# WUSH-KV: KV CACHE QUANTIZATION WITH DATA-ADAPTIVE TRANSFORMS

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

## ABSTRACT

KV cache memory and bandwidth costs grow with context length and batch size, which limits efficient long-context inference. To address this bottleneck, we introduce WUSH-KV for low-bit KV-cache quantization. It adapts WUSH, which constructs a data-aware transform from the second-order statistics of both factors in a matrix product to reduce quantization error. WUSH-KV uses calibration data to construct separate key and value transforms, with the value transform folded into the model weights and the key transform applied after RoPE. The transforms can be paired with clipped quantizers. For one such quantizer, QuEST INT, we show that, under mild assumptions, the WUSH transform is near-optimal. With this quantizer, WUSH-KV reduces layerwise reconstruction error and achieves the lowest end-to-end perplexity among other tested transforms. For end-to-end evaluation, we integrate WUSH-KV into SGLang using OSCAR-style percentile-clipped affine quantization. At 2-bit, WUSH-KV performs comparably to or outperforms the OSCAR transform across all evaluated models and downstream tasks.

## 1 INTRODUCTION

Large language models are increasingly used with long contexts and large serving batches. This puts growing pressure on the memory capacity and bandwidth of inference systems. One important source of this cost is the key-value (KV) cache. During autoregressive generation, each attention layer stores the keys and values of previous tokens so that they do not need to be recomputed. The cache grows linearly with sequence length and batch size, and it is repeatedly read as new tokens are generated. As a result, the KV cache can become a major bottleneck for long-context inference. Low-bit quantization provides a simple way to reduce its memory footprint and data movement. Yet, aggressive quantization can introduce large errors into both attention scores and outputs.

Changes of basis (linear transforms) can make cached vectors easier to quantize, but the quality of the transform should also reflect how quantization errors propagate through attention. Key errors affect query-key products, whereas value errors affect projected attention outputs. This motivates separate transforms that account for both cache statistics and downstream sensitivity. WUSH (Chen et al., 2026) was recently introduced as a principled approach to joint weight-activation quantization. It constructs a closed-form invertible transform from the second-order statistics of both factors in a matrix product. This product-aware view naturally extends to the KV cache.

In this work, we adapt WUSH from weight-activation quantization to KV-cache quantization. We call the resulting method *WUSH-KV*. For each KV head, WUSH-KV constructs separate transforms for keys and values using calibration-time statistics. The key transform accounts for the query heads that consume the cached keys. The value transform accounts for the output-projection blocks that consume the cached values. The value-side transform can be folded into the model weights and therefore adds no online transform cost. The key-side transform and its query-side compensation are applied after headwise normalization (when present) and rotary positional encoding. The WUSH transforms operate independently of the scalar quantizer and can be paired with per-token clipped quantizers. WUSH-KV also retains sink and recent cache entries in full precision. Our theoretical analysis focuses on the QuEST (Panferov et al., 2025) quantizer. Under an additive rounding-noise model and explicit conditions on normalized tails and clipping errors, we show that the ideal WUSH transform is near-optimal among transforms with balanced coordinate sensitivity. The controlled reconstruction study uses the same quantizer to isolate transform quality, while the end-to-end evaluations use



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

the percentile-clipped affine quantizer from the OSCAR (Zhou et al., 2026) concurrent work. In our experiments, WUSH-KV consistently reduces attention reconstruction error and achieves the lowest perplexity among all existing methods. In 2-bit downstream benchmarks, it scores higher than OSCAR on all four Qwen3-8B tasks and is competitive at 4B and 32B.

## 2 Related Work

**KV cache quantization.** There is a long line of work on this topic. KIVI (Liu et al., 2024) observes that keys and values have different quantization characteristics, so it quantizes keys per channel and values per token. Recent residual keys and values remain in full precision. KVQuant (Hooper et al., 2024) also exploits the channel-wise structure of keys. It quantizes keys before RoPE, uses sensitivity-weighted nonuniform quantization, and handles outliers with a per-vector dense-and-sparse representation. It keeps the first token in FP16 to protect the attention sink. AQUA-KV (Shutova et al., 2025) exploits cross-layer dependencies with compact adapters to predict cached keys and values and quantizes the remaining residual information. These methods focus on quantizer design, outlier handling, and selective high-precision retention.

**Transform-based quantization.** Another line of work changes the representation before quantization. QuaRot (Ashkboos et al., 2024) redistributes outliers with randomized Hadamard transforms. This enables end-to-end 4-bit quantization of weights, activations, and the KV cache. SpinQuant (Liu et al., 2025) shows that quantized accuracy can vary substantially across rotations. It learns two mergeable orthogonal rotations by minimizing a task loss on calibration data. For low-bit activations and KV caches, it also uses efficient fixed online Hadamard transforms. FlatQuant (Sun et al., 2025) learns Kronecker-structured transforms through calibration and applies head-wise transforms to keys and values in the KV cache. RotateKV (Su et al., 2025) specializes in rotations for aggressive 2-bit cache compression. It combines calibration-based channel reordering with grouped-head key rotations applied before RoPE. It also identifies additional attention-sink tokens and retains their KV entries in FP16. TurboQuant (Zandieh et al., 2026) randomly rotates vectors and applies scalar quantizers matched to the resulting coordinate distribution. For unbiased inner-product estimation, the published method additionally applies a one-bit-per-coordinate QJL sketch to the quantization residual. Subsequent analysis (Ben-Basat et al., 2026) notes that randomized Hadamard transforms can replace uniform random rotations in this family of quantizers. Practical implementations (e.g., vLLM[1]) likewise use efficient Hadamard transforms but drop the QJL sketch as harmful. This motivates the normalized Hadamard transform as our practical TurboQuant-style fixed-transform baseline. Concurrent with our work, OSCAR (Zhou et al., 2026) estimates attention-aware covariance statistics offline, and constructs separate fixed *orthogonal* transforms for keys and values. It also calibrates per-layer clipping parameters that determine token-wise clipping thresholds. The method is implemented in SGLang (Zheng et al., 2024), a serving system compatible with paged KV caches. OSCAR restricts its key and value transforms to be orthogonal, whereas WUSH-KV allows general invertible transforms. This additional freedom permits anisotropic rescaling to jointly balance cache statistics and downstream sensitivity, which an orthogonal transform cannot generally realize.

## 3 Background on the WUSH Transform

Post-training quantization replaces a model's weights, and sometimes its activations, with low-precision codes that share a scale within each group of elements. A handful of outliers carry far more amplitude than the rest, so they set the scale, and many of the available quantization levels go unused. Transforms are one of the remedies: for a matrix $\boldsymbol{X}$ and an invertible transform matrix $\boldsymbol{T}$, quantize $\boldsymbol{TX}$ rather than $\boldsymbol{X}$ itself, and undo the transform with $\boldsymbol{T}^{-1}$ afterwards. What makes a transform good is not a property of $\boldsymbol{X}$ alone: the error only matters through the matrix product (e.g., $\boldsymbol{W}^\top \boldsymbol{X}$) in which $\boldsymbol{X}$ is one factor. The WUSH transform (Chen et al., 2026) therefore builds $\boldsymbol{T}$ in closed form from second-order statistics of both factors of that product. According to their analysis, this choice is provably optimal for floating-point formats and asymptotically optimal for integer ones. The transform is block diagonal, and its block width is the same as the quantization group size. This keeps it cheap to apply to activations at inference time. We follow the original WUSH construction

[1] https://docs.vllm.ai/en/v0.29.0/api/vllm/model_executor/layers/quantization/turboquant/

and express it as a function of a Gram matrix and a Hessian, which is the form used throughout the rest of this paper.[2]

**The loss a transform has to control.** Let $\boldsymbol{W} \in \mathbb{R}^{d \times d_{\text{out}}}$ and $\boldsymbol{X} \in \mathbb{R}^{d \times d_{\text{batch}}}$ be the weights and activations of a linear layer, restricted to the $d$ input coordinates one block acts on. Chen et al. (2026) quantizes both factors at once. We will only ever quantize one of them, which is also the case to which their derivation reduces. We take that factor to be $\boldsymbol{X}$, keep $\boldsymbol{W}$ exact, and let $\mathcal{Q}$ be a quantizer, specified in Section 4. The round trip through the invertible transform $\boldsymbol{T} \in \mathbb{R}^{d \times d}$ leaves behind a perturbation $\boldsymbol{\varepsilon} = \boldsymbol{T}^{-1}\mathcal{Q}(\boldsymbol{T}\boldsymbol{X}) - \boldsymbol{X}$. With $\boldsymbol{H} = 2\boldsymbol{W}\boldsymbol{W}^\top$, the loss this perturbation causes at the layer output is $\ell = \left\|\boldsymbol{W}^\top\boldsymbol{\varepsilon}\right\|_{\text{F}}^2 = 2^{-1}\sum_j \boldsymbol{\varepsilon}_j^\top \boldsymbol{H}\boldsymbol{\varepsilon}_j$, a quadratic form in the columns $\boldsymbol{\varepsilon}_j$ of $\boldsymbol{\varepsilon}$. The matrix $\boldsymbol{H}$ is the Hessian of $\ell$ with respect to a column of $\boldsymbol{\varepsilon}$.

**The WUSH construction.** Let $\boldsymbol{M} = \boldsymbol{X}\boldsymbol{X}^\top$ be the Gram matrix of $\boldsymbol{X}$, $\boldsymbol{\mathcal{H}} \in \left\{\pm d^{-\frac{1}{2}}\right\}^{d \times d}$ a normalized (orthogonal) Hadamard matrix,[3] $\mathbf{I}$ the $d \times d$ identity, and $\gamma \geq 0$ a small damping ratio. The transform $\boldsymbol{T}$ applied to $\boldsymbol{X}$ is built as

$$\begin{aligned} \boldsymbol{L}\boldsymbol{L}^\top &= \boldsymbol{H} + \gamma d^{-1}\operatorname{tr}(\boldsymbol{H})\,\mathbf{I}, \qquad \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top = \boldsymbol{L}^\top\left(\boldsymbol{M} + \gamma d^{-1}\operatorname{tr}(\boldsymbol{M})\,\mathbf{I}\right)\boldsymbol{L}, \\ c &= (1+\gamma)^{\frac{1}{2}}\left(\operatorname{tr}(\boldsymbol{M})\right)^{\frac{1}{2}}\left(\operatorname{tr}\left(\boldsymbol{\Lambda}^{\frac{1}{2}}\right)\right)^{-\frac{1}{2}}, \qquad \boldsymbol{T} = c\,\boldsymbol{\mathcal{H}}\boldsymbol{\Lambda}^{-\frac{1}{4}}\boldsymbol{U}^\top\boldsymbol{L}^\top, \end{aligned} \tag{1}$$

where $\boldsymbol{L}\boldsymbol{L}^\top$ is the Cholesky decomposition and $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$ is the symmetric eigendecomposition. The scalar $c$ is introduced so that $\|\boldsymbol{T}\boldsymbol{X}\|_{\text{F}} = \|\boldsymbol{X}\|_{\text{F}}$ when $\gamma = 0$, and approximately so otherwise.

**Definition 1** *For symmetric positive semidefinite $\boldsymbol{M}, \boldsymbol{H} \in \mathbb{R}^{d \times d}$ and a damping ratio $\gamma \geq 0$ that makes both damped matrices positive definite, write $\textsc{Wush}(\boldsymbol{M}, \boldsymbol{H}, \gamma)$ for the transform $\boldsymbol{T}$ that Eq. (1) builds from them, in which $\boldsymbol{M}$ is the Gram matrix of the tensor being quantized and $\boldsymbol{H}$ is the Hessian of the loss with respect to that tensor's perturbation.*

We hold $\gamma$ fixed throughout and abbreviate $\textsc{Wush}(\boldsymbol{M}, \boldsymbol{H}, \gamma)$ to $\textsc{Wush}(\boldsymbol{M}, \boldsymbol{H})$. This construction is invariant to independent positive rescaling of its arguments, $\textsc{Wush}(\rho_{\text{M}}\boldsymbol{M}, \rho_{\text{H}}\boldsymbol{H}) = \textsc{Wush}(\boldsymbol{M}, \boldsymbol{H})$ for every $\rho_{\text{M}}, \rho_{\text{H}} > 0$, so how the two matrices are normalized never has to be tracked. What remains, for any tensor we wish to quantize, is to write down the Hessian of the loss with respect to its perturbation.

# 4 WUSH-KV

This section specializes in WUSH within the key/value cache of grouped-query attention. We first identify the cached tensors and formulate their transforms, then specify the quantizer and its guarantees, and finally describe cache management, storage, and computation costs.

## 4.1 KV Cache in Grouped-Query Attention

**Grouped-query attention (GQA).** In a model of dimension $d_{\text{model}}$, an attention module has $n_{\text{q}}$ query heads and $n_{\text{kv}}$ key/value heads of dimension $d$, so each key/value head is read by $n_{\text{q}}/n_{\text{kv}}$ query heads. A query head is indexed by the pair $(h, g)$: the key/value head $h$ it reads and its position $g = 1, \ldots, n_{\text{q}}/n_{\text{kv}}$ within that group. Let $\boldsymbol{X} \in \mathbb{R}^{d_{\text{model}} \times S}$ be the attention module input over a context of $S$ positions, one column each. Each head projects it with $\boldsymbol{W}_{\text{Q}(h,g)}, \boldsymbol{W}_{\text{K}(h)}, \boldsymbol{W}_{\text{V}(h)} \in \mathbb{R}^{d_{\text{model}} \times d}$. Let $\operatorname{Norm}$ denote any headwise normalization applied over the $d$ coordinates of each projected query or key. This includes root mean square (RMS) normalization and reduces to the identity when no such normalization is used. The query and key normalizations may have distinct parameters, which we suppress in the notation. The rotary embedding $\operatorname{RoPE}(\boldsymbol{A}) = [\boldsymbol{R}_1\boldsymbol{a}_1, \ldots, \boldsymbol{R}_S\boldsymbol{a}_S]$ rotates each column by an orthogonal $\boldsymbol{R}_s \in \mathbb{R}^{d \times d}$ fixed by its position $s$. Write $\tau$ for the softmax temperature and $\boldsymbol{\mathcal{M}} \in \{0, -\infty\}^{S \times S}$ for the causal mask, zero where a query is allowed to attend to a position and $-\infty$

[2] The form we give is an effectively equivalent rewrite of the original construction, which leads to simpler notation for the KV-cache derivation.

[3] We assume throughout that a Hadamard matrix of order $d$ exists.

elsewhere. The output projection $\boldsymbol{W}_{\mathrm{O}} \in \mathbb{R}^{n_{\mathrm{q}} d \times d_{\mathrm{model}}}$ splits into one row block $\boldsymbol{W}_{\mathrm{O}(h,g)} \in \mathbb{R}^{d \times d_{\mathrm{model}}}$ per query head. The attention module output is then

$$\begin{gathered}\boldsymbol{Q}_{h,g} = \mathrm{RoPE}\left(\mathrm{Norm}\left(\boldsymbol{W}_{\mathrm{Q}(h,g)}^{\top}\boldsymbol{X}\right)\right), \qquad \boldsymbol{K}_h = \mathrm{RoPE}\left(\mathrm{Norm}\left(\boldsymbol{W}_{\mathrm{K}(h)}^{\top}\boldsymbol{X}\right)\right), \\ \boldsymbol{V}_h = \boldsymbol{W}_{\mathrm{V}(h)}^{\top}\boldsymbol{X}, \qquad \boldsymbol{P}_{h,g} = \mathrm{softmax}\left(\tau^{-1}\boldsymbol{K}_h^{\top}\boldsymbol{Q}_{h,g} + \boldsymbol{\mathcal{M}}\right), \\ \boldsymbol{O}_{h,g} = \boldsymbol{V}_h\boldsymbol{P}_{h,g}, \qquad \boldsymbol{Y} = \sum_{h,g} \boldsymbol{W}_{\mathrm{O}(h,g)}^{\top}\boldsymbol{O}_{h,g},\end{gathered} \tag{2}$$

where the softmax acts on each column.

**Cached keys and values.** Autoregressive generation grows the context one position at a time and re-evaluates Eq. (2) at each new $S$. Of the three projections $\boldsymbol{Q}_{h,g}$, $\boldsymbol{K}_h$, and $\boldsymbol{V}_h$, only the keys and values must persist because every later position attends to all earlier ones, whereas the queries are used once and discarded. Each key/value head therefore caches $\boldsymbol{K}_h = [\boldsymbol{k}_{h,1}, \ldots, \boldsymbol{k}_{h,S}]$ and $\boldsymbol{V}_h = [\boldsymbol{v}_{h,1}, \ldots, \boldsymbol{v}_{h,S}]$ rather than recomputing them. Each new token appends one column to both caches, and later queries reuse the previously cached columns unchanged. Keys are stored after headwise normalization (when present) and rotary embedding, so $\boldsymbol{k}_{h,s}$ is read without further normalization or rotation.

### 4.2 WUSH-KV Transforms

**Transform placement.** A cached key is read only through $\boldsymbol{K}_h^{\top}\boldsymbol{Q}_{h,g}$ and a cached value only through $\boldsymbol{W}_{\mathrm{O}(h,g)}^{\top}\boldsymbol{V}_h$, and neither product changes when an invertible matrix is inserted between its factors,

$$\boldsymbol{K}_h^{\top}\boldsymbol{Q}_{h,g} = \left(\boldsymbol{T}_{\mathrm{K}(h)}\boldsymbol{K}_h\right)^{\top}\left(\boldsymbol{T}_{\mathrm{K}(h)}^{-\top}\boldsymbol{Q}_{h,g}\right), \quad \boldsymbol{W}_{\mathrm{O}(h,g)}^{\top}\boldsymbol{V}_h = \left(\boldsymbol{T}_{\mathrm{V}(h)}^{-\top}\boldsymbol{W}_{\mathrm{O}(h,g)}\right)^{\top}\left(\boldsymbol{T}_{\mathrm{V}(h)}\boldsymbol{V}_h\right). \tag{3}$$

With the quantizer of Section 4.3, we therefore cache $\mathcal{Q}\left(\boldsymbol{T}_{\mathrm{K}(h)}\boldsymbol{K}_h\right)$ and $\mathcal{Q}\left(\boldsymbol{T}_{\mathrm{V}(h)}\boldsymbol{V}_h\right)$ in place of the two tensors themselves. For each key/value head, we build one $d \times d$ transform for the keys and another for the values. The value-side transform is folded into the value and output projection weights offline before inference: we replace $\boldsymbol{W}_{\mathrm{V}(h)}$ with $\widehat{\boldsymbol{W}}_{\mathrm{V}(h)} = \boldsymbol{W}_{\mathrm{V}(h)}\boldsymbol{T}_{\mathrm{V}(h)}^{\top}$ and every $\boldsymbol{W}_{\mathrm{O}(h,g)}$ with $\widehat{\boldsymbol{W}}_{\mathrm{O}(h,g)} = \boldsymbol{T}_{\mathrm{V}(h)}^{-\top}\boldsymbol{W}_{\mathrm{O}(h,g)}$. The key side does not disappear into the projection weights because the inserted transforms act after Norm and the position-dependent RoPE, and a dense $\boldsymbol{T}_{\mathrm{K}(h)}$ cannot generally be moved across these operations.

This post-RoPE placement is deliberate. As detailed in Section A, fully folding the key transform and its query-side compensation into the projection weights requires each to commute with RoPE and its respective learned RMS normalization. These joint commutation constraints leave only paired sign-flip transforms, which are trivial because they do not redistribute coordinate magnitudes. A separate alternative is to place an unrestricted transform before RoPE rather than require it to commute with RoPE. This retains the full $d \times d$ freedom, but its compensation depends on the cached position. At each decoding step, this would require restoring and rotating every cached key or applying a different compensation at every cache position, adding position-dependent work across the cache and complicating the attention kernel. We therefore retain the full $d \times d$ transform after RoPE, accepting fixed online transforms for each new key and query in exchange for a simple cache representation.

**Transform formulation.** For the construction, we use local surrogate Hessians that retain the direct bilinear partner of each cached tensor: the queries of the group for a key and the output projection for a value. The transforms are

$$\begin{gathered}\boldsymbol{H}_{\mathrm{K}(h)} = 2\sum_{g} \boldsymbol{Q}_{h,g}\boldsymbol{Q}_{h,g}^{\top}, \qquad \boldsymbol{H}_{\mathrm{V}(h)} = 2\sum_{g} \boldsymbol{W}_{\mathrm{O}(h,g)}\boldsymbol{W}_{\mathrm{O}(h,g)}^{\top}, \\ \boldsymbol{T}_{\mathrm{K}(h)} = \textsc{Wush}\left(\boldsymbol{K}_h\boldsymbol{K}_h^{\top}, \boldsymbol{H}_{\mathrm{K}(h)}\right), \qquad \boldsymbol{T}_{\mathrm{V}(h)} = \textsc{Wush}\left(\boldsymbol{V}_h\boldsymbol{V}_h^{\top}, \boldsymbol{H}_{\mathrm{V}(h)}\right),\end{gathered} \tag{4}$$

with $\boldsymbol{K}_h\boldsymbol{K}_h^{\top}$ and $\boldsymbol{V}_h\boldsymbol{V}_h^{\top}$ the Gram matrices of the tensors being quantized. Algorithm 1 summarizes the offline calibration procedure for each attention module. We accumulate statistics over all token positions in each calibration sequence, then sum them across sequences to avoid storing all calibration activations and projections at once.

**Algorithm 1:** WUSH-KV calibration

**Input:** Weights $\boldsymbol{W}_{\mathrm{Q}}, \boldsymbol{W}_{\mathrm{K}}, \boldsymbol{W}_{\mathrm{V}}, \boldsymbol{W}_{\mathrm{O}}$, calibration sequences $\left\{\boldsymbol{X}^{(i)}\right\}_{i=1}^{n_{\mathrm{cal}}}$

**Output:** Key transforms $\boldsymbol{T}_{\mathrm{K}}$ and folded weights $\widehat{\boldsymbol{W}}_{\mathrm{V}}, \widehat{\boldsymbol{W}}_{\mathrm{O}}$

**for** $i = 1, \ldots, n_{\mathrm{cal}}$ independently **do**

$\quad \boldsymbol{Q}^{(i)} \leftarrow \mathrm{RoPE}\left(\mathrm{Norm}\left(\boldsymbol{W}_{\mathrm{Q}}^{\top}\boldsymbol{X}^{(i)}\right)\right); \quad \boldsymbol{K}^{(i)} \leftarrow \mathrm{RoPE}\left(\mathrm{Norm}\left(\boldsymbol{W}_{\mathrm{K}}^{\top}\boldsymbol{X}^{(i)}\right)\right)$

$\quad \boldsymbol{V}^{(i)} \leftarrow \boldsymbol{W}_{\mathrm{V}}^{\top}\boldsymbol{X}^{(i)}$

$\quad \boldsymbol{M}_{\mathrm{K}(h)}^{(i)} \leftarrow \boldsymbol{K}_h^{(i)}\boldsymbol{K}_h^{(i)\top} \quad \forall\, h; \quad \boldsymbol{M}_{\mathrm{V}(h)}^{(i)} \leftarrow \boldsymbol{V}_h^{(i)}\boldsymbol{V}_h^{(i)\top} \quad \forall\, h$

$\quad$ Compute Hessian matrices $\boldsymbol{H}_{\mathrm{K}(h)}^{(i)}, \boldsymbol{H}_{\mathrm{V}(h)}^{(i)}$ from $\boldsymbol{Q}^{(i)}, \boldsymbol{W}_{\mathrm{O}}$ and optionally $\boldsymbol{K}^{(i)}, \boldsymbol{V}^{(i)} \quad \forall\, h$

**end for**

$\boldsymbol{T}_{\mathrm{K}(h)} \leftarrow \textsc{Wush}\left(\sum_i \boldsymbol{M}_{\mathrm{K}(h)}^{(i)}, \sum_i \boldsymbol{H}_{\mathrm{K}(h)}^{(i)}\right) \quad \forall\, h$

$\boldsymbol{T}_{\mathrm{V}(h)} \leftarrow \textsc{Wush}\left(\sum_i \boldsymbol{M}_{\mathrm{V}(h)}^{(i)}, \sum_i \boldsymbol{H}_{\mathrm{V}(h)}^{(i)}\right) \quad \forall\, h$

$\widehat{\boldsymbol{W}}_{\mathrm{V}(h)} \leftarrow \boldsymbol{W}_{\mathrm{V}(h)}\boldsymbol{T}_{\mathrm{V}(h)}^{\top} \quad \forall\, h; \quad \widehat{\boldsymbol{W}}_{\mathrm{O}(h,g)} \leftarrow \boldsymbol{T}_{\mathrm{V}(h)}^{-\top}\boldsymbol{W}_{\mathrm{O}(h,g)} \quad \forall\, h, g$

### 4.3 Quantizer and Near-Optimality

We use per-token quantization: the $d$ channels of a key or value head form one group and share a scale. To analyze clipping, we use the projection step of QuEST (Panferov et al., 2025). For a nonzero group $\boldsymbol{x}$, it sets the step $\Delta$ from the group's root mean square (RMS) and reconstructs at bin centers,

$$\Delta = 2\alpha_b \left(2^b - 1\right)^{-1} d^{-\frac{1}{2}} \|\boldsymbol{x}\|, \qquad \mathcal{Q}(\boldsymbol{x}) = \Delta\left(\left\lfloor \Delta^{-1}\boldsymbol{x} \right\rfloor + 2^{-1}\right), \tag{5}$$

with the floor clamped to $\left[-2^{b-1}, 2^{b-1} - 1\right]$ and $\mathcal{Q}(\mathbf{0}) = \mathbf{0}$. The $2^b$ levels are evenly spaced between $\pm\alpha_b$ times the group RMS, and entries outside this range are clipped. The constant $\alpha_b$ minimizes the mean squared error of the corresponding fixed-scale grid on a standard Gaussian.

For the analysis, write $\ell(\boldsymbol{T})$ for the quadratic output loss from Section 3 after quantizing $\boldsymbol{TX}$ and undoing $\boldsymbol{T}$. We keep clipping exact and model rounding within the range by independent uniform noise, as specified in Section B.1. We call an invertible transform *sensitivity-balanced* when $\boldsymbol{T}^{-\top}\boldsymbol{H}\boldsymbol{T}^{-1}$ has equal diagonal entries.

**Theorem 2** *Let $\boldsymbol{M} = \boldsymbol{X}\boldsymbol{X}^{\top}$ and $\boldsymbol{H}$ be symmetric positive definite, and let $\boldsymbol{T}_\star$ be Eq. (1) with $\gamma = 0$. Under the Gaussian-tail and clipping-alignment conditions in Section B.3, with bounds independent of $d$ and $b$, every sensitivity-balanced invertible $\boldsymbol{T}$ satisfies*

$$\mathbb{E}\left[\ell\left(\boldsymbol{T}_\star\right)\right] \le \left(1 + O\left(\alpha_b^{-2}\right)\right)\mathbb{E}\left[\ell\left(\boldsymbol{T}\right)\right] \qquad \text{as } b \to \infty. \tag{6}$$

*The expectation is over rounding noise, and the constant depends only on the assumption bounds.*

The undamped WUSH transform is sensitivity-balanced by Eq. (20). Section B proves the result and gives an explicit finite-bit bound for $\alpha_b > 1$. The guarantee concerns the ideal transform, while the experiments use damping.

### 4.4 Full-Precision Cache Window

We combine the quantized cache with two full-precision windows and quantize newly eligible entries in batches. The first $S_{\mathrm{sink}}$ consecutive cache positions form a fixed full-precision sink window throughout generation. Among the remaining positions, the newest $S_{\mathrm{keep}}$ form a rolling recent window. Each new chunk is first appended in full precision and is used to compute its attention output. We then quantize the largest multiple of $S_{\mathrm{flush}}$ among the entries that lie outside both windows under the current cache length. Thus, $S_{\mathrm{sink}} + S_{\mathrm{keep}}$ entries per head remain persistently in full precision once the two windows no longer overlap, and fewer than $S_{\mathrm{flush}}$ additional entries can remain temporarily in full precision. Write $S_{\mathrm{quant}}$ for the inclusive right boundary of the quantized middle region, initialized to zero for an empty cache. Algorithm 2 gives this update for one attention module (some symbols are overloaded for transformed-coordinate counterparts).

**Algorithm 2:** WUSH-KV inference

**Input:** Weights $\boldsymbol{W}_{\mathrm{Q}}, \boldsymbol{W}_{\mathrm{K}}, \widehat{\boldsymbol{W}}_{\mathrm{V}}, \widehat{\boldsymbol{W}}_{\mathrm{O}}$, transforms $\boldsymbol{T}_{\mathrm{K}}$, cache $\boldsymbol{K}, \boldsymbol{V}$ in transformed coordinates with length $S$ and quantized boundary $S_{\mathrm{quant}}$, new activation chunk $\boldsymbol{X}'$ of length $S'$, window sizes $S_{\mathrm{sink}}, S_{\mathrm{keep}}$ with flush size $S_{\mathrm{flush}}$

**Output:** Attention output $\boldsymbol{Y}'$, updated cache $\boldsymbol{K}, \boldsymbol{V}$ of length $S$ and quantized boundary $S_{\mathrm{quant}}$

$\boldsymbol{Q}' \leftarrow \mathrm{RoPE}\left(\mathrm{Norm}\left(\boldsymbol{W}_{\mathrm{Q}}^{\top}\boldsymbol{X}'\right)\right)$; $\quad \boldsymbol{K}' \leftarrow \mathrm{RoPE}\left(\mathrm{Norm}\left(\boldsymbol{W}_{\mathrm{K}}^{\top}\boldsymbol{X}'\right)\right)$

$\boldsymbol{V}' \leftarrow \widehat{\boldsymbol{W}}_{\mathrm{V}}^{\top}\boldsymbol{X}'$

$\boldsymbol{Q}'_{h,g} \leftarrow \boldsymbol{T}_{\mathrm{K}(h)}^{-\top}\boldsymbol{Q}'_{h,g} \quad \forall\, h, g; \quad \boldsymbol{K}'_{h} \leftarrow \boldsymbol{T}_{\mathrm{K}(h)}\boldsymbol{K}'_{h} \quad \forall\, h$

Append $\boldsymbol{K}', \boldsymbol{V}'$ to $\boldsymbol{K}, \boldsymbol{V}$ in full precision; $\quad S \leftarrow S + S'$

Compute $\boldsymbol{O}'$ from $\boldsymbol{Q}', \boldsymbol{K}, \boldsymbol{V}$ using grouped-query attention (mixed precision)

$\boldsymbol{Y}' \leftarrow \widehat{\boldsymbol{W}}_{\mathrm{O}}^{\top}\boldsymbol{O}'$

$S_{\mathrm{start}} \leftarrow \max\left(S_{\mathrm{quant}}, S_{\mathrm{sink}}\right)$

$S_{\mathrm{quant}} \leftarrow \max\left(S_{\mathrm{quant}}, S_{\mathrm{start}} + \left\lfloor \left(S - S_{\mathrm{keep}} - S_{\mathrm{start}}\right)/S_{\mathrm{flush}} \right\rfloor S_{\mathrm{flush}}\right)$

**if** $S_{\mathrm{quant}} > S_{\mathrm{start}}$ **then**

  $\boldsymbol{k}_{h,s} \leftarrow \mathcal{Q}\left(\boldsymbol{k}_{h,s}\right) \quad \forall\, h,\ S_{\mathrm{start}} < s \leq S_{\mathrm{quant}}$

  $\boldsymbol{v}_{h,s} \leftarrow \mathcal{Q}\left(\boldsymbol{v}_{h,s}\right) \quad \forall\, h,\ S_{\mathrm{start}} < s \leq S_{\mathrm{quant}}$

**end if**

### 4.5 Storage and Computation Costs

For a model with $n_{\mathrm{layer}}$ attention modules, at sequence length $S$, the model caches $2n_{\mathrm{layer}}n_{\mathrm{kv}}dS$ key/value elements. Because the full-precision windows typically cover less than 1% of the maximum sequence length in the downstream tasks, they increase the effective cache quantization bitwidth only slightly. The method stores $n_{\mathrm{layer}}n_{\mathrm{kv}}$ key transforms of size $d \times d$, fixed once at calibration time and shared by every position. Their storage overhead is small relative to both the weights and the KV cache at practical sequence lengths. The value-side transform adds no online cost because both halves are folded into the weights. On the key side, each new key costs one $d \times d$ product, and so does each query, or $d^2$ multiply-accumulates per position for each of the $n_{\mathrm{kv}}$ key heads and the $n_{\mathrm{q}}$ query heads.

## 5 Experiments

We evaluate WUSH-KV at three levels: controlled attention-stage reconstruction error, WikiText-2 (Merity et al., 2017) perplexity, and downstream reasoning accuracy.

### 5.1 Attention-Stage Quantization Error

End-to-end metrics do not isolate the contribution of the KV transforms. We therefore conduct a controlled ablation of the transform choices, measuring attention-stage reconstruction error with keys and values quantized separately and jointly.

**Setup.** We study all 36 attention modules of Qwen3-8B. We calibrate on 128 FineWeb-Edu (Penedo et al., 2024) sequences and evaluate on 32 disjoint sequences, each of length 1024. We evaluate reconstruction at 2, 3, and 4 bits in a prefill-like setting. Every transform uses the QuEST quantizer in Eq. (5), which holds the quantization rule fixed and isolates the effect of the transform. The entire KV sequence is quantized without the full-precision cache windows in Section 4.4, which isolates the effect of the transforms on reconstruction error. Reference activations follow the clean BF16 model trajectory.

**Transforms.** We compare six key/value transform pairs: identity (I), random orthogonal (R), normalized Hadamard (H), OSCAR (Zhou et al., 2026), WUSH-KV (WUSH), and an attention-aware WUSH-KV variant (WUSH-A). This H choice reflects the practical TurboQuant-style implementations. H does not reproduce the complete TurboQuant codec because every method in this experiment uses the same QuEST quantizer, isolating the effect of the transform. WUSH, our proposed method, learns separate $\boldsymbol{T}_{\mathrm{K}}$ and $\boldsymbol{T}_{\mathrm{V}}$ from the Gram matrices and simple Hessians in Eq. (4). We use $\gamma = 10^{-2}$ for WUSH. WUSH-A uses the same construction and damping but replaces the simple Hessians with the attention-aware Hessians derived in Section C. The attention-aware Hessians require only

forward-pass quantities, so WUSH-A needs neither backpropagation nor Hessian estimation through automatic differentiation. An empirical-Fisher alternative could use language-model-loss gradients, but we do not evaluate it because backward-pass calibration is too expensive.

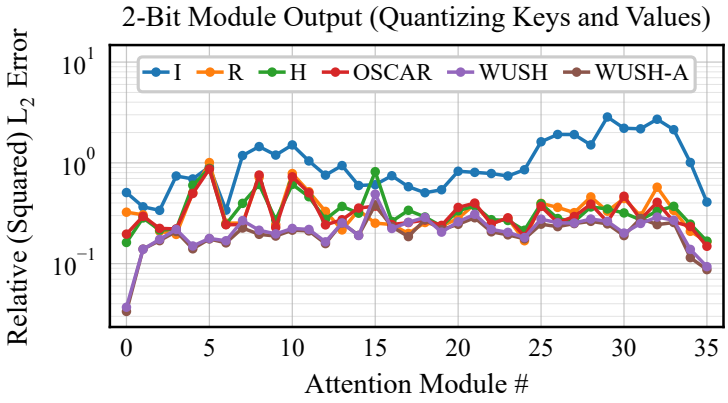


Figure 1: Relative (squared) $L_2$ module-output errors for I, R, H, OSCAR, WUSH, and WUSH-A with both keys and values quantized at 2-bit.

**Results.** For each module and measured quantity $\boldsymbol{A}$, we report $\|\widehat{\boldsymbol{A}} - \boldsymbol{A}\|_{\mathrm{F}}^2 / \|\boldsymbol{A}\|_{\mathrm{F}}^2$, summing the numerator and denominator separately over evaluation sequences. Figure 1 measures module-output error after the output projection and before the residual connection, with both keys and values quantized. The largest layerwise reductions occur in the first few attention modules. At 2-bit, when both keys and values are quantized, the geometric-mean module-output errors are $0.208$ for WUSH and $0.195$ for WUSH-A, compared with $0.325$ for H and $0.309$ for OSCAR. In Section D.1, the additional 2-bit plots in Figure 2 show that WUSH reduces error in the query-key dot products, softmax probabilities, and module output when only keys are quantized, as well as module-output error when only values are quantized. The 3-bit and 4-bit plots appear in Figure 3. Across bitwidths, WUSH and WUSH-A achieve lower module-output error than H and OSCAR for most attention modules in this setting and provide the strongest overall results among the evaluated transforms. WUSH-A offers only a modest improvement over WUSH here. We therefore use the simpler WUSH Hessians for the main method. WUSH-A remains a practical alternative for future models or settings in which attention-aware sensitivity provides a larger gain.

### 5.2 Perplexity Evaluations

We first test whether the lower attention-stage reconstruction error of WUSH translates into better language-model quality by measuring Qwen3-8B perplexity on WikiText-2.

**Setup.** For WUSH and OSCAR calibration, we use 128 FineWeb-Edu sequences of length 32,768. We use YaRN with factor $4$ for calibrations and evaluations on the Qwen3 model family. For WUSH, following Algorithm 1, an unquantized forward pass accumulates key Gram matrices after RMS normalization and RoPE, value Gram matrices, and simple per-head Hessians defined in Eq. (4). We then construct one key and one value transform for every attention module and KV head with $\gamma = 10^{-2}$. Calibration is performed once, and the resulting transforms are reused across all cache bitwidths. On a single NVIDIA L40S GPU, this one-time Qwen3-8B calibration takes about 12 minutes and peaks at 39 GiB of GPU memory. All quantized runs use the QuEST quantizer in Eq. (5), matching our theoretical analysis. We compare identity (I), random orthogonal (R), normalized Hadamard (H), OSCAR, and WUSH-KV (WUSH) at 4, 3, and 2 bits. For evaluation, we concatenate and tokenize the WikiText-2 test split, then divide it into non-overlapping sequences of length 2,048. Each sequence starts with an

Table 1: Qwen3-8B WikiText-2 perplexity. The full-precision baseline is 9.72.

| Transform | 4-Bit | 3-Bit | 2-Bit |
|---|---|---|---|
| I | 11.36 | 15.29 | 38.08 |
| R | 9.77 | 10.18 | 14.22 |
| H | 9.77 | 10.06 | 258.29 |
| OSCAR | 9.77 | 10.19 | 13.74 |
| WUSH | **9.73** | **9.82** | **10.51** |

empty cache and is processed in 16-token forward chunks. Perplexity is computed from the average next-token negative log-likelihood. For the quantized runs, we use the full-precision cache windows in Section 4.4 with $S_{\text{sink}} = 16$, $S_{\text{keep}} = 128$, and $S_{\text{flush}} = 16$.

**Results.** Table 1 shows that WUSH achieves the lowest perplexity among the quantized transforms at every bitwidth. OSCAR remains competitive at higher bitwidths, but WUSH's advantage grows under more aggressive quantization: at 2-bit, WUSH obtains $10.51$ perplexity compared with OSCAR's $13.74$. WUSH otherwise remains close to the full-precision baseline, while the other transforms degrade more sharply as the bitwidth decreases. The extreme 2-bit perplexity for H arises from spikes in the first attention module's key vectors that are concentrated in essentially one coordinate before the transform. The Hadamard transform spreads these spikes into nearly equal-magnitude coefficients close to the midpoints between adjacent 2-bit reconstruction levels, causing dense quantization error.

### 5.3 Downstream Benchmarks

We next evaluate whether WUSH-KV's relative advantage carries over to downstream reasoning accuracy. For these end-to-end evaluations, we integrate WUSH-KV into SGLang with the OSCAR quantizer (Zhou et al., 2026). This quantizer combines percentile clipping with asymmetric affine scaling and therefore does not readily admit the exact same proof as QuEST. Its adaptive zero point can use the available quantization levels more efficiently for shifted or asymmetric groups, while percentile clipping suppresses rare large errors. In our experiments, this tradeoff yields better end-to-end quality than symmetric QuEST, despite slightly higher average local squared error. Using the same quantizer also enables a direct comparison with the concurrent OSCAR method.

**Setup.** We evaluate Qwen3-4B-Thinking-2507, Qwen3-8B, and Qwen3-32B on AIME 2025 (Zhang & Team, 2025), MATH-500 (Lightman et al., 2024), GPQA Diamond (Rein et al., 2024), and LiveCodeBench v6 (Jain et al., 2025) with stochastic sampling over three seeds. We calibrate WUSH separately for each model on FineWeb-Edu, as in the perplexity evaluation, and reuse its transforms across downstream tasks. For the OSCAR baseline, we use its *released* GPQA Diamond calibrated transforms but were unable to exactly reproduce all scores reported in the concurrent preprint (Zhou et al., 2026), so we report our runs under the same SGLang evaluation pipeline. For a transformed token-head group $\boldsymbol{x} \in \mathbb{R}^d$ at bitwidth $b$, the OSCAR-style quantizer clips the coordinates at their $\kappa$-quantile, derives an affine scale $\Delta$ and zero point $z$ from the clipped range, rounds, and reconstructs,

$$\begin{gathered} x_{\min} = \max\left\{\min x_i, -\operatorname{quantile}_\kappa\left(|x_i|\right)\right\}, \quad x_{\max} = \min\left\{\max x_i, \operatorname{quantile}_\kappa\left(|x_i|\right)\right\}, \\ \Delta = \left(2^b - 1\right)^{-1}\left(x_{\max} - x_{\min}\right), \quad z = -\Delta^{-1} x_{\min}, \quad \mathcal{Q}\left(\boldsymbol{x}\right) = \Delta\left(\left\lfloor \Delta^{-1}\boldsymbol{x} + z\mathbf{1} \right\rceil - z\mathbf{1}\right). \end{gathered} \tag{7}$$

with $\operatorname{quantile}_\kappa\left(|x_i|\right)$ taken over $i = 1, \ldots, d$ and $\lfloor \cdot \rceil$ denoting round-to-nearest clamped to $\left[0, 2^b - 1\right]$. We use $\kappa_{\text{K}} = 0.96$ for all three models and $\kappa_{\text{V}} = 0.92$ for Qwen3-4B-Thinking-2507 and Qwen3-8B, increasing $\kappa_{\text{V}}$ to $0.96$ for Qwen3-32B. We reuse OSCAR's tuned clipping ratios for WUSH-KV without retuning them for the WUSH-transformed K/V distributions. These ratios may therefore be suboptimal for WUSH-KV, leaving potential headroom from WUSH-specific clipping optimization. We provide a quantizer ablation and calibration controls in Section D.2. All quantized benchmark runs use $S_{\text{sink}} = 64$, $S_{\text{keep}} = 256$, and $S_{\text{flush}} = 8$, matching OSCAR's BF16 sink and recent windows with page-sized INT2 flushing. The OSCAR SGLang integration updates cache storage before attention, while still using the current token or prefill chunk's K/V in BF16.

**Results.** Table 2 reports accuracy and generation behavior across model sizes. The budget-exhaustion rate is the fraction of samples that reach the task-specific generation limit without a valid answer. The limit is 65,536 tokens for MATH-500 and GPQA Diamond and 92,160 tokens for AIME 2025 and LiveCodeBench v6. For a strong direct comparison, we use OSCAR's selected 2-bit quantizer and cache policy without tuning them specifically for WUSH-KV. Even in this OSCAR-favored setup, WUSH-KV and OSCAR are close when OSCAR maintains strong accuracy, including across the 4B tasks and most 32B tasks. At 8B, WUSH-KV scores higher than OSCAR on all four benchmarks. When OSCAR degrades sharply, WUSH-KV can be substantially better, as on LiveCodeBench v6 for both 8B and 32B. Relative to BF16, WUSH-KV's degradation is comparatively consistent across model sizes, whereas OSCAR's varies more by model and task. WUSH-KV generates at most about $1.5\times$ as many tokens as BF16 in these tasks, well below the nominal $8\times$ reduction in storage per quantized KV entry from BF16 to INT2. Thus, the longer outputs do not erase the expected cache-memory benefit. Hadamard's large score losses and frequent budget exhaustion show that this fixed transform alone is insufficient for reliable 2-bit KV-cache quantization across these tasks.

Table 2: Downstream results with 2-bit KV caches. Each cell gives the task score (%) with its standard deviation over three seeds. The gray second line gives the mean generated-token count in thousands (k) and the budget-exhaustion rate. Bold marks the best quantized mean score and the shortest mean output for each model and task.

| Model | Method | AIME 2025 | MATH-500 | GPQA Diamond | LiveCodeBench v6 |
|---|---|---|---|---|---|
| Qwen3-4B-Thinking-2507 | BF16 | $77.8 \pm 5.1$<br>21.8k 0% | $89.5 \pm 1.5$<br>5.8k 0% | $69.9 \pm 2.0$<br>9.0k 0% | $57.1 \pm 0.6$<br>19.2k 0% |
| | TurboQuant-Style<br>*H Transform* | $0.0 \pm 0.0$<br>91.8k 100% | $18.7 \pm 1.0$<br>52.9k 76% | $16.2 \pm 1.8$<br>56.4k 68% | $0.0 \pm 0.0$<br>91.7k 99% |
| | OSCAR | $65.6 \pm 1.9$<br>**24.2k** 0% | $\mathbf{89.3} \pm 0.5$<br>6.6k 0% | $65.3 \pm 2.8$<br>10.1k 1% | $\mathbf{50.3} \pm 4.5$<br>25.0k 2% |
| | WUSH-KV<br>*WUSH Transform* | $\mathbf{67.8} \pm 6.9$<br>24.8k 1% | $88.3 \pm 1.1$<br>**6.6k** 0% | $\mathbf{66.7} \pm 1.0$<br>**9.8k** 0% | $49.5 \pm 1.2$<br>**24.4k** 3% |
| Qwen3-8B | BF16 | $67.8 \pm 1.9$<br>45.4k 11% | $95.1 \pm 0.3$<br>10.8k 1% | $57.4 \pm 3.8$<br>16.5k 1% | $47.8 \pm 2.2$<br>45.7k 11% |
| | TurboQuant-Style<br>*H Transform* | $0.0 \pm 0.0$<br>92.1k 98% | $21.9 \pm 0.1$<br>52.3k 75% | $30.1 \pm 2.8$<br>28.6k 29% | $1.0 \pm 0.3$<br>87.8k 89% |
| | OSCAR | $34.4 \pm 8.4$<br>63.0k 42% | $89.3 \pm 1.1$<br>16.0k 6% | $52.9 \pm 3.0$<br>18.3k 1% | $20.4 \pm 0.3$<br>73.8k 71% |
| | WUSH-KV<br>*WUSH Transform* | $\mathbf{46.7} \pm 3.3$<br>**53.4k** 12% | $\mathbf{92.9} \pm 0.1$<br>**12.2k** 2% | $\mathbf{54.5} \pm 2.3$<br>**16.3k** 1% | $\mathbf{35.2} \pm 0.7$<br>**53.9k** 26% |
| Qwen3-32B | BF16 | $74.4 \pm 5.1$<br>37.1k 6% | $94.7 \pm 1.2$<br>7.1k 1% | $66.7 \pm 2.7$<br>10.5k 0% | $61.9 \pm 0.3$<br>40.9k 3% |
| | TurboQuant-Style<br>*H Transform* | $1.1 \pm 1.9$<br>89.2k 91% | $0.0 \pm 0.0$<br>65.5k 100% | $0.0 \pm 0.0$<br>65.5k 100% | $1.0 \pm 1.2$<br>80.2k 73% |
| | OSCAR | $\mathbf{63.3} \pm 3.3$<br>**47.2k** 9% | $\mathbf{93.1} \pm 0.7$<br>**9.7k** 2% | $\mathbf{60.6} \pm 3.0$<br>13.6k 2% | $27.8 \pm 1.4$<br>71.4k 53% |
| | WUSH-KV<br>*WUSH Transform* | $60.0 \pm 5.8$<br>47.7k 9% | $91.9 \pm 1.4$<br>10.6k 2% | $59.3 \pm 2.0$<br>**12.1k** 4% | $\mathbf{43.1} \pm 2.3$<br>**54.3k** 9% |

We additionally evaluate long-context RULER NIAH (Hsieh et al., 2024) and MRCR (Vodrahalli et al., 2024), finding that WUSH-KV retains higher accuracy than OSCAR at the longest tested lengths. The detailed results are presented in Section D.3.

## 6 Conclusion

We presented WUSH-KV for low-bit KV-cache quantization in grouped-query attention. The method uses calibration data to construct WUSH transforms that are applied to cached keys and values before quantization. These transforms can be paired with per-token clipped quantizers. For QuEST, our analysis provides a clipping-aware near-optimality guarantee among sensitivity-balanced transforms under the stated noise model and assumptions. Our end-to-end downstream benchmarks pair the transforms with an OSCAR-style percentile-clipped affine quantizer. The results show that the WUSH transform can improve the quality of low-bit KV cache quantization. One limitation is that the dense key-side transform must remain online because it is applied after RoPE, which adds extra computation during inference. The value-side transform can be folded into the model weights, though. WUSH-KV supports the use of a more general calibration Hessian to construct the transforms. Future work could study cheaper structured transforms, stronger attention-aware sensitivity measures, and system-level optimization and throughput evaluation.

### AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic data sets, helping develop theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, implementing methods, assisting with translation, cleaning and reformatting dataset, supporting qualitative and thematic data analysis, and interpreting results. We reviewed all AI-assisted work. LLM-generated research ideas were checked for potential plagiarism and overlap with existing work through a manual literature survey. LLM-generated code was reviewed, modified, verified, and tested for correctness by the authors. AI-assisted mathematical content was independently checked by the authors for correctness and consistency with the arguments presented in the paper. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

### ETHICS STATEMENT

Throughout this work, we have strictly adhered to the ICLR Code of Ethics. All datasets utilized in our experiments are publicly available and widely recognized within the scientific community. We ensure that these datasets do not contain any personally identifiable information or sensitive content. Our work does not involve human subjects, animals, or any form of personal data collection. We have thoroughly considered potential dual-use concerns and do not foresee any harmful applications of our methods. There are no conflicts of interest to declare, and no external sponsorship influenced the outcomes of this research. All experiments were conducted with integrity and transparency.

### REPRODUCIBILITY STATEMENT

We are committed to ensuring that our work is transparent and reproducible. To facilitate this, clear explanations of any assumptions and a complete proof of the claims have been included in the main text and appendix. We also share the source code as part of the supplementary materials. The code is documented and includes instructions for setting up the environment, running the simulations, and reproducing the results presented in our paper. By making our resources openly available and providing detailed explanations, we aim to enable the research community to validate and build upon our findings.

## 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 A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 100213–100240. Curran Associates, Inc., 2024. doi: 10.52202/079017-3180. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b5b939436789f76f08b9d0da5e81af7c-Paper-Conference.pdf.

Ran Ben-Basat, Yaniv Ben-Itzhak, Gal Mendelson, Michael Mitzenmacher, Amit Portnoy, and Shay Vargaftik. A note on TurboQuant and the earlier DRIVE/EDEN line of work, 2026. URL https://arxiv.org/abs/2604.18555.

Jiale Chen, Vage Egiazarian, Roberto L. Castro, Torsten Hoefler, and Dan Alistarh. WUSH: Near-optimal adaptive transforms for LLM quantization. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ZsECxUkbKB.

Coleman Hooper, Sehoon Kim, Hiva Mohammadzadeh, Michael W. Mahoney, Yakun Sophia Shao, Kurt Keutzer, and Amir Gholami. KVQuant: Towards 10 million context length LLM inference with KV cache quantization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 1270–1303. Curran Associates, Inc., 2024. doi: 10.52202/079017-0040. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/028fcbcf85435d39a40c4d61b42c99a4-Paper-Conference.pdf.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the real context size of your long-context language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf.

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 Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 92009–92032, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/e5b1c0d4866f72393c522c8a00eed4eb-Paper-Conference.pdf.

Zirui Liu, Jiayi Yuan, Hongye Jin, Shaochen Zhong, Zhaozhuo Xu, Vladimir Braverman, Beidi Chen, and Xia Hu. KIVI: A tuning-free asymmetric 2bit quantization for KV cache. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32332–32344. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/liu24bz.html.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Andrei Panferov, Jiale Chen, Soroush Tabesh, Mahdi Nikdan, and Dan Alistarh. QuEST: Stable training of LLMs with 1-bit weights and activations. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 47820–47836. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/panferov25a.html.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 30811–30849. Curran Associates, Inc., 2024. doi: 10.52202/079017-0970. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/370df50ccfdf8bde18f8f9c2d9151bda-Paper-Datasets_and_Benchmarks_Track.pdf.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Alina Shutova, Vladimir Malinovskii, Vage Egiazarian, Denis Kuznedelev, Denis Mazur, Surkov Nikita, Ivan Ermakov, and Dan Alistarh. Cache me if you must: Adaptive key-value quantization for large language models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning*

*Research*, pp. 55451–55473. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/shutova25a.html.

Zunhai Su, Hanyu Wei, Zhe Chen, Wang Shen, Linge Li, Huangqi Yu, and Kehong Yuan. RotateKV: Accurate and robust 2-bit KV cache quantization for LLMs via outlier-aware adaptive rotations. In James Kwok (ed.), *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 6200–6208. International Joint Conferences on Artificial Intelligence Organization, 8 2025. doi: 10.24963/ijcai.2025/690. URL https://www.ijcai.org/proceedings/2025/690. Main Track.

Yuxuan Sun, Ruikang Liu, Haoli Bai, Han Bao, Kang Zhao, Yuening Li, Jiaxin Hu, Xianzhi Yu, Lu Hou, Chun Yuan, Xin Jiang, Wulong Liu, and Jun Yao. FlatQuant: Flatness matters for LLM quantization. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 57587–57613. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/sun25l.html.

Kiran Vodrahalli, Santiago Ontanon, Nilesh Tripuraneni, Kelvin Xu, Sanil Jain, Rakesh Shivanna, Jeffrey Hui, Nishanth Dikkala, Mehran Kazemi, Bahare Fatemi, Rohan Anil, Ethan Dyer, Siamak Shakeri, Roopali Vij, Harsh Mehta, Vinay Ramasesh, Quoc Le, Ed Chi, Yifeng Lu, Orhan Firat, Angeliki Lazaridou, Jean-Baptiste Lespiau, Nithya Attaluri, and Kate Olszewska. Michelangelo: Long context evaluations beyond haystacks via latent structure queries, 2024. URL https://arxiv.org/abs/2409.12640.

Amir Zandieh, Majid Daliri, Majid Hadian, and Vahab Mirrokni. TurboQuant: Online vector quantization with near-optimal distortion rate. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 56418–56439, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/5c802ef38ab6e366c2ea06eee554c088-Paper-Conference.pdf.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2025, 2025. URL https://huggingface.co/datasets/math-ai/aime25.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 62557–62583. Curran Associates, Inc., 2024. doi: 10.52202/079017-2000. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/724be4472168f31ba1c9ac630f15dec8-Paper-Conference.pdf.

Zhongzhu Zhou, Donglin Zhuang, Jisen Li, Ziyan Chen, Shuaiwen Leon Song, Ben Athiwaratkun, and Xiaoxia Wu. OSCAR: Offline spectral covariance-aware rotation for 2-bit KV cache quantization, 2026. URL https://arxiv.org/abs/2605.17757.

# A Alternative Key-Transform Placements

This appendix expands the design tradeoff summarized in Section 4. We first characterize transforms that can pass through RoPE and then consider unrestricted transforms placed before RoPE.

## A.1 RoPE-commuting transforms

On the basis that groups the rotary coordinate pairs, standard RoPE has the form

$$\boldsymbol{R}_s = \mathrm{blockdiag}_j\left(\boldsymbol{R}\left(s\omega_j\right)\right), \qquad \boldsymbol{R}\left(\theta\right) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{8}$$

For distinct nondegenerate rotary frequencies, the real invertible transforms that commute with every $\boldsymbol{R}_s$ are

$$\boldsymbol{T}_{\mathrm{K}(h)} = \mathrm{blockdiag}_j\left(\rho_{h,j}\boldsymbol{R}\left(\theta_{h,j}\right)\right), \qquad \boldsymbol{T}_{\mathrm{K}(h)}\boldsymbol{R}_s = \boldsymbol{R}_s\boldsymbol{T}_{\mathrm{K}(h)}, \qquad \rho_{h,j} > 0. \tag{9}$$

Each two-dimensional block lies in the span of $\left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right)$ and $\left(\begin{smallmatrix}0&-1\\1&0\end{smallmatrix}\right)$, while the distinct frequencies prevent mixing between rotary pairs. In the absence of post-projection normalization, such transforms and their inverse-transposes could pass through RoPE and be folded into the key and query projections.

A learned coordinatewise normalization further restricts exact folding. For example, write RMS normalization with gain $\boldsymbol{\nu}$ as

$$\mathrm{Norm}\left(\boldsymbol{x}\right) = d^{\frac{1}{2}} \left\|\boldsymbol{x}\right\|^{-1} \mathrm{diag}\left(\boldsymbol{\nu}\right) \boldsymbol{x}. \tag{10}$$

Moving a linear transform through this fixed normalization for every input requires

$$\boldsymbol{T}\,\mathrm{Norm}\left(\boldsymbol{x}\right) = \mathrm{Norm}\left(\boldsymbol{T}\boldsymbol{x}\right) \quad \forall\, \boldsymbol{x} \neq \boldsymbol{0} \quad \Longrightarrow \quad \boldsymbol{T}^{\top}\boldsymbol{T} = \mathbf{I}, \qquad \boldsymbol{T}\,\mathrm{diag}\left(\boldsymbol{\nu}\right) = \mathrm{diag}\left(\boldsymbol{\nu}\right)\boldsymbol{T}. \tag{11}$$

For nonzero gain entries, equality for every nonzero input forces $\boldsymbol{T}$ to commute with $\mathrm{diag}\left(\boldsymbol{\nu}\right)$ and preserve the RMS denominator ($\boldsymbol{T}^{\top}\boldsymbol{T} = \mathbf{I}$). The corresponding conditions must hold for the key transform and its inverse-transpose under the respective key and query normalizations. For generic learned gains, combining these conditions with RoPE commutation leaves only $\boldsymbol{T}_{\mathrm{K}(h),j} \in \left\{\left(\begin{smallmatrix}1&0\\0&1\end{smallmatrix}\right), \left(\begin{smallmatrix}-1&0\\0&-1\end{smallmatrix}\right)\right\}$. These paired sign flips leave coordinate magnitudes unchanged and therefore do not redistribute quantization difficulty for a symmetric quantizer such as Eq. (5).

If the learned normalization gains $\boldsymbol{\nu}$ are also allowed to change as part of the KV-cache quantization reparameterization, additional transforms may become foldable. However, such reparameterizations remain constrained by RoPE and by the structure of the normalization, and introduce additional design choices beyond folding a transform into fixed projection and normalization parameters. We therefore restrict our discussion to the fixed-normalization setting above.

### A.2 Full pre-RoPE transforms

A generic transform placed before RoPE restores the full $d \times d$ freedom but introduces position-dependent compensation. Let $\boldsymbol{k}_{h,s}$ and $\boldsymbol{q}_{h,g,t}$ denote the key at token position $s$ and the query at token position $t$, respectively. Since $\boldsymbol{R}_s$ is orthogonal, the normalized pre-RoPE key is $\boldsymbol{R}_s^{\top}\boldsymbol{k}_{h,s}$. If the cache instead stores $\boldsymbol{T}_{\mathrm{K}(h)}\boldsymbol{R}_s^{\top}\boldsymbol{k}_{h,s}$, then

$$\boldsymbol{k}_{h,s}^{\top}\boldsymbol{q}_{h,g,t} = \left(\boldsymbol{T}_{\mathrm{K}(h)}\boldsymbol{R}_s^{\top}\boldsymbol{k}_{h,s}\right)^{\top} \boldsymbol{T}_{\mathrm{K}(h)}^{-\top}\boldsymbol{R}_s^{\top}\boldsymbol{q}_{h,g,t}. \tag{12}$$

Decoding must therefore restore each cached key before applying its positional rotation or incorporate a different compensation for every cache position. The current post-RoPE placement instead retains a full transform and a simple cache representation at the cost of fixed online transforms for each new key and query.

## B Clipping-Aware Near-Optimality for the QuEST Quantizer

We prove Theorem 2 by starting from the unclipped rounding loss, which WUSH minimizes, and then controlling the effect of clipping. Clipping adds error outside the range and removes rounding noise from those coordinates. Both effects must be included in the comparison.

### B.1 Rounding and clipping errors

We keep $\boldsymbol{X}$ and $\boldsymbol{H}$ fixed, with $\boldsymbol{M} = \boldsymbol{X}\boldsymbol{X}^{\top}$ as in Section 3. Each column $\boldsymbol{x}_j$ is one quantization group. Following Chen et al. (2026), we approximate rounding with independent uniform noise, but keep clipping exact. This is a model of the rounding error, not an identity for the deterministic quantizer.

Fix an invertible $\boldsymbol{T}$. For group $j$, its RMS after the transform and its quantization step are

$$r_j = d^{-\frac{1}{2}} \left\|\boldsymbol{T}\boldsymbol{x}_j\right\|, \qquad \Delta_j = 2\alpha_b \left(2^b - 1\right)^{-1} r_j. \tag{13}$$

The clipping endpoints are $\pm\alpha_b r_j$. The error caused by clipping alone, before undoing the transform, is

$$\boldsymbol{e}_j = \mathrm{clip}\left(\boldsymbol{T}\boldsymbol{x}_j, -\alpha_b r_j, \alpha_b r_j\right) - \boldsymbol{T}\boldsymbol{x}_j. \tag{14}$$

Here clip acts coordinatewise, and $e_{j,i}$ denotes coordinate $i$ of group $j$. Inside the range, $e_{j,i} = 0$ and we replace the rounding error with $\Delta_j \xi_{j,i}$, where all $\xi_{j,i}$ are independent and uniformly distributed on $[-2^{-1}, 2^{-1}]$. Outside the range, the error is exactly $e_{j,i}$. Thus, the total round-trip perturbation $\boldsymbol{\varepsilon}_j$ satisfies

$$(\boldsymbol{T}\boldsymbol{\varepsilon}_j)_i = \begin{cases} \Delta_j \xi_{j,i}, & |(\boldsymbol{T}\boldsymbol{x}_j)_i| \le \alpha_b r_j, \\ e_{j,i}, & |(\boldsymbol{T}\boldsymbol{x}_j)_i| > \alpha_b r_j. \end{cases} \tag{15}$$

Zero groups have $r_j = \Delta_j = 0$ and zero error. All group quantities depend on $\boldsymbol{T}$. We suppress that dependence in their subscripts.

To measure errors in transformed coordinates, write $\boldsymbol{B_T} = \boldsymbol{T}^{-\top}\boldsymbol{H}\boldsymbol{T}^{-1}$. Then $\boldsymbol{\varepsilon}_j^\top \boldsymbol{H}\boldsymbol{\varepsilon}_j = (\boldsymbol{T}\boldsymbol{\varepsilon}_j)^\top \boldsymbol{B_T}(\boldsymbol{T}\boldsymbol{\varepsilon}_j)$. The diagonal entry $(\boldsymbol{B_T})_{ii}$ therefore measures sensitivity to an error in coordinate $i$. The modeled output loss is $\ell(\boldsymbol{T}) = 2^{-1}\sum_j \boldsymbol{\varepsilon}_j^\top \boldsymbol{H}\boldsymbol{\varepsilon}_j$. Expectations below are over the rounding noise.

**An unclipped reference.** Uniform rounding noise with step $\Delta_j$ has variance $12^{-1}\Delta_j^2$. Applying it to every coordinate without clipping would give the expected loss

$$\begin{aligned} \ell_{\mathrm{rnd}}(\boldsymbol{T}) &= 24^{-1}\sum_j \Delta_j^2 \operatorname{tr}(\boldsymbol{B_T}) \\ &= 6^{-1} d^{-1} \alpha_b^2 (2^b - 1)^{-2} \operatorname{tr}(\boldsymbol{T}\boldsymbol{M}\boldsymbol{T}^\top) \operatorname{tr}(\boldsymbol{B_T}). \end{aligned} \tag{16}$$

The last equality uses $\sum_j r_j^2 = d^{-1}\operatorname{tr}(\boldsymbol{T}\boldsymbol{M}\boldsymbol{T}^\top)$. This reference loss is a deterministic quantity, distinct from the clipped loss.

**The clipped loss.** Independence and zero mean remove all cross terms involving rounding noise, giving

$$\begin{aligned} \mathbb{E}[\ell(\boldsymbol{T})] = 24^{-1}\sum_j \Delta_j^2 \sum_{i:\,|(\boldsymbol{T}\boldsymbol{x}_j)_i| \le \alpha_b r_j} (\boldsymbol{B_T})_{ii} \\ + 2^{-1}\sum_j \boldsymbol{e}_j^\top \boldsymbol{B_T}\boldsymbol{e}_j. \end{aligned} \tag{17}$$

The first term counts rounding only inside the clipping range. The second measures the exact clipping error. Clipped coordinates no longer receive rounding noise, so the total can be smaller than the unclipped reference.

### B.2 The unclipped optimum

Only the trace product in Eq. (16) depends on the transform. The following result shows that WUSH minimizes it over all invertible transforms.

**Theorem 3** *Let $\boldsymbol{M}, \boldsymbol{H} \in \mathbb{R}^{d\times d}$ be symmetric positive definite, and let $\sigma_1, \dots, \sigma_d$ be the singular values of $\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{H}^{\frac{1}{2}}$.*[4] *Every invertible $\boldsymbol{T}$ satisfies*

$$\operatorname{tr}(\boldsymbol{T}\boldsymbol{M}\boldsymbol{T}^\top)\operatorname{tr}(\boldsymbol{T}^{-\top}\boldsymbol{H}\boldsymbol{T}^{-1}) \ge \left(\sum_i \sigma_i\right)^2, \tag{18}$$

*and Eq. (1) with $\gamma = 0$ attains the bound.*

**Proof** Take a singular value decomposition $\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{H}^{\frac{1}{2}} = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^\top$, where $\{\boldsymbol{u}_i\}$ and $\{\boldsymbol{v}_i\}$ are orthonormal bases. Inserting $\boldsymbol{T}^\top\boldsymbol{T}^{-\top} = \mathbf{I}$ and applying Cauchy-Schwarz gives

$$\begin{aligned} \sum_i \sigma_i &= \sum_i (\boldsymbol{T}\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{u}_i)^\top (\boldsymbol{T}^{-\top}\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{v}_i) \\ &\le \left(\sum_i \|\boldsymbol{T}\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{u}_i\|^2\right)^{\frac{1}{2}} \left(\sum_i \|\boldsymbol{T}^{-\top}\boldsymbol{H}^{\frac{1}{2}}\boldsymbol{v}_i\|^2\right)^{\frac{1}{2}}. \end{aligned} \tag{19}$$

[4] For a symmetric positive definite matrix, the square root is obtained by taking the positive square roots of its eigenvalues and keeping its eigenvectors.

The two sums of squared norms are the traces in Eq. (18), because the singular-vector bases are orthonormal. Squaring proves the lower bound.

For attainment, set $\gamma = 0$ in Eq. (1), so that $\boldsymbol{L}\boldsymbol{L}^\top = \boldsymbol{H}$ and $\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top = \boldsymbol{L}^\top \boldsymbol{M}\boldsymbol{L}$. Substituting the WUSH transform gives

$$\boldsymbol{T}\boldsymbol{M}\boldsymbol{T}^\top = c^2 \boldsymbol{\mathcal{H}}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{\mathcal{H}}^\top, \qquad \boldsymbol{T}^{-\top}\boldsymbol{H}\boldsymbol{T}^{-1} = c^{-2}\boldsymbol{\mathcal{H}}\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{\mathcal{H}}^\top. \tag{20}$$

The diagonal of $\boldsymbol{\Lambda}^{\frac{1}{2}}$ contains the singular values of $\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{L}$. These are the $\sigma_i$, since $\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{L}$ and $\boldsymbol{M}^{\frac{1}{2}}\boldsymbol{H}^{\frac{1}{2}}$ have the same product with their respective transposes. The trace product is therefore $(\operatorname{tr}(\boldsymbol{\Lambda}^{\frac{1}{2}}))^2 = (\sum_i \sigma_i)^2$. ■

An orthogonal transform leaves both traces unchanged and gives $\operatorname{tr}(\boldsymbol{M})\operatorname{tr}(\boldsymbol{H})$. This equals the minimum only when $\boldsymbol{M}$ and $\boldsymbol{H}$ are proportional.

For the clipped comparison, a transform is sensitivity-balanced when

$$(\boldsymbol{B}_{\boldsymbol{T}})_{ii} = d^{-1}\operatorname{tr}(\boldsymbol{B}_{\boldsymbol{T}}) \qquad i = 1, \ldots, d. \tag{21}$$

For the undamped WUSH transform this follows from Eq. (20), since every entry of $\boldsymbol{\mathcal{H}}$ has magnitude $d^{-\frac{1}{2}}$. Equal diagonal entries do not mean that $\boldsymbol{B}_{\boldsymbol{T}}$ is diagonal.

### B.3 The effect of clipping

We first bound clipping error for a scalar Gaussian. Two assumptions then transfer this bound to the WUSH-transformed groups. Finally, we bound how much clipping can reduce the rounding contribution of a competing transform.

**Gaussian clipping error.** Let $\zeta \sim \mathcal{N}(0,1)$. Clipping $\zeta$ to $[-\alpha, \alpha]$ incurs squared error $(|\zeta| - \alpha)_+^2$, where $(\cdot)_+$ takes the positive part. The next lemma concerns clipping alone, not the full quantization error.

**Lemma 4** *For the Gaussian-optimal clip constant $\alpha_b$ in Eq. (5),*

$$\mathbb{E}\left[(|\zeta| - \alpha_b)_+^2\right] \le (2^b - 1)^{-2}. \tag{22}$$

*Moreover, $\alpha_b \to \infty$ as $b \to \infty$.*

**Proof** For $\alpha > 0$, let $\mathcal{Q}_\alpha$ round to the nearest of the $2^b$ equally spaced levels from $-\alpha$ to $\alpha$. This scalar grid is fixed across samples, without a sample-dependent RMS scale. Write $\widehat{\zeta} = \mathcal{Q}_{\alpha_b}(\zeta)$ and $\Delta = 2\alpha_b(2^b - 1)^{-1}$ for the reconstruction and spacing at the optimal endpoint. All expectations below are over $\zeta$. We first use optimality to bound $\mathbb{E}[\zeta\widehat{\zeta}]$. We then relate this expectation to a weighted squared error that bounds the clipping error.

*1. Bound the expectation using the optimal scale.* If we change the endpoint from $\alpha_b$ to $\alpha$, each grid level is multiplied by $\alpha\alpha_b^{-1}$. Thus $\alpha\alpha_b^{-1}\widehat{\zeta}$ is a valid level on the new grid, although it need not be the nearest one to $\zeta$. For every $\alpha > 0$,

$$\mathbb{E}\left[(\widehat{\zeta} - \zeta)^2\right] \le \mathbb{E}\left[(\mathcal{Q}_\alpha(\zeta) - \zeta)^2\right] \le \mathbb{E}\left[(\alpha\alpha_b^{-1}\widehat{\zeta} - \zeta)^2\right]. \tag{23}$$

The first inequality uses the optimality of $\alpha_b$ among all endpoints. The second uses the fact that choosing the nearest level cannot be worse than choosing the rescaled old level. Both inequalities are equalities at $\alpha = \alpha_b$. Consequently, the last expectation is minimized there. Keeping $\widehat{\zeta}$ fixed and expanding that expectation gives the quadratic

$$\mathbb{E}\left[(\alpha\alpha_b^{-1}\widehat{\zeta} - \zeta)^2\right] = \alpha^2\alpha_b^{-2}\mathbb{E}[\widehat{\zeta}^2] - 2\alpha\alpha_b^{-1}\mathbb{E}[\zeta\widehat{\zeta}] + 1. \tag{24}$$

Here the last term is $\mathbb{E}[\zeta^2] = 1$. The derivative with respect to $\alpha$ vanishes at the positive minimizer $\alpha_b$, so

$$0 = 2\alpha_b^{-1}\left(\mathbb{E}[\widehat{\zeta}^2] - \mathbb{E}[\zeta\widehat{\zeta}]\right) \qquad \Longrightarrow \qquad \mathbb{E}[\widehat{\zeta}^2] = \mathbb{E}[\zeta\widehat{\zeta}]. \tag{25}$$

This equality is what makes Cauchy-Schwarz useful. Both second moments are finite, since $\zeta$ is standard Gaussian and $|\widehat{\zeta}| \leq \alpha_b$. Cauchy-Schwarz does not require the two variables to be independent and gives

$$\left|\mathbb{E}[\zeta\widehat{\zeta}]\right|^2 \leq \mathbb{E}[\zeta^2]\,\mathbb{E}[\widehat{\zeta}^2] = \mathbb{E}[\widehat{\zeta}^2]. \tag{26}$$

Substituting Eq. (25) into the left-hand side yields $(\mathbb{E}[\widehat{\zeta}^2])^2 \leq \mathbb{E}[\widehat{\zeta}^2]$. This second moment is nonnegative. If it is zero, it is already at most one. Otherwise, dividing by it gives $\mathbb{E}[\widehat{\zeta}^2] \leq 1$. Using Eq. (25) once more, we obtain

$$0 \leq \mathbb{E}[\zeta\widehat{\zeta}] = \mathbb{E}[\widehat{\zeta}^2] \leq 1. \tag{27}$$

Thus, the upper bound of one follows from optimality and Cauchy-Schwarz together, not from Cauchy-Schwarz alone.

*2. Relate this expectation to a squared error.* The moment bound in Eq. (27) does not yet bound clipping. For that, we use the Gaussian density $\phi$, whose derivative satisfies $\phi'(x) = -x\phi(x)$. Consider one quantization bin with reconstruction level $a$. On this bin, $a$ is constant, and the quantization error is $a - x$. Define

$$F_a(x) = a\left(8^{-1}\Delta^2 - 2^{-1}(x-a)^2\right), \qquad F_a'(x) = a(a-x). \tag{28}$$

The constant $8^{-1}\Delta^2$ is chosen to remove the boundary terms in integration by parts. Indeed, every finite endpoint of the bin is a midpoint between adjacent levels, where $|x - a| = 2^{-1}\Delta$ and hence $F_a(x) = 0$. For the two outer bins, $F_a(x)\phi(x)$ also tends to zero at the infinite endpoint because the Gaussian density decays faster than this quadratic grows. It follows that

$$\begin{aligned}\int_{\text{bin}} a(a-x)\phi(x)\,\mathrm{d}x &= [F_a(x)\phi(x)]_{\text{endpoints}} - \int_{\text{bin}} F_a(x)\phi'(x)\,\mathrm{d}x \\ &= \int_{\text{bin}} xF_a(x)\phi(x)\,\mathrm{d}x \\ &= 8^{-1}\Delta^2\int_{\text{bin}} xa\phi(x)\,\mathrm{d}x - 2^{-1}\int_{\text{bin}} xa(x-a)^2\phi(x)\,\mathrm{d}x.\end{aligned} \tag{29}$$

Now sum this equality over all bins. On the bin with level $a$, the reconstruction $\widehat{\zeta}$ equals $a$. The three integrals therefore become

$$\mathbb{E}[\widehat{\zeta}(\widehat{\zeta} - \zeta)] = 8^{-1}\Delta^2\mathbb{E}[\zeta\widehat{\zeta}] - 2^{-1}\mathbb{E}\left[\zeta\widehat{\zeta}(\widehat{\zeta} - \zeta)^2\right]. \tag{30}$$

The left-hand side is zero by Eq. (25). Moving the last term to the other side and multiplying by two gives the equality below. The inequality then follows by substituting the upper bound $\mathbb{E}[\zeta\widehat{\zeta}] \leq 1$ from Eq. (27).

$$\mathbb{E}\left[\zeta\widehat{\zeta}(\widehat{\zeta} - \zeta)^2\right] = 4^{-1}\Delta^2\mathbb{E}[\zeta\widehat{\zeta}] \leq 4^{-1}\Delta^2. \tag{31}$$

This is a bound on the total squared quantization error weighted by $\zeta\widehat{\zeta}$. The weight lets us extract the clipping error in the next step.

*3. Keep the contribution from clipped samples.* The symmetric grid reconstructs each nonzero input with the same sign, so $\zeta\widehat{\zeta} \geq 0$ everywhere. When $|\zeta| > \alpha_b$, the reconstruction is the endpoint with that sign. Thus $\zeta\widehat{\zeta} = \alpha_b|\zeta| \geq \alpha_b^2$ and $(\widehat{\zeta} - \zeta)^2 = (|\zeta| - \alpha_b)^2$. When $|\zeta| \leq \alpha_b$, the clipping error is zero and the weighted squared error is nonnegative. Combining these two cases gives the pointwise inequality

$$\alpha_b^2(|\zeta| - \alpha_b)_+^2 \leq \zeta\widehat{\zeta}(\widehat{\zeta} - \zeta)^2. \tag{32}$$

Taking expectations, applying Eq. (31), and then multiplying by $\alpha_b^{-2}$ yields

$$\mathbb{E}\left[(|\zeta| - \alpha_b)_+^2\right] \leq \alpha_b^{-2}\mathbb{E}\left[\zeta\widehat{\zeta}(\widehat{\zeta} - \zeta)^2\right] \leq 4^{-1}\alpha_b^{-2}\Delta^2 = (2^b - 1)^{-2}. \tag{33}$$

The last equality uses $\Delta = 2\alpha_b(2^b - 1)^{-1}$ and proves Eq. (22).

*4. Show that the optimal endpoint grows with bitwidth.* Consider the candidate endpoint $\alpha = \sqrt{b}$. Inside its clipping range, the error is at most half the grid spacing, or $\sqrt{b}(2^b - 1)^{-1}$. Outside the range, the error is exactly the distance to the nearest endpoint. Optimality of $\alpha_b$ therefore gives

$$\mathbb{E}\left[(\widehat{\zeta} - \zeta)^2\right] \leq \mathbb{E}\left[(\mathcal{Q}_{\sqrt{b}}(\zeta) - \zeta)^2\right] \leq b(2^b - 1)^{-2} + \mathbb{E}\left[(|\zeta| - \sqrt{b})_+^2\right] \longrightarrow 0. \tag{34}$$

The first term tends to zero because the number of levels grows exponentially in $b$. The second tends to zero by dominated convergence since its integrand tends pointwise to zero and is bounded by the integrable random variable $\zeta^2$. If $\alpha_b$ stayed bounded along a subsequence, the Gaussian tail beyond a fixed finite endpoint would give a positive lower bound on the clipping error along that subsequence. The total quantization error is at least its clipping contribution, contradicting Eq. (34). Hence $\alpha_b \to \infty$. ■

**Conditions on the transformed groups.** For the next two conditions, evaluate $r_j$ and $\boldsymbol{e}_j$ at $\boldsymbol{T}_\star$ and write $\boldsymbol{B}_\star = \boldsymbol{B}_{\boldsymbol{T}_\star}$. Dividing each nonzero transformed group by its own RMS gives

$$\widetilde{\boldsymbol{x}}_j = r_j^{-1} \boldsymbol{T}_\star \boldsymbol{x}_j, \qquad \|\widetilde{\boldsymbol{x}}_j\|^2 = d. \tag{35}$$

Set $\widetilde{\boldsymbol{x}}_j = \boldsymbol{0}$ for a zero group. Let $C_{\mathrm{tail}}, C_{\mathrm{align}} \geq 1$ be fixed bounds for the following conditions.

The first condition bounds the fraction of normalized entries above each threshold by a multiple of the Gaussian tail,

$$\left(d \sum_j r_j^2\right)^{-1} \sum_j r_j^2 \sum_{i=1}^{d} \mathbb{1}\{|\widetilde{x}_{j,i}| > \alpha\} \leq C_{\mathrm{tail}}\, \mathbb{P}(|\zeta| > \alpha) \qquad \forall\, \alpha \geq \alpha_b. \tag{36}$$

The weights $r_j^2$ account for each group's contribution to squared error. The second condition bounds how strongly clipping errors align with sensitive directions,

$$\sum_j \boldsymbol{e}_j^\top \boldsymbol{B}_\star \boldsymbol{e}_j \leq C_{\mathrm{align}} d^{-1} \operatorname{tr}(\boldsymbol{B}_\star) \sum_j \|\boldsymbol{e}_j\|^2. \tag{37}$$

This controls the direction of the clipping error, not its magnitude. It does not require independent errors and does not follow from equal diagonal entries alone. Both conditions concern WUSH only. A competing transform needs only to be sensitivity-balanced. The bounds are independent of $d$, and must also remain fixed across bitwidths for the asymptotic statement in Theorem 2.

**Upper bound at WUSH.** Integrating Eq. (36) against $2(\alpha - \alpha_b)\,\mathrm{d}\alpha$ over $\alpha \geq \alpha_b$ converts the tail bound into a bound on squared clipping error. Then Eq. (22) gives

$$\begin{aligned} \sum_j \|\boldsymbol{e}_j\|^2 &= \sum_j r_j^2 \sum_i (|\widetilde{x}_{j,i}| - \alpha_b)_+^2 \\ &\leq C_{\mathrm{tail}}\, d \sum_j r_j^2\, \mathbb{E}\left[(|\zeta| - \alpha_b)_+^2\right] \\ &\leq C_{\mathrm{tail}}\, d (2^b - 1)^{-2} \sum_j r_j^2. \end{aligned} \tag{38}$$

By Eq. (37), the clipping contribution to the loss is therefore at most

$$\begin{aligned} 2^{-1} \sum_j \boldsymbol{e}_j^\top \boldsymbol{B}_\star \boldsymbol{e}_j &\leq 2^{-1} C_{\mathrm{tail}} C_{\mathrm{align}} (2^b - 1)^{-2} \operatorname{tr}(\boldsymbol{B}_\star) \sum_j r_j^2 \\ &= 3 C_{\mathrm{tail}} C_{\mathrm{align}} \alpha_b^{-2} \ell_{\mathrm{rnd}}(\boldsymbol{T}_\star). \end{aligned} \tag{39}$$

The rounding contribution cannot exceed the unclipped reference, so Eq. (17) yields

$$\mathbb{E}[\ell(\boldsymbol{T}_\star)] \leq \left(1 + 3 C_{\mathrm{tail}} C_{\mathrm{align}} \alpha_b^{-2}\right) \ell_{\mathrm{rnd}}(\boldsymbol{T}_\star). \tag{40}$$

**Lower bound for a competing transform.** Now take any sensitivity-balanced invertible $\boldsymbol{T}$. For each nonzero group, $\|\boldsymbol{T}\boldsymbol{x}_j\|^2 = d r_j^2$, so at most $d\alpha_b^{-2}$ coordinates can exceed $\alpha_b r_j$ in magnitude. At least a fraction $1 - \alpha_b^{-2}$ therefore retain their rounding noise. All coordinates have the same sensitivity

by Eq. (21), so they retain at least that fraction of the reference loss. The clipping contribution is nonnegative, giving

$$\mathbb{E}[\ell(\boldsymbol{T})] \geq \left(1 - \alpha_b^{-2}\right) \ell_{\mathrm{rnd}}(\boldsymbol{T}). \tag{41}$$

Zero groups contribute no loss to either side. Finally, Theorem 3 gives $\ell_{\mathrm{rnd}}(\boldsymbol{T}_\star) \leq \ell_{\mathrm{rnd}}(\boldsymbol{T})$. Combining the upper and lower bounds proves, for $\alpha_b > 1$,

$$\mathbb{E}[\ell(\boldsymbol{T}_\star)] \leq \left(1 + (1 + 3C_{\mathrm{tail}}C_{\mathrm{align}})(\alpha_b^2 - 1)^{-1}\right) \mathbb{E}[\ell(\boldsymbol{T})]. \tag{42}$$

For fixed assumption bounds, this factor is independent of the group width. Since $\alpha_b \to \infty$, it is $1 + O(\alpha_b^{-2})$, proving Theorem 2. The comparison remains restricted to sensitivity-balanced transforms because balancing an arbitrary transform can change its clipped loss.

# C Exact Hessians for Grouped-Query Attention

This appendix derives attention-aware Hessians for perturbations of the queries, keys, and values in grouped-query attention. Unlike the per-head Hessians in Eq. (4), these exact expressions depend on the position of the perturbed vector. The final subsection aggregates the key and value Hessians to construct WUSH-A. The query case is included for completeness.

## C.1 Setup

Fix a key/value head $h$, and let $g = 1, \ldots, n_{\mathrm{q}}/n_{\mathrm{kv}}$ index the query heads that read it. We allow the query/key and value head dimensions to differ in this appendix, writing them as $d_{\mathrm{k}}$ and $d_{\mathrm{v}}$ (the main text takes both to be $d$). Write

$$\begin{aligned} \boldsymbol{Q}_{h,g} &= [\boldsymbol{q}_{h,g,1}, \ldots, \boldsymbol{q}_{h,g,S}] \in \mathbb{R}^{d_{\mathrm{k}} \times S}, \\ \boldsymbol{K}_h = [\boldsymbol{k}_{h,1}, \ldots, \boldsymbol{k}_{h,S}] \in \mathbb{R}^{d_{\mathrm{k}} \times S}, &\qquad \boldsymbol{V}_h = [\boldsymbol{v}_{h,1}, \ldots, \boldsymbol{v}_{h,S}] \in \mathbb{R}^{d_{\mathrm{v}} \times S}. \end{aligned} \tag{43}$$

The query and key vectors are taken after $\mathrm{Norm}$ and $\mathrm{RoPE}$, as in Eq. (2). Let $\boldsymbol{p}_{h,g,t}$ be column $t$ of $\boldsymbol{P}_{h,g}$, with entry $p_{h,g,t,s}$ at cache position $s$, and let $\boldsymbol{o}_{h,g,t} = \boldsymbol{V}_h \boldsymbol{p}_{h,g,t}$ be the corresponding head output. The causal mask gives $p_{h,g,t,s} = 0$ whenever $s > t$. The output-projection blocks $\boldsymbol{W}_{\mathrm{O}(h,g)} \in \mathbb{R}^{d_{\mathrm{v}} \times d_{\mathrm{model}}}$ induce the head-coupling matrices

$$\boldsymbol{\Gamma}_{h,g,g'} = \boldsymbol{W}_{\mathrm{O}(h,g)} \boldsymbol{W}_{\mathrm{O}(h,g')}^\top \in \mathbb{R}^{d_{\mathrm{v}} \times d_{\mathrm{v}}}. \tag{44}$$

The diagonal block $\boldsymbol{\Gamma}_{h,g,g}$ measures the sensitivity of query head $(h, g)$ after output projection, while $\boldsymbol{\Gamma}_{h,g,g'}$ for $g \neq g'$ couples two query heads that share $h$. Moreover, $\boldsymbol{\Gamma}_{h,g,g'}^\top = \boldsymbol{\Gamma}_{h,g',g}$.

## C.2 Hessian from Attention-Output Derivatives

To measure the local sensitivity of the attention output to quantization error, perturb one query, key, or value vector by $\boldsymbol{\delta}$. Write $\boldsymbol{Y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_S]$ and $\boldsymbol{Y}'(\boldsymbol{\delta}) = [\boldsymbol{y}'_1(\boldsymbol{\delta}), \ldots, \boldsymbol{y}'_S(\boldsymbol{\delta})]$ for the unperturbed and perturbed attention outputs, respectively, and define

$$\ell(\boldsymbol{\delta}) = \|\boldsymbol{Y}'(\boldsymbol{\delta}) - \boldsymbol{Y}\|_{\mathrm{F}}^2. \tag{45}$$

Let $y'_{t,j}(\boldsymbol{\delta})$ and $y_{t,j}$ denote coordinate $j$ of $\boldsymbol{y}'_t(\boldsymbol{\delta})$ and $\boldsymbol{y}_t$, respectively. Differentiating Eq. (45) once gives

$$\frac{\partial \ell}{\partial \boldsymbol{\delta}} = 2 \sum_{t=1}^{S} \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)^\top (\boldsymbol{y}'_t - \boldsymbol{y}_t). \tag{46}$$

Differentiating again gives the full Hessian

$$\frac{\partial^2 \ell}{\partial \boldsymbol{\delta} \partial \boldsymbol{\delta}^\top} = 2 \sum_{t=1}^{S} \left( \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)^\top \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right) + \sum_{j=1}^{d_{\mathrm{model}}} (y'_{t,j} - y_{t,j}) \frac{\partial^2 y'_{t,j}}{\partial \boldsymbol{\delta} \partial \boldsymbol{\delta}^\top} \right). \tag{47}$$

At zero perturbation,

$$\boldsymbol{Y}'(\boldsymbol{0}) - \boldsymbol{Y} = \boldsymbol{0}. \tag{48}$$

This single fact forces the gradient in Eq. (46) to vanish and, independently, removes the second term in Eq. (47). Therefore,

$$\begin{aligned} \left.\frac{\partial \ell}{\partial \boldsymbol{\delta}}\right|_{\boldsymbol{\delta}=\mathbf{0}} &= \mathbf{0}, \\ \left.\frac{\partial^2 \ell}{\partial \boldsymbol{\delta} \partial \boldsymbol{\delta}^\top}\right|_{\boldsymbol{\delta}=\mathbf{0}} &= 2 \sum_{t=1}^{S} \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)^\top \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)\Bigg|_{\boldsymbol{\delta}=\mathbf{0}}. \end{aligned} \tag{49}$$

### C.3 Query Hessian

Perturb the query $\boldsymbol{q}_{h,g,t}$ by $\boldsymbol{\delta} \in \mathbb{R}^{d_\mathrm{k}}$,

$$\boldsymbol{q}_{h,g,t} \mapsto \boldsymbol{q}_{h,g,t} + \boldsymbol{\delta}. \tag{50}$$

Only the query head $(h, g)$ at position $t$ changes. The derivative of the logit vector with respect to $\boldsymbol{\delta}^\top$ is $\tau^{-1}\boldsymbol{K}_h^\top$, while the softmax derivative is $\operatorname{diag}(\boldsymbol{p}_{h,g,t}) - \boldsymbol{p}_{h,g,t}\boldsymbol{p}_{h,g,t}^\top$. Chaining them gives the head-output and attention-output derivatives

$$\begin{aligned} \left.\frac{\partial \boldsymbol{o}'_{h,g,t}}{\partial \boldsymbol{\delta}^\top}\right|_{\boldsymbol{\delta}=\mathbf{0}} &= \tau^{-1}\boldsymbol{V}_h \left(\operatorname{diag}(\boldsymbol{p}_{h,g,t}) - \boldsymbol{p}_{h,g,t}\boldsymbol{p}_{h,g,t}^\top\right) \boldsymbol{K}_h^\top, \\ \left.\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right|_{\boldsymbol{\delta}=\mathbf{0}} &= \tau^{-1}\boldsymbol{W}_{\mathrm{O}(h,g)}^\top \boldsymbol{V}_h \left(\operatorname{diag}(\boldsymbol{p}_{h,g,t}) - \boldsymbol{p}_{h,g,t}\boldsymbol{p}_{h,g,t}^\top\right) \boldsymbol{K}_h^\top. \end{aligned} \tag{51}$$

Substituting the attention-output derivative into Eq. (49) gives the $d_\mathrm{k} \times d_\mathrm{k}$ query Hessian

$$\begin{aligned} &\boldsymbol{H}_{\mathrm{Q}(h,g,t)} = 2 \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)^\top \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)\Bigg|_{\boldsymbol{\delta}=\mathbf{0}} \\ &= 2\tau^{-2}\boldsymbol{K}_h \left(\operatorname{diag}(\boldsymbol{p}_{h,g,t}) - \boldsymbol{p}_{h,g,t}\boldsymbol{p}_{h,g,t}^\top\right) \boldsymbol{V}_h^\top \boldsymbol{\Gamma}_{h,g,g} \boldsymbol{V}_h \left(\operatorname{diag}(\boldsymbol{p}_{h,g,t}) - \boldsymbol{p}_{h,g,t}\boldsymbol{p}_{h,g,t}^\top\right) \boldsymbol{K}_h^\top. \end{aligned} \tag{52}$$

### C.4 Key Hessian

Perturb the shared key $\boldsymbol{k}_{h,s}$ by $\boldsymbol{\delta} \in \mathbb{R}^{d_\mathrm{k}}$,

$$\boldsymbol{k}_{h,s} \mapsto \boldsymbol{k}_{h,s} + \boldsymbol{\delta}. \tag{53}$$

This changes every query head that reads $h$ and every causal query position $t \geq s$. Differentiating the softmax with respect to the logit at cache position $s$ and summing against $\boldsymbol{V}_h$ gives the residual $\boldsymbol{v}_{h,s} - \boldsymbol{o}_{h,g,t}$. For one such head and position, the head-output derivative is

$$\left.\frac{\partial \boldsymbol{o}'_{h,g,t}}{\partial \boldsymbol{\delta}^\top}\right|_{\boldsymbol{\delta}=\mathbf{0}} = \tau^{-1} p_{h,g,t,s} \left(\boldsymbol{v}_{h,s} - \boldsymbol{o}_{h,g,t}\right) \boldsymbol{q}_{h,g,t}^\top. \tag{54}$$

The corresponding attention-output derivative is

$$\left.\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right|_{\boldsymbol{\delta}=\mathbf{0}} = \tau^{-1} \sum_g p_{h,g,t,s} \boldsymbol{W}_{\mathrm{O}(h,g)}^\top \left(\boldsymbol{v}_{h,s} - \boldsymbol{o}_{h,g,t}\right) \boldsymbol{q}_{h,g,t}^\top. \tag{55}$$

Substituting the attention-output derivative into Eq. (49) and expanding the head interactions gives the $d_\mathrm{k} \times d_\mathrm{k}$ key Hessian

$$\begin{aligned} &\boldsymbol{H}_{\mathrm{K}(h,s)} = 2 \sum_{t=1}^{S} \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)^\top \left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)\Bigg|_{\boldsymbol{\delta}=\mathbf{0}} \\ &= 2\tau^{-2} \sum_{t=1}^{S} \sum_{g,g'} p_{h,g,t,s} p_{h,g',t,s} \left(\boldsymbol{v}_{h,s} - \boldsymbol{o}_{h,g,t}\right)^\top \boldsymbol{\Gamma}_{h,g,g'} \left(\boldsymbol{v}_{h,s} - \boldsymbol{o}_{h,g',t}\right) \boldsymbol{q}_{h,g,t} \boldsymbol{q}_{h,g',t}^\top. \end{aligned} \tag{56}$$

The residual bilinear form is scalar, so each summand is a scaled query outer product.

### C.5 Value Hessian

Perturb the shared value $\boldsymbol{v}_{h,s}$ by $\boldsymbol{\delta} \in \mathbb{R}^{d_\mathrm{v}}$,

$$\boldsymbol{v}_{h,s} \mapsto \boldsymbol{v}_{h,s} + \boldsymbol{\delta}. \tag{57}$$

Values do not enter the softmax, so the attention probabilities remain fixed. For query head $(h, g)$ at position $t \geq s$, the head-output derivative is

$$\left.\frac{\partial \boldsymbol{o}'_{h,g,t}}{\partial \boldsymbol{\delta}^\top}\right|_{\boldsymbol{\delta}=\mathbf{0}} = p_{h,g,t,s}\mathbf{I}_{d_\mathrm{v}}. \tag{58}$$

The attention-output derivative is, therefore,

$$\left.\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right|_{\boldsymbol{\delta}=\mathbf{0}} = \sum_g p_{h,g,t,s}\boldsymbol{W}_{\mathrm{O}(h,g)}^\top. \tag{59}$$

Substituting into Eq. (49) gives the $d_\mathrm{v} \times d_\mathrm{v}$ value Hessian

$$\begin{aligned}\boldsymbol{H}_{\mathrm{V}(h,s)} &= 2\sum_{t=1}^{S}\left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)^\top\left(\frac{\partial \boldsymbol{y}'_t}{\partial \boldsymbol{\delta}^\top}\right)\Bigg|_{\boldsymbol{\delta}=\mathbf{0}} \\ &= 2\sum_{t=1}^{S}\sum_{g,g'} p_{h,g,t,s}p_{h,g',t,s}\boldsymbol{\Gamma}_{h,g,g'}.\end{aligned} \tag{60}$$

### C.6 Aggregation for WUSH-A

For the calibration sequence $i$, let $\boldsymbol{H}^{(i)}_{\mathrm{K}(h,s)}$ and $\boldsymbol{H}^{(i)}_{\mathrm{V}(h,s)}$ denote the per-position Hessians above. Using the Gram matrices from Algorithm 1, the resulting transforms are

$$\begin{aligned}\boldsymbol{T}_{\mathrm{K}(h)} &= \textsc{Wush}\left(\sum_{i=1}^{n_\mathrm{cal}}\boldsymbol{M}^{(i)}_{\mathrm{K}(h)}, \sum_{i=1}^{n_\mathrm{cal}}\sum_{s=1}^{S}\boldsymbol{H}^{(i)}_{\mathrm{K}(h,s)}\right), \\ \boldsymbol{T}_{\mathrm{V}(h)} &= \textsc{Wush}\left(\sum_{i=1}^{n_\mathrm{cal}}\boldsymbol{M}^{(i)}_{\mathrm{V}(h)}, \sum_{i=1}^{n_\mathrm{cal}}\sum_{s=1}^{S}\boldsymbol{H}^{(i)}_{\mathrm{V}(h,s)}\right).\end{aligned} \tag{61}$$

Thus, WUSH-A differs from WUSH only in the Hessian supplied to the transform construction.

## D Additional Experimental Results

### D.1 Additional Attention-Stage Errors

The 2-bit diagnostic plots from Section 5.1 appear in Figure 2, and the 3-bit and 4-bit module-output errors appear in Figure 3. For the diagnostics, query-key dot products are measured before scaling and masking, while softmax uses the model's native scaling and causal mask. The single-sided module-output plots quantize only keys or only values.

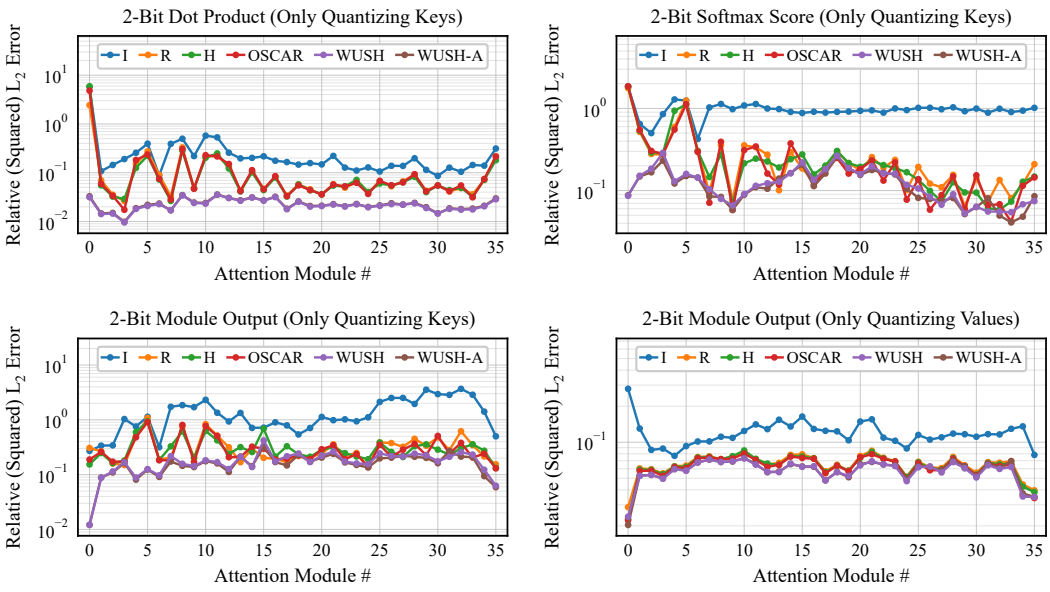


Figure 2: Relative (squared) $L_2$ errors at 2-bit for I, R, H, OSCAR, WUSH, and WUSH-A. The top row shows query-key dot-product and softmax errors with only keys quantized. The bottom row shows module-output errors with only keys or only values quantized.

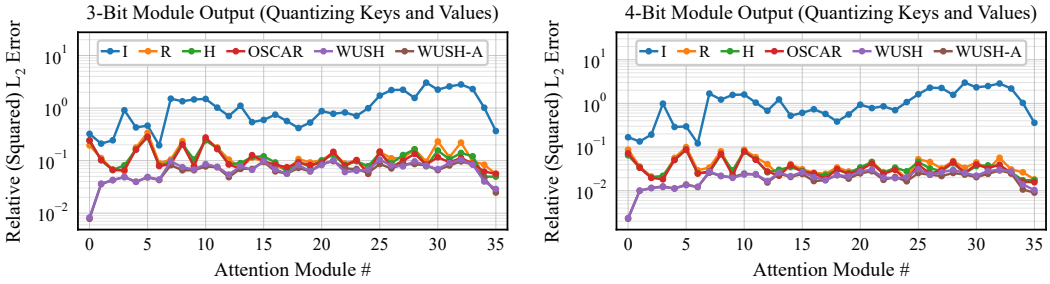


Figure 3: Relative (squared) $L_2$ module-output errors at 3-bit and 4-bit for I, R, H, OSCAR, WUSH, and WUSH-A.

### D.2 Quantizer and Calibration Choices

Table 3 compares the main Qwen3-8B configurations with two alternatives under the same 2-bit benchmark protocol. The main rows repeat the scores in Table 2, and the alternative rows come from separate control evaluations.

**Quantizer choice.** With the WUSH-KV transform and FineWeb-Edu calibration setup held fixed, the OSCAR-style quantizer yields higher end-to-end scores than QuEST on all four tasks. In a separate single-run AIME 2025 diagnostic, OSCAR calibrated on GPQA Diamond with QuEST quantizer scored 0.0%. In our local layerwise and attention-module-wise measurements, QuEST instead has lower $L_2$ quantization error for both transforms. Thus local $L_2$ reconstruction error need not predict downstream autoregressive quality, motivating the OSCAR-style quantizer in the main evaluation.

**Calibration choice.** WUSH-KV uses 128 FineWeb-Edu sequences of length 32,768, or about 4.2 million calibration tokens. By comparison, OSCAR's Qwen3-8B calibration targets about 8k GPQA Diamond tokens. Two preliminary AIME 2025 runs with GPQA-calibrated WUSH-KV and OSCAR-style quantizer scored 16.7% and 10.0%, suggesting that this setup was suboptimal for WUSH-KV in our tests. We also recalibrated OSCAR on a FineWeb-Edu set matched in size to WUSH-KV's calibration set to test whether more calibration data would improve its results. This alternative scores

Table 3: Qwen3-8B downstream scores (%) over three seeds for the main configurations and quantizer or calibration alternatives. The gray second line gives the mean generated-token count in thousands (k) and the budget-exhaustion rate. Main configurations are marked in bold.

| Calibration, Quantizer | AIME 2025 | MATH-500 | GPQA Diamond | LiveCodeBench v6 |
|---|---|---|---|---|
| *WUSH-KV* | | | | |
| **FineWeb-Edu, OSCAR** | **46.7 ± 3.3**<br>53.4k 12% | **92.9 ± 0.1**<br>12.2k 2% | **54.5 ± 2.3**<br>16.3k 1% | **35.2 ± 0.7**<br>53.9k 26% |
| FineWeb-Edu, QuEST | 28.9 ± 6.9<br>63.4k 42% | 88.7 ± 1.0<br>11.7k 4% | 50.5 ± 2.5<br>22.1k 4% | 33.1 ± 0.6<br>63.7k 46% |
| *OSCAR* | | | | |
| **GPQA Diamond, OSCAR** | **34.4 ± 8.4**<br>63.0k 42% | **89.3 ± 1.1**<br>16.0k 6% | **52.9 ± 3.0**<br>18.3k 1% | **20.4 ± 0.3**<br>73.8k 71% |
| FineWeb-Edu, OSCAR | 22.2 ± 1.9<br>70.8k 60% | 76.9 ± 1.3<br>21.2k 18% | 48.7 ± 0.3<br>19.2k 4% | 13.1 ± 1.2<br>79.1k 80% |

lower than the released OSCAR transforms calibrated on the GPQA Diamond on all four tasks, so we retain the released transforms in the main comparison.

### D.3 Long-Context Evaluations

We evaluate Qwen3-8B and Qwen3-32B on long-context tasks with 2-bit KV caches and BF16 baselines using the SGLang setup of Section 5.3. For Qwen3-8B, Table 4 reports RULER NIAH results from 4k to 128k, and Tables 5 and 6 reports OpenAI MRCR results across 0k-128k. We report the normalized area under the MRCR score vs context-length curve (AUC), following Context Arena. The separate Qwen3-32B MRCR results appear in Table 7. Scores in this subsection are averaged over multiple seeds, and ± denotes standard deviation when shown.

In the RULER NIAH results, WUSH-KV remains close to BF16 at 4k and 8k but degrades as the context length increases. WUSH-KV scores above OSCAR at every tested length, with the largest gap at 128k.

MRCR is difficult for all quantized methods, with substantial losses relative to BF16. On Qwen3-8B, WUSH-KV generally degrades more gradually across context lengths and retains higher scores at long contexts than the OSCAR baseline. On Qwen3-32B, OSCAR has slightly higher aggregate scores, while WUSH-KV scores higher at 64k-128k for all three needle counts.

Table 4: Qwen3-8B RULER NIAH accuracy (%) over eight subtasks by context length.

| Method | 4k | 8k | 16k | 32k | 64k | 128k | Overall |
|---|---|---|---|---|---|---|---|
| BF16 | 99.71 | 97.79 | 92.85 | 94.29 | 85.09 | 79.17 | 91.48 |
| WUSH-KV | **97.44** | **91.92** | **80.75** | **76.87** | **68.36** | **57.26** | **78.77** |
| OSCAR | **96.63** | **89.79** | **80.00** | 74.00 | 64.36 | 25.69 | 71.74 |

Table 5: Qwen3-8B MRCR summary accuracy (%).

| Method | 2-Needle | 4-Needle | 8-Needle | Overall | AUC |
|---|---|---|---|---|---|
| BF16 | 36.89 ± 1.96 | 22.66 ± 1.82 | 16.38 ± 0.44 | 25.31 | 21.95 |
| WUSH-KV | **10.55 ± 0.24** | **9.66 ± 0.37** | **8.45 ± 0.24** | **9.55** | **8.35** |
| OSCAR | 7.27 ± 0.61 | 6.98 ± 0.30 | 6.11 ± 0.05 | 6.79 | 4.72 |

Table 6: Qwen3-8B MRCR accuracy (%) by needle count and context-length bucket.

| Method | 0k-8k | 8k-16k | 16k-32k | 32k-64k | 64k-128k |
|---|---|---|---|---|---|
| *2-Needle* | | | | | |
| BF16 | $54.72 \pm 2.40$ | $41.14 \pm 1.01$ | $37.69 \pm 3.47$ | $27.48 \pm 2.17$ | $23.16 \pm 3.31$ |
| WUSH-KV | $\mathbf{18.05 \pm 0.26}$ | $\mathbf{10.82 \pm 0.92}$ | $\mathbf{10.14 \pm 0.18}$ | $\mathbf{8.14 \pm 0.79}$ | $\mathbf{5.49 \pm 0.14}$ |
| OSCAR | $15.36 \pm 1.30$ | $8.88 \pm 0.56$ | $7.05 \pm 0.40$ | $4.69 \pm 0.69$ | $0.29 \pm 0.11$ |
| *4-Needle* | | | | | |
| BF16 | $29.64 \pm 1.71$ | $22.73 \pm 1.16$ | $23.59 \pm 1.26$ | $19.48 \pm 3.17$ | $17.14 \pm 2.10$ |
| WUSH-KV | $\mathbf{12.83 \pm 0.36}$ | $9.15 \pm 0.08$ | $\mathbf{10.93 \pm 1.21}$ | $\mathbf{7.26 \pm 0.25}$ | $\mathbf{7.87 \pm 0.41}$ |
| OSCAR | $11.63 \pm 1.18$ | $\mathbf{9.44 \pm 0.65}$ | $8.66 \pm 0.25$ | $3.61 \pm 0.24$ | $1.06 \pm 0.12$ |
| *8-Needle* | | | | | |
| BF16 | $18.22 \pm 1.39$ | $20.67 \pm 1.39$ | $17.04 \pm 1.38$ | $13.78 \pm 1.31$ | $12.25 \pm 0.56$ |
| WUSH-KV | $8.99 \pm 0.19$ | $\mathbf{9.73 \pm 0.74}$ | $\mathbf{8.36 \pm 0.46}$ | $\mathbf{8.19 \pm 0.24}$ | $\mathbf{6.98 \pm 0.15}$ |
| OSCAR | $\mathbf{9.34 \pm 0.37}$ | $8.63 \pm 0.38$ | $7.11 \pm 0.78$ | $4.26 \pm 0.79$ | $1.09 \pm 0.27$ |

Table 7: Qwen3-32B MRCR accuracy (%) by needle count, including context-length buckets, overall scores, and AUC.

| Method | 0k-8k | 8k-16k | 16k-32k | 32k-64k | 64k-128k | Overall | AUC |
|---|---|---|---|---|---|---|---|
| *2-Needle* | | | | | | | |
| BF16 | 70.98 | 60.14 | 50.88 | 38.27 | 36.87 | $51.49 \pm 2.03$ | 43.70 |
| WUSH-KV | 16.39 | 10.79 | **8.93** | **10.48** | **6.18** | $10.57 \pm 0.69$ | **9.25** |
| OSCAR | **24.83** | **14.39** | **9.01** | **10.87** | 3.92 | $\mathbf{12.64 \pm 0.79}$ | **9.46** |
| *4-Needle* | | | | | | | |
| BF16 | 47.80 | 27.65 | 23.84 | 28.62 | 24.22 | $30.75 \pm 1.76$ | 27.03 |
| WUSH-KV | **12.25** | 4.79 | 8.26 | **8.41** | **8.44** | $8.50 \pm 0.54$ | **8.15** |
| OSCAR | **14.84** | **12.59** | **9.41** | 7.48 | 5.51 | $\mathbf{10.06 \pm 0.60}$ | **8.10** |
| *8-Needle* | | | | | | | |
| BF16 | 29.41 | 21.58 | 17.50 | 25.01 | 15.32 | $21.86 \pm 1.45$ | 20.72 |
| WUSH-KV | 7.79 | 7.39 | 7.25 | **8.43** | **6.95** | $7.57 \pm 0.45$ | **7.68** |
| OSCAR | **9.87** | **9.67** | **8.98** | **8.29** | 4.62 | $\mathbf{8.29 \pm 0.41}$ | **7.64** |