TEMPO: TEMPORAL ERROR-MODULATED PRECISION OVERLAYS FOR RECURSIVE TRANSFORMER QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recursive Transformers repeatedly reuse the same parameters across computational loops, yet the resulting quantization errors are neither uniform nor equally consequential across recurrent roles. We introduce TEMPO, a post-training quantization framework that models this asymmetry through Temporal Error-Modulated Precision Overlays. TEMPO retains a shared low-bit weight backbone and augments selected weight tiles with sparse INT2 residual planes whose contributions are controlled by compact temporal codes. Instead of ranking corrections by local reconstruction error, TEMPO estimates how loop-specific perturbations propagate through the recurrent trajectory and influence the final prediction. This terminal-sensitivity signal drives temporal-code assignment and residual selection, while a global budget allocator concentrates additional bits on corrections with the largest expected distortion reduction. The resulting representation separates globally shared quantized weights from lightweight loop-dependent overlays, preserving the parameter efficiency of recursive architectures without duplicating per-loop weights. TEMPO provides a compact, trajectory-aware mechanism for correcting recurrent quantization error and exposes temporal residual structure as a first-class compression primitive.

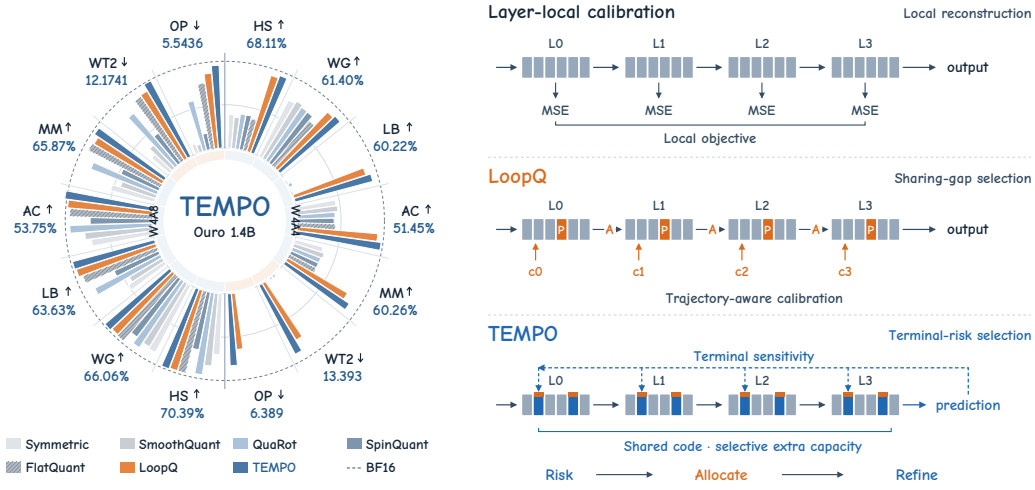


Figure 1: **Overview of TEMPO. Left:** Quantization results on **Ouro 1.4B** under **W4A4** and **W4A8**.The radial plot summarizes downstream accuracy and language-modeling perplexity, with all spokes normalized to the BF16 baseline so that larger radius always indicates better performance. TEMPO consistently delivers the best overall low-bit performance. **Right:** Comparison of quantization philosophies for recursive models. Unlike prior approaches based on local reconstruction, sharing-gap selection, or loop-wise adaptation, TEMPO estimates error importance through *terminal sensitivity* and allocates extra quantization capacity *selectively* to terminally high-risk locations.

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.

1 INTRODUCTION

Recursive Transformers achieve high parameter efficiency by reusing identical parameters across computational loops, offering a scalable alternative to deepening architectures with independent layer stacks Zhu et al. (2025b). This structural weight reuse, however, alters how quantization noise propagates through the network. A quantized perturbation injected into early recurrent states does not remain isolated; it is repeatedly processed across subsequent iterations, compounding through the recurrent trajectory before reaching the final prediction. Consequently, post-training quantization heuristics established for feedforward Transformers fail to maintain fidelity when applied to recursive topologies Lin et al. (2024); Ashkboos et al. (2024a); Liu et al. (2025a).

Existing post-training quantization methods predominantly follow a *local reconstruction* paradigm, mitigating precision loss by minimizing layer-wise or block-wise approximation errors in weights, activations, or intermediate outputs Frantar et al. (2022); Shao et al. (2024); Ma et al. (2024); Sun et al. (2025). Recent recursive compression techniques attempt to accommodate dynamic activations through loop-conditioned scaling factors or step-wise calibration Fang et al. (2026a). Despite their varied designs, these approaches rely on a common premise: *local approximation fidelity serves as a reliable proxy for downstream prediction accuracy*.

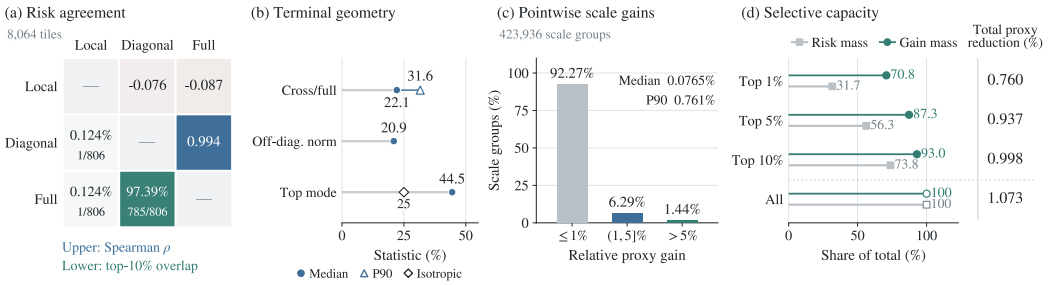


Figure 2: **Terminal risk and selective quantization capacity.** (a) Breakdown of local proxies against terminal prediction risk: local projection error exhibits negligible rank correlation with terminal risk, whereas terminal sensitivity aligns with the full trajectory ranking even prior to modeling cross-loop coupling. (b) Structural distribution of the terminal-risk proxy, contrasting cross-loop P90 metrics against isotropic baselines. (c) Performance returns from fixed-code loop-conditioned scaling. (d) Baseline-risk and recoverable-gain shares plotted along a common percentage axis, demonstrating that distortion recovery concentrates within a sparse subset of critical tiles.

As revealed in Figure 2(a), this foundational premise breaks down in recursive networks. Local projection errors exhibit near-zero Spearman rank correlation with the actual terminal-risk ordering. Conversely, introducing a terminal sensitivity metric aligns with the true trajectory ranking, even when evaluated prior to modeling fine-grained cross-loop interactions. This empirical discrepancy exposes a limitation in prior methodologies: *the consequence of a quantization error is governed by its downstream trajectory along recurrent unrolling rather than its isolated local magnitude*. In practice, weight tiles that appear difficult to quantize under isolated Euclidean metrics may exert negligible influence on the final prediction, whereas subtle distortions at sensitive recurrent phases cause catastrophic trajectory drift.

This insight dictates a rethinking of how quantization capacity should be spent. As characterized in Figure 2(b–d), cross-loop dependencies govern the exact magnitude of terminal degradation, yet the recoverable representation margin concentrates within a small fraction of high-risk weight regions. Uniformly scaling precision or expanding loop-specific parameters across the entire network introduces memory bandwidth overhead while yielding marginal fidelity returns. We therefore reframe recursive-model quantization as a *terminal-risk-driven resource allocation problem*: isolate parameter perturbations that compromise final representations, and allocate additional representation capacity exclusively to those coordinates.

To this end, we introduce **TEMPO** (**T**emporal **E**rror-**M**odulated **P**recision **O**verlays), a trajectory-aware quantization framework for recursive Transformers. TEMPO avoids uniform loop-wise adaptation in favor of structured, budget-constrained precision augmentation. Architecturally, TEMPO preserves a globally shared low-bit integer backbone across all recurrent steps, augmenting high-risk weight tiles with compact INT2 residual planes governed by lightweight temporal modulation

codes. Algorithmic execution follows an analytical terminal sensitivity metric: it evaluates the propagation trajectory of step-specific errors to guide both residual plane placement and temporal code calibration. By confining auxiliary precision to locations with the highest expected distortion reduction, TEMPO suppresses trajectory drift while avoiding the memory footprint of per-loop weight replication. We evaluate TEMPO on recursive architectures including Ouro 1.4B Zhu et al. (2025b) under W4A4 and W4A8 precision regimes. Compared with post-training baselines, TEMPO recovers full-precision zero-shot accuracy and language modeling perplexity across reasoning tasks. Because TEMPO retains uniform compute paths on the shared weight backbone and restricts temporal adaptations to tile-aligned residual vectors, it avoids memory bandwidth fragmentation, thereby offering the potential for improved inference throughput.

Our primary contributions are summarized as follows:

- We demonstrate the failure of the local reconstruction paradigm in recursive Transformers, showing that local approximation error is decoupled from end-to-end terminal degradation due to compounding recurrent dynamics.
- We reformulate recursive quantization as a terminal-risk-driven capacity allocation problem, showing that precision overhead must be concentrated within a sparse subset of trajectory-critical coordinates.
- We propose **TEMPO**, a hardware-compatible quantization framework combining shared low-bit weight tensors with sparse INT2 temporal residual planes, achieving near-lossless 4-bit recursive inference.

2 Related Works

Post-training quantization for LLMs. Post-training quantization (PTQ) compresses LLMs by mapping floating-point weights and activations to low-bit representations. GPTQ Frantar et al. (2022) optimizes weight rounding via inverse-Hessian approximations, while SmoothQuant Xiao et al. (2023b) migrates activation outliers to weights through mathematically equivalent scaling. AWQ Lin et al. (2024) further protects salient weight channels using activation magnitudes. Subsequent schemes improve quantizability by rotating representation spaces to suppress outliers, employing fixed orthogonal matrices in QuaRot Ashkboos et al. (2024a), learned rotations in SpinQuant Liu et al. (2025a), or learned affine transformations in FlatQuant Sun et al. (2025). While effective on feedforward architectures, these methods operate under an isolated layer-wise reconstruction objective; they minimize local output discrepancy under the assumption that preserving individual layer representations translates directly into global model quality.

Recursive architectures and recurrent quantization. Looped language models recycle Transformer layers across recurrent iterations, expanding effective depth while keeping parameter volume constant Zhu et al. (2025b). Quantizing such recurrent architectures introduces unique challenges because identical weight parameters are repeatedly traversed by evolving activation trajectories. LoopQ Fang et al. (2026a) addresses this setting by introducing loop-aware scaling, selective equivalent transformations, and calibration across recurrent unrolling. However, LoopQ and related approaches remain confined to uniform adaptation, attempting to reconcile loop-wise activation shifts through global transformation matrices or step-specific scale factors applied across all channels.

In contrast, our work demonstrates that quantization error propagation in looped networks is spatially sparse and governed by terminal prediction risk. Rather than applying uniform transformations to counteract distribution shifts, TEMPO allocates discrete representation capacity selectively to trajectory-critical coordinates, reconciling the parameter efficiency of shared weights with the requirements of recurrent stability.

3 Method

3.1 Framework Overview

Consider a recursive block whose projection weight $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ is reused across T recurrent iterations, generating states $y_t = Wx_t$ for $t \in \{1, \ldots, T\}$. Standard post-training quantization

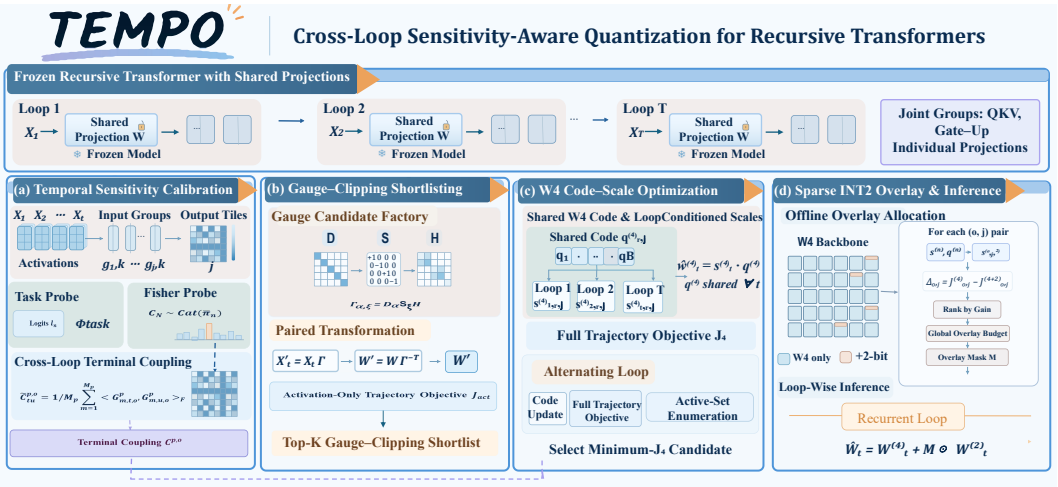


Figure 3: Overview of TEMPO. The framework consists of (a) terminal-sensitivity calibration, (b) gauge–clipping selection, (c) W4 code-scale optimization, and (d) sparse INT2 residual allocation for recursive-loop inference.

minimizes local layer-wise error, ignoring the compounding of stationary perturbations along the recurrent trajectory. To ensure stability without duplicating weights per loop, TEMPO decomposes the effective weight at iteration t as:

$$W_t = \hat{W}_{\text{base}} + c_t \odot \Delta W_{\text{res}}, \tag{1}$$

where $\hat{W}_{\text{base}}$ denotes a globally shared 4-bit integer backbone, ΔW_{res} represents a structured sparse INT2 residual plane, and c_t is a compact step-specific modulation vector.

3.2 Terminal-Sensitivity Calibration

TEMPO constructs a terminal-sensitivity surrogate to guide representation search, shared-code optimization, and residual precision allocation, using probes evaluated after the final recurrent iteration.

Let $X^p_{t,j} \in \mathbb{R}^{N_p \times B}$ denote calibration activations for input group j of projection group p at iteration t, with N_p aligned calibration positions and group size B. We record the uncentered cross-loop second moments:

$$H^{xx}_{p,j,tu} = \frac{1}{N_p} \left(X^p_{t,j}\right)^{\mathsf{T}} X^p_{u,j}, \qquad t, u \in \{1, \ldots, T\}. \tag{2}$$

On the full-precision calibration trajectory, we construct scalar probes Φ_m: next-token cross-entropy probes on calibration targets and categorical Fisher probes with targets sampled from the predictive distribution. For projection outputs $Y^p_{t,o} \in \mathbb{R}^{N_p \times |o|}$ in output tile o, we compute $G^p_{m,t,o} = \partial \Phi_m / \partial Y^p_{t,o}$ and form the cross-loop gradient coupling:

$$\widetilde{C}^{p,o}_{tu} = \frac{1}{M_p} \sum_{m=1}^{M_p} \left\langle G^p_{m,t,o}, G^p_{m,u,o} \right\rangle_F, \tag{3}$$

where M_p is the number of probes.

For local additive projection errors $E^p_{t,o}$ evaluated on the reference trajectory, consider the mean squared linearized probe response:

$$\begin{aligned} \mathcal{S}^{p,o}_{\text{probe}}(E) &= \frac{1}{M_p} \sum_{m=1}^{M_p} \left(\sum_{t=1}^{T} \left\langle G^p_{m,t,o}, E^p_{t,o} \right\rangle_F \right)^2 \\ &\approx \frac{1}{N_p |o|} \sum_{t,u=1}^{T} \widetilde{C}^{p,o}_{tu} \left\langle E^p_{t,o}, E^p_{u,o} \right\rangle_F. \end{aligned} \tag{4}$$

For numerical stability, each $\widetilde{C}^{p,o}$ is symmetrized, diagonally damped, and projected onto the positive semidefinite cone. All resulting matrices are divided by a common global maximum to obtain $C^{p,o}$. Subsequent objectives additionally omit cross-input-group, cross-tile, and cross-projection products for tractable optimization. The resulting objective is a probe-sensitivity surrogate, not a direct estimate of changes in task loss or downstream accuracy.

3.3 Terminal-Risk-Aware Representation Search

Before discrete code assignment, TEMPO optimizes an equivalent transformation space to suppress quantization sensitivity across recurrent unrolling.

For each projection group, we apply a full-rank, group-aligned gauge transformation:

$$\Gamma = DSH, \tag{5}$$

where D is a positive diagonal rescaling matrix, $S \in \{-1, +1\}^{B\times B}$ is a diagonal sign-flip matrix, and H denotes a normalized block-diagonal Hadamard matrix whose block width matches the quantization group size B. The gauge transforms both activations and projection weights:

$$X'_t = X_t\Gamma, \qquad W' = W\Gamma^{-\mathsf{T}}, \qquad X'_t(W')^{\mathsf{T}} = X_tW^{\mathsf{T}}. \tag{6}$$

While equivalent in floating-point precision, different gauge configurations reshape the spatial distribution of activation outliers and balance cross-channel dynamic range across unrolled iterations.

Following gauge projection, TEMPO executes per-token activation quantization on X'_t for each input group, using step-dependent clipping thresholds calibrated on activation quantiles. Let $\widetilde{X}_{t,j} = \mathcal{Q}(X'_{t,j})$ denote the quantized-and-dequantized activation corresponding to the transformed input $X'_{t,j}$. Because running full combinatorial weight optimization across all candidate transformations incurs prohibitive compute overhead, we introduce a decoupled screening phase based on activation-only perturbations. For output row r, input group j, and loop step t, the localized activation distortion is:

$$e^{\mathrm{act}}_{t,r,j} = \left(\widetilde{X}_{t,j} - X'_{t,j}\right) w'_{r,j}. \tag{7}$$

where $w'_{r,j}$ represents the row slice of the transformed weight tensor W'.

Rather than ranking candidate configurations by isolated reconstruction loss, TEMPO evaluates them using the trajectory-level objective:

$$\mathcal{J}_{\mathrm{act}} = \sum_o \sum_{r\in o} \sum_j \sum_{t,u=1}^{T} \frac{C^o_{tu}}{N_p} \left\langle e^{\mathrm{act}}_{t,r,j}, e^{\mathrm{act}}_{u,r,j} \right\rangle. \tag{8}$$

where the inner products are weighted by the cross-loop terminal coupling coefficients C^o_{tu} defined in Eq. equation 3. We conduct coordinate descent over clipping thresholds for each candidate gauge, accepting updates that decrease $\mathcal{J}_{\mathrm{act}}$. Only the Pareto-optimal gauge and clipping configurations under $\mathcal{J}_{\mathrm{act}}$ are forwarded to the subsequent W4 optimization stage, pruning the search space with minimal calibration overhead.

3.4 Shared-Code W4 Optimization

For each shortlisted representation, TEMPO uses a shared INT4 code across recurrent iterations with non-negative loop-conditioned scales. We work within a fixed projection group p and suppress its index where unambiguous:

$$\widehat{w}^{(4)}_{t,r,j} = s^{(4)}_{t,r,j} q^{(4)}_{r,j}, \qquad q^{(4)}_{r,j} \in \{-8, \ldots, 7\}^B, \qquad s^{(4)}_{t,r,j} \geq 0. \tag{9}$$

The integer codes are stored once, while compact scales accommodate loop-dependent activation statistics.

Let $w'_{r,j}$ be the gauge-transformed full-precision weight vector. Using the quantized-and-dequantized calibration activations $\widetilde{X}_{t,j}$, define the local projection error

$$e^{(4)}_{t,r,j} = \widetilde{X}_{t,j}\widehat{w}^{(4)}_{t,r,j} - X'_{t,j}w'_{r,j}. \tag{10}$$

The shared W4 configuration is optimized using the block-separable calibration surrogate

$$\mathcal{J}_4 = \sum_o \sum_{r \in o} \sum_j \frac{1}{N_p} \sum_{t,u=1}^{T} C_{tu}^o \left\langle e_{t,r,j}^{(4)}, e_{u,r,j}^{(4)} \right\rangle. \tag{11}$$

The full-model calibration objective sums these contributions over projection groups.

TEMPO initializes the shared codes using a coupling-weighted GPTQ-style rounding procedure. Cross-loop activation moments provide an initialization metric; the subsequent optimization is defined by Eq. equation 11, including activation quantization. The corresponding statistics are detailed in Appendix B.1.

For fixed codes, let $\mathbf{s}_{r,j} = [s_{1,r,j}^{(4)}, \ldots, s_{T,r,j}^{(4)}]^\mathsf{T}$. The loop-conditioned scales solve

$$\min_{\mathbf{s}_{r,j} \geq 0} \quad \mathbf{s}_{r,j}^\mathsf{T} A_{r,j} \mathbf{s}_{r,j} - 2\mathbf{b}_{r,j}^\mathsf{T} \mathbf{s}_{r,j}. \tag{12}$$

For a fixed row $r \in o$ and input group j, define $a_t = \widetilde{X}_{t,j} q_{r,j}^{(4)}$ and $y_t = X'_{t,j} w'_{r,j}$. Expanding Eq. equation 11 gives

$$(A_{r,j})_{tu} = \frac{C_{tu}^o}{N_p} a_t^\mathsf{T} a_u, \qquad (\mathbf{b}_{r,j})_t = \frac{1}{N_p} \sum_{u=1}^{T} C_{tu}^o a_t^\mathsf{T} y_u. \tag{13}$$

These coefficients require quantized-activation and mixed activation moments, rather than solely the original moments H^{xx}. Since C^o is positive semidefinite, $A_{r,j}$ is positive semidefinite and the fixed-code scale problem is convex. The derivation and compatible discrete updates are provided in Appendix B.

3.5 Sensitivity-Guided Precision Allocation

TEMPO allocates a limited number of INT2 residual overlays according to their reduction in the same calibration surrogate, rather than uniformly increasing weight precision. With the W4 backbone fixed, the augmented reconstruction is

$$\widehat{w}_{t,r,j} = s_{t,r,j}^{(4)} q_{r,j}^{(4)} + s_{t,r,j}^{(2)} q_{r,j}^{(2)}, \qquad q_{r,j}^{(2)} \in \{-2, -1, 0, 1\}^B, \qquad s_{t,r,j}^{(2)} \geq 0. \tag{14}$$

Here, $q_{r,j}^{(2)}$ is shared across iterations, and $s_{t,r,j}^{(2)}$ denotes its effective loop-conditioned scale.

For each output-tile/input-group pair (o, j), the residual codes and scales are fitted with the W4 parameters, calibration activations, and coupling matrices held fixed. The residual-fitting objective and its scale coefficients are given in Appendix B.6. The no-overlay solution is retained whenever a fitted candidate does not improve the surrogate.

We score each candidate by its achieved surrogate reduction:

$$\Delta_{o,j} = \mathcal{J}_{o,j}^{(4)} - \mathcal{J}_{o,j}^{(4+2)}, \tag{15}$$

where the two terms are the corresponding tile contributions to Eq. equation 11, evaluated before and after adding the fitted residual. These are reductions in the calibration surrogate, not estimates of signed changes in downstream accuracy or task loss.

Across all projection groups, candidate pairs are ranked by $\Delta_{o,j}$. We construct the binary tile mask $\mathcal{M}$ by selecting the top $\lfloor \alpha N_{\text{total}} \rfloor$ candidates, where N_{total} is the total number of candidate tiles and the projection-group identity is implicit in each candidate. Unselected tiles retain the W4-only representation. For fixed, equal-cost candidates, this ranking is optimal for the additive surrogate under the stated cardinality budget; it is not a claim of downstream-task optimality (Appendix B.7).

At recurrent iteration t, the effective weight is

$$\widehat{W}_t = W_t^{(4)} + \mathcal{M} \odot W_t^{(2)}, \qquad W_t^{(b)} = S_t^{(b)} \odot Q^{(b)}, \quad b \in \{4, 2\}, \tag{16}$$

with scales and the tile mask broadcast over their associated weight coordinates. The integer planes and selection mask are shared across iterations; only the scale tensors depend on t.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

1) Models and Benchmarks We evaluate TEMPO on the recursive language models Ouro-1.4B and Ouro-2.6B Zhu et al. (2025a). Evaluation spans five zero-shot reasoning tasks: HellaSwag (HS) Zellers et al. (2019), WinoGrande (WG) Sakaguchi et al. (2021), LAMBADA (LB) Paperno et al. (2016); Radford et al. (2019), ARC-Challenge (AC) Bhakthavatsalam et al. (2021), and MMLU (MM) Hendrycks et al. (2020). For continuous language modeling fidelity, we evaluate perplexity on WikiText-2 (WT2) Merity et al. (2016) and the LAMBADA OpenAI split (OP) Paperno et al. (2016); Radford et al. (2019).

2) Baselines. We compare TEMPO against established post-training quantization methods: vanilla Round-To-Nearest (RTN), SmoothQuant Xiao et al. (2023a), QuaRot Ashkboos et al. (2024b), SpinQuant Liu et al. (2025b), FlatQuant Sun et al. (2024), and the recurrent quantization baseline LoopQ Fang et al. (2026b), with full-precision BF16 models serving as uncompressed references. All methods are evaluated under their supported W4A4 and W4A8 configurations, reporting top-1 accuracy and perplexity across all evaluation sets.

4.2 MAIN RESULTS

Table 1: Main results under W4A4 and W4A8. BF16 denotes the full-precision reference model. HS, WG, LB, AC, and MM denote HellaSwag, WinoGrande, OpenAI LAMBADA, ARC-Challenge, and MMLU, respectively. WT2 and OP denote WikiText-2 word perplexity and LAMBADA perplexity, respectively. Bold numbers indicate the best quantized result within each model. Because no official LoopQ implementation was publicly available, the reported results for all other baseline methods, are taken directly from the LoopQ paper.

Model	Method	W4A4							W4A8						
		Tasks ↑					Perplexity ↓		Tasks ↑					Perplexity ↓	
		HS	WG	LB	AC	MM	WT2	OP	HS	WG	LB	AC	MM	WT2	OP
Ouro-1.4B	*BF16*	0.7159	0.6717	0.6505	0.5256	0.6823	12.03	5.30	0.7159	0.6717	0.6505	0.5256	0.6823	12.03	5.30
	Symmetric	0.2777	0.4941	0.0033	0.2389	0.2282	1.17e3	8.20e4	0.5022	0.5866	0.3220	0.3745	0.3569	43.89	46.72
	SmoothQuant	0.2575	0.5170	0.0000	0.2457	0.2499	7.71e4	2.91e7	0.6010	0.5722	0.2851	0.4104	0.2910	33.27	60.21
	QuaRot	0.2908	0.4972	0.0293	0.2270	0.2366	451.44	1.58e4	0.6817	0.6195	0.5523	0.4983	0.5577	15.27	8.95
	SpinQuant	0.2914	0.4925	0.0078	0.2210	0.2352	653.78	4.37e4	0.4838	0.5630	0.3647	0.3745	0.3240	27.88	27.61
	FlatQuant	0.2698	0.4483	0.0000	0.2278	0.2358	1.05e4	4.40e5	0.6948	0.6511	0.5962	0.4983	0.6336	13.53	6.92
	LoopQ	0.6582	0.5951	0.5659	0.4881	0.5465	15.81	8.18	0.6954	0.6456	0.6255	0.5145	0.6206	13.29	6.09
	TEMPO	**0.6811**	**0.6140**	**0.6022**	**0.5145**	**0.6026**	**13.39**	**6.39**	**0.7039**	**0.6606**	**0.6363**	**0.5375**	**0.6587**	**12.17**	**5.54**
Ouro-2.6B	*BF16*	0.7636	0.6867	0.7013	0.5802	0.7380	10.51	4.22	0.7636	0.6867	0.7013	0.5802	0.7380	10.51	4.22
	Symmetric	0.2811	0.5043	0.0361	0.2244	0.2311	962.74	7.81e4	0.5517	0.5777	0.4093	0.3729	0.3386	17.21	22.07
	SmoothQuant	0.2589	0.5012	0.0000	0.2662	0.2455	9.09e4	2.09e7	0.3219	0.5099	0.0693	0.2363	0.2297	198.38	1.81e3
	QuaRot	0.2967	0.4964	0.0782	0.2329	0.2307	301.26	3.65e3	0.5389	0.5659	0.4157	0.3874	0.4419	15.78	21.53
	SpinQuant	0.2805	0.4862	0.0200	0.2193	0.2316	621.14	5.17e4	0.5651	0.5754	0.4611	0.4044	0.4432	15.81	17.26
	FlatQuant	0.2560	0.4917	0.0000	0.2560	0.2455	3.07e4	1.62e7	0.6393	0.6085	0.3895	0.4309	0.4519	25.51	26.07
	LoopQ	0.6860	0.6227	0.5680	0.5034	0.5265	15.89	8.17	0.7555	0.6780	0.6837	**0.5819**	0.7007	**11.38**	4.47
	TEMPO	**0.7249**	**0.6890**	**0.6755**	**0.5495**	**0.6769**	**12.95**	**4.55**	**0.7561**	**0.6961**	**0.7064**	0.5700	**0.7210**	11.66	**4.09**

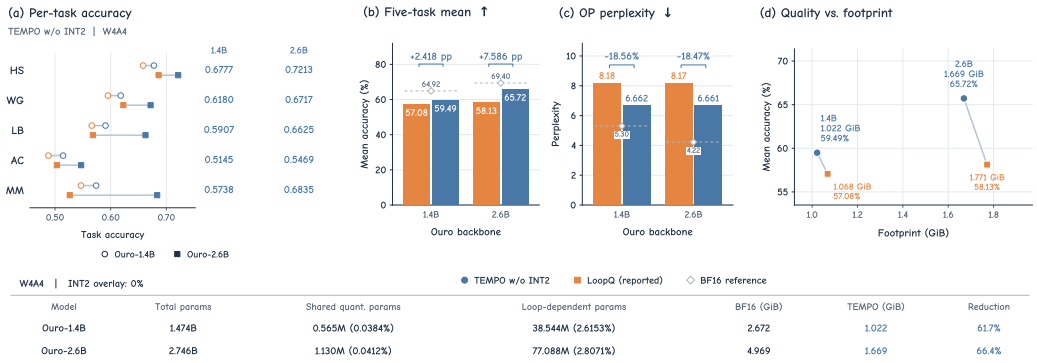
Table 1 compares TEMPO with existing post-training quantization methods under W4A4 and W4A8 regimes. The performance advantage is most pronounced under W4A4. On Ouro-1.4B, TEMPO outperforms LoopQ by 3.21% in average downstream accuracy, lowering WikiText-2 and LAMBADA perplexities to 13.39 and 6.39, respectively. On Ouro-2.6B, this average accuracy margin widens to 8.18%, with MMLU reaching 0.6769 (compared to 0.5265 for LoopQ) and WikiText-2 perplexity dropping to 12.95 (from 15.89).

Under W4A8, TEMPO closely tracks the BF16 reference across both model scales. On Ouro-1.4B, it retains 98.5% of full-precision average task accuracy, improving upon LoopQ by 1.91%. On Ouro-2.6B, TEMPO achieves an average accuracy of 0.6900, remaining within 0.4% of BF16 while yielding the best quantized results across most metrics. These evaluations demonstrate that modeling trajectory-level coupling prevents the severe degradation that affects standard baselines when activations are pushed to 4 bits.

Nominal precision alone does not establish equal storage budgets. We therefore evaluate **TEMPO without INT2** under W4A4 and report model-state footprints separately from code precision.

As shown in Figure 4, residual-free TEMPO improves the unweighted five-task mean accuracy over the model-matched LoopQ reports Fang et al. (2026a) from 57.076% to 59.494% on Ouro-1.4B and from 58.132% to 65.718% on Ouro-2.6B, corresponding to gains of **2.418** and **7.586** percentage points, respectively. OP perplexity decreases from 8.18 to **6.662** and from 8.17 to **6.661**. These results show that the performance advantage persists without additional INT2 codes.

The reported TEMPO model-state footprints are **1.022** and **1.669 GiB**, respectively. Interpreting LoopQ's published footprints of 1.147 and 1.902 GB as decimal units gives 1.068 and 1.771 GiB. Under this unit interpretation, TEMPO has higher reported accuracy and smaller reported footprints; establishing a matched-budget advantage additionally requires consistent model-state accounting and evaluation protocols.



Model	Total params	Shared quant. params	Loop-dependent params	BF16 (GiB)	TEMPO (GiB)	Reduction
Ouro-1.4B	1.474B	0.565M (0.0384%)	38.544M (2.6153%)	2.672	1.022	61.7%
Ouro-2.6B	2.746B	1.130M (0.0412%)	77.088M (2.8071%)	4.969	1.669	66.4%

Figure 4: **Residual-free TEMPO under W4A4.** (a) Per-task accuracy gains over model-matched LoopQ reports. (b) Mean accuracy over HS, WG, LB, AC and MM. (c) OP perplexity. (d) Mean accuracy versus reported model-state footprint. All TEMPO results disable INT2 overlays.

4.3 Sensitivity Analysis

Table 2: Sensitivity analysis of calibration sequences, INT2 residual ratio, and calibration seed for Ouro-1.4B under W4A4 quantization.

Configuration	HS↑	WG↑	LB↑	AC↑	MM↑	Avg. Acc.↑	WT2↓	OP↓	Avg. PPL↓
Number of calibration sequences									
$N_{\text{cal}} = 32$	0.6768	0.6361	0.5981	0.5162	0.6020	0.6058	13.4660	6.6640	10.0650
$N_{\text{cal}} = 64$	0.6761	0.6330	0.6175	0.5111	0.6029	0.6081	13.4390	6.1780	9.8085
$N_{\text{cal}} = 112^{\dagger}$	0.6811	0.6140	0.6022	0.5145	0.6026	0.6029	13.3930	6.3890	9.8910
INT2 residual ratio									
$\rho = 0.125$	0.6807	0.6433	0.5837	0.5128	0.5854	0.6012	13.5596	6.8787	10.2192
$\rho = 0.250^{\dagger}$	0.6811	0.6140	0.6022	0.5145	0.6026	0.6029	13.3930	6.3890	9.8910
$\rho = 0.375$	0.6835	0.6219	0.5989	0.5051	0.6043	0.6027	13.2077	6.5475	9.8776
$\rho = 0.500$	0.6849	0.6361	0.6033	0.5068	0.6113	0.6085	13.0595	6.4783	9.7689
Calibration seed									
$s = 0$	0.6820	0.6519	0.6189	0.5077	0.5968	0.6115	14.9787	6.2735	10.6261
$s = 42$	0.6800	0.6401	0.6253	0.4932	0.6050	0.6087	15.0102	6.1076	10.5589
$s = 2026^{\dagger}$	0.6811	0.6140	0.6022	0.5145	0.6026	0.6029	13.3930	6.3890	9.8910
$s = 3407$	0.6793	0.6346	0.6247	0.4863	0.6048	0.6059	14.9972	6.1215	10.5594

† Default configuration: $(N_{\text{cal}}, \rho, s) = (112, 0.25, 2026)$.

We analyze the stability of TEMPO under W4A4 quantization on Ouro-1.4B across three axes: calibration sample volume, INT2 residual budget ratio, and random calibration seeds. Unless varied explicitly, parameters are fixed to $(N_{\text{cal}}, \alpha, s) = (112, 0.25, 2026)$. Table 2 summarizes the results.

1) Calibration Sample Efficiency. As reported in Table 2, TEMPO exhibits rapid convergence with respect to calibration sequence count. This confirms that our cross-loop covariance tensor

H^{xx} and terminal coupling C^o saturate quickly with compact data samples, avoiding the need for computationally heavy calibration rollouts.

2) INT2 Residual Budget Scaling. Increasing the INT2 budget fraction α from 0.125 to 0.5 delivers continuous representation gains, lifting HellaSwag from 0.6807 to 0.6849, MMLU from 0.5854 to 0.6113, and reducing WikiText-2 perplexity from 13.56 to 13.06. However, higher overlay density scales the compressed checkpoint footprint from 709 MB to 847 MB.

3) Robustness Across Calibration Seeds. Across distinct calibration seeds, core reasoning benchmarks remain stable, with HellaSwag and MMLU exhibiting maximum variations of only 0.0027 and 0.0082, respectively. While smaller zero-shot splits like WinoGrande and ARC-Challenge display sample-level variance across calibration subsets, the absolute degradation remains tightly bounded, with all seeded configurations outperforming PTQ baselines by substantial margins.

4.4 Recurrent Error Propagation Dynamics

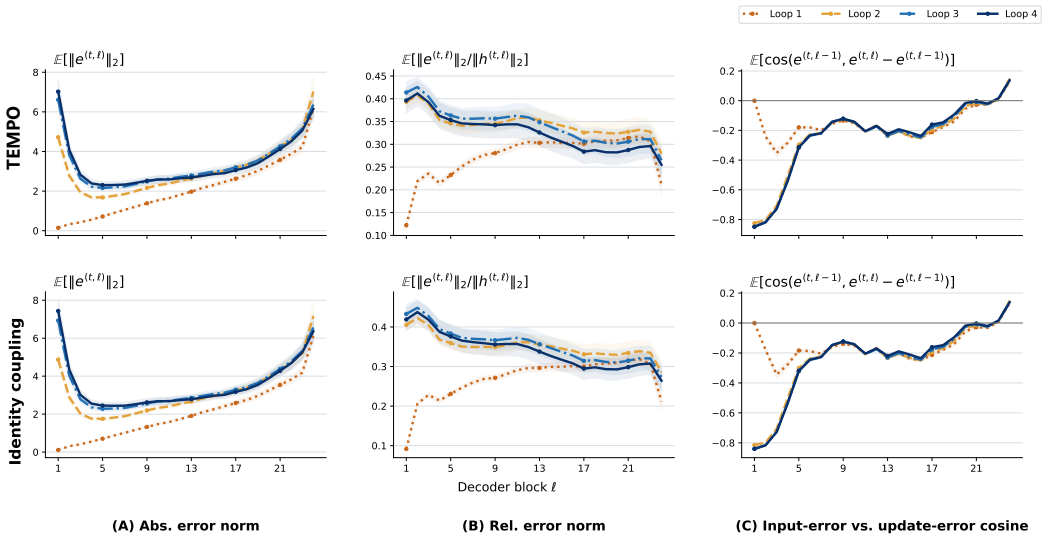


Figure 5: **Quantization error propagation across decoder blocks and recursive loops on Ouro-1.4B under W4A4.** Comparison between Full TEMPO and the Identity-coupling baseline evaluated on (a) absolute error norm, (b) relative state distortion norm, and (c) cosine similarity between input perturbation and layer update error.

Figure 5 evaluates state perturbations across recurrent iterations for Full TEMPO and the identity baseline ($C^o = I$). In Loops 2 to 4, the identity baseline ($C^o = I$) exhibits compounding state divergence, whereas Full TEMPO consistently suppresses both absolute and relative error growth (Figures 5(a) and 5(b)). This confirms that terminal coupling C^o successfully prevents long-horizon drift that isolated layer-wise proxies fail to constrain. Furthermore, while early decoder blocks exhibit intrinsic error cancellation via negative cosine alignment (Figure 5(c)), only Full TEMPO sustains this damping behavior into deeper recurrent iterations.

5 Conclusion

This paper presented TEMPO, a post-training quantization framework designed to prevent recursive error accumulation in unrolled Transformers. By evaluating cross-loop sensitivity via an analytical terminal coupling metric, TEMPO isolates trajectory-critical parameters and pairs a shared 4-bit integer backbone with sparse INT2 residual overlays. Evaluations on Ouro models confirm that accounting for multi-step error propagation maintains downstream reasoning accuracy under aggressive 4-bit weights and activations, closing the gap to full-precision baselines without multiplying parameter footprints.

AI USE STATEMENT

In this work, we used generative AI tools, including large-language-model-based assistants, to support several stages of the research process. In accordance with the ICLR 2027 AI Policy for Authors, we disclose the following uses.

Generative AI was used to provide feedback on research methodology and experimental design, to assist in analyzing and refining the presentation of the TEMPO method and its terminal-risk surrogate, and to support the development and checking of mathematical derivations. It was also used to assist in designing ablation studies, storage-accounting analyses, and experiments for validating the relationship between the proposed terminal-risk surrogate and measured terminal degradation. In addition, generative AI was used to assist with portions of software implementation, code inspection and debugging, and with the analysis and interpretation of experimental results and potential reviewer concerns.

We additionally used generative AI for drafting and editing portions of the manuscript, improving the clarity and organization of technical explanations, Chinese–English translation, preparing or refining tables and figure descriptions, and searching, summarizing, and comparing relevant literature. Generative AI was not used as a substitute for running the model experiments reported in this paper or for generating the reported empirical measurements. The reported model performance, ablation results, and resource measurements are obtained from executed experiments or are explicitly attributed to prior work. We did not use generative AI to generate synthetic datasets for the main experiments.

All AI-assisted research suggestions, mathematical content, code, literature summaries, and manuscript text were reviewed by the authors. The authors checked the final methodological and mathematical statements for correctness, verified literature-dependent claims against their original sources, and validated implementations and reported numerical results through actual code execution and experiments. AI-generated or AI-suggested material was included only after author review. All research decisions and conclusions were made by the authors, who take full responsibility for the final content of this work, including text, code, analyses, and other artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies post-training quantization for recursive Transformers and does not involve human-subject experiments or the collection of new personal, sensitive, or user-interaction data. Our experiments use existing publicly available models and standard public benchmark datasets, subject to their respective licenses and terms of use. We do not introduce a new dataset, and no human annotation or institutional-review-board-regulated study was conducted as part of this work.

The goal of TEMPO is to reduce the storage requirements and quantization degradation associated with low-bit deployment of recursive language models. As with other model-compression techniques, improved deployment efficiency can reduce the computational and storage resources required to run language models and may broaden their accessibility. At the same time, reducing deployment cost can also lower the barrier to deploying an existing model in harmful or otherwise inappropriate applications. TEMPO does not introduce new training data, knowledge, or intended capabilities beyond those of the underlying model; its downstream risks therefore largely inherit those of the base model and its deployment setting. Users should follow the licenses, safety guidance, and applicable legal and organizational requirements associated with the underlying models and datasets.

We aim to report experimental settings, resource overheads, methodological limitations, and comparisons with prior work accurately and transparently. System-level efficiency or hardware benefits that have not been directly measured are not presented as empirical findings.

REPRODUCIBILITY STATEMENT

We take several steps to support reproducibility. The main Method section specifies the complete TEMPO pipeline, including terminal-sensitivity calibration, equivalent-representation and clipping search, shared W4 code optimization with loop-conditioned scales, and budget-constrained INT2

residual allocation. The experimental section reports the evaluated models, quantization settings, calibration data, group size, output-tile size, default residual budget, hardware configuration, and evaluation protocol.

Additional implementation details are provided in Appendix D, including calibration-data construction, random seeds, probe construction, terminal-coupling estimation and normalization, gauge and clipping candidates, W4 and INT2 code/scale optimization, solver and stopping conditions, model-state and storage accounting, and evaluation procedures. For the principal ablations, we specify which component is changed relative to the full method while holding the remaining experimental conditions fixed whenever applicable.

We additionally provide anonymized source code, experiment configurations, and execution scripts in the supplementary material. These materials contain the entry points required to construct quantized checkpoints, run the principal benchmarks and ablations, measure model-state footprints, and reproduce the reported tables. Results imported from prior work are explicitly identified as reported baseline results and are distinguished from results that we obtain through our own execution.

REFERENCES

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 37: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024a.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. *Advances in Neural Information Processing Systems*, 37:100213–100240, 2024b.

Sumithra Bhakthavatsalam, Daniel Khashabi, Tushar Khot, Bhavana Dalvi Mishra, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, and Peter Clark. Think you have solved direct-answer question answering? try arc-da, the direct-answer AI2 reasoning challenge. *CoRR*, abs/2102.03315, 2021. URL https://arxiv.org/abs/2102.03315.

Rui Fang, Hsi-Wen Chen, and Ming-Syan Chen. Loopq: Quantization for recursive transformers. *CoRR*, abs/2605.16343, 2026a. doi: 10.48550/ARXIV.2605.16343. URL https://doi.org/10.48550/arXiv.2605.16343.

Rui Fang, Hsi-Wen Chen, and Ming-Syan Chen. Loopq: Quantization for recursive transformers. *arXiv preprint arXiv:2605.16343*, 2026b.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: accurate post-training quantization for generative pre-trained transformers. *CoRR*, abs/2210.17323, 2022. doi: 10.48550/ARXIV.2210.17323. URL https://doi.org/10.48550/arXiv.2210.17323.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: activation-aware weight quantization for on-device LLM compression and acceleration. In Phillip B. Gibbons, Gennady Pekhimenko, and Christopher De Sa (eds.), *Proceedings of the Seventh Annual Conference on Machine Learning and Systems, MLSys 2024, Santa Clara, CA, USA, May 13-16, 2024*. mlsys.org, 2024.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. Spinquant: LLM quantization with learned rotations. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025a.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. Spinquant: Llm quantization with learned rotations. In *International Conference on Learning Representations*, volume 2025, pp. 92009–92032, 2025b.

Yuexiao Ma, Huixia Li, Xiawu Zheng, Feng Ling, Xuefeng Xiao, Rui Wang, Shilei Wen, Fei Chao, and Rongrong Ji. Affinequant: Affine transformation quantization for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

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

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016. URL https://doi.org/10.18653/v1/p16-1144.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106, 2021. doi: 10.1145/3474381. URL https://doi.org/10.1145/3474381.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Jiaxin Hu, Xianzhi Yu, Lu Hou, Chun Yuan, et al. Flatquant: Flatness matters for llm quantization. *arXiv preprint arXiv:2410.09426*, 2024.

Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Jiaxin Hu, Xianzhi Yu, Lu Hou, Chun Yuan, Xin Jiang, Wulong Liu, and Jun Yao. Flatquant: Flatness matters for LLM quantization. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pp. 38087–38099, 2023a.

Guangxuan Xiao, Ji Lin, Mickaël Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099. PMLR, 2023b.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In Anna Korhonen, David R. Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/V1/P19-1472. URL https://doi.org/10.18653/v1/p19-1472.

Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, Qiyang Min, Hongzhi Huang, Xun Zhou, Wei Ye, Jiaheng Liu, Jian Yang, Yunfeng Shi, Chenghua Lin, Enduo Zhao, Tianle Cai, Ge Zhang, Wenhao Huang, Yoshua Bengio, and Jason Eshraghian. Scaling latent reasoning via looped language models. *CoRR*, abs/2510.25741, 2025a. doi: 10.48550/ARXIV.2510.25741. URL https://doi.org/10.48550/arXiv.2510.25741.

Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, Qiyang Min, Hongzhi Huang, Xun Zhou, Wei Ye, Jiaheng Liu, Jian Yang, Yunfeng Shi, Chenghua Lin, Enduo Zhao, Tianle Cai, Ge Zhang, Wenhao Huang, Yoshua Bengio, and Jason Eshraghian. Scaling latent reasoning via looped language models. *CoRR*, abs/2510.25741, 2025b. doi: 10.48550/ARXIV.2510.25741. URL https://doi.org/10.48550/arXiv.2510.25741.

A Terminal-Sensitivity Surrogate

This appendix derives the structured surrogate used in Section 3.2 and distinguishes its algebraic identities from its approximation assumptions.

A.1 Linearized probe responses

Fix a projection group p and output tile o. For readability, write $G_{m,t} = G^p_{m,t,o}$, $E_t = E^p_{t,o}$, $N = N_p$, $h = |o|$, and $d = Nh$. Each $E_t \in \mathbb{R}^{N\times h}$ is a local additive projection error evaluated on the full-precision reference trajectory, not an accumulated hidden-state difference.

The probe family contains task probes and categorical predictive-distribution probes. All identities below hold for the finite collection of scalar objectives actually used, with their scaling included in Φ_m. They do not require identifying the combined probe family with the Hessian or Fisher matrix of a single loss.

For perturbations restricted to the chosen projection group and output tile, the linearized response of probe m is

$$\delta\Phi_{m,\mathrm{lin}}(E) = \sum_{t=1}^{T} \langle G_{m,t}, E_t\rangle_F. \tag{17}$$

Let $g_{m,t} = \mathrm{vec}(G_{m,t})$ and $e_t = \mathrm{vec}(E_t)$. The mean squared linearized response is

$$\begin{aligned} \mathcal{S}_{\mathrm{probe}}(E) &= \frac{1}{M_p}\sum_{m=1}^{M_p}\left(\sum_{t=1}^{T} g_{m,t}^{\mathsf{T}} e_t\right)^2 \\ &= \sum_{t,u=1}^{T} e_t^{\mathsf{T}} K_{tu} e_u, \end{aligned} \tag{18}$$

where

$$K_{tu} = \frac{1}{M_p}\sum_{m=1}^{M_p} g_{m,t} g_{m,u}^{\mathsf{T}}. \tag{19}$$

Equation equation 18 is an exact algebraic identity for the linearized probe responses. It is not a second-order expansion of the task loss.

In particular, squaring the responses measures their magnitude rather than the sign of a task-loss change. The resulting objective is not identified with changes in accuracy or perplexity.

A.2 Isotropic compression of directional sensitivity

The coupling coefficient in Eq. equation 3 satisfies

$$\widetilde{C}_{tu} = \frac{1}{M_p} \sum_{m=1}^{M_p} g_{m,t}^{\mathsf{T}} g_{m,u} = \mathrm{tr}(K_{tu}). \tag{20}$$

Thus, it retains one scalar summary of each directional coupling block.

Projection identity. For any $K_{tu} \in \mathbb{R}^{d\times d}$,

$$\arg\min_{c\in\mathbb{R}} \|K_{tu} - cI_d\|_F^2 = \frac{\mathrm{tr}(K_{tu})}{d}. \tag{21}$$

Proof. Expanding the squared Frobenius norm gives

$$\|K_{tu} - cI_d\|_F^2 = \|K_{tu}\|_F^2 - 2c\ \mathrm{tr}(K_{tu}) + dc^2. \tag{22}$$

The right-hand side is strictly convex in c. Setting its derivative to zero yields Eq. equation 21. □

Substituting this projection into Eq. equation 18 gives

$$\mathcal{S}_{\mathrm{iso}}(E) = \frac{1}{Nh} \sum_{t,u=1}^{T} \widetilde{C}_{tu} \langle E_t, E_u \rangle_F. \tag{23}$$

This approximation preserves temporal coupling but discards directional information within the tile.

The raw matrix $\widetilde{C}$ is positive semidefinite. Indeed, for any $v \in \mathbb{R}^T$,

$$v^{\mathsf{T}} \widetilde{C} v = \frac{1}{M_p} \sum_{m=1}^{M_p} \left\| \sum_{t=1}^{T} v_t G_{m,t} \right\|_F^2 \geq 0. \tag{24}$$

Consequently, the block approximation $\mathbb{K}_{\mathrm{iso}} = (\widetilde{C}/d) \otimes I_d$ is also positive semidefinite.

The projection identity specifies the best approximation within a restricted family; it does not establish that the approximation error is small. Writing $\mathbb{K} = [K_{tu}]_{t,u}$ and $e = [e_1^{\mathsf{T}}, \ldots, e_T^{\mathsf{T}}]^{\mathsf{T}}$,

$$|\mathcal{S}_{\mathrm{probe}}(E) - \mathcal{S}_{\mathrm{iso}}(E)| \leq \|\mathbb{K} - \mathbb{K}_{\mathrm{iso}}\|_2 \|e\|_2^2. \tag{25}$$

This bounds compression error relative to the probe-response objective, not downstream task error.

A.3 Regularization and shared normalization

Let $D_{\mathrm{damp}}^{p,o}$ be the non-negative diagonal damping matrix used for projection group p and output tile o. The numerical regularization can be written as

$$\overline{C}^{p,o} = \Pi_{\succeq 0} \left(\frac{\widetilde{C}^{p,o} + (\widetilde{C}^{p,o})^{\mathsf{T}}}{2} + D_{\mathrm{damp}}^{p,o} \right), \tag{26}$$

where $\Pi_{\succeq 0}$ clips negative eigenvalues of a symmetric matrix to zero. The raw coupling is already positive semidefinite in exact arithmetic; the projection is a numerical safeguard.

A shared normalization gives

$$C^{p,o} = \frac{\overline{C}^{p,o}}{\gamma}, \qquad \gamma = \max\left\{ \varepsilon,\ \max_{p,o,t,u} |\overline{C}_{tu}^{p,o}| \right\}, \tag{27}$$

where $\varepsilon > 0$ prevents division by zero. This common positive rescaling preserves relative coefficients across the regularized matrices.

The projection identity in Eq. equation 21 applies to the raw coupling $\widetilde{C}$. After damping or other numerical modification, C defines a regularized surrogate rather than the unmodified projection.

A.4 Block separability and the optimization objectives

Let $E^p_{t,o,j}$ collect the error contributions of input group j across the rows of output tile o:

$$E^p_{t,o,j} = \left[e^p_{t,r,j}\right]_{r\in o}, \qquad E^p_{t,o} = \sum_j E^p_{t,o,j}. \tag{28}$$

The full within-tile inner product contains cross-input-group terms:

$$\langle E^p_{t,o}, E^p_{u,o}\rangle_F = \sum_{j,k} \langle E^p_{t,o,j}, E^p_{u,o,k}\rangle_F. \tag{29}$$

TEMPO keeps only terms with $j = k$ and sums projection groups and output tiles independently. This also omits the cross-projection and cross-tile products that would appear in a joint squared probe response. These omissions are additional structural approximations, not consequences of the isotropic projection.

Using the regularized coupling, the resulting objective is

$$\mathcal{J}_{\text{cal}} = \sum_{p,o,j} \frac{1}{N_p} \sum_{t,u=1}^{T} C^{p,o}_{tu} \left\langle E^p_{t,o,j}, E^p_{u,o,j}\right\rangle_F. \tag{30}$$

Since

$$\left\langle E^p_{t,o,j}, E^p_{u,o,j}\right\rangle_F = \sum_{r\in o} \left\langle e^p_{t,r,j}, e^p_{u,r,j}\right\rangle, \tag{31}$$

substituting the W4 errors from Eq. equation 10 recovers Eq. equation 11, summed over p. Substituting activation-only errors $(\widetilde{X}_{t,j} - X'_{t,j})w'_{r,j}$ gives the corresponding representation-screening objective.

To relate the normalization to Eq. equation 23, define the unregularized, block-separable isotropic score

$$\mathcal{S}_{\text{iso,sep}} = \sum_{p,o,j} \frac{1}{N_p|o|} \sum_{t,u=1}^{T} \widetilde{C}^{p,o}_{tu} \left\langle E^p_{t,o,j}, E^p_{u,o,j}\right\rangle_F. \tag{32}$$

When all output tiles have the same height h and numerical regularization leaves the raw matrices unchanged,

$$\mathcal{J}_{\text{cal}} = \frac{h}{\gamma}\mathcal{S}_{\text{iso,sep}}. \tag{33}$$

The common factor does not affect candidate rankings or minimizers. For unequal tile heights, $1/N_p$ introduces an additional tile-height weighting relative to $1/(N_p|o|)$; the common-factor equivalence must not be assumed for such tiles.

B Shared-Code Optimization and Residual Allocation

Throughout this appendix, calibration activations and coupling matrices are fixed within each shortlisted representation. Optimization of the surrogate does not differentiate through changes to the recurrent calibration trajectory.

B.1 Activation statistics and the initialization metric

For a fixed projection group p and input group j, define

$$\begin{aligned} H^{qq}_{p,j,tu} &= \frac{1}{N_p}\widetilde{X}^{\mathsf{T}}_{t,j}\widetilde{X}_{u,j}, \\ H^{qx}_{p,j,tu} &= \frac{1}{N_p}\widetilde{X}^{\mathsf{T}}_{t,j}X'_{u,j}, \\ H^{x'x'}_{p,j,tu} &= \frac{1}{N_p}(X'_{t,j})^{\mathsf{T}}X'_{u,j}. \end{aligned} \tag{34}$$

The superscripts q and x distinguish quantized-and-dequantized activations from transformed full-precision activations.

For a group-aligned gauge $X'_{t,j} = X_{t,j}\Gamma_j$,

$$H^{x'x'}_{p,j,tu} = \Gamma_j^{\mathsf{T}} H^{xx}_{p,j,tu}\Gamma_j. \tag{35}$$

Thus, the original moments in Eq. equation 2 can be expressed in the coordinates of each shortlisted representation.

A coupling-weighted rounding metric for initialization is

$$R^{\text{init}}_{p,o,j} = \sum_{t,u=1}^{T} C^{p,o}_{tu} H^{x'x'}_{p,j,tu}. \tag{36}$$

For a common weight perturbation δ,

$$\delta^{\mathsf{T}} R^{\text{init}}_{p,o,j}\delta = \frac{1}{N_p}\sum_{t,u=1}^{T} C^{p,o}_{tu}\langle X'_{t,j}\delta, X'_{u,j}\delta\rangle \geq 0. \tag{37}$$

This is an activation-reconstruction metric for GPTQ-style initialization, not a Hessian of downstream task loss. It does not include all terms induced by quantized activations and arbitrary loop-conditioned scales.

In particular, H^{xx} alone does not determine H^{qq} or H^{qx} after activation quantization. The latter statistics, or their equivalent direct inner products, are required for the exact subproblems below.

B.2 A COMMON ROW-GROUP FITTING PROBLEM

Fix a projection group, output row $r \in o$, and input group j. Suppress these indices and write $N = N_p$, $C = C^o$, and $\widetilde{X}_t = \widetilde{X}_{t,j}$. For prescribed target vectors $y_t \in \mathbb{R}^N$, consider

$$\begin{aligned} f(q, \mathbf{s}; y) &= \frac{1}{N}\sum_{t,u=1}^{T} C_{tu}\left\langle s_t\widetilde{X}_t q - y_t,\, s_u\widetilde{X}_u q - y_u\right\rangle, \\ q \in \mathcal{Q}_b^B, &\qquad \mathbf{s} \in \mathbb{R}^T_{\geq 0}, \end{aligned} \tag{38}$$

where $\mathcal{Q}_4 = \{-8, \ldots, 7\}$ and $\mathcal{Q}_2 = \{-2, -1, 0, 1\}$.

For W4 fitting, $b = 4$ and $y_t = X'_{t,j}w'_{r,j}$. For INT2 fitting with the W4 backbone fixed, $b = 2$ and $y_t = -e^{(4)}_{t,r,j}$. Both stages therefore share the same quadratic structure.

B.3 EXACT FIXED-CODE SCALE COEFFICIENTS

Fix q and let $a_t = \widetilde{X}_t q$. Expanding Eq. equation 38 yields

$$f(q, \mathbf{s}; y) = \mathbf{s}^{\mathsf{T}} A\mathbf{s} - 2\mathbf{b}^{\mathsf{T}}\mathbf{s} + c_y, \tag{39}$$

where

$$\begin{aligned} A_{tu} &= \frac{C_{tu}}{N} a_t^{\mathsf{T}} a_u, \\ b_t &= \frac{1}{N}\sum_{u=1}^{T} C_{tu} a_t^{\mathsf{T}} y_u, \\ c_y &= \frac{1}{N}\sum_{t,u=1}^{T} C_{tu} y_t^{\mathsf{T}} y_u. \end{aligned} \tag{40}$$

The constant c_y does not affect scale optimization.

For W4 fitting, these coefficients can equivalently be written as

$$\begin{aligned} A_{tu} &= C_{tu}\, q^{\mathsf{T}} H^{qq}_{p,j,tu} q, \\ b_t &= \sum_{u=1}^{T} C_{tu}\, q^{\mathsf{T}} H^{qx}_{p,j,tu} w'_{r,j}. \end{aligned} \tag{41}$$

The required quantities can be accumulated as direct inner products; materializing every moment matrix is not necessary.

To verify convexity, for $z \in \mathbb{R}^T$ define $v_n = [z_1 a_1[n], \ldots, z_T a_T[n]]^\mathsf{T}$. Then

$$z^\mathsf{T} A z = \frac{1}{N} \sum_{n=1}^{N} v_n^\mathsf{T} C v_n \geq 0. \tag{42}$$

Thus, A is positive semidefinite whenever C is positive semidefinite.

B.4 Active-set scale solution

For

$$\min_{\mathbf{s} \geq 0} \mathbf{s}^\mathsf{T} A \mathbf{s} - 2\mathbf{b}^\mathsf{T} \mathbf{s}, \tag{43}$$

the Karush–Kuhn–Tucker conditions are

$$\mathbf{s} \geq 0, \qquad A\mathbf{s} - \mathbf{b} \geq 0, \qquad \mathbf{s} \odot (A\mathbf{s} - \mathbf{b}) = 0. \tag{44}$$

Because the problem is convex, these conditions are sufficient for global optimality of the fixed-code subproblem.

For a candidate free set $\mathcal{F} \subseteq \{1, \ldots, T\}$, set $\mathbf{s}_{\mathcal{F}^c} = 0$ and solve

$$A_{\mathcal{F}\mathcal{F}} \mathbf{s}_{\mathcal{F}} = \mathbf{b}_{\mathcal{F}}. \tag{45}$$

A candidate is valid when

$$\mathbf{s}_{\mathcal{F}} \geq 0, \qquad A_{\mathcal{F}^c \mathcal{F}} \mathbf{s}_{\mathcal{F}} - \mathbf{b}_{\mathcal{F}^c} \geq 0. \tag{46}$$

Enumerating the 2^T free sets is feasible for a small recurrent horizon.

Singular active systems require a feasibility check rather than blindly accepting a pseudoinverse solution. When Eq. equation 45 is consistent, its solutions have the form

$$\mathbf{s}_{\mathcal{F}} = A^{\dagger}_{\mathcal{F}\mathcal{F}} \mathbf{b}_{\mathcal{F}} + Z_{\mathcal{F}} v, \tag{47}$$

where the columns of $Z_{\mathcal{F}}$ span the null space. A valid candidate exists only when some v satisfies Eq. equation 46 and non-negativity. The empty free set is also considered.

In finite precision, stationarity, feasibility, and complementarity are checked to the implemented numerical tolerance. Adding a ridge directly to A changes the subproblem and must be distinguished from solving the original surrogate.

B.5 Fixed-scale integer refinement

For fixed $\mathbf{s}$, Eq. equation 38 becomes

$$f(q, \mathbf{s}; y) = q^\mathsf{T} M_{\mathbf{s}} q - 2h_{\mathbf{s}}^\mathsf{T} q + c_y, \tag{48}$$

where

$$\begin{aligned} M_{\mathbf{s}} &= \frac{1}{N} \sum_{t,u=1}^{T} C_{tu} s_t s_u \widetilde{X}_t^\mathsf{T} \widetilde{X}_u, \\ h_{\mathbf{s}} &= \frac{1}{N} \sum_{t,u=1}^{T} C_{tu} s_t \widetilde{X}_t^\mathsf{T} y_u. \end{aligned} \tag{49}$$

For W4 fitting,

$$\begin{aligned} M_{\mathbf{s}} &= \sum_{t,u=1}^{T} C_{tu} s_t s_u H^{qq}_{p,j,tu}, \\ h_{\mathbf{s}} &= \sum_{t,u=1}^{T} C_{tu} s_t H^{qx}_{p,j,tu} w'_{r,j}. \end{aligned} \tag{50}$$

A compatible local refinement evaluates the finite alphabet for each integer coordinate. Replacing q_k by $v \in \mathcal{Q}_b$ changes the objective by

$$\Delta f_k(v) = 2(v - q_k)(M_{\mathbf{s}} q - h_{\mathbf{s}})_k + (v - q_k)^2 (M_{\mathbf{s}})_{kk}. \tag{51}$$

Thus, a coordinate update can choose

$$q_k^{\text{new}} \in \arg\min_{v \in \mathcal{Q}_b} \Delta f_k(v), \tag{52}$$

retaining the current value on ties. Finite enumeration avoids division by a potentially zero diagonal entry.

Given an initialized code, a compatible alternating solver first solves the scale subproblem, performs integer-coordinate updates using Eq. equation 52, and then re-solves the scales. The best feasible iterate is retained under the same objective. Iterations stop when no accepted improvement exceeds the chosen stopping tolerance or when the update budget is reached.

Accepted updates produce a non-increasing surrogate sequence. This property does not establish global optimality of the joint discrete–continuous problem.

B.6 INT2 RESIDUAL FITTING

For a fixed W4 row-group configuration, define

$$e_t^{(4)} = \widetilde{X}_t s_t^{(4)} q^{(4)} - X_t' w'. \tag{53}$$

Adding a shared INT2 residual gives

$$e_t^{(4+2)} = e_t^{(4)} + s_t^{(2)} \widetilde{X}_t q^{(2)}. \tag{54}$$

The residual-fitting problem is

$$\min_{\substack{q^{(2)} \in \mathcal{Q}_2^B \\ \mathbf{s}^{(2)} \geq 0}} \frac{1}{N} \sum_{t,u=1}^{T} C_{tu} \left\langle e_t^{(4)} + s_t^{(2)} \widetilde{X}_t q^{(2)}, e_u^{(4)} + s_u^{(2)} \widetilde{X}_u q^{(2)} \right\rangle. \tag{55}$$

The W4 codes and scales remain fixed throughout this problem.

For fixed $q^{(2)}$, let $a_t^{(2)} = \widetilde{X}_t q^{(2)}$. The scale coefficients are

$$\begin{aligned} A_{tu}^{(2)} &= \frac{C_{tu}}{N} (a_t^{(2)})^{\mathsf{T}} a_u^{(2)}, \\ b_t^{(2)} &= -\frac{1}{N} \sum_{u=1}^{T} C_{tu} (a_t^{(2)})^{\mathsf{T}} e_u^{(4)}. \end{aligned} \tag{56}$$

The negative sign follows from using $y_t = -e_t^{(4)}$ in Eq. equation 38. The constant term is the W4-only row-group surrogate.

For fixed residual scales, the integer coefficients are

$$\begin{aligned} M_{\mathbf{s}}^{(2)} &= \frac{1}{N} \sum_{t,u=1}^{T} C_{tu} s_t^{(2)} s_u^{(2)} \widetilde{X}_t^{\mathsf{T}} \widetilde{X}_u, \\ h_{\mathbf{s}}^{(2)} &= -\frac{1}{N} \sum_{t,u=1}^{T} C_{tu} s_t^{(2)} \widetilde{X}_t^{\mathsf{T}} e_u^{(4)}. \end{aligned} \tag{57}$$

Equation equation 52 then applies with the INT2 alphabet.

The zero-overlay configuration is feasible and reproduces the W4-only surrogate. After fitting a candidate, comparison with this fallback ensures that a retained overlay does not increase the surrogate. This does not require finding the global optimum of Eq. equation 55.

The zero overlay is a fallback, not an informative initialization for alternating code–scale optimization: initializing both the code and scales to zero can stall the updates. A residual initializer must therefore supply the candidate code or candidate set used before this comparison.

For output tile o and input group j, sum the fitted row-group objectives over $r \in o$ to obtain $\mathcal{J}_{o,j}^{(4+2)}$. The gain in Eq. equation 15 uses the achieved candidate value, not an assumed globally minimal residual objective.

B.7 BUDGETED OVERLAY SELECTION

Let $\mathcal{V}$ be the set of candidate tiles across all projection groups, with each $v \in \mathcal{V}$ identifying a tuple (p, o, j). Write

$$N_{\text{total}} = |\mathcal{V}|, \qquad K = \lfloor \alpha N_{\text{total}} \rfloor.$$

Hold the W4 backbone, calibration statistics, coupling matrices, and fitted residual candidates fixed.

For a binary selection variable m_v, block separability gives

$$\begin{aligned} \mathcal{J}(m) &= \sum_{v \in \mathcal{V}} \left[(1 - m_v) \mathcal{J}_v^{(4)} + m_v \mathcal{J}_v^{(4+2)} \right] \\ &= \mathcal{J}_{\text{W4}} - \sum_{v \in \mathcal{V}} m_v \Delta_v, \end{aligned} \tag{58}$$

where $\mathcal{J}_{\text{W4}} = \sum_v \mathcal{J}_v^{(4)}$.

For equal-cost candidates and an exact cardinality budget, mask selection is therefore

$$\max_{\substack{m_v \in \{0,1\} \\ \sum_v m_v = K}} \sum_{v \in \mathcal{V}} m_v \Delta_v. \tag{59}$$

Its solution selects the K largest gains. To see this, replacing a selected candidate with an unselected candidate of larger gain strictly improves the objective. Repeated exchanges produce a top-K selection. Ties can be resolved by a fixed candidate order.

When the budget is an upper bound, $\sum_v m_v \leq K$, zero-gain candidates may be left unselected. For unequal byte costs, ranking by raw gain is not generally optimal for a byte-constrained allocation problem; Eq. equation 59 is specifically a cardinality-budget result.

This optimality statement concerns only the binary mask, conditional on fixed candidates and the additive calibration surrogate. It does not imply globally optimal residual fitting, optimal joint W4–INT2 optimization, or optimal downstream task performance. The omitted cross-block interactions and changes to the recurrent trajectory are not included in Eq. equation 58.

C TARGETED COMPONENT ANALYSIS OF TEMPO

We examine terminal coupling, loop-conditioned scales, and INT2 residuals on Ouro-1.4B under W4A4 quantization. The three ablations respectively replace terminal coupling with the identity matrix, constrain scales to be shared across recurrent iterations, and remove INT2 residuals. The remaining settings follow the same experimental protocol.

This section selectively discusses metrics on which the components provide improvements. These benefits are metric-dependent: the full configuration does not achieve the highest five-task mean accuracy against every ablation.

C.1 TERMINAL COUPLING: GUIDING QUANTIZATION CALIBRATION WITH TERMINAL SENSITIVITY

Terminal coupling allows the quantization objective to account for the sensitivities of different recurrent positions and their interactions, rather than evaluating local error magnitudes alone. The calibration objective is

$$J_C(E) = \sum_{p,o,j} \frac{1}{N_p} \sum_{t,u=1}^{T} C_{tu}^{p,o} \left\langle E_{t,o,j}^p, E_{u,o,j}^p \right\rangle_F. \tag{60}$$

Setting $C^{p,o} = I_T$ reduces this objective to a sum of squared local projection errors. In contrast, constructing $C^{p,o}$ from terminal probes allows representation search, code optimization, and residual allocation to evaluate these errors according to terminal sensitivity.

On WT2, full TEMPO reduces perplexity from 13.560 with identity coupling to 13.393, a relative reduction of 1.23%. On ARC-Challenge, accuracy increases from 50.34% to 51.45%, a gain of 1.11

percentage points. These results connect terminal-sensitivity calibration to downstream benefits: terminal information not only changes the construction of the calibration objective but also improves the quantized model's predictive performance on these metrics.

This comparison evaluates the terminal-sensitivity metric as a whole, including diagonal sensitivity weights and off-diagonal interactions, rather than isolating the off-diagonal terms.

C.2 Loop-Conditioned Scales: Temporal Adaptation with Shared Integer Codes

TEMPO reuses the same integer codes across recurrent iterations while adjusting effective weight magnitudes through loop-conditioned scales. This allows the quantized representation to accommodate loop-dependent calibration activations without replicating integer weights for individual iterations.

For fixed integer codes, calibration activations, and coupling matrices, shared scales impose the constraint $s_1 = \cdots = s_T$. Their feasible set is therefore contained in the feasible set of loop-conditioned scales. For the same fixed-code fitting objective f, this yields

$$\min_{\boldsymbol{s}\in\mathbb{R}^T_{\geq 0}} f(\boldsymbol{s}) \leq \min_{\bar{s}\geq 0} f(\bar{s}\mathbf{1}_T). \tag{61}$$

This relation characterizes the additional surrogate-fitting flexibility of loop-conditioned scales. It applies to the fixed-code subproblem, rather than guaranteeing monotonic improvement on every downstream metric.

Compared with shared scales, full TEMPO reduces LAMBADA OpenAI (OP) perplexity from 6.712 to 6.389, a relative reduction of 4.81%. LAMBADA accuracy increases from 59.75% to 60.22%, and MMLU accuracy increases from 59.70% to 60.26%, corresponding to gains of 0.47 and 0.56 percentage points, respectively. These results show that the additional temporal adaptation can translate into predictive benefits on these tasks, supporting the representation design of shared integer codes with loop-conditioned modulation.

C.3 INT2 Residuals: Supplementing the Shared W4 Backbone

INT2 residuals introduce additional low-bit correction directions beyond the shared W4 backbone. For a selected row-group, the effective weight is

$$\widehat{w}_{t,r,j} = s^{(4)}_{t,r,j} q^{(4)}_{r,j} + s^{(2)}_{t,r,j} q^{(2)}_{r,j}. \tag{62}$$

Both integer codes are shared across recurrent iterations, while the residual scales control the correction magnitude at each iteration. TEMPO fits residuals with the W4 backbone fixed and selects their locations using the achieved surrogate reduction

$$\Delta_v = J^{(4)}_v - J^{(4+2)}_v. \tag{63}$$

Additional representational capacity is thus assigned to corrections that reduce the calibration surrogate.

Relative to the residual-free variant, full TEMPO increases MMLU accuracy from 57.38% to 60.26%, a gain of 2.88 percentage points, and LAMBADA accuracy from 59.07% to 60.22%, a gain of 1.15 percentage points. Five-task mean accuracy also increases from 0.5949 to 0.6029.

The benefits extend to both perplexity metrics. WT2 perplexity decreases from 13.752 to 13.393, a relative reduction of 2.61%, while OP perplexity decreases from 6.662 to 6.389, a relative reduction of 4.10%. The simultaneous improvements in mean accuracy and both perplexity metrics support the effectiveness of the INT2 residual module and its additional capacity in supplementing the shared W4 backbone. This comparison evaluates the residual module as a whole; it does not compare allocation rules at an equal storage budget.

C.4 Connecting Component Roles to Downstream Benefits

The analysis connects three design levels of TEMPO to concrete downstream benefits. Terminal coupling changes how quantization errors are evaluated and improves WT2 and ARC-Challenge.

Setting	Value	Source
Calibration corpus	C4, training data	Manuscript/code
Main-experiment sequence count	112	Manuscript
CLI default sequence count	128	Code default
Sequence-length argument	512	Manuscript/code
Calibration batch size	1	Code default
Aligned activation reservoir	512 positions/group	Code default
Gauge-screening sample size	128 positions/group	Code default
Input-group size B	128	Manuscript/code
Output-tile height	64	Manuscript/code
Residual budget α	0.25	Manuscript/code
Calibration seed	2026	Manuscript/code

Table 3: Reported calibration settings and implementation defaults. The 112-sequence main setting requires an explicit override of `--sequences`; it is not the parser default.

Loop-conditioned scales expand the temporal adaptation of shared codes and benefit OP, LAMBADA, and MMLU. INT2 residuals introduce structured correction capacity and improve both mean accuracy and the two perplexity metrics. Together, these results provide component-level evidence on selected metrics for TEMPO's terminal-sensitivity modeling, loop-conditioned modulation, and low-bit residual representation.

D Experimental Configuration and Implementation Details

We distinguish the settings reported for the main experiments from implementation defaults and metadata recorded in the bundled evaluation files. Table 3 identifies these sources explicitly. A command-line default does not establish which value was used for a particular reported checkpoint.

D.1 Models and quantization scope

The main experiments consider Ouro-1.4B and Ouro-2.6B under W4A4 and W4A8, with BF16 models as the full-precision references. The supplied implementation loads the model and its associated tokenizer through `transformers`, enables the model-specific implementation with `trust_remote_code=True`, and uses sequential recurrent execution. The recurrent horizon T is read from the model configuration rather than overwritten by the quantizer. Calibration and runtime checks require each projection member to be called exactly T times per model forward.

The quantized projections comprise attention query, key, value, and output projections, and MLP gate, up, and down projections. Query, key, and value projections form one joint bank; gate and up projections form another. Attention output and MLP down projections are processed individually. Other modules, including embeddings, normalization layers, and the language-modeling head, are not replaced by the quantizer. Projection biases, when present, remain in BF16.

Weight codes are shared across recurrent iterations, whereas scales are indexed by iteration, output row, and input-channel group. The input-group size is 128, and the output-tile height is 64. Incomplete output tiles are zero-padded for tile statistics and residual packing; padded entries do not contribute to the model output. The implementation requires input widths to be divisible by the input-group size.

D.2 Calibration data and randomization

The data reader consumes the supplied C4 records in source order. Text records are tokenized without automatically adding special tokens; pre-tokenized records are also supported. An EOS token is appended between records when available and not already present. Tokens are concatenated and partitioned into non-overlapping chunks. With a sequence-length argument of 512, the loader produces chunks of 513 token IDs and 512 shifted next-token prediction positions. Thus, the prediction length and the number of input token IDs differ by one in this implementation.

The reader does not shuffle documents or sample random text windows. The calibration seed initializes Python, NumPy, and PyTorch random states and controls stochastic probes and activation-reservoir sampling; changing this seed does not by itself change the order of source records. Each projection group maintains a priority reservoir over token positions, using the same selected positions across all recurrent iterations. Its capacity is 512 positions per group, and the reservoir seed is the calibration seed plus the CRC32 value of the group name. The categorical-probe generator uses the calibration seed XOR `0x5F3759DF`.

The full-precision model is in evaluation mode, its parameters are frozen during statistics collection, and gradients are enabled only as needed to obtain probe responses. KV caching and gradient checkpointing are disabled during this stage. Each calibration batch uses one task-probe forward/backward pass and two Fisher-probe forward/backward passes by default. Input moments are accumulated only once per batch; output-gradient Gram matrices are accumulated for all three probes.

Full-data input moments are accumulated in FP32. Retained activation samples are stored in BF16 and converted to FP32 for representation search and moment calculations. After a gauge is selected, the transformed full-data input moments are retained, while mixed quantized/full-precision and quantized/quantized moments are estimated from the aligned activation reservoir. These estimators therefore do not all use the same number of calibration positions.

D.3 Probe construction and coupling regularization

Let V be the number of valid shifted prediction positions in a calibration batch, z_i the logits at position i, y_i the observed next token, and $p_i = \mathrm{softmax}(z_i)$. The implemented probe scalars are

$$\begin{aligned} \Phi_{\mathrm{task}} &= \frac{1}{\sqrt{V}} \sum_{i=1}^{V} -\log p_i(y_i), \\ \Phi_{\mathrm{F},m} &= \frac{1}{\sqrt{V}} \sum_{i=1}^{V} \left[z_{i,c_{m,i}} - \sum_{v} \mathrm{sg}(p_i(v)) z_{i,v} \right], \qquad c_{m,i} \sim \mathrm{Categorical}(p_i), \end{aligned} \tag{64}$$

where sg denotes stop-gradient. The sampled labels are not fed back into the input sequence. The categorical probe has logit gradient $(\mathbf{e}_{c_{m,i}} - p_i)/\sqrt{V}$.

Each probe scalar is multiplied by the square root of its configured weight before backpropagation. Both weights default to 1.0. The gradient Gram matrices are averaged over the observed probe passes; with one task probe and two Fisher probes, this gives one task and two Fisher contributions per batch, not an equal aggregate mixture of the two probe types.

The coupling construction first symmetrizes the raw Gram matrix and adds 0.10 times its nonnegative diagonal. Eigenvalues are then floored at 10^{-8} times the largest absolute eigenvalue of the corresponding matrix. Let $\overline{C}^{p,o}$ denote the resulting matrix and let N_{tile} be the total number of output tiles across projection groups. The implemented global normalization is

$$C^{p,o} = \frac{N_{\mathrm{tile}} T}{\sum_{p',o'} \mathrm{tr}(\overline{C}^{p',o'})} \overline{C}^{p,o}. \tag{65}$$

This sets the global mean diagonal coefficient to one. It uses one shared scalar, not independent per-tile normalization or a global maximum-entry normalization.

D.4 Representation search and numerical optimization

For each projection group, the gauge search evaluates six combinations of diagonal balancing and Rademacher signs. Writing $r_{x,k}$ and $r_{w,k}$ for the activation and weight RMS of channel k, the diagonal candidate is

$$D_{kk} = \mathrm{clip}\left[\left(\frac{r_{w,k} + \epsilon_{\mathrm{FP32}}}{r_{x,k} + \epsilon_{\mathrm{FP32}}} \right)^{\beta}, \frac{1}{16}, 16 \right], \qquad \beta \in \{0, 0.5, 1\}. \tag{66}$$

Implementation parameter	Default value
Diagonal-balancing exponent candidates	$\{0, 0.5, 1\}$
Rademacher sign seeds	$\{17, 73\}$
Diagonal-rescaling interval	$[1/16, 16]$
Hadamard block width	128
A4 clipping-quantile candidates	$\{0.995, 0.999, 1\}$
A8 clipping-quantile candidates	$\{0.999, 0.9999, 1\}$
Clipping-coordinate passes	1
Gauge shortlist size	2
GPTQ initialization damping	0.01
Maximum W4 refinement iterations	3
Maximum INT2 refinement iterations	4
Output-row processing chunk	128
Scale-solver pseudoinverse tolerance	10^{-7}
Relative KKT-check coefficient	2×10^{-5}
Strict-improvement threshold	$10^{-6}(1 + \lvert\mathcal{J}_{r,j}\rvert)$

Table 4: Search and solver defaults in the supplied implementation. The iteration counts are upper bounds because refinement can stop early.

Here β is the gauge-balancing exponent, distinct from the residual budget α. The activation RMS is estimated using the gauge-screening subset. Each candidate uses a normalized block Hadamard transform.

Clipping thresholds are quantiles of absolute transformed activations, computed separately for each iteration and input group. Coordinate search starts from quantile 1.0 and selects one quantile index per iteration, shared across that iteration's input groups. The best two gauge–clipping candidates are retained by the activation surrogate; thresholds for retained candidates are recomputed on the full reservoir using the selected quantile indices. The final candidate is selected by its optimized W4 surrogate.

Activations use per-token, per-input-group signed uniform quantization. For a transformed activation block x' and its calibrated threshold $c_{t,j}$, the scale and reconstruction are

$$\begin{aligned} a(x', t, j) &= \frac{\max\{\min(\|x'\|_\infty, c_{t,j}), 10^{-8}\}}{q_{\max}}, \\ Q_A(x') &= a(x', t, j)\ \mathrm{clip}\left(\mathrm{round}\ \frac{x'}{a(x', t, j)}, q_{\min}, q_{\max}\right). \end{aligned} \tag{67}$$

The activation code ranges are $[-8, 7]$ for A4 and $[-127, 127]$ for A8. The W4 and INT2 weight-code ranges are $[-8, 7]$ and $[-2, 1]$, respectively.

The GPTQ-style initializer combines quantized-activation and mixed activation moments using the output-tile-averaged coupling matrix. Its damping is 0.01 times the mean absolute diagonal of the resulting metric. The initializer uses pseudoinverse tolerance 10^{-6} and a Cholesky jitter of 10^{-6} times the mean inverse-metric diagonal, with numerical floors as implemented.

For fixed integer codes, the scale solver symmetrizes and projects the small quadratic matrix onto the positive semidefinite cone, enumerates active sets, and includes zero-scale and shared-scale candidates. The default pseudoinverse tolerance is 10^{-7}, and KKT residuals are checked using $2 \times 10^{-5} \max(1, \|\mathbf{b}\|_\infty)$.

Integer refinement in the supplied code uses rounded proposals from a diagonal approximation to the fixed-scale quadratic, followed by scale re-solving. A proposed row-group update is accepted only when its full implemented surrogate does not increase. Refinement stops early when no row-group improves by more than $10^{-6}(1 + |\mathcal{J}_{r,j}|)$. This implementation should not be confused with exhaustive integer-coordinate enumeration.

For INT2 initialization, the transformed weight residual is computed relative to the mean W4 reconstruction over iterations. Each row-group residual is divided by its RMS, floored at 10^{-8}, rounded, and clamped to $[-2, 1]$. This RMS is only an initialization scale; persisted INT2 scales are obtained from the loop-conditioned scale solver. W4 codes and scales remain fixed during residual fitting.

Environment component	Recorded value
GPU model	NVIDIA A100 80GB PCIe
Operating system	Ubuntu 20.04.6 LTS
Python	3.11.15
PyTorch	2.11.0+cu128
CUDA used to build PyTorch	12.8
NVIDIA driver	535.183.06
Transformers	4.57.1
Language Model Evaluation Harness	0.4.12
Model loading dtype	BF16

Table 5: Environment recorded in the bundled evaluation JSON files, not inferred from dependency lower bounds. The host lists two GPUs; the provided entry points place each model on one selected CUDA device.

Across all candidate tiles, the implemented selection count is

$$K_{\text{impl}} = \min\left\{\text{round}(\alpha N_{\text{total}}), \#\{v : \Delta_v > 0\}\right\}, \tag{68}$$

using Python's integer rounding, with the count constrained to $[0, N_{\text{total}}]$. The largest positive gains are selected. Thus, α is a nominal upper budget; zero-gain tiles need not receive overlays.

D.5 Evaluation protocol and recorded environment

The evaluation adapter requires `lm-eval==0.4.12` and wraps models with its Hugging Face adapter. For all three tasks, they record zero-shot evaluation and no sample limit.

The logs record evaluation seeds of 0 for Python randomness and 1234 for NumPy, PyTorch, and few-shot sampling. These are separate from the calibration seed. The recorded evaluation batch sizes are 64 for both Ouro-1.4B settings, 32 for Ouro-2.6B W4A4, and 64 for Ouro-2.6B W4A8. These are logged values; the evaluation CLI's batch-size default is `auto`.

The main paper additionally reports LAMBADA accuracy, MMLU accuracy, WikiText-2 word perplexity, and LAMBADA OpenAI perplexity. Their complete run configurations are not present in the bundled evaluation files, so their task identifiers, aggregation details, and perplexity windowing settings are not inferred from the three logged tasks. Baseline values in the main comparison are taken from the LoopQ paper, rather than represented here as reruns in the recorded environment.

D.6 Existing ablation and sensitivity configurations

The main-text component ablations use Ouro-1.4B under W4A4 and modify one component at a time: removing INT2 overlays, constraining all iterations to a shared scale, or replacing the coupling with I_T. The reported sensitivity sweeps vary the sequence count over $\{32, 64, 112\}$, the residual fraction over $\{0.125, 0.25, 0.375, 0.5\}$, and the seed over $\{0, 42, 2026, 3407\}$. The other reported factors remain at $(N_{\text{cal}}, \alpha, s) = (112, 0.25, 2026)$. These are the existing manuscript configurations, not additional runs implied by the implementation defaults.

D.7 Execution path and checkpoint accounting

The supplied runtime is a PyTorch accuracy-evaluation path. Activations are quantized and dequantized, and effective weights are materialized in BF16 before calling PyTorch's `F.linear`. Loop-specific materialized weights are cached. The `--materialize-weights` option prepopulates these caches; without it, weights are cached on first use. The implementation does not provide native INT4/INT2 GEMM kernels or establish low-bit throughput gains.

INT4 and INT2 codes are packed into byte tensors using two and four codes per byte, respectively. Weight scales, gauge diagonals, and activation-clipping thresholds are serialized in FP16; gauge signs use INT8, selected residual-tile indices use INT32, and biases use BF16. For full, equal-sized tiles with realized residual coverage ρ, the integer-code cost is $4 + 2\rho$ bits per quantized weight, or 4.5 bits at $\rho = 0.25$, before scales, indices, transforms, and metadata.

Quantized projection banks are saved to `quantized.safetensors`, with structural metadata in `manifest.json`. The loader still requires the base checkpoint for non-quantized modules. Consequently, quantized-bank file size, complete model-state storage, and runtime GPU memory including BF16 weight caches are distinct quantities.

The calibration entry point writes its calibration and quantization options, seed, inferred loop count, projection-group descriptions, and PyTorch version into checkpoint metadata. Its separate calibration report records selected-gauge diagnostics, surrogate histories, and residual-selection statistics. These run-specific records, when available, take precedence over the defaults listed above.