felix Krahmer. Kv cache compression through the lens of transform coding. *arXiv preprint arXiv:2608.14191*, 2026.

Damien Lesens, Beheshteh T Rakhshan, and Guillaume Rabusseau. Kq-svd: Compressing the kv cache with provable guarantees on attention fidelity. *arXiv preprint arXiv:2512.05916*, 2025.

Guihong Li, Mehdi Rezagholizadeh, Mingyu Yang, Vikram Appia, and Emad Barsoum. X-ecomla: Upcycling pre-trained attention into mla for efficient and extreme kv compression. *arXiv preprint arXiv:2503.11132*, 2025.

Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. *arXiv preprint arXiv:2405.04434*, 2024a.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024b.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. Kivi: A tuning-free asymmetric 2bit quantization for kv cache. *arXiv preprint arXiv:2402.02750*, 2024c.

Fanxu Meng, Pingzhi Tang, Zengwei Yao, Xing Sun, and Muhan Zhang. Transmla: Migrating gqa models to mla with full deepseek compatibility and speedup. *Advances in Neural Information Processing Systems*, 38:81977–82019, 2026.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of machine learning and systems*, 5:606–624, 2023.

Konrad Staniszewski and Adrian Lancucki. Kv cache transform coding for compact storage in llm inference. In *International Conference on Learning Representations*, volume 2026, pp. 37768–37815, 2026.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Pei-Chun Su. eoptshrinkq: Near-lossless kv cache compression through optimal spectral denoising and quantization. *arXiv preprint arXiv:2605.02905*, 2026.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Xin Wang, Yu Zheng, Zhongwei Wan, and Mi Zhang. Svd-llm: Truncation-aware singular value decomposition for large language model compression. In *International Conference on Learning Representations*, volume 2025, pp. 19299–19319, 2025.

Zhiyuan Wang, Xuan Luo, Sirui Zeng, and Xifeng Yan. Dual dimensionality for local and global attention. *arXiv preprint arXiv:2606.18587*, 2026.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, volume 2024, pp. 21875–21895, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhongzhu Zhou, Fengxiang Bie, Ziyan Chen, Zhenyu Zhang, Yibo Yang, Junxiong Wang, Ben Athiwaratkun, Xiaoxia Shirley Wu, and Shuaiwen Song. Care: Covariance-aware and rank-enhanced decomposition for enabling multi-head latent attention. In *International Conference on Learning Representations*, volume 2026, pp. 43245–43278, 2026.

Yuxuan Zhu, David H Yang, Mohammad Mohammadi Amiri, Keerthiram Murugesan, Tejaswini Pedapati, and Pin-Yu Chen. Ojakv: Context-aware online low-rank kv cache compression. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 10161–10178, 2026.

A LIMITATIONS

Our main task-quality evaluation covers two pretrained 8B models and context lengths up to 4K tokens. We do not pretrain an MLA model from scratch, so our results do not establish whether Residual Feedback remains useful when the latent representation is learned throughout pretraining rather than introduced through conversion. Recovery training is evaluated only on Qwen3-8B-Base. Our current decoding implementation is also not the fastest among the compared conversion methods. As shown in Appendix B.6, LIVE-MLA is faster than MHA2MLA but slower than TransMLA across the measured settings. The gap reflects the exact window and the residual rows, which are read at every decoding step in addition to the latent cache. Improving decoding efficiency while preserving the benefits of Residual Feedback remains future work. Further evaluation on models trained from scratch, larger models, and substantially longer contexts is also needed.

B IMPLEMENTATION DETAILS

B.1 CONVERSION DETAILS

Shared rotary code. For each of the first $d_R/2$ RoPE pairs of every key head, a rank-one fit (Meng et al., 2026) produces a projection shared across the key-value heads and one fixed complex coefficient per key-value head. The shared projections form W_P and produce $p_i \in \mathbb{R}^{d_R}$, while the coefficients are stored in the converted model and incorporated into the corresponding query coordinates. The remaining content coordinates of an older compressed token contribute through c_i. Tokens in the exact window instead use their original keys and the pretrained RoPE operation over all coordinates.

Compressed-token score. For query head u at position t, let $W_{UK}^{(\pi(u))} \in \mathbb{R}^{r\times(d_k-d_R)}$ denote the block of W_{UK} for key-value head $\pi(u)$, let $q_{t,\mathrm{C}}^{(u)} \in \mathbb{R}^{d_k-d_R}$ denote the content coordinates of its query after any query normalization and before RoPE, and let $\widetilde{q}_{t,\mathrm{R}}^{(u)} \in \mathbb{R}^{d_R}$ denote the selected rotary coordinates of its query after incorporating the coefficients of key-value head $\pi(u)$. Let $\mathrm{RoPE}_t^{(d_R)}$ denote the rotary transformation at position t restricted to these d_R coordinates. For an older compressed token j, the score is

$$\ell_{tj}^{(u)} = \gamma_j^{(\pi(u))}\left[\left\langle q_{t,\mathrm{C}}^{(u)}\left(W_{UK}^{(\pi(u))}\right)^{\top}, c_j\right\rangle + \left\langle \mathrm{RoPE}_t^{(d_R)}\left(\widetilde{q}_{t,\mathrm{R}}^{(u)}\right), \mathrm{RoPE}_j^{(d_R)}\left(p_j\right)\right\rangle\right].$$

The attention scaling $1/\sqrt{d_k}$ is applied to both exact and compressed scores and is omitted from $\ell_{tj}^{(u)}$. The first term does not depend on the position of token j, so $W_{UK}^{(\pi(u))}$ is combined with the query projection and the product is computed directly against c_j. The second term applies RoPE only to the selected d_R coordinates, and the key-value-head coefficient is folded into $\widetilde{q}_{t,\mathrm{R}}^{(u)}$ because complex scaling commutes with the rotary transformation. When the model uses per-head key normalization, the stored factor $\gamma_j^{(\pi(u))}$ scales both terms.

Attention-weighted conversion. We jointly fit the content-key and value mappings with the weighted reduced-rank objective of Section 3.1. The content-key target is rescaled so that its average squared magnitude matches that of the value. Each fitted token is weighted by the attention it receives from the queries that read it from the compressed cache, that is, from queries more than W positions later. The weights are normalized to mean one for each layer and input sequence. We directly accumulate the weighted second moment and cross moment matrices and apply the reduced-rank factorization. This produces W_D, W_{UK}, and W_{UV} after the original key scale is restored in W_{UK}. The accumulated matrices do not depend on the rank, so several cache shares can be fitted from one pass over the input sequences.

Fitting configuration. We accumulate over 96 windows of $T_a = 512$ tokens from the concatenated Alpaca training split, each contributing its first $|\mathcal{I}_a| = 256$ positions. The attention mass is taken with exponent one and normalized to mean one within each (layer, window) pair, and the mask follows the sliding exact window with $W = 15$. The same tokens and weights are used for

every latent width r, so configurations differ only in the rank. The rotary projection W_P and its head-specific coefficients are fitted using 24 windows of 512 tokens from the same split with 1% damping. They are trained with the base mappings during recovery training.

Exact window and Residual Feedback. We set $W = 15$, so the cache retains up to W preceding tokens in exact form between decoding steps. Together with the current token, at most $W + 1 = 16$ exact tokens participate in attention for a query. The exact key and value of token t are used to compute its own attention output and are added to the cache afterward. Once this output has been computed, token $t - W$ leaves the window when $t \geq W$. Its value residual is recorded before the exact value is discarded and can first be used by query $t + 1$. For query t, the fitting set contains up to $S = 16$ residual observations from positions $t - W - S$ through $t - W - 1$. Each query solves an $n_t \times n_t$ ridge system in FP32 with $\rho = 10^{-3}$ and $\lambda_t = \rho\|\Phi_t\|_F^2/n_t$. Near the start of a sequence, the fit uses the available rows and is disabled when $n_t = 0$.

Computing the fits for a full sequence. During recovery training and prefill, the fits for multiple queries are computed together. Since nearby fitting sets overlap, we process queries in blocks of 128. Block matrix products provide the required inner products, Gram matrices, and products with the residual rows. The query-specific systems are then solved with a batched Cholesky factorization. This computation is algebraically equivalent to solving each query separately.

Decoding. The decoding state of each layer contains c_i, p_i, and γ_i when required. It also contains a ring buffer for the exact window and a ring buffer for the residual observations. Exact keys are stored after the pretrained RoPE operation. For an older compressed token, the content score is computed directly from c_i after incorporating W_{UK} into the query. The rotary score is computed from p_i over the selected coordinates, with the head-specific coefficients incorporated into the query. The stored normalization values are applied when required. The value projection W_{UV} is incorporated into the output projection. Full keys are not reconstructed for compressed tokens. A reconstructed value is materialized only for the token leaving the exact window so that its residual can be recorded. Each query solves an $n_t \times n_t$ ridge system with a single CUDA kernel implementing Gauss-Jordan elimination. The resulting decoding step can be captured as a CUDA graph.

B.2 Cache Accounting

We include the compressed token state, the exact window, and the residual rows in a single cache budget. For layer ℓ, let

$$D_\ell = G_\ell(d_{k,\ell} + d_{v,\ell}), \qquad \kappa_\ell = r_\ell + d_R + d_{N,\ell},$$

where D_ℓ is the original key and value width and κ_ℓ is the compressed width. The term $d_{N,\ell}$ counts the stored key-normalization values. It is 8 for Qwen3-8B-Base and 0 for Llama-3.1-8B-Instruct. After T tokens, define

$$w_T = \min(W, T), \qquad s_T = \min\{S, \max(0, T - W)\}.$$

All persistent state is stored in BF16. The storage used by layer ℓ is

$$\mathcal{M}_\ell(T) = 2\left(T\kappa_\ell + w_T D_\ell + s_T G_\ell d_{v,\ell}\right). \tag{1}$$

The three terms account for the compressed state of every token, the exact keys and values in the window, and the value residuals. The latent features used by Residual Feedback are read from the compressed token state, and the constant feature is not stored. The FP32 tensors used by the ridge solve are temporary and are not included in the persistent cache.

At the reference length $T_{\mathrm{ref}} = 2{,}048$, we choose the largest latent width satisfying

$$\sum_{\ell=1}^{L} \mathcal{M}_\ell(T_{\mathrm{ref}}) \leq 2\beta T_{\mathrm{ref}} \sum_{\ell=1}^{L} D_\ell,$$

Table 3: Exact per-layer cache storage at $T_{\text{ref}} = 2{,}048$. All storage columns are measured in bytes.

Model	β	r	κ	Compressed	Exact window	Residual rows	Total	Target
Qwen3-8B-Base	12.5%	161	233	954,368	61,440	32,768	1,048,576	1,048,576
Qwen3-8B-Base	25%	417	489	2,002,944	61,440	32,768	2,097,152	2,097,152
Qwen3-8B-Base	50%	929	1001	4,100,096	61,440	32,768	4,194,304	4,194,304
Llama-3.1-8B-Instruct	12.5%	169	233	954,368	61,440	32,768	1,048,576	1,048,576
Llama-3.1-8B-Instruct	25%	425	489	2,002,944	61,440	32,768	2,097,152	2,097,152
Llama-3.1-8B-Instruct	50%	937	1001	4,100,096	61,440	32,768	4,194,304	4,194,304

Table 4: Realized cache share at each context length. The target share is defined at $T_{\text{ref}} = 2{,}048$, shown in bold.

β	κ	1K	2K	4K	8K	16K
12.5%	233	13.62%	**12.500%**	11.94%	11.66%	11.52%
25%	489	26.12%	**25.000%**	24.44%	24.16%	24.02%
50%	1001	51.12%	**50.000%**	49.44%	49.16%	49.02%

where β is the target fraction of the original KV-cache storage. The corresponding memory reduction is $1 - \beta$. Both evaluated models have $D_\ell = 2{,}048$ and $G_\ell d_{v,\ell} = 1{,}024$. With $W = 15$ and $S = 16$, the exact window and residual rows require

$$WD_\ell + SG_\ell d_{v,\ell} = 47{,}104 = 23D_\ell$$

elements per layer. Since $T_{\text{ref}} = D_\ell = 2{,}048$, the target condition becomes

$$\kappa_\ell + 23 = 2{,}048\beta.$$

Table 3 reports the exact byte count for every model and target share. The total equals the target in every row, so no rounding slack is introduced.

For $T \geq W + S$, the realized cache share is

$$\widehat{\beta}_\ell(T) = \frac{T\kappa_\ell + WD_\ell + SG_\ell d_{v,\ell}}{TD_\ell}.$$

The two models have the same κ_ℓ and D_ℓ, so they have the same realized share at every context length. Their latent widths differ because Qwen3-8B-Base stores eight key-normalization values per token, whereas Llama-3.1-8B-Instruct does not. Table 4 shows how the fixed window and residual costs affect the realized share as the context length changes.

B.3 Training Details

Recovery training uses the 6B-token mixture released with TransMLA[1], which consists of fineweb-edu-dedup, cosmopedia-v2, open-web-math, python-edu and stackoverflow with proportions of 70%, 15%, 8%, 6% and 1%. We train Qwen3-8B-Base for 100M tokens under 50% memory reduction, 300M tokens under 75% reduction, and 1.5B tokens under 87.5% reduction. For the 87.5% run, we also evaluate intermediate checkpoints. Table 5 summarizes the configuration.

Trainable parameters. Recovery training updates the latent mappings, the attention projections, and the query and key normalizers. The embeddings, MLP parameters, other normalization layers and language-model head remain fixed.

[1] `fxmeng/transmla_pretrain_6B_tokens`

Table 5: Recovery-training hyperparameters.

Hyperparameter	Value
Model	Qwen3-8B-Base
Training data	TransMLA 6B-token mixture
Sequence length	4096
Sequences per optimizer step	128 (524,288 tokens)
Optimizer	AdamW
Learning rate	2×10^{-5}
Learning-rate schedule	constant
Weight decay	0.01
AdamW (β_1, β_2)	$(0.9, 0.999)$
AdamW ε	1×10^{-8}
Gradient clipping	1.0
Precision	bf16
Gradient checkpointing	enabled
Trainable parameters	attention-related parameters
Objective	cross entropy
Training tokens under 50% reduction	100M (191 steps)
Training tokens under 75% reduction	300M (573 steps)
Training tokens under 87.5% reduction	1.5B (2862 steps)
Residual Feedback during training	value only, $W = 15$, $S = 16$

Training objective and feedback. We optimize the next-token cross-entropy objective. Residual Feedback is active in the training forward pass with the same window, fitting set, residual precision and ridge coefficient as at evaluation. The fitting side of the ridge solve is treated as constant during backpropagation. Before training, we verify that the training forward pass closely matches the evaluation path. On 40 windows of 2048 tokens, the mean per-window NLL difference is -0.0006 nats and the largest absolute difference is 0.008.

B.4 Evaluation Details

Perplexity. For each backbone, we concatenate all examples in the WikiText-2 `wikitext-2-raw-v1` test split, tokenize it with the corresponding tokenizer, and divide it into non-overlapping windows. Each window is evaluated independently, and all methods derived from the same backbone use the same tokenized windows. We report the exponential of the token-level mean NLL. The main results use windows of 2,048 tokens, with 146 windows for Qwen3-8B-Base and 141 windows for Llama-3.1-8B-Instruct.

LongBench. We use LongBench v1 and keep nine of its sixteen English tasks. The selected tasks are `gov_report`, `qasper`, `repobench-p`, `hotpotqa`, `qmsum`, `musique`, `passage_count`, `narrativeqa`, and `passage_retrieval_en`. We exclude the five Chinese tasks (`multifieldqa_zh`, `dureader`, `vcsum`, `lsht`, and `passage_retrieval_zh`). We also exclude seven English tasks whose inputs often fall below our 2,048-token input limit. These are `multi_news`, `lcc`, `multifieldqa_en`, `2wikimqa`, `samsum`, `trec`, and `triviaqa`. The target KV-cache memory reduction is defined at 2,048 tokens, and our method has a fixed per-sequence cost (the exact window and the feedback state). A shorter input would therefore yield a different realized reduction for each method. We keep a task only if at most 1% of its test samples are shorter than 2,048 tokens under every tokenizer considered during preprocessing. In the excluded English tasks, between 2% and 51% of samples are shorter. In the kept tasks, only one sample in `qasper` and one sample in `hotpotqa` are shorter. We score both at their full length. We truncate longer inputs to 2,048 tokens with the benchmark's middle-truncation rule, score each task with its official metric, and report the unweighted mean over the nine tasks.

RULER. We use the official generator at context lengths of 2,048 and 4,096 tokens with 500 samples per task and seed 1234. We evaluate all thirteen tasks, consisting of `niah_single_1/2/3`, `niah_multikey_1/2/3`, `niah_multiquery`, `niah_multivalue`, `cwe`, `fwe`, `vt`, `qa_1`,

Table 6: Rotary width d_R and shared latent width r for methods with this parameterization. CARE uses separate key and value ranks and is not included in this table.

Model	Method	d_R	r@-50%	r@-75%	r@-87.5%
Llama-3.1-8B-Instruct	TransMLA	64	960	448	192
	MHA2MLA	128	896	384	128
	LIVE-MLA	64	937	425	169
Qwen3-8B-Base	TransMLA	64	952	440	184
	MHA2MLA	128	896	384	128
	LIVE-MLA	64	929	417	161

and `qa_2`. The same generated prompts are shared across all conditions. We report the unweighted mean over the thirteen tasks.

Efficiency. Decoding latency is measured with CUDA graphs for all methods, after a prefill of the given length and warm-up steps, as the median over decoding steps. The uncompressed model uses a FlashAttention (Dao, 2024) decoding kernel with a static cache, which attends to the actual context length without replicating key-value heads. Compared methods are measured in the same job on the same GPU. Memory is measured by prefilling each request separately, merging the caches layer by layer, and decoding a batch. The maximum batch size is the largest power of two that completes without running out of memory.

B.5 Baseline Configurations

Implementation. All baselines are converted with their official repositories using the default uniform-rank setting of each method. We do not report the dynamic-rank variant of CARE in Table 1 because the official converter does not save a converted checkpoint in this mode. The official TransMLA and CARE converters do not apply the per-head query and key normalization of Qwen3. We port this normalization by folding the key scale into the key projection, caching one RMS value per key-value head within the same cache budget, and applying query normalization before absorption. The ported converter reproduces the original Qwen3-8B perplexity when evaluated without low-rank truncation. For the efficiency comparison, we use an absorbed TransMLA decoding path that caches only the latent and rotary states and verify it against the official forward pass.

Reproduction checks. For CARE, the official repository at rank 128 on Llama-3.1-8B-Instruct gives a perplexity of 398.99, matching the 398.91 reported in the CARE paper. For MHA2MLA, our loader reproduces the official conversion on Qwen3-0.6B with identical tensors and logits.

Table 6 lists the latent and rotary widths for methods that use a shared-latent layout. CARE uses separate method-specific key and value ranks, so these columns do not describe its cache layout. LIVE-MLA additionally includes the exact window and residual rows in the same cache budget.

B.6 Decoding Latency

We measure decoding latency on Qwen3-8B-Base under 87.5% memory reduction using an NVIDIA RTX 6000 Ada GPU. All methods are measured in one process on the same GPU. Each measurement uses a CUDA graph after prefill and excludes the first eight decoding steps as warm-up. We report the median latency over the remaining decoding steps. Layer fusion is applied to every method. We tune the method-specific kernel settings and report the fastest valid configuration for each method. TransMLA is evaluated with split multipliers of 2 and 4, while LIVE-MLA is evaluated with head blocks of 16 and 32. For every entry, we compare the decoding NLL with a cache-free full forward pass. The largest difference is 0.02 nats.

The original model uses FlashAttention with a static cache and attends only to the active context. TransMLA, MHA2MLA, and LIVE-MLA compute attention scores directly from their latent caches through absorbed decoding paths. CARE does not use partial RoPE in its one-shot conversion, so its decoding path reconstructs the keys online.

Table 7: Decoding latency in milliseconds per token on Qwen3-8B-Base under 87.5% memory reduction.

Method	8K × 1	8K × 4	8K × 16	32K × 1	32K × 4	32K × 16
Original	12.62	14.09	24.53	15.45	23.98	64.11
TransMLA	12.08	12.12	15.13	13.31	14.28	24.28
MHA2MLA	13.34	14.27	17.36	14.51	16.94	27.23
CARE	13.75	17.70	38.29	18.20	35.33	116.90
LIVE-MLA	**12.84**	**12.72**	**16.61**	**13.87**	**15.33**	**26.45**

LIVE-MLA has latency close to that of the original model at batch size 1 and becomes faster as the context length and batch size increase. At 32K with batch size 16, it reduces latency by 58.7% relative to the original model. CARE is slower throughout because reconstructing full keys introduces work that grows with both context length and batch size. LIVE-MLA is 4% to 10% slower than TransMLA because it additionally maintains $W = 15$ preceding exact tokens and fits Residual Feedback from $S = 16$ rows at each step. Its combined latent and rotary state is about 9% narrower, with 225 elements per token and layer compared with 248 for TransMLA. This comparison is limited to one model and one memory-reduction level.

C Sequence-Specific Residual Structure and Attention-Weighted Recovery

We characterize sequence-specific residual prediction by Residual Feedback (RF), the identifiability of an attention-weighted residual sum, and the effects of temporal covariance and regularization on output risk. The geometric statements are deterministic, and the risk identities are conditional on explicitly specified design information. Except for the joint key–value perturbation analysis, the attention weights are fixed throughout each comparison. The results therefore concern value-induced error for a specified attention configuration, independently of how its keys and logits are parameterized. The argument proceeds from conditional recoverability to feature-moment matching, output-risk decomposition, spectral regularization, and exact finite-sample identities, and then connects these results to the other components of LIVE-MLA and to empirical evaluations of key correction, byte allocation, and transfer to older tokens. The scope is layerwise attention output, not a guarantee of lower end-to-end negative log-likelihood (NLL).

C.1 Scope, notation, and conditional moment assumptions

Fix a layer, a query head, and a query position t. We use the causal sets $\mathcal{R}_t$, $\mathcal{O}_t$, and $\mathcal{F}_t$ from the method and suppress t where convenient. Let $F = \mathcal{F}_t$, $n = |F| > 0$, and $p = r + 1$. For the key-value head associated with the query head, write $e_i = v_i - \hat{v}_i \in \mathbb{R}^{1\times d_v}$, $\phi_i = [c_i, 1] \in \mathbb{R}^{1\times p}$, and let $\Phi \in \mathbb{R}^{n\times p}$ and $E_F \in \mathbb{R}^{n\times d_v}$ stack the fitting features and residuals. The full-layer implementation stacks $d_V = Gd_v$ value coordinates; our headwise statements apply to the corresponding columns of its shared solve. We reserve r for the latent width and B for a residual coefficient map. Output-risk gains are denoted by $\mathcal{G}$, mean departures by τ_i, and remaining value errors by ε_i^V.

We analyze the affine ridge estimator used by Residual Feedback. The feature-scaled regularization convention is

$$\lambda = \rho\frac{\|\Phi\|_F^2}{n}, \qquad \rho > 0, \qquad \hat{B} = \Phi^\top(\Phi\Phi^\top + \lambda I_n)^{-1}E_F. \tag{2}$$

The reported default is $\rho = 10^{-3}$; the results below hold for every positive design-dependent λ. The constant feature is regularized, and λ depends on the features, not on the observed residual targets. When $n = 0$, RF is disabled and all corrections below are zero.

Let $\mathcal{C}_t$ be the causal source set of the configuration under consideration, and let $\alpha_j \geq 0$ with $\sum_{j\in\mathcal{C}_t}\alpha_j = 1$. In the method, these are the weights of the hybrid key path, which uses exact scores for $\mathcal{R}_t \cup \{t\}$ and compressed scores for $\mathcal{O}_t$. The exact-value reference and the value-corrected output use the same weights. This isolates the value path; no particular RoPE convention or key-reconstruction parameterization is needed for the following identities. For a target set $J \subseteq \mathcal{O}_t \cap \mathcal{C}_t$,

let Φ_J and E_J stack its feature and residual rows, and define

$$a_J = (\alpha_j)_{j\in J}, \qquad x_J = \sum_{j\in J} \alpha_j \phi_j, \qquad d_J = \sum_{j\in J} \alpha_j e_j = a_J E_J. \tag{3}$$

The correction restricted to J is $\hat{d}_J = x_J \hat{B}$. Here d_J is precisely the value-induced output error contributed by J before correction. When $J = \mathcal{O}_t \subseteq \mathcal{C}_t$, x_J is the attention-weighted latent sum, including the final mass coordinate, used by RF.

What is random? The checkpoint and conversion are fixed. Randomness refers to an ensemble of input sequences and query configurations; the reconstruction residual of a particular hidden state is deterministic. In the conditional analysis, $\mathcal{D}$ denotes the design information: the relevant latent features, attention weights, index sets, and any sequence-level conditioning variables. All expectations in Sections C.4–C.5 are conditional on $\mathcal{D}$. We require finite conditional second moments. Features can be temporally dependent and need not be independently sampled. Conditioning on the design does not itself make the residual errors independent or make their conditional means linear in ϕ_i.

Statistical models and covariance assumptions. Hsu et al. (2014, Section 3.1) analyze fixed-design ridge regression with independent responses and separate approximation error from centered noise. For KV-cache quantization, Laus et al. (2026, Assumption 1 and Remark 2) use independent, zero-mean errors and explicitly identify cross-token independence as an idealization. In contrast, the residual covariance model of Su (2026, Section 4.1) allows temporal dependence. These are precedents for particular statistical assumptions, not moment conditions imposed on the MLA innovations defined below. Bach (2013, Section 4.1) derives fixed-design kernel-ridge prediction risk under finite noise covariance without requiring independent or identically distributed noise. We likewise retain covariance terms and specialize only where stated. The identities require neither Gaussianity nor coordinate independence nor a concentration assumption.

C.2 Why globally discarded information can be locally predictable

For the value path of a fixed conversion, let

$$c = hW_D, \qquad v = hW_V, \qquad e = hA_V, \qquad A_V = W_V - W_D W_{UV}. \tag{4}$$

Here W_V and W_{UV} contain the columns of the selected value head. Although c and e are both functions of h, compression generally prevents e from being a deterministic function of c.

Proposition C.1 (Global orthogonality and conditional recoverability) *Assume finite second moments and suppose a decoder U_0 minimizes the unregularized population objective $\mathbb{E}\|v - cU\|^2$ for fixed W_D. Its residual $e_0 = v - cU_0$ satisfies $\mathbb{E}[c^\top e_0] = 0$. This does not imply zero conditional covariance within every sequence class. For the residual in (4) and a sequence condition z,*

$$\Sigma_{ce|z} = W_D^\top \Sigma_{h|z} A_V. \tag{5}$$

After centering c and e within z, the best linear residual predictor has coefficient $B_z^\star = \Sigma_{cc|z}^\dagger \Sigma_{ce|z}$, and reduces the population squared error by

$$\Delta_z = \operatorname{tr}\Big(\Sigma_{ec|z} \Sigma_{cc|z}^\dagger \Sigma_{ce|z}\Big) \geq 0. \tag{6}$$

The dagger denotes the Moore–Penrose pseudoinverse.

Proof. Differentiating the decoder objective gives $\mathbb{E}[c^\top(v - cU_0)] = 0$. Conditional covariance under deterministic linear transformations gives (5). For centered variables, the residual-prediction objective is

$$L_z(B) = \mathbb{E}[\|e\|^2 \mid z] - 2\operatorname{tr}(B^\top \Sigma_{ce|z}) + \operatorname{tr}(B^\top \Sigma_{cc|z} B).$$

Every vector in the nullspace of $\Sigma_{cc|z}$ has zero conditional variance when projected onto c, and therefore has zero covariance with e. Thus the columns of $\Sigma_{ce|z}$ lie in the range of $\Sigma_{cc|z}$, so the stated pseudoinverse solution satisfies the normal equations. Substitution gives (6); nonnegativity follows because $\Sigma_{cc|z}^\dagger$ is positive semidefinite. ∎

The centering in this proposition concerns a population projection. The implemented RF remains the regularized affine fit in (2). Relative to a zero predictor for uncentered residuals, the optimal affine population predictor can additionally recover $\|\mathbb{E}[e \mid z]\|^2$. Neither this population optimum nor Δ_z is a finite-buffer guarantee.

An illustrative sequence model. Let a sequence share $z \in \{-1, +1\}$ with equal probabilities, and let c_i be centered, independent of z, with variance one. Set $e_i = z\beta c_i + \epsilon_i$, where $\mathbb{E}[\epsilon_i \mid c_i, z] = 0$. Pooled across sequences, $\mathbb{E}[c_i e_i] = 0$, but $\mathbb{E}[e_i \mid c_i, z] = z\beta c_i$. The conditional regression coefficient is $z\beta$, shared by recent and older tokens in the same sequence. Thus global orthogonality is compatible with a nonzero within-sequence residual map.

Relation to attention-weighted conversion. An exact attention-weighted least-squares decoder instead satisfies $\mathbb{E}[mc^\top e_0] = 0$, equivalently the normal equation under the reweighted distribution proportional to m, provided $0 < \mathbb{E}[m] < \infty$. Damping, numerical approximation, or a newly introduced intercept can also leave a globally predictable residual. These effects are distinct from the conditional covariance term in (6); global predictability and sequence-conditional predictability can coexist.

C.3 RF ESTIMATES AN ATTENTION-WEIGHTED FUNCTIONAL

The method need not reconstruct each older residual before aggregating it. Define the row vector

$$w_J = x_J \Phi^\top (\Phi\Phi^\top + \lambda I_n)^{-1}. \tag{7}$$

Then the correction is exactly $\hat{d}_J = w_J E_F$. This identity applies to any target set J, independently of a noise model.

Proposition C.2 (Regularized feature-moment matching) *For $\lambda > 0$, the weights in (7) uniquely solve*

$$w_J = \arg\min_{w \in \mathbb{R}^{1\times n}} \left\{ \|x_J - w\Phi\|^2 + \lambda \|w\|^2 \right\}. \tag{8}$$

Thus RF matches the attention-weighted feature sum of the target tokens by a regularized linear combination of recent fitting features, and transfers that same combination to the observed residuals.

Proof. The derivative is $2(w\Phi - x_J)\Phi^\top + 2\lambda w$. Setting it to zero yields $w(\Phi\Phi^\top + \lambda I_n) = x_J \Phi^\top$. The matrix is positive definite, giving uniqueness and the stated solution. Multiplication of (2) by x_J gives $\hat{d}_J = w_J E_F$. ∎

The weights w_J need not be nonnegative or sum to one. Their role is not a second softmax. The constant feature makes the target mass $A_J = \sum_{j\in J} \alpha_j$ one of the moments being matched; for positive λ, even this coordinate need not be matched exactly.

Proposition C.3 (Identifiability of the required output direction) *In the noiseless shared-map model $E_F = \Phi B$, the quantity $x_J B$ is uniquely determined by (Φ, E_F, x_J) for every $B \in \mathbb{R}^{p\times d_v}$ if and only if $x_J \in \mathrm{rowspan}(\Phi)$. This can hold when $n \ll p$ and B itself is not identifiable.*

Proof. If $x_J = w\Phi$, then $x_J B = w E_F$ for every B consistent with the observations. Conversely, if x_J is outside the row space, its projection onto the nullspace of Φ is nonzero. There is a column vector u such that $\Phi u = 0$ and $x_J u \neq 0$. For any nonzero row vector b, B and $B + ub$ give the same fitting observations but different values of $x_J B$. Exact identification is therefore impossible over this unrestricted coefficient class. ∎

This proposition concerns what the observations identify. With positive λ, the implemented ridge estimator still introduces shrinkage bias; it need not achieve exact recovery even when the target is identifiable.

C.4 OUTPUT-RISK DECOMPOSITION WITH ARBITRARY TEMPORAL COVARIANCE

For each relevant token, define its conditional mean and centered innovation by

$$\mu_i = \mathbb{E}[e_i \mid \mathcal{D}], \qquad \xi_i = e_i - \mu_i, \qquad \Gamma_{ik} = \mathbb{E}[\langle \xi_i, \xi_k \rangle \mid \mathcal{D}]. \tag{9}$$

The matrix Γ is a positive-semidefinite token covariance obtained by summing the coordinatewise covariances. It does not assume independence between coordinates. Let M_F and M_J stack the means. We abbreviate $w = w_J$, $a = a_J$, and

$$b_J = aM_J - wM_F. \tag{10}$$

The following result is valid whether or not the means are approximately linear in the latent features.

Proposition C.4 (General conditional output risk) *For $J \cap F = \emptyset$, RF has conditional output risk*

$$R_{\mathrm{RF}}(J) := \mathbb{E}[\|d_J - wE_F\|^2 \mid \mathcal{D}] = \|b_J\|^2 + a\Gamma_{JJ}a^\top + w\Gamma_{FF}w^\top - 2a\Gamma_{JF}w^\top. \tag{11}$$

Without feedback,

$$R_0(J) = \|aM_J\|^2 + a\Gamma_{JJ}a^\top. \tag{12}$$

Consequently the exact conditional gain is

$$\mathcal{G}(J) := R_0(J) - R_{\mathrm{RF}}(J) = \|aM_J\|^2 - \|b_J\|^2 - w\Gamma_{FF}w^\top + 2a\Gamma_{JF}w^\top. \tag{13}$$

RF reduces this MSE if and only if $\mathcal{G}(J) > 0$.

Proof. Write $E_J = M_J + \Xi_J$ and $E_F = M_F + \Xi_F$. The output error is $b_J + a\Xi_J - w\Xi_F$. The conditional means of both innovation matrices are zero, so all cross terms with b_J vanish. For deterministic coefficient rows u, v, the definition of Γ gives $\mathbb{E}[\langle u\Xi_A, v\Xi_B\rangle \mid \mathcal{D}] = u\Gamma_{AB}v^\top$. Expanding the square yields (11). Expanding the uncorrected error yields (12); subtracting gives (13). ∎

The fitting–target covariance has a signed effect. Positive $a\Gamma_{JF}w^\top$ can help prediction, while a negative value can hurt. Temporal dependence is therefore not equivalent to an additional positive noise penalty. Target–target covariance cancels from the gain because the same target is compared before and after correction.

Shared structure and drift. To connect the general means to a shared local map, write

$$\mu_i = \phi_i B_F + \tau_i, \qquad M_F = \Phi B_F + \mathsf{T}_F, \qquad \tau_J = \sum_{j \in J} \alpha_j \tau_j, \tag{14}$$

where T_F stacks the fitting-side mean departures from the shared map. This decomposition imposes no restriction on their size. The output bias becomes

$$b_J = (x_J - w\Phi)B_F + \tau_J - w\mathsf{T}_F. \tag{15}$$

If the conditional mean is $\phi_i B_i$, then $\tau_i = \phi_i(B_i - B_F)$. Thus the required stability is small *attention-projected* coefficient drift, rather than a claim that all coefficients or all tokenwise residuals remain constant over long distances. The identity is valid for arbitrary drift. Only its projection through a_J and w_J contributes to the output bias.

Corollary C.5 (Uncorrelated fitting-noise special case) *Suppose $\mu_i = \phi_i B$ for $i \in F \cup J$, $J \cap F = \emptyset$, and, conditionally on $\mathcal{D}$,*

$$\Gamma_{JF} = 0, \qquad \Gamma_{FF} = \sigma^2 I_n. \tag{16}$$

No restriction is imposed on the target–target covariance Γ_{JJ}. Then

$$R_{\mathrm{RF}}(J) = \|(x_J - w\Phi)B\|^2 + a\Gamma_{JJ}a^\top + \sigma^2\|w\|^2, \tag{17}$$

$$\mathcal{G}(J) = \|x_J B\|^2 - \|(x_J - w\Phi)B\|^2 - \sigma^2\|w\|^2. \tag{18}$$

Thus RF improves conditional output MSE precisely when structural output-error reduction exceeds the fitting-noise contribution.

Proof. The shared-map assumption gives $M_J = \Phi_J B$ and $M_F = \Phi B$. Substitute these expressions and (16) into Proposition C.4. The target-noise term $a\Gamma_{JJ}a^\top$ appears in both risks and cancels from their difference. ∎

Here $\sigma^2 = \mathbb{E}[\|\xi_i\|^2 \mid \mathcal{D}]$ for $i \in F$ is the innovation energy of a value head, not variance per coordinate. The additional assumption $\Gamma_{JJ} = \sigma^2 I_{|J|}$ reduces the target-noise term to $\sigma^2\|a\|^2$, but is unnecessary for the gain formula. For heteroscedastic, uncorrelated fitting innovations, replace $\sigma^2\|w\|^2$ by $\sum_{i \in F} w_i^2 \sigma_i^2$. Equal fitting energies are used only in this simplification; the general identity has no such requirement. Zero conditional cross moments suffice; full conditional independence is not required.

C.5 WHY FEW OBSERVATIONS CAN SUFFICE: TARGET GEOMETRY AND REGULARIZATION

Let $\Phi = U_\Phi \operatorname{diag}(\varsigma_1, \ldots, \varsigma_{q_\Phi}) V_\Phi^\top$ be a thin SVD with positive singular values, where $q_\Phi = \operatorname{rank}(\Phi)$ and υ_k is column k of V_Φ. Set

$$P_\lambda = \Phi^\top (\Phi\Phi^\top + \lambda I_n)^{-1}\Phi = \sum_{k=1}^{q_\Phi} \frac{\varsigma_k^2}{\varsigma_k^2 + \lambda} \upsilon_k \upsilon_k^\top. \tag{19}$$

Then $w_J\Phi = x_J P_\lambda$. The matrix P_λ is a shrinkage operator, not an orthogonal projection when $\lambda > 0$. Writing $x_\perp = x_J(I - V_\Phi V_\Phi^\top)$, orthogonality gives

$$\|x_J - w_J\Phi\|^2 = \|x_\perp\|^2 + \sum_{k=1}^{q_\Phi} \left(\frac{\lambda}{\varsigma_k^2 + \lambda}\right)^2 (x_J \upsilon_k)^2, \tag{20}$$

$$\|w_J\|^2 = \sum_{k=1}^{q_\Phi} \frac{\varsigma_k^2}{(\varsigma_k^2 + \lambda)^2} (x_J \upsilon_k)^2. \tag{21}$$

To verify the second identity, substitute the SVD into (7) and use $U_\Phi^\top U_\Phi = I_{q_\Phi}$; the first follows by separating row-space and nullspace components.

These identities make precise what a small fitting set requires. In the method's configuration $n \le S = 16$, and even the largest diagnostic buffer in Table 11 has $n \le 64$, whereas $p = r + 1$ ranges from 162 to 938, and x_J aggregates the latent codes of hundreds or thousands of tokens. The target feature therefore generally lies outside the row space of Φ, and the term $\|x_\perp\|^2$ in (20) cannot be made zero. A positive gain then requires the residual map to carry little energy along the unobserved part of x_J. This is an assumption about the residual map that the fitting observations alone cannot verify, and Table 10 tests it empirically. Poorly observed directions trade ridge bias against noise amplification. Neither small feature mismatch nor small $\|w_J\|$ alone guarantees improvement, because the structural signal B and temporal covariance also matter.

Proposition C.6 (Minimax justification within a linear estimation class) *Consider $E_F = \Phi B + \Xi_F$, conditionally centered innovations satisfying $\Gamma_{FF} = \sigma^2 I_n$, and the noiseless target functional $x_J B$. Let $R_B > 0$, $\sigma^2 > 0$, and restrict to coefficient matrices $\|B\|_F \le R_B$. Among estimators wE_F whose weights depend on the design but not on the residual targets, the worst-case conditional risk is*

$$\sup_{\|B\|_F \le R_B} \mathbb{E}[\|x_J B - wE_F\|^2 \mid \mathcal{D}] = R_B^2 \|x_J - w\Phi\|^2 + \sigma^2 \|w\|^2. \tag{22}$$

Its unique minimizing weights are (7) with $\lambda = \sigma^2/R_B^2$.

Proof. Centering eliminates the signal–noise cross term, and the noise contribution is $\sigma^2\|w\|^2$. Let $u = x_J - w\Phi$. The inequality $\|uB\|^2 \le \|u\|^2\|B\|_F^2$ is sharp: if $u \ne 0$, choose $B = R_B(u^\top/\|u\|)b$ for any unit row vector b. If $u = 0$, the signal term is zero for every B. Taking the supremum gives (22). Divide by R_B^2 and apply Proposition C.2. ∎

This is a characterization of the ridge form, not a proof that the method's feature-scaled coefficient $\rho\|\Phi\|_F^2/n$ equals σ^2/R_B^2. It is also restricted to linear, design-dependent estimators under the stated covariance model, rather than all possible estimators. For the target residual sum d_J, its risk equals the noiseless-functional risk plus $a\Gamma_{JJ}a^\top$ when $a\Gamma_{JF} = 0$, in particular when $\Gamma_{JF} = 0$. Otherwise, the additional term $-2a\Gamma_{JF}w^\top$ also depends on the chosen weights, as in (11).

Why a recent residual mean is a different estimator. Let $\bar\phi_F = n^{-1}\sum_{i\in F}\phi_i$ and $\bar e_F = n^{-1}\sum_{i\in F} e_i$. For $A_J > 0$, define the attention-weighted residual mean $\bar e_{\alpha,J} = d_J/A_J$. A mean-only correction has the exact aggregate error

$$d_J - A_J\bar e_F = A_J(\bar e_{\alpha,J} - \bar e_F). \tag{23}$$

The tokenwise errors $e_j - \bar e_F$ need not be collinear and may cancel; using the same prediction for every token does not prevent cancellation. The aggregate error depends on the difference between the two means. Under a shared linear conditional mean, its conditional bias is

$$b_{\text{mean}} = (x_J - A_J\bar\phi_F)B, \tag{24}$$

whereas RF has bias $(x_J - w_J\Phi)B$. Mean-only correction is structurally exact precisely when $(x_J - A_J\bar{\phi}_F)B = 0$; equality of the two feature means is sufficient but not necessary if B has a nontrivial left nullspace. Regularized moment matching trades feature mismatch against the noise contribution of its chosen weights. Neither estimator dominates unconditionally. If $A_J = 0$, the target and both corrections vanish.

C.6 Exact finite-sample gain decomposition

Proposition C.7 (Realized gain, alignment, and reference invariance) *Fix the original residual arrays, features, attention weights, and RF weights. With $d_J = a_J E_J$ and $\hat{d}_J = w_J E_F$, the realized output gain is*

$$g_{\mathrm{obs}} = \|d_J\|^2 - \|d_J - \hat{d}_J\|^2 = 2\langle d_J, \hat{d}_J\rangle - \|\hat{d}_J\|^2. \tag{25}$$

Consequently RF reduces this squared error if and only if $2\langle d_J, \hat{d}_J\rangle > \|\hat{d}_J\|^2$. For nonzero d_J and $\hat{d}_J$, define $\zeta_J = \|\hat{d}_J\|/\|d_J\|$ and $\chi_J = \langle d_J, \hat{d}_J\rangle/(\|d_J\|\|\hat{d}_J\|)$. Then

$$\frac{g_{\mathrm{obs}}}{\|d_J\|^2} = 2\zeta_J\chi_J - \zeta_J^2. \tag{26}$$

For any reference matrix $B_\circ$, including a data-dependent one, writing $\widetilde{\Xi}_F = E_F - \Phi B_\circ$ and $\widetilde{\Xi}_J = E_J - \Phi_J B_\circ$ does not change g_{obs} when the original arrays and RF correction are fixed.

Proof. Expanding $\|d_J - \hat{d}_J\|^2$ gives (25); dividing by $\|d_J\|^2$ gives (26). For invariance, the identities

$$x_J B_\circ + a_J\widetilde{\Xi}_J = d_J, \qquad (x_J - w_J\Phi)B_\circ + a_J\widetilde{\Xi}_J - w_J\widetilde{\Xi}_F = d_J - \hat{d}_J$$

hold pointwise for every $B_\circ$. Squaring them recovers the same gain. ∎

If $\hat{d}_J = 0$, the gain is zero; if $d_J = 0$, it is $-\|\hat{d}_J\|^2$. No linear conditional-mean model or covariance assumption is used in this proposition. The normalized formula separates alignment with the required correction from its magnitude: positive alignment alone does not exclude overshooting.

Output aggregation and tokenwise errors. For several query heads, squared error of the concatenated head outputs is $\sum_h \|d_J^{(h)}\|^2$, and its gain is the sum of the headwise gains. Averaging attention weights across heads before taking a squared norm instead introduces cross-head cancellation and is a different objective. For a single head, the remaining tokenwise errors obey

$$\left\|\sum_{j\in J}\alpha_j\varepsilon_j^V\right\|^2 = \sum_{j\in J}\alpha_j^2\|\varepsilon_j^V\|^2 + 2\sum_{j<k\,;\,j,k\in J}\alpha_j\alpha_k\langle\varepsilon_j^V, \varepsilon_k^V\rangle. \tag{27}$$

Thus an unweighted tokenwise MSE and the required output error need not order two corrections identically. As a purely algebraic example, with two scalar residuals $(1, 1)$ and weights $(1/2, 1/2)$, replacing the remaining residuals by $(2, -2)$ increases mean tokenwise squared error from 1 to 4 but reduces output squared error from 1 to 0.

For errors before and after feedback, define

$$D_0 = \sum_{j\in J}\alpha_j^2\|e_j\|^2, \qquad D_1 = \sum_{j\in J}\alpha_j^2\|\varepsilon_j^V\|^2,$$
$$C_0 = \|d_J\|^2 - D_0, \qquad C_1 = \|d_J - \hat{d}_J\|^2 - D_1.$$

Then the gain also has the reference-free decomposition

$$g_{\mathrm{obs}} = (D_0 - D_1) + (C_0 - C_1). \tag{28}$$

This follows by subtracting the two instances of (27). The C_b terms are realized cross-token inner products, not covariances of conditional innovations. This distinction allows output improvement to be separated from the diagonal contribution without choosing a pilot mean model.

Aggregation across observations. Let v index combinations of window, query position, and query head, and stack the original errors and corrections into vectors $\boldsymbol{d}$ and $\widehat{\boldsymbol{d}}$. For nonzero stacked vectors, write

$$Z = \frac{\|\widehat{\boldsymbol{d}}\|}{\|\boldsymbol{d}\|}, \qquad X = \frac{\langle \boldsymbol{d}, \widehat{\boldsymbol{d}} \rangle}{\|\boldsymbol{d}\|\|\widehat{\boldsymbol{d}}\|}.$$

Applying Proposition C.7 to these vectors gives

$$1 - \frac{\sum_v \|d_v - \hat{d}_v\|^2}{\sum_v \|d_v\|^2} = 2ZX - Z^2. \tag{29}$$

An arithmetic mean of per-observation cosines or magnitude ratios cannot in general be substituted for X or Z. All these metrics concern the concatenated head outputs before the learned output projection; applying that projection induces a different quadratic metric.

The population identities have an algebraic counterpart that holds for arbitrary residual arrays, without a distributional assumption. Choose any reference coefficient matrix $B_\circ$ and define $\tilde{\xi}_i = e_i - \phi_i B_\circ$. For disjoint J, F, set

$$m_0 = x_J B_\circ, \qquad m_1 = (x_J - w\Phi)B_\circ, \qquad u_J = \sum_{j \in J} \alpha_j \tilde{\xi}_j, \qquad u_F = \sum_{i \in F} w_i \tilde{\xi}_i.$$

The realized squared-error gain admits the exact decomposition

$$\begin{aligned} g_{\text{obs}} &:= \|d_J\|^2 - \|d_J - wE_F\|^2 = g_{\text{diag}} + c_{\text{dep}} + c_{\text{mean}}, \\ g_{\text{diag}} &= \|m_0\|^2 - \|m_1\|^2 - \sum_{i \in F} w_i^2 \|\tilde{\xi}_i\|^2, \\ c_{\text{dep}} &= -\left(\|u_F\|^2 - \sum_{i \in F} w_i^2 \|\tilde{\xi}_i\|^2 \right) + 2\langle u_J, u_F \rangle, \\ c_{\text{mean}} &= 2\langle m_0 - m_1, u_J \rangle + 2\langle m_1, u_F \rangle. \end{aligned} \tag{30}$$

Indeed, $d_J = m_0 + u_J$ and $d_J - wE_F = m_1 + u_J - u_F$; expanding both squares proves the identity. No exact conditional-mean representation is needed for this equality. If $B_\circ$ is $\mathcal{D}$-measurable and $\phi_i B_\circ = \mu_i$ for all $i \in F \cup J$, then

$$\mathbb{E}[c_{\text{mean}} \mid \mathcal{D}] = 0, \qquad \mathbb{E}[c_{\text{dep}} \mid \mathcal{D}] = -w(\Gamma_{FF} - \Gamma_{FF}^{\text{diag}})w^\top + 2a\Gamma_{JF}w^\top.$$

Under these conditions the expected decomposition reduces to (13). Otherwise, the same algebraic identity still holds, with c_{mean} retaining the signal–remainder cross terms.

For clarity, set $Q_F = \sum_{i \in F} w_i^2 \|\tilde{\xi}_i\|^2$ and separate

$$c_{\text{dep}} = c_{FF} + c_{JF}, \qquad c_{FF} = -\big(\|u_F\|^2 - Q_F\big), \qquad c_{JF} = 2\langle u_J, u_F \rangle. \tag{31}$$

The term $2\langle m_1, u_F \rangle$ belongs to c_{mean}, not to c_{JF}. These are algebraic components relative to $B_\circ$; they are not reference-invariant measures of causal contribution. In particular, $B_\circ = 0$ makes $c_{\text{mean}} = 0$ without imposing any property on the underlying data. Their conditional covariance interpretation requires the conditional-mean conditions stated above.

Proposition C.8 (Orthogonality induced by a same-sample least-squares reference) *Let B_{LS} be an exact minimizer of the unregularized fitting objective $\|E_F - \Phi B\|_F^2$, and put $U_F = E_F - \Phi B_{\text{LS}}$. For every $\lambda > 0$ and every target feature sum x_J, the RF weights satisfy*

$$\Phi^\top U_F = 0, \qquad w_J U_F = 0. \tag{32}$$

If Φ has full row rank, then $U_F = 0$. These statements do not require independent residuals.

Proof. The first identity is the least-squares normal equation. The dual–primal identity gives

$$w_J = x_J(\Phi^\top\Phi + \lambda I_p)^{-1}\Phi^\top, \qquad w_J U_F = x_J(\Phi^\top\Phi + \lambda I_p)^{-1}\Phi^\top U_F = 0.$$

If Φ has full row rank, the coefficient matrix $\Phi^\top(\Phi\Phi^\top)^{-1}E_F$ attains zero fitting error, so every least-squares minimizer has $U_F = 0$. ∎

With this reference, terms involving $u_F = w_J U_F$ vanish by construction. Their smallness therefore does not establish weak temporal covariance of raw residuals or true conditional innovations. Without full row rank, U_F may be nonzero even though $u_F = 0$; then $c_{FF} = Q_F$ need not vanish. Full-row-rank interpolation can instead absorb all fitting-side variation into the reference map. An algebraic decomposition with such a reference remains valid, but its components are not an independently identified signal–noise decomposition. The actual RF fit still uses positive ridge regularization; the least-squares reference is only an analysis device.

C.7 Connections to the other components of LIVE-MLA

Attention-weighted conversion without an independence assumption. For nonnegative weights with $A_J = \sum_{j\in J} \alpha_j \le 1$, weighted Cauchy–Schwarz gives the deterministic bound

$$\left\| \sum_{j\in J} \alpha_j e_j \right\|^2 \le A_J \sum_{j\in J} \alpha_j \|e_j\|^2 \le \sum_{j\in J} \alpha_j \|e_j\|^2. \tag{33}$$

For $A_J > 0$, apply Jensen's inequality with weights α_j/A_J and multiply by A_J^2; the zero-mass case is immediate. Summing over queries and heads produces received-attention-mass weights for a value-reconstruction surrogate. Sequencewise normalization changes the relative weighting of sequences but not the within-sequence prioritization. This yields a value-side surrogate, not an identification of joint key–value fitting with exact output MSE. In general, an expected squared output contains cross-token second moments in addition to the α_j^2-weighted diagonal terms.

Key correction remains a different perturbation. Let $\boldsymbol{\ell}$ be reference attention logits, $\pi = \mathrm{softmax}(\boldsymbol{\ell})$, and $o = \sum_j \pi_j v_j$. These weights need not equal the method's α_j used above. Write the compared path as $\tilde{\boldsymbol{\ell}} = \boldsymbol{\ell} - \eta$ and $\tilde{v}_j = v_j - \varepsilon_j^V$. Its first-order decomposition is

$$o - \tilde{o} = \underbrace{\sum_j \pi_j \varepsilon_j^V}_{d_V^{\mathrm{out}}} + \underbrace{\sum_j \pi_j \eta_j (v_j - o)}_{d_K^{\mathrm{out}}} + \mathcal{R}, \tag{34}$$

$$\|\mathcal{R}\| \le M_V \|\eta\|_\infty^2 + 2M_\varepsilon \|\eta\|_\infty, \tag{35}$$

where $M_V = \max_j \|v_j\|$ and $M_\varepsilon = \max_j \|\varepsilon_j^V\|$. Thus the scale of the value vectors is explicit in the quadratic key-only remainder.

To prove the bound, let $\pi(u) = \mathrm{softmax}(\boldsymbol{\ell} - u\eta)$ for $u \in [0,1]$ and $\bar{\eta}(u) = \sum_j \pi_j(u)\eta_j$. Then

$$\pi_j'(u) = -\pi_j(u)(\eta_j - \bar{\eta}(u)), \qquad \pi_j''(u) = \pi_j(u)\left[(\eta_j - \bar{\eta}(u))^2 - \mathrm{Var}_{\pi(u)}(\eta)\right].$$

Since $\mathrm{Var}_{\pi(u)}(\eta) \le \|\eta\|_\infty^2$, $\|\pi''(u)\|_1 \le 2\|\eta\|_\infty^2$. Taylor's integral remainder therefore has ℓ_1 norm at most $\|\eta\|_\infty^2$. Also $\|\pi(1) - \pi(0)\|_1 \le 2\|\eta\|_\infty$. Expanding $\tilde{o} = \sum_j \pi_j(1)(v_j - \varepsilon_j^V)$ and using $\sum_j \pi_j(v_j - o) = 0$ gives (34) and (35). The bound holds for finite logit perturbations; the first-order approximation is useful when its remainder is small. The leading squared output error includes $2\langle d_V^{\mathrm{out}}, d_K^{\mathrm{out}}\rangle$, so neither key-space reconstruction MSE nor a separate key-only norm determines the joint change in output error.

For a score path that applies position-dependent rotary maps after key reconstruction, write q_t^{pre} and $\varepsilon_j^{K,\mathrm{pre}} = k_j^{\mathrm{pre}} - \tilde{k}_j^{\mathrm{pre}}$ as rows after model-specific normalization and before RoPE. Let R_t and R_j denote the rotary matrices. With the same query in the two paths, the effective score error is

$$\eta_{tj} = \frac{q_t^{\mathrm{pre}} \mathsf{R}_t^\top \mathsf{R}_j (\varepsilon_j^{K,\mathrm{pre}})^\top}{\sqrt{d_k}}. \tag{36}$$

Under this score parameterization, the effective key error depends on query alignment, normalization, and relative position. The formula is a conditional specialization of the logit perturbation η, not a restriction on the value-path analysis. The ridge estimator analyzed above corrects values with the attention weights held fixed; Appendix C.8 evaluates key correction empirically.

Allocating a fixed byte budget. Consider a configuration that retains at most W preceding tokens with exact keys and values. Together with the current token, at most $W + 1$ tokens use exact representations for each query. Consider an additional budget equal to m stored exact tokens. Option (a) retains the set $\mathcal{N}$ of the m most recent compressed tokens in exact form, which raises W to $W + m$. Option (b) keeps W unchanged and spends the budget on Residual Feedback. With S value-residual rows stored in BF16, the state costs $2SGd_v$ bytes. This equals the cost of $S/2$ exact key-value tokens when $d_k = d_v$. Our configuration with $S = 16$ therefore corresponds to $m = 8$.

Hold the attention weights fixed and consider only value error. Let

$$d = \sum_{j \in \mathcal{O}_t} \alpha_j e_j, \qquad d_{\mathcal{N}} = \sum_{j \in \mathcal{N}} \alpha_j e_j, \qquad \hat{d} = \sum_{j \in \mathcal{O}_t} \alpha_j \hat{e}_j.$$

Option (a) removes the value error of $\mathcal{N}$ exactly, and option (b) subtracts $\hat{d}$. The remaining squared output errors are

$$U_{\text{win}} = \|d - d_{\mathcal{N}}\|^2, \qquad U_{\text{RF}} = \|d - \hat{d}\|^2.$$

Proposition C.9 (Allocation criterion) *$U_{\text{RF}} < U_{\text{win}}$ if and only if*

$$\underbrace{2\langle d, d_{\mathcal{N}} \rangle - \|d_{\mathcal{N}}\|^2}_{\textit{gain of a larger window}} < \underbrace{2\langle d, \hat{d} \rangle - \|\hat{d}\|^2}_{\textit{gain of Residual Feedback}} . \tag{37}$$

Proof. For any correction x, $\|d - x\|^2 = \|d\|^2 - (2\langle d, x \rangle - \|x\|^2)$. Apply this with $x = d_{\mathcal{N}}$ and $x = \hat{d}$ and cancel $\|d\|^2$. ∎

Both sides are realized gains in the sense of Proposition C.7, applied to two corrections of the same target. The gain of a larger window need not be positive: making the values of $\mathcal{N}$ exact removes errors that may have partly canceled errors elsewhere in $\mathcal{O}_t$. The criterion retains the cross-token terms that a triangle-inequality proxy discards, and Table 12 shows that these terms carry most of the gain of Residual Feedback in the diagnostic configuration. The next paragraph gives a coarser norm-level version in terms of attention mass. The criterion does not represent the change in key scores caused by a larger exact window, so we verify the comparison with the language modeling loss in Appendix C.8.

C.8 Empirical evaluations

The observations below accompany the mathematical results without replacing their assumptions. Layerwise diagnostics and end-to-end benchmarks are reported separately because they concern different quantities and configurations.

Key correction ablation. Table 8 compares value-only and key-plus-value Residual Feedback in the final configuration without recovery training. The value-only column reproduces the training-free perplexity reported in Table 1, while the RF-off column removes the correction at the same latent width and is therefore not one of the budget-matched configurations. Value-only feedback lowers the perplexity on both models. Key-plus-value feedback is model-dependent. It helps Llama-3.1-8B-Instruct but raises the perplexity of Qwen3-8B-Base far above the uncorrected model. The lower block splits the predicted key correction into the component aligned with the true key residual and the orthogonal component. The aligned component reduces the loss on both models, while the orthogonal component, a logit error η in the sense of (34), adds 2.11 nats on Qwen3-8B-Base and is negligible on Llama-3.1-8B-Instruct.

The two models also differ in how the key enters the score. Qwen3 normalizes each key head before RoPE, and the compressed-token score multiplies the content term by the normalization factor γ_j stored from the exact key. A correction applied to the reconstructed content key is therefore not renormalized, so an error orthogonal to the true residual can change the logit scale directly. We state this as a possible explanation for the model dependence; Table 8 does not isolate it.

Byte-matched window comparison. Table 9 compares the final configuration with $W = 15$ and $S = 16$ BF16 residual rows against a byte-matched no-feedback configuration with $W = 23$ at the same latent width, the two options of Proposition C.9. Under 87.5% memory reduction, Residual

Table 8: Value-only and key-plus-value Residual Feedback without training under 87.5% memory reduction, with the feedback refitted for every query on the 16 tokens that have just left the 15-token sliding window. The upper block reports WikiText-2 perplexity using 146 windows for Qwen3-8B-Base and 141 windows for Llama-3.1-8B-Instruct, at the same latent width r in every column (161 and 169). The lower block reports the NLL change when a component of the predicted key correction is added, averaged over the same windows.

Model, Memory reduction	RF off	value only	key + value
Qwen3-8B-Base, 87.5%	16.84	13.90	121.83
Llama-3.1-8B-Instruct, 87.5%	19.59	15.29	13.13
	full key correction	aligned component	orthogonal component
Qwen3-8B-Base, 87.5%	$+2.171$	-0.133	$+2.109$
Llama-3.1-8B-Instruct, 87.5%	-0.152	-0.173	-0.040

Table 9: Final configuration ($W = 15$, $S = 16$) minus the byte-matched no-feedback configuration with $W = 23$ at the same latent width on WikiText-2. ΔNLL is averaged over windows with 95% bootstrap intervals. The final column reports the number of windows in which Residual Feedback is better. Negative values favor Residual Feedback.

Model	Mem.	Length	ΔNLL [95% interval], windows won
Qwen3-8B-Base	-87.5%	2K	-0.049 [-0.053, -0.045], 142/146
	-75%	2K	-0.017 [-0.019, -0.016], 136/146
	-50%	2K	$+0.0002$ [-0.0003, $+0.0007$], 70/146
Llama-3.1-8B-Instruct	-87.5%	2K	-0.088 [-0.092, -0.084], 141/141
	-75%	2K	-0.041 [-0.043, -0.039], 141/141
	-50%	2K	$+0.0008$ [$+0.0004$, $+0.0013$], 57/141

Feedback improves NLL on both models. The margin is smaller under 75% reduction. Under 50% reduction, the two configurations are nearly identical on Qwen3-8B-Base, while the larger window is slightly better on Llama-3.1-8B-Instruct. These results support Residual Feedback as a useful allocation of memory in the reported settings. They do not assert uniform superiority across memory-reduction levels or identify an NLL change with a layerwise MSE change. On the 146 Qwen3-8B-Base windows under 87.5% memory reduction at a 2K context length, storing the residual rows in BF16 instead of FP32 changes the NLL by 0.0000 nats with a 95% interval of $[-0.0003, +0.0003]$.

Transfer to older tokens. Table 10 applies the correction only to compressed tokens in a given distance band while keeping the fit unchanged. With $W = 15$, $S = 16$, and 87.5% memory reduction, the fitting set covers distances 16 to 31, so every band starting at distance 32 is disjoint from F, as in Proposition C.4. On both evaluated models, every such band lowers NLL, including distances of at least 512. At distances 32 to 127, the changes are -0.091 and -0.137 nats/token for Qwen3-8B-Base and Llama-3.1-8B-Instruct. Beyond 512, they are -0.046 and -0.051. Thus the usefulness of recent observations is not confined to the tokens on which the mapping is fit. The bands differ in size and in the attention mass they receive, so their entries are not per-token comparisons. They are also not additive, because corrections propagate through subsequent layers.

Replacing the fitted mapping with the mean recent residual raises the NLL on every model by 0.64 to 2.52 nats, while observed pairing outperforms shuffled pairing in every model. Shuffling the pairing between codes and residuals retains 47% and 50% of the all-token gain on Qwen3-8B-Base and Llama-3.1-8B-Instruct, respectively. For a uniform random permutation matrix Π, the exact identity

$$\mathbb{E}_\Pi[w_J \Pi E_F] = (w_J \mathbf{1}_n)\bar{e}_F, \qquad \bar{e}_F = n^{-1}\mathbf{1}_n^\top E_F \tag{38}$$

follows from $\mathbb{E}[\Pi] = \mathbf{1}_n\mathbf{1}_n^\top/n$. Shuffling removes the row pairing but preserves the residual collection. Its expected correction is therefore $(w_J\mathbf{1}_n)\bar{e}_F$, whereas the mean-only control applies $A_J\bar{e}_F$, so the two controls differ through the ratio $w_J\mathbf{1}_n/A_J$ and through the variation of a single permutation around its expectation. The identity concerns correction vectors, not expected NLL.

Table 10: Change in NLL relative to no feedback when correction is applied only to compressed tokens in the given distance range. Results use the final configuration with $W = 15$, $S = 16$ BF16 residual rows, and 87.5% memory reduction. The evaluation uses 146 windows for Qwen3-8B-Base and 141 windows for Llama-3.1-8B-Instruct. The fitting set covers distances 16 to 31.

Correction applied to	Qwen3-8B-Base	Llama-3.1-8B-Instruct
all compressed tokens	-0.154	-0.243
distance 16 to 31	-0.079	-0.115
distance 32 to 127	-0.091	-0.137
distance 128 to 511	-0.069	-0.101
distance at least 512	-0.046	-0.051
mean residual only	$+0.638$	$+2.52$
shuffled pairing	-0.072	-0.121

Headwise diagnostic setting. The offline diagnostic uses a Qwen3-8B-Base checkpoint trained with Residual Feedback in the final configuration under 87.5% memory reduction after 1.5B recovery-training tokens. The evaluation uses 32 tokenized windows of 2,048 tokens. It includes 12 query positions per window given by $t = 534, 662, \ldots, 1942$, all 32 query heads with 8 shared key-value heads, and layers $0, 8, 16, 24, 32$.

Each query uses exact keys and values for the current token and at most $W = 15$ preceding tokens. The fitting and target sets are

$$F_t = [t - W - S, t - W), \qquad J_t = [t - 511, t - 128).$$

These intervals are half-open. The target set contains 383 tokens at distances 129 to 511, and $F_t \cap J_t = \emptyset$. The checkpoint and latent width remain fixed while $S \in \{4, 8, 16, 32, 64\}$ varies. This is not a matched-memory sweep. The default method uses $S = 16$, while the larger values are included to examine the effect of increasing the fitting buffer. All results use $\rho = 10^{-3}$.

For this diagnostic configuration, reconstructed projections define logits $\widetilde{\ell}_i$. The attention weights on J_t are

$$\beta_j = \frac{e^{\widetilde{\ell}_j}}{\sum_{i \in J_t \cup \mathcal{R}_t \cup \{t\}} e^{\widetilde{\ell}_i}}, \qquad j \in J_t. \tag{39}$$

All reported percentages use this source set and these recorded activations. They are not identified with the full causal attention-output error or with the end-to-end NLL results reported elsewhere.

For observation v, which indexes the window, query position, and query head, let d_v^β and $\widehat{d}_v^\beta$ denote the residual sum and correction under these weights. The pooled output-error reduction is

$$G_\beta = 100 \left(1 - \frac{\sum_v \left\| d_v^\beta - \widehat{d}_v^\beta \right\|^2}{\sum_v \left\| d_v^\beta \right\|^2} \right). \tag{40}$$

Squared errors are summed over heads before computing the ratio. Attention weights are not averaged across heads. Uncertainty is estimated using 2,000 paired percentile-bootstrap resamples of entire windows with seed 20260924. The intervals are pointwise and are not simultaneous across the configuration grid.

Functional recovery and large-buffer failures. Table 11 shows positive gains in every measured layer for $S \leq 32$. At $S = 64$, layer 24 instead has a gain of -10.35% with interval $[-17.18, -3.38]$, and layer 16 has a gain of -1.93% with interval $[-7.29, 3.38]$. Across the 25 grid points, 23 intervals are entirely positive, one is entirely negative, and one contains zero. The observations therefore

Table 11: Output-error reduction G_β in percent across layers and residual buffer sizes. Positive values indicate improvement. The complete layer and buffer-size grid includes the large-buffer failures.

Layer	$S = 4$	$S = 8$	$S = 16$	$S = 32$	$S = 64$
0	$+65.64$	$+78.48$	$+86.91$	$+91.25$	$+92.73$
8	$+10.11$	$+14.37$	$+17.11$	$+17.13$	$+7.15$
16	$+9.95$	$+13.07$	$+14.03$	$+13.04$	-1.93
24	$+11.53$	$+14.34$	$+16.51$	$+12.86$	-10.35
32	$+14.52$	$+19.37$	$+23.62$	$+24.18$	$+10.92$

Table 12: Headwise diagnostic results at $S = 64$ for the same checkpoint. This buffer is larger than the default $S = 16$. Positive token gain denotes a reduction in unweighted tokenwise MSE. The diagonal and cross-token contributions sum to G_β up to rounding.

Layer	G_β (%)	95% interval	Token gain (%)	Diagonal	Cross-token
0	$+92.73$	$[91.34, 94.04]$	$+29.30$	$+1.43$	$+91.30$
8	$+7.15$	$[1.42, 12.52]$	-53.40	-9.69	$+16.84$
16	-1.93	$[-7.29, 3.38]$	-57.41	-8.92	$+7.00$
24	-10.35	$[-17.18, -3.38]$	-48.09	-14.97	$+4.62$
32	$+10.92$	$[7.74, 14.05]$	-37.54	-12.39	$+23.31$

support useful recovery in this configuration without implying improvement for every layer or buffer size.

At $S = 64$, output error decreases in layers 0, 8, and 32. In layers 8 and 32, this occurs even though uniformly averaged tokenwise reconstruction error increases, as shown in Table 12. This is the separation described by (27). Estimating the required attention-weighted functional and reconstructing every token are different objectives.

Output geometry rather than tokenwise MSE alone. The reference-free decomposition in Table 12 shows that the diagonal contribution is negative at layers 8, 16, 24, and 32, while the change in cross-token inner products is positive in every measured layer. At layer 8, the two contributions give $-9.69 + 16.84 = 7.15$ percentage points. These values are realized inner products between error vectors rather than estimated covariances of the underlying conditional innovations. The results show that improved tokenwise reconstruction is neither necessary nor sufficient for reducing the attention-weighted output error.

Interpretation of reference-dependent terms. An external ridge reference fitted on the first 1,024 tokens of separate windows 60 and 61 gives $c_{\mathrm{dep}}/g_{\mathrm{obs}}$ values of 16.27 and 13.49 at $S = 64$ in layers 8 and 32, where the observed gain is positive. In layers 16 and 24, the corresponding ratios are -70.43 and -17.98. The observed gain is near zero or negative in these layers, so these ratios should not be interpreted as positive shares of the gain.

The original residuals and the Residual Feedback correction remain unchanged under this decomposition. Cross terms therefore cannot be omitted from the realized decomposition relative to this reference. The reported ratios are not causal shares and do not identify the covariance of the underlying conditional innovations. A reference error may itself contain predictable structure.

A same-fitting-set least-squares reference instead gives $w_J U_F$ numerically equal to zero and $|c_{\mathrm{dep}}| \leq 2 \times 10^{-11}$. This agrees with Proposition C.8, which predicts the result from least-squares orthogonality. The general covariance identity therefore remains the primary risk statement, while the diagonal formula remains a special case under its stated assumptions.

Conclusion of the analysis. The resulting explanation has three parts. Sequence conditioning can make residual structure predictable even when a global decoder satisfies an orthogonality condition. RF uses that structure to estimate an attention-weighted functional, for which the relevant observation geometry can be much smaller than the full residual-map dimension. Its output gain

is then determined by residual alignment, regularization, and the signed cross terms retained in the exact risk and finite-sample decompositions. The empirical observations illustrate these distinctions within their stated configurations; the mathematical claims retain the conditions under which they were proved.