# CAPQUANT: COVARIANCE-AWARE PERMUTATION FOR GROUP-WISE ACTIVATION QUANTIZATION OF LARGE LANGUAGE MODELS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Low-bit Post-training Quantization (PTQ) is widely used to reduce the memory and latency footprint of large language models (LLMs), but activation outliers can compromise accuracy. Recent methods address this challenge by dividing channels into fixed-size groups that share a quantization scale. Which channels share a scale matters, making their ordering an important design choice. We introduce **CAPQuant**, a training-free method that optimizes this ordering using activation means and cross-channel covariances. To derive this ordering, we model the error introduced by asymmetric groupwise quantization. Our derivation shows that the variance term remains constant across groups, whereas mean differences and covariances influence the grouping decision. We express these relationships as pairwise channel distances and use offline clustering to minimize their sum within fixed-size groups. The resulting permutation is folded into the adjacent weights and fused with the activation kernels, requiring no additional matmul or separate activation pass. Across Qwen3-4B, Llama-3.2-1B, Llama-3-8B, and Llama-2-7B, grouping and ordering together reduce perplexity by $1.5\%$–$9.0\%$ and increase zero-shot accuracy by $0.94$–$2.25$ percentage points over OSTQuant. Relative to contiguous grouping, ordering alone reduces perplexity in 15 of 16 comparisons by up to $2.5\%$ and improves accuracy in 7 of 8 comparisons by up to $1.49$ points. Perplexity gains also extend to Llama-2-13B and Llama-30B. CAPQuant also reduces perplexity on the frozen QuaRot- and SpinQuant-quantized Llama-3-8B models by up to $10.7\%$ and $4.1\%$, respectively. The fused permutation merely adds $3.6$–$6.9\%$ additional time to the prefill latency. These results support channel ordering as a calibration-time improvement for groupwise quantization.

## 1 INTRODUCTION

Deploying large language models at low precision requires quantizing both weights and activations. Weight-only quantization at 4 bits is comparatively well handled by GPTQ (Frantar et al., 2022) and AWQ (Lin et al., 2026), but activation quantization remains challenging because a small number of channels can carry outlier values, affecting the shared per-token or per-tensor scale (Dettmers et al., 2022; Xiao et al., 2024). *Group-wise* activation quantization, which assigns one scale to each block of $g$ consecutive channels, bounds how far an outlier can propagate and has been widely adopted by the MXFP4 format at $g = 32$ (Rouhani et al., 2023; Yuan et al., 2023; Ashkboos et al., 2024a). However, the channels placed together determine the range that this shared scale must cover and how accurately their values can be represented. Channel ordering, therefore, offers a way to reduce quantization error without changing the group size or bit width.

Existing methods reduce outliers directly rather than selecting which channels share a scale. Transformations spread outlier magnitude across the basis through smoothing, rotation, or learned orthogonal and affine maps (Xiao et al., 2024; Ashkboos et al., 2024b; Liu et al., 2025; Hu et al., 2025; Sun et al., 2024), and extraction keeps a few outlier channels at higher precision (Dettmers et al., 2022; Ashkboos et al., 2024a). The permutations that do exist order channels by a marginal statistic such as magnitude or range (Zhao et al., 2024; Yuan et al., 2023), and none is derived from the error of the shared scale itself. A transformation fixes the basis but leaves the channel order open. In OSTQuant,

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.

the two RMSNorm outputs and the MLP SwiGLU product still share a single scale across the entire 2560- or 9728-wide row, which is precisely the regime that a permutation acts on.

We introduce **CAPQuant**, a training-free method that chooses the channel order for group-wise asymmetric activation quantization. Each group shares a scale and a zero-point determined by its activation range. CAPQuant uses channel means and cross-channel covariance to construct groups, then refines them through channel swaps that reduce a quantization-derived dispersion objective (Figure 1). We apply it to the transformed activations of an existing low-bit model and fold the resulting permutation into adjacent weights and activation kernels (Figure 2).

This allows CAPQuant to build on existing quantization methods without retraining their weights or transformations. We evaluate it on OSTQuant and apply it to frozen QuaRot and SpinQuant models, where it reduces perplexity by $10.3$–$10.7\%$ and $2.4$–$4.1\%$, respectively (Table 1). These results show that optimizing channel order can improve the quality of models produced by different base methods.

Our contributions are as follows.

- **Derived quantization objective.** Starting from the expected error of asymmetric per-token per-group INT4 quantization, we bound the per-group range by the within-group dispersion and reduce the dispersion exactly to a sum of pairwise channel distances $d_{ij} = \mathbb{E}_t[(y_{ti} - y_{tj})^2]$. Expanding $d_{ij}$ shows that the per-channel variances contribute a partition-independent constant, leaving $D_{ij} = (m_i - m_j)^2 - 2C_{ij}$ as the only quantity the ordering can change. Prior orderings cluster on marginal statistics and cannot see the covariance term (Section 4).
- **Balanced refinement of partitions.** We refine fixed-size groups through channel swaps that strictly decrease the projected dispersion objective (Algorithm 1). The procedure starts from a hierarchical clustering of the channels and accepts any balanced starting partition.
- **Efficient inference integration.** CAPQuant incorporates the fixed ordering into adjacent weights and activation kernels, requiring no retaining, additional matrix multiplication, or separate activation pass.
- **Accuracy gains and measured permutation overhead.** Across four model families and two corpora, grouping and ordering together improve on OSTQuant in every reported perplexity and accuracy comparison, by $1.5$–$9.0\%$ perplexity and $0.94$–$2.25$ accuracy points, and ordering alone accounts for up to $2.5\%$ perplexity and $1.49$ points over contiguous grouping (Table 2). In simulated-quantization prefill with BF16 matrix multiplication, fusion reduces the permutation overhead from 29.53 ms to 2.34 ms ($+4.5\%$) (Table 6).

## 2 Related Work

Activation quantization in LLMs is hard because a few channels carry values orders of magnitude larger than the rest, and under a uniform integer grid, their range sets the step size for every channel that shares a scale with them (Dettmers et al., 2022; Xiao et al., 2024).

**Reshaping the distribution.** These works change the basis used to define activations so that no single channel dominates. SmoothQuant (Xiao et al., 2024) migrates quantization difficulty from activations to weights through a per-channel scaling that leaves the layer output unchanged, and AWQ (Lin et al., 2026) applies the same idea in the weight-only setting to protect salient weights. Rotation methods go further. QuaRot (Ashkboos et al., 2024b) multiplies hidden states, feed-forward activations, attention components, and the KV cache by Hadamard matrices, which spreads outlier mass across all channels and enables end-to-end 4-bit quantization without any channel held in higher precision. OSTQuant (Hu et al., 2025) replaces the fixed Hadamard with a learned orthogonal transformation optimized on the Stiefel manifold, composed with a learned diagonal scaling. SpinQuant (Liu et al., 2025) learns the rotations end to end with a quantization-aware objective, and FlatQuant (Sun et al., 2024) learns per-layer affine transformations that flatten both weights and activations. All of them address activation outliers by choosing the basis, while assigning transformed channels to fixed-size groups remains a further design choice. An orthogonal map preserves $\mathrm{tr}(\mathbf{\Sigma})$ and $\|\mathbf{\Sigma}\|_F$, so it can redistribute covariance across channels without eliminating their joint structure. CAPQuant uses that structure to select a grouping after the base transformation.

**Removing the outliers.** Rather than reshape the distribution, LLM.int8() (Dettmers et al., 2022) isolates the few feature dimensions that carry outliers and computes them in 16-bit while quantizing the remainder with vector-wise 8-bit scales. QUIK (Ashkboos et al., 2024a) extends this to joint 4-bit weight-and-activation quantization, keeping a small set of outlier weights and activations in higher precision. Extraction determines which channels enter the low-bit computation, and grouping determines which of those channels share a scale.

**Choosing who shares a scale.** The closest work to ours accepts the basis and the channel set as given and asks only which channels should be quantized together. Atom (Zhao et al., 2024) reorders channels so that the outliers form one group kept at INT8 and quantizes the remaining $g$=128 groups to INT4 with dedicated kernels. Its reordering serves extraction, and the remaining channels are ordered by the same per-channel magnitude statistic rather than by an objective on their shared scales. RPTQ (Yuan et al., 2023) shares our pipeline of reordering and then grouping contiguously, but represents each channel by its calibration (min, max) and clusters them using $k$-means. These summaries describe each channel's marginal range but do not capture how channels vary together across tokens. Under dynamic groupwise quantization, channels with similar marginal ranges can still produce different group ranges when their extrema occur at different tokens. The pairwise dispersion $\mathbb{E}_t[(y_{ti} - y_{tj})^2] = (m_i - m_j)^2 + \sigma_i^2 + \sigma_j^2 - 2C_{ij}$ captures this joint behavior through covariance. DuQuant (Lin et al., 2024) interleaves a block-wise rotation built from the outlier dimensions with a zigzag permutation that spreads those dimensions evenly across blocks. PermuQuant (Cheng et al., 2026) sorts the channels of diffusion transformers by a joint second moment of activations and weights, without training, under per-group symmetric scales. These approaches establish channel order as a useful quantization variable. Our focus is on a balanced grouping objective derived from pairwise activation dispersion.

CAPQuant differs on three counts. It works from pairs of channels rather than per-channel summaries, so it captures cross-channel covariance. It selects the order after the base transformation and can therefore build on an existing quantized model without retraining it. It adds no separate inference pass when the permutation can be folded into consumers and fused into activation kernels. We evaluate this integration with OSTQuant, QuaRot, and SpinQuant.

## 3 Quantization Setup

We consider the transformed activations $\mathbf{y}_t \in \mathbb{R}^d$ at a quantization site. Each token row is divided into $K = d/g$ contiguous groups of size $g$, each with a dynamic asymmetric scale and zero-point. A channel permutation $\mathcal{P}$ determines which channels share a scale. For a $b$-bit quantizer, the group range and step size are

$$R_{tk} = \max_{i \in G_k} y_{ti} - \min_{i \in G_k} y_{ti}, \qquad s_{tk} = \frac{R_{tk}}{2^b - 1}. \tag{1}$$

Under the standard uniform-noise model, the expected reconstruction error is proportional to the range objective

$$J_R(\mathcal{P}) = \sum_k \mathbb{E}_t\left[R_{tk}(\mathcal{P})^2\right]. \tag{2}$$

Appendix B.1 gives the quantizer equations, and Appendix B.2 accounts for the zero-inclusive range used in our implementation.

CAPQuant selects groups after the base transformation. In OSTQuant, it replaces the single per-row scale at the two normalization outputs and the MLP SwiGLU product with permuted $g = 128$ groups, leaving already-grouped sites unchanged. A group of 128 channels carries one 16-bit scale and one 4-bit zero-point, so grouping adds $0.16$ bits per activation, the same metadata budget as the $g = 128$ groups OSTQuant already uses at its other sites, and the permutation adds no metadata. Let $\mathbf{\Sigma}$ denote the transformed activation covariance, with diagonal entries $\sigma_i^2$ and off-diagonal entries $C_{ij}$. Rotation can redistribute this covariance without eliminating it. Appendix B.6 gives the OSTQuant details.

## 4 Method

CAPQuant chooses a channel ordering at each ungrouped site in OSTQuant. We formalize the grouping objective, then describe its optimization and integration into the inference process.

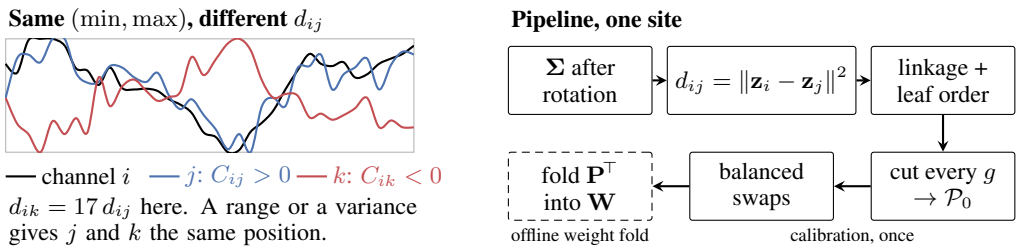


Figure 1: **Left:** three channels with identical $(\min, \max)$ over the tokens shown. RPTQ's descriptor and any variance-based sort cannot tell $j$ from $k$, while $d_{ij} = (m_i - m_j)^2 + \sigma_i^2 + \sigma_j^2 - 2C_{ij}$ separates them by $4|C|$. **Right:** the CAPQuant pipeline. The inverse permutation is folded into weights offline. At inference, the forward permutation is applied through indexed activation reads.

### 4.1 Objective

Let $m_i = \mathbb{E}_t[y_{ti}]$ and $C_{ij} = \mathrm{Cov}_t(y_{ti}, y_{tj})$ denote the activation means and covariances over calibration tokens $t$. We seek a partition $\mathcal{P}$ of the $d$ channels into groups of size $g$ that minimizes

$$\arg\min_{\mathcal{P}} \sum_{G \in \mathcal{P}} \sum_{\substack{i<j \\ i,j \in G}} \left[ (m_i - m_j)^2 - 2C_{ij} \right]. \tag{3}$$

We derive this objective by isolating the variance term, which is constant across balanced partitions.

**From range to dispersion.** The range objective $J_R$ in Equation (2) depends only on the two extreme activations of each group. We replace it with a dispersion that counts every channel. For a group $G_k$ at token $t$ with mean $\bar{y}_{tk}$, an exact bias–variance identity relates the within-group dispersion to the pairwise squared differences,

$$V_{tk} = \sum_{i \in G_k} (y_{ti} - \bar{y}_{tk})^2 = \frac{1}{g} \sum_{\substack{i<j \\ i,j \in G_k}} (y_{ti} - y_{tj})^2. \tag{4}$$

**Proposition 1** (Range–dispersion sandwich)**.** *For every group and token, $\frac{4}{g} V_{tk} \le R_{tk}^2 \le 2V_{tk}$. Summing over groups and averaging over tokens, the dispersion objective $\Psi(\mathcal{P}) = \sum_k \mathbb{E}_t[V_{tk}(\mathcal{P})]$ satisfies*

$$\frac{4}{g}\, \Psi(\mathcal{P}) \;\le\; J_R(\mathcal{P}) \;\le\; 2\, \Psi(\mathcal{P}). \tag{5}$$

The proof is in Section A.2. Both constants are independent of the partition, so reducing $\Psi$ lowers an upper bound on the range objective. It need not reduce the realized range or quantization error, but empirically it tracks both and model quality: across eight orderings of a frozen Llama-3-8B checkpoint, $\Psi$ and WikiText-2 perplexity have a Spearman correlation of $0.83$, and the realized group range and the quantizer's reconstruction error rank the orderings identically (Section A.11). For the zero-inclusive quantizer, the augmented dispersion ranks balanced partitions in the same way (Section B.2).

**From dispersion to pairs.** Substituting Equation (4) into $\Psi$ and exchanging the sum with the expectation,

$$\Psi(\mathcal{P}) = \frac{1}{g} \sum_k \sum_{\substack{i<j \\ i,j \in G_k(\mathcal{P})}} d_{ij}, \qquad d_{ij} := \mathbb{E}_t[(y_{ti} - y_{tj})^2] = \sigma_i^2 + \sigma_j^2 + D_{ij}, \tag{6}$$

with $\sigma_i^2 = \mathrm{Var}_t(y_{ti})$ and $D_{ij} := (m_i - m_j)^2 - 2C_{ij}$. The partition decides only which pairs are counted. The quantity counted, $d_{ij}$, is fixed by the calibration data.

**Proposition 2** (The variance term is partition-free)**.** *Each channel appears in exactly $g-1$ within-group pairs, so for every balanced partition*

$$\Psi(\mathcal{P}) = \frac{g-1}{g}\mathrm{tr}(\mathbf{\Sigma}) + \frac{1}{g}\sum_{G\in\mathcal{P}}\sum_{\substack{i<j\\ i,j\in G}} D_{ij}, \tag{7}$$

*and minimizing $\Psi$ is equivalent to Equation (3).*

The proof is in Section A.5. Balance is what makes the trace term constant. For variable-size groups, the variance contribution to the total within-group dispersion is $\sum_G(1-1/|G|)\sum_{i\in G}\sigma_i^2$, which generally depends on the partition. Without covariance, the objective of Proposition 2 also has a **closed-form minimizer**.

**Corollary 1** (Uncorrelated channels)**.** *If $C_{ij}=0$ for all $i\neq j$, sorting the channels by $m_i$ and cutting the sorted order every $g$ channels minimizes $\Psi$ over all balanced partitions.*

The proof in Section A.6 is an inverse argument for the fact that if channels are uncorrelated, then no per-channel sort keys improve on mean sorting, and the effect of covariance on a channel's marginal distribution remains invisible. The mean-sorted order is therefore the covariance-free baseline of Section 5.

**Why rotation creates the opportunity.** Rotation moves covariance rather than removing it. An orthogonal transform leaves $\mathrm{tr}(\mathbf{\Sigma})$ and $\|\mathbf{\Sigma}\|_F$ unchanged, and Equation (43) splits the latter into a diagonal and an off-diagonal part, $\|\mathbf{\Sigma}\|_F^2 = \sum_i \sigma_i^4 + 2\sum_{i<j} C_{ij}^2$. The diagonal part is smallest when all variances are equal, since $\sum_i \sigma_i^4 \geq \mathrm{tr}(\mathbf{\Sigma})^2/d$, so a rotation that flattens the variances, which is what outlier suppression aims for, pushes the covariance mass toward its ceiling $\|\mathbf{\Sigma}\|_F^2 - \mathrm{tr}(\mathbf{\Sigma})^2/d$ (Section A.9). OSTQuant's diagonal scaling breaks the exact conservation, and a large covariance mass does not say how the signed $C_{ij}$ and the mean differences in $D_{ij}$ weigh against each other, which Section 5 measures. Still, rotated activations sit far from the uncorrelated regime of Corollary 1, where mean sorting would already be optimal.

### 4.2 Channel geometry and dimensionality reduction

For $T$ calibration tokens, represent each channel by its mean and its centered trace,

$$\mathbf{u}_i[t] = \frac{y_{ti}-m_i}{\sqrt{T}}, \qquad \mathbf{z}_i = (m_i, \mathbf{u}_i), \qquad \|\mathbf{z}_i - \mathbf{z}_j\|^2 = d_{ij}. \tag{8}$$

Equation (3) is therefore balanced $k$-means on the points $\mathbf{z}_i$, and the points come directly from the centered activations without forming $\mathbf{\Sigma}$.

We use reduced coordinates for refinement. To reduce its computational cost, we project onto the leading $r$ eigenvectors $\mathbf{V}_r$ of the token Gram matrix $\mathbf{G} = \mathbf{Y}_c\mathbf{Y}_c^\top/T$, where $\mathbf{Y}_c = \mathbf{Y} - \mathbf{1}\mathbf{m}^\top$. This costs $O(T^2 d)$ rather than $O(d^2 T)$, the cheaper direction at the widest sites, where $d > T$. We target $90\%$ of the energy subject to $r \leq 128$. The reduced points and objective are

$$\widetilde{\mathbf{u}}_i = \mathbf{V}_r^\top \mathbf{u}_i, \qquad \widetilde{\mathbf{z}}_i = (m_i, \widetilde{\mathbf{u}}_i), \qquad \Psi_r(\mathcal{P}) = \sum_{G\in\mathcal{P}}\sum_{i\in G} \left\|\widetilde{\mathbf{z}}_i - \bar{\widetilde{\mathbf{z}}}_G\right\|^2. \tag{9}$$

Mean and covariance coordinates enter with equal weight. Equation (6) fixes their relative scale, so any reweighting $\alpha \neq 1$ optimizes a different distance from the dispersion surrogate. Section B.4 gives the rank rule, the projection bound $\Psi = \Psi_r + \Psi_\perp$, and the $\alpha$ ablation.

### 4.3 Algorithm

**Initialization.** CAPQuant accepts any starting partition $\mathcal{P}_0$ with exactly $g$ channels per group. We build $\mathcal{P}_0$ by capacity-constrained hierarchical two-means on the reduced channel points. Each node is split into two children of prescribed sizes until every leaf holds exactly $g$ channels, so the start already uses the full channel geometry of Section 4.2. The mean-sorted order of Corollary 1 is the covariance-free control in Section 5.

**Algorithm 1** CAPQuant permutation for one activation site.

**Require:** Calibration activations $\mathbf{Y} \in \mathbb{R}^{T \times d}$, group size $g$, site type $s$, balanced initializer $\mathcal{I}$.
1: $\mathcal{P} \leftarrow \mathcal{I}(\mathbf{Y}, g)$, $\pi_0 \leftarrow$ concatenation of the groups of $\mathcal{P}$.
2: **if** $s =$ `input_layernorm` **then return** $\pi_0$
3: **end if**
4: Project centered traces onto $\mathbf{V}_r$ and form $\widetilde{\mathbf{z}}_i = (m_i, \widetilde{\mathbf{u}}_i)$.
5: **repeat**
6: For every group pair, take the swap maximizing $\Delta F_{ij}$ in Equation (10).
7: Accept improving candidates in descending order of gain, each group at most once, and apply them.
8: **until** no improving candidate remains
9: **return** the concatenation of the groups of $\mathcal{P}$.

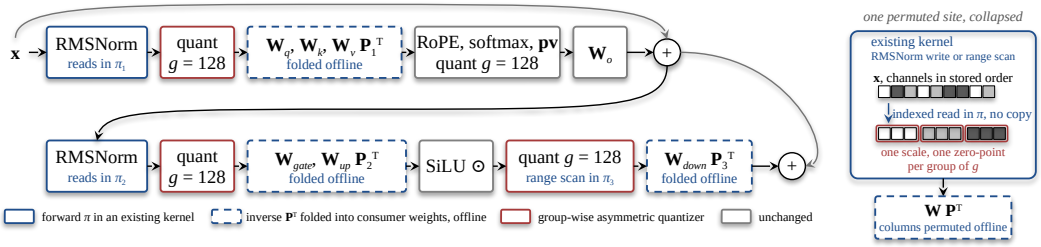


Figure 2: **CAPQuant inside one decoder block.** The two RMSNorm outputs and the MLP product receive permuted $g$=128 groups. Each forward permutation is carried by a kernel that already reads every channel, the RMSNorm write for $\pi_1, \pi_2$ and the quantizer's range scan for $\pi_3$ (Section A.10), and its inverse is folded into every consumer, so $\mathbf{W}\mathbf{P}^\top\mathbf{P}\mathbf{x} = \mathbf{W}\mathbf{x}$. Nothing is copied and no matrix product is added. Only the assignment of channels to scales changes. **Right:** one permuted site, collapsed. The existing kernel reads the channels in permuted order, the quantizer assigns one scale and zero-point to each group of $g$ consecutive channels in that order, and the consumer's weight columns are permuted once, offline.

**Balanced refinement.** For group sums $\mathbf{s}_G = \sum_{i \in G} \widetilde{\mathbf{z}}_i$, the reduced objective is $\Psi_r = \sum_i \|\widetilde{\mathbf{z}}_i\|^2 - \frac{1}{g}\sum_G \|\mathbf{s}_G\|^2$, so exchanging $i \in A$ and $j \in B$ with $\mathbf{h} = \widetilde{\mathbf{z}}_j - \widetilde{\mathbf{z}}_i$ changes it by

$$\Delta\Psi_r = -\frac{\Delta F_{ij}}{g}, \qquad \Delta F_{ij} = 2\,\langle \mathbf{s}_A - \mathbf{s}_B,\, \mathbf{h}\rangle + 2\,\|\mathbf{h}\|^2 . \tag{10}$$

Each round scores every cross-group swap in $O(K^2 g^2 r)$, keeps the best candidate per group pair, and accepts candidates in descending gain with each group used at most once (Algorithm 1). Swaps preserve group sizes, and because no group participates in two accepted swaps per round, their gains add up.

**Proposition 3** (Monotone descent)**.** *Every accepted swap strictly decreases $\Psi_r$, so $\Psi_r(\mathcal{P}_{\text{final}}) \le \Psi_r(\mathcal{P}_0)$. In exact arithmetic, exhaustive improving-swap search over the finite set of balanced partitions terminates at a single-swap local optimum. In the implementation, this conclusion holds up to the acceptance tolerance when the search converges before its safety limit.*

The guarantee is on the projected dispersion. Quantization error and model quality are measured separately in Section 5. The stopping rule is in Section B.5.

**Site selection.** We keep $\mathcal{P}_0$ at attention-input normalization sites (`input_layernorm`) and refine it at pre-MLP normalization and MLP-product sites (`post_attention_layernorm` and `mlp.mul`). This rule is fixed across all experiments. All variants leave the base transforms and quantized weights fixed.

Table 1: **CAPQuant as a plug-in for Llama-3-8B.** Highlighted rows show the effect of adding CAPQuant to each base method.

| | W4A4KV16 | | | | W4A4KV4 | | | |
|---|---|---|---|---|---|---|---|---|
| Method | Wiki↓ | Red.↑ | C4↓ | Red.↑ | Wiki↓ | Red.↑ | C4↓ | Red.↑ |
| OSTQuant | 7.248 | – | 11.880 | – | 7.384 | – | 12.229 | – |
| OSTQuant + contiguous | 6.956 | 4.0% | 11.299 | 4.9% | 7.104 | 3.8% | 11.722 | 4.1% |
| **OSTQuant + CAPQuant** | **6.901** | 4.8% | **11.231** | 5.5% | **7.083** | 4.1% | **11.667** | 4.6% |
| QuaRot | 7.895 | – | 12.937 | – | 8.068 | – | 13.229 | – |
| QuaRot + contiguous | 7.157 | 9.4% | 11.650 | 9.9% | 7.293 | 9.6% | 11.898 | 10.1% |
| **QuaRot + CAPQuant** | **7.080** | 10.3% | **11.561** | 10.6% | **7.222** | 10.5% | **11.817** | 10.7% |
| SpinQuant | 7.287 | – | 12.098 | – | 7.358 | – | 12.215 | – |
| SpinQuant + contiguous | 7.148 | 1.9% | 11.639 | 3.8% | 7.230 | 1.7% | 11.782 | 3.5% |
| **SpinQuant + CAPQuant** | **7.110** | 2.4% | **11.616** | 4.0% | **7.183** | 2.4% | **11.710** | 4.1% |

"Red." denotes perplexity reduction relative to each base row.

### 4.4 Placement and inference cost

CAPQuant is applied after the rotation and scaling transforms (Figure 2). Each site's permutation is computed once at calibration and held fixed. For an activation vector $\mathbf{x}$ and permutation matrix $\mathbf{P}$, $\mathbf{W}\mathbf{P}^\top\mathbf{P}\mathbf{x} = \mathbf{W}\mathbf{x}$, so reordering the input columns of every consuming linear layer absorbs the inverse permutation. With quantized weights, the fold must preserve their decoded values and associated scale metadata. Only the grouping, and hence the quantized values, change. The forward permutation is fused into kernels that already read every channel. The RMS statistic is invariant to channel order, so at the 72 RMSNorm sites of Qwen3-4B, the fused kernel reads activations and normalization weights in permuted order and writes the result directly (Section A.10). At the 36 MLP-product sites, the quantizer reads channels in permuted order as it computes group ranges. Neither kernel adds an activation-copy pass. Indexed access still incurs a cost measured by Section 5.

## 5 Experiments

We evaluate W4A4KV16 and W4A4KV4 with $g = 128$, frozen evaluation inputs, and fixed quantized weights within each comparison. C4 uses GPTQ's first 1,100-document protocol. Accuracy is the macro-average over nine zero-shot tasks. Grouping uses two 2,048-token calibration sequences. Experimental settings have been provided in more detail under Section C

### 5.1 Improving existing quantization methods

Table 1 shows that CAPQuant lowers perplexity for OSTQuant, QuaRot, and SpinQuant on both corpora and in both precision settings. Perplexity reductions reach $10.7\%$ for QuaRot and $4.1\%$ for SpinQuant. On the released OSTQuant checkpoint under W4A4KV16, WikiText-2 and C4 perplexities fall by $4.8\%$ and $5.5\%$, respectively. Zero-shot accuracy reaches $67.00\%$, a gain of $1.33$ percentage points (Table 2). These gains require no retraining or transformations. The contiguous rows separate the benefit of additional scales from the contribution of channel ordering. The largest perplexity gain occurs with Llama-3.2-1B under the W4A4KV4 transform and the largest accuracy gain with Qwen3-4B under W4A4KV16.

### 5.2 The contribution of channel ordering

Table 2 introduces contiguous grouping into fixed OSTQuant checkpoints before comparing alternative assignments. Across all four models and both precision settings, the highlighted CAPQuant variant improves every perplexity and accuracy metric over OSTQuant. Perplexity falls by $1.5$–$9.0\%$, and zero-shot accuracy rises by $0.94$–$2.25$ percentage points. The largest perplexity gain occurs with Llama-3.2-1B under W4A4KV4, where the KV cache is also quantized to 4 bits, and the largest accuracy gain with Qwen3-4B under W4A4KV16.

Table 2: **Channel-ordering analysis.** Grouping controls and CAPQuant across models under one configuration.

| Method / configuration | Qwen3-4B | | | Llama-3.2-1B | | | Llama-3-8B | | | Llama-2-7B | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Acc.↑ | Wiki↓ | C4↓ | Acc.↑ | Wiki↓ | C4↓ | Acc.↑ | Wiki↓ | C4↓ | Acc.↑ | Wiki↓ | C4↓ |
| *W4A4KV16* | | | | | | | | | | | | |
| OSTQuant | 57.89 | 14.720 | 22.547 | 49.47 | 13.590 | 21.452 | 65.67 | 7.248 | 11.880 | 62.63 | 5.823 | 8.058 |
| + Contiguous grouping | 59.29 | 14.285 | 21.923 | 50.10 | 12.668 | 19.932 | 66.64 | 6.956 | 11.299 | 63.86 | 5.724 | **7.920** |
| + RPTQ-style order | 59.57 | **14.108** | 21.751 | 50.95 | 12.716 | 19.992 | 66.73 | 6.933 | 11.271 | **64.00** | 5.726 | 7.942 |
| + Random order | 58.93 | 14.358 | 21.907 | **51.85** | 12.718 | 20.053 | 66.92 | 6.959 | 11.290 | 63.64 | 5.728 | 7.928 |
| + Mean-sorted order | 58.97 | 14.182 | 21.735 | 50.08 | 12.677 | 19.957 | 66.66 | 6.926 | 11.311 | 63.46 | 5.716 | 7.938 |
| + **CAPQuant** | **60.14** | 14.209 | **21.683** | 51.59 | **12.543** | **19.767** | **67.00** | **6.901** | **11.231** | 63.57 | **5.715** | 7.941 |
| *W4A4KV4* | | | | | | | | | | | | |
| OSTQuant | 57.40 | 14.926 | 22.957 | 47.44 | 14.861 | 23.300 | 65.10 | 7.384 | 12.229 | 62.44 | 5.921 | 8.113 |
| + Contiguous grouping | 57.89 | 14.462 | 22.218 | 48.97 | 13.848 | 21.648 | 65.84 | 7.104 | 11.722 | 63.03 | 5.809 | 7.908 |
| + **CAPQuant** | **58.69** | **14.101** | **21.699** | **49.47** | **13.560** | **21.205** | **66.54** | **7.083** | **11.667** | **63.43** | **5.795** | **7.903** |

Accuracy is reported in percent. Best values are bold, and second-best values are underlined within each model and precision setting.

Channel ordering contributes beyond the additional scales. Relative to contiguous grouping, CAPQuant lowers perplexity in 15 of 16 comparisons, by up to $2.5\%$, and improves accuracy in 7 of 8 comparisons, by up to $1.49$ percentage points. The mean-sorted row is the exact minimizer of the objective without its covariance term (Corollary 1), so the comparison against it isolates that term: under W4A4KV16, CAPQuant lowers perplexity in six of eight comparisons against mean sorting and in seven of eight against the RPTQ-style ordering, with reductions reaching $1.1\%$ and $1.4\%$, respectively. Over mean sorting, the covariance term helps by $1.0$–$1.1\%$ on Llama-3.2-1B and $0.4$–$0.7\%$ on Llama-3-8B, and is within $0.25\%$ on Qwen3-4B and $0.05\%$ on Llama-2-7B. These comparisons support using joint channel behavior to select groups.

**Group size.** Table 4 repeats the comparison at $g \in \{32, 64\}$ under W4A4KV16. Ordering improves on contiguous grouping in all 12 entries. On Qwen3-4B, its gain grows as groups shrink, from $0.5\%$ of WikiText-2 perplexity at $g = 128$ to $2.6\%$ at $g = 32$, the MXFP4 group size, while on Llama-3-8B, it stays between $0.6\%$ and $0.8\%$ at every $g$.

The benefit still depends on the model and metric. Taken together, CAPQuant attains the lowest perplexity in 14 of the 16 entries of Table 2 and the highest accuracy in 6 of 8. Accuracy differences among orderings for the 7B and 8B models are within 1 percentage point and should be read alongside the perplexity trend, which is consistent across models.

On MGSM, a multilingual generation task, grouping recovers 4.7–7.5 points of the accuracy Qwen3-4B loses under W4A4, and ordering adds 3.1 points under W4A4KV4 and 0.8 under W4A4KV16, where only the former is significant (Section E).

### 5.3 Extending CAPQuant to larger models

Table 3 extends the comparison to three model sizes. CAPQuant attains the lowest perplexity in eleven of the twelve model–precision–corpus cells, improving over OSTQuant at every size, with reductions of $1.46$–$2.89\%$, and over contiguous grouping in all cells except Llama-2-7B C4 under W4A4KV16. The same decomposition persists across sizes: Channel ordering improves eleven of twelve configurations. Llama-2-7B and Llama-2-13B are from the same generation, so the comparison between them reflects model size within the same family. Construction time and peak GPU allocation increase with model size (Table 5).

### 5.4 Inference cost of the quality gains

Our timing experiments measure prefill with simulated activation quantization and BF16 matrix multiplication. Folding and fusion remove $92\%$ of the explicit gather–scatter overhead on Qwen3-4B (Table 6), based on a single-layer measurement scaled to 36 layers. Permuting only the 72 RMSNorm

Table 3: **CAPQuant across model sizes.** Perplexity before and after channel grouping.

| | W4A4KV16 | | W4A4KV4 | |
|---|---|---|---|---|
| Model / configuration | Wiki↓ | C4↓ | Wiki↓ | C4↓ |
| *Llama-2-7B* | | | | |
| OSTQuant | 5.823 | 8.058 | 5.921 | 8.113 |
| + Contiguous grouping | 5.724 | **7.920** | 5.809 | 7.908 |
| + **CAPQuant** | **5.715** | 7.941 | **5.795** | **7.903** |
| Reduction vs. OSTQuant | 1.87% | 1.46% | 2.12% | 2.59% |
| *Llama-2-13B* | | | | |
| OSTQuant | 5.260 | 7.377 | 5.261 | 7.380 |
| + Contiguous grouping | 5.145 | 7.195 | 5.179 | 7.248 |
| + **CAPQuant** | **5.127** | **7.165** | **5.161** | **7.225** |
| Reduction vs. OSTQuant | 2.53% | 2.89% | 1.90% | 2.10% |
| *Llama-30B* | | | | |
| OSTQuant | 4.463 | 6.608 | 4.479 | 6.551 |
| + Contiguous grouping | 4.381 | 6.514 | 4.366 | 6.421 |
| + **CAPQuant** | **4.367** | **6.500** | **4.352** | **6.407** |
| Reduction vs. OSTQuant | 2.15% | 1.63% | 2.84% | 2.19% |

Bold marks the lowest perplexity within each setting.

Table 4: **Group size.** WikiText-2 and C4 perplexity under W4A4KV16 with contiguous groups and with CAPQuant.

| | Wiki↓ | | C4↓ | |
|---|---|---|---|---|
| $g$ | Contig. | CAPQuant | Contig. | CAPQuant |
| *Qwen3-4B* | | | | |
| 32 | 14.153 | **13.788** | 21.711 | **21.210** |
| 64 | 14.236 | **14.002** | 21.730 | **21.385** |
| 128 | 14.285 | **14.209** | 21.923 | **21.683** |
| *Llama-3-8B* | | | | |
| 32 | 6.861 | **6.822** | 11.140 | **11.056** |
| 64 | 6.904 | **6.862** | 11.214 | **11.147** |
| 128 | 6.956 | **6.901** | 11.299 | **11.231** |

Bold marks the lower perplexity of each pair.

sites costs 0.01 ms, so almost all of the remaining 2.34 ms sits at the 36 MLP-product sites, where the online Hadamard forces the permutation into the quantizer's range scan.

With the same fused quantizer for contiguous grouping and CAPQuant, the added latency is $3.58\%$ on Llama-2-7B and $6.88\%$ on Llama-30B (Table 5). These measurements isolate permutation overhead in the reference implementation. They do not establish native INT4 speedups, which we did not have the hardware to test. Because the matrix multiplications run in BF16, the same absolute overhead would be a larger fraction of a faster INT4 forward pass.

## 6 Conclusion

CAPQuant finds a better channel grouping to achieve higher performance across multiple quantization configurations and methods. Rotation-based quantizers only fix the basis and leave out the free gains from channel ordering on the table. We derive an exact formulation for the asymmetric groupwise quantization error and show that it depends on the mean differences and covariances of the channels, while the variance terms remain constant across balanced groups. We further work out a clean objective and a closed-form baseline for our grouping objective. CAPQuant performs balanced swaps offline at calibration time and folds the results into the model's existing weights. On top of OSTQuant, CAPQuant lowers perplexity by $1.5$–$9.0\%$ and raises zero-shot accuracy by $0.9$–$2.3$ points across four models of up to 30B parameters and two precisions. Our gains carry over to frozen initializations from QuaRot and SpinQuant with perplexity reductions of up to $10.7\%$. Multilingual generation evaluated on MGSM also receives a $3.1$ percentage-point accuracy boost in the W4A4KV4 setting. The permutation costs $0.01$ ms at the normalization sites and $3.6$–$6.9\%$ of prefill in our reference implementation, with no retraining and no extra pass. Thus, a quantized model that leaves its channels in arbitrary order is leaving performance on the table.

In the future, we want to expand this work to native INT4 kernels and evaluate larger models across different architectures, such as a mixture of experts (MoEs). There is also potential to explore exciting directions combining rotational transformations and channel ordering to achieve higher performance.

### AI use statement

In this work, we used generative AI tools to refine the captions of our figures and tables, to fix the formatting of our tables, check the math of the appendix derivations, and proofread the text and experiments in the paper. We also used AI-assisted coding to refine the permutation kernels in Triton and to automate some experiments along the way for reproducibility. We did not use generative AI tools for proposing the method. Synthetic data generation and qualitative data analysis are not

components of our work. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

### Reproducibility statement

All quantized checkpoints, calibration data, and evaluation tensors are frozen and described in Section C, which also states the model revisions, calibration seeds, and timing protocol. Complete derivations of every claim in Section 4 are given in Section A, and implementation details of the quantizer, initializers, projection, and stopping rule in Section B. Anonymous source code and the saved permutations are provided as supplementary material.

## References


Saleh Ashkboos, Ilia Markov, Elias Frantar, Tingxuan Zhong, Xincheng Wang, Jie Ren, Torsten Hoefler, and Dan Alistarh. Quik: Towards end-to-end 4-bit inference on generative large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3355–3371, 2024a.

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

Stella Biderman, Hailey Schoelkopf, Lintang Sutawika, Leo Gao, Jonathan Tow, Baber Abbasi, Alham Fikri Aji, Pawan Sasanka Ammanamanchi, Sidney Black, Jordan Clive, et al. Lessons from the trenches on reproducible evaluation of language models. *arXiv preprint arXiv:2405.14782*, 2024.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pp. 7432–7439, 2020.

Yongsen Cheng, Kai Liu, Kaiwen Tao, Junxian Li, Zhixin Wang, Zhikai Chen, Renjing Pei, and Yulun Zhang. Permuquant: Lowering per-group quantization error by reordering channels for diffusion models. *arXiv preprint arXiv:2605.09503*, 2026.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2924–2936, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35: 30318–30332, 2022.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

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

Xing Hu, Yuan Cheng, Dawei Yang, Zukang Xu, Zhihang Yuan, Jiangyong Yu, Chen Xu, Zhe Jiang, and Sifan Zhou. Ostquant: Refining large language model quantization with orthogonal and scaling transformations for better distribution fitting. *arXiv preprint arXiv:2501.13987*, 2025.

Haokun Lin, Haobo Xu, Yichen Wu, Jingzhi Cui, Yingtao Zhang, Linzhan Mou, Linqi Song, Zhenan Sun, and Ying Wei. Duquant: Distributing outliers via dual transformation makes stronger quantized llms. *Advances in Neural Information Processing Systems*, 37:87766–87800, 2024.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration, 2026. URL https://arxiv.org/abs/2306.00978.

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

Kelly Marchisio, Saurabh Dash, Hongyu Chen, Dennis Aumiller, Ahmet Üstün, Sara Hooker, and Sebastian Ruder. How does quantization affect multilingual llms? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 15928–15947, 2024.

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

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2381–2391, 2018.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc-Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 1525–1534, 2016.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Bita Darvish Rouhani, Ritchie Zhao, Ankit More, Mathew Hall, Alireza Khodamoradi, Summer Deng, Dhruv Choudhary, Marius Cornea, Eric Dellinger, Kristof Denolf, et al. Microscaling data formats for deep learning. *arXiv preprint arXiv:2310.10537*, 2023.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 4463–4473, 2019.

Freda Shi, Mirac Suzgun, Markus Freitag, Xuezhi Wang, Suraj Srivats, Soroush Vosoughi, Hyung Won Chung, Yi Tay, Sebastian Ruder, Denny Zhou, et al. Language models are multilingual chain-of-thought reasoners. *arXiv preprint arXiv:2210.03057*, 2022.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models, 2024. URL https://arxiv.org/abs/2211.10438.

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

Zhihang Yuan, Lin Niu, Jiawei Liu, Wenyu Liu, Xinggang Wang, Yuzhang Shang, Guangyu Sun, Qiang Wu, Jiaxiang Wu, and Bingzhe Wu. Rptq: Reorder-based post-training quantization for large language models. *arXiv preprint arXiv:2304.01089*, 2023.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. Atom: Low-bit quantization for efficient and accurate llm serving. *Proceedings of Machine Learning and Systems*, 6:196–209, 2024.

# A Complete derivations

This appendix proves every step of the algorithm across Sections A.2–A.5 establish the chain

$$J_R \xrightarrow[\text{Lemma 1}]{\text{sandwich}} \Psi \xrightarrow[\text{Lemma 2}]{\text{pairwise identity}} \tfrac{1}{g}\sum d_{ij} \xrightarrow[\text{Lemma 3}]{\text{decomposition}} \sigma_i^2+\sigma_j^2+D_{ij} \xrightarrow[\text{Theorem 1}]{\text{tr is partition-free}} \sum_G \sum_{i<j} D_{ij}.$$

Section A.6 proves that mean sorting is optimal for uncorrelated channels (Corollary 1). Section A.7 gives the embedding that realises $d_{ij}$ as a Euclidean distance, A.8 the token-space reduction, A.9 the rotation-invariance argument, and A.10 the two facts about permutation matrices that make the deployment exact. Section A.11 checks empirically that the surrogate $\Psi$ tracks perplexity across orderings.

## A.1 Setup and notation

Fix one activation site. Let $\mathbf{Y} \in \mathbb{R}^{T\times d}$ collect the site's activations over $T$ calibration tokens and $d$ channels, with entries $y_{ti}$. Write

$$m_i = \mathbb{E}_t[y_{ti}], \qquad \sigma_i^2 = \mathrm{Var}_t(y_{ti}), \qquad C_{ij} = \mathrm{Cov}_t(y_{ti}, y_{tj}), \tag{11}$$

and let $\mathbf{\Sigma} \in \mathbb{R}^{d\times d}$ be the channel covariance, $\mathbf{\Sigma}_{ij} = C_{ij}$, so $\mathbf{\Sigma}_{ii} = \sigma_i^2$ and $\mathrm{tr}(\mathbf{\Sigma}) = \sum_i \sigma_i^2$. All expectations $\mathbb{E}_t$ are over the empirical token distribution.

A *partition* $\mathcal{P} = (G_1, \ldots, G_K)$ cuts the $d$ channels into $K = d/g$ groups of exactly $g$. Because groups are cut contiguously from the channel ordering, $\mathcal{P}$ is determined by a permutation $\pi \in S_d$, and we use the two interchangeably.

## A.2 From range to dispersion

Under the standard high-resolution uniform-noise model, a $b$-bit asymmetric quantizer applied to group $G_k$ with range $R_k = \max_{i\in G_k} y_{ti} - \min_{i\in G_k} y_{ti}$ has step $s_k = R_k/(2^b - 1)$ and per-element error variance $s_k^2/12$. Summing over groups and taking expectation over tokens gives the range objective

$$J_R(\mathcal{P}) \;=\; \sum_k \mathbb{E}_t\big[R_k(\mathcal{P})^2\big]\,, \tag{12}$$

which is what we wish to minimise over $\mathcal{P}$. The obstruction is that $R_k$ is an extreme-value statistic: it is not a smooth function of the group membership and admits no useful algebraic expansion. We replace it by a dispersion quantity that does.

For a group $G$ at a single token, write $\bar{y} = \frac{1}{g}\sum_{i\in G} y_{ti}$ and

$$V \;=\; \sum_{i\in G} (y_{ti} - \bar{y})^2. \tag{13}$$

**Lemma 1** (Range–dispersion sandwich). *For any group $G$ of size $g \geq 2$ at any token,*

$$\frac{4}{g} V \;\leq\; R^2 \;\leq\; 2\,V. \tag{14}$$

*Proof.* Let $a = \min_{i \in G} y_{ti}$, $b = \max_{i \in G} y_{ti}$, and $R = b - a$. For the upper bound, the two extreme observations give

$$V \geq (a - \bar{y})^2 + (b - \bar{y})^2 \geq \frac{(b-a)^2}{2},$$

so $R^2 \leq 2V$. For the lower bound, set $c = (a+b)/2$. The mean minimizes the sum of squared deviations, hence

$$V = \sum_{i \in G} (y_{ti} - c)^2 - g(\bar{y} - c)^2 \leq \sum_{i \in G} (y_{ti} - c)^2 \leq \frac{gR^2}{4}.$$

Rearranging gives $4V/g \leq R^2$. □

Both bounds are tight: the upper bound with equality when $G$ contains only the two extremes ($g = 2$), the lower bound when all mass sits at the two endpoints in equal proportion. Defining

$$\Psi(\mathcal{P}) \;=\; \sum_k \mathbb{E}_t[V_k(\mathcal{P})]\,, \tag{15}$$

and summing Equation 14 over groups and tokens,

$$\frac{4}{g}\,\Psi(\mathcal{P}) \;\leq\; J_R(\mathcal{P}) \;\leq\; 2\,\Psi(\mathcal{P}). \tag{16}$$

*Remark* 1. Equation 16 controls $J_R$ within a factor of $g/2$ but does *not* imply $\arg\min \Psi = \arg\min J_R$. $\Psi$ is a surrogate whose optimum is provably within that band of the true optimum; it is not the true objective. We state this explicitly because the gap is where the method's residual headroom lies. The band is nevertheless a guarantee: if $\mathcal{P}^\star$ minimizes $\Psi$ and $\mathcal{P}^\dagger$ minimizes $J_R$, then $J_R(\mathcal{P}^\star) \leq 2\Psi(\mathcal{P}^\star) \leq 2\Psi(\mathcal{P}^\dagger) \leq \frac{g}{2} J_R(\mathcal{P}^\dagger)$.

### A.3 Dispersion as a sum of pairwise distances

**Lemma 2** (Bias–variance identity). *For any finite multiset $\{y_1, \ldots, y_g\}$ with mean $\bar{y}$,*

$$\sum_{i=1}^{g} (y_i - \bar{y})^2 \;=\; \frac{1}{g} \sum_{i<j} (y_i - y_j)^2. \tag{17}$$

*Proof.* Expand the right-hand side:

$$\sum_{i<j} (y_i - y_j)^2 = \frac{1}{2} \sum_{i,j} (y_i - y_j)^2 = \frac{1}{2} \left( 2g \sum_i y_i^2 - 2\Big( \sum_i y_i \Big)^2 \right) = g \sum_i y_i^2 - g^2 \bar{y}^2.$$

Dividing by $g$ gives $\sum_i y_i^2 - g\bar{y}^2$, which is exactly $\sum_i (y_i - \bar{y})^2$. □

Applying Lemma 2 inside each group, exchanging the finite sums with $\mathbb{E}_t$, and writing

$$d_{ij} \;\coloneqq\; \mathbb{E}_t\big[(y_{ti} - y_{tj})^2\big]\,, \tag{18}$$

we obtain

$$\Psi(\mathcal{P}) \;=\; \frac{1}{g} \sum_k \sum_{\substack{i<j \\ i,j \in G_k}} d_{ij}. \tag{19}$$

The grouping problem is now *pairwise*: place channels with small $d_{ij}$ together.

### A.4 Decomposition of the pairwise distance

**Lemma 3** (Decomposition). *With $m_i, \sigma_i^2, C_{ij}$ as in Equation 11,*

$$d_{ij} \;=\; (m_i - m_j)^2 \;+\; \sigma_i^2 \;+\; \sigma_j^2 \;-\; 2C_{ij} \;=\; \sigma_i^2 + \sigma_j^2 + D_{ij}, \qquad D_{ij} \coloneqq (m_i - m_j)^2 - 2C_{ij}. \tag{20}$$

*Proof.* $\mathbb{E}[y_i^2] = \sigma_i^2 + m_i^2$ and $\mathbb{E}[y_i y_j] = C_{ij} + m_i m_j$. Hence

$$d_{ij} = \mathbb{E}[y_i^2] - 2\mathbb{E}[y_i y_j] + \mathbb{E}[y_j^2] = (\sigma_i^2 + m_i^2) - 2(C_{ij} + m_i m_j) + (\sigma_j^2 + m_j^2),$$

and collecting the mean terms into $(m_i - m_j)^2 = m_i^2 - 2m_i m_j + m_j^2$ gives the claim. □

### A.5 The variance term is partition-free

**Theorem 1** (CAPQuant objective). *For any balanced partition $\mathcal{P}$ into groups of exactly $g$,*

$$\Psi(\mathcal{P}) \;=\; \underbrace{\frac{g-1}{g}\,\mathrm{tr}(\mathbf{\Sigma})}_{\text{independent of } \mathcal{P}} \;+\; \frac{1}{g} \sum_{G \in \mathcal{P}} \sum_{\substack{i<j \\ i,j \in G}} D_{ij}, \tag{21}$$

*and consequently*

$$\arg\min_{\mathcal{P}} \Psi(\mathcal{P}) \;=\; \arg\min_{\mathcal{P}} \sum_{G \in \mathcal{P}} \sum_{\substack{i<j \\ i,j \in G}} \Big[ (m_i - m_j)^2 - 2C_{ij} \Big]. \tag{22}$$

*Proof.* Substitute Lemma 3 into Equation 19. Within one group $G$ of size $g$, each channel $i \in G$ appears in exactly $g-1$ unordered pairs $\{i, j\}$ with $j \in G$, $j \neq i$. Therefore

$$\sum_{\substack{i<j \\ i,j \in G}} \big(\sigma_i^2 + \sigma_j^2\big) \;=\; (g-1) \sum_{i \in G} \sigma_i^2,$$

and summing over all groups,

$$\sum_{G} \sum_{\substack{i<j \\ i,j \in G}} \big(\sigma_i^2 + \sigma_j^2\big) \;=\; (g-1) \sum_{G} \sum_{i \in G} \sigma_i^2 \;=\; (g-1) \sum_{i=1}^{d} \sigma_i^2 \;=\; (g-1)\,\mathrm{tr}(\mathbf{\Sigma}),$$

where the middle equality holds because $\mathcal{P}$ is a partition: every channel lies in exactly one group, so the double sum enumerates each $\sigma_i^2$ once. This quantity depends on neither the grouping nor the ordering. Dividing by $g$ and adding the $D_{ij}$ terms gives Equation 21; dropping the constant gives Equation 22. □

**Corollary 2.** *For fixed group size, the explicit contribution of the individual channel variances to $\Psi$ is independent of the partition. Group selection can therefore be evaluated using the remaining mean-difference and covariance terms.*

This identity does not imply that sorting channels by variance cannot help: such an ordering also changes the within-group mean and covariance terms. Its effect on quantization error and model quality must be measured.

*Remark* 2 (Balance is required). For groups of variable size, applying the pairwise identity separately to each group gives the variance contribution $\sum_G (1 - 1/|G|) \sum_{i \in G} \sigma_i^2$ to the total within-group dispersion. This term generally depends on the partition. Fixed-size groups make it constant and match the grouping constraint used in our experiments.

### A.6 UNCORRELATED CHANNELS

*Proof of Corollary 1.* With $C_{ij} = 0$ for $i \neq j$, Equation 21 and Lemma 2 applied to the means within each group give

$$\Psi(\mathcal{P}) = \frac{g-1}{g}\mathrm{tr}(\mathbf{\Sigma}) + \frac{1}{g}\sum_{G\in\mathcal{P}}\sum_{\substack{i<j\\ i,j\in G}}(m_i - m_j)^2$$

$$= \frac{g-1}{g}\mathrm{tr}(\mathbf{\Sigma}) + \sum_{i=1}^{d} m_i^2 - \frac{1}{g}\sum_{G\in\mathcal{P}} S_G^2, \qquad S_G = \sum_{i\in G} m_i.$$

Minimizing $\Psi$ is therefore equivalent to maximizing $F(\mathcal{P}) = \sum_G S_G^2$. Take two groups $A$ and $B$ with $S_A \leq S_B$, and suppose that some $a \in A$ and $b \in B$ satisfy $m_a > m_b$. Exchanging $a$ and $b$ changes the two sums to $S_A + h$ and $S_B - h$ with $h = m_b - m_a < 0$, so

$$\Delta F = (S_A + h)^2 + (S_B - h)^2 - S_A^2 - S_B^2 = 2h\,(S_A - S_B + h) > 0,$$

because both factors are negative. A minimizer of $\Psi$ exists because the set of balanced partitions is finite, and by the argument above it contains no such pair. Ordering its groups by their sums, every mean in a group is therefore at most every mean in the next group, so the groups are consecutive blocks of the channels sorted by $m_i$. When means are tied, all such blocks attain the same value of $\Psi$. □

The same exchange is the one-dimensional case of Equation 10. Sorting by any other per-channel statistic, such as the variance, the second moment, or the calibration range, produces a balanced partition as well, so for uncorrelated channels none of them can attain a lower $\Psi$ than mean sorting.

### A.7 GEOMETRIC REALISATION

**Proposition 4** (Euclidean embedding)**.** *Define the centred and scaled token trace of channel $i$ as*

$$\mathbf{u}_i \in \mathbb{R}^T, \qquad \mathbf{u}_i[t] = \frac{y_{ti} - m_i}{\sqrt{T}}, \tag{23}$$

*and the channel point $\mathbf{z}_i = (m_i,\, \mathbf{u}_i) \in \mathbb{R}^{1+T}$. Then*

$$\langle \mathbf{u}_i,\, \mathbf{u}_j \rangle = C_{ij}, \qquad \|\mathbf{u}_i\|^2 = \sigma_i^2, \qquad \|\mathbf{z}_i - \mathbf{z}_j\|^2 = d_{ij}. \tag{24}$$

*Proof.* The inner product is $\langle \mathbf{u}_i,\, \mathbf{u}_j \rangle = \frac{1}{T}\sum_t (y_{ti} - m_i)(y_{tj} - m_j) = C_{ij}$ by definition of the empirical covariance, and $\|\mathbf{u}_i\|^2 = C_{ii} = \sigma_i^2$. For the distance,

$$\|\mathbf{z}_i - \mathbf{z}_j\|^2 = (m_i - m_j)^2 + \|\mathbf{u}_i - \mathbf{u}_j\|^2 = (m_i - m_j)^2 + \sigma_i^2 + \sigma_j^2 - 2C_{ij} = d_{ij}$$

by Lemma 3. □

So minimising $\Psi$ is exactly balanced $k$-means on the points $\{\mathbf{z}_i\}$, an NP-hard problem. Algorithm 1 performs local refinement of a projected objective at selected site types and does not guarantee a local optimum of the full-trace objective. Note that $\mathbf{\Sigma}$ is never formed: only the centred matrix $\mathbf{Y}_c$ is required, which is $T \times d$ rather than $d \times d$.

### A.8 TOKEN-SPACE REDUCTION

Let $\mathbf{G} = \mathbf{Y}_c\mathbf{Y}_c^\top/T$ be the token-space Gram matrix.

**Proposition 5.** *$\mathbf{G}$ and $\mathbf{\Sigma}$ have the same nonzero spectrum. Projection of the channel traces $\mathbf{u}_i$ onto the leading $r$ eigenvectors of $\mathbf{G}$ minimizes $\sum_i \|\mathbf{u}_i - \mathbf{P}_r\mathbf{u}_i\|^2$ among rank-$r$ orthogonal projections $\mathbf{P}_r$.*

*Proof.* Writing $\mathbf{Y}_c = \mathbf{U}\mathbf{S}\mathbf{V}^\top$ gives $\mathbf{G} = \mathbf{U}\mathbf{S}^2\mathbf{U}^\top/T$ and $\mathbf{\Sigma} = \mathbf{V}\mathbf{S}^2\mathbf{V}^\top/T$ on their nonzero eigenspaces. The leading left singular vectors minimize the Frobenius reconstruction error of $\mathbf{Y}_c/\sqrt{T}$, whose columns are $\mathbf{u}_i$. □

This reconstruction guarantee is not a guarantee of minimum pairwise-distance distortion: the latter depends on the scatter of the channel points around their mean across channels. Appendix B.4 instead bounds the residual contribution to the grouping dispersion directly.

Forming $\mathbf{G}$ costs $O(T^2 d)$, compared with $O(d^2 T)$ for forming the channel covariance. The token-space direction is cheaper when $T < d$; eigendecomposition is an additional cost. The final rank rule, with cap 128, is specified in Appendix B.4. The cap does not guarantee retention of 90% of the covariance energy.

### A.9 Why rotation creates the opportunity

**Proposition 6** (Conservation). *For symmetric $\mathbf{\Sigma}$,*

$$\|\mathbf{\Sigma}\|_F^2 \;=\; \sum_i \sigma_i^4 \;+\; 2\sum_{i<j} C_{ij}^2, \tag{25}$$

*and for any orthogonal $\mathbf{O}$, both $\mathrm{tr}(\mathbf{O\Sigma O}^\top) = \mathrm{tr}(\mathbf{\Sigma})$ and $\left\|\mathbf{O\Sigma O}^\top\right\|_F = \|\mathbf{\Sigma}\|_F$.*

*Proof.* Equation 25 is the definition $\|\mathbf{\Sigma}\|_F^2 = \sum_{i,j} \mathbf{\Sigma}_{ij}^2$ split into diagonal and off-diagonal parts, using symmetry to pair $(i, j)$ with $(j, i)$. Invariance of the trace is cyclicity and invariance of the Frobenius norm follows from $\left\|\mathbf{O\Sigma O}^\top\right\|_F^2 = \mathrm{tr}(\mathbf{O\Sigma O}^\top\mathbf{O\Sigma}^\top\mathbf{O}^\top) = \mathrm{tr}(\mathbf{\Sigma\Sigma}^\top) = \|\mathbf{\Sigma}\|_F^2$. □

**Corollary 3** (Flattening drives covariance to its ceiling). *Since $\mathrm{tr}(\mathbf{\Sigma}) = \sum_i \sigma_i^2$ is fixed, Cauchy–Schwarz gives $\sum_i \sigma_i^4 \geq (\mathrm{tr}\mathbf{\Sigma})^2/d$ with equality iff all $\sigma_i^2$ are equal. Hence the off-diagonal mass satisfies*

$$2\sum_{i<j} C_{ij}^2 \;\leq\; \|\mathbf{\Sigma}\|_F^2 - \frac{\mathrm{tr}(\mathbf{\Sigma})^2}{d} \;=:\; \mathrm{od}_{\max}, \tag{26}$$

*attained exactly when the diagonal is flat. A transformation that equalises variances therefore pushes covariance mass to $\mathrm{od}_{\max}$ as a matter of arithmetic, not of empirical accident.*

This is the formal version of the claim that rotation creates CAPQuant's opportunity. Measured on Qwen3-4B at `layers.0.input_layernorm`, OSTQuant moves $\max \sigma_i^2 / \min \sigma_i^2$ from $244.5\times$ to $4.2\times$ and the off-diagonal share $2\sum C_{ij}^2 / \|\mathbf{\Sigma}\|_F^2$ from $93.13\%$ to $99.92\%$. These squared-mass statistics motivate retaining cross-channel covariance, but do not establish that the signed covariance term dominates the mean differences in $D_{ij}$.

*Remark* 3. The full OSTQuant pipeline is not purely orthogonal: it fuses the RMSNorm affine $\gamma$ forward into the next linear before rotating, a per-channel diagonal rescaling. Hence $\mathrm{tr}(\mathbf{\Sigma})$ and $\|\mathbf{\Sigma}\|_F$ are *not* expected to match between FP16 and post-OSTQuant activations. The reported diagonal flatness and off-diagonal share describe the covariance shape; they do not assert conservation across the complete pipeline.

### A.10 Two facts about the permutation

Let $\mathbf{P} \in \mathbb{R}^{d\times d}$ be the permutation matrix of $\pi$.

**Proposition 7** (Orthogonality and the fold). *$\mathbf{P}^\top\mathbf{P} = \mathbf{I}$, hence for any $\mathbf{W} \in \mathbb{R}^{d\times n}$,*

$$Q(\mathbf{xP})\,\mathbf{P}^\top\mathbf{W} \;=\; Q(\mathbf{xP})\,\mathbf{W}', \qquad \mathbf{W}' = \mathbf{P}^\top\mathbf{W}, \tag{27}$$

*and $\mathbf{W}'$ is $\mathbf{W}$ with its rows permuted — a relabelling computable once, offline.*

*Proof.* $\mathbf{P}$ has exactly one 1 per row and column, so $(\mathbf{P}^\top\mathbf{P})_{ij} = \sum_k \mathbf{P}_{ki}\mathbf{P}_{kj} = \delta_{ij}$. Left-multiplying $\mathbf{W}$ by $\mathbf{P}^\top$ permutes its rows. □

**Proposition 8** (RMSNorm is permutation-invariant). *For $\mathrm{rms}(\mathbf{x}) = \sqrt{\frac{1}{d}\sum_i x_i^2 + \epsilon}$ and any permutation matrix $\mathbf{P}$, $\mathrm{rms}(\mathbf{xP}) = \mathrm{rms}(\mathbf{x})$, and consequently*

$$\mathrm{RMSNorm}(\mathbf{x})\,\mathbf{P} \;=\; \mathrm{RMSNorm}(\mathbf{xP}) \tag{28}$$

*provided the affine scale is permuted alongside, $\gamma' = \gamma\mathbf{P}$.*

*Proof.* $\sum_i (x\mathbf{P})_i^2 = \sum_i x_{\pi(i)}^2 = \sum_i x_i^2$, since $\pi$ is a bijection. The normaliser is therefore unchanged, and dividing a permuted vector by a scalar commutes with the permutation. □

Proposition 8 is what licenses fusing the forward permutation into the RMSNorm's indexed load: the kernel already reads every channel to form the sum of squares, and issuing those loads in $\pi$ order changes neither the normaliser nor the number of tensor elements read. Indexed access can still change memory transactions and latency.

*Remark* 4 (Why the MLP product needs a different site). A permutation and a Hadamard do *not* commute: $\mathbf{H}(\mathbf{P}\mathbf{v}) \neq \mathbf{P}(\mathbf{H}\mathbf{v})$ in general, since $\mathbf{H}$ mixes all coordinates. We measure $\max|\mathbf{H}\mathbf{P}\mathbf{v} - \mathbf{P}\mathbf{H}\mathbf{v}| / \max|\mathbf{H}\mathbf{v}| = 1.24$ at $d = 512$. The MLP product site sits behind an online Hadamard in OSTQuant's W4A4 configuration, so its permutation cannot be folded backwards through that transform; it must be carried by a kernel downstream of the Hadamard, which is why we place it in the quantizer's range computation rather than in the producing kernel. (Permutations of the *index bits* do commute with $\mathbf{H}$, at relative error $1.4 \times 10^{-7}$, but that subgroup is far too small to express the orderings CAPQuant finds.)

### A.11 The objective predicts model quality

On the released Llama-3-8B W4A4KV16 checkpoint of Table 2 we take eight channel orderings: contiguous grouping and the random, RPTQ-style, and mean-sorted controls of that table, and four orderings built from the full channel geometry of Section 4.2, namely the hierarchical initializer of Section 4.3 without refinement, Algorithm 1 started from the mean-sorted order instead, Algorithm 1 with refinement at all three site types, and CAPQuant itself. For each, we compute $\Psi$ from the calibration activations at full rank and sum over all 96 reordered sites. Figure 3 plots $\Psi$ against the resulting WikiText-2 perplexity. Orderings with lower $\Psi$ give lower perplexity, with Spearman $\rho = 0.83$ and Pearson $r = 0.96$. The checkpoint, the quantized weights, and the evaluation tensors are identical across all eight, so the ordering is the only source of variation.

The surrogate also tracks the quantities it stands in for. On the same activations, the range objective $J_R$ of Equation (2), the mean zero-inclusive group range of Section B.2, and the squared reconstruction error of the 4-bit quantizer of Section B.1 all rank the eight orderings identically to $\Psi$, so each has the same $\rho = 0.83$ with perplexity. CAPQuant lowers $\Psi$ by $24\%$, the mean group range by $12\%$, and the relative reconstruction error from $1.00\%$ to $0.77\%$ relative to contiguous grouping. On activations from a different calibration draw (seed 102) that no ordering was built from, the reductions shrink to $15\%$, $7\%$, and $1.00\% \to 0.86\%$, and the ranking of the eight orderings is unchanged ($\rho = 0.83$).

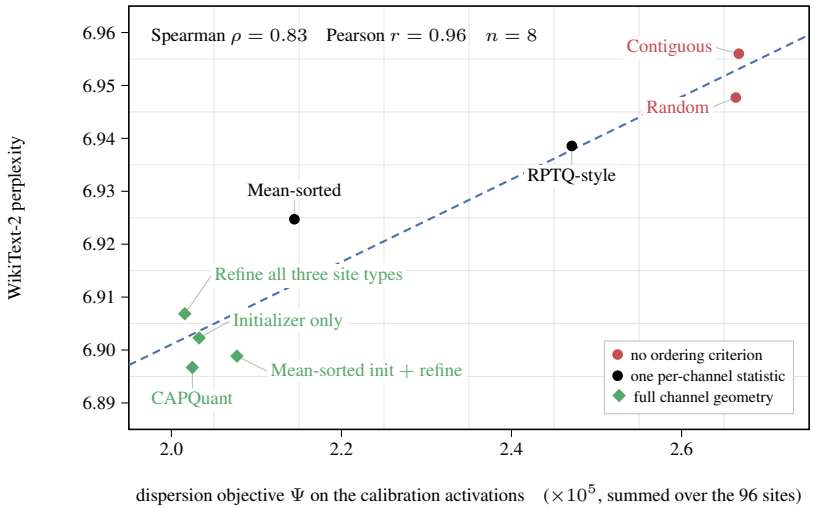


Figure 3: **The dispersion objective predicts model quality.** Each point is one channel ordering applied to the same frozen Llama-3-8B W4A4KV16 checkpoint (released OSTQuant transform, calibration draw 101). The horizontal axis is $\Psi$ computed at full rank from the calibration activations and summed over the 96 reordered sites; the vertical axis is the resulting WikiText-2 perplexity. Contiguous and random grouping (red) have the largest $\Psi$ and the worst perplexity. The RPTQ-style and mean-sorted orders (black) each rank channels by a per-channel statistic and cannot express the covariance term. The four orderings built from the full channel geometry (green; Section 4.3) sit at the low end of both axes. Spearman $\rho = 0.83$, Pearson $r = 0.96$. The relationship is strong but not exact, as expected of a surrogate reached through a bound and a projection.

# B CAPQuant derivation and implementation details

## B.1 Groupwise asymmetric quantizer

We quantize activations per token with dynamic scales, subdividing each token row $\mathbf{y}_t \in \mathbb{R}^d$ into $K = d/g$ contiguous groups $G_1, \dots, G_K$ of size $g$, each with its own scale. Because activations after RMSNorm and SwiGLU are sign-skewed, CAPQuant uses an asymmetric quantizer that stores a scale $s_{tk}$ and a zero-point $z_{tk}$ per group and shifts the integer grid $\{0, \dots, 2^b - 1\}$ to the group's range:

$$\begin{aligned} R_{tk} &= \max_{i \in G_k} y_{ti} - \min_{i \in G_k} y_{ti}, \qquad s_{tk} = \frac{R_{tk}}{2^b - 1}, \qquad z_{tk} = -\mathrm{round}\left(\frac{\min_{i \in G_k} y_{ti}}{s_{tk}}\right), \\ q_{ti} &= \mathrm{clip}\big(\mathrm{round}(y_{ti}/s_{tk}) + z_{tk},\ 0,\ 2^b - 1\big), \qquad \hat{y}_{ti} = (q_{ti} - z_{tk})\, s_{tk}. \end{aligned} \tag{29}$$

## B.2 Zero-inclusive quantization range

Our asymmetric quantizer includes zero in its range. Ignoring its small numerical floor, the relevant span is

$$R_{0,tk} = \max\left(\max_{i \in G_k} y_{ti}, 0\right) - \min\left(\min_{i \in G_k} y_{ti}, 0\right). \tag{30}$$

Adding zero as an extra observation gives

$$V_{0,tk} = \sum_{i \in G_k} y_{ti}^2 - \frac{\left(\sum_{i \in G_k} y_{ti}\right)^2}{g+1}, \qquad \frac{4}{g+1} V_{0,tk} \le R_{0,tk}^2 \le 2 V_{0,tk}. \tag{31}$$

For fixed-size groups, the corresponding total dispersion satisfies

$$\Psi_0(\mathcal{P}) = \sum_k \mathbb{E}_t[V_{0,tk}] = \frac{g}{g+1} \Psi(\mathcal{P}) + \frac{1}{g+1} \sum_i \mathbb{E}_t[y_{ti}^2]. \tag{32}$$

The last term is independent of the partition, so $\Psi_0$ and $\Psi$ rank balanced partitions identically. CAPQuant therefore retains the ordinary dispersion objective. It does not directly optimize the quantizer's range or rounding error.

### B.3 MULTISCALE INITIALIZATION

A mean-sorted partition uses only one coordinate of the channel geometry. Channels with similar means can still have different token-wise behavior, and a local search started from this partition can miss useful groupings. We instead initialize from the channel points. The reported configuration recursively splits the reduced points $\widetilde{\mathbf{z}}_i$ with capacity-constrained two-means, assigning each node's channels to two children of prescribed sizes by their margin between the two centroids, until every leaf holds exactly $g$ channels. The leaves, read left to right, form $\pi_0$. This avoids any $O(d^2)$ distance matrix and builds an 11,008-channel site in about one second. An alternative hierarchy uses full activation traces, whose Euclidean distance satisfies

$$\delta_{ij} = \frac{1}{\sqrt{T}} \|\mathbf{Y}[:, i] - \mathbf{Y}[:, j]\| = \|\mathbf{z}_i - \mathbf{z}_j\| = \sqrt{d_{ij}}. \tag{33}$$

We construct an average-linkage hierarchy by repeatedly merging the clusters with smallest average cross-cluster distance,

$$L(A, B) = \frac{1}{|A||B|} \sum_{i \in A} \sum_{j \in B} \delta_{ij}. \tag{34}$$

The hierarchy uses Euclidean distances, while the dispersion objective uses their squares. Both use the full channel geometry, but they serve different purposes: linkage proposes an initialization, and dispersion scores the balanced partition. The implementation uses raw-trace distances $\sqrt{T}\,\delta_{ij}$. This common factor leaves the linkage decisions and leaf-ordering objective unchanged.

We then apply optimal leaf ordering within the hierarchy, minimizing the sum of Euclidean distances between consecutive leaves among orders compatible with the fixed tree. Cutting the ordered leaves every $g$ positions gives

$$\pi_0 = \mathrm{OLO}(\mathcal{H}(\delta), \delta)\,, \qquad \mathcal{P}_0 = \mathrm{cut}_g(\pi_0). \tag{35}$$

The hierarchy captures relationships at several merge scales, while the cut enforces exactly $g$ channels per group. These groups need not be subtrees. Neither hierarchy globally minimizes the balanced dispersion objective; each supplies a starting partition for the refinement.

### B.4 PROJECTION AND ADAPTIVE-WEIGHTING ABLATION

The initializer and the local swaps use a reduced representation. With $T = 4096$ calibration tokens, we compute the eigendecomposition of $\mathbf{G} = \mathbf{Y}_c \mathbf{Y}_c^\top / T \in \mathbb{R}^{T \times T}$, where $\mathbf{Y}_c = \mathbf{Y} - \mathbf{1}\mathbf{m}^\top$. Let its eigenvalues be ordered as $\lambda_1 \geq \cdots \geq \lambda_T \geq 0$. For nonzero total energy, set

$$r = \min\left( \min\left\{ k : \frac{\sum_{\ell=1}^{k} \lambda_\ell}{\sum_{\ell=1}^{T} \lambda_\ell} \geq 0.90 \right\}, 128,\ \#\{\ell : \lambda_\ell > 10^{-10} \lambda_1\} \right). \tag{36}$$

The rank cap limits computation, and the relative eigenvalue cutoff excludes numerically negligible directions. Because of these caps, the retained energy can be below 90%. The implementation rejects zero-energy sites rather than evaluating an undefined energy ratio.

Let $\mathbf{V}_r$ contain the retained orthonormal eigenvectors. The reduced points and objective are

$$\widetilde{\mathbf{u}}_i = \mathbf{V}_r^\top \mathbf{u}_i, \qquad \widetilde{\mathbf{z}}_i = (m_i, \widetilde{\mathbf{u}}_i), \tag{37}$$

$$\Psi_r(\mathcal{P}) = \sum_{G \in \mathcal{P}} \sum_{i \in G} \left\| \widetilde{\mathbf{z}}_i - \bar{\widetilde{\mathbf{z}}}_G \right\|^2. \tag{38}$$

CAPQuant retains the mean coordinate and uses unit weight for the projected covariance coordinates. Projection is the only change to the derived metric. Orthogonality gives a direct bound on the approximation. Define $\mathbf{u}_i^\perp = (\mathbf{I} - \mathbf{V}_r \mathbf{V}_r^\top)\mathbf{u}_i$ and $E_\perp = \sum_i \left\| \mathbf{u}_i^\perp \right\|^2$. Then

$$\Psi(\mathcal{P}) = \Psi_r(\mathcal{P}) + \Psi_\perp(\mathcal{P}), \qquad 0 \leq \Psi_\perp(\mathcal{P}) \leq E_\perp, \tag{39}$$

where $\Psi_\perp$ is the within-group dispersion of the residual coordinates. Thus, if refinement decreases $\Psi_r$ by $\eta$, the full objective satisfies

$$\Psi(\mathcal{P}_{\text{final}}) - \Psi(\mathcal{P}_0) \le -\eta + E_\perp. \tag{40}$$

A decrease in the projected objective alone need not decrease the full objective.

**Adaptive weighting.** An earlier version of the method used points $(m_i, \alpha\widetilde{\mathbf{u}}_i)$ with rank capped at 64 and the adaptive weight

$$M_{ij} = (m_i - m_j)^2, \qquad \text{rr} = \frac{\text{sd}_{i\neq j}(2C_{ij})}{\text{sd}_{i\neq j}(M_{ij})}, \qquad \alpha = \frac{\text{rr}}{\text{rr} + 0.35}. \tag{41}$$

This rule attenuates covariance coordinates when their pairwise variation is small relative to the mean term. Because any $\alpha \neq 1$ optimizes a different distance from the dispersion surrogate, no reported result uses it. Every result uses rank cap 128 and $\alpha = 1$.

### B.5 Swap gain and stopping rule

For $\mathbf{s}_G = \sum_{i\in G} \widetilde{\mathbf{z}}_i$, the reduced objective satisfies

$$\Psi_r(\mathcal{P}) = \sum_i \|\widetilde{\mathbf{z}}_i\|^2 - \frac{1}{g} F(\mathcal{P}), \qquad F(\mathcal{P}) = \sum_{G\in\mathcal{P}} \|\mathbf{s}_G\|^2 . \tag{42}$$

Swapping $i \in A$ and $j \in B$ changes the two sums to $\mathbf{s}_A + \mathbf{h}$ and $\mathbf{s}_B - \mathbf{h}$, where $\mathbf{h} = \widetilde{\mathbf{z}}_j - \widetilde{\mathbf{z}}_i$. Expanding their squared norms gives Equation 10 and $\Delta\Psi_r = -\Delta F_{ij}/g$. Swaps on disjoint group pairs have additive gains because they change separate sums.

Each round scores every cross-group swap and retains the best candidate for each group pair. Candidates with $\Delta F > 10^{-12}$ are considered in descending order. A candidate is accepted only when neither group has already been used in the round. When no candidate remains, the partition is locally optimal with respect to single swaps up to this tolerance. The implementation has a 20,000-round safety limit and records whether convergence was reached.

### B.6 OSTQuant and covariance after rotation

OSTQuant (Hu et al., 2025) learns orthogonal transformations $\mathbf{O}$ (rotations on the Stiefel manifold plus fixed Hadamard blocks) and diagonal smoothing scales, fuses them into the weight matrices, and applies GPTQ (Frantar et al., 2022) to the weights. Activations are quantized per token with dynamic scales, so the calibration set determines $\mathbf{O}$ rather than any static per-channel statistics. CAPQuant operates on the transformed activations $\mathbf{y}_t$ at each site. Writing $\mathbf{\Sigma}$ for their covariance, with $\sigma_i^2$ the channel variances and $C_{ij}$ the pairwise covariances,

$$\|\mathbf{\Sigma}\|_F^2 \;=\; \sum_i \sigma_i^4 \;+\; 2\sum_{i<j} C_{ij}^2. \tag{43}$$

Since a purely orthogonal rotation preserves $\|\mathbf{\Sigma}\|_F$ (Section 2), equalizing the diagonal moves energy into the off-diagonal terms rather than removing it. At the layer-0 input LayerNorm site of Qwen3-4B, OSTQuant reduces the spread of channel variances from $244.5\times$ to $4.2\times$ while the off-diagonal share of $\|\mathbf{\Sigma}\|_F^2$ rises from $93.1\%$ to $99.9\%$. This example motivates retaining cross-channel covariance when selecting groups. Section 4 shows how covariance and mean differences enter the grouping objective.

CAPQuant acts on the activation sites OSTQuant leaves with a single per-row scale: the input LayerNorm output, the post-attention LayerNorm output, and the MLP SwiGLU product in every layer. Each is converted to a permuted $g = 128$ grouping, and all other sites, already grouped at $g = 128$ by OSTQuant, are unchanged. Per-model site counts are given in Section C.

### B.7 Compatibility with a base quantization method

CAPQuant requires a group-wise activation quantizer whose groups are defined by contiguous channel positions and an exact place to absorb the inverse permutation into each consumer. It is independent

of the weight-rounding rule and can follow an activation rotation because the channel order is selected from the already-transformed activations. We validate this integration with OSTQuant, QuaRot, and SpinQuant. For pipelines that already specify an ordering, such as RPTQ, DuQuant, or PermuQuant, replacing that stage with CAPQuant would require checking the following conditions. Without an exact inverse fold, the reordering does not preserve the original computation. Without kernel fusion, applying the forward permutation requires an additional runtime operation.

FlatQuant (Sun et al., 2024) illustrates both conditions. Its online affine transformation precedes every activation quantizer, so the forward permutation could be written by that transformation's kernel, as it is by our RMSNorm kernel, and the inverse folded into the following weights as in Section 4.4. FlatQuant, however, learns its activation clipping thresholds jointly with its transformations for a single per-token scale, and grouped scales would change the quantity those thresholds were trained to clip. A faithful integration would therefore re-learn the thresholds, which is outside a training-free plug-in, and we leave it to future work.

## C Experimental settings

**Models and transforms.** We evaluate Qwen3-4B (Yang et al., 2025) (revision `1cfa9a72`), Llama-3.2-1B (Grattafiori et al., 2024) (`4e20de36`), Llama-3-8B (`315b2009`), Llama-2-7B and Llama-2-13B (Touvron et al., 2023b) (`8efe6c9b`), and Llama-1-30B (Touvron et al., 2023a) (`2b1edcdb`). They have 36, 16, 32, 32, 40, and 60 layers, so CAPQuant reorders 108, 48, 96, 96, 120, and 180 sites. We use the released OSTQuant transforms wherever they exist. The release covers Llama-1-30B, Llama-2-13B, Llama-3-8B, and Llama-2-7B at W4A4KV4 only. For Qwen3-4B, Llama-3.2-1B, and Llama-2-7B at W4A4KV16, we trained transforms using the released recipe. Weight calibration uses 128 WikiText-2 sequences at seed 42. Within each comparison, every arm shares one transform, one set of quantized weights, and one set of frozen evaluation tensors. The W4A4KV16 and W4A4KV4 columns use different released transforms and are not nested.

**CAPQuant configuration.** Every CAPQuant result uses one configuration of Algorithm 1. Groups have $g = 128$ channels. Calibration uses two 2,048-token windows drawn at random from the WikiText-2 training split at seed 42. The initializer is the GPU-capacity-constrained hierarchical two-means on the reduced channel points, with a rank cap of 128 at 90% energy and $\alpha = 1$. Swap refinement runs to convergence at the post-attention normalization and MLP-product sites and keeps the initializer at the attention-input sites. The RPTQ-style, random, and mean-sorted controls use the same two calibration windows. The RPTQ-style control represents each channel by its calibration minimum and maximum, clusters these pairs with $k$-means into $d/g$ clusters, and cuts the resulting order into groups of exactly $g$ channels. It reproduces the reordering criterion of RPTQ (Yuan et al., 2023) but not its full method, which uses unequal cluster sizes and static per-cluster scales, whereas every arm here uses balanced groups and dynamic per-token scales. The random control draws one uniformly random permutation per site.

**Evaluation.** Perplexity uses a sequence length of 2,048 and complete blocks. C4 (Raffel et al., 2020) is prepared as in GPTQ: the first 1,100 documents of validation shard 00000, joined by single spaces, tokenized with `use_fast=False`, and capped at 524,288 tokens, which gives 252 blocks for Llama-3/3.2 and 256 otherwise. WikiText-2 (Merity et al., 2016) uses the full test set. Accuracy is the nine-task macro-average over ARC-Challenge and ARC-Easy (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), LAMBADA (Paperno et al., 2016), OpenBookQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), SocialIQA (Sap et al., 2019), and WinoGrande (Sakaguchi et al., 2021), computed with `lm-evaluation-harness` 0.4.13 at batch size 16. All quality numbers use simulated activation quantization with the quantizer of Section B.1 and no activation clipping. Table 4 keeps uniform integer quantization at every $g$, so $g = 32$ matches the MXFP4 group size without implementing the format.

**Timing.** We measure prefill at a sequence length of 2,048 on one H200 using BF16 matrix multiplication and simulated activation quantization. Table 6 averages 60 calls after 15 warmups on one Qwen3-4B decoder layer and scales to 36 layers. Table 5 reports the median of three paired differences with alternating arm order, five warmups, and 20 timed forwards per arm at batch size one. Both arms use the same fused quantizer with identity or learned indices and folded consumer

Table 5: **Construction cost and added inference latency.** The cost of applying CAPQuant to 7B and 30B models.

| Model | Sites | Build (s) | Build GPU (GiB) | Added (ms) | Added (%) |
|---|---|---|---|---|---|
| Llama-2-7B | 96 | 82.04 | 3.65 | 3.27 | 3.58 |
| Llama-1-30B | 180 | 231.49 | 9.64 | 22.80 | 6.88 |

Build time is summed across sites, and GPU memory is the peak allocation per construction worker. Added latency is relative to contiguous ordering with the same fused quantizer. Both models use one H200, batch size one, and sequence length 2,048 under W4A4KV16.

Table 6: **CAPQuant permutation implementation ablation.** OSTQuant + CAPQuant on Qwen3-4B W4A4KV16, sequence length 2,048, on one H200. Cost is the per-layer overhead scaled to all 36 layers. Measurements use simulated activation quantization and BF16 matrix multiplication.

| # | Configuration | Perm. cost |
|---|---|---|
| 1 | + explicit gather–scatter | 29.53 ms |
| 2 | + inverse folded | 5.48 ms |
| 3 | + folded + RMSNorm fused | 3.85 ms |
| 4 | + folded + norm/quant fused | **2.34 ms** |
| 5 | + single fused quantizer, all sites | <u>2.42 ms</u> |
| 6 | + norm sites only | 0.01 ms |

Cost is the added latency over the 51.859 ms contiguous-grouping baseline. Rows 1–5 implement the same full-site permutation and differ only in how it is materialized; bold and underline mark the best and second-best among them. Row 6 permutes only the 72 normalization sites and leaves the 36 MLP-product sites untouched.

weights. Construction time sums the per-site work and excludes weight calibration and activation capture. Construction memory is the peak GPU allocation per worker.

# D Construction and inference cost

## D.1 Cost across model sizes

Table 5 separates the offline cost of constructing the channel order from the latency of applying it during inference. The model-size comparison retains its quality gains with added latency of $3.58\%$ on Llama-2-7B and $6.88\%$ on Llama-1-30B. These measurements use BF16 matrix multiplication and simulated activation quantization.

## D.2 Permutation implementation ablation

Table 6 holds the permutation fixed and varies only how it is materialized. A naive implementation that gathers and scatters the activation tensor adds 29.53 ms. Absorbing the inverse permutation into the weight columns of each consuming linear layer removes most of that, leaving 5.48 ms. Fusing the forward permutation into the RMSNorm write brings it to 3.85 ms, and fusing the MLP-product sites into the quantizer's range computation brings it to 2.34 ms, or $4.5\%$ of the 51.859 ms contiguous-grouping baseline. Routing all sites through a single fused quantizer instead of two kernels costs 2.42 ms, within 0.08 ms of the best configuration. Folding and fusion therefore remove $92\%$ of the naive cost.

Restricting the permutation to the 72 normalization sites and leaving the 36 MLP-product sites untouched costs 0.01 ms. This configuration applies the permutation at fewer sites than the full-site comparisons. Nearly all of the 2.34 ms therefore comes from the MLP-product sites, where the online Hadamard forces the permutation into the quantizer's range scan (Section A.10), and permuting at the RMSNorm write is effectively free.

Table 7: **MGSM accuracy (%) on Qwen3-4B.** Strict match averaged over ten languages (Bengali excluded, see text) and flexible extraction averaged over all eleven. Δ is the gain over the row above.

| | Strict match (10 lang.) | | Flexible extraction (11 lang.) | |
|---|---|---|---|---|
| Method | Acc.↑ | Δ | Acc.↑ | Δ |
| FP16 | 60.1 | – | 66.3 | – |
| *W4A4KV16* | | | | |
| OSTQuant | 33.6 | – | 36.2 | – |
| + Contiguous grouping | 41.0 | +7.5 | 41.9 | +5.7 |
| + **CAPQuant** | **41.8** | +0.8 | **42.0** | +0.2 |
| *W4A4KV4* | | | | |
| OSTQuant | 27.7 | – | 31.4 | – |
| + Contiguous grouping | 32.4 | +4.7 | 36.0 | +4.7 |
| + **CAPQuant** | **35.5** | +3.1 | **38.5** | +2.4 |

Table 8: **Per-language MGSM strict-match accuracy (%) on Qwen3-4B.** Latin-script languages first. Each language has 250 problems, so an entry carries a standard error of about three points. Bengali is omitted because the strict-match filter scores it at zero for every arm.

| Precision | Method | en | de | es | fr | sw | ja | ru | te | th | zh | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FP16 | – | 90.0 | 68.4 | 81.6 | 72.8 | 18.4 | 14.8 | 78.8 | 30.4 | 74.0 | 71.6 | 60.1 |
| W4A4KV16 | OSTQuant | 54.8 | 45.6 | 49.6 | 40.0 | 4.0 | 4.0 | 50.8 | 9.2 | 34.0 | 43.6 | 33.6 |
| | + Contiguous | 70.8 | 52.4 | 56.4 | 52.8 | 6.4 | 7.2 | 58.0 | 14.8 | 41.2 | 50.4 | 41.0 |
| | + CAPQuant | 67.2 | 53.6 | 60.4 | 52.0 | 9.6 | 5.6 | 58.8 | 16.4 | 42.4 | 52.4 | 41.8 |
| W4A4KV4 | OSTQuant | 52.8 | 37.6 | 39.2 | 39.6 | 6.4 | 3.2 | 41.6 | 2.0 | 21.6 | 33.2 | 27.7 |
| | + Contiguous | 54.8 | 44.4 | 44.0 | 44.0 | 7.2 | 4.4 | 49.2 | 5.2 | 27.2 | 43.6 | 32.4 |
| | + CAPQuant | 61.6 | 45.6 | 48.8 | 51.6 | 6.4 | 4.0 | 50.8 | 8.4 | 30.4 | 47.6 | 35.5 |

## E MULTILINGUAL REASONING UNDER W4A4

Perplexity and multiple-choice accuracy can hide damage that only surfaces during generation, and four-bit quantization is known to hurt multilingual mathematical reasoning far more than English-commonsense tasks (Marchisio et al., 2024). We therefore evaluated MGSM (Shi et al., 2022) on Qwen3-4B with the lm-evaluation-harness (Biderman et al., 2024) (version 0.4.9, eight-shot native chain-of-thought prompts, greedy decoding, 384 generated tokens, batch size 32, context length 8,192). All arms share the FP16 weights, the quantized weights, the transformation, and the calibration permutations of Table 2, and the KV-cache quantizer is active during generation. The harness's strict-match filter requires an English answer phrase for Bengali and scores it at zero for every arm including FP16, so Table 7 averages strict-match accuracy over the other ten languages and reports the harness's flexible-extraction filter over all eleven. The 384-token budget truncates some FP16 responses in Telugu and Bengali, and all arms have the same budget.

W4A4 costs Qwen3-4B roughly half of its FP16 accuracy, and grouping recovers a large part of it. The confidence intervals below are paired 95% intervals across the 250 translated problems, with each problem contributing its mean across languages to the corresponding average. Under W4A4KV4, contiguous grouping raises the strict-match accuracy of OSTQuant by 4.7 points ($[3.1, 6.3]$) and CAPQuant adds a further 3.1 points ($[1.5, 4.8]$). Under W4A4KV16, grouping adds 7.5 points ($[5.7, 9.3]$) and ordering adds 0.8 points ($[-1.0, 2.6]$). Flexible extraction gives ordering gains of 2.4 points ($[0.7, 4.2]$) under W4A4KV4 and 0.2 points ($[-1.7, 2.0]$) under W4A4KV16. As in perplexity, grouping accounts for most of the recovery, and ordering adds a smaller gain that is significant only under W4A4KV4. Table 8 lists per-language strict-match accuracy, where individual entries carry a standard error of about three points.