ARCH-CPTQ: REPLAY-FREE CONTINUAL POST-TRAINING QUANTIZATION WITH CURVATURE HISTORY AND ANCHORING

Anonymous authors
Paper under double-blind review

ABSTRACT

Calibration-based post-training quantization (PTQ) can be sensitive to differences between calibration and deployment data, especially at very low precision. Adaptive quantization addresses distribution shifts, but updates based only on current reconstruction objectives may degrade accuracy on earlier domains. We therefore formalize replay-free continual PTQ (CPTQ), which retains reconstruction history while updating a shared low-bit checkpoint using unlabeled calibration data and a fixed high-precision reference. Our derivation shows that weighted quadratic reconstruction losses on fixed inputs can be represented exactly by one fixed-dimensional curvature matrix per layer, but this representation alone does not determine how to guide subsequent updates. We propose anchored reconstruction with curvature history (ARCH)-CPTQ, which emphasizes poorly reconstructed current units, balances current and historical curvature using a Riemannian discrepancy, and penalizes changes from the previous checkpoint along historically sensitive directions. Across six large language models and five precision settings, ARCH-CPTQ improves model-averaged sequence-wide average performance (AP) and retention over equal-domain curvature averaging (CPTQ-AVG) under matched solver and representation settings, with a 2.16-percentage-point AP gain in our main W2A16 (2-bit weights, 16-bit activations) setting. In this setting, all six models achieve higher final accuracy on earlier domains, on average, than their own scores immediately after each domain was first calibrated, without replay.

1 INTRODUCTION

Low-bit quantization can reduce the memory and computation costs of large language models (LLMs) (Dettmers et al., 2022; Yao et al., 2022). Calibration-based post-training quantization (PTQ) uses representative inputs to reduce reconstruction error between high-precision and quantized layer outputs (Nagel et al., 2020; Li et al., 2021; Frantar et al., 2023). However, minimizing reconstruction error on a calibration set does not ensure accuracy on other input distributions (Hubara et al., 2021; Liu et al., 2024; Chimoto et al., 2026). This concern is particularly relevant at very low precision, where generalization degradation can be substantial and task-dependent (Liu et al., 2024). Therefore, a checkpoint initially calibrated on general text may need recalibration when deployed for specialized question answering.

Adaptive quantization addresses such shifts through target-domain recalibration (Zhao et al., 2024) and prompt-time quantization (Koike-Akino et al., 2026). Related work also studies continual adaptation of quantized vision models, including reuse of domain knowledge (Xiao et al., 2024; Deng et al., 2025), and merging target-specific quantized checkpoints (Shin et al., 2025). However, when new domains extend rather than replace existing workloads, recalibrating a shared checkpoint solely on the current reconstruction objective omits earlier-domain requirements and may degrade accuracy on those domains. This motivates a continual-learning perspective in which historical importance informs subsequent updates (Kirkpatrick et al., 2017; Zenke et al., 2017; Ritter et al., 2018). We formalize this requirement as replay-free *continual post-training quantization* (CPTQ), which updates the shared low-bit checkpoint using successive unlabeled calibration sets against a fixed

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.

high-precision reference. Calibration data are discarded after each update, while persistent state retains reconstruction history; inference uses only the checkpoint.

For this setting, we use reconstruction curvature as persistent memory of how weight errors affect layer outputs on earlier calibration inputs. This curvature is a Hessian factor of the fixed-input quadratic reconstruction loss. Our derivation shows that weighted reconstruction losses can be represented by a weighted sum of curvature factors when the reference, layer inputs, and sample weights are fixed. This allows reconstruction history to be accumulated in one fixed-dimensional matrix per quantized layer. However, representing reconstruction history does not determine how it should guide the next update. Equal-domain curvature averaging (CPTQ-AVG) provides a direct baseline, but does not emphasize poorly reconstructed current inputs, assigns equal weight to every past and current calibration domain, and introduces no separate penalty on changes from the previous checkpoint.

We hence propose anchored reconstruction with curvature history (ARCH)-CPTQ, which uses this memory to guide both reconstruction and checkpoint changes. The method gives greater weight to current calibration units that the previous checkpoint reconstructs poorly, then balances their aggregate curvature against stored history using a Riemannian discrepancy. Quantized-state anchoring adds a separate penalty on changes along historically sensitive directions. An exact reformulation of the anchored quadratic objective supplies an effective target–curvature pair to the shared GPTQ solver (Frantar et al., 2023). Frozen-checkpoint comparisons and component, fixed-coefficient, and distance-scale controls assess the value of continued updating and the proposed memory use. Across six LLMs and five precision settings, ARCH-CPTQ improves model-averaged sequence-wide average performance (AP) and retention over CPTQ-AVG under matched solver and representation settings (Tables 2 and 6), with a 2.16-percentage-point AP gain in our main W2A16 (2-bit weights, 16-bit activations) setting. In this setting, all six models achieve higher final accuracy on earlier domains, on average, than their own scores immediately after each domain was first calibrated, without replay (Appendix Figure 5).

2 Background and Problem Formulation

2.1 Calibration-Dependent Reconstruction Geometry

Quantization errors of equal magnitude can incur different output reconstruction costs depending on the calibration inputs. The calibration distribution therefore affects both the costs assigned to quantization errors and the preferred low-bit solution (Hubara et al., 2021; Liu et al., 2024; Chimoto et al., 2026). Throughout, t indexes calibration updates. We describe one of the L quantized linear layers at a time, omitting its index ℓ. Let $W^\star$ denote its fixed high-precision reference weights, and let Q be a candidate weight matrix representable under the chosen quantization protocol, written in dequantized real-valued form. We distinguish this candidate Q from Q_t, the layer weights returned by update t. Let X_t collect the layer inputs from the calibration data at update t, with N_t input tokens. With these inputs fixed, the mean output reconstruction error is (Frantar & Alistarh, 2022; Frantar et al., 2023; Chee et al., 2023)

$$\mathcal{L}_t(Q) = \frac{1}{N_t}\,\|(W^\star - Q)X_t\|_F^2 = \|W^\star - Q\|_{H_t}^2\,, \qquad H_t = \frac{1}{N_t}X_tX_t^\top, \tag{1}$$

where $\|\cdot\|_F^2$ is the sum of squared matrix entries, and $\|A\|_H^2 = \mathrm{tr}(AHA^\top)$ is the curvature-weighted squared error. Eq. (1) evaluates reconstruction error on X_t through its curvature H_t. The eigenvectors of H_t identify input directions, while its eigenvalues quantify reconstruction sensitivity along those directions. The matrix retains directional error costs even when scalar summaries of total sensitivity are similar. This activation Gram matrix is, up to a constant factor, a row-wise Gauss–Newton curvature factor for reconstruction, not the downstream task-loss Hessian. Appendix B.1 derives this curvature interpretation and its directional sensitivities.

Second-order sensitivity guides pruning and mixed-precision allocation (Hassibi & Stork, 1992; Dong et al., 2019; 2020), while PTQ methods improve low-bit performance through activation-aware reconstruction, scaling, and learned transformations (Wei et al., 2022; Lin et al., 2024; Xiao et al., 2023; Wei et al., 2023; Shao et al., 2024; Ashkboos et al., 2024; Liu et al., 2025). Other work explores quantization representations based on dense-and-sparse weights, lattice codebooks, and additive codebooks (Kim et al., 2024; Tseng et al., 2024; Egiazarian et al., 2024). GPTAQ addresses

calibration asymmetry between reference and quantized paths (Li et al., 2025). However, choosing a representation or improving current reconstruction does not by itself specify how to carry earlier reconstruction requirements into later updates without past data.

2.2 Sequential Calibration without Historical Data

At update t, an unlabeled calibration dataset $\mathcal{D}_t$ from one domain becomes available. Bold symbols $\mathbf{W}^\star$ and $\mathbf{Q}_t$ denote the full reference model and checkpoint; $W^{\star(\ell)}$ and $Q_t^{(\ell)}$ are their weights at quantized layer ℓ. Initialization at $t = 0$ uses $\mathcal{D}_0$ to construct $\mathbf{Q}_0$ and state $\mathcal{S}_0$. Before update t, we carry forward $\mathcal{S}_{t-1}$, which summarizes the information retained from earlier calibration updates. Subsequent updates take the form

$$\big(\mathbf{W}^\star,\, \mathcal{D}_t,\, \mathbf{Q}_{t-1},\, \mathcal{S}_{t-1}\big) \longmapsto \big(\mathbf{Q}_t,\, \mathcal{S}_t\big), \qquad t \geq 1.$$

Calibration data are discarded after use, but $\mathbf{Q}_t$ must continue to support earlier domains. A current-only objective omits the earlier reconstruction losses. Evaluating those losses directly would require revisiting past samples. Between updates, inference uses only the fixed checkpoint $\mathbf{Q}_t$; the reference and state are used only for calibration. Appendix A.1 compares current-only and full-data-access objectives, while Appendix Figure 4 contrasts one-time quantization, current-only recalibration, and recalibration with historical state. Replay-free CPTQ therefore requires a state that retains information about earlier reconstruction costs after discarding the calibration data.

2.3 Curvature as Replay-Free Persistent State

The curvature matrices in Eq. (1) retain these reconstruction costs without the original inputs. Because the losses share the same weight error $W^\star - Q$, their weighted sum can be represented by a single curvature matrix. Let $s = 0, \ldots, t$ index updates from initialization through the current update, and let $\rho_{t,s}$ weight the loss from update s in the current objective. We hold $W^\star$, the inputs X_s, and the weights $\rho_{t,s}$ fixed during optimization of Q. For $\rho_{t,s} \geq 0$ and $\sum_{s=0}^{t} \rho_{t,s} = 1$, we obtain

$$\bar{H}_t = \sum_{s=0}^{t} \rho_{t,s} H_s, \qquad \sum_{s=0}^{t} \rho_{t,s} \mathcal{L}_s(Q) = \|W^\star - Q\|^2_{\bar{H}_t}. \tag{2}$$

The left-hand side combines the reconstruction losses of the same candidate Q on past and current inputs; the right-hand side evaluates that objective using only $\bar{H}_t$. Retaining this aggregate lets us evaluate the weighted reconstruction objective without reloading past inputs. For a given layer, all H_s have the same shape, so the dimensions of $\bar{H}_t$ do not grow with calibration updates. In our method, $\mathcal{S}_t = \{\bar{H}_t^{(\ell)}\}_{\ell=1}^{L}$ stores the accumulated curvature separately for each quantized linear layer. Appendix B.2 shows how one curvature matrix represents these weighted reconstruction losses under a shared reference, fixed inputs, and fixed weights.

Continual learning retains parameter importance (Kirkpatrick et al., 2017; Zenke et al., 2017), function targets (Li & Hoiem, 2016), replay exemplars (Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019; Buzzega et al., 2020), or structured posterior curvature (Ritter et al., 2018). Rather than averaging past checkpoints $\mathbf{Q}_s$, we carry forward $\bar{H}_t$ to evaluate candidate weights Q through their reconstruction cost $\|W^\star - Q\|^2_{\bar{H}_t}$. However, representing this history does not determine how it should guide the next update. Unlike equal-domain curvature averaging (CPTQ-AVG), ARCH-CPTQ weights current units by reconstruction error and recursively mixes their curvature with historical curvature.

3 ARCH-CPTQ: Anchored Reconstruction with Curvature History

3.1 Overview

At update $t \geq 1$, each quantized layer uses current data $\mathcal{D}_t$, previous weights Q_{t-1}, and accumulated curvature $\bar{H}_{t-1}$. The reference $W^\star$ remains fixed (Figure 1). Reconstruction residuals from the previous checkpoint $\mathbf{Q}_{t-1}$ weight the current curvature H_t^{cur}. A blockwise discrepancy based on

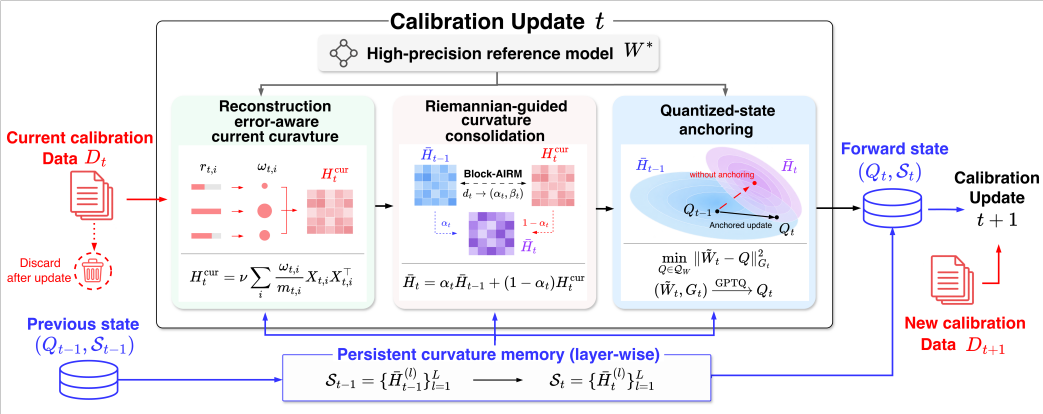


Figure 1: One ARCH-CPTQ calibration update, applied per quantized linear layer (index ℓ omitted). $\mathcal{S}_t$ stores the layer-wise curvature history; GPTQ approximately solves the reformulated reconstruction objective. Matrix shading, contours, and update arrows are schematic. The fixed reference $\mathbf{W}^\star$ serves calibration only; current data are discarded after the update, $(\mathbf{Q}_t, \mathcal{S}_t)$ is carried forward, and inference uses $\mathbf{Q}_t$ alone.

the affine-invariant Riemannian metric (AIRM), termed Block-AIRM, compares H_t^{cur} with $\bar{H}_{t-1}$ to set the history–current mixing coefficient. Anchoring then limits movement from Q_{t-1}, and GPTQ uses the resulting target–curvature pair $(\widetilde{W}_t, G_t)$ to obtain Q_t.

3.2 Reconstruction-Error-Aware Current Curvature

We emphasize units that the previous checkpoint reconstructs poorly. Let n_t be the number of calibration units (execution rows of tokens) and $m_{t,i}$ the token count of unit i, with $N_t = \sum_i m_{t,i}$. For the decoder block containing the current layer, let $Y_{t,i}^\star, Y_{t,i}^{\text{prev}} \in \mathbb{R}^{m_{t,i} \times d_h}$ denote its outputs for unit i under the fixed reference $\mathbf{W}^\star$ and previous checkpoint $\mathbf{Q}_{t-1}$, where d_h is the block-output width. The mean squared error (MSE) over tokens and output features is

$$r_{t,i} = \frac{\left\| Y_{t,i}^\star - Y_{t,i}^{\text{prev}} \right\|_F^2}{m_{t,i} d_h}. \tag{3}$$

Clipping residuals to $c_{t,i} = \min(r_{t,i}, q_{t,0.9})$, where $q_{t,0.9}$ is their batchwise 90th percentile, limits extreme values. We mix token coverage and residual emphasis with $\gamma \in [0,1]$; $\eta = 10^{-6}$ keeps normalization defined when residuals vanish. Let $X_{t,i}$ collect the columns of X_t corresponding to unit i. With these inputs fixed, we construct the calibration-unit weights $\omega_{t,i}$ and current curvature H_t^{cur} as follows:

$$\begin{aligned} \omega_{t,i} &= n_t \left[(1-\gamma) \frac{m_{t,i}}{N_t} + \gamma \frac{m_{t,i}(c_{t,i}+\eta)}{\sum_j m_{t,j}(c_{t,j}+\eta)} \right], \\ H_t^{\text{cur}} &= \nu \sum_{i=1}^{n_t} \frac{\omega_{t,i}}{m_{t,i}} X_{t,i} X_{t,i}^\top, \end{aligned} \tag{4}$$

where $\nu > 0$ is the backend scale, fixed across units and updates for each layer. The calibration-unit weights have mean one, so $\sum_{i=1}^{n_t} \omega_{t,i} = n_t$. Block-output residuals set the unit weights, while the current linear layer's token-averaged input Grams determine its directional reconstruction costs. We compute weights once from the previous checkpoint and keep them fixed during candidate-Q optimization. Appendix A.3 gives numerical details, and Appendix B.3 derives the normalization and the corresponding weighted quadratic reconstruction loss.

3.3 Riemannian-Guided Curvature Consolidation

We combine this error-aware current curvature with the accumulated curvature $\bar{H}_{t-1}$. Whereas CPTQ-AVG gives each calibration update equal weight, ARCH-CPTQ adjusts the history–current

balance according to the discrepancy between the two factors. For distance evaluation only, $\Phi(H)$ symmetrizes, damps, and trace-normalizes each factor (Appendix A.3). We compute the discrepancy and apply the coefficient schedules independently for each layer, omitting ℓ:

$$d_t = d_{\mathrm{AIRM}}\big(\Phi(\bar{H}_{t-1}), \Phi(H_t^{\mathrm{cur}})\big)\,. \tag{5}$$

The Block-AIRM discrepancy aggregates AIRM distances over 16×16 diagonal blocks of the curvature matrices using the square root of the sum of their squares (Pennec et al., 2006). Appendix A.3 specifies this computation.

Consolidation uses

$$\alpha_t = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\frac{d_t}{d_t + \kappa_\alpha}, \qquad \bar{H}_t = \alpha_t \bar{H}_{t-1} + (1-\alpha_t)H_t^{\mathrm{cur}}, \tag{6}$$

where $0 \le \alpha_{\min} \le \alpha_{\max} \le 1$ and $\kappa_\alpha > 0$. Larger discrepancies increase the historical weight α_t, limiting abrupt changes to the reconstruction objective; smaller discrepancies give current curvature greater relative weight. With n_t and ν fixed, we mix the full curvature matrices arithmetically, without applying distance normalization to the stored state. Thus $\bar{H}_t$ and the solver curvature G_t retain off-block entries. Appendix B.4 expands this recursion to show how much each update's curvature contributes to $\bar{H}_t$.

3.4 Quantized-State Anchoring

Reconstruction under the consolidated curvature $\bar{H}_t$ remains centered on the fixed reference $W^\star$. Quantized-state anchoring additionally penalizes changes from Q_{t-1}, weighted by historical curvature $\bar{H}_{t-1}$. Let $\mathcal{C}$ denote the fixed quantization configuration, including bit-widths and grouping, and let $\mathcal{Q}_W$ be its representable weight family. For positive semidefinite (PSD) curvature factors, we use the weighted squared error defined in Section 2.1 to form a proximally regularized objective (Rockafellar, 1976):

$$Q_t^{\mathrm{opt}} \in \operatorname*{arg\,min}_{Q \in \mathcal{Q}_W} \left\{ \|W^\star - Q\|^2_{\bar{H}_t} + \beta_t \|Q - Q_{t-1}\|^2_{\bar{H}_{t-1}} \right\}, \qquad \beta_t \ge 0. \tag{7}$$

The same discrepancy d_t sets the anchoring strength through a separate schedule, $\beta_t = \beta_{\min} + (\beta_{\max} - \beta_{\min})d_t/(d_t + \kappa_\beta)$, where $0 \le \beta_{\min} \le \beta_{\max}$ and $\kappa_\beta > 0$.

Solver-compatible form. The combined objective admits an equivalent reconstruction form with effective target $\widetilde{W}_t$ and curvature G_t:

$$\begin{aligned} G_t &= \bar{H}_t + \beta_t \bar{H}_{t-1}, \\ \widetilde{W}_t &= \big(W^\star \bar{H}_t + \beta_t Q_{t-1} \bar{H}_{t-1}\big)\, G_t^\dagger, \end{aligned} \tag{8}$$

where $G_t^\dagger$ is the Moore–Penrose pseudoinverse. Eq. (7) can be written as $\|\widetilde{W}_t - Q\|^2_{G_t}$ plus a constant independent of Q. Appendix B.5 gives the derivation, including singular G_t. We approximately solve this equivalent reconstruction problem with the shared GPTQ solver, obtaining $Q_t = \mathrm{GPTQ}(\widetilde{W}_t, G_t; \mathcal{C})$ (Frantar et al., 2023). Recomputing scales and zero points can shift the grid within $\mathcal{C}$, while rounding remains discrete. The solver-matched baselines use the same quantization rule.

Initialization and deployment. At $t = 0$, we use uniform unit weights, set $\beta_0 = 0$ and $\bar{H}_0 = H_0^{\mathrm{cur}}$, and construct Q_0 without anchoring. For $t \ge 1$, $\mathcal{D}_t$ is discarded after the update, while $(Q_t, \bar{H}_t)$ is retained for the next one; G_t is temporary. The state size stays fixed, and inference uses only $\mathbf{Q}_t$. The supplementary ZIP includes core reference code, a synthetic CPU demo, and numerical tests.

4 Experimental Setup

Models and quantization. We evaluated LLaMA 2 (L2) 7B/13B (Touvron et al., 2023), LLaMA 3.1 (L3.1) 8B (Grattafiori et al., 2024), Qwen2.5 (Q) 7B/14B (Yang et al., 2024), and DeepSeek-R1-Distill-Qwen-14B (Guo et al., 2025). Each calibration update produced one shared

low-bit checkpoint. Of five precision settings, W2A16 was primary; W3A16/W4A16 varied weight precision, while W4A8/W4A4 also used dynamic per-token activation quantization–dequantization (QDQ). Solver-matched comparisons used one fixed GPTQ setup (group size 32, asymmetric group-wise quantization, activation order, and 0.01 damping). We evaluated accuracy and retention with dense dequantized weights, applying activation QDQ only in W4A8/W4A4. Appendix A.3 specifies these quantization and evaluation conventions.

Sequential protocol. The canonical domain order was WikiText-2 → BoolQ → PIQA → WinoGrande → ARC-Easy → ARC-Challenge → TruthfulQA → MedQA (Merity et al., 2017; Clark et al., 2019; Bisk et al., 2020; Sakaguchi et al., 2020; Clark et al., 2018; Lin et al., 2022; Jin et al., 2021). Each domain contributed 128 source sequences of 512 tokens (65,536 tokens). WikiText-2 bootstrapped the checkpoint. Longer streams appended HellaSwag, OpenBookQA, CommonsenseQA, GSM8K, MMLU-STEM, MMLU-Humanities, C4, and SciQ in this fixed order, using the first $K = 8/12/16$ domains, including the bootstrap. WikiText-2 and C4 were excluded from accuracy-based AP and evaluated separately using perplexity (PPL; Appendix E.4). We used fixed held-out `lm-eval` (Gao et al., 2024) examples for accuracy evaluation. Past calibration data were used only by the explicit replay/joint references. Main comparisons and canonical controls used seeds $\{0, 1, 2\}$; Appendices C.3–C.5 give the remaining robustness and development/evaluation protocols. Seeds affected sampling and within-set order, but not domain order. For TruthfulQA, we split the validation set into fixed, non-overlapping calibration and evaluation subsets shared across all methods and seeds. Appendices A.3–A.4 specify execution and splits; Appendices E.1–E.3 explain curvature-based domain/order construction and the longer streams.

Comparisons and metrics. CPTQ denotes the problem setting; CPTQ-AVG denotes the solver-matched baseline that equally averages calibration-update curvatures according to Eq. (12). At each update, GPTQ quantized $\mathbf{W}^\star$ using only $\mathcal{D}_t$ and carried no history. GPTQ, CPTQ-AVG, and ARCH-CPTQ shared the solver, representation, and fixed high-precision reference $\mathbf{W}^\star$. External PTQ rows provided complementary method-level references (Lin et al., 2024; Cheng et al., 2024; Shao et al., 2024; Wang et al., 2026). Round-to-nearest (RTN) was static, while replay/joint rows served as data-access references. We compared ARCH-CPTQ with SpQR (Dettmers et al., 2024) and SliM-LLM (Huang et al., 2025) at similar packed-model bits per weight (BPW), each in its native representation (Table 4). Except for controls retaining 32-bit floating-point (FP32) curvature, ARCH-CPTQ and CPTQ-AVG stored packed curvature between updates and restored it to FP32 for the next update; inference used only the checkpoint. On both measured models, ARCH-CPTQ matched CPTQ-AVG's persistent storage (Appendices F.1–F.2) and met the prespecified mean-degradation tolerance (Appendix F.3). Average performance (AP) averages accuracy across the full checkpoint–task matrix; current-domain adaptation (CA) averages arrival-time diagonal scores. Average forgetting (AF) averages each earlier task's drop from its best pre-final score to its final score; mean backward transfer (BWT) averages final-minus-arrival scores. Higher AP/CA/BWT and lower AF indicate better performance. Unless noted otherwise, we averaged models equally per seed and reported the mean and sample standard deviation (SD) across seeds $\{0, 1, 2\}$ without imputing missing values. Appendix A.4 gives exact formulas and aggregation details.

5 Results

5.1 Overall Adaptation–Retention Trade-off

We compared sequence-wide accuracy and retention across PTQ methods (Table 1). Across five models, ARCH-CPTQ had the highest AP and BWT and the lowest AF. AP was comparable to SliderQuant, with less forgetting and higher backward transfer; SliderQuant had higher CA. Appendix D.1 reports model-level results for all four external methods. Two-model AP was similar to SliderQuant and exceeded CPTQ-AVG. At similar sequence-wide AP, ARCH-CPTQ retains earlier-domain accuracy better than SliderQuant.

5.2 Performance across Precision Settings

We fixed the GPTQ solver and representation across methods (Table 2). ARCH-CPTQ improved mean AP and BWT and reduced AF over CPTQ-AVG in all four settings and at W4A16 (Table 6). W2A16 gave the largest AP gain: every backbone improved AP and retention, and all

Table 1: Canonical $K = 8$ W2A16 results: (a) two-model AP; (b) five-model means excluding DeepSeek-R1-Distill-Qwen-14B. Entries average seeds $\{0, 1, 2\}$; static RTN AF/BWT are N/A. AP covers the full checkpoint–task matrix (Appendix A.4).

(a) Model-wise W2A16 Performance

Method	L2-7B	L2-13B	2-model Avg.
RTN	0.4467	0.4672	0.4570
GPTQ	0.4302	0.4320	0.4311
AWQ	0.4410	0.4516	0.4463
AutoRound	0.4484	0.4603	0.4544
OmniQuant	0.4474	0.4655	0.4565
SliderQuant	0.4750	0.5230	0.4990
CPTQ-AVG	0.4528	0.4833	0.4681
ARCH-CPTQ	**0.4752**	**0.5227**	**0.4990**

(b) Five-model Average Continual Metrics

Method	Models	AP ↑	AF ↓	BWT ↑	CA ↑
RTN	5	0.4292	N/A	N/A	0.4292
GPTQ	5	0.4524	0.0308	-0.0248	0.4737
AutoRound	5	0.4915	0.0193	-0.0124	0.5021
SliderQuant	5	0.5181	0.0399	-0.0001	0.5182
CPTQ-AVG	5	0.4954	0.0087	0.0039	0.4921
ARCH-CPTQ	**5**	**0.5198**	**-0.0012**	**0.0256**	**0.4979**

Table 2: Canonical solver-matched results averaged across six models and seeds $\{0, 1, 2\}$. Appendix C, Table 6, reports all five precision settings with seed-level uncertainty.

(a) Weight-only

Setting	Method	AP ↑	AF ↓	BWT ↑	CA ↑
W2A16	GPTQ	0.4652	0.0329	-0.0279	0.4891
	CPTQ-AVG	0.5078	0.0097	0.0021	0.5060
	ARCH-CPTQ	**0.5294**	**0.0009**	**0.0228**	**0.5099**
W3A16	GPTQ	0.6490	0.0105	-0.0066	0.6547
	CPTQ-AVG	0.6502	0.0105	-0.0037	0.6534
	ARCH-CPTQ	**0.6568**	**0.0013**	**0.0055**	**0.6521**

(b) Weight–activation

Setting	Method	AP ↑	AF ↓	BWT ↑	CA ↑
W4A8	GPTQ	0.6650	0.0059	-0.0024	0.6671
	CPTQ-AVG	0.6676	0.0036	-0.0005	0.6680
	ARCH-CPTQ	**0.6684**	**0.0033**	**-0.0003**	**0.6687**
W4A4	GPTQ	0.3899	0.0100	-0.0003	0.3902
	CPTQ-AVG	0.3909	0.0082	-0.0007	0.3915
	ARCH-CPTQ	**0.3981**	**0.0074**	**0.0009**	**0.3973**

six models had positive mean BWT without replay (Appendix C.1, Table 7, and Figure 5). At W2A16, each model's mean final accuracy on earlier domains exceeded its own mean arrival accuracy. W4A8 gains were smaller with slightly negative mean BWT; W3A16 had lower CA. These comparisons support ARCH-CPTQ's use of curvature history over equal-domain averaging, most clearly at W2A16.

5.3 Effects of Curvature Memory and Update Components

We compared curvature memory with references that used past calibration data (Table 3(a)). Without replay, ARCH-CPTQ outperformed limited replay and Joint-Matched in AP, AF, and BWT. Joint-All obtained higher AP, whereas ARCH-CPTQ had lower AF and higher BWT. These data-access references do not establish a downstream upper bound. Continued recalibration also improved AP over the frozen bootstrap checkpoint (Appendix C.1, Table 8).

We examined the update components (Table 3(b)). Removing residual weighting, adaptive mixing, or anchoring worsened AP and retention. Paired-bootstrap intervals support the contributions of mixing and anchoring to AP and retention (Appendix C.2). Block-AIRM had the highest AP and

Table 3: Canonical $K = 8$ W2A16 controls on L2-7B, L3.1-8B, and Q-7B. (a) Data-access references; (b) component and distance controls. Entries are equally weighted model–seed means. ARCH denotes ARCH-CPTQ; Frobenius/LogE denote normalized Frobenius/Block Log-Euclidean discrepancies. Appendix C.2 reports schedules and uncertainty.

(a) Data-Access Reference

Method	Calibration seqs.	AP ↑	AF ↓	BWT ↑	CA ↑
Replay-16	128 current +16 past	0.4879	0.0183	-0.0004	0.4882
Replay-64	128 current +64 past	0.4923	0.0265	-0.0008	0.4930
Joint-Matched	128 joint	0.4701	0.0199	-0.0010	0.4710
Joint-All	1024 joint	0.5112	0.0171	0.0150	0.4983
ARCH-CPTQ	**128 current**	**0.5006**	**0.0012**	**0.0168**	**0.4862**

(b) Component and Distance Controls

Method	AP ↑	AF ↓	BWT ↑	CA ↑
CPTQ-AVG	0.4755	0.0144	0.0023	0.4735
ARCH w/o residual weighting	0.4859	0.0019	0.0155	0.4726
ARCH w/o adaptive mixing	0.4875	0.0102	-0.0007	0.4881
ARCH w/o anchoring	0.4894	0.0086	0.0083	0.4823
ARCH-Frobenius	0.4219	0.0362	-0.0154	0.4351
ARCH-LogE	0.4386	0.0227	0.0153	0.4255
ARCH-CPTQ (Block-AIRM)	**0.5006**	**0.0012**	**0.0168**	**0.4862**

BWT and the lowest AF among the tested distances. We evaluated development-selected fixed exponential moving average (EMA) mixing and anchoring (Appendix C.3); AIRM's advantage remained after distance-scale calibration (Appendix C.4). These controls support weighting current errors and using curvature history to guide both reconstruction and checkpoint changes.

5.4 Sensitivity to the Initial Calibration Condition

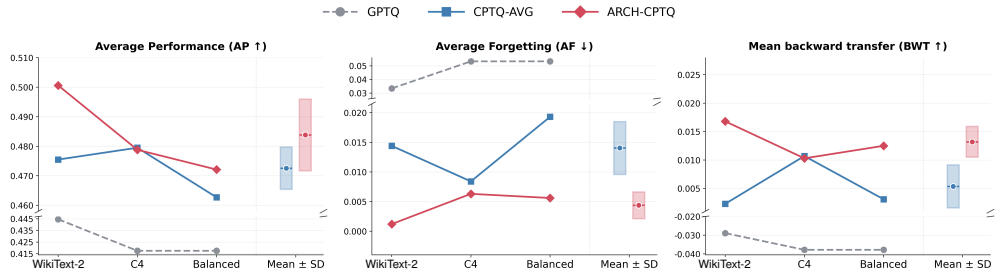


Figure 2: W2A16 bootstrap sensitivity (WikiText-2, C4, or balanced); seven later updates stayed fixed. Summaries report mean and SD directly across nine result values. Axis breaks are marked.

We varied the bootstrap source while fixing the seven later updates (Figure 2). ARCH-CPTQ had the lowest mean AF across all three sources, although AF remained positive. AP and BWT exceeded CPTQ-AVG with WikiText-2 or balanced initialization, but were slightly lower with C4. Even so, lower AF across all three sources supports forgetting reduction beyond the default WikiText-2 bootstrap.

5.5 Robustness to Domain Order and Longer Streams

We compared three domain orders chosen without downstream scores (Figure 3(a); Appendix E.2). ARCH-CPTQ had the highest mean AP and BWT and the lowest AF. We used the first 8, 12, or 16 domains of one fixed sequence (Figure 3(b); Appendix E.3). At $K = 16$, ARCH-CPTQ exceeded GPTQ in AP and had lower AF and higher BWT than CPTQ-AVG. Longer streams add evaluation tasks and do not isolate update count. The distance and schedule settings stayed fixed, while the mixing and anchoring coefficients adapted to the discrepancy. Under these settings, Block-AIRM improved AP and retention over CPTQ-AVG in all nine model–condition combinations (Appendix C.5, Table 14). ARCH-CPTQ had lower final WikiText-2 and C4 PPL than GPTQ

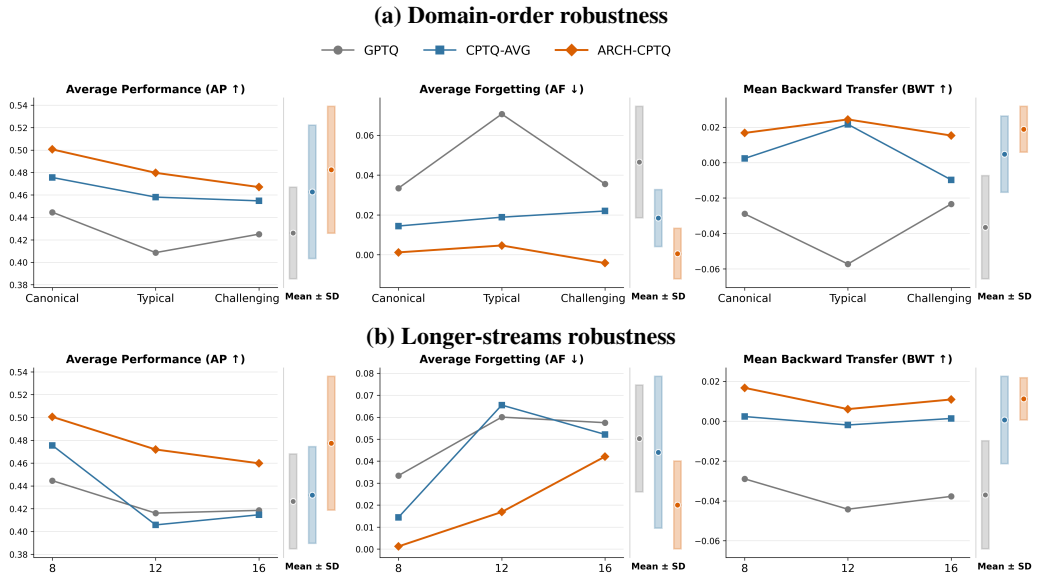


Figure 3: W2A16 robustness: (a) three $K = 8$ orders; (b) first 8, 12, or 16 domains of one fixed sequence, including the bootstrap. Each summary reports the mean and SD of its nine result values. At $K = 16$, only the C4 PPL column is excluded; all 15 checkpoint rows remain.

and CPTQ-AVG in all tested W2A16 model–corpus–stream comparisons (Appendix E.4). These comparisons support history-aware recalibration across the tested orders and expanding domain sets.

5.6 Comparison at Matched Packed-Model BPW

We compared native representations at similar packed-model BPW (Table 4). ARCH-CPTQ achieved the highest AP and BWT and the lowest AF in both bands, including positive upper-band mean BWT. This comparison does not match calibration resources or establish integer-kernel speed (Appendices F.1–F.2). Even so, lower AF and higher BWT support retention advantages at similar payload sizes.

Table 4: Packed-model BPW on L2-7B/L3.1-8B, matched within 0.15 of ARCH-CPTQ by two-model mean in each band. ARCH-CPTQ used W2A16/W3A16; baselines used native formats.

BPW band	Method	Packed-model BPW†	L2-7B AP ↑	L3.1-8B AP ↑	Avg. AF ↓	Avg. BWT ↑
3–3.5 BPW	SpQR	3.332	0.4269	0.3873	0.0138	-0.0009
	SliM-LLM	3.417	0.4142	0.3806	0.0676	-0.0420
	ARCH-CPTQ	**3.317**	**0.4752**	**0.4759**	**-0.0010**	**0.0184**
4–4.5 BPW	SpQR	4.247	0.5762	0.6266	0.0137	-0.0089
	SliM-LLM	4.333	0.5627	0.5998	0.0140	-0.0077
	ARCH-CPTQ	**4.240**	**0.5787**	**0.6334**	**0.0065**	**0.0023**

BPW† counts packed-model payloads (Appendix A.3); accuracy and retention used dense dequantized low-bit weights. Reference, curvature, and update resources are reported separately (Appendices F.1–F.2).

6 Conclusion

ARCH-CPTQ uses curvature history to guide reconstruction and limit changes to a shared low-bit checkpoint without replay. In solver-matched experiments, it improved AP and retention over CPTQ-AVG across all five precision settings, with the largest AP gain at W2A16. In canonical W2A16, all six models also finished with higher mean accuracy on earlier domains than at their arrival updates. Inference requires only the low-bit checkpoint; we use the fixed reference and curvature state during calibration. Packing reduced curvature storage by over 80% on both measured models, but calibration memory and update time remain higher (Appendices F.1–F.3). Future work can reduce this calibration cost while preserving curvature information, and test native packed-integer inference instead of the dense dequantized evaluation used here.

AI USE STATEMENT

ChatGPT (OpenAI) was used only for grammar correction and clarity refinement. The authors reviewed and are fully responsible for all content.

REFERENCES

V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Log-Euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic Resonance in Medicine*, 2006.

S. Ashkboos *et al.* QuaRot: Outlier-free 4-bit inference in rotated LLMs. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2024.

Y. Bisk, R. Zellers, R. Le Bras, J. Gao, and Y. Choi. PIQA: Reasoning about physical commonsense in natural language. in *Proc. AAAI Conference on Artificial Intelligence* (AAAI), 2020.

P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara. Dark experience for general continual learning: a strong, simple baseline. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2020.

A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. in *Proc. European Conference on Computer Vision* (ECCV), 2018.

A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny. Efficient lifelong learning with A-GEM. in *Proc. International Conference on Learning Representations* (ICLR), 2019.

J. Chee, Y. Cai, V. Kuleshov, and C. M. De Sa. QuIP: 2-bit quantization of large language models with guarantees. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2023.

W. Cheng *et al.* Optimize weight rounding via signed gradient descent for the quantization of LLMs. in *Proc. Findings of the Association for Computational Linguistics* (EMNLP), 2024.

E. A. Chimoto, M. Elhoushi, and B. Bassett. Calibrating beyond English: Language diversity for better quantized multilingual LLMs. in *Proc. Conference of the European Chapter of the Association for Computational Linguistics* (EACL), 2026.

C. Clark *et al.* BoolQ: Exploring the surprising difficulty of natural yes/no questions. in *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (NAACL-HLT), 2019.

P. Clark *et al.* Think you have solved question answering? try ARC, the AI2 reasoning challenge. arXiv preprint arXiv:1803.05457, 2018.

K. Cobbe *et al.* Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168, 2021.

Z. Deng *et al.* Test-time model adaptation for quantized neural networks. in *Proc. ACM International Conference on Multimedia* (ACM MM), 2025.

T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2022.

T. Dettmers *et al.* SpQR: A sparse-quantized representation for near-lossless LLM weight compression. in *Proc. International Conference on Learning Representations* (ICLR), 2024.

Z. Dong, Z. Yao, A. Gholami, M. W. Mahoney, and K. Keutzer. HAWQ: Hessian AWare quantization of neural networks with mixed-precision. in *Proc. IEEE/CVF International Conference on Computer Vision* (ICCV), 2019.

Z. Dong *et al.* HAWQ-V2: Hessian aware trace-weighted quantization of neural networks. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2020.

V. Egiazarian *et al.* Extreme compression of large language models via additive quantization. in *Proc. International Conference on Machine Learning* (ICML), 2024.

E. Frantar and D. Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2022.

E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh. OPTQ: Accurate quantization for generative pre-trained transformers. in *Proc. International Conference on Learning Representations* (ICLR), 2023.

L. Gao *et al.* The language model evaluation harness, 2024. URL `https://github.com/EleutherAI/lm-evaluation-harness`. Software version used: 0.4.11.dev0.

A. Grattafiori *et al.* The Llama 3 herd of models. arXiv preprint arXiv:2407.21783, 2024.

D. Guo *et al.* DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 2025.

B. Hassibi and D. G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 1992.

D. Hendrycks *et al.* Measuring massive multitask language understanding. in *Proc. International Conference on Learning Representations* (ICLR), 2021.

W. Huang *et al.* SliM-LLM: Salience-driven mixed-precision quantization for large language models. in *Proc. International Conference on Machine Learning* (ICML), 2025.

I. Hubara, Y. Nahshan, Y. Hanani, R. Banner, and D. Soudry. Accurate post training quantization with small calibration sets. in *Proc. International Conference on Machine Learning* (ICML), 2021.

D. Jin *et al.* What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 2021.

S. Kim *et al.* SqueezeLLM: Dense-and-sparse quantization. in *Proc. International Conference on Machine Learning* (ICML), 2024.

J. Kirkpatrick *et al.* Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 2017.

T. Koike-Akino, J. Liu, and Y. Wang. TTQ: Activation-aware test-time quantization to accelerate LLM inference on the fly. arXiv preprint arXiv:2603.19296, 2026.

Y. Li *et al.* BRECQ: Pushing the limit of post-training quantization by block reconstruction. in *Proc. International Conference on Learning Representations* (ICLR), 2021.

Y. Li, R. Yin, D. Lee, S. Xiao, and P. Panda. GPTAQ: Efficient finetuning-free quantization for asymmetric calibration. in *Proc. International Conference on Machine Learning* (ICML), 2025.

Z. Li and D. Hoiem. Learning without forgetting. in *Proc. European Conference on Computer Vision* (ECCV), 2016.

J. Lin *et al.* AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. in *Proc. Machine Learning and Systems* (MLSys), 2024.

S. Lin, J. Hilton, and O. Evans. TruthfulQA: Measuring how models mimic human falsehoods. in *Proc. Annual Meeting of the Association for Computational Linguistics* (ACL), 2022.

Y. Liu *et al.* Evaluating the generalization ability of quantized LLMs: Benchmark, analysis, and toolbox. arXiv preprint arXiv:2406.12928, 2024.

Z. Liu *et al.* SpinQuant: LLM quantization with learned rotations. in *Proc. International Conference on Learning Representations* (ICLR), 2025.

D. Lopez-Paz and M. Ranzato. Gradient episodic memory for continual learning. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2017.

S. Merity, C. Xiong, J. Bradbury, and R. Socher. Pointer sentinel mixture models. in *Proc. International Conference on Learning Representations* (ICLR), 2017.

T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. in *Proc. Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2018.

M. Nagel, R. A. Amjad, M. Van Baalen, C. Louizos, and T. Blankevoort. Up or down? adaptive rounding for post-training quantization. in *Proc. International Conference on Machine Learning* (ICML), 2020.

X. Pennec, P. Fillard, and N. Ayache. A Riemannian framework for tensor computing. *International Journal of Computer Vision*, 2006.

C. Raffel *et al.* Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 2020.

H. Ritter, A. Botev, and D. Barber. Online structured Laplace approximations for overcoming catastrophic forgetting. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2018.

R. T. Rockafellar. Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 1976.

K. Sakaguchi, R. Le Bras, C. Bhagavatula, and Y. Choi. WinoGrande: An adversarial winograd schema challenge at scale. in *Proc. AAAI Conference on Artificial Intelligence* (AAAI), 2020.

W. Shao *et al.* OmniQuant: Omnidirectionally calibrated quantization for large language models. in *Proc. International Conference on Learning Representations* (ICLR), 2024.

J. Shin, M. Seok, S. Kim, and E. Park. Merge-friendly post-training quantization for multi-target domain adaptation. in *Proc. International Conference on Machine Learning* (ICML), 2025.

A. Talmor, J. Herzig, N. Lourie, and J. Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. in *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (NAACL-HLT), 2019.

H. Touvron *et al.* Llama 2: Open foundation and fine-tuned chat models. arXiv preprint arXiv:2307.09288, 2023.

A. Tseng, J. Chee, Q. Sun, V. Kuleshov, and C. De Sa. QuIP#: Even better LLM quantization with Hadamard incoherence and lattice codebooks. in *Proc. International Conference on Machine Learning* (ICML), 2024.

S. Wang *et al.* SliderQuant: Accurate post-training quantization for LLMs. in *Proc. International Conference on Learning Representations* (ICLR), 2026.

X. Wei, R. Gong, Y. Li, X. Liu, and F. Yu. QDrop: Randomly dropping quantization for extremely low-bit post-training quantization. in *Proc. International Conference on Learning Representations* (ICLR), 2022.

X. Wei *et al.* Outlier Suppression+: Accurate quantization of large language models by equivalent and effective shifting and scaling. in *Proc. Conference on Empirical Methods in Natural Language Processing* (EMNLP), 2023.

J. Welbl, N. F. Liu, and M. Gardner. Crowdsourcing multiple choice science questions. in *Proc. Workshop on Noisy User-generated Text* (W-NUT), 2017.

G. Xiao *et al.* SmoothQuant: Accurate and efficient post-training quantization for large language models. in *Proc. International Conference on Machine Learning* (ICML), 2023.

J. Xiao, Z. Li, L. Yang, Y. Mei, and Q. Gu. TTAQ: Towards stable post-training quantization in continuous domain adaptation. arXiv preprint arXiv:2412.09899, 2024.

A. Yang *et al.* Qwen2.5 technical report. arXiv preprint arXiv:2412.15115, 2024.

Z. Yao *et al.* ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. in *Proc. Advances in Neural Information Processing Systems* (NeurIPS), 2022.

R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. HellaSwag: Can a machine really finish your sentence? in *Proc. Annual Meeting of the Association for Computational Linguistics* (ACL), 2019.

F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. in *Proc. International Conference on Machine Learning* (ICML), 2017.

J. Zhao *et al.* LRQuant: Learnable and robust post-training quantization for large language models. in *Proc. Annual Meeting of the Association for Computational Linguistics* (ACL), 2024.

Appendix guide. Appendix A covers implementation and metrics, B derivations, C controlled comparisons, D–E additional baselines and stream protocols, and F resources and packing. Shaded bold rows denote ARCH-CPTQ.

A Problem Definition, Protocol, and Reproducibility

A.1 Sequential Resource Contract

The replay-free contract (Section 2.2) specifies what calibration retains between updates: the checkpoint, the fixed reference, and the layer-wise curvature state defined in Section 2.3, but no past raw samples. The model-level state $\mathcal{S}_t$ is a collection of curvature matrices, one for each quantized layer; it does not sum matrices across layers. Each matrix keeps the same dimensions across updates. Figure 4 compares this contract with static and current-only PTQ.

Current-only and full-data-access references. For updates $t = 0, \ldots, T$, let $\mathfrak{Q}(\mathcal{C})$ be the model-level feasible family and $\mathcal{J}_s(\mathbf{Q})$ the reconstruction objective over the quantized layers at update s.

The current-only and full-data-access references are

$$\mathbf{Q}_t^{\mathrm{cur}} \in \arg\min_{\mathbf{Q}\in\Omega(\mathcal{C})} \mathcal{J}_t(\mathbf{Q}), \tag{9a}$$

$$\mathbf{Q}_t^{\mathrm{joint}} \in \arg\min_{\mathbf{Q}\in\Omega(\mathcal{C})} \sum_{s=0}^{t} \rho_{t,s}\mathcal{J}_s(\mathbf{Q}), \tag{9b}$$

where $\rho_{t,s} \geq 0$ and $\sum_s \rho_{t,s} = 1$. The current-only GPTQ reference requantizes the same fixed reference model using only the current calibration data at each update. Constructing the joint objective from raw data instead requires revisiting $\mathcal{D}_{0:t-1}$. These definitions alone neither imply that current-only updates must forget nor make joint reconstruction an upper bound on downstream performance.

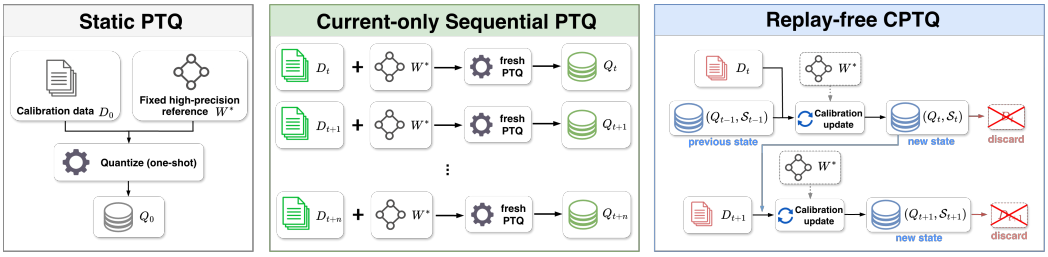


Figure 4: Update contracts. Static PTQ calibrates once; current-only PTQ recalibrates without history; replay-free CPTQ carries historical state with the checkpoint but discards calibration data.

A.2 Update Algorithm

Each quantized layer followed these steps at update t:

1. **Receive** $W^\star$, fixed inputs $\{X_{t,i}\}_{i=1}^{n_t}$, and, for $t > 0$, $(Q_{t-1}, \bar{H}_{t-1})$.
2. **Weight** units uniformly at the WikiText-2 bootstrap ($\omega_{0,i} = 1$); otherwise freeze residual weights from Eqs. (3)–(4), with the numerical guards in Table 5.
3. **Form** H_t^{cur} by Eq. (4); Appendix B.3 gives the equal-length raw-Gram equivalence.
4. **Consolidate** using Eqs. (5)–(6). At initialization, set $\alpha_0 = \beta_0 = 0$ and $\bar{H}_0 = H_0^{\mathrm{cur}}$.
5. **Anchor** through Eq. (8). Use $G_0 = \bar{H}_0$, $\widetilde{W}_0 = W^\star$ at initialization, a direct solve for $G_t \succ 0$, and the minimum-Frobenius solution in Appendix B.5 for singular G_t.
6. **Quantize** $(\widetilde{W}_t, G_t)$ with GPTQ, recomputing each target-weight group's scale and zero point.
7. **Discard** $\mathcal{D}_t$ and carry $(Q_t, \bar{H}_t)$, not temporary G_t. Calibration retains $W^\star$; inference uses only Q_t.

Appendix B.5 establishes exact objective equivalence for the reformulation using the same curvature G_t; GPTQ still solves the resulting quantization problem approximately.

A.3 Implementation Specification

All three distances used the following FP32 preprocessing once, with $S(H)$ denoting a temporary symmetrized and damped matrix:

$$\begin{aligned} \delta(H) &:= 10^{-4}\,\mathrm{mean}_j \max(|H_{jj}|, 10^{-4}), \\ S(H) &:= \tfrac{1}{2}(H + H^\top) + \delta(H)I, \\ \Phi(H) &:= S(H)/\operatorname{tr} S(H). \end{aligned} \tag{10}$$

After whole-matrix trace normalization, all three distances used diagonal blocks reconstructed with eigenvalues floored at 10^{-4} and no upper clipping. Only temporary copies were processed; stored curvature retained the backend reconstruction scale.

Block-AIRM discrepancy. For $A = \Phi(\bar{H}_{t-1})$ and $B = \Phi(H_t^{\mathrm{cur}})$, let A_b, B_b be their reconstructed blocks of at most 16×16, including the final smaller block. We use

$$d_{\mathrm{AIRM}}(A,B) := \left[\sum_b \left\|\log\left(A_b^{-1/2} B_b A_b^{-1/2}\right)\right\|_F^2\right]^{1/2} = \left[\sum_b \sum_j \log^2 \lambda_j(A_b^{-1} B_b)\right]^{1/2}, \quad (11)$$

where $\log$ is the principal matrix logarithm and λ_j are the positive generalized eigenvalues of (B_b, A_b). Flooring makes the blocks symmetric positive definite (SPD). This discrepancy differs from log-Euclidean distance $\|\log A - \log B\|_F$ (Arsigny et al., 2006). AIRM's affine invariance does not extend to the full pipeline with preprocessing, fixed blocks, and eigenvalue flooring.

These layer-curvature blocks differ from 32-column weight-quantization groups, decoder blocks whose output errors set residual weights, and curvature-codec blocks (Appendix F.3).

In the quantization specifications, b denotes the bit width of the quantity being quantized.

Table 5: Nominal W2/W3/W4A16 implementation.

Component	Executed specification
Residual weighting	Decoder-block output MSE; batchwise clipping at $q_{t,0.9}$; token-coverage/residual mixture with $\eta = 10^{-6}$ and denominator clamp 10^{-8}; no temperature or pre-residual activation/token normalization; FP32 before and after unit-mean normalization. Non-finite residuals become zero; non-finite final weights become one.
Riemannian-guided curvature consolidation	FP32 preprocessing by Eq. (10); 16×16 Block-AIRM (Pennec et al., 2006) with `torch.linalg.eigh`. Failure selects $\alpha_{\min}, \beta_{\min}$ and records a non-finite distance.
Quantized-state anchoring	$\beta_0 = 0$ at the WikiText-2 bootstrap; BoolQ onward uses the same saturating distance map as α_t and anchors to Q_{t-1}.
Weight quantizer	W2/W3/W4, A16; group size 32; asymmetric per-output-channel, groupwise input columns; activation order; no static groups; GPTQ damping 0.01. Min/max uniform codes $0, \ldots, 2^b - 1$ without MSE grid search; scale and zero point recomputed per target-weight group.
Calibration/runtime	Per domain: 128 source sequences $\times$ 512 tokens (65,536 total), reshaped without extra tokens to 32 execution rows $\times$ 2,048 tokens. Each row is one residual-weighting unit: $n_t = 32$, $m_{t,i} = 2{,}048$. Long prompts use random contiguous crops; short prompts use right padding. Model dtype `auto`; curvature, residuals, and solver arithmetic in FP32.
Export/reload	`save_pretrained` exports dense, dequantized model weights on the quantization grid for evaluation. Curvature is stored separately in packed form and restored to FP32 for the next update (Appendix F.1); Table 4 measures packed-model payloads, not this dense export.

Packed-model BPW accounting. BPW is $8B/N$, where N counts all original model parameters. Packed tensor bytes B included codes, scales, zero points, indices, outliers, metadata where present, and half-precision floating-point (FP16) embeddings, `lm_head`, and normalization parameters. Shared storage was counted once; the reference, curvature, dense evaluation copy, and file headers were excluded. Table 4 reports two-model means; within each band, ΔBPW is the baseline mean minus ARCH-CPTQ's, with $|\Delta\mathrm{BPW}| \leq 0.15$.

Coefficients. We used these precision-specific adaptive ranges:

Precision	γ	$\alpha_{\min}$	$\alpha_{\max}$	κ_α	$\beta_{\min}$	$\beta_{\max}$	κ_β
W2A16	0.75	0.10	0.75	1.0	0	7×10^{-5}	1.0
W3A16	0.50	0.10	0.70	1.0	0	1×10^{-4}	1.0
W4A16	0.75	0.10	0.75	1.0	0	6×10^{-5}	1.0

W4A8/W4A4 reused the W4A16 weight quantizer and coefficients, changing only activation-QDQ precision. A16 used native 16-bit floating activations under `torch_dtype=auto`, without explicit activation quantization.

Distance substitutions in Tables 3, 9, and 14 changed only the distance option: the W2A16 coefficients ($\gamma = 0.75$, $\kappa_\alpha = \kappa_\beta = 1$), solver, inputs, preprocessing, weighting, and packing stayed fixed. These scales were not median-calibrated.

Normalized Frobenius and Block Log-Euclidean used the same floored blocks:

$$d_{\mathrm{F}}(A, B) = \sqrt{\sum_b \|A_b - B_b\|_F^2}, \qquad d_{\mathrm{LogE}}(A, B) = \sqrt{\sum_b \|\log A_b - \log B_b\|_F^2}.$$

No further normalization by Frobenius norm, block size, or block count was used.

Fixed-mixture controls. CPTQ-AVG used coverage weighting ($\gamma = 0$), no anchoring ($\beta_t = 0$), and equal weighting of update curvatures. After initialization,

$$\bar{H}_t = \frac{t}{t+1}\bar{H}_{t-1} + \frac{1}{t+1}H_t^{\mathrm{cur}} = \frac{1}{t+1}\sum_{s=0}^{t} H_s^{\mathrm{cur}}. \tag{12}$$

The fixed-α control ("w/o adaptive mixing") used $\alpha_t = 0.425$ with adaptive β_t; the fixed–fixed control used

$$\alpha_t = 0.425, \qquad \beta_t = 3.5 \times 10^{-5}. \tag{13}$$

The other ablations set only $\gamma = 0$ ("w/o residual weighting") or $\beta_t = 0$ ("w/o anchoring").

Dynamic activation-QDQ extension. At each eligible linear call, W4A8/W4A4 quantized the current token's activation vector x to b bits; x_d denotes feature d.

Field	Executed W4A8/W4A4 activation-QDQ specification
Observer / refresh	No stored observer/statistics; scale refreshed at every non-`lm_head` `Linear` forward call.
Quantization rule	Dynamic symmetric QDQ with $q_{\min} = -2^{b-1}$, $q_{\max} = 2^{b-1} - 1$, zero point 0, and $s(x) = \max(\max_d \lvert x_d\rvert, 10^{-8})/q_{\max}$. The integer code is rounded and clipped to $[q_{\min}, q_{\max}]$ before dequantization.
Granularity / dtype	Per token over the full final hidden dimension; statistics and QDQ arithmetic in FP32; outputs in bfloat16 (BF16).
Export / reload	Dense dequantized weight-QDQ Hugging Face checkpoint and activation-QDQ pre-hook; no serialized activations, packed integer checkpoint, or integer kernel.
Interpretation	W4 plus dynamic per-token A8/A4 QDQ stress test; not a packed INT deployment benchmark.

Runtime environments. Both paths used PyTorch 2.3.0+cu121, Transformers 4.44.2, datasets 3.0.1, `lm-eval` 0.4.11.dev0, NumPy 1.26.4, CUDA 12.1, driver 570.133.07, and an NVIDIA A100 80GB PCIe.

A.4 Dataset, Calibration, Evaluation, and Metric Contract

Domains and aggregation. We used the canonical stream and six models in Section 4. Longer streams appended HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), CommonsenseQA (Talmor et al., 2019), GSM8K (Cobbe et al., 2021), MMLU-STEM and MMLU-Humanities (Hendrycks et al., 2021), C4 (Raffel et al., 2020), and SciQ (Welbl et al., 2017). The $K = 8/12/16$ streams used the first 8, 12, or 16 domains of this one fixed sequence, retaining the earlier domain order as domains were added. C4 separately supplied the C4-only and balanced bootstrap conditions. Checkpoints stayed fixed between updates. For tabulated main comparisons and canonical controls, we computed model-level metrics from unrounded values, averaged models equally per seed, and reported the mean and sample SD across seeds $\{0, 1, 2\}$ (`ddof`=1). Each caption identifies the models and conditions included. We reported fixed protocol and storage values directly rather than averaging over seeds. We kept static and sequential results separate, without imputing missing or invalid values or substituting another backend. AF and BWT are not defined for static RTN (`N/A`).

Calibration/evaluation separation. Calibration retained questions, context, choices, and optional prompt markers such as `Answer:`, but excluded answer/label targets. TruthfulQA used fixed, example-disjoint calibration and evaluation subsets of its validation split, shared across methods and seeds; evaluation used only the held-out subset.

Data-access references. Replay-16/64 added a fixed total of 16/64 past sequences to 128 current sequences (144/192 maximum). Past buffers were divided equally among earlier domains, assigning remainders in canonical order. Joint-Matched divided 128 sequences among all domains seen through update t with the same rule; Joint-All used 128 per domain (1,024 at the final eight-domain update). All used length 512 and independent seeds $\{0, 1, 2\}$.

Metric definitions. The stream length K includes the WikiText-2 bootstrap, leaving $T_K = K - 1$ downstream updates. Let $\mathcal{A}_K \subseteq \{1, \ldots, T_K\}$ index accuracy tasks by their arrival update, and let $\mathcal{P}_K = \mathcal{A}_K \cap \{1, \ldots, T_K - 1\}$ index earlier tasks. Accuracy $R^{(K)}_{t,k}$ is measured on task k after update t. Excluding the bootstrap checkpoint, we used full-matrix average performance (AP), arrival-time current-domain adaptation (CA), max-based average forgetting (AF) (Chaudhry et al., 2018), and mean backward transfer (BWT) (Lopez-Paz & Ranzato, 2017):

$$\mathrm{AP}^{(K)} = \frac{1}{T_K\,|\mathcal{A}_K|} \sum_{t=1}^{T_K} \sum_{k \in \mathcal{A}_K} R^{(K)}_{t,k}, \tag{14a}$$

$$\mathrm{CA}^{(K)} = \frac{1}{|\mathcal{A}_K|} \sum_{k \in \mathcal{A}_K} R^{(K)}_{k,k}, \tag{14b}$$

$$\mathrm{AF}^{(K)} = \frac{1}{|\mathcal{P}_K|} \sum_{k \in \mathcal{P}_K} \left(\max_{t \in \{k, \ldots, T_K - 1\}} R^{(K)}_{t,k} - R^{(K)}_{T_K,k} \right), \tag{14c}$$

$$\mathrm{BWT}^{(K)} = \frac{1}{|\mathcal{P}_K|} \sum_{k \in \mathcal{P}_K} \left(R^{(K)}_{T_K,k} - R^{(K)}_{k,k} \right). \tag{14d}$$

AP averages the included checkpoint rows and accuracy-task columns, including task scores both before and after their calibration update. CA uses each task's original arrival index, not its position after columns are selected. The checkpoint rows and task columns included in AP are

K	AP matrix	$\mathcal{A}_K$	$(\lvert\mathcal{A}_K\rvert, \lvert\mathcal{P}_K\rvert)$
8	7×7 (49 entries)	$\{1, \ldots, 7\}$	$(7, 6)$
12	11×11	$\{1, \ldots, 11\}$	$(11, 10)$
16	15×14	$\{1, \ldots, 13, 15\}$	$(14, 13)$

In each case, $\mathcal{P}_K = \mathcal{A}_K \setminus \{T_K\}$. At $K = 16$, C4 remains calibration update $\mathcal{D}_{14}$. Only its PPL column is excluded from AP; all 15 checkpoint rows, including $\mathbf{Q}_{14}$, remain in the average. SciQ therefore retains arrival index 15. We tracked $\mathrm{PPL}^{\mathrm{C4}}_t$ (`c4_ppl`; lower is better) on held-out C4 validation data after $t = 1, \ldots, 15$, without inverting or pooling it into accuracy macros. Appendix E.4 reports corpus-specific PPL separately for WikiText-2 and C4.

In Figures 2 and 3, we computed each summary's mean and SD directly across its nine result values. The longer-stream summary omits CA; CA cannot be recovered from AP and BWT. Positive BWT means that final accuracy on earlier tasks exceeds their arrival-time accuracy on average. Negative AF means that final scores exceed pre-final maxima on average. Neither statement implies improvement on every task. External references used different solvers (Appendix D.1).

B Formal Derivations

B.1 How Curvature Determines Directional Reconstruction Costs

Let d_{in} and d_{out} be the current layer's input and output widths. With fixed $X_t \in \mathbb{R}^{d_{\mathrm{in}} \times N_t}$, define $\Delta_Q = W^\star - Q \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}}$. Expanding Eq. (1) gives

$$\begin{aligned} \mathcal{L}_t(Q) &= \frac{1}{N_t} \operatorname{tr}\left(\Delta_Q X_t X_t^\top \Delta_Q^\top\right) \\ &= \operatorname{tr}\left(\Delta_Q H_t \Delta_Q^\top\right) \\ &= \operatorname{vec}(\Delta_Q)^\top \left(H_t \otimes I_{d_{\mathrm{out}}}\right) \operatorname{vec}(\Delta_Q), \end{aligned} \tag{15}$$

where vec stacks columns, $\otimes$ is the Kronecker product, and $I_{d_{\mathrm{out}}}$ is the identity. Thus $\nabla^2_{\mathrm{vec}(Q)} \mathcal{L}_t = 2H_t \otimes I_{d_{\mathrm{out}}}$. For $H_t = U_t \Lambda_t U_t^\top$ with orthonormal eigenvectors $u_{t,k}$ and corresponding eigenvalues

$\lambda_{t,k}$, we obtain

$$\mathcal{L}_t(Q) = \sum_k \lambda_{t,k} \left\| \Delta_Q u_{t,k} \right\|_2^2 . \tag{16}$$

Larger eigenvalues penalize errors more along their input directions; GPTQ's inverse-curvature compensation does not reverse this ordering. Thus H_t describes reconstruction sensitivity, not the downstream task-loss Hessian or individual-weight importance.

B.2 Representing Reconstruction History with One Curvature Matrix

For shared $W^\star$ and fixed executed inputs and weights $\rho_{t,s}$, one curvature matrix represents the weighted reconstruction history. Expanding the losses and using the linearity of the trace gives:

$$\begin{aligned} \sum_{s=0}^{t} \rho_{t,s} \mathcal{L}_s(Q) &= \sum_{s=0}^{t} \rho_{t,s} \operatorname{tr}(\Delta_Q H_s \Delta_Q^\top) \\ &= \operatorname{tr}\left(\Delta_Q \Big[\sum_{s=0}^{t} \rho_{t,s} H_s \Big] \Delta_Q^\top \right). \end{aligned} \tag{17}$$

For fixed positive-semidefinite curvature matrices, Eq. (17) lets the accumulated matrix evaluate the weighted reconstruction objective. It does not guarantee downstream retention or retain all information in past data. Replacing H_s by H_s^{cur} with fixed calibration-unit weights gives ARCH-CPTQ's weighted surrogate rather than the unweighted H_s objective. Equation (6) mixes these matrices at their reconstruction scale. Mixing preprocessed matrices need not equal preprocessing their mixture, so distance preprocessing is not applied to the stored state (Table 5).

B.3 Deriving the Residual-Weighted Reconstruction Objective

For finite residuals (Eq. (3)), $m_{t,i} > 0$, $\eta > 0$, and $0 \le \gamma \le 1$, each of the two fractions in Eq. (4) sums to one over calibration units. Multiplication by n_t gives $\sum_i \omega_{t,i} = n_t[(1-\gamma)+\gamma] = n_t$: the average calibration-unit weight is one. This fixes the average weight, not the trace, norm, or spectrum of H_t^{cur}. With the executed inputs and the weights computed from $\mathbf{Q}_{t-1}$ held fixed while optimizing Q, the weighted reconstruction loss is

$$\begin{aligned} \mathcal{L}_t^{\mathrm{err}}(Q) &:= \nu \sum_{i=1}^{n_t} \frac{\omega_{t,i}}{m_{t,i}} \left\| \Delta_Q X_{t,i} \right\|_F^2 \\ &= \operatorname{tr}\big(\Delta_Q H_t^{\mathrm{cur}} \Delta_Q^\top \big) . \end{aligned} \tag{18}$$

Decoder-block output MSE determines how much each calibration unit is weighted. The layer-input Gram matrices determine the reconstruction cost along different input directions. This construction does not jointly optimize over past raw samples. For equal-length calibration units, raw and token-average Grams differ only by a common factor absorbed into ν. Token averaging prevents unit length from being counted twice, once through the coverage weight and again through the Gram matrix. Under the coverage-only setting ($\gamma = 0$), $H_t^{\mathrm{cur}} = n_t \nu H_t$; throughout our experiments, $n_t = 32$ and the layer-specific $\nu > 0$ remained fixed, so the factor $n_t \nu$ was unchanged across updates. Changing n_t would rescale this factor. Curvature factors have shape $d_{\mathrm{in}} \times d_{\mathrm{in}}$; $W^\star$, Q, Q_t, and $\widetilde{W}_t$ have shape $d_{\mathrm{out}} \times d_{\mathrm{in}}$.

B.4 How Recursive Mixing Weights Past Curvatures

For each layer, expanding Eq. (6) from $\bar{H}_0 = H_0^{\mathrm{cur}}$ gives the weight $\rho_{t,s}$ of the curvature computed at update s. These weights are determined by the mixing coefficients already used:

$$\rho_{t,s} = \begin{cases} \displaystyle\prod_{u=1}^{t} \alpha_u, & s = 0, \\ \displaystyle (1-\alpha_s) \prod_{u=s+1}^{t} \alpha_u, & 1 \le s \le t, \end{cases} \tag{19a}$$

$$\bar{H}_t = \sum_{s=0}^{t} \rho_{t,s} H_s^{\mathrm{cur}}, \qquad \sum_{s=0}^{t} \rho_{t,s} = 1. \tag{19b}$$

For $0 \le \alpha_u \le 1$, the weights are nonnegative and sum to one, so $\bar{H}_t$ is a weighted average of the curvature matrices from updates $0, \dots, t$. If each H_s^{cur} is positive semidefinite, then $\bar{H}_t$ is positive semidefinite. Since $\beta_t \ge 0$, $G_t = \bar{H}_t + \beta_t \bar{H}_{t-1}$ is also positive semidefinite. If all H_s^{cur} are positive definite, both $\bar{H}_t$ and G_t are positive definite. These statements concern the exact recurrence without lossy storage; Appendix F.3 reports the reduced-precision performance comparison.

B.5 Reformulating the Anchored Objective for GPTQ

To express the anchored objective in GPTQ's reconstruction form, set $A_t = \bar{H}_t$, $P_t = \bar{H}_{t-1}$, and $B_t = W^\star A_t + \beta_t Q_{t-1} P_t$ in Eq. (8). For $A_t, P_t \succeq 0$ and $\beta_t \ge 0$, every vector in the null space of G_t satisfies $B_t v = 0$. Hence $B_t = B_t G_t^\dagger G_t = \widetilde{W}_t G_t$, even for singular G_t. Expanding the objective $F_t(Q)$ from Eq. (7) gives

$$\begin{aligned} F_t(Q) &= \operatorname{tr}(Q G_t Q^\top) - 2\operatorname{tr}(Q B_t^\top) + \zeta_t \\ &= \|\widetilde{W}_t - Q\|_{G_t}^2 + \zeta_t - \operatorname{tr}(B_t G_t^\dagger B_t^\top), \end{aligned} \tag{20}$$

with the original constant term

$$\zeta_t = \operatorname{tr}(W^\star A_t W^{\star\top}) + \beta_t \operatorname{tr}(Q_{t-1} P_t Q_{t-1}^\top). \tag{21}$$

With the same G_t, Eqs. (20)–(21) express the anchored objective as a reconstruction loss plus a constant independent of Q. For $G_t \succ 0$, we solve $\widetilde{W}_t G_t = B_t$ directly, without forming the pseudoinverse. In a scalar or shared-eigenvector direction with curvatures a, p, the target is $(aw^\star + \beta_t p q_{t-1})/(a + \beta_t p)$ when $a + \beta_t p > 0$: larger historical curvature relative to a strengthens anchoring. A zero denominator leaves the direction unconstrained; the pseudoinverse selects its minimum-norm value. Inversion sets the target, while error costs remain proportional to G_t and Q_{t-1} remains the anchor.

C Controlled Results and Schedule/Scale Diagnostics

We compared five precisions with matched GPTQ solvers and quantization representations (Table 6). Only W4A8 and W4A4 included activation QDQ, beyond the weight-side derivation.

Table 6: Solver-matched results across five precisions (canonical $K = 8$): six-model means $\pm$ sample SD across seeds $\{0, 1, 2\}$.

Precision	Method	Models	AP ↑	AF ↓	BWT ↑	CA ↑
W2A16	GPTQ	6	0.4652 ± 0.0037	0.0329 ± 0.0024	-0.0279 ± 0.0031	0.4891 ± 0.0028
	CPTQ-AVG	6	0.5078 ± 0.0029	0.0097 ± 0.0018	0.0021 ± 0.0015	0.5060 ± 0.0026
	ARCH-CPTQ	**6**	**0.5294 ± 0.0023**	**0.0009 ± 0.0011**	**0.0228 ± 0.0017**	**0.5099 ± 0.0021**
W3A16	GPTQ	6	0.6490 ± 0.0028	0.0105 ± 0.0019	-0.0066 ± 0.0017	0.6547 ± 0.0025
	CPTQ-AVG	6	0.6502 ± 0.0024	0.0105 ± 0.0016	-0.0037 ± 0.0014	0.6534 ± 0.0022
	ARCH-CPTQ	**6**	**0.6568 ± 0.0019**	**0.0013 ± 0.0010**	**0.0055 ± 0.0012**	**0.6521 ± 0.0018**
W4A16	GPTQ	6	0.6638 ± 0.0022	0.0079 ± 0.0015	-0.0035 ± 0.0013	0.6668 ± 0.0020
	CPTQ-AVG	6	0.6667 ± 0.0019	0.0043 ± 0.0013	-0.0016 ± 0.0011	0.6681 ± 0.0018
	ARCH-CPTQ	**6**	**0.6709 ± 0.0016**	**0.0004 ± 0.0010**	**0.0032 ± 0.0011**	**0.6682 ± 0.0015**
W4A8	GPTQ	6	0.6650 ± 0.0021	0.0059 ± 0.0017	-0.0024 ± 0.0015	0.6671 ± 0.0019
	CPTQ-AVG	6	0.6676 ± 0.0018	0.0036 ± 0.0014	-0.0005 ± 0.0012	0.6680 ± 0.0017
	ARCH-CPTQ	**6**	**0.6684 ± 0.0015**	**0.0033 ± 0.0012**	**-0.0003 ± 0.0011**	**0.6687 ± 0.0014**
W4A4	GPTQ	6	0.3899 ± 0.0034	0.0100 ± 0.0022	-0.0003 ± 0.0018	0.3902 ± 0.0030
	CPTQ-AVG	6	0.3909 ± 0.0030	0.0082 ± 0.0019	-0.0007 ± 0.0016	0.3915 ± 0.0027
	ARCH-CPTQ	**6**	**0.3981 ± 0.0026**	**0.0074 ± 0.0017**	**0.0009 ± 0.0014**	**0.3973 ± 0.0024**

ARCH-CPTQ improved AP and BWT and reduced AF over CPTQ-AVG at all five precisions. CA was lower at W3A16, and W4A8 gains were small with slightly negative BWT. The model-averaged results support better sequence-wide AP and retention than CPTQ-AVG across these precision settings, with gains that vary by precision.

C.1 Model-Level W2A16 Results

We compared backbones under canonical $K = 8$, W2A16, and group size 32 (Tables 7–8). D-14B denotes DeepSeek-R1-Distill-Qwen-14B.

Table 7: Model-level W2A16 results (canonical $K = 8$): mean ± sample SD across seeds $\{0, 1, 2\}$.

Model	Method	AP ↑	AF ↓	BWT ↑
L2-7B	GPTQ	0.4302 ± 0.0051	0.0132 ± 0.0036	-0.0076 ± 0.0131
	CPTQ-AVG	0.4528 ± 0.0068	0.0052 ± 0.0049	0.0012 ± 0.0061
	ARCH-CPTQ	$\mathbf{0.4752 \pm 0.0043}$	$\mathbf{-0.0081 \pm 0.0057}$	$\mathbf{0.0251 \pm 0.0072}$
L2-13B	GPTQ	0.4320 ± 0.0051	0.0102 ± 0.0023	-0.0012 ± 0.0071
	CPTQ-AVG	0.4833 ± 0.0062	-0.0017 ± 0.0051	0.0108 ± 0.0074
	ARCH-CPTQ	$\mathbf{0.5227 \pm 0.0056}$	$\mathbf{-0.0114 \pm 0.0063}$	$\mathbf{0.0551 \pm 0.0081}$
L3.1-8B	GPTQ	0.4225 ± 0.0003	0.0339 ± 0.0035	-0.0318 ± 0.0046
	CPTQ-AVG	0.4407 ± 0.0015	0.0237 ± 0.0058	-0.0057 ± 0.0028
	ARCH-CPTQ	$\mathbf{0.4759 \pm 0.0038}$	$\mathbf{0.0061 \pm 0.0097}$	$\mathbf{0.0117 \pm 0.0067}$
Q-7B	GPTQ	0.4811 ± 0.0051	0.0532 ± 0.0073	-0.0474 ± 0.0116
	CPTQ-AVG	0.5331 ± 0.0037	0.0144 ± 0.0103	0.0115 ± 0.0105
	ARCH-CPTQ	$\mathbf{0.5506 \pm 0.0040}$	$\mathbf{0.0056 \pm 0.0033}$	$\mathbf{0.0134 \pm 0.0057}$
Q-14B	GPTQ	0.4962 ± 0.0063	0.0433 ± 0.0081	-0.0358 ± 0.0094
	CPTQ-AVG	0.5672 ± 0.0048	0.0020 ± 0.0041	0.0017 ± 0.0053
	ARCH-CPTQ	$\mathbf{0.5748 \pm 0.0042}$	$\mathbf{0.0017 \pm 0.0035}$	$\mathbf{0.0227 \pm 0.0062}$
D-14B	GPTQ	0.5293 ± 0.0054	0.0438 ± 0.0082	-0.0438 ± 0.0082
	CPTQ-AVG	0.5696 ± 0.0048	0.0149 ± 0.0056	-0.0068 ± 0.0053
	ARCH-CPTQ	$\mathbf{0.5771 \pm 0.0042}$	$\mathbf{0.0113 \pm 0.0049}$	$\mathbf{0.0090 \pm 0.0058}$

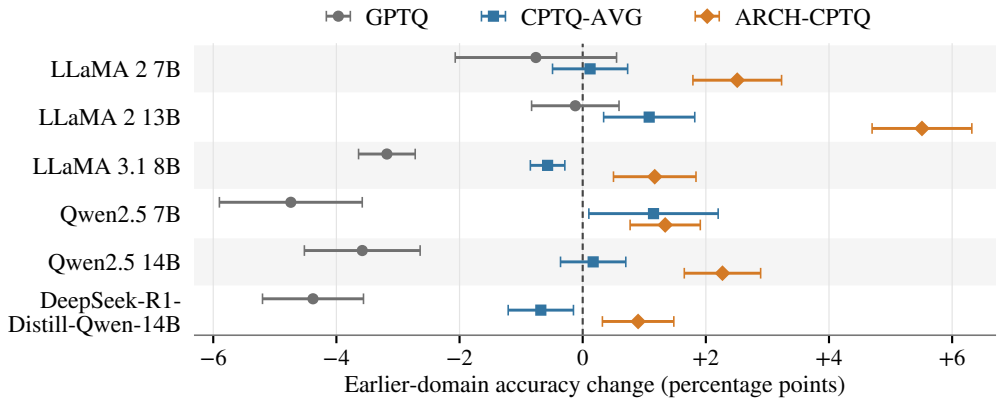


Figure 5: Final-minus-arrival accuracy on six earlier tasks: $100\times$(BWT mean ± 1 sample SD) from Table 7.

For the frozen comparison, we evaluated $\mathbf{Q}_0$ without recalibration throughout the 7×7 checkpoint–task matrix. We computed AP differences between matched model–seed runs, giving each model equal weight (Table 8); ARCH denotes ARCH-CPTQ.

Table 8: Frozen-bootstrap versus sequential AP (canonical $K = 8$, W2A16): means across seeds $\{0, 1, 2\}$.

Model	Frozen $\mathbf{Q}_0$ reference	Sequential ARCH AP	ΔAP
L2-7B	0.4305	0.4752	+0.0447
L3.1-8B	0.4543	0.4759	+0.0216
Q-7B	0.5195	0.5506	+0.0311
Model-macro average	0.4681	0.5006	+0.0325

ARCH-CPTQ exceeded frozen $\mathbf{Q}_0$ in AP on all three models. All six backbones had positive mean BWT (Figure 5): their final accuracy on earlier tasks exceeded their own arrival scores on average. This does not imply gains on every task or update. The frozen comparison supports continued recalibration with curvature history over retaining the bootstrap checkpoint.

C.2 Effects of Mixing, Anchoring, and Distance Choice

We varied mixing, anchoring, and distance (Table 9) on the three models in Table 3, following Appendix A.3 (group size 32). The fixed settings differed from the tuned control in Appendix C.3.

The abbreviations adapt. and sched. mean adaptive and scheduled; Frobenius and LogE denote Normalized Frobenius and Block Log-Euclidean; ARCH-CPTQ used Block-AIRM.

We paired nine model–seed runs and computed differences before rounding (Table 10); positive AF benefit means lower forgetting. The 95% paired-bootstrap intervals for AP, AF benefit, and BWT were above zero. The CA interval was above zero for anchoring but included zero for mixing. These intervals cover only mixing and anchoring removal, not residual-weighting or distance comparisons. The paired results support both components' contributions to AP and retention under these conditions.

Table 9: Schedule and distance controls (canonical $K = 8$, W2A16): three-model means $\pm$ sample SD across seeds $\{0, 1, 2\}$.

Variant	Schedule $(\alpha_t; \beta_t)$	AP ↑	AF ↓	BWT ↑	CA ↑
CPTQ-AVG	$t/(t+1)$; 0	$0.4755 \pm .0018$	$0.0144 \pm .0020$	$0.0023 \pm .0015$	$0.4735 \pm .0022$
fixed–fixed	0.425 ; 3.5×10^{-5}	$0.4857 \pm .0017$	$0.0178 \pm .0023$	$-0.0030 \pm .0014$	$0.4883 \pm .0021$
w/o adaptive mixing	0.425; sched.	$0.4875 \pm .0015$	$0.0102 \pm .0018$	$-0.0007 \pm .0012$	$0.4881 \pm .0018$
w/o anchoring	adapt.; 0	$0.4894 \pm .0013$	$0.0086 \pm .0011$	$0.0083 \pm .0016$	$0.4823 \pm .0019$
Frobenius	adapt.; sched.	$0.4219 \pm .0027$	$0.0362 \pm .0031$	$-0.0154 \pm .0025$	$0.4351 \pm .0030$
LogE	adapt.; sched.	$0.4386 \pm .0024$	$0.0227 \pm .0026$	$0.0153 \pm .0021$	$0.4255 \pm .0028$
ARCH-CPTQ	**adapt.; sched.**	$\mathbf{0.5006 \pm .0012}$	$\mathbf{0.0012 \pm .0009}$	$\mathbf{0.0168 \pm .0015}$	$\mathbf{0.4862 \pm .0018}$

Table 10: Full-method improvements over removal controls: means and 95% paired-bootstrap intervals across nine matched model–seed runs.

Removal control	Cells	ΔAP	ΔAF benefit	ΔBWT	ΔCA
w/o adaptive mixing	9	0.0131 [0.0109, 0.0154]	0.0090 [0.0067, 0.0114]	0.0175 [0.0154, 0.0197]	−0.0019 [−0.0044, 0.0005]
w/o anchoring	9	0.0112 [0.0094, 0.0130]	0.0074 [0.0058, 0.0091]	0.0085 [0.0063, 0.0108]	0.0039 [0.0013, 0.0065]

C.3 Comparison with Tuned Fixed Mixing and Anchoring

We tested nine fixed settings: $\alpha \in \{0.100, 0.425, 0.750\}$ and $\beta \in \{1.0, 3.5, 5.0\} \times 10^{-5}$. We selected the setting with the highest final-checkpoint accuracy on OpenBookQA, CommonsenseQA, and SciQ, rather than AP; lower configuration index broke ties (Table 11). These tasks were absent from canonical $K = 8$ but included in $K = 16$. We transferred the selected $(\alpha, \beta) = (0.75, 1.0 \times 10^{-5})$ to L3.1-8B and Q-7B without reselecting the coefficients using target-model scores, matching calibration inputs under Appendix A.3.

Table 11: Fixed-EMA development scores on L2-7B (canonical $K = 8$, W2A16, seed 0). Boldface marks the selected setting.

$\alpha \backslash \beta$	1.0×10^{-5}	3.5×10^{-5}	5.0×10^{-5}
0.100	0.3550	0.3907	0.3525
0.425	0.3488	0.3718	0.3896
0.750	**0.4469**	0.4201	0.4346

Table 12: Transfer of selected fixed-EMA $(\alpha, \beta) = (0.75, 1.0 \times 10^{-5})$: canonical $K = 8$ W2A16 means across seeds $\{0, 1, 2\}$.

Model	Method	Seeds	AP ↑	AF ↓	BWT ↑	CA ↑
L3.1-8B	Tuned fixed-EMA	0,1,2	0.4052	0.0330	−0.0179	0.4205
L3.1-8B	**ARCH-CPTQ**	**0,1,2**	**0.4759**	**0.0061**	**0.0117**	**0.4659**
Q-7B	Tuned fixed-EMA	0,1,2	0.5452	0.0246	−0.0019	0.5468
Q-7B	**ARCH-CPTQ**	**0,1,2**	**0.5506**	**0.0056**	**0.0134**	**0.5391**

ARCH-CPTQ improved AP and BWT and reduced AF on both models, turning BWT positive; CA decreased on Q-7B. This supports adaptive mixing and anchoring over the selected fixed setting, complementing the separate removal controls in Appendix C.2.

C.4 Comparing Distance Metrics after Scale Calibration

We calibrated each distance's typical scale on 1,568 common current–history curvature pairs from an L2-7B Block-AIRM trajectory (canonical $K = 8$, seed 0), using the preceding $\bar{H}_{t-1}$ as history. For each distance g, $\kappa_g = \mathrm{median}_{u \in \mathcal{U}} \delta_g^{(u)}$, where $\delta_g^{(u)}$ is distance g on pair u in the shared set $\mathcal{U}$. We fixed these scales without downstream scores and evaluated each distance on the three Table 3 models with seeds $\{0, 1, 2\}$ (nine trajectories; Table 13), including this development run. Both schedules used κ_g:

$$r_t^{(g)} = \frac{d_t^{(g)}}{d_t^{(g)} + \kappa_g}, \qquad \alpha_t^{(g)} = 0.10 + 0.65 r_t^{(g)}, \qquad \beta_t^{(g)} = 7 \times 10^{-5} r_t^{(g)}, \quad t \geq 1.$$

We omit layer indices; this ratio differs from residual MSE $r_{t,i}$. Only the discrepancy and scale changed from Appendix A.3, with matched calibration inputs within each model–seed.

Table 13: Common-pair distance-scale control (canonical $K = 8$, W2A16, group size 32): three-model means across seeds $\{0, 1, 2\}$.

Scale-calibrated distance	κ	AP ↑	AF ↓	BWT ↑	CA ↑
Block-AIRM	**23.6164**	**0.4944**	**0.0067**	**0.0089**	**0.4868**
Block Log-Euclidean	19.7693	0.4526	0.0279	−0.0175	0.4676
Normalized Frobenius	0.019365	0.4459	0.0342	−0.0156	0.4593

Scale-calibrated Block-AIRM had the highest AP and BWT and lowest AF. Its AP exceeded CPTQ-AVG but remained below default Block-AIRM ($\kappa_\alpha = \kappa_\beta = 1$). Checkpoints and history evolved differently, so the full distributions of mixing and anchoring coefficients were not matched. This supports Block-AIRM's advantage after calibrating typical distance magnitudes.

C.5 Domain-Order and Longer-Stream Robustness without Retuning

We tested the three models listed in Table 3 under canonical $K = 8$, challenging $K = 8$, and canonical $K = 16$ (Table 14). Canonical streams used the first 8 or 16 domains of one fixed sequence. Distance and schedule settings remained fixed, while mixing and anchoring coefficients adapted to discrepancy. Three of nine model–condition combinations overlapped development; calibration inputs were matched within each model–condition–seed.

We compared three-seed means with CPTQ-AVG, weighting combinations equally. Positive AF benefit means lower AF; a win required a positive unrounded difference. Worst ΔAP is the smallest AP difference across the nine combinations.

Table 14: W2A16 robustness across nine model–condition cells (seeds $\{0, 1, 2\}$): differences from CPTQ-AVG.

Distance	AP/AF/BWT wins	Mean ΔAP	Mean AF benefit	Mean ΔBWT	Worst ΔAP
Normalized Frobenius	4/9; 3/9; 3/9	+0.0023	+0.0022	-0.0057	-0.0560
Block Log-Euclidean	4/9; 3/9; 5/9	+0.0054	-0.0006	-0.0030	-0.0595
Block-AIRM	**9/9; 9/9; 9/9**	**+0.0275**	**+0.0165**	**+0.0163**	**+0.0021**

On the six additional combinations, Frobenius and LogE gained 0.03025 and 0.02655 mean AP, respectively, but their mean BWT differences across all nine remained negative. Block-AIRM improved all three metrics in all nine combinations. These results support reusing Block-AIRM's distance and schedule settings across the tested orders and stream lengths without retuning.

D External Baselines: Evaluated Models and Conditions

D.1 External PTQ Baselines: Model-Level Results

We evaluated external PTQ methods under canonical $K = 8$ W2A16 (Table 15). AutoRound and SliderQuant covered five models, excluding DeepSeek-R1-Distill-Qwen-14B; AWQ and OmniQuant covered L2-7B and L2-13B only, matching the model sets in Table 1. AP used the checkpoint–task matrix, whereas CA used arrival-time scores (Appendix A.4). These method-level comparisons are separate from the shared-GPTQ controls.

Table 15: External PTQ results: canonical $K = 8$, W2A16, seeds $\{0, 1, 2\}$ (model-wise mean $\pm$ sample SD). (a,b) Five shared models; (c,d) two L2 models. Bold rows report model-macro means without SD.

Model	AP ↑	AF ↓	BWT ↑	CA ↑
(a) SliderQuant — Five shared models				
L2-7B	0.4750 ± 0.0024	0.0558 ± 0.0031	-0.0205 ± 0.0029	0.4926 ± 0.0026
L2-13B	0.5230 ± 0.0027	-0.0141 ± 0.0025	0.0474 ± 0.0023	0.4824 ± 0.0028
L3.1-8B	0.4490 ± 0.0033	0.0822 ± 0.0036	-0.0251 ± 0.0032	0.4705 ± 0.0030
Q-7B	0.5493 ± 0.0025	0.0404 ± 0.0028	-0.0004 ± 0.0026	0.5496 ± 0.0024
Q-14B	0.5942 ± 0.0021	0.0350 ± 0.0024	-0.0017 ± 0.0022	0.5957 ± 0.0023
Five-model mean	**0.5181**	**0.0399**	**−0.0001**	**0.5182**
(b) AutoRound — Five shared models				
L2-7B	0.4484 ± 0.0026	0.0242 ± 0.0030	-0.0168 ± 0.0029	0.4590 ± 0.0027
L2-13B	0.4603 ± 0.0025	0.0148 ± 0.0026	-0.0094 ± 0.0025	0.4705 ± 0.0026
L3.1-8B	0.4405 ± 0.0033	0.0267 ± 0.0035	-0.0192 ± 0.0032	0.4637 ± 0.0030
Q-7B	0.5281 ± 0.0027	0.0160 ± 0.0028	-0.0081 ± 0.0026	0.5572 ± 0.0026
Q-14B	0.5802 ± 0.0023	0.0148 ± 0.0025	-0.0085 ± 0.0024	0.5601 ± 0.0024
Five-model mean	**0.4915**	**0.0193**	**−0.0124**	**0.5021**
(c) AWQ — L2 models only				
L2-7B	0.4410 ± 0.0027	0.0286 ± 0.0032	-0.0164 ± 0.0029	0.4557 ± 0.0028
L2-13B	0.4516 ± 0.0025	0.0202 ± 0.0027	-0.0108 ± 0.0025	0.4609 ± 0.0026
Two-model mean	**0.4463**	**0.0244**	**−0.0136**	**0.4583**
(d) OmniQuant — L2 models only				
L2-7B	0.4474 ± 0.0025	0.0249 ± 0.0030	-0.0132 ± 0.0027	0.4602 ± 0.0026
L2-13B	0.4655 ± 0.0024	0.0177 ± 0.0026	-0.0079 ± 0.0024	0.4741 ± 0.0025
Two-model mean	**0.4565**	**0.0213**	**−0.0106**	**0.4672**

On the five shared models, ARCH-CPTQ had similar AP to SliderQuant and higher AP than AutoRound, with lower AF and higher BWT than both (Table 1(b)). Both baselines had higher CA; the comparison highlights stronger earlier-domain retention at competitive AP.

E Domain Order and Longer-Stream Evaluation

E.1 Domain Selection

We selected domains by curvature distance, without downstream scores (Section 4). They spanned 98.0% of the candidates' Block-AIRM range and both clusters (Figure 6(a)). Similar distance rankings across four reference checkpoints support consistent domain selection, not identical curvature matrices (Figure 6(b)).

E.2 Domain Order Construction

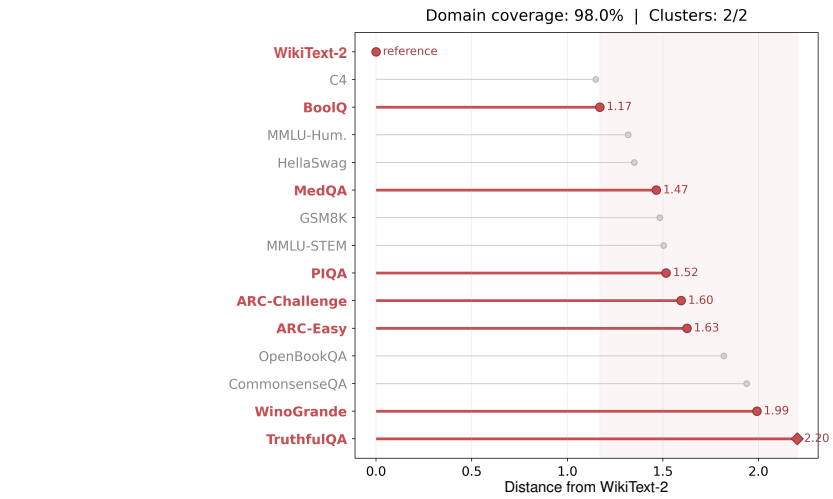
We fixed WikiText-2 first and scored all $7! = 5{,}040$ remaining-domain permutations (Figure 7). For each fixed reference, we applied Φ (Eq. (10)) to unweighted input Gram matrices and combined pairwise Block-AIRM distances across quantized linear layers as the square root of the sum of squared distances. These distances used neither residual weights nor the previous quantized checkpoint.

We summed each order's seven adjacent-domain distances, standardized scores within each reference, then averaged the four standardized scores. Percentiles were the percentages of scores no greater than each order's score. The canonical order was fixed before scoring (5.595238th percentile); Typical and Challenging were closest to the 50th and 95th percentiles. Neither required the smallest-`order_index` tie-break. Selection used no random seed or downstream score. The orders were:

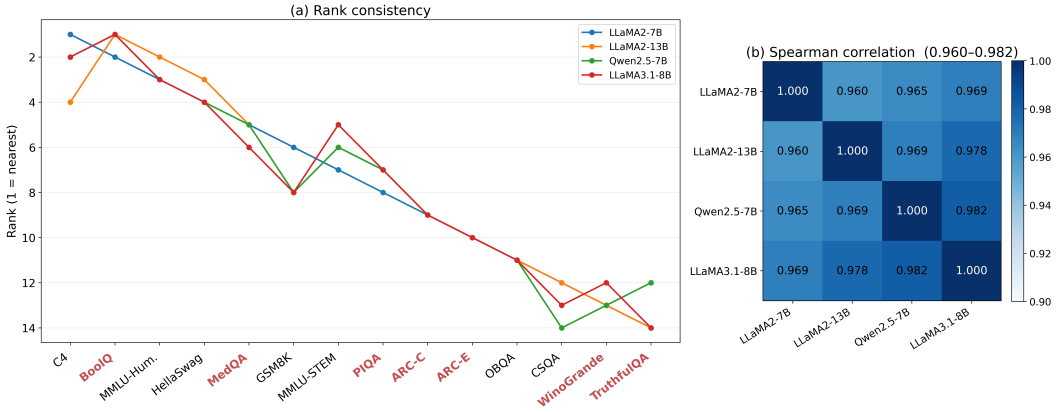
Canonical (index 0; 5.595238th): WikiText-2, BoolQ, PIQA, WinoGrande, ARC-Easy, ARC-Challenge, TruthfulQA, MedQA.

Typical (index 2368; 50th): WikiText-2, ARC-Easy, PIQA, TruthfulQA, ARC-Challenge, MedQA, BoolQ, WinoGrande.

Challenging (index 4244; 95th): WikiText-2, TruthfulQA, MedQA, PIQA, ARC-Challenge, WinoGrande, BoolQ, ARC-Easy.



(a) Domain-set coverage and candidate construction



(b) Cross-model domain-distance consistency

Figure 6: **Curvature-based domain selection.** (a) Selected domains spanned 98.0% of the candidate Block-AIRM range and both clusters. (b) Cross-checkpoint rank agreement: Spearman $\rho_S = 0.960$–0.982. OBQA and CSQA denote OpenBookQA and CommonsenseQA, respectively.

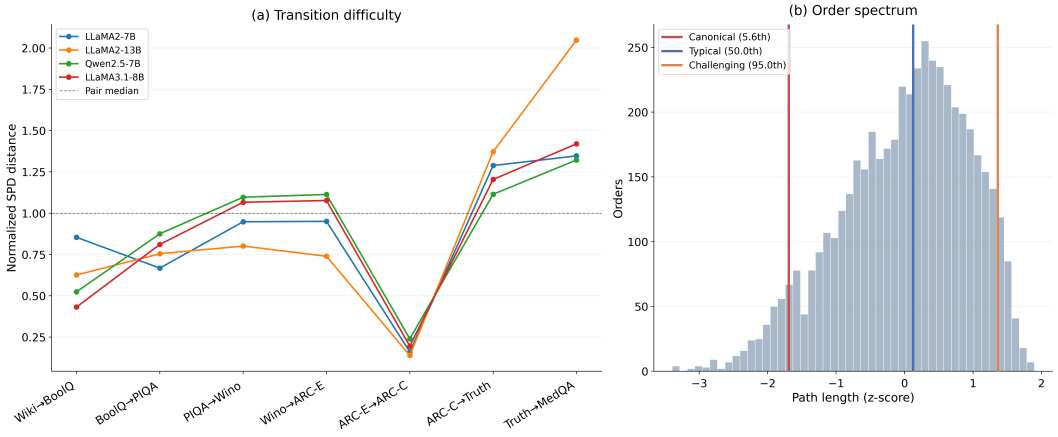


Figure 7: **Domain orders selected without downstream scores.** Left: distances for the seven canonical transitions at four reference checkpoints; the ordering of checkpoints by distance differed between transitions. Right: selected orders near the 5.6th, 50th, and 95th percentiles of all $7! = 5{,}040$ paths.

E.3 Longer Streams: Construction and Results

We used the first 8, 12, or 16 domains of one fixed sequence, preserving their order. Each evaluated stream of length K included the WikiText-2 bootstrap and $K-1$ subsequent updates (Table 16; Figure 3(b)). Appendix A.4 defines the evaluation matrices. At $K = 16$, C4 remained calibration update $\mathcal{D}_{14}$: AP excluded only its PPL column and retained all 15 checkpoint rows, including $\mathbf{Q}_{14}$. SciQ retained its original arrival index of 15.

Table 16: W2A16 longer-stream results using the first 12 or 16 domains of the same fixed sequence (mean $\pm$ sample SD, seeds $\{0, 1, 2\}$). Table 7 gives canonical $K = 8$ results.

K	Model	Method	AP ↑	AF ↓	BWT ↑
12	L2-7B	GPTQ	0.3976 ± 0.0062	0.0643 ± 0.0087	-0.0567 ± 0.0104
		CPTQ-AVG	0.4045 ± 0.0058	0.0595 ± 0.0074	0.0360 ± 0.0091
		ARCH-CPTQ	$\mathbf{0.4466 \pm 0.0047}$	$\mathbf{0.0086 \pm 0.0042}$	$\mathbf{0.0156 \pm 0.0065}$
12	L3.1-8B	GPTQ	0.3932 ± 0.0069	0.0279 ± 0.0061	-0.0010 ± 0.0075
		CPTQ-AVG	0.4143 ± 0.0052	0.0203 ± 0.0054	0.0011 ± 0.0068
		ARCH-CPTQ	$\mathbf{0.4229 \pm 0.0046}$	$\mathbf{0.0175 \pm 0.0049}$	$\mathbf{0.0042 \pm 0.0059}$
12	Q-7B	GPTQ	0.4576 ± 0.0073	0.0879 ± 0.0091	-0.0748 ± 0.0112
		CPTQ-AVG	0.3986 ± 0.0081	0.1167 ± 0.0104	-0.0428 ± 0.0097
		ARCH-CPTQ	$\mathbf{0.5463 \pm 0.0051}$	$\mathbf{0.0245 \pm 0.0058}$	$\mathbf{-0.0016 \pm 0.0063}$
16	L2-7B	GPTQ	0.3838 ± 0.0076	0.0806 ± 0.0093	-0.0706 ± 0.0115
		CPTQ-AVG	0.4037 ± 0.0063	0.0576 ± 0.0078	-0.0069 ± 0.0086
		ARCH-CPTQ	$\mathbf{0.4157 \pm 0.0052}$	$\mathbf{0.0562 \pm 0.0066}$	$\mathbf{0.0057 \pm 0.0074}$
16	L3.1-8B	GPTQ	0.3814 ± 0.0070	0.0445 ± 0.0072	-0.0127 ± 0.0089
		CPTQ-AVG	0.4071 ± 0.0061	0.0352 ± 0.0065	-0.0091 ± 0.0077
		ARCH-CPTQ	$\mathbf{0.4127 \pm 0.0050}$	$\mathbf{0.0288 \pm 0.0056}$	$\mathbf{-0.0010 \pm 0.0068}$
16	Q-7B	GPTQ	0.4904 ± 0.0078	0.0474 ± 0.0080	-0.0299 ± 0.0098
		CPTQ-AVG	0.4332 ± 0.0072	0.0637 ± 0.0085	0.0201 ± 0.0092
		ARCH-CPTQ	$\mathbf{0.5510 \pm 0.0054}$	$\mathbf{0.0413 \pm 0.0062}$	$\mathbf{0.0280 \pm 0.0076}$

ARCH-CPTQ had the highest AP and lowest AF in all six model–length comparisons and the highest BWT except on L2-7B at $K = 12$, where CPTQ-AVG had higher BWT (Table 16). Some BWT values remained negative. Adding domains increases both calibration updates and evaluation tasks, so this comparison does not isolate update count. Across the expanded domain sets, the results support higher AP and lower forgetting.

E.4 WikiText-2 and C4 Language-Modeling Performance

We evaluated WikiText-2 and C4 separately from accuracy-based AP under the same W2A16 streams (group size 32; Table 17). WikiText-2 was evaluated at bootstrap $\mathbf{Q}_0$ and final $\mathbf{Q}_7$/$\mathbf{Q}_{15}$; C4 was evaluated after its arrival update $\mathbf{Q}_{14}$ and after SciQ at $\mathbf{Q}_{15}$.

Table 17: WikiText-2 and C4 PPL ↓: mean $\pm$ sample SD across seeds $\{0, 1, 2\}$, separately by model; W2A16, group size 32.

Model	Method	WikiText-2		C4 ($K = 16$)	
		$K = 8$ Final $\mathbf{Q}_7$	$K = 16$ Final $\mathbf{Q}_{15}$	Arrival $\mathbf{Q}_{14}$	Final $\mathbf{Q}_{15}$
L2-7B	GPTQ	69.4681 ± 0.0068	83.5330 ± 0.0076	16.8725 ± 0.0034	24.0278 ± 0.0047
	CPTQ-AVG	24.7404 ± 0.0037	16.7682 ± 0.0035	16.4385 ± 0.0029	16.2414 ± 0.0031
	ARCH-CPTQ	$\mathbf{14.7466 \pm 0.0029}$	$\mathbf{14.2997 \pm 0.0027}$	$\mathbf{16.4092 \pm 0.0026}$	$\mathbf{16.0071 \pm 0.0028}$
L3.1-8B	GPTQ	70.4006 ± 0.0071	102.8343 ± 0.0084	51.3844 ± 0.0059	57.9151 ± 0.0065
	CPTQ-AVG	45.1757 ± 0.0046	38.4284 ± 0.0042	53.4486 ± 0.0051	47.9889 ± 0.0048
	ARCH-CPTQ	$\mathbf{40.5099 \pm 0.0038}$	$\mathbf{36.5521 \pm 0.0039}$	$\mathbf{40.4345 \pm 0.0044}$	$\mathbf{40.1663 \pm 0.0041}$
Q-7B	GPTQ	65.6244 ± 0.0063	34.6491 ± 0.0049	71.5290 ± 0.0072	73.9145 ± 0.0077
	CPTQ-AVG	15.0841 ± 0.0032	14.4653 ± 0.0027	23.7809 ± 0.0038	22.3795 ± 0.0035
	ARCH-CPTQ	$\mathbf{13.0849 \pm 0.0021}$	$\mathbf{12.8404 \pm 0.0022}$	$\mathbf{17.3251 \pm 0.0031}$	$\mathbf{17.1955 \pm 0.0029}$

WikiText-2 bootstrap $\mathbf{Q}_0$ (identical across reported methods and streams): L2-7B 31.9089 ± 0.0041; L3.1-8B 37.5372 ± 0.0053; Q-7B 13.2627 ± 0.0028. C4 is outside the $K = 8$ calibration prefix.

ARCH-CPTQ had lower final PPL than GPTQ and CPTQ-AVG in all nine model–corpus–stream comparisons: WikiText-2 at two stream lengths and C4 at $K = 16$ on three models. Its WikiText-2 PPL improved from bootstrap except on L3.1-8B at $K = 8$ (40.5099 versus 37.5372), while its C4 PPL improved from arrival on all three models. Although bootstrap-level retention is not uniform, these nine comparisons support better final language-modeling performance than the baselines.

F Storage, Runtime Costs, and Curvature Precision

F.1 What the Storage Totals Include

ARCH-CPTQ and CPTQ-AVG stored accumulated layer-wise curvature in packed form between updates. Each update read and Zstd-decompressed the state, restored dense FP32 curvature for calibration, then packed and stored the updated curvature. This state is separate from the checkpoint and excludes temporary solver curvature G_t.

Storage included the fixed reference, dense dequantized checkpoint, and curvature (Table 18(a)). The reference model and bootstrap checkpoint each occupied 13.479 GB for L2-7B and 16.070 GB for L3.1-8B. Packed curvature occupied 4.778 GB and 7.160 GB, respectively. We summed bytes before rounding and included embedded metadata. Shared data, including bytewise-verified aliases for query/key/value (Q/K/V) projections or gate/up, were counted once; separate metadata occupied 0 GB. Unlike Table 4, these totals include the reference.

F.2 Update and Inference Costs

We measured peak CPU and GPU memory over all layer-wise updates in each eight-domain run (Table 18(b)). Mean per-domain update time included restoring curvature to FP32, but excluded saving the checkpoint. Inference used only the dense dequantized checkpoint, with batch size 1 and 2,048 tokens. Activation QDQ was used for A8/A4 only; A16 used native floating-point activations. Storage is reported in GB and memory in GiB. We did not measure packed-integer inference.

Packing reduced curvature storage by 83.2%/81.7% on L2-7B/L3.1-8B. ARCH-CPTQ matched CPTQ-AVG's storage with 21.1%/17.5% longer updates. Against all-FP32 history, packing increased CPU peaks and update time but left GPU peaks and dense-inference memory unchanged. On these two models, ARCH-CPTQ requires more calibration time and peak memory than CPTQ-AVG but the same persistent storage. Inference still uses only the checkpoint.

Table 18: Canonical $K = 8$ resources: (a) persistent storage and packing error; (b) update and dense-inference costs. Storage totals include the fixed reference.

(a) Persistent storage and packing error

Model	Method / curvature format	Persistent-curvature storage (GB)	Reference-included total (GB)	Curvature rel. error
L2-7B	GPTQ, no state	0.00	26.96	N/A
L2-7B	CPTQ-AVG, packed	4.78	31.74	1.31×10^{-4}
L2-7B	**ARCH-CPTQ, packed**	**4.78**	**31.74**	$\mathbf{1.31\times10^{-4}}$
L2-7B	ARCH-CPTQ, all-FP32	28.40	55.36	0
L3.1-8B	GPTQ, no state	0.00	32.14	N/A
L3.1-8B	CPTQ-AVG, packed	7.16	39.30	1.89×10^{-4}
L3.1-8B	**ARCH-CPTQ, packed**	**7.16**	**39.30**	$\mathbf{1.89\times10^{-4}}$
L3.1-8B	ARCH-CPTQ, all-FP32	39.19	71.33	0

(b) Update memory, inference, and update time

Model	Method / curvature format	Peak update GPU memory (GiB)	Peak update CPU memory (GiB)	Dense infer. GPU mem. (GiB)	Update time per domain (s)
L2-7B	GPTQ, no state	5.07	2.55	12.94	738.0
L2-7B	CPTQ-AVG, packed	5.49	3.25	12.94	902.6
L2-7B	**ARCH-CPTQ, packed**	**8.06**	**3.72**	**12.94**	**1093.2**
L2-7B	ARCH-CPTQ, all-FP32	8.06	3.21	12.94	916.7
L3.1-8B	GPTQ, no state	7.29	4.38	16.45	715.8
L3.1-8B	CPTQ-AVG, packed	8.06	5.64	16.45	1008.9
L3.1-8B	**ARCH-CPTQ, packed**	**11.82**	**6.19**	**16.45**	**1185.4**
L3.1-8B	ARCH-CPTQ, all-FP32	11.82	4.93	16.45	974.7

F.3 Effect of Curvature Precision on Performance

We compared packed and all-FP32 curvature at $K = 8$, W2A16, group size 32, and seeds $\{0, 1, 2\}$ (Table 19(a)). The codec kept FP32 diagonals and scaled blocks to prevent overflow during lossy FP16 conversion above the diagonal. It then applied lossless byte-shuffling and Zstd-9 compression. Decompression and inverse shuffling recovered the encoded FP16 bits exactly, not the original FP32 values. Stored scales restored upper entries to FP32; symmetry restored the lower triangle. All restored values were finite. We computed relative matrix error between restored $\widehat{H}^{(\ell)}$ and original $\bar{H}^{(\ell)}$ as $\left(\sum_\ell \|\widehat{H}^{(\ell)} - \bar{H}^{(\ell)}\|_F^2 / \sum_\ell \|\bar{H}^{(\ell)}\|_F^2\right)^{1/2}$ (Table 18).

Table 19: Packed versus all-FP32 curvature: (a) continual metrics (mean ± sample SD); (b) L2-7B six-past-task BWT decomposition.

(a) Packed curvature state fidelity

Model	Curvature format	AP ↑	AF ↓	BWT ↑
L2-7B	**Packed**	$\mathbf{0.4752 \pm 0.0043}$	$\mathbf{-0.0081 \pm 0.0057}$	$\mathbf{0.0251 \pm 0.0072}$
	All-FP32	0.4724 ± 0.0041	-0.0015 ± 0.0035	0.00153 ± 0.0035
L3.1-8B	**Packed**	$\mathbf{0.4759 \pm 0.0038}$	$\mathbf{0.0061 \pm 0.0097}$	$\mathbf{0.0117 \pm 0.0067}$
	All-FP32	0.4775 ± 0.0040	0.0052 ± 0.0063	0.0132 ± 0.0065

(b) L2-7B previous-task BWT decomposition

Curvature format	Mean arrival accuracy	Mean final accuracy	BWT ↑
Packed	0.46380	0.48890	0.02510
All-FP32	0.47860	0.48013	0.00153
Packed − FP32	−0.01480	+0.00877	+0.02357

All six model–metric comparisons (two models; AP, AF, and BWT) met the prespecified 0.002 tolerance on degradation of three-seed means. We averaged arrival $R_{k,k}$ and final $R_{7,k}$ scores over six earlier tasks ($k = 1, \ldots, 6$) within seeds, then across seeds (Table 19(b)). Final minus arrival equals BWT (Eq. (14d)); these means differ from AP and CA. Both higher final and lower arrival accuracy contributed to L2-7B's BWT gain. These comparisons concern observed means; they do not establish identical checkpoints or update sequences, or non-inferiority based on confidence intervals. The positive-semidefiniteness result in Appendix B.4 applies to the exact recurrence and need not hold after lossy conversion. With the storage measurements (Table 18), these mean comparisons (Table 19) support packed curvature as a lower-storage history representation for the two evaluated models.