DUALQUANT: DUAL-SCALE CALIBRATION-FREE WEIGHT-AND-ACTIVATION QUANTIZATION FOR LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Low-bit inference on accelerators with native 4-bit support requires quantizing both weights *and* activations to realize the full throughput gains these formats provide. While post-training weight quantization has been extensively studied, activation quantization remains considerably more challenging and typically requires calibration data. In this work, we introduce DUALQUANT, a dual-scale, calibration-free post-training quantization framework that directly minimizes the weight-reconstruction Mean Squared Error (MSE) and, as an immediate by-product, yields per-channel activation rescalings prior to activation quantization. Our key observation is that the scaling factors obtained by minimizing weight-reconstruction error also provide effective activation rescalings, enabling competitive activation quantization without calibration data. We formulate DUALQUANT as a non-differentiable weight-reconstruction problem and develop an efficient solver based on variable splitting and block coordinate descent, where each subproblem admits a closed-form update. By treating the quantizer as a black box and projecting scales onto their *format-specific admissible set* after each update, DUALQUANT provides a unified algorithm supporting a wide range of formats, including INT4, INT8, the full MX family, and NVFP4. To our knowledge, DUALQUANT is the first dual-scaling method to formally support MX and NV block formats. Across Llama and Qwen models evaluated on WikiText-2, C4, and zero-shot downstream tasks, DUALQUANT matches or exceeds the accuracy of calibration-based SmoothQuant across most models and formats, and substantially outperforms calibration-free baselines, with particularly large gains for MX and NV block formats. We further validate these gains with a real packed W4A4 inference path for Llama-3.1-8B on an RTX 4090. The resulting implementation closely matches the accuracy of fake quantization while delivering up to $2.61\times$ memory reduction and up to $2.46\times$ throughput speedup over BF16. Our anonymized codebase is available here.

1 INTRODUCTION

Large Language Models (LLMs) have grown rapidly in scale, from dense models (Grattafiori et al., 2024; Yang et al., 2025) to mixture-of-experts architectures exceeding a trillion parameters (DeepSeek-AI, 2024; Jiang et al., 2024), placing enormous pressure on memory and compute at inference time. Post-Training Quantization (PTQ) addresses this by compressing weights, activations, and KV-cache to low-bit representations without retraining. Modern accelerators such as NVIDIA Blackwell GPUs provide native 4-bit matrix multiplication (Rouhani et al., 2023; NVIDIA, 2025) delivering up to $4\times$ the throughput of FP16, but realizing this gain requires *both* weights and activations in a low-bit format. The central obstacle is that activation tensors exhibit systematic large-magnitude outliers in specific channels (Dettmers et al., 2022; Bondarenko et al., 2021; Sun et al., 2024) that dominate the quantization grid. Calibration-based methods address activation quantization effectively but require representative data, introducing distribution sensitivity and data-access overhead prohibitive in privacy-sensitive or on-device settings. This includes both scale-based approaches such as SmoothQuant (Xiao et al., 2023) and rotation-based approaches such as SpinQuant (Liu et al., 2025), which optimizes rotation matrices using calibration data. Calibration-free rotation-based methods such as QuaRot (Ashkboos et al., 2024) avoid this dependency but do not target activation-quantization error. In other words, no prior method achieves competitive calibration-free joint Weight and Activation (W+A) quantization across modern MX and NV formats.

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.

In this work, we introduce DUALQUANT, a calibration-free dual-scale (per-row and per-column) PTQ framework that jointly quantizes weights and activations *without any access to calibration data or activation statistics*. The key insight is that channels with large-magnitude weight columns contribute disproportionately to the layer output and, as shown empirically, concentrate activation outliers (Sun et al., 2024; Dettmers et al., 2022; Bondarenko et al., 2021). Consequently, the column scales that minimize weight-reconstruction MSE attenuate these same channels, so they not only quantize the weights well but also implicitly define activation rescalings that we observe to substantially improve activation quantization—playing a role similar to SmoothQuant's calibrated scales (Xiao et al., 2023), yet derived from weights alone. DUALQUANT obtains these row and column scales via Block Coordinate Descent (BCD) with closed-form updates and provable convergence guarantees (Tseng, 2001; Razaviyayn et al., 2013), treating the quantizer as a black-box operator and projecting scales to their admissible format set after each update, yielding a single unified algorithm that operates across INT4, MXINT4, MXFP4, and NVFP4. Evaluation on Qwen and Llama families confirms that DUALQUANT matches or surpasses SmoothQuant on W+A quantization across most formats and models, without any calibration data. Our main contributions are:

1. We formalize dual-scale weight quantization as a constrained non-convex optimization problem and solve it via BCD with closed-form per-coordinate updates, format-aware scale projection, and provable convergence guarantees, requiring no inner loop, no Hessian, and no calibration data. The same algorithm operates uniformly across INT4, MXINT4, MXFP4, and NVFP4, making DUALQUANT the first dual-scaling method that formally supports the full MX and NV formats.
2. We show that the column scales obtained from weight MSE minimization also serve as effective activation rescalings, enabling joint W+A quantization without any access to calibration data. Across Qwen and Llama families and four 4-bit formats, DUALQUANT closely matches or surpasses calibration-based SmoothQuant and significantly outperforms calibration-free baselines QuaRot and SINQ+, with especially large margins on MX and NV block data formats.
3. We validate DUALQUANT through a real packed W4A4 deployment of Llama-3.1-8B on an RTX 4090. The packed implementation matches fake-quantization accuracy to within 0.01 perplexity on WikiText-2 while reducing memory usage by 2.51–$2.61\times$ and improving end-to-end speed by 1.39–$2.46\times$ over BF16. Incorporating the scales into the activation-quantization kernel avoids additional kernel launches and limits the quantization overhead to 4.3-6.1% of total GPU time.

2 BACKGROUND AND RELATED WORK

2.1 POST-TRAINING WEIGHT QUANTIZATION

A rich body of PTQ methods has been proposed to close the gap between quantized and full-precision quality: GPTQ (Frantar et al., 2023) compensates for quantization error via second-order weight correction; AWQ (Lin et al., 2024) protects salient channels via activation-weighted scaling. Calibration-free approaches HQQ (Badri & Shaji, 2023) and QuantEase (Behdin et al., 2023) minimize per-layer reconstruction objectives without requiring calibration data while AdpQ (Ghaffari et al., 2024) minimizes weight distribution divergence via adaptive LASSO. SINQ (Müller et al., 2026) applies dual row and column scaling via Sinkhorn-Knopp iteration (Sinkhorn & Knopp, 1967), minimizing matrix imbalance as a proxy for weight quantization error. Despite their variety, these methods quantize weights only, leaving activations in BF16 and forgoing the full throughput of low-precision accelerators.

2.2 ACTIVATION QUANTIZATION

Activations are significantly harder to quantize than weights (Dettmers et al., 2022; Bondarenko et al., 2021): they exhibit systematic large-magnitude outliers in specific channels (Xiao et al., 2023; Wei et al., 2022) that dominate the quantization grid and leave normal channels with very few effective bits. LLM.int8() (Dettmers et al., 2022) handles outlier channels via mixed-precision decomposition. SmoothQuant (Xiao et al., 2023) migrates the quantization burden from activations to weights via calibration-based per-channel rescaling. ZeroQuant (Yao et al., 2022) applies dynamic per-token activation quantization alongside group-wise weight quantization while OmniQuant (Shao et al., 2024) and AffineQuant (Ma et al., 2024) jointly optimize scaling and clipping via learnable transformations. Rotation-based methods (Tseng et al., 2024; Ashkboos et al., 2024; Sun et al., 2025; Czakó et al., 2025; Liu et al., 2025; Saxena et al., 2025; Zandieh et al., 2026) suppress outliers by spreading

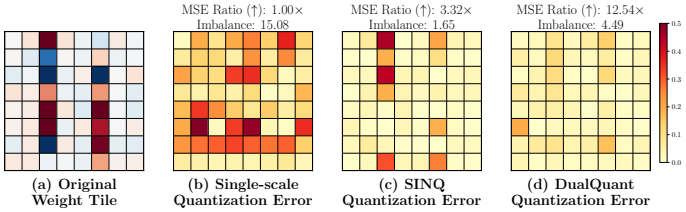


Figure 1: **Per-element quantization error $|\mathrm{W}-\hat{\mathrm{W}}|$ under different scale designs.** Each panel reports the MSE ratio (↑) relative to single-scale and the matrix imbalance (SINQ's objective). **(a)** Original weight tile. **(b)** Single-scale: one outlier inflates the entire row. **(c)** SINQ minimizes matrix imbalance (1.65) but achieves only $3.32\times$ MSE reduction. **(d)** DUALQUANT minimizes MSE, achieving $12.54\times$ reduction, illustrating that low imbalance does not imply low reconstruction error.

activation energy incoherently across dimensions via Hadamard, orthogonal rotations, or learned affine transforms. While effective, most of these approaches depend on calibration data, motivating the search for calibration-free alternatives that can achieve competitive W+A quantization. We propose DUALQUANT, a calibration-free joint W+A quantization method that leverages a principled MSE optimization of weight quantization via block coordinate descent with closed-form updates and provable convergence guarantees (Appendix C), extending naturally to MX and NV format families. Most significantly, we show that the column scales derived from minimizing weight-reconstruction error also serve as effective activation rescalings, enabling joint activation and weight quantization that matches and surpasses SmoothQuant across most formats and models, without any access to activations or calibration data. As illustrated in Figure 1, directly targeting the MSE yields substantially lower reconstruction error than methods such as SINQ (Müller et al., 2026) that optimize a proxy objective, and we show in Section 5 that this improvement translates directly to better downstream performance.

3 QUANTIZATION PRELIMINARIES

Quantization maps a tensor to a finite, format-specific set of representable values. Given a tensor W, let $\mathcal{G}$ denote the canonical, unit-scale grid of the target format, namely, the integers $\{-2^{B-1}, \ldots, 2^{B-1}-1\}$ for INTB, or the non-uniform exponent-mantissa grid for a floating-point format. A quantizer $Q(\cdot)$ with scale Δ and zero-point z maps each element to the nearest representable value, $Q(W) = \Delta \cdot (\mathrm{proj}_{\mathcal{G}}(W/\Delta + z) - z)$, where $\mathrm{proj}_{\mathcal{G}}(\cdot)$ rounds and clips its argument to the nearest point on $\mathcal{G}$, reducing to standard round-and-clip for integer formats and to nearest-level projection on the non-uniform exponent-mantissa grid for floating-point formats. Symmetric quantization sets $z = 0$ with $\Delta = \max(|W|)/q_{\max}$, where $q_{\max}$ denotes the maximum positive quantization level; asymmetric quantization uses nonzero z to span $[\min(W), \max(W)]$. To reduce sensitivity to outliers, block-wise quantization partitions each row into groups of G contiguous elements with independent scales. The OCP microscaling formats (Rouhani et al., 2023) (MXFP4, MXINT4) use $G = 32$ with a shared E8M0 scale restricted to powers of two; NVFP4 (NVIDIA, 2025) uses $G = 16$ with an FP8 UE4M3 scale. This motivates the format-agnostic design of DUALQUANT, treating $Q(\cdot)$ as a black-box with an explicit scale constraint (Section 4.2).

4 METHODOLOGY

4.1 PROBLEM DESCRIPTION

For a linear layer computing $\mathbf{X}\mathbf{W}^\top$ with activation matrix $\mathbf{X} \in \mathbb{R}^{b\times n}$ and weight matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$, the joint W+A quantization error can be written as

$$\mathbf{X}\mathbf{W}^\top - Q(\mathbf{X})\,Q(\mathbf{W})^\top = (\mathbf{X} - Q(\mathbf{X}))\mathbf{W}^\top + Q(\mathbf{X})(\mathbf{W} - Q(\mathbf{W}))^\top, \tag{1}$$

where the activation error is amplified by $\mathbf{W}^\top$ and the weight error is amplified by $Q(\mathbf{X})$. Minimizing the two terms independently does not necessarily minimize the joint error, and minimizing equation 1 directly is intractable without calibration data on $\mathbf{X}$. SmoothQuant (Xiao et al., 2023) targets this

joint problem by collecting per-channel activation statistics from a calibration dataset to construct a per-channel rescaling applied prior to quantization, $\mathbf{X}\mathbf{W}^\top = (\mathbf{X}\,\mathrm{Diag}(\mathbf{s})^{-1})(\mathbf{W}\,\mathrm{Diag}(\mathbf{s}))^\top$. We take a different approach: rather than collecting activation statistics from data, we show that the column scales obtained from minimizing the *weight* reconstruction MSE on static weights implicitly perform a per-channel rescaling of the same form, without any access to $\mathbf{X}$, as developed next.

4.2 Dual-scale parameterization and the activation factorization

Standard block quantization assigns a single shared scale per block along each row of the weight matrix, so an outlier at position (i, j) inflates the scale shared by every element in that block, while the column dimension remains entirely unaddressed. Figure 1 illustrates this: the single-scale error matrix exhibits pronounced column-wise outlier patterns that persist across all rows. We mitigate this by introducing a second, column-wise scale alongside the per-block row scale. The resulting dual scaling distributes the error across both dimensions, substantially reducing the peak element-wise reconstruction error (Figure 1(d)). Specifically, we quantize $\mathbf{W} \in \mathbb{R}^{m\times n}$ (or a block thereof) as

$$\mathbf{W} \approx \mathrm{Diag}(\boldsymbol{\alpha})^{-1}\, Q(\mathrm{Diag}(\boldsymbol{\alpha})\,\mathbf{W}\,\mathrm{Diag}(\boldsymbol{\beta}))\,\mathrm{Diag}(\boldsymbol{\beta})^{-1} \;\triangleq\; \mathrm{Diag}(\boldsymbol{\alpha})^{-1}\,\hat{\mathbf{Q}}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\,\mathrm{Diag}(\boldsymbol{\beta})^{-1}, \quad (2)$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are respectively row and column scales, $Q(\cdot)$ is a fixed format-specific quantizer, and $\hat{\mathbf{Q}}_{\boldsymbol{\alpha},\boldsymbol{\beta}} = Q(\mathrm{Diag}(\boldsymbol{\alpha})\,\mathbf{W}\,\mathrm{Diag}(\boldsymbol{\beta}))$ denotes the quantized representation stored at low precision. The inverse scales are constrained to their format-admissible sets: $\boldsymbol{\alpha}^{\odot-1} \in \mathbb{F}_\alpha^m$ and $\boldsymbol{\beta}^{\odot-1} \in \mathbb{F}_\beta^n$, where $(\cdot)^{\odot-1}$ denotes the element-wise inverse, and $\mathbb{F}_\alpha^m$, $\mathbb{F}_\beta^n$ are the admissible scale sets of the target format (BF16 for INT4; E8M0 for MXINT4/MXFP4; FP8 UE4M3 for NVFP4). The row scale absorbs row-level dynamic range as in standard formats, while the column scale captures the systematic column-wise structure that single-scale formats ignore. Hence, an outlier at position (i, j) is now handled jointly by α_i and β_j, distributing the quantization burden across both dimensions.

Column scales as activation rescalings. Substituting equation 2 into $\mathbf{X}\mathbf{W}^\top$ yields

$$\mathbf{X}\mathbf{W}^\top \approx \left(\mathbf{X}\,\mathrm{Diag}(\boldsymbol{\beta})^{-1}\right)\hat{\mathbf{Q}}_{\boldsymbol{\alpha},\boldsymbol{\beta}}^\top\,\mathrm{Diag}(\boldsymbol{\alpha})^{-1} \;\triangleq\; \widetilde{\mathbf{X}}\,\widetilde{\mathbf{W}}^\top, \quad (3)$$

where $\widetilde{\mathbf{X}} = \mathbf{X}\,\mathrm{Diag}(\boldsymbol{\beta})^{-1} \in \mathbb{R}^{b\times n}$ and $\widetilde{\mathbf{W}}^\top = \hat{\mathbf{Q}}_{\boldsymbol{\alpha},\boldsymbol{\beta}}^\top\,\mathrm{Diag}(\boldsymbol{\alpha})^{-1} \in \mathbb{R}^{n\times m}$. The quantity quantized at inference is not X but the rescaled activation $\widetilde{X}$, whose per-channel statistics are reshaped by $\boldsymbol{\beta}$. The column scales therefore play two simultaneous roles: reduce weight quantization error through equation 2 and rescale activations through equation 3. This resembles the per-channel rescaling of SmoothQuant, but with reversed information flow: SmoothQuant derives its scales from activation statistics on a calibration dataset, whereas $\boldsymbol{\beta}$ is obtained from the weight matrix alone.

This weight-derived rescaling benefits activation quantization; it is not merely an algebraic by-product. Intuitively, the output of a linear layer at position (b, k) is $\sum_j X_{bj}W_{kj}$, so channels with large $\|W_{:,j}\|_2$ contribute disproportionately to the output, and these are empirically the channels where activation outliers concentrate (Sun et al., 2024; Dettmers et al., 2022; Bondarenko et al., 2021). Since $\boldsymbol{\beta}$ is optimized to reduce weight quantization error, it attenuates these same high-magnitude channels using the weights alone. Its mechanism differs from calibration: rather than reproducing the per-channel statistics that methods such as SmoothQuant estimate from activation data, $\boldsymbol{\beta}$ has a direct, causal effect on activation quantization. Note that the correlation between β and such statistics is heterogeneous and projection-dependent, as shown in Table 5 in Section 5.3. With the quantized weights held fixed, applying $\boldsymbol{\beta}$ to the activations measurably reduces their quantization error, without any calibration data (Appendix E isolates this by measuring per-projection activation quantization error with and without $\boldsymbol{\beta}$ under fixed quantized weights). Because this rescaling is applied inside the activation-quantization kernel, its runtime cost is negligible (Section 5.2).

4.3 MSE optimization via block coordinate descent

We optimize the row and column scales by minimizing the weight reconstruction MSE:

$$\min_{\boldsymbol{\alpha},\,\boldsymbol{\beta}} \;\left\|\mathbf{W} - \mathrm{Diag}(\boldsymbol{\alpha})^{-1}\,\hat{\mathbf{Q}}_{\boldsymbol{\alpha},\boldsymbol{\beta}}\,\mathrm{Diag}(\boldsymbol{\beta})^{-1}\right\|_F^2 \quad \text{s.t.} \quad \boldsymbol{\alpha}^{\odot-1} \in \mathbb{F}_\alpha^m,\;\; \boldsymbol{\beta}^{\odot-1} \in \mathbb{F}_\beta^n. \quad (4)$$

Note that the scales determine which grid points the quantizer assigns when forming $\hat{\mathbf{Q}}_{\boldsymbol{\alpha},\boldsymbol{\beta}}$ and simultaneously rescale the quantized output for reconstruction. Since grid-point selection is piecewise-constant, this joint dependence is *non-differentiable*.

Algorithm 1 DUALQUANT: Dual-Scale Quantization via Coordinate Descent

Require: Weight tile $\mathbf{W}$, quantizer $Q(\cdot)$, admissible sets $\mathbb{F}_\alpha^m$, $\mathbb{F}_\beta^n$, number of iterations T, initialization $(\boldsymbol{\alpha}_0, \boldsymbol{\beta}_0)$

$\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha}_0, \quad \boldsymbol{\beta} \leftarrow \boldsymbol{\beta}_0$
for $t = 1, \ldots, T$ **do**
  $\hat{\mathbf{Q}} \leftarrow Q(\mathrm{Diag}(\boldsymbol{\alpha})\, \mathbf{W}\, \mathrm{Diag}(\boldsymbol{\beta}))$
  $\mathbf{R} \leftarrow \hat{\mathbf{Q}}\, \mathrm{Diag}(\boldsymbol{\beta})^{-1}$
  $\boldsymbol{\alpha}_{\text{out}}^{\odot -1} \leftarrow \Pi_{\mathbb{F}_\alpha^m}\big((\mathbf{W} \odot \mathbf{R})\, \mathbf{1}_n \;\oslash\; \|\mathbf{R}\|_{2,\text{row}}^2\big)$ and $\alpha \leftarrow 1 \oslash \alpha_{\text{out}}^{\odot -1}$ ▷ Row scale update + project
  $\hat{\mathbf{Q}} \leftarrow Q(\mathrm{Diag}(\boldsymbol{\alpha})\, \mathbf{W}\, \mathrm{Diag}(\boldsymbol{\beta}))$
  $\mathbf{S} \leftarrow \mathrm{Diag}(\boldsymbol{\alpha})^{-1} \hat{\mathbf{Q}}$
  $\boldsymbol{\beta}_{\text{out}}^{\odot -1} \leftarrow \Pi_{\mathbb{F}_\beta^n}\big((\mathbf{W} \odot \mathbf{S})^\top \mathbf{1}_m \;\oslash\; \|\mathbf{S}\|_{2,\text{col}}^2\big)$ and $\beta \leftarrow 1 \oslash \beta_{\text{out}}^{\odot -1}$ ▷ Col scale update + project
end for
return $\boldsymbol{\alpha}_{\text{out}}^{\odot -1},\ \boldsymbol{\beta}_{\text{out}}^{\odot -1},\ Q(\mathrm{Diag}(\boldsymbol{\alpha})\, \mathbf{W}\, \mathrm{Diag}(\boldsymbol{\beta}))$

Notation: $\|\cdot\|_{2,\text{row}}^2$ and $\|\cdot\|_{2,\text{col}}^2$: squared ℓ_2-norm along each row and column; $\oslash$: element-wise division; $\odot$: element-wise multiplication; $(\cdot)^{\odot -1}$: element-wise inverse; $\Pi_{\mathbb{F}}$: projection to nearest value in $\mathbb{F}$.

Variable splitting. To tackle equation 4, we duplicate each scale into inner copies controlling grid assignment and outer copies controlling reconstruction (Boyd et al., 2011), yielding

$$\begin{aligned} \min_{\boldsymbol{\alpha}_{\text{in}}, \boldsymbol{\beta}_{\text{in}}, \boldsymbol{\alpha}_{\text{out}}, \boldsymbol{\beta}_{\text{out}}} \quad & \left\| \mathbf{W} - \mathrm{Diag}(\boldsymbol{\alpha}_{\text{out}})^{-1}\, \hat{\mathbf{Q}}_{\boldsymbol{\alpha}_{\text{in}}, \boldsymbol{\beta}_{\text{in}}}\, \mathrm{Diag}(\boldsymbol{\beta}_{\text{out}})^{-1} \right\|_F^2 \\ \text{s.t.} \quad & \boldsymbol{\alpha}_{\text{in}} = \boldsymbol{\alpha}_{\text{out}},\ \ \boldsymbol{\beta}_{\text{in}} = \boldsymbol{\beta}_{\text{out}},\ \ \boldsymbol{\alpha}_{\text{out}}^{\odot -1} \in \mathbb{F}_\alpha^m,\ \ \boldsymbol{\beta}_{\text{out}}^{\odot -1} \in \mathbb{F}_\beta^n. \end{aligned} \tag{5}$$

With the inner copies fixed, $\hat{\mathbf{Q}}_{\boldsymbol{\alpha}_{\text{in}}, \boldsymbol{\beta}_{\text{in}}}$ is a constant matrix and the objective becomes bilinear in $\boldsymbol{\alpha}_{\text{out}}^{\odot -1}$ and $\boldsymbol{\beta}_{\text{out}}^{\odot -1}$, decoupling into m independent scalar subproblems over $\boldsymbol{\alpha}$ and n over $\boldsymbol{\beta}$. A block coordinate descent algorithm is then used (Tseng, 2001; Razaviyayn et al., 2013) to alternate between updating the outer scales (in closed form) and re-synchronizing the inner copies (Appendix A.2).

Closed-form scale updates. Hereafter, we write $\hat{\mathbf{Q}}$ for $\hat{\mathbf{Q}}_{\boldsymbol{\alpha}_{\text{in}}, \boldsymbol{\beta}_{\text{in}}}$ for brevity. Defining $\mathbf{R}_{i,:} = \hat{\mathbf{Q}}_{i,:}\, \mathrm{Diag}(\boldsymbol{\beta}_{\text{out}})^{-1}$ and $\mathbf{S}_{:,j} = \mathrm{Diag}(\boldsymbol{\alpha}_{\text{out}})^{-1} \hat{\mathbf{Q}}_{:,j}$, the optimal outer scales are (Appendix B): $\alpha_i^* = \frac{\|\mathbf{R}_{i,:}\|_2^2}{\mathbf{W}_{i,:}\, \mathbf{R}_{i,:}^\top}$, and $\beta_j^* = \frac{\|\mathbf{S}_{:,j}\|_2^2}{\mathbf{S}_{:,j}^\top\, \mathbf{W}_{:,j}}$. Both updates have the same structure: the optimal scale is the ratio of the energy of the rescaled quantized slice to its inner product with the original weight slice. After each update, the scale is projected onto the admissible set by rounding α_i^{-1} (resp. β_j^{-1}) to the nearest value in $\mathbb{F}_\alpha^m$ (resp. $\mathbb{F}_\beta^n$). The BCD scheme cycles through: (i) update $\boldsymbol{\alpha}_{\text{out}}$, (ii) enforce consistency and re-quantize, (iii) update $\boldsymbol{\beta}_{\text{out}}$, (iv) enforce consistency and re-quantize, as summarized in Algorithm 1.

Convergence analysis We establish convergence under the following mild assumption.

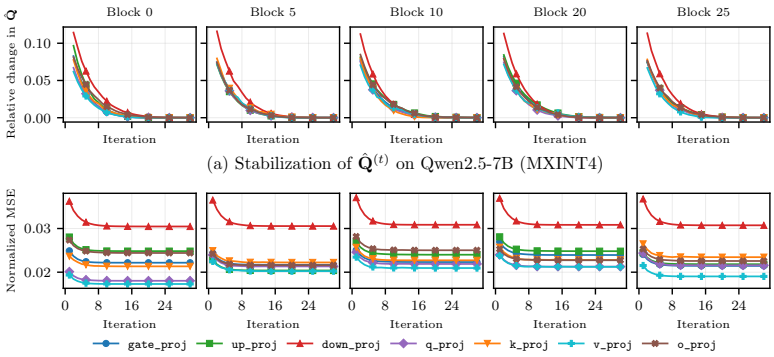
Assumption 1 (Non-degenerate quantization)**.** *For all iterations t and rows i, no row of $\mathrm{Diag}(\boldsymbol{\alpha}^{(t)})\mathbf{W}\mathrm{Diag}(\boldsymbol{\beta}^{(t)})$ is mapped entirely to zero by $Q(\cdot)$. This holds generically since LLM weight rows follow near-Gaussian distributions with non-negligible ℓ_2 norm (Dettmers et al., 2023). In the degenerate case $\hat{\mathbf{Q}}_{i,:} = \mathbf{0}$, the scale α_i does not affect the objective and is set to 1 by convention.*

Theorem 1 (Convergence of DUALQUANT; proof in Appendix C)**.** *Let Assumption 1 hold, and let $\{(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\beta}^{(t)})\}_{t \geq 1}$ be the sequence generated by Algorithm 1. Suppose $\hat{\mathbf{Q}}^{(t)}$ stabilizes at some finite iteration T, i.e., $\hat{\mathbf{Q}}^{(t)} = \hat{\mathbf{Q}}^{(T)}$ for all $t \geq T$. Then:*

(i) ***Descent guarantee.*** $\mathrm{MSE}_{\mathbf{W}}(\boldsymbol{\alpha}^{(t+1)}, \boldsymbol{\beta}^{(t+1)}) \leq \mathrm{MSE}_{\mathbf{W}}(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\beta}^{(t)})$ *for all $t \geq T$.*

(ii) ***Stationarity.*** *Every limit point $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ is a coordinate-wise stationary point of equation 4, satisfying $\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} \mathrm{MSE}_{\mathbf{W}}(\boldsymbol{\alpha}, \boldsymbol{\beta}^*)$ and $\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} \mathrm{MSE}_{\mathbf{W}}(\boldsymbol{\alpha}^*, \boldsymbol{\beta})$.*

The stabilization condition on $\hat{\mathbf{Q}}$ is empirically supported by Figure 2(a), where its relative change decays to near zero within approximately $T \approx 15$ iterations for all layers and blocks. Furthermore, Figure 2(b) confirms Theorem 1(i): the MSE decreases monotonically and plateaus within a finite number of iterations.



(a) Stabilization of $\hat{\mathbf{Q}}^{(t)}$ on Qwen2.5-7B (MXINT4)

(b) Convergence of the DUALQUANT algorithm on Qwen2.5-7B (MXINT4)

Figure 2: **(a)** Relative change $\|\hat{\mathbf{Q}}^{(t)}-\hat{\mathbf{Q}}^{(t-1)}\|_F/\|\hat{\mathbf{Q}}^{(t-1)}\|_F$ vs. BCD iteration for five transformer blocks. The change decays to near zero by $t\approx16$ for all layers and blocks, validating the stabilization condition of Theorem 1. **(b)** Normalized MSE $\|\mathbf{W}-\hat{\mathbf{W}}\|_F^2/\|\mathbf{W}\|_F^2$ vs. BCD iteration for the same blocks. All curves decrease monotonically and plateau within ≈10 iterations, confirming Theorem 1(i).

4.4 INFERENCE-TIME PROCEDURE AND OVERHEAD

The offline quantization procedure yields three artifacts per linear layer: the quantized weight matrix $\hat{\mathbf{Q}}$, row scales $\boldsymbol{\alpha}$, and column scales $\boldsymbol{\beta}$. At inference, the layer computes $Q(\widetilde{\mathbf{X}})\,\hat{\mathbf{Q}}^\top\,\mathrm{Diag}(\boldsymbol{\alpha})^{-1}$, where $\widetilde{\mathbf{X}}=\mathbf{X}\,\mathrm{Diag}(\boldsymbol{\beta})^{-1}$, and $Q(\widetilde{\mathbf{X}})$ denotes the format-native row-wise quantization of the rescaled activations. The row scale $\boldsymbol{\alpha}$ is absorbed into the per-row dequantization scale of $\hat{\mathbf{Q}}$ at quantization time and incurs no runtime cost. The column scale $\boldsymbol{\beta}$ introduces a per-channel rescaling of the activations *before* quantization. It does not replace the format-native activation scale: $Q(\cdot)$ still computes its own per-row scale on $\widetilde{\mathbf{X}}$ as usual. The two operate at different granularities: $\boldsymbol{\beta}$ attenuates outlier channels globally, while the format-native scale captures the residual per-token dynamic range. Unlike SmoothQuant and AWQ, which share a single scale across projections and fold it offline, DUALQUANT optimizes $\boldsymbol{\beta}$ independently per layer, which allows each layer to reach its own optimum. The resulting per-channel multiply is applied inside the activation-quantization kernel without a separate kernel launch (Section 5.2).

5 EXPERIMENTS

Models and formats We evaluate two model families: Qwen (Qwen2.5-7B, Qwen3-0.6B, Qwen3-14B) and Llama (Llama-2-7B, Llama-3.1-8B, Llama-3.2-1B), covering small, medium, and large scales across two architectures. We target 4-bit quantization formats spanning the major families of modern low-bit deployment: INT4 (block size $G=64$), MXINT4 ($G=32$), MXFP4 ($G=32$), and NVFP4 ($G=16$). INT4 is the standard integer baseline (Frantar et al., 2023; Lin et al., 2024). For all formats, the obtained scales are constrained to $\mathbb{F}$ as described in Section 4.3. The corresponding licenses for all models are included in Appendix J.

Evaluation setup We evaluate DUALQUANT across a range of tasks capturing different aspects of model performance: *language modeling ability* via perplexity on WikiText-2 (Merity et al., 2017) and C4 (Raffel et al., 2020) (test splits), *commonsense reasoning* and *language understanding* via normalized zero-shot accuracy on MMLU (Hendrycks et al., 2021), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), ARC-Easy (Clark et al., 2018), and WinoGrande (Sakaguchi et al., 2021). The corresponding licenses are included in Appendix J. All methods requiring calibration data (AWQ, SmoothQuant) use 256 randomly sampled sequences of length 8192 from the WikiText-2 training split; calibration and evaluation sets are strictly disjoint. We quantize all linear projections in every transformer block; `lm_head` is kept in full precision. All fake-quantization experiments are conducted on NVIDIA GH200 80 GB GPUs, with weights dequantized to FP16/BF16 during inference. Inference efficiency is evaluated separately using the real packed W4A4 implementation on a single NVIDIA RTX 4090 with 24 GB of memory (Section 5.2). Weight-only results and 8-bit W+A results are provided in Appendix H and Appendix G.2, respectively.

Table 1: WikiText-2 and C4 perplexity (↓) for joint W+A quantization at 4-bit precision on the Llama family. DUALQUANT applies β-rescaling to activations without calibration data. SmoothQuant and AWQ+ use activation statistics from 256 calibration samples. SINQ+ is our extension of SINQ to W+A. QuaRot rotates activations/weights online with a Hadamard transform (no calibration). **Bold**: best calibration-free (**CF**) result. (†): degenerate result ($PPL > 1000$); method incompatible with format. ✓: calibration-free. ✗: requires calibration data.

Bits	Format	Method	CF	WikiText-2 PPL ↓			C4 PPL ↓		
				Llama-2-7B	Llama-3.1-8B	Llama-3.2-1B	Llama-2-7B	Llama-3.1-8B	Llama-3.2-1B
BF16 (reference)			–	5.11	5.62	8.67	7.04	8.82	13.53
4-bit	A: INT4 W: INT4 (G=64)	RTN	✓	5.73	7.11	12.06	7.97	11.99	19.69
		AWQ+	✗	5.57	6.83	12.46	7.73	11.51	20.24
		SmoothQuant	✗	5.68	6.98	13.37	7.91	11.75	21.99
		SINQ+	✓	5.61	6.83	12.36	7.80	11.45	20.68
		QuaRot	✓	6.20	7.05	12.67	8.89	12.10	20.96
		DUALQUANT	✓	**5.56**	**6.73**	**11.88**	**7.71**	**11.35**	**19.23**
	A: MXINT4 W: MXINT4 (G=32)	RTN	✓	5.97	7.63	17.43	8.28	12.92	29.72
		AWQ+	✗	5.74	7.28	17.18	7.97	12.28	28.17
		SmoothQuant	✗	5.91	7.48	15.79	8.31	12.60	26.86
		SINQ+	✓	8.58	20.87	174.71	14.33	34.40	261.27
		QuaRot	✓	12.20	8.76	21.16	17.35	15.36	35.50
		DUALQUANT	✓	**5.76**	**7.15**	**13.44**	**7.94**	**11.98**	**21.86**
	A: MXFP4 W: MXFP4 (G=32)	RTN	✓	7.29	7.62	14.35	11.45	12.84	23.76
		AWQ+	✗	6.44	7.82	13.57	9.10	12.94	22.26
		SmoothQuant	✗	6.77	7.44	13.94	10.67	12.51	23.11
		SINQ+	✓	32.88	52.76	135.57	71.64	62.52	194.51
		QuaRot	✓	11.04	8.27	15.91	14.63	14.00	26.76
		DUALQUANT	✓	**6.38**	**7.35**	**13.13**	**9.70**	**12.70**	**21.66**
	A: NVFP4 W: NVFP4 (G=16)	RTN	✓	5.70	9.04	14.10	7.92	16.17	23.56
		AWQ+	✗	6.51	22.79	15.90	9.53	41.31	26.33
		SmoothQuant	✗	5.53	7.32	12.40	7.63	12.30	20.26
		SINQ+	✓	Div†	Div†	Div†	Div†	Div†	Div†
		QuaRot	✓	Div†	Div†	Div†	Div†	Div†	Div†
		DUALQUANT	✓	**5.54**	**6.83**	**10.91**	**7.64**	**11.53**	**17.45**

5.1 JOINT WEIGHT-AND-ACTIVATION QUANTIZATION

Tables 1 (Llama family, main text) and 7 (Qwen family, Appendix G) compare DUALQUANT against five baselines: **RTN**, which applies round-to-nearest quantization to both weights and activations; **AWQ+**, which extends AWQ (Lin et al., 2024) with RTN activation quantization; **SmoothQuant** (Xiao et al., 2023), which uses calibration-based per-channel activation rescaling; **SINQ+**, our extension of SINQ (Müller et al., 2026) (see Appendix D.2); and **QuaRot** (Ashkboos et al., 2024), a calibration-free approach that applies rotation matrices to weights and activations to suppress outliers prior to quantization, followed by RTN activation quantization. Additional details on baselines are provided in Appendix F.

Table 1 reports joint W+A quantization perplexity on the Llama family across four 4-bit formats. DUALQUANT consistently matches or surpasses the calibration-based SmoothQuant, while requiring no calibration data. On MXFP4, DUALQUANT improves over SmoothQuant on WikiText-2 for all three models, substantially on Llama-2-7B and Llama-3.2-1B and narrowly on Llama-3.1-8B; on C4 the same holds except for Llama-3.1-8B, where DUALQUANT trails slightly. On MXINT4, DUALQUANT is better than SmoothQuant across both WikiText-2 and C4 for all three models. On INT4, DUALQUANT surpasses both SmoothQuant and AWQ+ across all models, demonstrating that weight-derived scales provide effective activation rescaling even on the standard integer format. On NVFP4, DUALQUANT achieves the strongest results among all methods on Llama-3.1-8B and Llama-3.2-1B, with the same overall trend on C4. In contrast, both calibration-free baselines degrade substantially on MX and NV formats: SINQ+ degrades on MXINT4 and MXFP4 and diverges completely on NVFP4, as its format-unaware scales fall outside the FP8 representable range; QuaRot similarly diverges on NVFP4 and degrades substantially on MX formats, as its Hadamard rotation redistributes activation magnitude in a way incompatible with block floating-point scaling. DUALQUANT avoids both failure modes by design, incorporating format constraints directly into the optimization and requiring no rotation or calibration data. Corresponding results on the Qwen family (Table 7, Appendix G) show consistent trends.[1]

Table 2 reports zero-shot accuracy on MMLU, HellaSwag, PIQA, ARC-Easy, and WinoGrande for Llama-3.1-8B across all four W+A quantization formats. Corresponding results for the Qwen3-14B

[1]The DUALQUANT–SINQ+ gap persists without our extensions: in SINQ's native weight-only INT4 setting, DUALQUANT is more accurate on three of four models (Appendix H), so it stems from the *objective* itself.

Table 2: Llama-3.1-8B downstream accuracy (0-shot, %, higher is better ↑) for RTN, AWQ+, SmoothQuant, SINQ+, DUALQUANT, and QuaRot, across all four evaluated formats; QuaRot is omitted entirely for NVFP4, since QuaRot's Hadamard rotation redistributes activation magnitude in a way that collapses NVFP4's narrow-range E4M3 block scale. **Bold**: best calibration-free (CF) result per column within each model+format block. ✓: calibration-free. ✗: requires calibration data.

Llama-3.1-8B							
Format	Method	CF	MMLU	HellaSwag	PIQA	ARC-Easy	WinoGrande
INT4 (G=64)	RTN	✓	54.89	56.44	**77.86**	77.40	68.90
	AWQ+	✗	55.78	56.65	78.51	77.19	69.06
	SmoothQuant	✗	55.24	55.93	77.31	77.44	68.75
	SINQ+	✓	55.50	56.68	77.09	77.65	67.72
	QuaRot	✓	54.21	56.16	77.04	77.40	**70.17**
	DUALQUANT	✓	**55.61**	**56.83**	77.37	**77.68**	69.77
MXINT4 (G=32)	RTN	✓	50.94	55.06	76.39	74.79	68.27
	AWQ+	✗	52.30	55.31	76.77	73.82	69.61
	SmoothQuant	✗	52.93	54.83	76.61	75.88	66.14
	SINQ+	✓	30.32	42.41	68.55	63.26	58.33
	QuaRot	✓	43.53	50.16	74.76	71.17	67.48
	DUALQUANT	✓	**53.53**	**55.28**	**77.37**	**76.89**	**70.32**
MXFP4 (G=32)	RTN	✓	51.51	54.17	76.71	76.43	68.03
	AWQ+	✗	49.85	52.75	76.39	75.34	68.11
	SmoothQuant	✗	51.60	54.11	76.12	73.86	70.09
	SINQ+	✓	25.36	37.60	64.36	54.21	55.64
	QuaRot	✓	48.69	52.81	75.84	**77.19**	68.90
	DUALQUANT	✓	**52.10**	**54.66**	**76.77**	76.22	**69.69**
NVFP4 (G=16)	RTN	✓	43.87	53.02	74.81	71.80	65.67
	AWQ+	✗	31.19	45.10	71.06	63.17	62.75
	SmoothQuant	✗	55.67	55.60	77.53	75.80	68.90
	SINQ+	✓	25.51	25.41	53.43	24.62	49.17
	DUALQUANT	✓	**57.76**	**56.65**	**78.24**	**77.90**	**71.67**

are reported in Table 8 in Appendix G. DUALQUANT achieves the best calibration-free accuracy across nearly all settings. On NVFP4, DUALQUANT surpasses every baseline, including the calibration-based SmoothQuant, across all five tasks, while both SINQ+ and QuaRot diverge completely on NVFP4, making DUALQUANT by far the strongest calibration-free method on this format. On MXINT4 and MXFP4, DUALQUANT substantially outperforms both calibration-free baselines on most tasks: SINQ+ collapses toward random-guess level, while QuaRot also degrades significantly. On INT4, where all methods remain viable, DUALQUANT achieves the best calibration-free accuracy on MMLU, HellaSwag, and ARC-Easy.

5.2 RUNTIME AND MEMORY ANALYSIS

Offline quantization cost The optimization that produces $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ operates entirely on the static weight matrix with no calibration forward passes, no Hessian, and no inner optimization loop. Each coordinate update is a single closed-form expression, making the offline cost comparable to RTN and orders of magnitude cheaper than GPTQ or AWQ.

Real packed W4A4 deployment. To verify that fake-quantization results translate to actual deployment, we implement a packed W4A4 inference path for Llama-3.1-8B (Grattafiori et al., 2024) on a single NVIDIA RTX 4090 (NVIDIA Corporation, 2022). The implementation uses INT4 with group size $G{=}64$ and is built on the Nunchaku/SVDQuant (Li et al., 2025) W4A4 GEMM backend (Appendix I). Accuracy, memory usage, and latency are measured directly on the packed deployment path *without* fake quantization. WikiText-2 perplexity is 6.74, closely matching the fake-quantization result of 6.73 (Table 1) with an absolute difference of 0.01. This agreement confirms that the fake-quantization results in Section 5 accurately reflect the deployed model.

Inference and memory overhead. The row scale α is folded into the native group scale and introduces no runtime overhead. The column scale β is incorporated into the activation-quantization kernel without an additional kernel launch, while weights remain packed throughout inference. Thus, activation quantization, including the application of β, constitutes the only per-step quantization overhead, accounting for $4.3{-}6.1\%$ of total W4A4 GPU time (Table 11). Despite this overhead, DUALQUANT W4A4 achieves $1.39{-}2.46\times$ end-to-end speedup over BF16 across prefill and decode at batch sizes 1 and 16 (Table 3). For Llama-3.1-8B relative to BF16, the packed deployment reduces

Table 3: End-to-end memory usage and throughput of the real packed W4A4 deployment versus BF16 for Llama-3.1-8B on an RTX 4090 with INT4, G=64. Prefill uses an 82-token prompt without a KV cache. Decode generates 100 tokens with a 182-token KV cache. Memory includes weights, scales, unquantized embedding and lm_head, activations, and the KV cache for decode.

Workload	Memory (GB) BF16	Memory (GB) W4A4	Memory Reduction	Throughput (tok/s) BF16	Throughput (tok/s) W4A4	Speedup
Prefill (b=1)	16.33	6.25	2.61×	3904.6	6421.4	1.64×
Prefill (b=16)	16.66	6.58	2.53×	8940.3	22003.4	2.46×
Decode (b=1)	16.34	6.26	2.61×	56.6	83.7	1.48×
Decode (b=16)	16.75	6.67	2.51×	694.5	964.1	1.39×

Table 4: Scale ablation for W+A quantization (WikiText-2 PPL ↓). **Bold**: best per column. $\boldsymbol{\alpha}$-only fixes $\boldsymbol{\beta}=\mathbf{1}_n$; $\boldsymbol{\beta}$-only fixes $\boldsymbol{\alpha}=\mathbf{1}_m$.

Variant	Qwen2.5-7B		Llama-3.1-8B	
	MXFP4	NVFP4	MXFP4	NVFP4
RTN (no scales)	7.78	7.07	7.60	9.04
$\boldsymbol{\alpha}$-only	7.78	7.05	7.60	8.27
$\boldsymbol{\beta}$-only	7.58	6.88	7.46	7.79
Both ($\boldsymbol{\alpha}+\boldsymbol{\beta}$)	**7.45**	**6.53**	**7.35**	**6.83**

Table 5: Per-layer Pearson correlation between DUALQUANT's β and SmoothQuant's calibrated scale s, on Llama-3.1-8B.

Projection	λ=0	λ=0.5	λ=1.0
q_proj	+0.96	+0.09	−0.41
k_proj	+0.97	+0.06	−0.57
v_proj	+0.93	+0.53	+0.07
gate_proj	+0.91	+0.55	−0.34
up_proj	+0.83	+0.41	−0.28

the weight footprint from 16.20 GB to 5.92 GB and end-to-end memory usage by 2.51–2.61× (Table 3). Component-level timing and memory breakdowns are provided in Appendix I (Table 12).

5.3 ABLATION STUDIES

Row scales, column scales, or both Table 4 compares three variants of DUALQUANT on MXFP4 and NVFP4 W+A quantization. The **$\boldsymbol{\alpha}$-only** variant improves over RTN primarily on NVFP4, whose finer FP8 scale grid offers more degrees of freedom than MXFP4's coarser E8M0 grid (e.g., 9.04 → 8.27 on Llama-3.1-8B), while the **$\boldsymbol{\beta}$-only** variant yields consistently larger gains by simultaneously reducing column-wise weight quantization error and rescaling activations via the factorization of Section 4.2. Combining both yields the best results in all cases with roughly additive improvement, confirming that row and column scales target complementary sources of error.

Hyperparameter selection DUALQUANT has two hyperparameters: BCD iteration, T, and initialization $(\boldsymbol{\alpha}_0, \boldsymbol{\beta}_0)$. The monotone-descent objective guarantees convergence from any starting point, but a good initialization reduces the iterations needed. We initialize $\alpha_{0,i} = \max_j |W_{ij}|$ (format-native absmax) and $\beta_{0,j} = \sum_i |W_{ij}|$ (per-column ℓ_1-norm). Considering Figure 2, we use T=15, which we found sufficient across the evaluated settings.

Correlation between the scales of DUALQUANT and SmoothQuant To relate DUALQUANT's weight-derived rescaling β to SmoothQuant's calibration-derived scale, we compute the per-layer Pearson correlation (Pearson, 1895) across $\lambda \in \{0, 0.5, 1.0\}$, the migration strength in SmoothQuant's scale equation, on Llama-3.1-8B (Table 5). At $\lambda = 0$ (purely weight-derived s), β aligns near-perfectly with SmoothQuant's scale; correlation remains positive across most projections at $\lambda = 0.5$ and decreases monotonically toward $\lambda = 1.0$ (activation-dominated s), as expected for a weight-only method. Correlation is descriptive and does not imply equivalent quantization behavior.

6 CONCLUSION

We present DUALQUANT, a calibration-free PTQ framework that jointly quantizes weights and activations by minimizing weight-reconstruction MSE via BCD with closed-form updates and provable convergence, operating across INT4, the MX family, and NVFP4. Its weight-driven column scales double as activation rescalings, so on Llama and Qwen DUALQUANT matches or surpasses calibration-based SmoothQuant and clearly outperforms calibration-free QuaRot and SINQ+, without any calibration data. A packed W4A4 path for Llama-3.1-8B further delivers up to 2.61× lower memory use and 2.46× higher throughput than BF16 on an RTX 4090.

AI USE STATEMENT

In this work, we used generative AI tools for polishing and refining manuscript text for clarity and readability; formatting LaTeX tables, equations, and generating Python scripts to produce plots and figures from simulation results. We also used generative AI tools for limited assistance in developing the packed W4A4 CUDA kernel (Appendix I), with all kernel logic, correctness verification, and performance validation performed by the authors. We have **not** used generative AI tools for formulating mathematical claims or problem statements; deriving closed-form scale updates (Appendix B); writing or verifying convergence proofs (Appendix C); designing the algorithm, methodology, or experiments; implementing the core quantization pipeline, BCD solver, or evaluation harness; or interpreting experimental results. These tasks were performed entirely by the authors. We have reviewed all AI-assisted work: all AI-suggested text revisions were checked by the authors for factual accuracy and consistency with the paper's technical claims and experimental results before inclusion, all AI-generated plotting scripts were verified to correctly reproduce the underlying simulation data, and all AI-assisted kernel code was reviewed, tested, and validated for correctness by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. It involves no human subjects, crowdsourcing, or personal data. All experiments use publicly available pre-trained models (Llama and Qwen families) and public benchmarks (WikiText-2, C4, HellaSwag, MMLU, PIQA, ARC-Easy, WinoGrande) under their respective licenses (Appendix J), for academic research only; we do not redistribute model weights or datasets. DUALQUANT is a compression method: it adds no capabilities to a model, but by removing the need for calibration data and reducing memory and compute at inference, it lowers the cost of deploying LLMs, including on consumer hardware. This broadens access and reduces energy per token, but equally lowers the barrier to misuse of the underlying models. Because DUALQUANT requires no data, it also enables quantization in settings where calibration data is proprietary or privacy-sensitive. Our evaluation covers perplexity and task accuracy only; we do not measure whether quantization affects safety alignment, bias, or refusal behavior, and quantized models should be re-evaluated for these properties before deployment.

REPRODUCIBILITY STATEMENT

The DUALQUANT algorithm is fully specified in Section 4 and Algorithm 1, with the closed-form scale updates derived in Appendix B and the convergence proof, including all assumptions, in Appendix C. DUALQUANT is deterministic and data-free: given a weight matrix, target format, iteration count, and initialization (Section 5.3), the resulting scales and quantized weights are uniquely determined. Models, formats, block sizes, scale constraints, and evaluation benchmarks are described in Section 5. Configurations for all baselines, including the shared calibration set, AWQ search settings, and SmoothQuant migration strength, are given in Appendix F and our extension of SINQ to W+A and MX/NV formats is detailed in Appendix D.2. The packed W4A4 implementation, measurement protocol, and component-level timing and memory breakdowns are in Section 5.2 and Appendix I. An anonymized repository containing the DUALQUANT implementation, baseline scripts, and the packed inference path is linked in the abstract and will be released publicly upon acceptance.

REFERENCES

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-free 4-bit inference in rotated LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Hicham Badri and Appu Shaji. Half-quadratic quantization of large machine learning models. `https://dropbox.github.io/hqq_blog/`, 2023.

Kayhan Behdin, Ayan Acharya, Aman Gupta, Qingquan Song, Siyu Zhu, Sathiya Keerthi, and Rahul Mazumder. QuantEase: Optimization-based quantization for language models. *arXiv preprint arXiv:2309.01885*, 2023.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pp. 7432–7439, 2020.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Understanding and overcoming the challenges of efficient transformer quantization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

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.

Patrik Czakó, Gábor Kertész, and Sándor Szénási. SmoothRot: Combining channel-wise scaling and rotation for quantization-friendly LLMs. *arXiv preprint arXiv:2506.05413*, 2025.

DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. LLM.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

Alireza Ghaffari, Sharareh Younesian, Vahid Partovi Nia, Boxing Chen, and Masoud Asgharian. AdpQ: A zero-shot calibration free adaptive post training quantization method for LLMs. *arXiv preprint arXiv:2405.13358*, 2024.

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.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Muyang Li, Yujun Lin, Zhekai Zhang, Tianle Cai, Xiuyu Li, Junxian Guo, Enze Xie, Chenlin Meng, Jun-Yan Zhu, and Song Han. SVDQuant: Absorbing outliers by low-rank components for 4-bit diffusion models. In *International Conference on Learning Representations (ICLR)*, 2025.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Conference on Machine Learning and Systems (MLSys)*, 2024.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. SpinQuant: LLM quantization with learned rotations. In *International Conference on Learning Representations (ICLR)*, 2025.

Yuexiao Ma, Huixia Li, Xiawu Zheng, Feng Ling, Xuefeng Xiao, Rui Wang, Shilei Wen, Fei Chao, and Rongrong Ji. AffineQuant: Affine transformation quantization for large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017.

Lorenz K. Müller, Philippe Bich, Jiawei Zhuang, Ahmet Celik, Luca Benfenati, and Lukas Cavigelli. SINQ: Sinkhorn-normalized quantization for calibration-free low-precision LLM weights. In *International Conference on Machine Learning (ICML)*, 2026.

NVIDIA. Introducing NVFP4 for efficient and accurate low-precision inference. `https://developer.nvidia.com/blog/introducing-nvfp4-for-efficient-and-accurate-low-precision-inference`, 2025.

NVIDIA Corporation. NVIDIA GeForce RTX 4090. `https://www.nvidia.com/en-us/geforce/graphics-cards/40-series/rtx-4090/`, 2022. Accessed: 2026-04-28.

Karl Pearson. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58:240–242, 1895. doi: 10.1098/rspl.1895.0041.

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.

Meisam Razaviyayn, Mingyi Hong, and Zhi-Quan Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23 (2):1126–1153, 2013.

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.

Utkarsh Saxena, Sayeh Sharify, Kaushik Roy, and Xin Wang. ResQ: Mixed-precision quantization of large language models with low-rank residuals. In *International Conference on Machine Learning (ICML)*, 2025.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. OmniQuant: Omnidirectionally calibrated quantization for large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The Annals of Mathematical Statistics*, 35(2):876–879, 1964. doi: 10.1214/aoms/1177703591.

Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967. doi: 10.2140/pjm.1967.21.343.

Mingjie Sun, Xinlei Chen, J. Zico Kolter, and Zhuang Liu. Massive activations in large language models. In *Conference on Language Modeling (COLM)*, 2024.

Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Jiaxin Hu, Xianzhi Yu, Lu Hou, Chun Yuan, Xin Jiang, et al. FlatQuant: Flatness matters for LLM quantization. In *International Conference on Machine Learning (ICML)*, 2025.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. QuIP#: Even better LLM quantization with Hadamard incoherence and lattice codebooks. In *International Conference on Machine Learning (ICML)*, 2024.

Paul Tseng. Applications of a splitting algorithm to decomposition in convex programming and variational inequalities. *SIAM Journal on Control and Optimization*, 29(1):119–138, 1991.

Paul Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001.

Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. Outlier suppression: Pushing the limit of low-bit transformer language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning (ICML)*, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. ZeroQuant: Efficient and affordable post-training quantization for large-scale transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Amir Zandieh, Majid Daliri, Majid Hadian, and Vahab Mirrokni. TurboQuant: Online vector quantization with near-optimal distortion rate. In *International Conference on Learning Representations (ICLR)*, 2026.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 4791–4800, 2019.

A Optimization background

The dual-scale optimization problem introduced in Section 4.3 is non-convex and involves a non-differentiable quantization operator $Q(\cdot)$, making direct gradient-based optimization infeasible. We address this by combining two classical techniques from optimization: *variable splitting*, which decouples the conflicting roles of the scale vectors, and *block coordinate descent*, which exploits the resulting structure to obtain closed-form updates. We briefly review both tools here.

A.1 Variable splitting

Variable splitting (Boyd et al., 2011; Tseng, 1991) is a technique for handling optimization problems where a single variable plays multiple conflicting roles in the objective or constraints. The idea is to introduce separate copies of the variable, one for each role, and enforce equality between the copies as an explicit constraint. This transforms a tightly coupled, difficult problem into a more structured one where each copy can be optimized independently given the others, subject to the coupling constraint.

In our dual-scale formulation in equation 4, each scale vector plays two distinct roles: as an *inner* scale applied to the argument of $Q(\cdot)$, which determines the quantization grid assignment, i.e., which discrete value each weight is mapped to, and as an *outer* scale applied after applying $Q(\cdot)$, which controls how those discrete values are reconstructed in full precision. These two roles are in tension: the inner scale must be chosen to minimize quantization error by aligning the weight distribution with the quantization grid, while the optimal outer scale, given a fixed quantized matrix, is determined by a least-squares criterion that is blind to the quantization grid itself. Because both roles are played by the same variable, the objective in equation 4 is non-differentiable and non-convex in $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ jointly.

Variable splitting resolves this conflict by introducing independent copies $(\boldsymbol{\alpha}_{\text{in}}, \boldsymbol{\alpha}_{\text{out}})$ and $(\boldsymbol{\beta}_{\text{in}}, \boldsymbol{\beta}_{\text{out}})$, one pair for each scaling axis (row and column), and enforcing the consistency constraints $\boldsymbol{\alpha}_{\text{in}} = \boldsymbol{\alpha}_{\text{out}}$ and $\boldsymbol{\beta}_{\text{in}} = \boldsymbol{\beta}_{\text{out}}$ explicitly. With the inner scales (and hence $\hat{\mathbf{Q}}_{\boldsymbol{\alpha}_{\text{in}},\boldsymbol{\beta}_{\text{in}}}$ held fixed,) the relaxed objective becomes bilinear in the outer scales $\boldsymbol{\alpha}_{\text{out}}^{-1}$ and $\boldsymbol{\beta}_{\text{out}}^{-1}$, and decomposes into independent scalar least-squares problems per row and per column, each with a closed-form solution as derived in Section 4.3. After each pair of closed-form outer scale updates, the consistency constraints are re-enforced by synchronizing the inner scales with the newly computed outer scales, after which $\hat{\mathbf{Q}}_{\boldsymbol{\alpha}_{\text{in}},\boldsymbol{\beta}_{\text{in}}}$ is recomputed before the next BCD iteration. For brevity, we drop the subscripts and write $\hat{\mathbf{Q}}$ hereafter. This two-step structure is what makes the overall algorithm tractable and efficient in practice. The iterative procedure for cycling through these updates is formalised as Block Coordinate Descent, described next.

A.2 Block coordinate descent

Block Coordinate Descent (BCD) (Tseng, 2001) is a classical iterative optimization strategy for problems involving multiple groups of variables. Rather than jointly optimizing all variables simultaneously, which may be intractable or computationally prohibitive, BCD partitions the variable set into blocks and cycles through them, optimizing each block exactly (or approximately) while holding the remaining blocks fixed. Formally, given an objective $f(\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_K)$, BCD performs the update

$$\mathbf{u}_k^{(t+1)} = \arg\min_{\mathbf{u}_k} \; f\Big(\mathbf{u}_1^{(t+1)}, \ldots, \mathbf{u}_{k-1}^{(t+1)}, \, \mathbf{u}_k, \, \mathbf{u}_{k+1}^{(t)}, \ldots, \mathbf{u}_K^{(t)}\Big) \tag{6}$$

for each block $k = 1, \ldots, K$ in sequence, using the most recently updated values of all other blocks. Under mild regularity conditions, in particular when each subproblem admits a unique minimizer, BCD is guaranteed to converge to a stationary point of the objective (Tseng, 2001).

The practical appeal of BCD lies in its ability to exploit problem structure: even when the joint problem is difficult, individual subproblems are often tractable. In our setting, once the inner scales and the quantized matrix $\hat{\mathbf{Q}}$ are held fixed, the objective in equation 5 decomposes into independent scalar least-squares subproblems for each row scale $\alpha_{\text{out},i}$ and each column scale $\beta_{\text{out},j}$, each of which admits a closed-form solution. This makes BCD a natural and efficient solver for the dual-scale problem.

B CLOSED-FORM OPTIMAL SOLUTIONS FOR SCALES

This appendix derives the closed-form updates via variable splitting. We introduce independent inner and outer copies of each scale vector. With inner scales $(\boldsymbol{\alpha}_{\text{in}}, \boldsymbol{\beta}_{\text{in}})$ fixed, the objective function is

$$\mathcal{L}(\boldsymbol{\alpha}_{\text{out}}, \boldsymbol{\beta}_{\text{out}}) = \left\| \mathbf{W} - \text{Diag}(\boldsymbol{\alpha}_{\text{out}})^{-1} \hat{\mathbf{Q}} \, \text{Diag}(\boldsymbol{\beta}_{\text{out}})^{-1} \right\|_F^2. \tag{7}$$

Closed-Form Update for $\boldsymbol{\alpha}_{\text{out}}$. With $\boldsymbol{\beta}_{\text{out}}$ and $\hat{\mathbf{Q}}$ fixed, we define the auxiliary matrix $\mathbf{R} = \hat{\mathbf{Q}} \, \text{Diag}(\boldsymbol{\beta}_{\text{out}})^{-1}$, which absorbs the fixed column scaling into the quantized values. The objective in equation 5 then reduces to

$$\text{MSE}_{\mathbf{W}}(\boldsymbol{\alpha}_{\text{out}}) = \left\| \mathbf{W} - \text{Diag}(\boldsymbol{\alpha}_{\text{out}})^{-1} \mathbf{R} \right\|_F^2 = \sum_{i=1}^{m} \left\| \mathbf{W}_{i,:} - \frac{1}{\alpha_{\text{out},i}} \mathbf{R}_{i,:} \right\|_2^2. \tag{8}$$

This objective decomposes into m independent scalar subproblems, one per row, each involving only the scalar $\alpha_{\text{out},i}$. To cast each subproblem as a standard least-squares problem, we substitute $u_i = \alpha_{\text{out},i}^{-1}$, which linearizes the dependence on the scale. Setting the derivative with respect to u_i to zero:

$$\frac{\partial \, \text{MSE}_{\mathbf{W}}}{\partial u_i} = -2 \, \mathbf{R}_{i,:} (\mathbf{W}_{i,:} - u_i \, \mathbf{R}_{i,:})^\top = 0, \tag{9}$$

and solving for u_i gives

$$u_i^* = \frac{\mathbf{W}_{i,:} \, \mathbf{R}_{i,:}^\top}{\|\mathbf{R}_{i,:}\|_2^2}. \tag{10}$$

Inverting back to the original parametrization yields the closed-form update for the i-th row scale:

$$\alpha_{\text{out},i}^* = \frac{\|\mathbf{R}_{i,:}\|_2^2}{\mathbf{W}_{i,:} \, \mathbf{R}_{i,:}^\top}, \quad \text{where} \quad \mathbf{R}_{i,:} = \hat{\mathbf{Q}}_{i,:} \, \text{Diag}(\boldsymbol{\beta}_{\text{out}})^{-1}. \tag{11}$$

Intuitively, $\alpha_{\text{out},i}^*$ scales the i-th row so that the quantized row $\mathbf{R}_{i,:}$ best approximates the original row $\mathbf{W}_{i,:}$ in the least-squares sense: it is large when the quantized row underestimates the original, and small when it overestimates. After computing $\boldsymbol{\alpha}_{\text{out}}^*$, the format constraint $\boldsymbol{\alpha}_{\text{out}}^{\odot -1} \in \mathbb{F}_\alpha^m$ is enforced by projecting each $1/\alpha_{\text{out},i}^*$ to the nearest representable value in $\mathbb{F}_\alpha$.

Closed-Form Update for $\boldsymbol{\beta}_{\text{out}}$. The derivation for the column scales follows the same structure. With $\boldsymbol{\alpha}_{\text{out}}$ and $\hat{\mathbf{Q}}$ fixed, we define $\mathbf{S} = \text{Diag}(\boldsymbol{\alpha}_{\text{out}})^{-1} \hat{\mathbf{Q}}$, which absorbs the fixed row scaling into the quantized values. The objective reduces to

$$\text{MSE}_{\mathbf{W}}(\boldsymbol{\beta}_{\text{out}}) = \left\| \mathbf{W} - \mathbf{S} \, \text{Diag}(\boldsymbol{\beta}_{\text{out}})^{-1} \right\|_F^2 = \sum_{j=1}^{n} \left\| \mathbf{W}_{:,j} - \frac{1}{\beta_{\text{out},j}} \mathbf{S}_{:,j} \right\|_2^2, \tag{12}$$

which decomposes into n independent scalar subproblems, one per column. Substituting $v_j = \beta_{\text{out},j}^{-1}$ and setting the derivative with respect to v_j to zero:

$$\frac{\partial \, \text{MSE}_{\mathbf{W}}}{\partial v_j} = -2 \, \mathbf{S}_{:,j}^\top (\mathbf{W}_{:,j} - v_j \, \mathbf{S}_{:,j}) = 0, \tag{13}$$

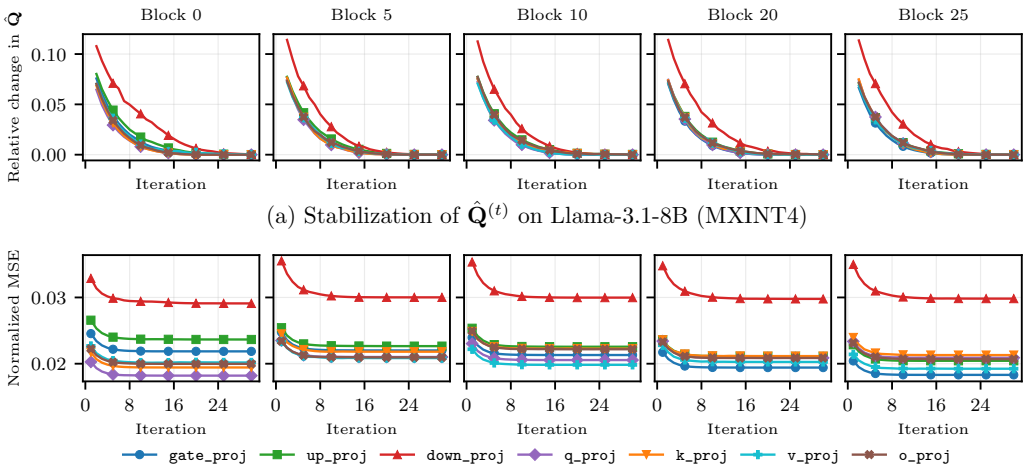
and solving for v_j gives

$$v_j^* = \frac{\mathbf{S}_{:,j}^\top \, \mathbf{W}_{:,j}}{\|\mathbf{S}_{:,j}\|_2^2}. \tag{14}$$

Inverting back yields the closed-form update for the j-th column scale:

$$\beta_{\text{out},j}^* = \frac{\|\mathbf{S}_{:,j}\|_2^2}{\mathbf{S}_{:,j}^\top \, \mathbf{W}_{:,j}}, \quad \text{where} \quad \mathbf{S}_{:,j} = \text{Diag}(\boldsymbol{\alpha}_{\text{out}})^{-1} \hat{\mathbf{Q}}_{:,j}. \tag{15}$$

As with the row scales, the format constraint $\boldsymbol{\beta}_{\text{out}}^{\odot -1} \in \mathbb{F}_\beta^n$ is enforced by projecting each $1/\beta_{\text{out},j}^*$ to the nearest representable value in $\mathbb{F}_\beta$ after the update.



(a) Stabilization of $\hat{\mathbf{Q}}^{(t)}$ on Llama-3.1-8B (MXINT4)

(b) Convergence of the DUALQUANT algorithm on Llama-3.1-8B (MXINT4)

Figure 3: **(a)** Relative change $\|\hat{\mathbf{Q}}^{(t)}-\hat{\mathbf{Q}}^{(t-1)}\|_F/\|\hat{\mathbf{Q}}^{(t-1)}\|_F$ vs. BCD iteration for five transformer blocks. The $\hat{\mathbf{Q}}^{(t)}$ change decays to near zero by $t{\approx}16$ for all layers and blocks, validating the stabilization condition of Theorem 1. **(b)** Normalized MSE $\|\mathbf{W}-\hat{\mathbf{W}}\|_F^2/\|\mathbf{W}\|_F^2$ vs. BCD iteration for the same blocks. All curves decrease monotonically and plateau within $\approx$10 iterations, confirming Theorem 1(i). Together with the Qwen2.5-7B results in Figure 2, this illustrates that the convergence behavior of DUALQUANT is consistent across the two model families evaluated here. Notably, `down_proj` consistently shows the highest residual MSE, consistent with it being among the harder projections to quantize (Frantar et al., 2023).

C PROOF OF THEOREM 1.

We first establish the uniqueness of the per-block subproblem solutions (Lemmas 1 and 2), then prove the theorem. For brevity, define

$$g(\boldsymbol{\alpha},\boldsymbol{\beta}) := \left\|\mathbf{W} - \mathrm{Diag}(\boldsymbol{\alpha})^{-1}\,\hat{\mathbf{Q}}^{(T)}\,\mathrm{Diag}(\boldsymbol{\beta})^{-1}\right\|_F^2, \tag{16}$$

the objective equation 4 with $\hat{\mathbf{Q}}$ *frozen* at its post-stabilization value $\hat{\mathbf{Q}}^{(T)}$. By the stabilization hypothesis, as empirically verified in Figure 2 (main text) and Figure 3 (appendix), $\hat{\mathbf{Q}}^{(t)} = \hat{\mathbf{Q}}^{(T)}$ for all $t \geq T$, so for $t \geq T$ we have $g(\boldsymbol{\alpha}^{(t)},\boldsymbol{\beta}^{(t)}) = \mathrm{MSE}_{\mathbf{W}}(\boldsymbol{\alpha}^{(t)},\boldsymbol{\beta}^{(t)})$. For $t < T$, g and $\mathrm{MSE}_{\mathbf{W}}$ differ in general because $\hat{\mathbf{Q}}^{(t)} \neq \hat{\mathbf{Q}}^{(T)}$. Working with the function g in the post-stabilization regime allows us to invoke BCD convergence theorem, which requires a stationary objective.

Lemma 1 (Unique row-scale minimizer)**.** *Under Assumption 1, for any fixed $\boldsymbol{\beta} \in \mathbb{R}_+^n$, the subproblem $\min_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha},\boldsymbol{\beta})$ admits a unique minimizer $\boldsymbol{\alpha}^* \in \mathbb{R}_+^m$.*

Proof. The objective decomposes across rows as $g(\boldsymbol{\alpha},\boldsymbol{\beta}) = \sum_{i=1}^m \|\alpha_i^{-1}\mathbf{R}_{i,:} - \mathbf{W}_{i,:}\|_2^2$, where $\mathbf{R}_{i,:} = \hat{\mathbf{Q}}_{i,:}^{(T)}\mathrm{Diag}(\boldsymbol{\beta})^{-1}$. Substituting $u_i = \alpha_i^{-1}$ yields a strictly convex quadratic in u_i with unique minimizer $u_i^* = \mathbf{W}_{i,:}\,\mathbf{R}_{i,:}^\top/\|\mathbf{R}_{i,:}\|_2^2$, which is positive under Assumption 1. Specifically, Assumption 1 ensures $\|\mathbf{R}_{i,:}\|_2^2 > 0$ (non-zero denominator). The numerator $\mathbf{W}_{i,:}\,\mathbf{R}_{i,:}^\top$ is positive because all quantization formats considered use sign-preserving quantizers, meaning $Q(x)$ and x share the same sign for any nonzero x. Since $\alpha_i, \beta_j > 0$, it follows that $W_{i,k}$ and $R_{i,k}$ share the same sign for all k, making the inner product strictly positive. Therefore $\alpha_i^* = 1/u_i^* \in \mathbb{R}_+$ is the unique minimizer. □

Lemma 2 (Unique column-scale minimizer)**.** *Under Assumption 1, for any fixed $\boldsymbol{\alpha} \in \mathbb{R}_+^m$, the subproblem $\min_{\boldsymbol{\beta}} g(\boldsymbol{\alpha},\boldsymbol{\beta})$ admits a unique minimizer $\boldsymbol{\beta}^* \in \mathbb{R}_+^n$.*

Proof. Symmetric to the proof of Lemma 1, with rows and columns interchanged. □

Proof of Theorem 1. We prove the two parts of the theorem in turn: monotone descent (i) and stationarity of limit points (ii).

Part (i) — monotone descent. At iteration $t \geq T$ with $\boldsymbol{\beta}^{(t)}$ and $\hat{\mathbf{Q}}^{(T)}$ fixed, Algorithm 1 sets $\boldsymbol{\alpha}^{(t+1)}$ to the unique minimizer of $g(\boldsymbol{\alpha}, \boldsymbol{\beta}^{(t)})$ over $\boldsymbol{\alpha}$ (Lemma 1). Since $\boldsymbol{\alpha}^{(t)}$ is feasible:

$$g(\boldsymbol{\alpha}^{(t+1)}, \boldsymbol{\beta}^{(t)}) \leq g(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\beta}^{(t)}). \tag{17}$$

By the symmetric argument, with $\boldsymbol{\alpha}^{(t+1)}$ and $\hat{\mathbf{Q}}^{(T)}$ fixed, Algorithm 1 sets $\boldsymbol{\beta}^{(t+1)}$ to the unique minimizer of $g(\boldsymbol{\alpha}^{(t+1)}, \boldsymbol{\beta})$ over $\boldsymbol{\beta}$ (Lemma 2), giving:

$$g(\boldsymbol{\alpha}^{(t+1)}, \boldsymbol{\beta}^{(t+1)}) \leq g(\boldsymbol{\alpha}^{(t+1)}, \boldsymbol{\beta}^{(t)}). \tag{18}$$

Chaining equation 17 and equation 18:

$$g(\boldsymbol{\alpha}^{(t+1)}, \boldsymbol{\beta}^{(t+1)}) \leq g(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\beta}^{(t)}) \quad \forall\, t \geq T, \tag{19}$$

establishing part (i). Since $g \geq 0$, the sequence $\{g(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\beta}^{(t)})\}_{t \geq T}$ converges to a finite limit $g^* \geq 0$.

Part (ii) — Stationarity of limit points. We invoke Theorem 4.1(c) of (Tseng, 2001), which states that for cyclic block coordinate descent on a continuous objective where each block subproblem has a unique minimizer, every limit point of the iterates is a coordinate-wise stationary point. The three required conditions hold as follows.

(a) **Continuity of g.** With $\hat{\mathbf{Q}}^{(T)}$ fixed, $g(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is a sum of rational functions of α_i and β_j with nonzero denominators on the open positive orthant $\{\alpha_i, \beta_j > 0\}$, hence continuously differentiable on its domain. All iterates remain in this domain by construction under Assumption 1.

(b) **Unique per-block minimizer.** Lemmas 1 and 2 give uniqueness for both block subproblems. Uniqueness follows from the strict convexity of each scalar subproblem after the substitution $u_i = \alpha_i^{-1}$ (resp. $v_j = \beta_j^{-1}$), which yields a strictly convex quadratic with a single minimizer.

(c) **Boundedness of the iterates.** The level set $\{(\boldsymbol{\alpha}, \boldsymbol{\beta}) : g(\boldsymbol{\alpha}, \boldsymbol{\beta}) \leq g(\boldsymbol{\alpha}^{(T)}, \boldsymbol{\beta}^{(T)})\}$ is compact [2]: $g \to \infty$ as any α_i or β_j approaches 0 or ∞ (whenever $\hat{\mathbf{Q}}^{(T)} \neq \mathbf{0}$, ensured by Assumption 1). Combined with the monotone decrease of part (i), all iterates remain in this compact set.

By Theorem 4.1(c) of (Tseng, 2001), every limit point $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ of $\{(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\beta}^{(t)})\}_{t \geq T}$ satisfies the coordinate-wise stationarity conditions $\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha}, \boldsymbol{\beta}^*)$ and $\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} g(\boldsymbol{\alpha}^*, \boldsymbol{\beta})$, completing the proof of part (ii). □

Remark 1 (Format-constrained projection)**.** *Theorem 1 is stated for unconstrained $\boldsymbol{\alpha} \in \mathbb{R}_+^m$ and $\boldsymbol{\beta} \in \mathbb{R}_+^n$. In practice, each scale is projected onto the discrete admissible set after each closed-form update. Since each subproblem in $u_i = \alpha_i^{-1}$ (resp. $v_j = \beta_j^{-1}$) is a strictly convex quadratic, the objective increases monotonically with distance from the unconstrained minimizer. The nearest admissible point to this unconstrained minimizer therefore always achieves the lowest constrained objective, so projecting to the nearest representable value guarantees monotone decrease at each step.*

Remark 2 (On the stabilization assumption)**.** *Theorem 1 assumes that $\hat{\mathbf{Q}}^{(t)}$ stabilizes at a finite iteration T. Since $\hat{\mathbf{Q}}$ takes values in the finite set $\mathbb{F}^{m \times n}$, stabilization is guaranteed whenever the objective decreases monotonically across $\hat{\mathbf{Q}}$ updates. We numerically observe this across model families and quantization formats. For example, for Qwen2.5-7B and Llama3.1-8B with MXINT4, Figures 2 and 3 show that $\hat{\mathbf{Q}}^{(t)}$ stabilizes within $\approx$ 12–16 iterations across every projection layer of every evaluated decoder block, well within our default of $T = 15$ iterations used in all experiments.*

D SINQ BASELINE: BACKGROUND AND EXTENSION

This appendix documents the SINQ (Müller et al., 2026) baseline used in our experiments. We briefly review SINQ's scale-selection procedure (Appendix D.1), describe our extension to MX and NV formats (Appendix D.2), and explain why this extension exhibits the divergence behavior observed in Table 1 on those formats (Appendix D.3).

[2] After fixing the trivial reciprocal scaling ambiguity ($(\boldsymbol{\alpha}, \boldsymbol{\beta}) \sim (c\boldsymbol{\alpha}, \boldsymbol{\beta}/c)$, e.g., by normalizing one of the scale vectors.

D.1 BACKGROUND

SINQ is a calibration-free dual-scaling method that selects row scales $\boldsymbol{\alpha} \in \mathbb{R}_+^m$ and column scales $\boldsymbol{\beta} \in \mathbb{R}_+^n$ to normalize the weight matrix prior to quantization,

$$\breve{\mathbf{W}} = \mathrm{Diag}(\boldsymbol{\alpha})^{-1}\, \mathbf{W}\, \mathrm{Diag}(\boldsymbol{\beta})^{-1}. \tag{20}$$

The scales are obtained by minimizing a *matrix imbalance* objective that drives the per-row and per-column standard deviations of $\breve{\mathbf{W}}$ toward uniformity. The imbalance is reported as the ratio of the largest to the smallest per-axis standard deviation of $\breve{\mathbf{W}}$,

$$\mathcal{I}(\breve{\mathbf{W}}) = \frac{\max\big(\max_i \sigma_i(\breve{\mathbf{W}}),\ \max_j \tau_j(\breve{\mathbf{W}})\big)}{\min\big(\min_i \sigma_i(\breve{\mathbf{W}}),\ \min_j \tau_j(\breve{\mathbf{W}})\big)}, \tag{21}$$

where σ_i and τ_j denote the standard deviations of row i and column j of $\breve{\mathbf{W}}$. A perfectly balanced matrix has $\mathcal{I}(\breve{\mathbf{W}}) = 1$. SINQ minimizes $\mathcal{I}$ via a Sinkhorn–Knopp-style alternating normalization (Sinkhorn & Knopp, 1967) that alternately divides each row and column of $\breve{\mathbf{W}}$ by its current standard deviation, accumulating the reciprocal products into $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. The procedure is calibration-free, processes each layer independently, and runs at roughly $1.1\times$ the cost of RTN.

SINQ was originally designed and evaluated for INT3, INT4, and NF4, all of which store the per-block scales in unconstrained BF16.

D.2 EXTENDING SINQ TO W+A AND MX/NV FORMATS

To produce baseline W+A results for MXINT4, MXFP4, and NVFP4, we extend SINQ in two directions: *(i)* to joint W+A quantization by applying SINQ's column scales to rescale activations before quantization (same format as weights), and *(ii)* to MX and NV formats by running SINQ's algorithm to completion and then projecting the resulting scales to its target data format (E8M0 for MX, FP8 UE4M3 for NVFP4) as a post-processing step. Since SINQ's optimization is unaware of scale format constraints, this projection only partially mitigates the mismatch: on MXINT4 and MXFP4 it leads to noticeable quality degradation, and on NVFP4 the scales drift entirely outside the FP8 representable range, causing complete divergence (marked † in the tables).

Both extensions are our own and have not been validated by the original authors; SINQ+ results on MX and NV formats should accordingly be treated as best-effort baselines. We note that even in its original weight-only setting without any of our extensions, SINQ produces inferior scales to DUALQUANT across all formats (Appendix H).

D.3 WHY SINKHORN ITERATION DIVERGES UNDER FORMAT CONSTRAINTS

The large perplexity values observed for SINQ on MXINT4 and MXFP4 in Tables 1 and 7, and the complete divergence on NVFP4 (marked †), indicate a fundamental incompatibility between SINQ's format-unaware Sinkhorn iteration and the discrete scale constraints of MX and NV formats. In the unconstrained BF16 setting, Sinkhorn–Knopp iteration converges to a unique fixed point $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ (Sinkhorn, 1964). The fixed-point scale values lie on a continuum and can be represented exactly in BF16. Once the admissible set is restricted to a discrete format such as E8M0, however, this property is lost. The fixed-point Sinkhorn target generally lies between two representable E8M0 values, and the projection step that follows each Sinkhorn iteration snaps the iterate to the nearest representable scale rather than the true fixed point. The next Sinkhorn step then re-derives a target that may point away from the projected iterate, causing the iteration to oscillate between nearby E8M0 representatives without converging. DUALQUANT avoids this failure mode by construction. Its closed-form coordinate updates are derived under the format constraint from the outset and the projection step preserves the monotone descent property of the BCD scheme (Theorem 1), so iterates remain in the admissible set throughout.

D.4 COMPARISON OF OBJECTIVES

Beyond the format-compatibility issue analyzed in Appendix D.3, SINQ's matrix imbalance objective equation 21 measures a different quantity from the post-quantization reconstruction error: a lower

imbalance does not necessarily translate to a lower MSE, since the imbalance metric is a function of $\breve{\mathbf{W}}$ alone and makes no reference to the quantization grid, which means the same imbalance value corresponds to different post-quantization errors under INT4, MXINT4, MXFP4, and NVFP4. Our weight reconstruction MSE, in contrast, contains the format-specific quantizer $Q(\cdot)$ explicitly, so the optimization is aware of the grid that the scales will be evaluated against and can adapt the scale assignment to non-uniform grids such as those of FP4 and NVFP4. This explains why a method optimized for low matrix imbalance does not automatically achieve low quantization error, as illustrated in Figure 1. Furthermore, the reconstruction MSE is more directly tied to downstream metrics such as perplexity and accuracy than proxy objectives such as matrix imbalance, which helps explain why DUALQUANT, by minimizing reconstruction error directly, exhibits stronger downstream performance than SINQ across most evaluated scenarios.

E ISOLATED ACTIVATION-SIDE EFFECT OF β

Section 4.2 establishes algebraically that the column scale β reshapes the per-channel statistics of the activations presented to the quantizer, and argues mechanistically that weight-MSE-optimal scales attenuate the same channels where activation outliers concentrate. To confirm this is a genuine improvement in activation quantization quality, and not merely a by-product of better weight quantization, we isolate β's effect on activations by holding the quantized weights fixed.

Setup. We take DUALQUANT's quantized weight matrix $\hat{Q}$ for each projection in block 0 of Llama-3.1-8B (INT4, G=64) as fixed and measure the Signal-to-Quantization-Noise Ratio (SQNR, dB, higher is better) of the quantized activations with and without applying β, using 256 representative input sequences. Block 0 is a representative early block; the pattern holds across depth. All other conditions are identical between the two runs.

Results. Table 6 reports per-projection activation SQNR. Applying β improves activation SQNR by $+0.33$ dB on average across projections and by $+0.53$ dB at the block output level. The improvement is consistent across all seven projection types, with the largest gains on `k_proj` ($+0.81$ dB) and `v_proj` ($+0.65$ dB) — precisely the projections where activation outliers are most pronounced (Dettmers et al., 2022; Sun et al., 2024). This confirms that β provides a real, partial activation-side benefit that is separate from and additive to its role in reducing weight quantization error. The gain is not calibration substitution: unlike SmoothQuant, β is derived entirely from the static weight matrix, yet still measurably reduces the activation quantization error.

Table 6: Per-projection activation SQNR (dB, ↑) with and without DUALQUANT's β, on Llama-3.1-8B block 0 (INT4, G=64). Quantized weights $\hat{Q}$ are held fixed; only the presence of the β pre-scaling varies. $\Delta > 0$ indicates β reduces activation quantization error. Each Δ is computed at full precision and may differ slightly from the difference of the rounded columns. Other blocks exhibit the same qualitative behavior.

Projection	**β applied**	**β not applied**	**Δ (dB)**
`k_proj`	17.61	16.80	$+0.81$
`v_proj`	17.45	16.80	$+0.65$
`up_proj`	18.48	18.20	$+0.27$
`q_proj`	17.05	16.80	$+0.25$
`gate_proj`	18.36	18.20	$+0.15$
`o_proj`	17.40	17.34	$+0.07$
`down_proj`	16.76	16.70	$+0.06$
Mean	17.59	17.26	$+0.33$
Block Output SQNR	17.98	17.44	$+0.53$

F ADDITIONAL DETAILS ON BASELINES

This appendix details each baseline used in the joint W+A quantization comparison (Tables 1 and 7). All calibration-based baselines (AWQ+ and SmoothQuant) use the same calibration set: 256 sequences of length 8192 randomly sampled from the WikiText-2 training split, strictly disjoint from the evaluation splits. All baselines use the same block sizes and scale formats as DUALQUANT for each evaluated format (INT4 G=64; MXINT4/MXFP4 G=32 with E8M0; NVFP4 G=16 with FP8 UE4M3). Below we describe the six baselines in detail.

- **RTN.** Applies round-to-nearest quantization independently to weights and activations using format-native per-row or per-block absmax scaling, with no rescaling, calibration, or optimization. Activations are quantized dynamically at inference using per-token statistics. RTN serves as the lower-bound baseline.
- **AWQ+.** AWQ (Lin et al., 2024) is a weight-only, activation-aware method that rescales weights based on high-activation channels. We extend it to W+A by adding RTN activation quantization after AWQ rescaling, and denote this method as AWQ+. Scales are computed using the official AWQ search procedure with a 20-candidate grid and clipping ratio search enabled.
- **SmoothQuant.** SmoothQuant (Xiao et al., 2023) uses calibration data to move activation outliers into weights via per-channel rescaling $s_j \propto \max_i |\mathbf{X}_{ij}|^{\gamma} / \max_i |\mathbf{W}_{ij}|^{1-\gamma}$. We evaluate the migration strength $\gamma \in \{0.5, 0.85\}$ as suggested in the original paper (Xiao et al., 2023); since $\gamma = 0.5$ consistently yields better performance across our experiments, we adopt it for SmoothQuant baselines. After rescaling, weights and activations are quantized using the same block-wise quantizer as DUALQUANT. It serves as the main calibration-based W+A baseline.
- **SINQ+.** SINQ (Müller et al., 2026) is a calibration-free weight-only dual-scale method originally designed for INT3/INT4 and NF4 with unconstrained BF16 scales. We extend it to W+A and MX/NV formats as described in Appendix D.2; however, since SINQ's optimization is entirely unaware of scale format constraints, projecting its output scales to the target data format only partially mitigates the mismatch, causing degraded results on MXINT4 and MXFP4 and complete divergence on NVFP4 (marked †). SINQ+ results on MX and NV formats should be treated as best-effort baselines.
- **QuaRot.** QuaRot (Ashkboos et al., 2024) is a calibration-free W+A method that suppresses activation outliers by applying randomized Hadamard rotations to the hidden states, spreading outlier energy across channels while leaving the network output unchanged, since each rotation is absorbed into the following weights. The residual-stream rotation is fused offline; the value-path and down-projection rotations run online. Following rotation, weights and activations are quantized using the same block-wise quantizer as DUALQUANT, with activations quantized dynamically based on post-rotation per-token statistics. QuaRot serves as the rotation-based counterpart to RTN, isolating the effect of rotation without weight-error compensation.

G ADDITIONAL W+A QUANTIZATION RESULTS

This appendix provides supplementary quantization results complementing the main experiments of Section 5.1. We report 4-bit W+A results on the Qwen model family and 8-bit W+A results across both families, confirming that DUALQUANT's calibration-free activation rescaling generalizes across model architectures, quantization formats, and bit-widths.

G.1 4-BIT W+A QUANTIZATION

Perplexity. Table 7 reports WikiText-2 and C4 perplexity for joint W+A quantization on the Qwen family across all four 4-bit formats. Results are broadly consistent with the Llama family (Table 1 in the main text): DUALQUANT closely matches or surpasses SmoothQuant on the MX and NV formats without calibration data, and outperforms the calibration-free baselines QuaRot and SINQ+ in most settings, as these baselines frequently diverge or degrade severely on the block formats. On NVFP4, DUALQUANT surpasses SmoothQuant on Qwen2.5-7B and Qwen3-0.6B (e.g., 6.53 vs. 6.70 and 23.77 vs. 24.16 on WikiText-2) and closely matches it on Qwen3-14B (7.75 vs. 7.73), while SINQ+ diverges on all three models and QuaRot either diverges or degrades drastically (e.g., 12.11 and

Table 7: WikiText-2 and C4 perplexity (↓) for joint weight-and-activation quantization at 4-bit precision on the Qwen family. DUALQUANT applies β-rescaling to activations without calibration data. SmoothQuant and AWQ+ use activation statistics from 256 calibration samples. SINQ+ is our extension of SINQ to W+A. QuaRot rotates activations/weights online with a Hadamard transform (no calibration). **Bold**: best calibration-free (**CF**) result. (†): degenerate result ($PPL > 1000$); method incompatible with format. ✓: calibration-free. ✗: requires calibration data.

				WikiText-2 PPL ↓			C4 PPL ↓		
Bits	**Format**	**Method**	**CF**	**Qwen2.5-7B**	**Qwen3-0.6B**	**Qwen3-14B**	**Qwen2.5-7B**	**Qwen3-0.6B**	**Qwen3-14B**
BF16 (reference)			–	6.15	18.14	7.54	9.12	27.30	11.06
4-bit	A: INT4 W: INT4 (G=64)	RTN	✓	7.47	49.44	8.39	13.50	74.78	14.83
		AWQ+	✗	7.36	38.80	8.19	13.21	59.20	14.43
		SmoothQuant	✗	7.10	30.67	8.12	12.74	50.73	14.21
		SINQ+	✓	7.28	30.21	8.02	12.94	49.05	14.15
		QuaRot	✓	**6.97**	48.47	8.86	**12.53**	75.07	15.11
		DUALQUANT	✓	7.38	**29.91**	**7.99**	13.11	**47.87**	**14.14**
	A: MXINT4 W: MXINT4 (G=32)	RTN	✓	7.62	48.48	8.61	13.60	74.73	15.22
		AWQ+	✗	7.52	38.61	8.44	13.39	60.99	14.73
		SmoothQuant	✗	7.36	33.63	8.42	13.08	52.34	14.75
		SINQ+	✓	9.65	353.83	10.55	16.75	902.98	17.48
		QuaRot	✓	7.64	131.31	9.95	13.58	202.08	17.01
		DUALQUANT	✓	**7.59**	**32.99**	**8.42**	**13.45**	**52.79**	**14.79**
	A: MXFP4 W: MXFP4 (G=32)	RTN	✓	7.79	29.37	8.70	13.82	46.71	**15.14**
		AWQ+	✗	7.62	29.88	8.62	13.59	50.44	15.05
		SmoothQuant	✗	7.68	28.66	8.53	13.63	47.83	14.95
		SINQ+	✓	10.05	195.98	11.16	17.63	395.39	18.30
		QuaRot	✓	**7.38**	68.96	9.45	**13.22**	109.84	16.40
		DUALQUANT	✓	7.45	**27.46**	**8.59**	13.49	**42.59**	15.16
	A: NVFP4 W: NVFP4 (G=16)	RTN	✓	7.07	26.12	7.93	12.70	40.75	13.98
		AWQ+	✗	6.80	25.60	7.95	12.24	41.95	13.91
		SmoothQuant	✗	6.70	24.16	7.73	12.17	37.00	13.64
		SINQ+	✓	Div†	Div†	Div†	Div†	Div†	Div†
		QuaRot	✓	12.11	173.38	Div†	26.37	280.53	Div†
		DUALQUANT	✓	**6.53**	**23.77**	**7.75**	**11.81**	**36.64**	**13.66**

173.38 on Qwen2.5-7B and Qwen3-0.6B), leaving DUALQUANT as the strongest calibration-free method on this format. On MXFP4, DUALQUANT outperforms SmoothQuant on Qwen3-0.6B (e.g., 27.46 vs. 28.66 on WikiText-2) and achieves substantially better C4 perplexity (42.59 vs. 47.83), while SINQ+ collapses (195.98) and QuaRot degrades substantially (68.96); on Qwen2.5-7B, QuaRot remains competitive (7.38), narrowly ahead of DUALQUANT (7.45). On MXINT4, DUALQUANT outperforms SmoothQuant on Qwen3-0.6B (32.99 vs. 33.63) and matches it on Qwen3-14B (8.42 vs. 8.42), while SINQ+ degrades severely (353.83) and QuaRot collapses (131.31). On INT4, DUALQUANT remains competitive with SmoothQuant across all three models, though slightly below on Qwen2.5-7B (7.38 vs. 7.10); QuaRot achieves the best calibration-free result on Qwen2.5-7B (6.97) but degrades substantially on Qwen3-0.6B (48.47), while SINQ+ remains competitive on this format. The results on C4 are consistent with WikiText-2, confirming that the calibration-free activation rescaling generalizes across evaluation distributions. Overall, these results show that DUALQUANT provides consistently strong calibration-free W+A quantization across formats without relying on calibration data or format-specific rotation behavior.

Zero-shot downstream tasks. Table 8 reports zero-shot accuracy on MMLU, HellaSwag, PIQA, ARC-Easy, and WinoGrande for Qwen3-14B across all four W+A quantization formats. Results are consistent with the Llama-3.1-8B results (Table 2 in the main text): DUALQUANT achieves the best calibration-free accuracy on nearly all format-task combinations. On MXFP4 and MXINT4, DUALQUANT outperforms both calibration-free baselines QuaRot and SINQ+ on every task, often by a large margin, and stays close to calibration-based SmoothQuant, surpassing it on WinoGrande (72.06 vs. 69.14 on MXFP4 and 70.64 vs. 66.30 on MXINT4) while trailing narrowly on the remaining tasks (e.g., 72.91 vs. 73.26 MMLU on MXFP4 and 71.47 vs. 72.50 MMLU on MXINT4). On NVFP4, DUALQUANT is the strongest calibration-free method: SINQ+ collapses to near-random accuracy (23.39 MMLU), whereas DUALQUANT stays competitive with calibration-based SmoothQuant (e.g.,

Table 8: Qwen3-14B downstream accuracy (0-shot, %, higher is better ↑) for RTN, AWQ+, SmoothQuant, SINQ+, DUALQUANT, and QuaRot, across all 4 weight formats; QuaRot is omitted entirely for NVFP4, since QuaRot's Hadamard rotation redistributes activation magnitude in a way that collapses NVFP4's narrow-range E4M3 block scale. **Bold**: best calibration-free (CF) result per column within each model+format block. ✓: calibration-free. ✗: requires calibration data.

Qwen3-14B							
Format	Method	CF	MMLU	HellaSwag	PIQA	ARC-Easy	WinoGrande
INT4 (G=64)	RTN	✓	70.33	75.23	77.64	77.36	69.06
	AWQ+	✗	71.92	75.03	77.86	79.08	69.53
	SmoothQuant	✗	73.04	76.74	77.91	80.77	71.51
	SINQ+	✓	72.30	76.23	**77.80**	79.59	70.17
	QuaRot	✓	71.30	73.52	76.77	77.99	67.88
	DUALQUANT	✓	**72.33**	**76.54**	76.71	**80.51**	**71.09**
MXINT4 (G=32)	RTN	✓	70.08	74.31	76.69	74.74	66.08
	AWQ+	✗	72.42	74.93	77.37	77.53	66.93
	SmoothQuant	✗	72.50	75.84	77.97	78.62	66.30
	SINQ+	✓	62.38	69.26	74.59	70.50	63.54
	QuaRot	✓	68.72	69.92	75.63	75.72	63.85
	DUALQUANT	✓	**71.47**	**75.27**	**77.04**	**76.68**	**70.64**
MXFP4 (G=32)	RTN	✓	72.36	**75.59**	77.15	79.76	69.14
	AWQ+	✗	72.99	75.89	77.15	79.46	70.72
	SmoothQuant	✗	73.26	75.53	77.97	79.46	69.14
	SINQ+	✓	62.58	68.53	74.16	73.02	63.85
	QuaRot	✓	68.34	71.01	76.71	72.85	68.90
	DUALQUANT	✓	**72.91**	75.55	**77.48**	**80.01**	**72.06**
NVFP4 (G=16)	RTN	✓	74.55	76.97	78.29	80.09	71.27
	AWQ+	✗	74.84	77.32	78.78	81.73	71.03
	SmoothQuant	✗	75.97	77.55	78.94	82.79	72.77
	SINQ+	✓	23.39	26.38	50.22	26.56	50.59
	DUALQUANT	✓	**75.06**	**77.09**	**78.99**	**81.90**	**72.64**

75.06 vs. 75.97 MMLU, 78.99 vs. 78.94 PIQA), closing most of the gap without any access to calibration data. On INT4, DUALQUANT matches SmoothQuant closely across all five tasks and attains the best calibration-free accuracy on four of the five, trailing only SINQ+ on PIQA (76.71 vs. 77.80). As on Llama, SINQ+ degrades severely on MX formats (62.38–62.58 MMLU on MXINT4 and MXFP4) and collapses on NVFP4, confirming that the perplexity improvements of DUALQUANT over calibration-free baselines translate directly to downstream task quality.

G.2 8-BIT W+A QUANTIZATION

Table 9 reports WikiText-2 perplexity for joint W+A quantization at 8-bit precision across INT8, MXINT8, and MXFP8. At 8 bits the quantization error is already small, so there is limited room for improvement over BF16; nonetheless, DUALQUANT remains the strongest calibration-free method and is effectively lossless. On MXINT8 and MXFP8 it matches or surpasses the calibration-based SmoothQuant on every model except MXFP8 Qwen3-14B, where it trails by only 0.01 (7.64 vs. 7.63), all without any calibration data; representative gains include 18.13 vs. 18.18 on Qwen3-0.6B and 7.53 vs. 7.55 on Qwen3-14B for MXINT8. On MXINT8, DUALQUANT tracks the BF16 reference almost exactly (e.g., 6.15 vs. 6.15 on Qwen2.5-7B and 18.13 vs. 18.14 on Qwen3-0.6B), confirming near-lossless quantization. In contrast, the other calibration-free baseline, SINQ+, degrades severely on the MX formats even at 8 bits (e.g., 47.87 and 53.41 on Qwen3-0.6B for MXINT8 and MXFP8), confirming that its format-unaware scales remain incompatible with MX block constraints regardless of bit-width and underscoring the benefit of DUALQUANT's format-aware dual scaling.

H WEIGHT-ONLY QUANTIZATION RESULTS

While the primary contribution of DUALQUANT is calibration-free joint W+A quantization, Table 10 evaluates its weight-only performance, with activations kept in BF16, to assess whether the MSE-

Table 9: WikiText-2 perplexity (↓) for 8-bit W+A quantization. DUALQUANT applies β-rescaling without calibration data. SmoothQuant uses 256 calibration samples. SINQ+ is our extension of SINQ to W+A. **Bold**: best calibration-free (**CF**) result. ✓: calibration-free. ✗: requires calibration data.

Bits	Format	Method	CF	Qwen2.5-7B	Qwen3-0.6B	Qwen3-14B	Llama-3.1-8B
BF16 (reference)			–	6.15	18.14	7.54	5.62
8-bit	A: INT8 W: INT8 (G=128)	RTN	✓	**6.27**	18.42	7.55	5.72
		SmoothQuant	✗	6.25	18.28	7.50	5.70
		SINQ+	✓	6.67	18.33	**7.49**	5.70
		DUALQUANT	✓	6.64	**18.24**	7.77	**5.69**
	A: MXINT8 W: MXINT8 (G=32)	RTN	✓	6.15	**18.06**	7.54	5.63
		SmoothQuant	✗	6.15	18.18	7.55	5.63
		SINQ+	✓	7.37	47.87	9.04	9.51
		DUALQUANT	✓	**6.15**	18.13	**7.53**	**5.63**
	A: MXFP8 W: MXFP8 (G=32)	RTN	✓	6.22	18.54	**7.62**	5.72
		SmoothQuant	✗	6.22	18.46	7.63	5.71
		SINQ+	✓	7.49	53.41	9.21	10.48
		DUALQUANT	✓	**6.21**	**18.23**	7.64	**5.71**

based dual scaling also yields competitive weight reconstruction in isolation. DUALQUANT achieves the best calibration-free results on nearly all format–model combinations (on Qwen3-14B it is marginally behind SINQ on INT4 and RTN on NVFP4), closely matching calibration-based AWQ on INT4 and MX formats and remaining within a small margin of GPTQ without requiring any calibration data or Hessian computation. While GPTQ relies on costly second-order weight updates and is inapplicable when training data is proprietary, DUALQUANT operates purely from weight statistics, yet closely follows its perplexity in INT4 and MX formats and *outperforms* it on NVFP4 for three of the four models.

On MXINT4 and MXFP4, DUALQUANT substantially outperforms SINQ, whose format-unaware scales cause severe degradation (e.g., 112.00 and 94.84 on Qwen3-0.6B for MXINT4 and MXFP4 respectively). On NVFP4, DUALQUANT achieves the strongest results among all methods on Qwen2.5-7B (6.33) and Qwen3-0.6B (19.82).

These results confirm that DUALQUANT is highly effective in the weight-only setting, with MSE minimization directly translating into measurable perplexity gains. More significantly, the same optimization simultaneously enables calibration-free W+A quantization (Table 1), distinguishing DUALQUANT from prior weight-only methods.

Importantly, the SINQ results in Table 10 use the original SINQ algorithm (Müller et al., 2026) with no activation scaling involved. For MX and NV formats the only modification is projecting SINQ's output scales to the target data format after the algorithm completes, as there is no other way to deploy them in these formats. Even in this purely weight-only setting, DUALQUANT outperforms SINQ on nearly all formats and models. Notably, on INT4, where SINQ operates exactly as designed with unconstrained BF16 scales, DUALQUANT achieves better perplexity on three out of four models. On MX and NV formats the gap widens dramatically, confirming that DUALQUANT provides better generalizability to block quantization formats.

I REAL PACKED W4A4 DEPLOYMENT

Implementation. The packed inference path targets INT4 with group size G=64 and is built on the packed W4A4 GEMM backend from Nunchaku/SVDQuant (Li et al., 2025). The quantized weights, $\hat{Q}$, and scales produced by Algorithm 1 for all 224 linear layers of Llama-3.1-8B (Grattafiori et al., 2024) (7 projections $\times$ 32 transformer blocks) are packed offline into the target storage layout and executed through this backend. Embedding, lm_head, and normalization layers remain in BF16. We extend the backend with two components: (i) a CUDA kernel that incorporates the column scale β into activation quantization, avoiding a separate kernel launch, and (ii) an allocation-free GEMM entry point that enables CUDA Graph capture for the full model.

Table 10: WikiText-2 perplexity (↓) for 4-bit weight-only quantization. Activations in BF16. DUALQUANT applies β-rescaling without calibration data. AWQ and GPTQ use 256 calibration samples. (†): degenerate result (PPL$>$ 1000); method incompatible with format. **Bold**: best calibration-free (**CF**) result. ✓: calibration-free. ✗: requires calibration data.

Bits	Format	Method	CF	Qwen2.5-7B	Qwen3-0.6B	Qwen3-14B	Llama-3.1-8B
BF16 (reference)			–	6.15	18.14	7.54	5.62
4-bit	W: INT4 (G=64)	RTN	✓	6.67	26.33	7.93	6.32
		AWQ	✗	6.50	21.83	7.72	6.12
		GPTQ	✗	6.44	20.39	7.73	5.56
		SINQ	✓	6.57	22.21	**7.76**	6.16
		DUALQUANT	✓	**6.55**	**21.35**	7.88	**6.12**
	W: MXINT4 (G=32)	RTN	✓	6.66	25.99	7.98	6.41
		AWQ	✗	6.49	21.90	7.85	6.16
		GPTQ	✗	6.49	20.09	7.79	6.25
		SINQ	✓	7.98	112.00	9.67	13.09
		DUALQUANT	✓	**6.60**	**23.02**	**7.89**	**6.19**
	W: MXFP4 (G=32)	RTN	✓	6.79	22.38	8.04	6.29
		AWQ	✗	6.58	20.98	7.94	6.31
		GPTQ	✗	6.47	19.79	7.79	6.11
		SINQ	✓	8.03	94.84	9.85	16.16
		DUALQUANT	✓	**6.63**	**22.35**	**7.95**	**6.19**
	W: NVFP4 (G=16)	RTN	✓	6.85	23.35	**7.81**	8.12
		AWQ	✗	6.61	22.56	7.89	7.93
		GPTQ	✗	6.55	20.41	7.74	7.61
		SINQ	✓	Div†	Div†	Div†	Div†
		DUALQUANT	✓	**6.33**	**19.82**	8.04	**6.43**

Evaluation setup. All measurements are performed on a single NVIDIA RTX 4090 (NVIDIA Corporation, 2022) with 24 GB of memory. WikiText-2 perplexity is evaluated at sequence length seq $= 8192$. Latency and memory are measured using an 82-token prompt for prefill without a KV cache and 100 generated tokens for decode, corresponding to a 182-token KV cache. All reported measurements are collected from warm runs and remain stable within 0.2% across repetitions.

Accuracy. The packed W4A4 implementation achieves a WikiText-2 perplexity of 6.74, compared with 6.73 under fake quantization (Table 1), a difference of only 0.01. The BF16 baseline measured using the same evaluation harness has a perplexity of 5.61, compared with 5.62 reported in Table 1.

Latency breakdown. Table 11 reports the GPU time for individual components under BF16 and DUALQUANT W4A4 with INT4 (G=64) format. Quantized weights remain packed throughout inference, and group scales are applied within the GEMM, eliminating runtime weight dequantization. Activation quantization, including the β pre-scaling operation, costs 0.70–3.09 ms and accounts for 4.3–6.1% of total W4A4 GPU time, constituting the only per-step quantization overhead. The largest reduction comes from the quantized linear layers, whose execution time decreases from 17.33 to 8.82 ms for prefill with batch size 1 and from 121.49 to 32.98 ms for prefill with batch size 16. The remaining computation consists primarily of components retained in BF16, including lm_head, attention, normalization, RoPE, residual operations, and KV-cache updates, which limits the achievable decode speedup. Note that component-level timings are collected in a separate instrumented run; the associated measurement overhead makes the totals in Table 11 slightly larger than the end-to-end times reported in Table 3.

Memory breakdown. Table 12 provides an end-to-end memory breakdown of the DUALQUANT packed deployment for Llama-3.1-8B when quantized to INT4 (G=64). Quantization reduces the storage for the 224 linear layers from 13.96 GB to 3.49 GB, with an additional 0.22 GB required for per-group weight scales. Together with the unquantized BF16 embedding and lm_head (2.10 GB), the total weight footprint decreases from 16.20 GB to 5.92 GB. The remaining activation and KV-cache components stay in BF16 and therefore have the same footprint in both paths. Prefill uses `use_cache=False`, so no KV cache is allocated; its activation footprint is dominated by the

Table 11: GPU time (ms) per component for BF16 and DUALQUANT W4A4, INT4 (G=64), on Llama-3.1-8B using an RTX 4090.

Component	Prefill (b=1)		Prefill (b=16)		Decode (b=1)		Decode (b=16)	
	BF16	W4A4	BF16	W4A4	BF16	W4A4	BF16	W4A4
Quantized linears (224)	17.33	8.82	121.49	32.98	14.75	8.24	16.32	9.18
Activation quantization (incl. β)	–	0.82	–	3.09	–	0.70	–	0.72
lm_head (BF16)	1.19	1.19	9.26	9.26	1.07	1.07	1.10	1.10
Attention	0.19	0.20	0.93	0.94	0.42	0.42	1.73	1.74
Norms / RoPE / KV cache / residual	2.43	2.33	14.03	13.37	1.60	1.64	4.18	4.14
Total GPU time	**21.14**	**13.36**	**145.71**	**59.64**	**17.84**	**12.07**	**23.33**	**16.88**

Table 12: End-to-end memory breakdown (GB) for BF16 and DUALQUANT W4A4 on Llama-3.1-8B using an RTX 4090.

Component	Prefill (b=1)		Prefill (b=16)		Decode (b=1)		Decode (b=16)	
	BF16	W4A4	BF16	W4A4	BF16	W4A4	BF16	W4A4
Quantized linear weights	13.96	3.49	13.96	3.49	13.96	3.49	13.96	3.49
Per-group weight scales (BF16)	–	0.22	–	0.22	–	0.22	–	0.22
Embedding + lm_head (BF16)	2.10	2.10	2.10	2.10	2.10	2.10	2.10	2.10
Runtime/allocator overhead	0.14	0.11	0.14	0.11	0.14	0.11	0.14	0.11
Weights subtotal	**16.20**	**5.92**	**16.20**	**5.92**	**16.20**	**5.92**	**16.20**	**5.92**
First-forward scratch	0.10	0.30	0.10	0.30	0.10	0.30	0.10	0.30
KV cache	0	0	0	0	0.02	0.02	0.38	0.38
Other activations	0.03	0.03	0.36	0.36	0.02	0.02	0.07	0.07
Total	**16.33**	**6.25**	**16.66**	**6.58**	**16.34**	**6.26**	**16.75**	**6.67**

$[\text{seq} \times 128256]$ logits tensor. Consequently, memory usage for prefill with batch size 1 is slightly lower than for decode with batch size 1.

J LICENSES FOR EXISTING ASSETS

All models, datasets, and software libraries used in this work are publicly available and used in accordance with their respective licenses. Table 13 summarizes the assets along with their licenses and sources. The Llama models are released under Meta's Llama Community Licenses and the Qwen models under Apache 2.0. The evaluation datasets are released under a mix of permissive and share-alike licenses: WikiText-2 (CC BY-SA 3.0), C4 (ODC-BY 1.0), HellaSwag (MIT), MMLU (MIT), PIQA (AFL-3.0), ARC-Easy (CC BY-SA 4.0), and WinoGrande (CC-BY). All assets were used solely for academic research purposes consistent with their terms of use.

Table 13: Licenses of existing assets used in this work.

Category	Assets and Licenses
Models	Llama-2-7B (Llama 2 Community); Llama-3.1-8B (Llama 3.1 Community); Llama-3.2-1B (Llama 3.2 Community); Qwen2-1.5B, Qwen2.5-7B, Qwen3-0.6B, Qwen3-14B (Apache 2.0)
Datasets	WikiText-2 (CC BY-SA 4.0); C4 (ODC-BY 1.0); MMLU (MIT); HellaSwag (MIT); PIQA (AFL-3.0); ARC-Easy (CC BY-SA 4.0); WinoGrande (CC-BY)