ACTIVATION DENOISING: A ROBUSTNESS VIEW ON PARALLEL VS SEQUENTIAL LLM QUANTIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Post-training quantization is a powerful tool for compressing large language models. The most scalable methods quantize every layer in parallel, but quantization errors then compound through the residual stream, as no layer corrects for the errors of the layers before it. Sequential quantization accounts for this error compounding by re-calibrating each layer on the already-quantized outputs of its predecessors, yielding stronger results but at the cost of a serial schedule that becomes a bottleneck at scale. As a solution, we propose *parallel quantization with activation denoising*, which recovers much of the sequential benefit while keeping quantization fully parallel. Rather than re-calibrating layer-by-layer, we take a robustness perspective and model the upstream error as noise, regularizing to be robust to it through a preprocessing step followed by metric-weighted rounding. Applied at every layer, this regularization forms a depth-compounding smoothness penalty that dampens how strongly quantization errors amplify through the model. Unlike orthogonal rotations commonly used in quantization, which must preserve the model's function, we multiply the weights by a more general linear transformation. We find that the two are complementary and their effects compound. Empirically, our robustness regularization recovers a significant part of sequential quantization's benefit in a single parallel pass, at a fraction of its cost. Overall, by treating compounding quantization errors as a robustness problem, we offer a principled foundation for more efficient and accurate LLM quantization at scale.

1 INTRODUCTION

Post-training quantization (PTQ) is a powerful tool for deploying large language models (LLMs) under memory and bandwidth budgets, mapping high-precision weights to a few bits without retraining (Dettmers et al., 2022; Dettmers & Zettlemoyer, 2023; Frantar et al., 2023). Quantization usually involves two steps: (1) a forward pass that collects each layer's input activations, and (2) rounding the weights onto the quantization grid, a challenging optimization problem that dominates quantization time. The most scalable PTQ methods therefore round all layers independently and in parallel.

The key problem of existing parallel methods is that they sacrifice accuracy for efficiency: each layer is calibrated on clean inputs and never corrects for quantization errors it receives at inference time from previous layers, so errors compound through the model (Arai & Ichikawa, 2025). Sequential quantization corrects this by calibrating each layer on outputs of the layers quantized before it (Frantar et al., 2023; Li et al., 2025). This becomes a bottleneck at scale since layers must then be quantized one after another. Recovery fine-tuning (Tseng et al., 2024a; Malinovskii et al., 2024) can partially recover lost accuracy, but only as a separate costly training stage.

As a solution, we take a robustness perspective on the gap between parallel and sequential quantization and propose *parallel quantization with activation denoising* (Figure 1): instead of correcting for the upstream quantization error one layer at a time, we make each layer robust to upstream perturbations. In a single additional forward pass through the full-precision model, we inject random noise into the activations and propagate it through the model. Then, for each weight matrix in parallel, we construct a linear denoising filter that estimates cleaner from noisy inputs and is absorbed into the weights. After this step, we round the modified weights in a noise-aware metric, independently and in parallel. The denoising filter acts as a depth-compounding smoothness penalty that limits how strongly errors from quantized layers amplify through the LLM at inference.

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.

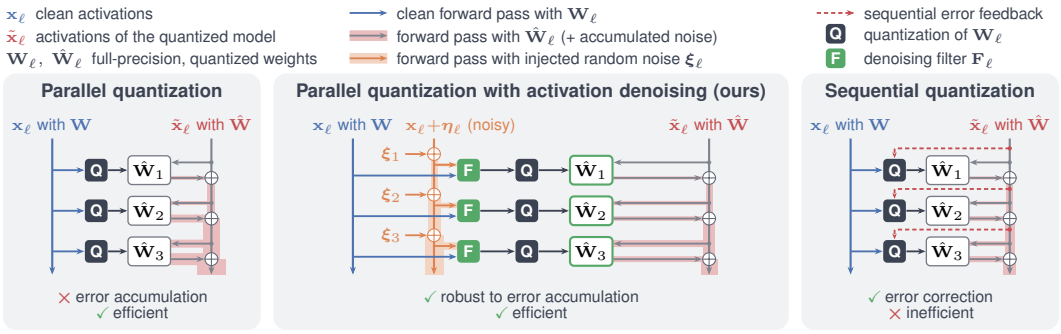


Figure 1: **A robustness perspective on the parallel-sequential quantization gap.** Quantization errors enter the residual stream at every layer and accumulate with depth. Parallel quantization (left) is efficient but ignores this error; sequential quantization (right) corrects it, but only one layer after another. Instead of measuring and correcting this error layer by layer, we regularize every layer against noise (middle): we inject random noise in a single extra forward pass and use its moments to construct a denoising filter $\mathbf{F}_\ell$ that is absorbed into each layer's weights before rounding. Every layer thereby becomes more robust to noise, at no additional inference cost. Overall, our findings suggest that accumulated quantization error can be treated as a robustness problem: regularizing all layers against noise in parallel offers an efficient alternative to correcting the error sequentially.

Activation denoising adds weighted column-wise L_2 regularization, complementing rotations. Rotations reduce weight outliers by redistributing magnitude across channels (Chee et al., 2023; Ashkboos et al., 2024b), but preserve the Frobenius norm. To preserve the model's function, each rotation must be paired with its inverse, either fused into adjacent weights or applied online at inference. Our denoising filter instead absorbs its regularization into a general linear weight transform. Empirically, it lowers the Frobenius norm and improves over both random and learned rotations.
Our main contributions are:

- **Activation denoising framework.** We propose a quantization objective that makes each layer robust to perturbed inputs and generalizes existing parallel and sequential objectives. We show that it reduces exactly to metric-weighted rounding with a transformed target and metric, allowing the use of existing rounding algorithms and adding no inference cost.
- **Noise injection as a robustness regularizer.** To enable parallel and robust quantization, we estimate the required statistics from a single additional forward pass with injected Gaussian noise at the input of each layer, and prove that the local injection induces a weighted column-wise L_2 penalty toward down-scaled targets.
- **Empirical validation.** We demonstrate that parallel activation denoising consistently improves 3- and 4-bit scalar and 2-bit vector quantization across models from 1B to 13B parameters, recovering a significant part of the benefit of sequential quantization while all layers are still quantized in parallel. Finally, we show that the remaining gap to sequential quantization can be closed through cheap trade-offs rather than a full sequential pass.

Overall, addressing quantization errors from a robustness perspective pushes the efficiency–accuracy frontier of LLM quantization, narrowing the gap to sequential quantization at a fraction of its time.

2 Related Work

Post-training quantization for LLMs. PTQ compresses a pretrained LLM using a small calibration set without retraining (Yao et al., 2022; Dettmers et al., 2022; Dettmers & Zettlemoyer, 2023). Below 4 bits, the dominant approach minimizes a layer-wise reconstruction error with second-order information (Nagel et al., 2020; Frantar & Alistarh, 2022; Frantar et al., 2023), recently refined with Kronecker-factored curvature (Tseng et al., 2025; Lee et al., 2026). Complementary directions protect salient or outlier weights (Lin et al., 2024b; Dettmers et al., 2024; Kim et al., 2024), allocate mixed precision (Huang et al., 2025; Zhao et al., 2024), learn quantization parameters block-wise (Shao et al., 2024; Liu et al., 2024a), or replace the uniform grid with floating-point formats (Liu

et al., 2023) or vector or trellis codes (Egiazarian et al., 2024; van Baalen et al., 2024; Tseng et al., 2024a;b; van der Ouderaa et al., 2024). Our proposed framework is general and supports both scalar and vector quantization. Moreover, we focus on finetuning-free PTQ, avoiding the extra training stage of quantization-aware training and recovery fine-tuning (Liu et al., 2024b; Malinovskii et al., 2024) or the inference cost of full-precision low-rank corrections (Scetbon & Hensman, 2024).

Outliers and incoherence processing. Outliers in weights and activations limit low-bit precision (Wei et al., 2022b; Bondarenko et al., 2023; Zhang et al., 2024) and can be migrated between activations and weights by equivalent channel-wise scaling (Xiao et al., 2023; Wei et al., 2023). Incoherence processing instead rotates weights and activations using random orthogonal or Hadamard matrices (Chee et al., 2023; Tseng et al., 2024a), which spread outliers without changing the model output, but modify the forward pass. Exploiting the computational invariance of transformers (Ashkboos et al., 2024a), some of these rotations can be fully fused into weights, leaving the forward pass unchanged but tying the rotations across layers (Ashkboos et al., 2024b). Recent work optimizes these transformations rather than sampling them (Lin et al., 2024a; Liu et al., 2025; Akhondzadeh et al., 2025; Shao et al., 2025; Gadhikar et al., 2025; Sun et al., 2025). These transformations make weights and activations easier to quantize while preserving the full-precision model's output. Our denoising instead changes each layer's mapping to compensate for upstream errors and reduces the weights' Frobenius norm, complementing the outlier redistribution of orthogonal rotations.

Cross-layer error compensation. Sequential methods compensate for the error accumulated in already-quantized layers (Nagel et al., 2020; Li et al., 2021; Ding et al., 2025): GPTAQ (Li et al., 2025) matches each quantized layer to the full-precision output within GPTQ's column-wise updates, QEP (Arai & Ichikawa, 2025) achieves the same through a closed-form weight correction applied before rounding, Qronos (Zhang et al., 2025) corrects it within the rounding updates, and CoreQ (Cha et al., 2026) derives the correction strength in closed form. The corrections above require measuring the accumulated error in a fully sequential pass. We instead replace the measurement with a noise model estimated in parallel and account for it in the rounding target and metric; our objective recovers those of GPTQ, GPTAQ, and QEP as special cases (section 4). Our random noise model relates to randomly dropping activation quantization during reconstruction (Wei et al., 2022a), but acts as a quantization-time regularizer that requires no gradient-based optimization.

3 Background

Post-training quantization. Post-training quantization (PTQ) compresses a trained model by mapping its weights from high precision (e.g. 16 bits) to a few bits, without retraining. Let $\mathcal{G}_b$ denote the set of weight matrices representable with b bits per weight: uniform scalar quantization rounds each weight to one of 2^b levels of a uniform grid, with scale and zero-point shared within a row or group, whereas vector quantization rounds groups of g weights jointly to one of 2^{bg} entries. Since rounding weights independently ignores how their errors interact, a standard PTQ objective instead minimizes the layer's output error on a small calibration set: for weights $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d}$ and layer inputs $\mathbf{x} \in \mathbb{R}^d$ from a calibration set $\mathcal{D}$, $\min_{\hat{\mathbf{W}} \in \mathcal{G}_b} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} \|(\hat{\mathbf{W}} - \mathbf{W})\mathbf{x}\|^2$. To second order, and treating each layer independently, this approximates the KL divergence between the full-precision and the quantized model, decoupling row by row within each weight matrix $\mathbf{W}$, with curvature $\mathbf{H} = \mathbb{E}[\mathbf{x}\mathbf{x}^\top]$.

Metric-weighted rounding. More generally, this is a metric-weighted rounding problem: given a continuous target $\mathbf{W}^\star$, a positive-definite metric $\mathbf{M}$ (in the case above $\mathbf{M} = \mathbf{H}$), and a grid $\mathcal{G}_b$,

$$\min_{\hat{\mathbf{W}} \in \mathcal{G}_b} \operatorname{tr}\big[(\hat{\mathbf{W}} - \mathbf{W}^\star)\,\mathbf{M}\,(\hat{\mathbf{W}} - \mathbf{W}^\star)^\top\big], \tag{1}$$

where $\mathbf{M}$ decides which weight perturbations are cheap. Equation 1 is an expensive combinatorial optimization problem. Existing rounding algorithms solve it approximately: GPTQ (Frantar et al., 2023), for example, greedily quantizes one column at a time and feeds each column's rounding error onto the remaining columns through $\mathbf{M}^{-1}$. We write $\operatorname{Round}^b_{\mathbf{M}}(\mathbf{W}^\star)$ for the approximate solution returned by any such rounding algorithm in the metric $\mathbf{M}$.

Sequential vs. parallel quantization. Layer-wise PTQ methods differ in whether a layer is calibrated on clean inputs $\mathbf{x}$ or on perturbed inputs $\tilde{\mathbf{x}}$ it receives once the earlier layers are quantized. Parallel quantization uses $\mathbf{x}$, so all layers quantize independently but ignore the upstream error. Existing sequential quantization compensates for the upstream error either symmetrically, $\mathbb{E}\|(\hat{\mathbf{W}} - \mathbf{W})\tilde{\mathbf{x}}\|^2$ using only $\tilde{\mathbf{x}}$, or asymmetrically, $\mathbb{E}\|\hat{\mathbf{W}}\tilde{\mathbf{x}} - \mathbf{W}\mathbf{x}\|^2$ using both $\tilde{\mathbf{x}}$ and $\mathbf{x}$ (see e.g. (Li et al., 2025)).

4 ACTIVATION DENOISING QUANTIZATION: A FILTERING PERSPECTIVE

Quantization errors compound across layers: each layer's rounding error propagates downstream. Existing parallel methods ignore the upstream error during quantization. Sequential quantization corrects it, but has to round the layers one after another, which prevents parallelizing the rounding step. This leaves a gap between efficient parallel and accurate sequential quantization. As a solution, our main idea is to denoise each layer's input activations in parallel. To this end, we propose a denoising quantization objective and show that it reduces to a linear weight transform followed by rounding in a new metric, at zero additional inference cost.

4.1 ACTIVATION DENOISING QUANTIZATION OBJECTIVE

We model the layer's input as $\mathbf{x}+\lambda\boldsymbol{\eta}$ and ask its quantized weights to match the full-precision output on $\mathbf{x}+\rho\boldsymbol{\eta}$. Keeping less noise in the target than in the input ($\rho<\lambda$) encourages denoising:

$$\min_{\hat{\mathbf{W}}\in\mathcal{G}_b} \ \mathbb{E}_{\mathbf{x},\boldsymbol{\eta}}\big\|\hat{\mathbf{W}}\big(\mathbf{x}+\lambda\,\boldsymbol{\eta}\big)-\mathbf{W}\big(\mathbf{x}+\rho\,\boldsymbol{\eta}\big)\big\|^2 \ + \ \alpha\,\|\hat{\mathbf{W}}-\mathbf{W}\|_F^2, \tag{2}$$

where $\mathbf{W}$ is the full-precision weight matrix, $\boldsymbol{x}$ is its input activations from the full-precision model, and $\boldsymbol{\eta}$ a perturbation of the clean activations $\boldsymbol{x}$. The expectation is over their joint distribution, with $\mathbf{x}\sim\mathcal{D}$. Here, the first term is a reconstruction objective on the layer's output with two parameters: the input noise scale $\lambda\in[0,1]$ (how much error the quantized layer's input carries), and target noise scale $\rho\in[0,1]$ (how much of that error the reconstruction target keeps). The second term is a proximity penalty on the weights, where $\alpha\geq 0$ is a proximity ridge.

Partial vs. full denoising. Note that $\rho{=}\lambda$ yields a symmetric correction, where the layer adapts to the noise without removing it, while $\rho{=}0$ is fully asymmetric and requires the layer to remove the noise entirely. Values in between ($0<\rho<\lambda$) correspond to partial denoising. We treat λ and ρ as hyperparameters and find that partial denoising typically performs best in practice, even when $\boldsymbol{\eta}$ is the exact sequential error (see Appendix A.2).

The activation denoising objective establishes a general and flexible quantization framework. Notably, the objective we aim to minimize in Equation 2 generalizes existing parallel and sequential quantization objectives into a flexible framework. Specifically, we can recover the parallel GPTQ objective (Frantar et al., 2023) for $\lambda{=}0$ and $\rho{=}0$. When using the sequential error $\boldsymbol{\eta}_{seq}=\tilde{\mathbf{x}}-\mathbf{x}$ (where $\tilde{\mathbf{x}}$ is the input when already quantized up to this layer), we can further recover sequential GPTQ ($\lambda{=}1$, $\rho{=}1$), as well as sequential GPTAQ (Li et al., 2025) and QEP (Arai & Ichikawa, 2025) for $\lambda{=}1$ with $1{-}\rho$ being their correction strength hyperparameter. We show this in detail in Appendix C. In contrast to these baselines, our framework is more general and supports general noise models beyond the sequential $\boldsymbol{\eta}_{seq}$ and flexible noise scales λ and ρ. Together, this allows us to find better trade-offs at the efficiency-accuracy frontier of LLM quantization.

4.2 DENOISING QUANTIZATION IN PRACTICE: PREPROCESS THEN ROUND

In the following, we show how to solve the optimization problem in Equation 2 by transforming the weights and rounding them in a new metric using existing rounding algorithms.

Theorem 1 (Pretransform-round equivalence)**.** *For $\alpha>0$, the optimization problem in Equation 2 is equivalent to the one in Equation 1 with a specific metric and target: it is equivalent to*

$$\min_{\hat{\mathbf{W}}\in\mathcal{G}_b} \ \mathrm{tr}\big[(\hat{\mathbf{W}}-\tilde{\mathbf{W}})\,\mathbf{G}\,(\hat{\mathbf{W}}-\tilde{\mathbf{W}})^\top\big], \quad \tilde{\mathbf{W}}\triangleq\mathbf{W}\mathbf{F}$$

where

$$\mathbf{G} \ \triangleq \ \mathbf{H}+\alpha\mathbf{I}+\lambda^2\mathbf{N}+\lambda(\mathbf{C}+\mathbf{C}^\top), \quad \mathbf{F} \ \triangleq \ \big(\mathbf{H}+\alpha\mathbf{I}+\lambda\mathbf{C}+\rho\mathbf{C}^\top+\rho\lambda\mathbf{N}\big)\,\mathbf{G}^{-1}. \tag{3}$$

and $\mathbf{H}=\mathbb{E}[\mathbf{x}\mathbf{x}^\top]$ is the input covariance, $\mathbf{C}=\mathbb{E}[\mathbf{x}\boldsymbol{\eta}^\top]$ the signal-noise cross-covariance, and $\mathbf{N}=\mathbb{E}[\boldsymbol{\eta}\boldsymbol{\eta}^\top]$ the noise covariance.

Proof sketch. Expanding Equation 2 and completing the square shows that the objectives differ only by a constant independent of $\hat{\mathbf{W}}$, so they have the same minimizers. Full proof in Appendix D.2. □

As a result, we can use any approximate solver of the rounding problem in Equation 1 and apply it to the pretransformed weights $\tilde{\mathbf{W}}$ in the metric $\mathbf{G}$ to solve our denoising objective in Equation 2.

Interestingly, at $\rho = 0$ (and $\alpha{=}0$ when the inverse exists), $\mathbf{F}$ corresponds to the linear transformation known as the generalized Wiener filter (Wiener, 1949; Kailath et al., 2000), which denoises the input: $\mathbf{F}\tilde{\mathbf{x}}$ estimates the clean input activations $\mathbf{x}$ from the corrupted input $\tilde{\mathbf{x}} = \mathbf{x} + \lambda\boldsymbol{\eta}$.

Parallel denoising quantization in practice. Theorem 1 turns the denoising quantization objective into a practical quantization procedure as follows. Given upstream noise $\boldsymbol{\eta}$ (either the true sequential quantization error $\boldsymbol{\eta}_{\text{seq}}$, or a parallel approximation of it), we estimate the second-order statistics required by Theorem 1 as empirical token averages over a calibration set of n tokens: $\mathbf{H} = \frac{1}{n}\sum_i \mathbf{x}_i\mathbf{x}_i^\top$, $\mathbf{C} = \frac{1}{n}\sum_i \mathbf{x}_i\boldsymbol{\eta}_i^\top$, and $\mathbf{N} = \frac{1}{n}\sum_i \boldsymbol{\eta}_i\boldsymbol{\eta}_i^\top$.

If $\boldsymbol{\eta}$ can be computed in parallel for all layers, each layer's solution depends only on its own moments and not on other quantized layers, and all layers can be quantized independently: we form metric $\mathbf{G}$, compute the pretransform $\tilde{\mathbf{W}} = \mathbf{W}\mathbf{F}$, and round it onto the grid $\mathcal{G}_b$ in the metric $\mathbf{G}$. In practice, we round using the GPTQ rounding proposed in (Frantar et al., 2023) and QuIP# (Tseng et al., 2024a). We summarize this procedure in Algorithm 1.

Algorithm 1 Activation denoising quantization

Input: full-precision weights $\{\mathbf{W}_m\}$, bit-width b, emp. moments $\{\mathbf{H}_m, \mathbf{N}_m, \mathbf{C}_m\}$, noise scales λ, ρ, ridge α

for all linears m **in parallel do**
  $\mathbf{G}_m \leftarrow \mathbf{H}_m + \alpha\mathbf{I} + \lambda^2\mathbf{N}_m + \lambda(\mathbf{C}_m + \mathbf{C}_m^\top)$
  $\mathbf{F}_m \leftarrow (\mathbf{H}_m + \alpha\mathbf{I} + \lambda\mathbf{C}_m + \rho\mathbf{C}_m^\top + \rho\lambda\mathbf{N}_m)\mathbf{G}_m^{-1}$
  $\tilde{\mathbf{W}}_m \leftarrow \mathbf{W}_m\mathbf{F}_m$
  $\hat{\mathbf{W}}_m \leftarrow \mathrm{Round}^b_{\mathbf{G}_m}(\tilde{\mathbf{W}}_m)$
return $\{\hat{\mathbf{W}}_m\}$

Parallel denoising can thus be implemented as an efficient quantization-time, per-layer weight pretransform followed by standard rounding, adding no additional inference cost and requiring no sequential computation given an estimate for the upstream noise $\boldsymbol{\eta}$.

5 Activation noise model for efficient parallel denoising

Sequential methods instantiate the noise $\boldsymbol{\eta}$ in the denoising objective (Equation 2) with the true sequential upstream error, which is only available after quantizing all previous layers in sequence. Instead, our key idea is to inject random noise into the input activations of every layer during an additional calibration forward pass through the full-precision model, and to propagate it downstream. Each layer is thereby perturbed by a total noise $\boldsymbol{\eta}_m = \boldsymbol{\eta}_m^{\uparrow} + \boldsymbol{\xi}_m$: the upstream error $\boldsymbol{\eta}_m^{\uparrow}$ propagated from earlier layers, plus a new local injection $\boldsymbol{\xi}_m$.

Local random noise injection. We inject independent, zero-mean, per-channel Gaussian noise into the input activations at each linear weight $\mathbf{W}_m$, sampled i.i.d. for every calibration token t:

$$\boldsymbol{\xi}_{m,t} \sim \mathcal{N}\big(\mathbf{0}, \boldsymbol{\Sigma}_m\big), \qquad \boldsymbol{\Sigma}_m = \mathrm{diag}\left(\sigma_{m,1}^2, \ldots, \sigma_{m,d}^2\right), \qquad \sigma_{m,c} \triangleq \gamma \max_t \left|x_{m,t,c} + \eta^{\uparrow}_{m,t,c}\right|,$$

where $x_{m,t,c}$ denotes channel c of input $\mathbf{x}_{m,t} \in \mathbb{R}^d$. Here, the noise injected into channel c has its own standard deviation $\sigma_{m,c}$ corresponding to the largest absolute value the channel attains across calibration tokens, scaled by a model-wide constant $\gamma \geq 0$. High-magnitude outlier channels thus receive proportionally more noise, so the layer is regularized to be robust where its inputs are largest.

Noise propagation. We propagate the injected noise downstream. Starting without upstream noise ($\boldsymbol{\eta}_1^{\uparrow} = \mathbf{0}$), the total noise $\boldsymbol{\eta}_m = \boldsymbol{\eta}_m^{\uparrow} + \boldsymbol{\xi}_m$ at module m propagates through the model to form the next module's upstream noise $\boldsymbol{\eta}_{m+1}^{\uparrow}$. Early injections thus accumulate through depth, whereas each local noise injection $\boldsymbol{\xi}_m$ is a new, independent sample.

Efficient parallel solution. Crucially, no separate optimization is needed: one clean and one noise-injected forward pass (both at full-precision) yield the per-weight moments $\mathbf{H}_m, \mathbf{C}_m, \mathbf{N}_m$, which the pretransform absorbs before rounding each weight independently at no additional optimization or inference cost. We summarize this procedure in Algorithm 2. Since the moments come from full-precision forward passes, we can quantize all layers independently in parallel afterwards using Algorithm 1.

Algorithm 2 Estimating denoising moments

Input: n calibration tokens; linear weights $\{\mathbf{W}_m\}$

$\{\mathbf{x}_{m,t}\} \leftarrow$ per-linear inputs from full-precision forward
$\{\tilde{\mathbf{x}}_{m,t}\} \leftarrow$ per-linear inputs from full-precision forward that locally injects and propagates noise $\boldsymbol{\xi}_{m,t}{\sim}\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_m)$
for all linears m **in parallel do**
  $\boldsymbol{\eta}_{m,t} \leftarrow \tilde{\mathbf{x}}_{m,t} - \mathbf{x}_{m,t}$ (calib. token $t{=}1{:}n$)
  $\mathbf{H}_m \leftarrow \frac{1}{n}\sum_t \mathbf{x}_{m,t}\mathbf{x}_{m,t}^\top$
  $\mathbf{C}_m \leftarrow \frac{1}{n}\sum_t \mathbf{x}_{m,t}\boldsymbol{\eta}_{m,t}^\top$
  $\mathbf{N}_m \leftarrow \frac{1}{n}\sum_t \boldsymbol{\eta}_{m,t}\boldsymbol{\eta}_{m,t}^\top$
return moments $\{\mathbf{H}_m, \mathbf{N}_m, \mathbf{C}_m\}$

5.1 RANDOM NOISE INJECTION AS ROBUSTNESS REGULARIZATION

We show that denoising against random noise regularizes weights to be more robust to input perturbations. It induces two effects: *(i) local column-wise regularization* from injected noise $\boldsymbol{\xi}$, and *(ii) depth-compounding regularization* from propagated noise $\boldsymbol{\eta}$, which controls how much error accumulates across layers. The following theorem formalizes effect (i): the local injection combines with the proximity ridge into weighted L_2 penalties toward down-scaled weight columns:

Theorem 2 (Local noise injection acts as a magnitude-reducing ridge)**.** *Let $\boldsymbol{\eta} = \boldsymbol{\eta}^{\uparrow} + \boldsymbol{\xi}$, where $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ is independent of $(\mathbf{x}, \boldsymbol{\eta}^{\uparrow})$ and $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_d^2)$. Up to a constant independent of the weights, the objective in Equation 2 equals its reconstruction term with $\boldsymbol{\eta} = \boldsymbol{\eta}^{\uparrow}$ plus the modified ridge regularization*

$$\sum_c \left(\alpha + \lambda^2 \sigma_c^2\right) \left\| \hat{\mathbf{W}}_{:,c} - s_c \, \mathbf{W}_{:,c} \right\|_2^2, \qquad s_c = \frac{\alpha + \rho\lambda\sigma_c^2}{\alpha + \lambda^2\sigma_c^2},$$

For $\rho < \lambda$ and $\sigma_c > 0$, the penalty favors a down-scaled target $s_c \mathbf{W}_{:,c}$ for column c.

Proof sketch. Splitting the $\boldsymbol{\xi}$-dependent part, the cross term vanishes due to independence, and the quadratic term simplifies to $\sum_c \sigma_c^2 \|\lambda \tilde{\mathbf{W}}_{:,c} - \rho \mathbf{W}_{:,c}\|_2^2$. Since $\boldsymbol{\Sigma}$ is diagonal, adding the ridge decouples over columns, resulting in the stated form. Full proof in Appendix D.3. □

In the denoising regime $\rho < \lambda$, this penalty encourages smaller column-wise L_2 norms by pulling weights toward down-scaled targets. In section 6, we empirically show that this reduces the matrix Frobenius norm, which upper-bounds the module's spectral norm and hence its Lipschitz constant. The smaller norm reduces average amplification of isotropic input perturbations, making layers more robust to them. This complements rotations, which redistribute magnitude but preserve norm.

Taking both effects (i) and (ii) together, we regularize each layer to be robust to noise: quantization stays parallel, and the robustness dampens how strongly quantization errors compound across layers.

6 EXPERIMENTAL EVALUATION

Parallel quantization processes layers independently but ignores upstream quantization errors. Sequential quantization corrects them, but must process layers one after another. Our method keeps quantization fully parallel: it regularizes each layer to be robust to input noise (section 5) via activation denoising (section 4). In this section we demonstrate the effectiveness of our parallel denoising and show that, across model scales, it recovers a significant fraction of the sequential quality. We report additional results and ablations in Appendix A and describe the full setup in Appendix B.

Setup. We quantize Llama-3.2-1B, Llama-3.2-3B, Llama-3-8B, and Llama-2-13B (Touvron et al., 2023; Llama Team, 2024), which are widely used in the quantization literature and span a broad range of scales. We focus on weight quantization, keeping activations and KV cache at 16 bits, and calibrate on 128 2048-token sequences of the WikiText-2 (Merity et al., 2017) training set.

Solvers. Our denoising is agnostic to the rounding solver, and we evaluate it with scalar and vector quantization via GPTQ (Frantar et al., 2023) at 3 bits and QuIP# (Tseng et al., 2024a) (without fine-tuning) at 2 bits. For GPTQ, we rotate weights before quantization, using either the random Hadamard rotations R_1, R_2, R_4 of QuaRot (Ashkboos et al., 2024b) or with R_1, R_2 learned as in SpinQuant (Liu et al., 2025). QuIP# applies its own randomized Hadamard incoherence transform. The QuaRot rotations R_1 and R_2 are fused into the model's weights and thus modify the forward pass during moment estimation, whereas QuIP#'s rotations are applied during quantization, leave the forward pass unchanged during moment estimation, and modify it instead at inference.

Baselines. For each solver, we compare against the two ends of the parallel–sequential spectrum: parallel quantization on clean inputs, and sequential quantization on outputs of the already quantized layers, with either the symmetric correction of GPTQ (Frantar et al., 2023) or the asymmetric correction of GPTAQ (Li et al., 2025). The asymmetric correction is the strongest sequential baseline.

Metrics. We report perplexity averaged over WikiText-2 (Merity et al., 2017), C4 (Raffel et al., 2020), and FineWeb (Penedo et al., 2024) test (i.e. not calibration) sequences. For each method, we report the best configuration under this average; selecting on WikiText-2 alone and evaluating on held-out C4 and FineWeb yields the same conclusions (Appendix A.6). We also report zero-shot accuracy averaged over eight commonsense tasks (Liu et al., 2025), and the fraction of the gap in avg. perplexity from 0% (parallel) to 100% (sequential asymmetric) for each solver separately.

Table 1: **Parallel denoising closes a large part of the parallel–sequential gap for both scalar and vector quantization.** 3-bit GPTQ and 2-bit QuIP# on Llama-3.2-1B/3B, Llama-3-8B, and Llama-2-13B. Acc: zero-shot accuracy; Avg: average perplexity; Gap: fraction of the perplexity gap closed, from 0% (parallel) to 100% (sequential asym. correction). On Llama-2-13B at 2 bits, the two methods nearly coincide, so we only indicate whether a method is below the sequential (>100%) or above the parallel (<0%). Per-dataset and learned-rotation results in Appendix A.

		Quantization objective	Llama 1B			Llama 3B			Llama 8B			Llama 13B		
			Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑
FP16			54.8	11.33	–	62.6	9.17	–	67.1	7.57	–	66.5	5.73	–
3-bit GPTQ	Par.	symmetric correction	50.0	16.97	0%	58.5	12.11	0%	63.5	9.93	0%	64.0	6.40	0%
		Denoising reg. (**ours**)	49.6	16.50	**28%**	58.9	11.91	**29%**	63.4	9.78	**35%**	64.6	6.37	**17%**
	Seq.	symmetric correction	49.8	16.53	26%	59.5	11.99	17%	63.5	9.81	27%	63.7	6.38	10%
		asymmetric correction	50.3	15.29	100%	58.4	11.41	100%	64.0	9.50	100%	64.4	6.21	100%
2-bit QuIP#	Par.	symmetric correction	44.3	27.05	0%	53.4	16.48	0%	58.8	13.58	0%	61.9	7.27	0%
		Denoising reg. (**ours**)	45.1	25.51	**38%**	53.9	15.57	**84%**	59.1	12.67	**128%**	61.4	7.19	**>100%**
	Seq.	symmetric correction	43.8	26.36	17%	53.7	16.29	17%	58.0	13.89	−44%	60.9	7.39	<0%
		asymmetric correction	45.1	23.04	100%	51.3	15.40	100%	56.2	12.87	100%	60.0	7.26	100%

6.1 Activation denoising improves quantization quality in parallel

Table 1 reports our main results. At 3 bits, parallel denoising closes 17–35% of the perplexity gap across all four scales, recovering a significant fraction of the sequential benefit in a single parallel pass. At 2 bits it closes 38% and 84% of the gap on Llama-3.2-1B and 3B, and on Llama-3-8B and Llama-2-13B it even outperforms the sequential method. We see two plausible reasons why our denoising outperforms the sequential method at 2 bits. First, QuIP#'s randomized incoherence transform may already spread errors well enough that sequential correction helps less: at 3B–13B it recovers only 1–15% of the perplexity lost by parallel QuIP#, compared to 18–28% for GPTQ with QuaRot rotations, whereas our denoising recovers a similar or larger fraction under QuIP#. Second, the sequential method is the asymmetric correction of Li et al. (2025) applied to QuIP#. While it is a state-of-the-art baseline, it is not necessarily optimal: it is a special case (λ=1) of our objective with the sequential error (Appendix C), which fixes how much upstream error the layer is assumed to receive. Our framework is more flexible and treats λ as a free parameter, and tuning it improves on the asymmetric correction even with the sequential error (Appendix A.2). Lastly, on zero-shot accuracy, most differences are within about one point, so we view our denoising as comparable to parallel and sequential quantization on accuracy; only at 2 bits does the sequential asymmetric correction fall behind, which our denoising avoids.

Our denoising achieves the quality improvements while remaining fully parallel, which is significantly more efficient at scale (Figure 2). Beyond the clean calibration forward pass that every method requires, our denoising needs only one additional forward pass on noised inputs. This noisy pass is slightly slower than the clean one (about 1.3×), since it additionally samples noise and accumulates noise and cross-covariance statistics. Yet for QuIP#, the cost of both forward passes remains negligible compared to the rounding that solves the underlying weight-optimization problem. On Llama-2-13B the two passes take about 95 seconds combined, under 1% of the quantization time. Given the collected statistics, each layer's pretransform and rounding are independent and can be distributed across GPUs. Using measured single-GPU times for QuIP#, we project a 7.1–7.5× quantization time reduction on eight A100s relative to the single-GPU sequential baseline, assuming balanced work allocation and negligible communication overhead (cf. Appendix A.8).

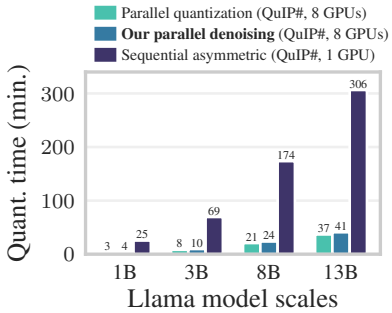


Figure 2: Projected multi-GPU quantization time (A100, QuIP#).

6.2 Parallel denoising composes effectively with orthogonal LLM rotations

Robustness regularization complements rotations by lowering the weight norm. Orthogonal rotations reduce weight outliers by redistributing magnitude across channels, but preserve the Frobenius norm, $\|\mathbf{WR}\|_F = \|\mathbf{W}\|_F$, and the singular values of $\mathbf{W}$. In contrast, by Theorem 2, local noise

injection and the proximity ridge induce weighted column-wise L_2 penalties toward down-scaled targets. The resulting filter $\mathbf{F}$ changes each layer's mapping and can reduce $\|\mathbf{WF}\|_F$. We demonstrate this on the whole-matrix Frobenius norm in Figure 4 (left), which plots the Frobenius-norm ratio $\|\mathbf{WF}\|_F/\|\mathbf{W}\|_F$ across depth and module type for Llama-3.2-1B, comparing denoised and original weights before rounding. The ratio is below 1 for all 112 linears, with the strongest reduction in the attention output projection. This verifies that denoising reduces a quantity that orthogonal rotations cannot change, rather than merely redistributing weight magnitude. The following ablation evaluates the quantization improvements when both are combined.

Parallel denoising significantly improves under model rotations. We perform an ablation study to understand the role of randomization in our method and the baselines. We consider two sources of randomization: the random Hadamard rotation used to make weights easier to quantize, and the injected Gaussian noise we use for regularization. We quantize Llama-3.2-1B at 3-bit precision 100 times for each method, resampling rotations for the baselines and the noise injection for our method (Figure 3). We find that rotations matter greatly for parallel GPTQ: its perplexity varies widely across random Hadamard rotations (large standard deviation, σ=0.16) and a learned SpinQuant rotation improves a lot (16.98→16.41). At the same time, we observe that sequential GPTAQ is quite robust under random rotations (smaller σ=0.05) and barely improved by learning it (15.36 → 15.28). In contrast, our parallel denoising composes effectively with rotations, significantly reducing the gap.

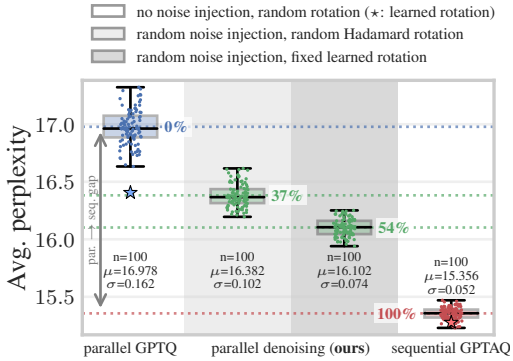


Figure 3: **Random rotation and noise ablation** (W3 Llama-3.2-1B). Avg. perplexity over 100 draws: parallel GPTQ and sequential GPTAQ resample the Hadamard rotation (stars: learned-rotation), while for our denoising we resample noise either for random or learned rotations (gray backgrounds). Notably, our parallel denoising significantly reduces the parallel–sequential gap and is insensitive to the random noise draw.

Together, these results show rotation and denoising are complementary: rotations reduce outliers while preserving the Frobenius norm, whereas denoising lowers this norm and improves quantization under both random and learned rotations.

6.3 Two efficient alternatives to full sequential quantization

Our denoising already recovers much of the gap in a single parallel pass. In this section we explore two approaches to further close the remaining gap without quantizing every layer sequentially.

Trading quantization time: a short sequential prefix suffices. First, we compute sequential GPTAQ for the first K of L blocks and continue quantizing the remaining blocks with parallel-GPTQ (Figure 4, middle). We find that the benefit of sequential quantization is strongly front-loaded across all models: the first quarter of the blocks already closes 40–60% of the gap, whereas closing the last few percent requires sequential quantization to span most of the depth. We see two complementary explanations for this front-loading: First, quantization error compounds through depth, and errors introduced at early blocks propagate through every downstream block, so correcting them early may have a disproportionately large effect. Second, later layers may contribute less to the output, consistent with evidence that language models do not use their depth efficiently and that deeper layers can often be removed with little effect (Csordás et al., 2025; Men et al., 2025), leaving less error to correct. We leave attempts to disentangle these effects to future work.

Trading inference cost: a few 4-bit modules suffice. Second, we explore promoting a small prefix of modules to 4 bits at inference time, while keeping the rest of the network at 3 bits. We find that the value projections v_{proj} dominate: promoting only them to 4 bits in a single parallel pass matches full sequential-GPTAQ quality at an average width of just 3.03 bits, i.e. the whole gap for 0.03 bits over W3. Because v_{proj} carries so few parameters under grouped-query attention, this undercuts every finer group size (g32 needs a full 3.5 bits for the same quality) and any other prefix in the comparison. When a small inference-time cost is acceptable, this is a cheap and effective way to close the remaining gap without any sequential quantization.

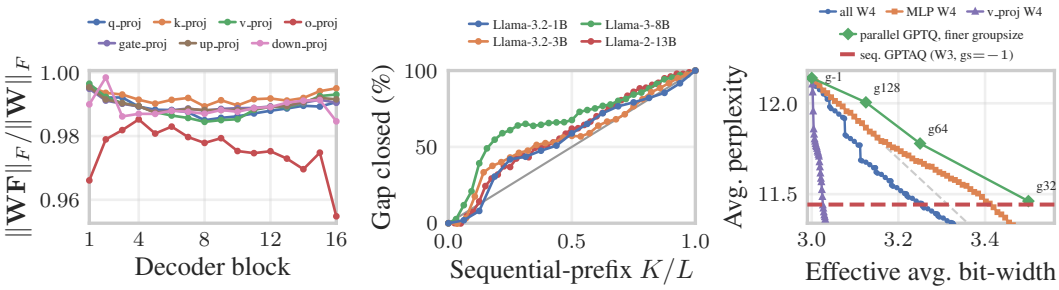


Figure 4: **Denoising reduces weight norms, and two alternatives to full sequential quantization.** *Left* (Frobenius-norm reduction on Llama-3.2-1B): ratio $\|\mathbf{WF}\|_F / \|\mathbf{W}\|_F$ of denoised to original weights before rounding, across depth and module type. The ratio is below 1 for all 112 linears, most strongly in the attention output projection. *Middle* (quantization-time trade-off): quantizing the first K blocks sequentially and the remaining $L-K$ blocks in parallel, we find the sequential benefit is front-loaded: the first quarter of blocks closes 40–60% of the gap, but the last few percent need a prefix spanning most of the depth. *Right* (inference-time trade-off on Llama-3.2-3B): avg. perplexity vs. avg. effective bit-width, promoting a prefix of all/MLP-only/v_{proj}-only modules to 4 bits (green diamonds: finer group-size; red dashed: seq. GPTAQ). We find promoting only v_{proj} to 4 bits reaches sequential quality while raising the avg. bit-width by only $\approx$0.03 bits.

7 Discussion

Future work. We see several promising directions for future work: extending our activation denoising to activation and KV-cache quantization, combining it with stronger rounding schemes (Tseng et al., 2025; Lee et al., 2026), and jointly learning the noise injections and rotations. Recovery fine-tuning (Tseng et al., 2024a; Malinovskii et al., 2024) is a further, orthogonal future direction.

Closing the remaining gap. In a single parallel pass, our denoising already recovers a large fraction of the sequential benefit, though a gap to fully sequential quantization remains. We see the robustness perspective as a promising way to close it further, e.g. by using noise models that carry signal-correlated structure. We demonstrate this in Appendix A.3: the propagated error of an already parallel-quantized model provides a signal-correlated noise model, so an existing quantized model can be improved by denoising against its measured error, closing more of the gap without any sequential quantization. Moreover, because all rounding steps stay independent, the efficiency advantage over sequential quantization should grow at larger scale, which we already demonstrate up to 13B-parameter models. Lastly, where a small trade-off is acceptable, the remaining gap can already be closed with a short sequential prefix or by promoting a few modules to 4 bits (section 6.3).

Robustness regularization. Our results suggest that much of the benefit of sequential quantization does not require the exact sequential upstream error: making each layer robust to propagated perturbations already recovers a significant part of the gap. While our regularization introduces additional hyperparameters that require tuning (noise scales λ, ρ; injection magnitude γ), we believe this can be overcome, since their optimal values are similar across models (Appendix A). More broadly, our regularization is similar to the equivalence between training with input noise and Tikhonov regularization (Bishop, 1995), here it arises at quantization time. This corresponds to average-case robustness against random perturbations; its worst-case, adversarial counterpart (Szegedy et al., 2014; Goodfellow et al., 2015; Madry et al., 2018) is a promising direction for future work.

8 Conclusion

We introduce activation denoising, which recovers much of the benefit of sequential post-training quantization while keeping quantization fully parallel. Our central take-away is that the upstream quantization errors, which sequential methods handle through expensive layer-by-layer recalibration, can instead be addressed by regularizing every layer to be robust to noise, in a single parallel pass. Treating quantization errors as a robustness problem thus offers a principled and more efficient alternative to sequential quantization, which we believe can substantially lower quantization costs, providing a promising direction for efficient and accurate LLM compression at scale.

AI USE STATEMENT

In this work, we used generative AI tools (including Claude Opus 4.8, Opus 5, and Fable 5; and OpenAI's GPT-6 Astra) to aid in developing our main theoretical framework (section 4), to formulate mathematical claims of Theorem 1 and Theorem 2 including finding critical ingredients for proving them. We also used generative AI tools to assist in the writing of proofs, propose and refining hypotheses, in particular to derive improved versions of mathematical claims to enhance clarity. We also used generative AI tools to implement methods, baselines and evaluation code (building up on the SpinQuant codebase (Liu et al., 2025)), starting and monitoring experiments, and support data analysis in general, such as plotting experiment results. We also used generative AI tools to polish the paper to improve readability, fix grammar and spelling, help with formatting figures, tables and improve notation for clarity and consistency across the paper.

We did not use generative AI for generating our main research ideas. Specifically, our idea to take a robustness perspective on the parallel-sequential quantization gap and to propagate noise for robustness regularization is the authors' and not generated by AI. We have further not used generative AI for providing feedback on research methodology, assisting with translation or interpreting results. Generating synthetic datasets or cleaning and reformatting datasets are not applicable to this work (we do not have synthetic data and use existing, public benchmarks for our experiments). We have also not used AI to insert or format references, i.e. all references were manually added and verified.

We have reviewed all AI-assisted work. In particular, we verified LLM-generated mathematical claims, derivations and proofs for correctness. We have also verified that all LLM-generated code is correct and runs as intended, this includes verifying that it reproduces results reported in the literature. Specifically, we ran tests against the original SpinQuant codebase (Liu et al., 2025) and reproduced previously reported results on parallel quantization (Frantar et al., 2023) as well as asymmetric sequential quantization (Li et al., 2025; Arai & Ichikawa, 2025). When using generative AI tools for the paper, we have been careful to ensure that the content is accurate and does not contain any hallucinations or fabricated information. In particular, we ensure that experimental results in figures and tables are reproducible and correspond to actual experimental results without passing them through generative AI tools themselves. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our work is improving the efficiency and accuracy of quantizing LLMs, which can reduce the energy and carbon footprint of deploying LLMs in practice. We do not foresee any negative societal consequences of our work beyond the general risks of LLMs, which are not specific to our work.

REPRODUCIBILITY STATEMENT

We have taken several steps to ensure that our results can be reproduced. Complete proofs of all theoretical claims, together with the assumptions they rely on, are given in Appendix D. All models, datasets and benchmarks used in our experiments are publicly available and referenced in the paper. Appendix B documents the full experimental setup, including all hyperparameters and the settings of every sweep, including hardware details. We fix random seeds throughout, and we reran selected experiments to confirm that they are deterministic and yield identical results across runs. We further quantify the sensitivity of our approach to randomness introduced by rotations and random noise injections in a dedicated ablation in Figure 3. Finally, we provide a minimal implementation of our method as supplementary material sufficient to verify claims and reproduce results, and will release the full codebase required to reproduce all result upon acceptance.

REFERENCES

Mohammad Sadegh Akhondzadeh, Aleksandar Bojchevski, Evangelos Eleftheriou, and Martino Dazzi. Kurtail : Kurtosis-based LLM quantization. In *EMNLP (Findings)*, pp. 17404–17419. Association for Computational Linguistics, 2025.

Yamato Arai and Yuma Ichikawa. Quantization error propagation: Revisiting layer-wise post-training quantization. In *NeurIPS*, 2025.

Saleh Ashkboos, Maximilian L. Croci, Marcelo Gennari Do Nascimento, Torsten Hoefler, and James Hensman. Slicegpt: Compress large language models by deleting rows and columns. In *ICLR*. OpenReview.net, 2024a.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. Quarot: Outlier-free 4-bit inference in rotated llms. In *NeurIPS*, 2024b.

Christopher M. Bishop. Training with noise is equivalent to tikhonov regularization. *Neural Comput.*, 7(1):108–116, 1995.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *AAAI*, pp. 7432–7439. AAAI Press, 2020.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Quantizable transformers: Removing outliers by helping attention heads do nothing. In *NeurIPS*, 2023.

Seohyeon Cha, Huancheng Chen, Dongjun Kim, Haoran Zhang, Kevin Chan, Gustavo de Veciana, and Haris Vikalo. Coreq: Learning-free mismatch correction and successive rounding for quantization. *arXiv preprint arXiv:2602.05902*, 2026.

Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. Quip: 2-bit quantization of large language models with guarantees. In *NeurIPS*, 2023.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL-HLT (1)*, pp. 2924–2936. Association for Computational Linguistics, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the AI2 reasoning challenge. *CoRR*, abs/1803.05457, 2018.

Róbert Csordás, Christopher D. Manning, and Christopher Potts. Do language models use their depth efficiently? In *NeurIPS*, 2025.

Tim Dettmers and Luke Zettlemoyer. The case for 4-bit precision: k-bit inference scaling laws. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pp. 7750–7774. PMLR, 2023.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Gpt3.int8(): 8-bit matrix multiplication for transformers at scale. In *NeurIPS*, 2022.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. Spqr: A sparse-quantized representation for near-lossless LLM weight compression. In *ICLR*. OpenReview.net, 2024.

Xin Ding, Xiaoyu Liu, Zhijun Tu, Yun Zhang, Wei Li, Jie Hu, Hanting Chen, Yehui Tang, Zhiwei Xiong, Baoqun Yin, and Yunhe Wang. CBQ: cross-block quantization for large language models. In *ICLR*. OpenReview.net, 2025.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization. In *ICML*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12284–12303. PMLR / OpenReview.net, 2024.

Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. In *NeurIPS*, 2022.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. OPTQ: accurate quantization for generative pre-trained transformers. In *ICLR*. OpenReview.net, 2023.

Advait Gadhikar, Riccardo Grazzi, and James Hensman. Optrot: Mitigating weight outliers via data-free rotations for post-training quantization. *CoRR*, abs/2512.24124, 2025.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR (Poster)*, 2015.

Wei Huang, Haotong Qin, Yangdong Liu, Yawei Li, Qinshuo Liu, Xianglong Liu, Luca Benini, Michele Magno, Shiming Zhang, and Xiaojuan Qi. Slim-llm: Salience-driven mixed-precision quantization for large language models. In *ICML*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

Thomas Kailath, Ali H Sayed, and Babak Hassibi. *Linear estimation*. Prentice Hall, 2000.

Sehoon Kim, Coleman Hooper, Amir Gholami, Zhen Dong, Xiuyu Li, Sheng Shen, Michael W. Mahoney, and Kurt Keutzer. Squeezellm: Dense-and-sparse quantization. In *ICML*, volume 235 of *Proceedings of Machine Learning Research*, pp. 23901–23923. PMLR / OpenReview.net, 2024.

Donghyun Lee, Yuhang Li, Ruokai Yin, and Priyadarshini Panda. Kronq: LLM quantization via kronecker-factored hessian. *CoRR*, abs/2607.07964, 2026.

Yuhang Li, Ruihao Gong, Xu Tan, Yang Yang, Peng Hu, Qi Zhang, Fengwei Yu, Wei Wang, and Shi Gu. BRECQ: pushing the limit of post-training quantization by block reconstruction. In *ICLR*. OpenReview.net, 2021.

Yuhang Li, Ruokai Yin, Donghyun Lee, Shiting Xiao, and Priyadarshini Panda. GPTAQ: efficient finetuning-free quantization for asymmetric calibration. In *ICML*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

Haokun Lin, Haobo Xu, Yichen Wu, Jingzhi Cui, Yingtao Zhang, Linzhan Mou, Linqi Song, Zhenan Sun, and Ying Wei. Duquant: Distributing outliers via dual transformation makes stronger quantized llms. In *NeurIPS*, 2024a.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Guangxuan Xiao, and Song Han. AWQ: activation-aware weight quantization for on-device LLM compression and acceleration. *GetMobile Mob. Comput. Commun.*, 28(4):12–17, 2024b.

Jing Liu, Ruihao Gong, Xiuying Wei, Zhiwei Dong, Jianfei Cai, and Bohan Zhuang. QLLM: accurate and efficient low-bitwidth quantization for large language models. In *ICLR*. OpenReview.net, 2024a.

Shih-Yang Liu, Zechun Liu, Xijie Huang, Pingcheng Dong, and Kwang-Ting Cheng. LLM-FP4: 4-bit floating-point quantized transformers. In *EMNLP*, pp. 592–605. Association for Computational Linguistics, 2023.

Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. LLM-QAT: data-free quantization aware training for large language models. In *ACL (Findings)*, volume ACL 2024 of *Findings of ACL*, pp. 467–484. Association for Computational Linguistics, 2024b.

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 *ICLR*. OpenReview.net, 2025.

Llama Team. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR (Poster)*. OpenReview.net, 2018.

Vladimir Malinovskii, Denis Mazur, Ivan Ilin, Denis Kuznedelev, Konstantin Burlachenko, Kai Yi, Dan Alistarh, and Peter Richtárik. Pv-tuning: Beyond straight-through estimation for extreme LLM compression. In *NeurIPS*, 2024.

Xin Men, Mingyu Xu, Qingyu Zhang, Qianhao Yuan, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen. Shortgpt: Layers in large language models are more redundant than you expect. In *ACL (Findings)*, volume ACL 2025 of *Findings of ACL*, pp. 20192–20204. Association for Computational Linguistics, 2025.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *ICLR (Poster)*. OpenReview.net, 2017.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *EMNLP*, pp. 2381–2391. Association for Computational Linguistics, 2018.

Markus Nagel, Rana Ali Amjad, Mart van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, pp. 7197–7206. PMLR, 2020.

Guilherme Penedo, Hynek Kydlícek, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin A. Raffel, Leandro von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *NeurIPS*, 2024.

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

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

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. *CoRR*, abs/1904.09728, 2019.

Meyer Scetbon and James Hensman. Low-rank correction for quantized llms. *CoRR*, abs/2412.07902, 2024.

Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. Omniquant: Omnidirectionally calibrated quantization for large language models. In *ICLR*. OpenReview.net, 2024.

Yuantian Shao, Yuanteng Chen, Peisong Wang, Jianlin Yu, Jing Lin, Yiwu Yao, Zhihui Wei, and Jian Cheng. Dartquant: Efficient rotational distribution calibration for LLM quantization. In *NeurIPS*, 2025.

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 *ICML*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR (Poster)*, 2014.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen

Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip#: Even better LLM quantization with hadamard incoherence and lattice codebooks. In *ICML*, volume 235 of *Proceedings of Machine Learning Research*, pp. 48630–48656. PMLR / OpenReview.net, 2024a.

Albert Tseng, Qingyao Sun, David Hou, and Christopher De Sa. QTIP: quantization with trellises and incoherence processing. In *NeurIPS*, 2024b.

Albert Tseng, Zhaofeng Sun, and Christopher De Sa. Model-preserving adaptive rounding. *CoRR*, abs/2505.22988, 2025.

Mart van Baalen, Andrey Kuzmin, Markus Nagel, Peter Couperus, Cédric Bastoul, Eric Mahurin, Tijmen Blankevoort, and Paul N. Whatmough. GPTVQ: the blessing of dimensionality for LLM quantization. *CoRR*, abs/2402.15319, 2024.

Tycho F. A. van der Ouderaa, Maximilian L. Croci, Agrin Hilmkil, and James Hensman. Pyramid vector quantization for llms. *CoRR*, abs/2410.16926, 2024.

Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. In *ICLR*. OpenReview.net, 2022a.

Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. Outlier suppression: Pushing the limit of low-bit transformer language models. In *NeurIPS*, 2022b.

Xiuying Wei, Yunchen Zhang, Yuhang Li, Xiangguo Zhang, Ruihao Gong, Jinyang Guo, and Xianglong Liu. Outlier suppression+: Accurate quantization of large language models by equivalent and optimal shifting and scaling. *CoRR*, abs/2304.09145, 2023.

Norbert Wiener. *Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications*. The MIT press, 1949.

Guangxuan Xiao, Ji Lin, Mickaël Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099. PMLR, 2023.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. In *NeurIPS*, 2022.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *ACL (1)*, pp. 4791–4800. Association for Computational Linguistics, 2019.

Aozhong Zhang, Naigang Wang, Yanxia Deng, Xin Li, Zi Yang, and Penghang Yin. Magr: Weight magnitude reduction for enhancing post-training quantization. In *NeurIPS*, 2024.

Shihao Zhang, Haoyu Zhang, Ian Colbert, and Rayan Saab. Qronos: Correcting the past by shaping the future... in post-training quantization. *CoRR*, abs/2505.11695, 2025.

Yilong Zhao, Chien-Yu Lin, Kan Zhu, Zihao Ye, Lequn Chen, Size Zheng, Luis Ceze, Arvind Krishnamurthy, Tianqi Chen, and Baris Kasikci. Atom: Low-bit quantization for efficient and accurate LLM serving. In *MLSys*. mlsys.org, 2024.

APPENDIX OVERVIEW

This appendix augments our main findings with further experiments and experimental setup details, additional ablations, and full proofs and derivations. We organize the appendix as follows:

- **Appendix A** reports additional experimental results and ablations that extend our main findings.
- **Appendix B** describes the full experimental setup in detail including hyperparameter details.
- **Appendix C** relates our denoising objective to the sequential corrections of GPTAQ and QEP.
- **Appendix D** provides the full proofs and derivations of our theoretical results.

A ADDITIONAL EXPERIMENTAL RESULTS AND ABLATIONS

In this section we provide additional experimental results and ablations extending our main findings.

A.1 NOISE-MODEL MECHANISM ABLATION: LOCAL VS. PROPAGATED NOISE INJECTION

Section 5.1 attributes the effect of the injected noise to two mechanisms: a local column-wise regularization from the injection $\boldsymbol{\xi}$ at each layer (effect (i), formalized by Theorem 2), and a depth-compounding regularization from propagating the injected noise downstream (effect (ii)). Figure 5 isolates the two by ablating the noise model on the 5×5 (ρ, δ) grid for Llama-3.2-1B at W3A16 (Gaussian injection, random Hadamard rotation, ridge α=0.01), under a shared color scale.

Turning off all injected noise reduces our objective to parallel GPTQ: the noise moments vanish ($\mathbf{N}=\mathbf{C}=\mathbf{0}$), so ρ and δ have no effect and the grid is flat at the parallel-GPTQ value (here 17.07 at the grid's ridge). Injecting the noise locally without propagation, i.e., on top of the clean forward pass (so each layer sees only its own fresh $\boldsymbol{\xi}$), already closes 11% of the parallel–sequential gap, confirming that effect (i) is a real standalone regularizer. With propagation, so that the injected noise compounds across depth as in our full method, the gain roughly doubles to 28%. The optimum sits at the same cell (ρ=0.45, δ=0.0375) in both cases. Thus both effects contribute, with the propagated, depth-compounding term (ii) providing the majority of the benefit.

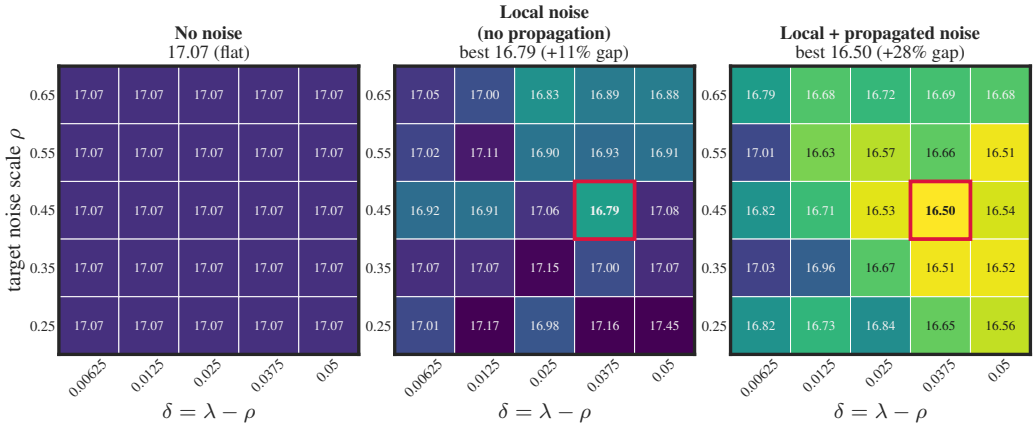


Figure 5: **Noise-model mechanism ablation** (Llama-3.2-1B W3A16, Gaussian denoising, random Hadamard rotation, α=0.01). Average perplexity over the 5×5 (ρ, δ) grid. Each title reports the best value and the % of the parallel–sequential gap it closes. *Left*: no injected noise, so the objective collapses to parallel GPTQ and ρ, δ have no effect. *Middle*: local injection only (noise enters the moments but does not propagate), closing 11% of the gap. *Right*: the full method with propagated noise, closing 28%. Local injection alone already improves over parallel GPTQ, while propagation accounts for the majority of the improvement.

A.2 NOISE-SCALE LANDSCAPE: PARTIAL DENOISING AND GENERALIZING SEQUENTIAL QUANTIZATION

We explore the full range of both noise scales, $\rho, \lambda \in [0, 1]$, on a 9×9 grid for Llama-3.2-1B W3A16:

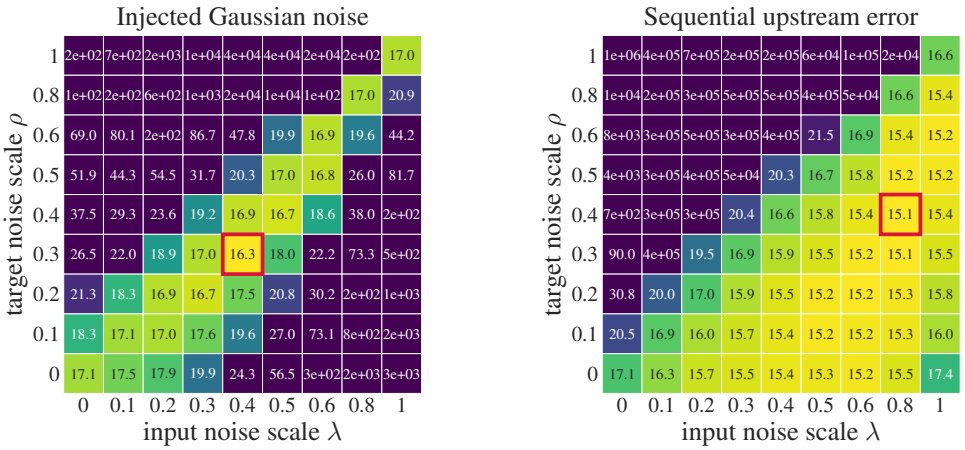


Figure 6: **Full $\rho \times \lambda$ sweep**, Llama-3.2-1B W3A16 (9×9 grid over $[0, 1]^2$). *Left*: injected Gaussian noise (best avg. perplexity 16.31 at λ=0.4, ρ=0.3). *Right*: the real sequential upstream error (best 15.12 at λ=0.8, ρ=0.4). In both, λ is the input noise scale and ρ the target scale; the low-perplexity region is below the diagonal ($\rho<\lambda$).

Improving upon sequential asymmetric correction. We compare the injected-noise proxy against the real sequential upstream error $\boldsymbol{\eta}_{seq}$ (Figure 6, right). With this error, the asymmetric corrections of GPTAQ (Li et al., 2025) and QEP (Arai & Ichikawa, 2025) are special cases of our objective: both fix the input noise scale to λ=1 and only tune the correction strength $1-\rho$ (Appendix C), which restricts them to the rightmost column of the landscape. Our framework instead treats λ as a free parameter. Along the column λ=1, the best configuration reaches an average perplexity of 15.2 ($\rho \in \{0.5, 0.6\}$), whereas the optimum of our sweep at λ=0.8, ρ=0.4 reaches 15.12. Scaling down the upstream error ($\lambda<1$) thus improves upon the best asymmetric correction, even though both use the same sequential error. The flexibility of our objective therefore benefits not only parallel quantization with a noise proxy, but also sequential quantization.

Why the optimum lies below the diagonal. We find the best configurations sit below the diagonal ($\rho<\lambda$). This shape follows from the objective in Equation 2. On the diagonal $\rho=\lambda$, the first factor of $\mathbf{F}$ in Equation 3 equals $\mathbf{G}$, so $\mathbf{F}=\mathbf{I}$ and $\tilde{\mathbf{W}}=\mathbf{W}$: the noise only changes the rounding metric $\mathbf{G}$ and does not correct the weights. Off the diagonal, Theorem 2 shows that injected noise pulls each weight column toward $s_c\mathbf{W}_{:,c}$, where $s_c<1$ if and only if $\rho<\lambda$. Below the diagonal the objective therefore shrinks the weights, which makes the layer more robust to input perturbations. Above the diagonal ($\rho>\lambda$) we instead have $s_c>1$: the target carries more noise than the input, so the layer is asked to amplify the error and its weights grow. This is consistent with the poor performance above the diagonal and with the divergence with large ρ and small λ.

A.3 FULL RESULT TABLES WITH HYPERPARAMETER SWEEP HEATMAPS

We provide the full tables of our W2/W3/W4 weight quantization results with detailed hyperparameter heatmaps of the sweeps used to optimize hyperparameters for each model and bit-width.

Proxy-model quantization. The random noise models a signal-independent input perturbation, providing a principled regularization approach to input activation denoising. Here we provide additional results for a second noise model: we propose to estimate upstream error from a model quantized in parallel, which is particularly efficient when a previously quantized model is already available and can be reused to further improve the quantization quality. Concretely, we quantize an LLM once in a single parallel pass (for example with parallel GPTQ at the deployed bit-width), and estimate the noise from the difference of input activations between the full-precision and proxy-quantized models. Instead of expensive sequential quantization, the cost of this proxy model corresponds to an additional parallel quantization pass to obtain the proxy quantized model, which can then be used for denoising the input activations in a second parallel quantization pass. In practice one may quantize a dedicated proxy or just reuse an existing quantized model at no additional cost. In either case, the proxy model can provide a more accurate estimate of the deployed error than random noise injection alone. The following results therefore include both random noise injection and proxy-model denoising results, which we find to be working best when combined. The results suggest that any existing parallel-GPTQ quantized model can be used as a proxy to improve the quantization quality in a second parallel pass with our activation denoising.

The tables and figures are organized as follows:

- Table 2: W3A16 summary across all four models and both rotations.
- Table 3: Llama-3.2-1B at W3A16 (GPTQ), sweeps in Figure 7 and Figure 8.
- Table 4: Llama-3.2-3B at W3A16 (GPTQ), sweeps in Figure 9 and Figure 10.
- Table 5: Llama-3-8B at W3A16 (GPTQ), sweeps in Figure 11 and Figure 12.
- Table 6: Llama-2-13B at W3A16 (GPTQ), sweeps in Figure 13 and Figure 14.
- Table 7: Llama-3.2-1B at W4A16 (GPTQ), sweeps in Figure 15 and Figure 16.
- Table 8: Llama-3.2-1B at W2A16 (QuIP#), sweeps in Figure 17.
- Table 9: Llama-3.2-3B at W2A16 (QuIP#), sweeps in Figure 18.
- Table 10: Llama-3-8B at W2A16 (QuIP#), sweeps in Figure 19.
- Table 11: Llama-2-13B at W2A16 (QuIP#), sweeps in Figure 20.

A.3.1 W3A16 GPTQ SUMMARY ACROSS MODELS

Table 2: **W3A16 GPTQ summary across Llama-3.2-1B/3B, Llama-3-8B, and Llama-2-13B.** Same quantization objectives as Table 1, here reported for the fixed random-Hadamard and the learned SpinQuant rotation side by side, including the GPTQ-W3 proxy noise.

		Quantization objective	Llama 1B			Llama 3B			Llama 8B			Llama 13B		
			Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑
		FP16	54.8	11.33	–	62.6	9.17	–	67.1	7.57	–	66.5	5.73	–
Random rot.	Parallel	symmetric correction	50.0	16.97	0%	58.5	12.11	0%	63.5	9.93	0%	64.0	6.40	0%
		Denoising reg. (**ours**)	49.6	16.50	28%	58.9	11.91	29%	63.4	9.78	35%	64.6	6.37	17%
		+ GPTQ-W3 proxy	50.1	16.24	**43**%	58.3	11.85	**38**%	63.5	9.76	**40**%	64.2	6.33	**34**%
	Seq.	symmetric correction	49.8	16.53	26%	59.5	11.99	17%	63.5	9.81	27%	63.7	6.38	10%
		asymmetric correction	50.3	15.29	100%	58.4	11.41	100%	64.0	9.50	100%	64.4	6.21	100%
Learned rot.	Parallel	symmetric correction	49.6	16.39	0%	58.3	11.88	0%	63.8	9.70	0%	65.2	6.33	0%
		Denoising reg. (**ours**)	49.1	16.02	31%	58.5	11.82	11%	63.4	9.65	20%	65.1	6.31	12%
		+ GPTQ-W3 proxy	49.5	15.94	**37**%	58.3	11.76	**23**%	63.9	9.65	**20**%	64.6	6.30	**20**%
	Seq.	symmetric correction	49.9	16.15	20%	59.2	11.83	9%	64.0	9.65	20%	64.8	6.32	7%
		asymmetric correction	50.2	15.17	100%	59.8	11.35	100%	63.8	9.43	100%	65.0	6.19	100%

A.3.2 LLAMA-3.2-1B (W3A16 GPTQ)

Table 3: **Llama-3.2-1B at W3A16 (GPTQ).** The upper block uses the fixed random-Hadamard rotation, the lower block the SpinQuant-learned rotation; rows marked *(ours)* are our denoising objective. Gauss. denotes injected Gaussian noise, W3 corresponds to the additional GPTQ-W3 proxy noise, and W3+Gauss combines both.

		Method	Bits	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
		FP16	16	–	–	54.8	9.76	11.54	12.70	11.33	–
Random rot.	Parallel	GPTQ	3.00	25	damp=0.1	50.0	13.44	18.53	18.92	16.97	0%
		Gauss. *(ours)*	3.00	25	ρ=0.45, δ=0.0375	49.6	12.95	18.06	18.49	16.50	28%
		W3 *(ours)*	3.00	25	ρ=0.65, δ=0.05	50.1	13.01	18.04	18.53	16.53	26%
		W3+Gauss *(ours)*	3.00	25	ρ=0.35, δ=0.0375	50.1	12.85	17.64	18.23	16.24	43%
	Seq.	GPTQ	3.00	25	damp=0.05	49.8	12.99	18.03	18.56	16.53	26%
		GPTAQ	3.00	25	β=0.5833	50.3	11.87	16.77	17.22	15.29	100%
Learned rot.	Parallel	GPTQ	3.00	25	damp=0.2	49.6	12.94	17.81	18.44	16.39	0%
		Gauss. *(ours)*	3.00	25	ρ=0.55, δ=0.05	49.1	12.58	17.39	18.08	16.02	31%
		W3 *(ours)*	3.00	25	ρ=0.35, δ=0.0375	50.0	12.76	17.80	18.26	16.27	10%
		W3+Gauss *(ours)*	3.00	25	ρ=0.35, δ=0.025	49.5	12.58	17.39	17.84	15.94	37%
	Seq.	GPTQ	3.00	25	damp=0.1	49.9	12.77	17.52	18.17	16.15	20%
		GPTAQ	3.00	25	β=0.4167	50.2	11.86	16.56	17.11	15.17	100%

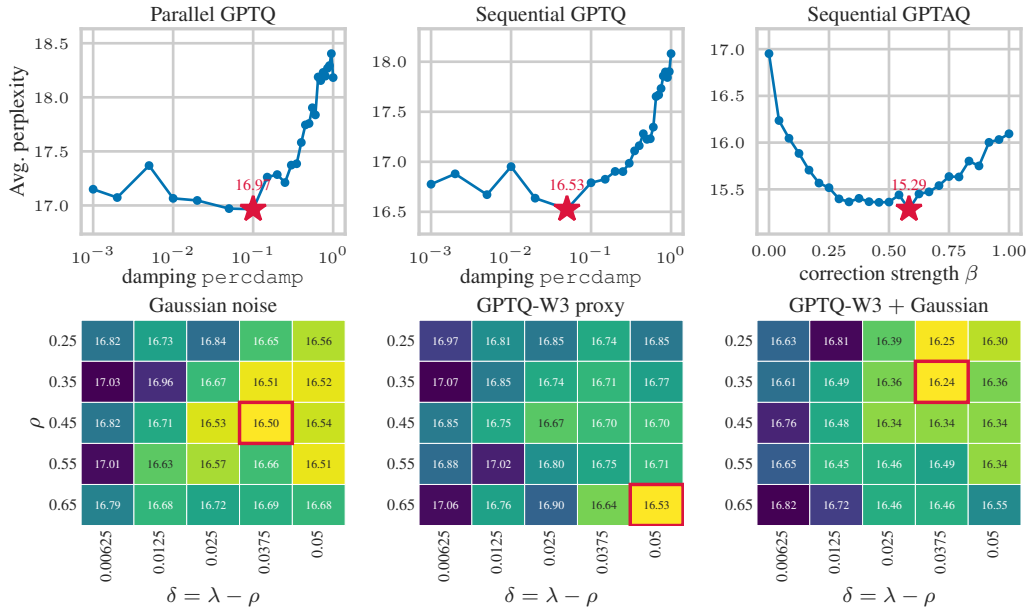


Figure 7: **Llama-3.2-1B W3A16 hyperparameter sweeps, fixed random-Hadamard rotation.** *Top*: the one-dimensional baseline sweeps, avg. perplexity against the swept knob, with the selected optimum starred. *Bottom*: our denoising objective over its 5×5 grid of the two swept knobs ρ and δ (which sets the noise scale through λ=ρ+δ); one heatmap per upstream-error proxy (Gaussian noise, the GPTQ-W3 proxy, and their combination), with the best (minimum-perplexity) cell boxed. The starred/boxed optima are the values reported in the random-rotation block of Table 3.

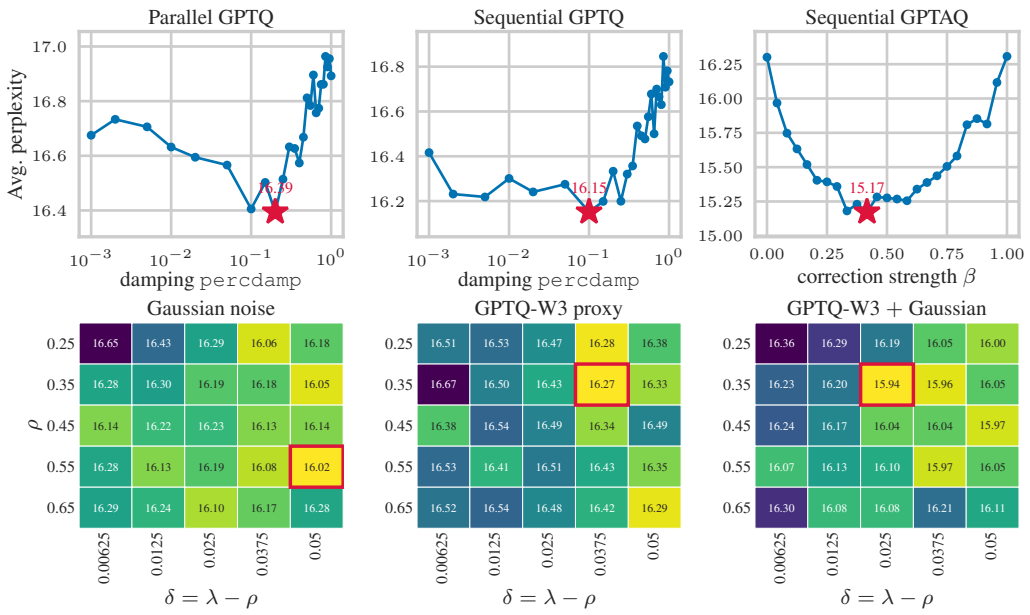


Figure 8: **Llama-3.2-1B W3A16 hyperparameter sweeps, learned SpinQuant rotation.** Panels as in Fig. 7; the starred/boxed optima are the values reported in the learned-rotation block of Tab. 3.

A.3.3 LLAMA-3.2-3B (W3A16 GPTQ)

Table 4: **Llama-3.2-3B at W3A16 (GPTQ).** Columns and rotation blocks as in Table 3. Gauss. denotes injected Gaussian noise, W3 corresponds to the additional GPTQ-W3 proxy noise, and W3+Gauss combines both.

		Method	Bits	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
		FP16	16	–	–	62.6	7.82	9.29	10.41	9.17	–
Random rot.	Parallel	GPTQ	3.00	25	damp=0.001	58.5	9.61	13.14	13.60	12.11	0%
		Gauss. *(ours)*	3.00	25	ρ=0.55, δ=0.025	58.9	9.47	12.88	13.37	11.91	29%
		W3 *(ours)*	3.00	25	ρ=0.35, δ=0.05	58.7	9.46	13.02	13.45	11.98	19%
		W3+Gauss *(ours)*	3.00	25	ρ=0.45, δ=0.025	58.3	9.45	12.80	13.30	11.85	38%
	Seq.	GPTQ	3.00	25	damp=0.05	59.5	9.65	12.93	13.40	11.99	17%
		GPTAQ	3.00	25	β=0.4583	58.4	8.94	12.44	12.85	11.41	100%
Learned rot.	Parallel	GPTQ	3.00	25	damp=0.4	58.3	9.41	12.86	13.38	11.88	0%
		Gauss. *(ours)*	3.00	25	ρ=0.45, δ=0.025	58.5	9.31	12.82	13.33	11.82	11%
		W3 *(ours)*	3.00	25	ρ=0.55, δ=0.05	58.7	9.34	12.89	13.41	11.88	0%
		W3+Gauss *(ours)*	3.00	25	ρ=0.55, δ=0.025	58.3	9.29	12.74	13.26	11.76	23%
	Seq.	GPTQ	3.00	25	damp=0.05	59.2	9.39	12.83	13.27	11.83	9%
		GPTAQ	3.00	25	β=0.4167	59.8	8.93	12.36	12.77	11.35	100%

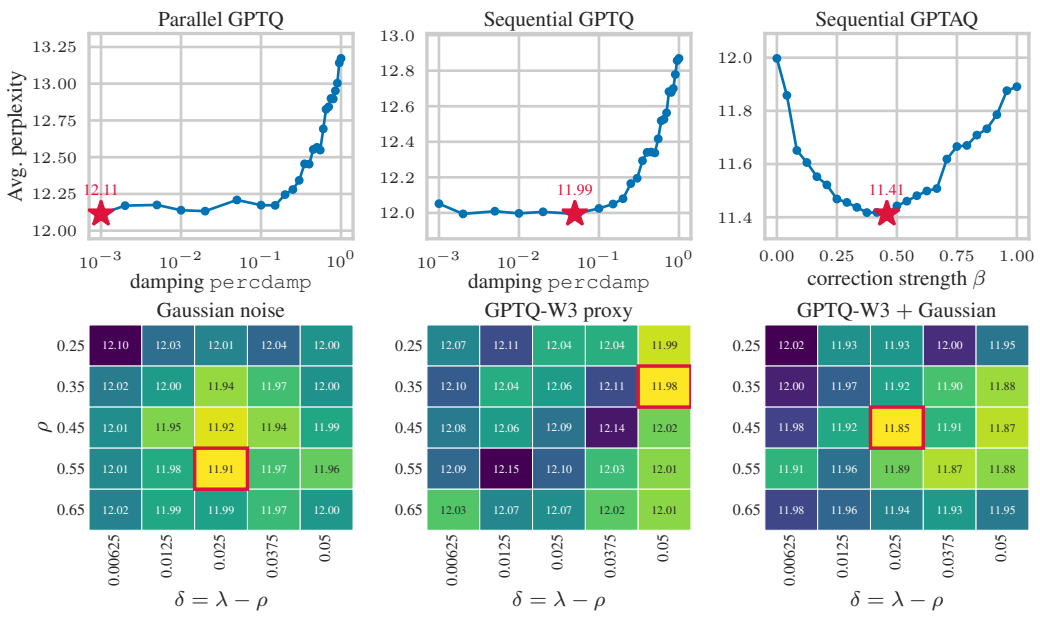


Figure 9: **Llama-3.2-3B W3A16 hyperparameter sweeps, fixed random-Hadamard rotation.** *Top*: the one-dimensional baseline sweeps, avg. perplexity against the swept knob, with the selected optimum starred. *Bottom*: our denoising objective over its 5×5 grid of the two swept knobs ρ and δ (which sets the noise scale through $\lambda{=}\rho{+}\delta$); one heatmap per upstream-error proxy (Gaussian noise, the GPTQ-W3 proxy, and their combination), with the best (minimum-perplexity) cell boxed.

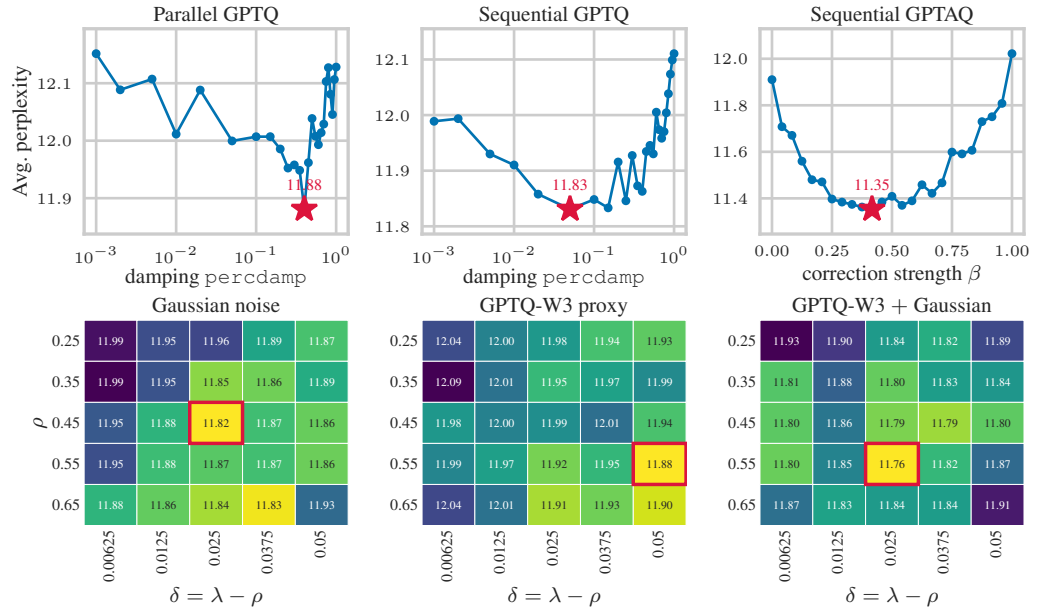


Figure 10: **Llama-3.2-3B W3A16 hyperparameter sweeps, learned SpinQuant rotation.** Panels as in Fig. 9; the starred/boxed optima are the values reported in the learned-rotation block of Tab. 4.

A.3.4 LLAMA-3-8B (W3A16 GPTQ)

Table 5: **Llama-3-8B at W3A16 (GPTQ).** Columns and rotation blocks as in Table 3. Gauss. denotes injected Gaussian noise, W3 corresponds to the additional GPTQ-W3 proxy noise, and W3+Gauss combines both.

		Method	Bits	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
		FP16	16	–	–	67.1	6.14	7.78	8.78	7.57	–
Random rot.	Parallel	GPTQ	3.00	25	damp=0.05	63.5	7.64	10.84	11.31	9.93	0%
		Gauss. *(ours)*	3.00	25	ρ=0.65, δ=0.00625	63.4	7.57	10.65	11.12	9.78	35%
		W3 *(ours)*	3.00	25	ρ=0.55, δ=0.05	64.0	7.53	10.78	11.19	9.83	22%
		W3+Gauss *(ours)*	3.00	25	ρ=0.65, δ=0.00625	63.5	7.55	10.64	11.09	9.76	40%
	Seq.	GPTQ	3.00	25	damp=0.02	63.5	7.56	10.72	11.16	9.81	27%
		GPTAQ	3.00	25	β=0.3333	64.0	7.24	10.44	10.82	9.50	100%
Learned rot.	Parallel	GPTQ	3.00	25	damp=0.25	63.8	7.43	10.60	11.07	9.70	0%
		Gauss. *(ours)*	3.00	25	ρ=0.65, δ=0.00625	63.4	7.40	10.56	10.98	9.65	20%
		W3 *(ours)*	3.00	25	ρ=0.65, δ=0.05	63.3	7.43	10.66	11.08	9.72	−8%
		W3+Gauss *(ours)*	3.00	25	ρ=0.45, δ=0.025	63.9	7.37	10.56	11.01	9.65	20%
	Seq.	GPTQ	3.00	25	damp=0.15	64.0	7.40	10.56	10.98	9.65	20%
		GPTAQ	3.00	25	β=0.3333	63.8	7.19	10.35	10.75	9.43	100%

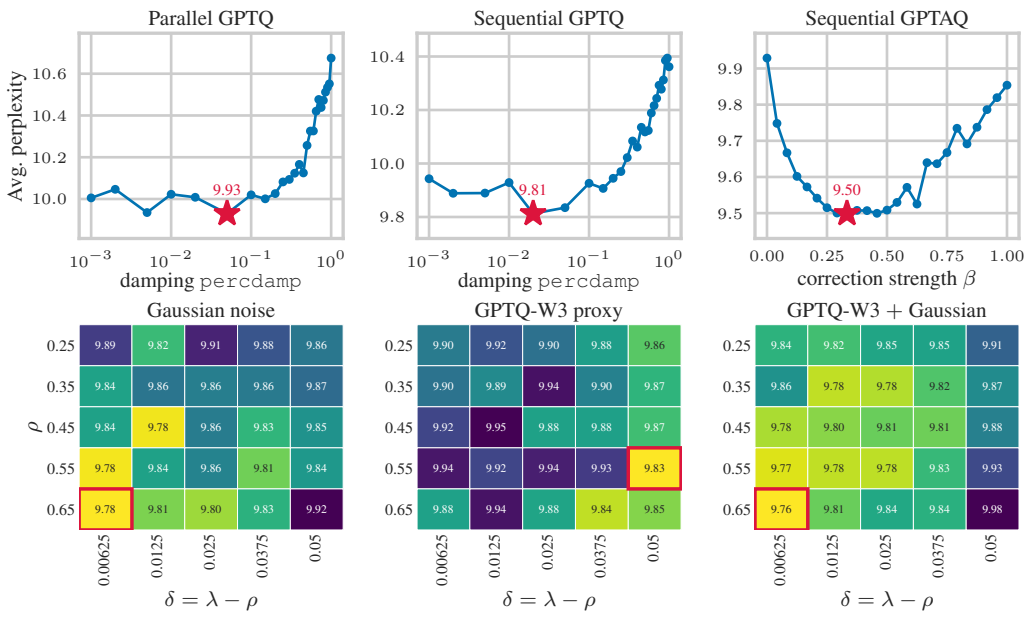


Figure 11: **Llama-3-8B W3A16 hyperparameter sweeps, fixed random-Hadamard rotation.** *Top*: the one-dimensional baseline sweeps, avg. perplexity against the swept knob, with the selected optimum starred. *Bottom*: our denoising objective over its 5×5 grid of the two swept knobs ρ and δ (which sets the noise scale through λ=ρ+δ); one heatmap per upstream-error proxy (Gaussian noise, the GPTQ-W3 proxy, and their combination), with the best (minimum-perplexity) cell boxed. The starred/boxed optima are the values reported in the random-rotation block of Table 5.

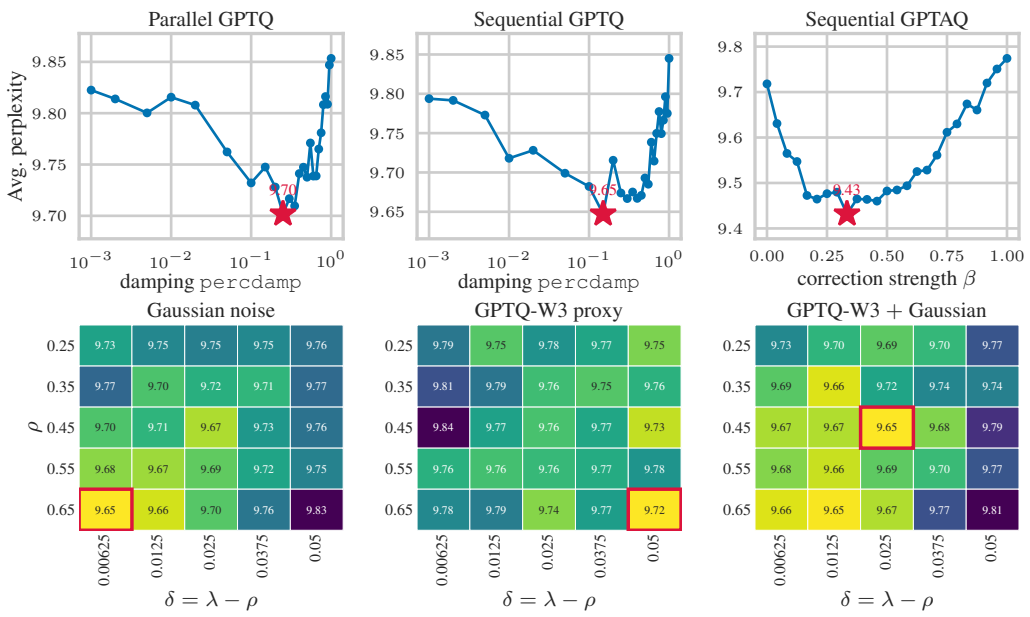


Figure 12: **Llama-3-8B W3A16 hyperparameter sweeps, learned SpinQuant rotation.** Panels as in Fig. 11; the starred/boxed optima are the values reported in the learned-rotation block of Tab. 5.

A.3.5 LLAMA-2-13B (W3A16 GPTQ)

Table 6: **Llama-2-13B at W3A16 (GPTQ).** Columns and rotation blocks as in Table 3. Gauss. denotes injected Gaussian noise, W3 corresponds to the additional GPTQ-W3 proxy noise, and W3+Gauss combines both.

		Method	Bits	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
		FP16	16	–	–	66.5	4.88	5.80	6.49	5.73	–
Random rot.	Parallel	GPTQ	3.00	25	damp=0.02	64.0	5.38	6.65	7.17	6.40	0%
		Gauss. *(ours)*	3.00	25	ρ=0.55, δ=0.00625	64.6	5.35	6.61	7.14	6.37	17%
		W3 *(ours)*	3.00	25	ρ=0.25, δ=0.025	64.3	5.33	6.62	7.14	6.36	18%
		W3+Gauss *(ours)*	3.00	25	ρ=0.35, δ=0.0375	64.2	5.29	6.60	7.11	6.33	34%
	Seq.	GPTQ	3.00	25	damp=0.002	63.7	5.35	6.64	7.15	6.38	10%
		GPTAQ	3.00	25	β=0.5417	64.4	5.21	6.44	6.99	6.21	100%
Learned rot.	Parallel	GPTQ	3.00	25	damp=0.1	65.2	5.27	6.58	7.12	6.33	0%
		Gauss. *(ours)*	3.00	25	ρ=0.35, δ=0.025	65.1	5.25	6.58	7.10	6.31	12%
		W3 *(ours)*	3.00	25	ρ=0.25, δ=0.05	64.2	5.26	6.58	7.10	6.31	10%
		W3+Gauss *(ours)*	3.00	25	ρ=0.35, δ=0.025	64.6	5.25	6.56	7.09	6.30	20%
	Seq.	GPTQ	3.00	25	damp=0.05	64.8	5.27	6.57	7.11	6.32	7%
		GPTAQ	3.00	25	β=0.4583	65.0	5.19	6.42	6.97	6.19	100%

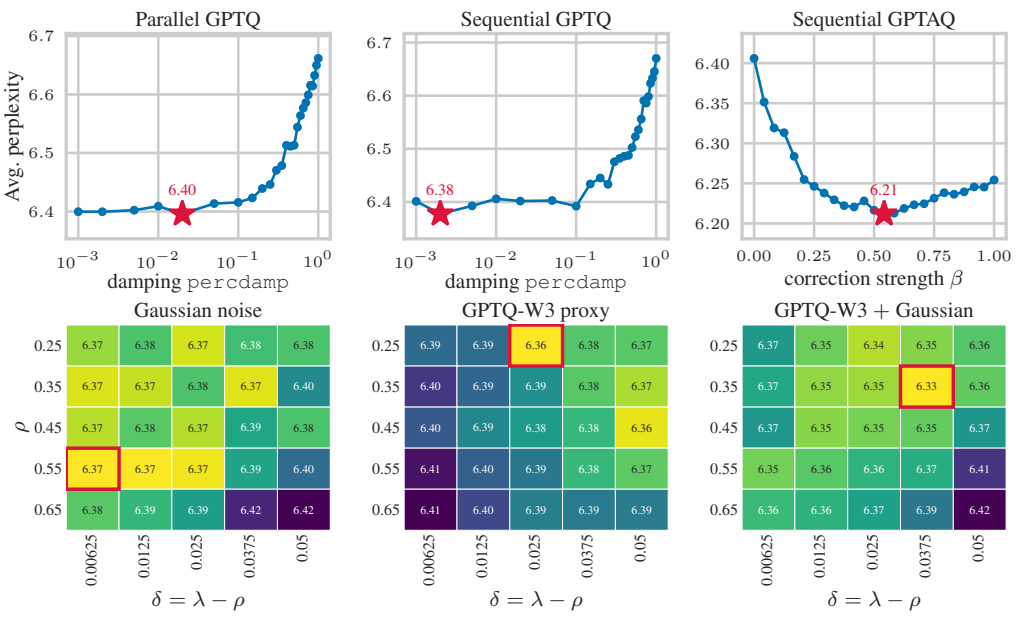


Figure 13: **Llama-2-13B W3A16 hyperparameter sweeps, fixed random-Hadamard rotation.** *Top*: the one-dimensional baseline sweeps, avg. perplexity against the swept knob, with the selected optimum starred. *Bottom*: our denoising objective over its 5×5 grid of the two swept knobs ρ and δ (which sets the noise scale through $\lambda{=}\rho{+}\delta$); one heatmap per upstream-error proxy (Gaussian noise, the GPTQ-W3 proxy, and their combination), with the best (minimum-perplexity) cell boxed. The starred/boxed optima are the values reported in the random-rotation block of Table 6.

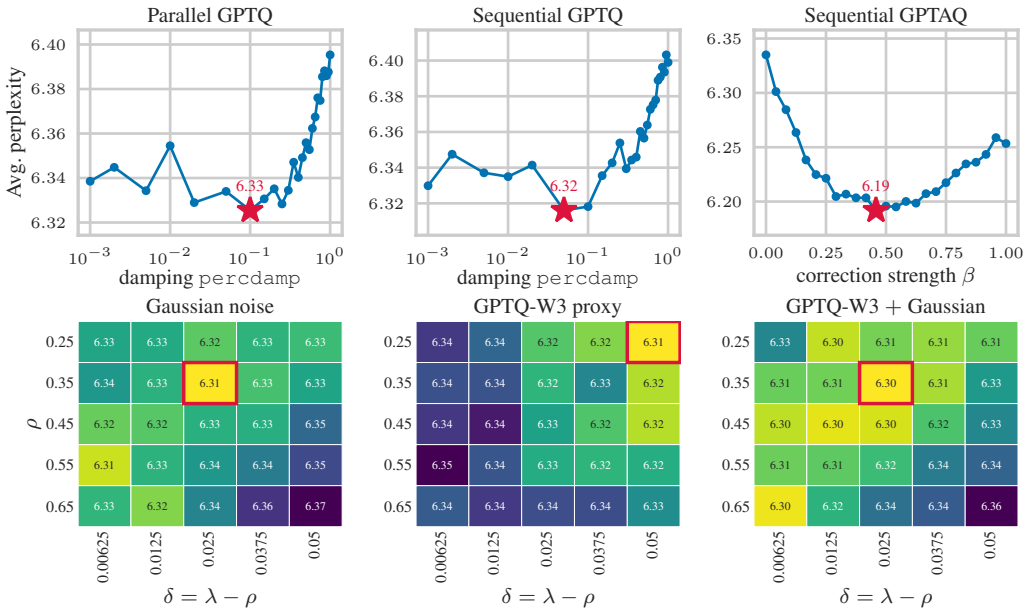


Figure 14: **Llama-2-13B W3A16 hyperparameter sweeps, learned SpinQuant rotation.** Panels as in Fig. 13; the starred/boxed optima are the values reported in the learned-rotation block of Tab. 6.

A.3.6 LLAMA-3.2-1B (W4A16 GPTQ)

Table 7: **Llama-3.2-1B at W4A16 (GPTQ).** Columns and rotation blocks as in Table 3. Gauss. denotes injected Gaussian noise, W4 corresponds to the additional GPTQ-W4 proxy noise, and W4+Gauss combines both. At 4 bits the parallel–sequential gap is small (≈ 0.26 perplexity).

		Method	Bits	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
		FP16	16	–	–	54.8	9.76	11.54	12.70	11.33	–
Random rot.	Parallel	GPTQ	4.00	25	damp=0.05	53.2	10.49	13.27	14.11	12.62	0%
		Gauss. *(ours)*	4.00	25	ρ=0.45, δ=0.025	53.4	10.42	13.20	14.05	12.56	26%
		W4 *(ours)*	4.00	25	ρ=0.55, δ=0.0375	53.9	10.47	13.26	14.09	12.61	6%
		W4+Gauss *(ours)*	4.00	25	ρ=0.65, δ=0.0125	53.6	10.44	13.16	14.04	12.55	30%
	Seq.	GPTQ	4.00	25	damp=0.1	53.6	10.50	13.24	14.08	12.61	6%
		GPTAQ	4.00	25	β=0.4583	54.2	10.25	13.01	13.84	12.37	100%
Learned rot.	Parallel	GPTQ	4.00	25	damp=0.15	53.5	10.44	13.20	14.06	12.57	0%
		Gauss. *(ours)*	4.00	25	ρ=0.65, δ=0.025	53.5	10.37	13.14	14.03	12.51	29%
		W4 *(ours)*	4.00	25	ρ=0.65, δ=0.05	53.0	10.40	13.25	14.07	12.57	−5%
		W4+Gauss *(ours)*	4.00	25	ρ=0.65, δ=0.025	53.0	10.35	13.18	14.05	12.53	19%
	Seq.	GPTQ	4.00	25	damp=0.05	53.4	10.41	13.20	14.06	12.56	4%
		GPTAQ	4.00	25	β=0.4167	52.9	10.27	13.01	13.86	12.38	100%

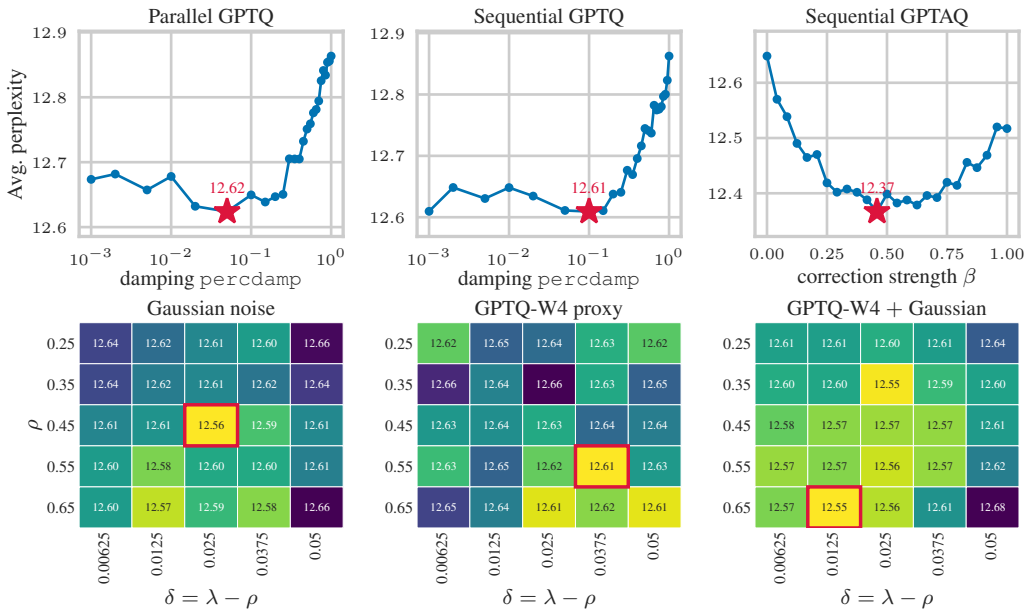


Figure 15: **Llama-3.2-1B W4A16 hyperparameter sweeps, fixed random-Hadamard rotation.** *Top*: the one-dimensional baseline sweeps, avg. perplexity against the swept knob, with the selected optimum starred. *Bottom*: our denoising objective over its 5×5 grid of the two swept knobs ρ and δ (which sets the noise scale through λ=ρ+δ); one heatmap per upstream-error proxy (Gaussian noise, the GPTQ-W4 proxy, and their combination), with the best (minimum-perplexity) cell boxed. The starred/boxed optima are the values reported in the random-rotation block of Table 7.

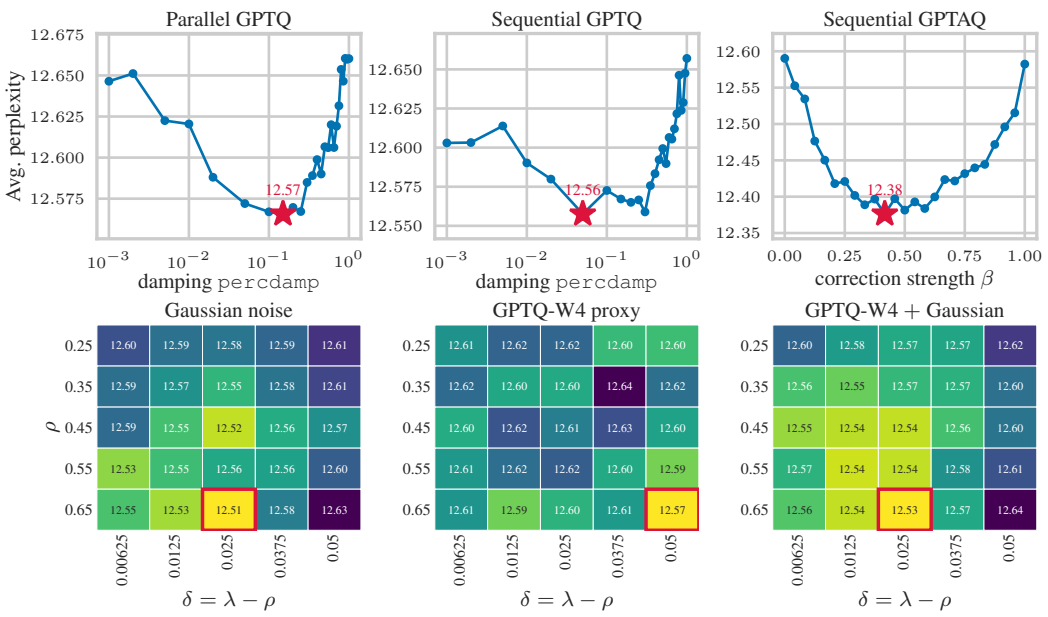


Figure 16: **Llama-3.2-1B W4A16 hyperparameter sweeps, learned SpinQuant rotation.** Panels as in Fig. 15; the starred/boxed optima are the values reported in the learned-rotation block of Tab. 7.

A.3.7 LLAMA-3.2-1B (W2A16 QUIP#)

Table 8: **QuIP# at W2A16 (2-bit), Llama-3.2-1B.** Parallel: symmetric correction quantizes each layer from clean activations, Denoising reg. is our objective (Gaussian noise injection, ρ/δ grid). Sequential: symmetric correction uses the symmetric objective on the sequential input, i.e. the outputs of the already-quantized upstream layers; asymmetric correction additionally targets the clean full-precision output (GPTAQ-correction applied to QuIP#). Both then apply E8 rounding. Acc: zero-shot average accuracy (%). Gap: 0%=parallel symmetric, 100%=sequential asymmetric.

	Method	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
	FP16	–	–	54.8	9.76	11.54	12.70	11.33	–
Parallel	symmetric correction	25	damp=0.55	44.3	19.13	30.36	31.66	27.05	0%
Parallel	Denoising reg. (**ours**)	25	ρ=0.25, δ=0.0125	45.1	18.01	28.46	30.07	25.51	38%
Seq.	symmetric correction	25	damp=0.3	43.8	19.05	29.14	30.88	26.36	17%
Seq.	asymmetric correction	25	β=0.3333	45.1	16.30	25.67	27.15	23.04	100%

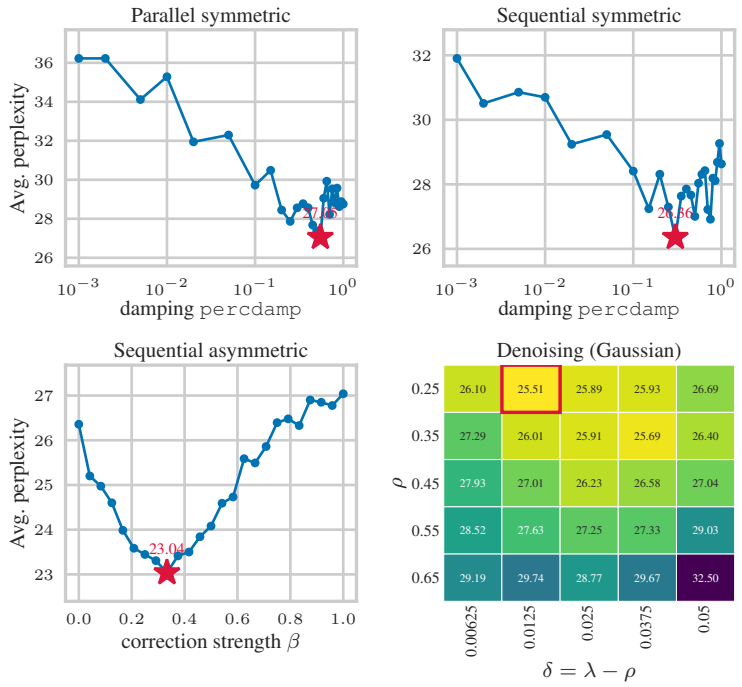


Figure 17: **QuIP# W2A16 hyperparameter sweeps, Llama-3.2-1B.** The three baseline sweeps (parallel symmetric, sequential symmetric, sequential asymmetric), avg. perplexity against the swept knob with the selected optimum starred, and our denoising objective over its 5×5 grid of ρ and δ (which sets the noise scale through λ=ρ+δ), best cell boxed. The starred/boxed optima are the values reported in Table 8.

A.3.8 LLAMA-3.2-3B (W2A16 QUIP#)

Table 9: **QuIP# at W2A16 (2-bit), Llama-3.2-3B.** Columns and blocks as in Table 8.

	Method	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
	FP16	–	–	62.6	7.82	9.29	10.41	9.17	–
Parallel	symmetric correction	25	damp=1	53.4	12.44	18.12	18.87	16.48	0%
Parallel	Denoising reg. (**ours**)	25	ρ=0.35, δ=0.025	53.9	11.61	17.21	17.90	15.57	84%
Seq.	symmetric correction	25	damp=0.8	53.7	12.42	17.84	18.61	16.29	17%
Seq.	asymmetric correction	25	β=0.3333	51.3	11.23	17.12	17.84	15.40	100%

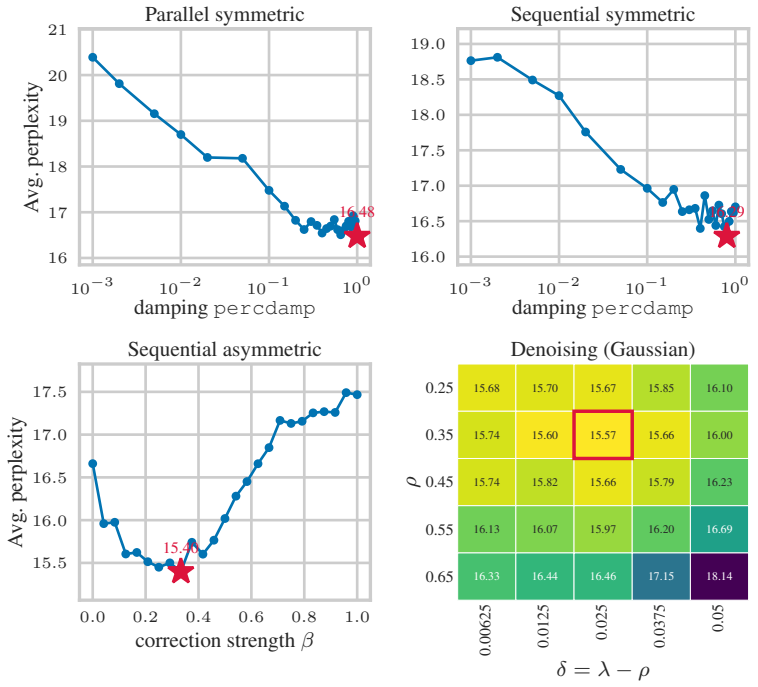


Figure 18: **QuIP# W2A16 hyperparameter sweeps, Llama-3.2-3B.** Panels as in Figure 17; the starred/boxed optima are the values reported in Table 9.

A.3.9 LLAMA-3-8B (W2A16 QUIP#)

Table 10: **QuIP# at W2A16 (2-bit), Llama-3-8B.** Columns and blocks as in Table 8.

	Method	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
	FP16	–	–	67.1	6.14	7.78	8.78	7.57	–
Parallel	symmetric correction	25	damp=0.7	58.8	9.69	15.26	15.78	13.58	0%
Parallel	Denoising reg. (**ours**)	25	ρ=0.35, δ=0.0375	59.1	9.17	14.16	14.70	12.67	128%
Seq.	symmetric correction	25	damp=0.35	58.0	10.02	15.55	16.10	13.89	−44%
Seq.	asymmetric correction	25	β=0.3333	56.2	9.05	14.55	15.01	12.87	100%

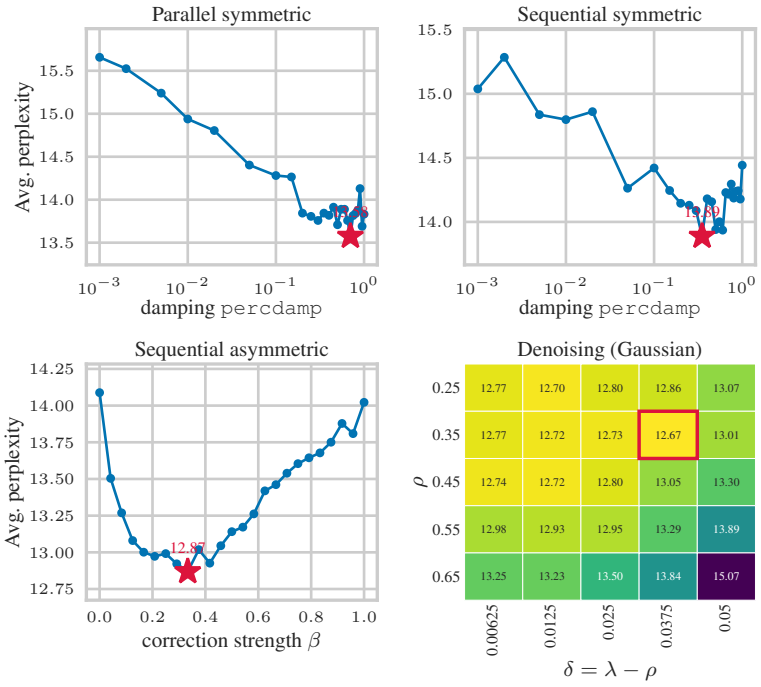


Figure 19: **QuIP# W2A16 hyperparameter sweeps, Llama-3-8B.** Panels as in Figure 17; the starred/boxed optima are the values reported in Table 10.

A.3.10 LLAMA-2-13B (W2A16 QUIP#)

Table 11: **QuIP# at W2A16 (2-bit), Llama-2-13B.** Columns and blocks as in Table 8. Here parallel-symmetric and sequential-asymmetric nearly coincide.

	Method	Grid	Setting	Acc ↑	Wiki ↓	C4 ↓	FW ↓	Avg ↓	Gap ↑
	FP16	–	–	66.5	4.88	5.80	6.49	5.73	–
Parallel	symmetric correction	25	damp=0.8	61.9	5.91	7.78	8.11	7.27	0%
Parallel	Denoising reg. (**ours**)	25	ρ=0.35, δ=0.0125	61.4	5.84	7.69	8.04	7.19	>100%
Seq.	symmetric correction	25	damp=0.85	60.9	6.00	7.92	8.26	7.39	<0%
Seq.	asymmetric correction	25	β=0.25	60.0	5.81	7.84	8.14	7.26	100%

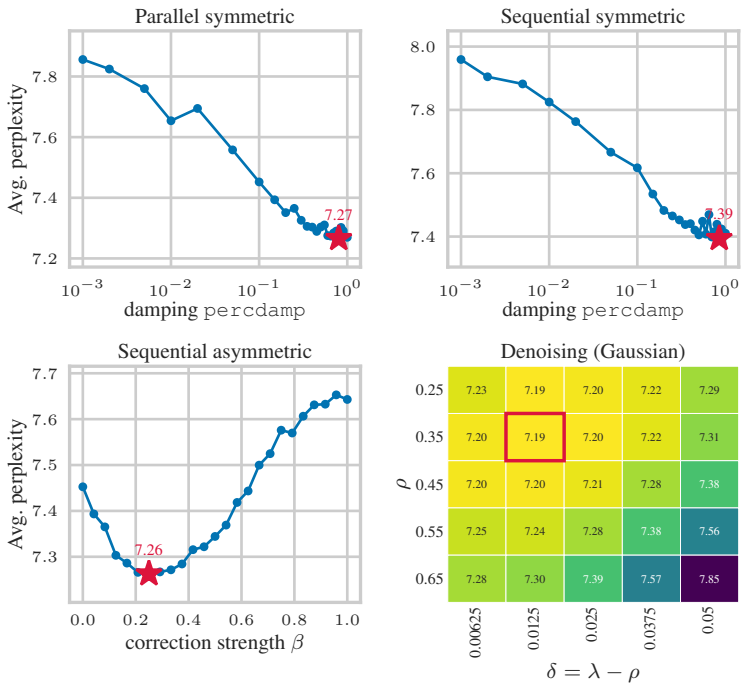


Figure 20: **QuIP# W2A16 hyperparameter sweeps, Llama-2-13B.** Panels as in Figure 17; the starred/boxed optima are the values reported in Table 11.

A.4 Injected-noise scale ablation

We fix the injected-noise scale to γ=0.05 throughout (Appendix B.3). Figure 21 sweeps γ for our objective on Llama-3.2-1B W3A16 (at λ=0.4, ρ=0.3, random Hadamard rotation), over ten log-spaced values in $[0.01, 0.2]$. The best average perplexity (16.59) is reached at γ=0.053, and nearby values perform similarly. For γ above 0.07, the perplexity increases quickly.

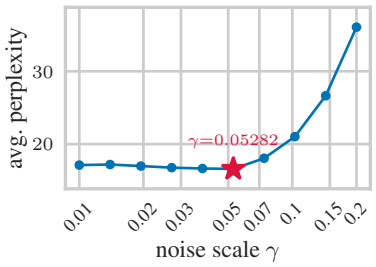


Figure 21: **Injected-noise scale γ sweep** (Llama-3.2-1B W3A16, λ=0.4, ρ=0.3, random Hadamard rotation). Average perplexity against γ over ten log-spaced values in $[0.01, 0.2]$.

A.5 Random noise model under forward pass modifications

For a fair comparison, we tune every method with equally sized hyperparameter sweeps (Appendix B.3) and use a noise scale of γ=0.05 throughout. Figure 22 compares two noise models on Llama-3.2-1B for W3 GPTQ and W2 QuIP#: module-wise isotropic noise $\mathbf{\Sigma}_m=\sigma_m^2\mathbf{I}$ with $\sigma_m=\gamma\,\mathbb{E}_t\|\mathbf{x}_{m,t}\|_\infty$, and per-channel noise $\mathbf{\Sigma}_m=\mathrm{diag}(\sigma_{m,c}^2)$ with $\sigma_{m,c}=\gamma\max_t|x_{m,c,t}|$.

Per-channel noise improves over the parallel baseline in both settings. Isotropic noise also works for GPTQ, but for QuIP# it stays above the parallel baseline for every ρ and every ridge α we tried. We attribute this to the forward pass used to estimate the noise moments: QuaRot fuses its rotations into the model before we estimate the moments, so the activations in this forward pass are already rotated. QuIP#, in contrast, applies its incoherence transform only to the weights after we estimate moments, so the activations are not rotated and potentially contain more outliers. Per-channel noise scales with the magnitude of each channel and therefore regularizes these outlier channels more strongly. Since QuaRot is already processing such outliers, we use uniform noise as the default for W3 GPTQ, where Theorem 2 reduces to a uniform down-scaling of all weight columns. For W2 QuIP#, we use per-channel noise as discussed.

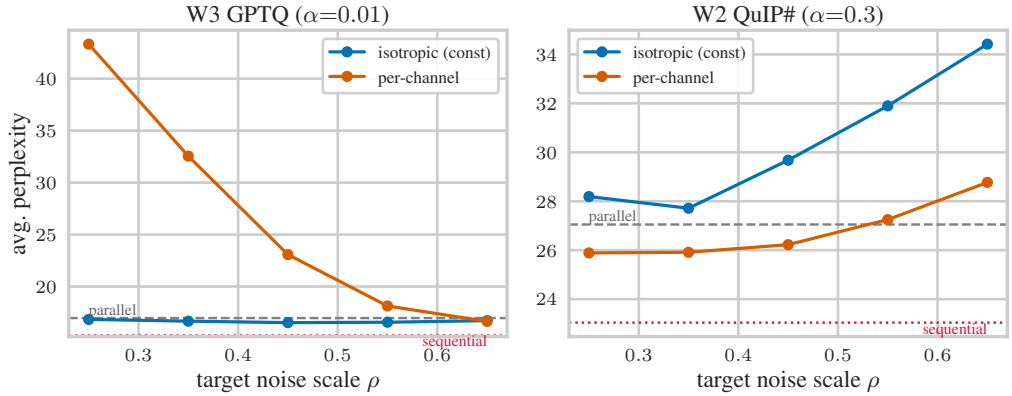


Figure 22: **Isotropic vs. per-channel Gaussian noise** (Llama-3.2-1B, δ=0.025, random Hadamard rotation, γ=0.05). Average perplexity against the target noise scale ρ; grey dashed marks the parallel baseline (0% gap) and red dotted the sequential baseline (100%). Left (W3 GPTQ, α=0.01): both isotropic and per-channel drop below the parallel baseline. Right (W2 QuIP#, α=0.3): only per-channel improves over the baseline; isotropic noise stays above it throughout.

A.6 GENERALIZATION ABLATION: HELD-OUT EVALUATION

Throughout the paper, every method quantizes models using a calibration set (128 WikiText-2 "train" sequences, Appendix B), which is separate from the test data we evaluate on. In the main paper, we report for every method (baselines and our denoising objective alike) the best setting of its hyperparameter sweep, measured by the test perplexity averaged over three evaluation sets: WikiText-2, C4, and FineWeb. This is not part of our method and only shows the best result each method can achieve, so that our method and the baselines are compared on equal terms. For transparency, we also report all results without this averaging step separately in Appendix A.3.

Nevertheless, we verify that reporting the best setting in this way does not change our conclusions. To this end, we choose each configuration using a validation set (we use perplexity on WikiText-2 "test" sequences for the validation), and report the avg. perplexity on C4 and FineWeb, which are separate from both the calibration and validation sets. This way, calibration and validation use disjoint splits of WikiText-2, and evaluation uses separate datasets. Notably, we find the qualitative conclusions remain the same.

In detail, Table 12 repeats Table 1 under this protocol. The parallel-to-sequential gap that our denoising closes is preserved across the board: at 3 bits it closes $34/21/32/12\%$ of the gap on Llama-3.2-1B/3B, Llama-3-8B, and Llama-2-13B (versus 17–35% in the main table), and at 2 bits it again closes a large fraction on the smaller models and matches or surpasses the sequential baseline on the larger ones. The qualitative conclusions are identical to Table 1.

Table 12: **Held-out generalization ablation.** Same objectives, models, and sweeps as Table 1, but every configuration is selected by WikiText-2 test perplexity alone, and the "Avg" column reports the average of the two held-out sets (C4 and FineWeb), which are unseen by the selection. The conclusions match Table 1, showing that the qualitative improvements persist under held-out evaluation.

		Quantization objective	Llama 1B			Llama 3B			Llama 8B			Llama 13B		
			Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑	Acc ↑	Avg ↓	Gap ↑
FP16			54.8	12.12	–	62.6	9.85	–	67.1	8.28	–	66.5	6.15	–
3-bit GPTQ	Par.	symmetric correction	50.0	19.00	0%	58.5	13.37	0%	63.5	11.10	0%	64.0	6.92	0%
		Denoising reg. (**ours**)	49.6	18.32	**34**%	58.9	13.22	**21**%	63.4	10.96	**32**%	64.6	6.90	**12**%
	Seq.	symmetric correction	49.8	18.30	35%	59.5	13.24	18%	63.5	10.94	36%	63.7	6.89	13%
		asymmetric correction	50.3	17.00	100%	58.4	12.65	100%	64.0	10.65	100%	64.4	6.72	100%
2-bit QuIP#	Par.	symmetric correction	44.3	31.01	0%	53.4	18.81	0%	58.8	15.80	0%	61.9	7.96	>100%
		Denoising reg. (**ours**)	45.1	29.27	**38**%	53.9	17.55	**95**%	59.1	14.50	**127**%	61.4	7.89	>**100**%
	Seq.	symmetric correction	43.8	30.01	22%	53.7	18.81	0%	58.0	15.92	−12%	60.9	8.23	<0%
		asymmetric correction	45.1	26.41	100%	51.3	17.48	100%	56.2	14.78	100%	60.0	8.02	<0%

A.7 W4-PREFIX PROMOTION PER-MODULE ABLATION ACROSS MODELS

We provide additional results for the two trade-offs shown in Figure 4 (middle and right). Figure 24 extends the quantization-time trade-off, where the first K blocks are quantized sequentially and the remaining ones in parallel, to all four models, and additionally shows the per-block signal-to-noise ratio (SNR) of the residual stream. The SNR measures how close the quantized model's residual stream stays to the full-precision one after each block, $\mathrm{SNR}=-20\log_{10}\left(\mathbb{E}\,\|\mathbf{h}_q-\mathbf{h}\|/\|\mathbf{h}\|\right)$ in dB, with $\mathbf{h}$ and $\mathbf{h}_q$ the full-precision and quantized residual streams (higher is better). Figure 23 extends the inference-time trade-off, where a prefix of modules is promoted to 4 bits, to Llama-3.2-1B/3B and Llama-3-8B. In both cases, the conclusions of the main text hold across all models.

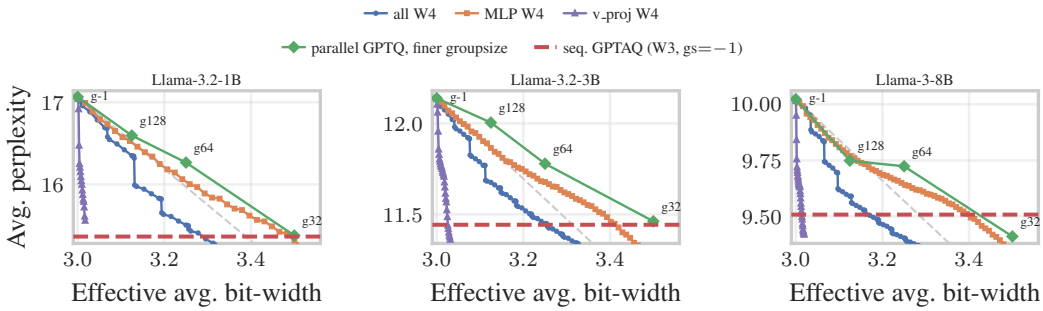


Figure 23: **Effective bit-width vs. perplexity** on Llama-3.2-1B/3B and Llama-3-8B (W3A16). Same plot as Figure 4 (right) for Llama-3.2-3B, extended to Llama-3.2-1B and Llama-3-8B.

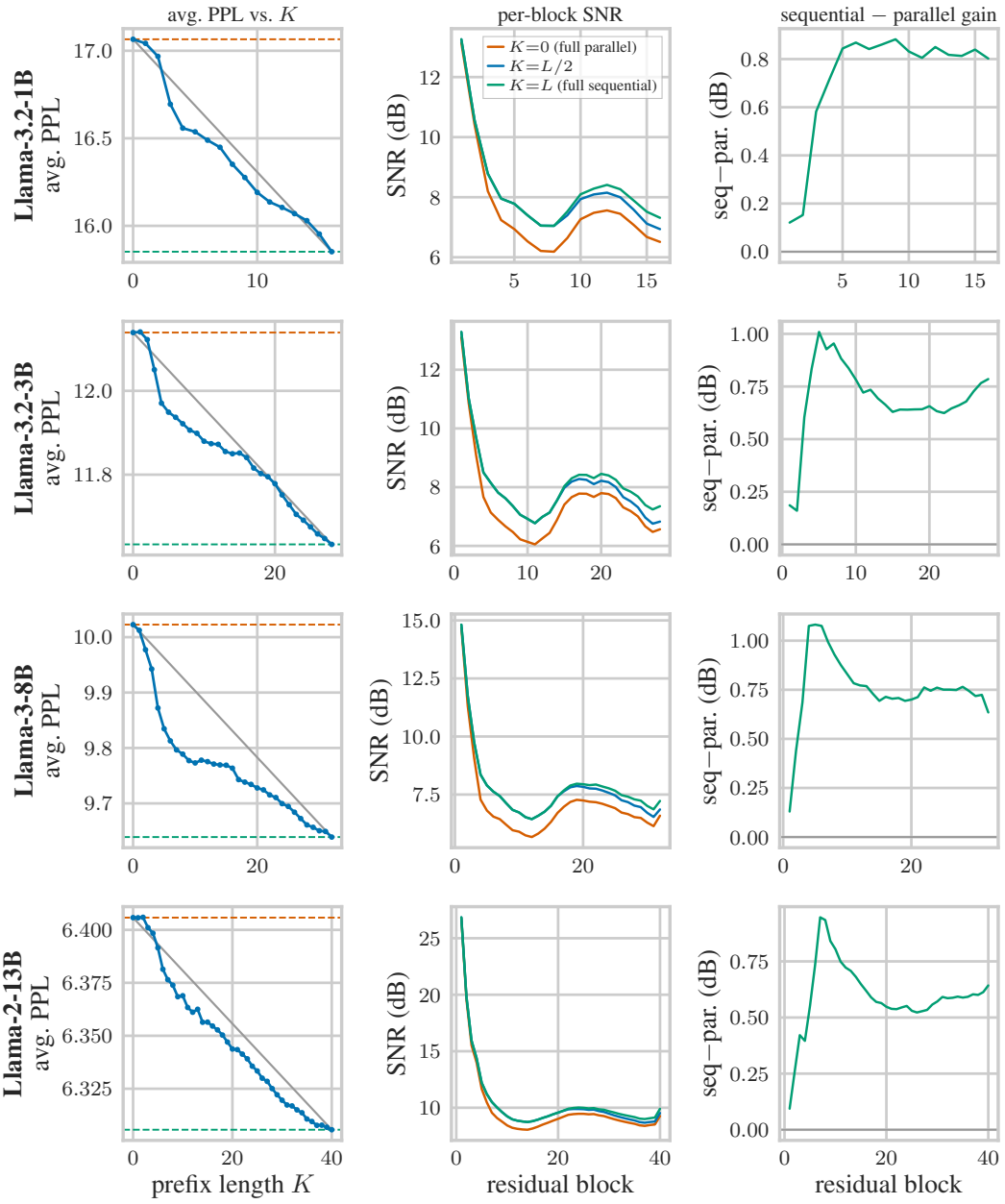


Figure 24: **Per-model sequential-prefix diagnostics** (W3A16), one row per model. The first K blocks are quantized sequentially (GPTAQ, β=0.1) and the remaining ones in parallel (GPTQ), at fixed hyperparameters. Left: avg. perplexity vs. K; the gray line interpolates linearly between K=0 and K=L, so a curve below it means the gains are front-loaded. Middle: per-block residual-stream SNR for $K \in \{0, L/2, L\}$. Right: per-block SNR gain of sequential over parallel quantization.

A.8 RUNTIME MEASUREMENTS AND PROJECTED MULTI-GPU TIME

Measurement details. All runtimes in this paper are wall-clock times measured on a single GPU. Our implementation runs parallel quantization on one GPU in two stages. First, it runs the calibration forward passes that collect the statistics of every layer (for our denoising, the clean pass and the noisy pass that accumulates the noise moments). Second, it computes each layer's pretransform and rounds its weights. In the parallel schedule, the second stage of a layer depends only on the statistics of that layer, so all layers could be processed at the same time. On a single GPU, however, we process them one after another, so the measured time is the sum over all layers. The sequential schedule instead alternates between the two stages: each layer is rounded only after all previous layers are quantized, since its statistics depend on their quantized outputs.

Single-GPU runtimes. Table 13 reports the mean per-run wall-clock time on a single NVIDIA A100 GPU, split into quantization (Quant) and perplexity evaluation (Eval), for each method row of the W3A16 tables. For our methods, Quant covers the full parallel pipeline (proxy quantization, moment estimation, the weight pretransform, and rounding). Learned-rotation runs load a precomputed rotation, and we report the time for learning rotations separately in Appendix B. For Llama-2-13B, which is close to the memory of a single 40 GB A100, we use numerically exact CPU offloading of the residual activation streams for configurations that do not otherwise fit (the learned-rotation sequential GPTAQ), which adds a small overhead to the reported time. Table 14 reports the same breakdown for the QuIP# W2A16 sweeps, on a single A100 for Llama-3.2-1B/3B and Llama-3-8B and on a single H200 for Llama-2-13B, so the 13B column is not directly comparable to the others.

Overhead of denoising depends on the solver and the noise model. Gaussian-only denoising needs no preliminary quantization pass, but its noisy forward pass, moment estimation, and pretransform add cost on a single GPU. How much this matters depends on the rounding solver. With GPTQ, rounding is cheap, so this overhead dominates: on Llama-3-8B, our denoising takes 29.4 minutes, compared to 11.6 for parallel GPTQ and 24.1 for sequential GPTAQ. With QuIP#, rounding is roughly an order of magnitude more expensive (e.g. $\approx$24 vs. $\approx$2.5 minutes on Llama-3.2-1B), so the same overhead is small in relative terms (e.g. 183.2 vs. 173.3 minutes for the sequential baseline on Llama-3-8B). The proxy-based variants (W3 and W3+Gauss rows) additionally require a preliminary parallel quantization pass, unless an existing quantized model is reused (Appendix A.3). In all cases, the per-layer processing remains independent.

Projected multi-GPU time. We do not run multi-GPU quantization experiments. Instead, Figure 2 and Figure 25 report a projection computed from measured single-GPU QuIP# times. For these figures, we re-measure the Llama-2-13B times on a single A100, so that all models are timed on the same GPU. We split the measured quantization time into the time of the forward passes, T_{fwd}, and the time of the rounding, T_{round}, and approximate the time of a parallel method on N GPUs as

$$T_{\mathrm{par}}(N) = T_{\mathrm{fwd}} + T_{\mathrm{round}}/N.$$

For the sequential schedule, we keep the single-GPU time, $T_{\mathrm{seq}}(N) = T_{\mathrm{seq}}(1)$, since its layers cannot be rounded concurrently. Note that the projection assumes that the layers are split evenly across GPUs and ignores communication overhead, so it is an upper bound on the achievable speedup. Since the rounding of different layers is independent given the collected statistics, distributing it across GPUs does not change the quantized model.

Figure 25 shows the projection for Llama-2-13B (W2A16 QuIP#). On a single GPU, all three methods take a comparable $\approx$290–316 minutes. The projected time of the parallel methods decreases roughly as $1/N$, so that on 8 GPUs our denoising takes $\approx$41 minutes compared to $\approx$306 for the sequential baseline (a $\approx 7.5\times$ reduction), at a small overhead over parallel QuIP#. Across the models in Figure 2, the projected reduction on 8 GPUs ranges from $\approx 7.1\times$ (Llama-3.2-1B) to $\approx 7.5\times$ (Llama-2-13B).

Forward-pass cost of the noise injection is negligible. Beyond the per-layer rounding, every parallel method runs one clean full-precision forward pass over the calibration set to build its moments. Our denoising objective needs one additional forward pass with the injected (per-channel Gaussian) noise to estimate the noise statistics. Table 15 reports both passes per model on a single A100 (n=128 calibration sequences, seqlen 2048). The noisy pass is only 1.25–$1.40\times$ the clean one (it additionally samples the noise and accumulates the noise and cross-covariance statistics). The overhead of our extra pass is a few seconds and negligible next to the per-layer rounding that dominates the quantization time in Table 13 and Table 14.

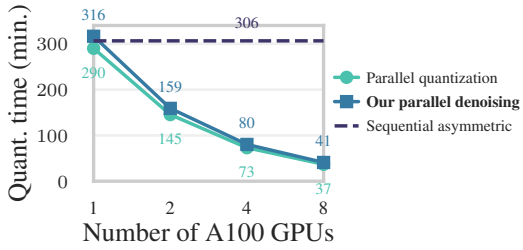


Figure 25: **Projected quantization time vs. number of GPUs on Llama-2-13B (W2A16 QuIP#).** Projection from measured single-GPU times (see text). Parallel quantization and our parallel denoising distribute the per-block rounding across GPUs, whereas the sequential asymmetric correction cannot be parallelized.

Table 13: **Per-row runtime on a single A100** (minutes, mean over each sweep), split into quantization (Quant) and perplexity evaluation (Eval), for Llama-3.2-1B/3B, Llama-3-8B, and Llama-2-13B at W3A16.

Method	Rot.	Llama 1B		Llama 3B		Llama 8B		Llama 13B	
		Quant	Eval	Quant	Eval	Quant	Eval	Quant	Eval
Parallel GPTQ	×	2.5	1.0	5.6	1.9	11.6	3.4	16.6	5.2
	✓	2.5	1.0	5.6	1.9	11.6	3.4	16.6	5.2
Gaussian *(ours)*	×	5.7	0.9	13.1	1.9	29.4	3.5	43.2	5.2
	✓	5.8	1.0	13.1	1.9	29.4	3.5	43.3	5.2
W3 proxy *(ours)*	×	8.2	1.0	19.0	1.9	42.4	3.5	61.2	5.2
	✓	8.2	1.0	19.0	1.9	42.5	3.5	61.0	5.2
W3 proxy + Gauss. *(ours)*	×	8.2	1.0	19.0	1.9	42.4	3.5	60.8	5.2
	✓	8.3	1.0	19.0	1.9	41.7	3.5	60.5	5.2
Sequential GPTQ	×	3.2	1.0	6.9	1.9	14.1	3.4	20.5	5.2
	✓	3.2	1.0	7.0	1.9	14.1	3.4	20.4	5.2
Sequential GPTAQ	×	5.2	1.0	11.4	1.9	24.1	3.5	34.7	5.2
	✓	5.2	1.0	11.5	1.9	24.1	3.5	36.9	5.2

Table 14: **Per-schedule runtime for the QuIP# W2A16 sweeps** (minutes, mean over each sweep). A100 for 1B/3B/8B, H200 for 13B. QuIP#'s rounding is more expensive than GPTQ's (table 13).

Schedule	Llama 1B		Llama 3B		Llama 8B		Llama 13B	
	Quant	Eval	Quant	Eval	Quant	Eval	Quant	Eval
Parallel symmetric	23.6	0.5	64.7	1.1	163.3	2.3	138.9	1.4
Denoising reg. (ours)	27.3	0.5	73.9	1.1	183.2	2.3	145.1	1.4
Sequential symmetric	23.8	0.5	65.3	1.1	164.1	2.3	140.1	1.4
Sequential asymmetric	25.4	0.5	69.2	1.1	173.3	2.3	142.1	1.4

Table 15: **Clean vs. noisy calibration forward-pass time on a single A100** (seconds, n=128 sequences, seqlen 2048). Clean is the full-precision forward pass every parallel method runs to build its moments. Noisy is the additional pass with injected per-channel Gaussian noise that our denoising objective needs. Overhead = Noisy − Clean is the extra cost of our second pass.

Model	Clean (s)	Noisy (s)	Overhead (s)	Noisy/Clean
Llama-3.2-1B	5.1	7.1	2.0	1.40
Llama-3.2-3B	12.3	16.6	4.3	1.35
Llama-3-8B	25.7	33.3	7.6	1.29
Llama-2-13B	42.9	53.5	10.6	1.25

B Detailed experimental setup and hyperparameters

B.1 Fixed setting

Unless a hyperparameter is explicitly swept, all experiments use the following fixed setting.

Weight grid and activations. Symmetric per-channel weights (groupsize -1) with MSE clipping; 16-bit activations and KV cache throughout (weight-only quantization).

Rounding solvers and their ridges. We use two rounding solvers, both regularized with the standard damping convention, i.e. a ridge proportional to the mean Hessian diagonal $\mathrm{tr}(\mathbf{H})/d$:

- **GPTQ** (W3/W4): OBS-style column rounding with Hessian damping $\texttt{percdamp} \cdot \mathrm{tr}(\mathbf{H})/d$, block size 128, no activation-order permutation. Default $\texttt{percdamp}{=}0.01$.
- **QuIP#** (W2): randomized-Hadamard incoherence with E_8-lattice block-LDLQ (block 8) and 10 refinement iterations. The trace-normalized LDL metric is ridged by σ_{reg}. We do not use recovery fine-tuning.

Our denoising objective (Theorem 1) reuses these same solvers as a drop-in rounder on the pretransformed weights $\tilde{\mathbf{W}}{=}\mathbf{WF}$ in the metric $\mathbf{G}$. Its proximity ridge α follows the same convention as $\texttt{percdamp}$ ($\alpha \cdot \mathrm{tr}(\cdot)/d$ of the metric being inverted; default $\alpha{=}0.01$), so the two are directly comparable. Because $\mathbf{G}$ already carries the $\alpha\mathbf{I}$ ridge, both solvers round in exactly $\mathbf{G}$ and add no further damping, avoiding a double ridge: the GPTQ sweep inverts $\mathbf{G}$ directly rather than re-damping with $\texttt{percdamp}$, and the QuIP# solver sets $\sigma_{\mathrm{reg}}{=}10^{-6}$ instead of its default $\texttt{percdamp}$.

Scheduling. Sequential methods quantize modules in the order q/k/v $\to$ o $\to$ up/gate $\to$ down. Parallel methods quantize every block from full-precision activations in a single pass.

Noise model. Injected Gaussian noise at scale $\gamma{=}0.05$: isotropic for GPTQ and per-channel for QuIP# (Appendix B.3); for an ablation of the noise model, see Appendix A.5.

Calibration. 128 WikiText-2 sequences of 2048 tokens; seed 0; no BOS token.

Evaluation. Perplexity on WikiText-2, C4, and FineWeb (2048-token sequences). Zero-shot accuracy on ARC-easy and ARC-challenge (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), OBQA (Mihaylov et al., 2018), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019), and WinoGrande (Sakaguchi et al., 2021), following the setup of (Liu et al., 2025).

Determinism and hardware. We use deterministic kernels and one single GPU per run, a single 40 GB NVIDIA A100 unless stated differently.

B.2 QuaRot rotations and learned SpinQuant rotations

The rotations described here apply to the GPTQ solver (W3/W4). The QuIP# solver (W2) carries its own randomized-Hadamard incoherence transform (Appendix B.1) and is therefore run without an external rotation, using neither the QuaRot random rotation nor the learned SpinQuant rotation.

Fixed random rotations. By default we apply the QuaRot randomized-Hadamard rotation (R_1, R_2, R_4) before weight quantization. We fix the random seed used to draw the Hadamard matrices, so that the same random rotations are used across all experiments; we ablate the effect of this randomness separately in Appendix B.4.

Learned rotation. We also replace the fixed random-Hadamard R_1 (and per-head R_2) by a learned SpinQuant-style rotation, optimized on the Stiefel manifold with Cayley SGD against a straight-through weight-fake-quant loss. We use 100 steps, learning rate 0.3 with momentum 0.9, a cosine schedule with 10% linear warm-up, MSE weight clipping (matching the deployment grid), and a training sequence length of 1024. Only R_1 and R_2 are learned; the online Hadamard rotations R_3 (post-RoPE keys) remain fixed random Hadamards. The learned rotation depends only on the model and its weight grid/bit-width (not on the downstream quantizer or the parallel/sequential schedule) so we compute it once per model and reuse it across all experiments. The 1B and 3B rotations are learned on a single A100. For Llama-3-8B and Llama-2-13B we learn it on a single NVIDIA H200 (141 GB) using gradient-checkpointing.

We learn rotations for each model and bitwidth once and use the resulting rotation matrices as the fixed rotation. For W3, learning the rotation took 36 minutes for Llama-3.2-1B (A100), 83 minutes for Llama-3.2-3B (A100), 73 minutes for Llama-3-8B (H200), and 122 minutes for Llama-2-13B (H200). For W4, it took 33 minutes for Llama-3.2-1B (A100), 77 minutes for Llama-3.2-3B (A100), 76 minutes for Llama-3-8B (H200), and 117 minutes for Llama-2-13B (H200).

B.3 Main tables hyperparameter sweeps

This section gives the hyperparameter sweeps behind the detailed per-model tables:

- Table 3, Table 4, Table 5, Table 6: Llama-3.2-1B/3B, Llama-3-8B, and Llama-2-13B at W3A16.
- Table 7: Llama-3.2-1B at W4A16.
- Table 8, Table 9, Table 10, Table 11: Llama-3.2-1B/3B, Llama-3-8B, Llama-2-13B at W2A16.

The rounding solver depends on the bit-width: the W3/W4 tables use the GPTQ (Frantar et al., 2023) solver (the same solver as the GPTQ baselines) and the W2 tables use the QuIP# (E_8) solver ((Tseng et al., 2024a)). For W3/W4 (GPTQ) we additionally repeat every configuration once with the fixed random rotation and once with the fixed SpinQuant-learned rotation.

Baselines (W3/W4, GPTQ).

- **Parallel / Sequential GPTQ (`percdamp` sweep).** GPTQ damping `percdamp` over 25 values, $\{0.001, 0.002, 0.005, 0.01, 0.02, 0.05\} \cup \{0.1, 0.15, 0.2, \ldots, 1.0\}$.
- **Sequential GPTAQ (β sweep).** Correction strength β over 25 values evenly spaced in $[0, 1]$, at `percdamp`=0.01 (β is called `alpha` in the GPTAQ code; see Appendix C).

Baselines (W2, QuIP#). The same schedules with the QuIP# solver, where `percdamp` plays the role of the LDL ridge σ_{reg}:

- **Parallel / Sequential QuIP# (`percdamp` sweep).** The same 25-value `percdamp` grid; the sequential run keeps the asymmetric correction off.
- **Sequential asymmetric QuIP# (β sweep).** Correction strength β over 25 values evenly spaced in $[0, 1]$ at `percdamp`=0.3 (the symmetric-sequential optimum).

Our denoising objective. We run the same 5×5 grid over the two knobs of our objective, $\rho \in \{0.25, 0.35, 0.45, 0.55, 0.65\}$ and $\delta \in \{0.00625, 0.0125, 0.025, 0.0375, 0.05\}$ with $\lambda = \rho + \delta$ (25 configurations per proxy). For W3/W4 we use the proximity ridge α=0.01 and the following parallel proxies for the upstream error $\boldsymbol{\eta}$:

- **Gaussian noise injection.** Injected, propagated isotropic Gaussian noise at γ=0.05 (see ablation in Appendix A.5).
- **Parallel-GPTQ (W3A16) upstream error.** Upstream weight-quantization error of a parallel-GPTQ W3A16 model, no injected noise.
- **Both.** The parallel-GPTQ W3A16 upstream error together with injected Gaussian noise at γ=0.05.

For W2 we use α=0.3 and the per-channel Gaussian proxy. We reuse the same fixed random seed for the injected Gaussian noise across all experiments (as for the fixed rotations), and ablate this randomness separately in Appendix B.4.

B.4 Randomness ablation

We ablate the randomness of each method on Llama-3.2-1B at W3A16. For the two baselines the only source of randomness is the fixed random Hadamard rotation, which we resample over 100 seeds. We additionally report a single run with the fixed learned rotation used in our main experiments. For our denoising objective the rotation is held fixed at that same learned rotation, so the only source of randomness is the injected Gaussian noise, which we resample over 100 seeds.

- **Parallel GPTQ.** At fixed `percdamp`=0.1, we run 100 independent runs with different random seeds for the random Hadamard rotation.
- **Sequential GPTAQ.** At fixed `percdamp`=0.01 and $\beta = 0.5$, we run 100 independent runs with different random seeds for the random Hadamard rotation. Note that GPTAQ is a special case of our denoising objective with $\boldsymbol{\eta}_{seq}$ the sequential upstream error and $\lambda = 1$.
- **Our denoising objective with Gaussian noise injection (random rotation, random noise).** At fixed `percdamp`=0.01, $\rho = 0.5$, $\delta = 0.05$, we run 100 independent runs with different random seeds for the injected Gaussian noise (γ=0.05; no quantized proxy, i.e. Gaussian noise only), *and* with different random seeds for the random Hadamard rotation.
- **Our denoising objective with Gaussian noise injection (learned rotation, random noise).** At fixed `percdamp`=0.01, $\rho = 0.5$, $\delta = 0.05$, and the fixed learned rotation, we run 100 independent runs with different random seeds for the injected Gaussian noise (γ=0.05; no quantized proxy, i.e. Gaussian noise only).

B.5 Weight outlier ablation

Section 6.2 analyzes how our pretransform $\tilde{\mathbf{W}} = \mathbf{WF}$ reshapes the weights, comparing the Frobenius norm before and after the correction (Figure 4, left). We use the fixed setting of Appendix B on Llama-3.2-1B, with $\rho = 0.5$, $\delta = 0.05$ (so $\lambda = \rho + \delta = 0.55$).

C GPTAQ AND QEP AS SPECIAL CASES

In this section, we relate our denoising objective to the two recent works that aim to correct for upstream quantization noise using sequential quantization: GPTAQ (Li et al., 2025) and QEP (Arai & Ichikawa, 2025). We show that both methods can be expressed as special cases of our denoising objective when passing the sequential upstream error, and we discuss the differences in how they motivate the correction strength β. We show the closed forms of both methods in Proposition 1, and then show that they are both special cases of our denoising objective in Corollary 1.

C.1 BACKGROUND ON GPTAQ AND QEP

GPTAQ (Li et al., 2025) is a sequential method that separates the two roles the input plays in the objective, $\min_{\hat{\boldsymbol{W}}} \mathbb{E}\|\boldsymbol{W}\boldsymbol{x}-\hat{\boldsymbol{W}}\tilde{\boldsymbol{x}}\|^2$: the target $\boldsymbol{W}\boldsymbol{x}$ uses the clean activations $\boldsymbol{x}$ of the original model, while the quantized weights $\hat{\boldsymbol{W}}$ act on the perturbed activations $\tilde{\boldsymbol{x}}$ propagated from the previously quantized layers. Each layer thus corrects for the upstream quantization noise it will actually receive, at the price of a sequential pass to measure $\tilde{\boldsymbol{x}}$.

QEP (Arai & Ichikawa, 2025) shares this goal but realizes the correction differently: rather than folding it into GPTQ's column sweep, it first applies a closed-form (ordinary least-squares) pretransform to the weights and then runs a standard GPTQ round on the transformed weights.

GPTAQ and QEP solve the same problem. Both keep GPTQ's damping ridge and additionally introduce a "correction strength" $\beta \in [0,1]$ scaling the asymmetric correction: GPTAQ's `alpha`, fixed at 0.25 in its released code, and QEP's `perccorr`, 0.5 by default.

Proposition 1 (Closed forms of GPTAQ and QEP)**.** *Let $\tilde{\mathbf{x}}$ be the layer input once all previous layers are quantized, $\boldsymbol{\eta} \triangleq \tilde{\mathbf{x}} - \mathbf{x}$ the sequential upstream error, assume $\mathbf{D} \triangleq \mathbb{E}[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^\top]$ is invertible, and let $\beta \in [0,1]$ be the correction strength. Both GPTAQ and QEP approximately solve the rounding problem $\min_{\hat{\mathbf{W}}\in\mathcal{G}_b} \operatorname{tr}\left[(\hat{\mathbf{W}}-\mathbf{W}^\star)\,\mathbf{D}\,(\hat{\mathbf{W}}-\mathbf{W}^\star)^\top\right]$ (Equation 1 with metric $\mathbf{D}$) with target*

$$\begin{aligned}\mathbf{W}^\star &= \mathbf{W} + \beta\,\mathbf{W}\,\mathbb{E}\big[(\mathbf{x}-\tilde{\mathbf{x}})\tilde{\mathbf{x}}^\top\big]\,\mathbf{D}^{-1}\\ &= (1-\beta)\,\mathbf{W} + \beta\,\mathbf{W}\,\mathbb{E}\big[\mathbf{x}\tilde{\mathbf{x}}^\top\big]\,\mathbf{D}^{-1}.\end{aligned}$$

GPTAQ realizes this correction inside GPTQ's column sweep, whereas QEP applies it as an explicit weight pretransform followed by a standard GPTQ round.

Proof. At $\beta = 1$, the target $\mathbf{W}^\star$ is the continuous minimizer of the asymmetric objective $\mathbb{E}\|\hat{\mathbf{W}}\tilde{\mathbf{x}} - \mathbf{W}\mathbf{x}\|^2$ (Arai & Ichikawa, 2025, Prop. 5.1). With $\boldsymbol{\delta} = \mathbf{X}-\tilde{\mathbf{X}}$ and $\hat{\mathbf{H}} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top$ stacking the calibration tokens as columns, QEP writes the scaled target as $\mathbf{W} + \beta\,\mathbf{W}\boldsymbol{\delta}\tilde{\mathbf{X}}^\top\hat{\mathbf{H}}^{-1}$ (Arai & Ichikawa, 2025, Eq. (6)) and rounds it in the metric $\hat{\mathbf{H}}$ (Arai & Ichikawa, 2025, Eq. (5)); the $1/n$ normalizations of $\mathbb{E}[(\mathbf{x}-\tilde{\mathbf{x}})\tilde{\mathbf{x}}^\top]$ and $\mathbf{D}$ cancel. GPTAQ scales the same correction term ($\Delta\mathbf{X}\tilde{\mathbf{X}}^\top$ in its notation) by β within its column sweep, with $\mathbf{D}$ as its Hessian (Li et al., 2025). The second form follows by linearity of the expectation and $\mathbb{E}[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^\top] = \mathbf{D}$:

$$\begin{aligned}\mathbf{W} + \beta\,\mathbf{W}\,\mathbb{E}\big[(\mathbf{x}-\tilde{\mathbf{x}})\tilde{\mathbf{x}}^\top\big]\,\mathbf{D}^{-1} &= \mathbf{W} + \beta\,\mathbf{W}\big(\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top] - \mathbb{E}[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^\top]\big)\mathbf{D}^{-1}\\ &= \mathbf{W} + \beta\,\mathbf{W}\big(\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top] - \mathbf{D}\big)\mathbf{D}^{-1}\\ &= (1-\beta)\,\mathbf{W} + \beta\,\mathbf{W}\,\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top]\,\mathbf{D}^{-1}.\end{aligned}$$

□

Here β linearly interpolates in weight space between the full-precision weights ($\beta = 0$, symmetric sequential GPTQ) and the fully corrected solution ($\beta = 1$). But the two works motivate this parameter β differently. QEP presents β as a tunable propagation mechanism that prevents overfitting the correction to the calibration set and reports that a fixed $\beta = 1/2$ works well in most cases. GPTAQ does not present its β in the paper at all; it appears only in the released code, justified in a GitHub issue as an empirical choice.[1] We argue that neither provides a principled motivation for this parameter, and find that the mechanism of their empirical correction is better understood as noise scaling in our framework, which we show below.

[1] `https://github.com/Intelligent-Computing-Lab-Panda/GPTAQ/issues/4`

C.2 GPTAQ AND QEP AS SPECIAL CASES OF ACTIVATION DENOISING

We show that both GPTAQ and QEP are special cases of our denoising objective (Equation 2) when passing the sequential upstream error $\boldsymbol{\eta} = \tilde{\mathbf{x}} - \mathbf{x}$, with λ=1 and α=0. The correction strength β of GPTAQ and QEP is exactly $1-\rho$, where ρ is the noise scaling parameter of our denoising objective.

Corollary 1 (GPTAQ and QEP as special cases of activation denoising)**.** *Let $\boldsymbol{\eta} = \tilde{\mathbf{x}} - \mathbf{x}$ be the sequential upstream error and assume $\mathbf{D} = \mathbb{E}[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^\top]$ is invertible. Then our denoising objective (Equation 2) with noise $\boldsymbol{\eta}$ at $\lambda = 1$ and $\alpha = 0$ satisfies, for every $\hat{\mathbf{W}}$,*

$$\mathbb{E}\big\|\hat{\mathbf{W}}\tilde{\mathbf{x}} - \mathbf{W}(\mathbf{x} + \rho\,\boldsymbol{\eta})\big\|^2 = \operatorname{tr}\big[(\hat{\mathbf{W}} - \mathbf{W}\mathbf{F})\,\mathbf{D}\,(\hat{\mathbf{W}} - \mathbf{W}\mathbf{F})^\top\big] + c,$$

where c does not depend on $\hat{\mathbf{W}}$, so both objectives have the same minimizers over $\mathcal{G}_b$, and

$$\begin{aligned}\mathbf{W}\mathbf{F} &= \mathbf{W} + (1-\rho)\,\mathbf{W}\,\mathbb{E}\big[(\mathbf{x} - \tilde{\mathbf{x}})\tilde{\mathbf{x}}^\top\big]\,\mathbf{D}^{-1} \\ &= \rho\,\mathbf{W} + (1-\rho)\,\mathbf{W}\,\mathbb{E}\big[\mathbf{x}\tilde{\mathbf{x}}^\top\big]\,\mathbf{D}^{-1}.\end{aligned}$$

Comparing with Proposition 1, this is exactly the GPTAQ and QEP rounding problem with correction strength $\beta = 1 - \rho$.

Proof. Setting $\lambda = 1$ and $\alpha = 0$ in the metric of Theorem 1 gives

$$\mathbf{G} = \mathbf{H} + \alpha\mathbf{I} + \lambda^2\mathbf{N} + \lambda(\mathbf{C} + \mathbf{C}^\top) = \mathbf{H} + \mathbf{C} + \mathbf{C}^\top + \mathbf{N}.$$

Expanding $\mathbf{D}$ with $\tilde{\mathbf{x}} = \mathbf{x} + \boldsymbol{\eta}$ and using linearity of the expectation,

$$\mathbf{D} = \mathbb{E}\big[(\mathbf{x}+\boldsymbol{\eta})(\mathbf{x}+\boldsymbol{\eta})^\top\big] = \mathbb{E}[\mathbf{x}\mathbf{x}^\top] + \mathbb{E}[\mathbf{x}\boldsymbol{\eta}^\top] + \mathbb{E}[\boldsymbol{\eta}\mathbf{x}^\top] + \mathbb{E}[\boldsymbol{\eta}\boldsymbol{\eta}^\top] = \mathbf{H} + \mathbf{C} + \mathbf{C}^\top + \mathbf{N},$$

so $\mathbf{G} = \mathbf{D}$.

Likewise, the filter becomes $\mathbf{F} = (\mathbf{H}+\alpha\mathbf{I}+\lambda\mathbf{C}+\rho\mathbf{C}^\top+\rho\lambda\mathbf{N})\mathbf{G}^{-1} = (\mathbf{H}+\mathbf{C}+\rho\,\mathbf{C}^\top+\rho\,\mathbf{N})\mathbf{D}^{-1}$. The stated identity is the completed square from the proof of Theorem 1, which only uses that $\mathbf{G}$ is symmetric and invertible; this holds for $\alpha = 0$ since $\mathbf{G} = \mathbf{D}$ is invertible by assumption.

It remains to write $\mathbf{W}\mathbf{F}$ in terms of the moments of $\tilde{\mathbf{x}}$. We group the numerator of $\mathbf{F}$ into the terms without and with ρ, $\mathbf{F} = \big(\mathbf{H}+\mathbf{C}+\rho\,(\mathbf{C}^\top+\mathbf{N})\big)\mathbf{D}^{-1}$. The first group is $\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top] = \mathbb{E}[\mathbf{x}(\mathbf{x}+\boldsymbol{\eta})^\top] = \mathbf{H} + \mathbf{C}$, and by the expansion of $\mathbf{D}$ above, the second group is $\mathbf{C}^\top + \mathbf{N} = \mathbf{D} - \mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top]$. Hence

$$\begin{aligned}\mathbf{W}\mathbf{F} &= \mathbf{W}\big(\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top] + \rho\,(\mathbf{D} - \mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top])\big)\mathbf{D}^{-1} \\ &= \rho\,\mathbf{W} + (1-\rho)\,\mathbf{W}\,\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top]\,\mathbf{D}^{-1},\end{aligned}$$

which is the second form. To obtain the first form, we split $\rho\,\mathbf{W} = \mathbf{W} - (1-\rho)\,\mathbf{W}$ and collect the terms with $(1-\rho)$:

$$\begin{aligned}\rho\,\mathbf{W} + (1-\rho)\,\mathbf{W}\,\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top]\,\mathbf{D}^{-1} &= \mathbf{W} - (1-\rho)\,\mathbf{W} + (1-\rho)\,\mathbf{W}\,\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top]\,\mathbf{D}^{-1} \\ &= \mathbf{W} + (1-\rho)\,\mathbf{W}\big(\mathbb{E}[\mathbf{x}\tilde{\mathbf{x}}^\top] - \mathbf{D}\big)\mathbf{D}^{-1} \\ &= \mathbf{W} + (1-\rho)\,\mathbf{W}\,\mathbb{E}\big[(\mathbf{x} - \tilde{\mathbf{x}})\tilde{\mathbf{x}}^\top\big]\,\mathbf{D}^{-1},\end{aligned}$$

where the second step uses $\mathbf{W} = \mathbf{W}\mathbf{D}\mathbf{D}^{-1}$ and the last step uses $\mathbf{D} = \mathbb{E}[\tilde{\mathbf{x}}\tilde{\mathbf{x}}^\top]$ and linearity of the expectation. □

The correction strength of GPTAQ and QEP therefore corresponds to our target noise scale: full correction $\beta = 1$ is full denoising (ρ=0), and $\beta = 0$ is symmetric sequential GPTQ ($\rho = \lambda = 1$). We believe our noise scaling perspective is more principled than the empirical motivations of GPTAQ and QEP, and we find that it generalizes to other upstream error proxies beyond the sequential upstream error.

D FULL PROOFS AND DERIVATIONS

In this section we restate the theorems of the main text and provide their full proofs.

D.1 SETUP AND NOTATION

Let $\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d}$ be the full-precision weights of a linear layer with input dimension d. Let $\mathbf{x} \in \mathbb{R}^d$ be the clean input activation and $\boldsymbol{\eta} \in \mathbb{R}^d$ the input noise, jointly distributed with finite second moments; all expectations are over this joint distribution (for empirical moments, the uniform distribution over the n calibration tokens). We abbreviate the noisy input and the target input of the denoising objective in Equation 2 as

$$\mathbf{u} \triangleq \mathbf{x} + \lambda\boldsymbol{\eta}, \qquad \mathbf{v} \triangleq \mathbf{x} + \rho\boldsymbol{\eta},$$

so that the objective reads

$$J(\hat{\mathbf{W}}) \triangleq \mathbb{E}\big\|\hat{\mathbf{W}}\mathbf{u} - \mathbf{W}\mathbf{v}\big\|^2 + \alpha\,\|\hat{\mathbf{W}} - \mathbf{W}\|_F^2. \tag{4}$$

With $\mathbf{H} = \mathbb{E}[\mathbf{x}\mathbf{x}^\top]$, $\mathbf{C} = \mathbb{E}[\mathbf{x}\boldsymbol{\eta}^\top]$ (hence $\mathbb{E}[\boldsymbol{\eta}\mathbf{x}^\top] = \mathbf{C}^\top$) and $\mathbf{N} = \mathbb{E}[\boldsymbol{\eta}\boldsymbol{\eta}^\top]$, expanding the outer products gives the second moments

$$\mathbb{E}[\mathbf{u}\mathbf{u}^\top] = \mathbf{H} + \lambda(\mathbf{C} + \mathbf{C}^\top) + \lambda^2\mathbf{N}, \tag{5}$$

$$\mathbb{E}[\mathbf{v}\mathbf{u}^\top] = \mathbf{H} + \lambda\mathbf{C} + \rho\mathbf{C}^\top + \rho\lambda\mathbf{N}, \tag{6}$$

$$\mathbb{E}[\mathbf{v}\mathbf{v}^\top] = \mathbf{H} + \rho(\mathbf{C} + \mathbf{C}^\top) + \rho^2\mathbf{N}. \tag{7}$$

In terms of these, the two matrices of Equation 3 are

$$\mathbf{G} = \mathbb{E}[\mathbf{u}\mathbf{u}^\top] + \alpha\mathbf{I}, \qquad \mathbf{B} \triangleq \mathbb{E}[\mathbf{v}\mathbf{u}^\top] + \alpha\mathbf{I} = \mathbf{H} + \alpha\mathbf{I} + \lambda\mathbf{C} + \rho\mathbf{C}^\top + \rho\lambda\mathbf{N}, \qquad \mathbf{F} = \mathbf{B}\mathbf{G}^{-1}. \tag{8}$$

Assumption. Throughout, we assume $\alpha > 0$, as in Theorem 1. Since $\mathbb{E}[\mathbf{u}\mathbf{u}^\top]$ is a second-moment matrix and therefore positive semidefinite, $\mathbf{G}$ is positive definite. In particular, $\mathbf{G}$ is symmetric and invertible, so $\mathbf{F}$ is well defined.

D.2 PROOF OF THEOREM 1

Theorem 1 (Pretransform-round equivalence)**.** *For $\alpha > 0$, the optimization problem in Equation 2 is equivalent to*

$$\min_{\hat{\mathbf{W}} \in \mathcal{G}_b} \operatorname{tr}\big[(\hat{\mathbf{W}} - \mathbf{W}\mathbf{F})\,\mathbf{G}\,(\hat{\mathbf{W}} - \mathbf{W}\mathbf{F})^\top\big],$$

where

$$\mathbf{G} \triangleq \mathbf{H} + \alpha\mathbf{I} + \lambda^2\mathbf{N} + \lambda(\mathbf{C} + \mathbf{C}^\top),$$
$$\mathbf{F} \triangleq \big(\mathbf{H} + \alpha\mathbf{I} + \lambda\mathbf{C} + \rho\mathbf{C}^\top + \rho\lambda\mathbf{N}\big)\,\mathbf{G}^{-1}.$$

Here $\mathbf{H} = \mathbb{E}[\mathbf{x}\mathbf{x}^\top]$, $\mathbf{C} = \mathbb{E}[\mathbf{x}\boldsymbol{\eta}^\top]$, and $\mathbf{N} = \mathbb{E}[\boldsymbol{\eta}\boldsymbol{\eta}^\top]$ are assumed finite, with expectations over the joint distribution of $(\mathbf{x}, \boldsymbol{\eta})$.

Proof. Using the notation of subsection D.1, let $\mathbf{D} \triangleq \mathbb{E}[\mathbf{v}\mathbf{v}^\top] + \alpha\mathbf{I}$. Expanding the reconstruction and proximity terms in Equation 4 and using the invariance of trace under transposition yields

$$J(\hat{\mathbf{W}}) = \operatorname{tr}\big[\hat{\mathbf{W}}\mathbf{G}\hat{\mathbf{W}}^\top\big] - 2\operatorname{tr}\big[\mathbf{W}\mathbf{B}\hat{\mathbf{W}}^\top\big] + \operatorname{tr}\big[\mathbf{W}\mathbf{D}\mathbf{W}^\top\big].$$

Set $\tilde{\mathbf{W}} = \mathbf{W}\mathbf{F}$. Since $\mathbf{F} = \mathbf{B}\mathbf{G}^{-1}$, we have $\tilde{\mathbf{W}}\mathbf{G} = \mathbf{W}\mathbf{B}$. Using this equality and the symmetry of $\mathbf{G}$ to complete the square gives

$$J(\hat{\mathbf{W}}) = \operatorname{tr}\big[(\hat{\mathbf{W}} - \tilde{\mathbf{W}})\mathbf{G}(\hat{\mathbf{W}} - \tilde{\mathbf{W}})^\top\big] + c,$$
$$c = \operatorname{tr}\big[\mathbf{W}\big(\mathbf{D} - \mathbf{F}\mathbf{G}\mathbf{F}^\top\big)\mathbf{W}^\top\big].$$

The constant c does not depend on $\hat{\mathbf{W}}$. This identity holds for every real candidate, hence for every candidate on the same grid $\mathcal{G}_b$. Adding a constant does not change the set of minimizers, proving the stated equivalence. □

D.3 Proof of Theorem 2

We consider a single layer with propagated upstream noise $\boldsymbol{\eta}^{\uparrow}$ and independent local injection $\boldsymbol{\xi}$, so the total noise is $\boldsymbol{\eta} = \boldsymbol{\eta}^{\uparrow} + \boldsymbol{\xi}$. The proof uses only $\mathbb{E}[\boldsymbol{\xi}] = \mathbf{0}$, $\mathbb{E}[\boldsymbol{\xi}\boldsymbol{\xi}^{\top}] = \boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1^2, \dots, \sigma_d^2)$, and independence of $\boldsymbol{\xi}$ from $(\mathbf{x}, \boldsymbol{\eta}^{\uparrow})$; Gaussianity is not needed.

Theorem 2 (Local noise injection acts as a magnitude-reducing ridge)**.** *Let $\boldsymbol{\eta} = \boldsymbol{\eta}^{\uparrow} + \boldsymbol{\xi}$, where $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ is independent of $(\mathbf{x}, \boldsymbol{\eta}^{\uparrow})$ and $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_1^2, \dots, \sigma_d^2)$. Up to a constant independent of the weights, Eq. 2 equals its reconstruction term with $\boldsymbol{\eta} = \boldsymbol{\eta}^{\uparrow}$ plus the modified ridge regularization*

$$\sum_c \left(\alpha + \lambda^2\sigma_c^2\right) \left\|\hat{\mathbf{W}}_{:,c} - s_c\, \mathbf{W}_{:,c}\right\|_2^2, \qquad s_c = \frac{\alpha + \rho\lambda\sigma_c^2}{\alpha + \lambda^2\sigma_c^2},$$

assuming $\alpha + \lambda^2\sigma_c^2 > 0$ for every c. For $\rho < \lambda$ and $\sigma_c > 0$, the penalty favors a down-scaled target $s_c\mathbf{W}_{:,c}$ for column c.

Proof. Step 1: the injection adds a column-wise penalty. With the injected noise, the reconstruction term of the objective is

$$\mathbb{E}_{\mathbf{x},\boldsymbol{\eta}^{\uparrow},\boldsymbol{\xi}} \left\|\hat{\mathbf{W}}\big(\mathbf{x} + \lambda(\boldsymbol{\eta}^{\uparrow} + \boldsymbol{\xi})\big) - \mathbf{W}\big(\mathbf{x} + \rho(\boldsymbol{\eta}^{\uparrow} + \boldsymbol{\xi})\big)\right\|_2^2 = \mathbb{E}\left\|\mathbf{a} + \mathbf{D}\boldsymbol{\xi}\right\|_2^2,$$

where $\mathbf{a} \triangleq \hat{\mathbf{W}}(\mathbf{x} + \lambda\boldsymbol{\eta}^{\uparrow}) - \mathbf{W}(\mathbf{x} + \rho\boldsymbol{\eta}^{\uparrow})$ collects everything that does not depend on $\boldsymbol{\xi}$, and $\mathbf{D} \triangleq \lambda\hat{\mathbf{W}} - \rho\mathbf{W}$. Expanding the square,

$$\mathbb{E}\left\|\mathbf{a} + \mathbf{D}\boldsymbol{\xi}\right\|_2^2 = \mathbb{E}\|\mathbf{a}\|_2^2 + 2\,\mathbb{E}\left[\mathbf{a}^{\top}\mathbf{D}\boldsymbol{\xi}\right] + \mathbb{E}\left\|\mathbf{D}\boldsymbol{\xi}\right\|_2^2.$$

The cross term vanishes: $\mathbf{a}$ is a function of $(\mathbf{x}, \boldsymbol{\eta}^{\uparrow})$ only, and $\boldsymbol{\xi}$ is independent of $(\mathbf{x}, \boldsymbol{\eta}^{\uparrow})$ with zero mean, so $\mathbb{E}[\mathbf{a}^{\top}\mathbf{D}\boldsymbol{\xi}] = \mathbb{E}[\mathbf{a}]^{\top}\mathbf{D}\,\mathbb{E}[\boldsymbol{\xi}] = 0$. For the last term, using $\mathbb{E}[\boldsymbol{\xi}\boldsymbol{\xi}^{\top}] = \boldsymbol{\Sigma}$ and writing $\mathbf{D}_{:,c}$ for column c of $\mathbf{D}$,

$$\mathbb{E}\left\|\mathbf{D}\boldsymbol{\xi}\right\|_2^2 = \mathrm{tr}\left[\mathbf{D}\,\boldsymbol{\Sigma}\,\mathbf{D}^{\top}\right] = \left\|\mathbf{D}\boldsymbol{\Sigma}^{1/2}\right\|_F^2 = \sum_c \sigma_c^2 \left\|\mathbf{D}_{:,c}\right\|_2^2 = \sum_c \sigma_c^2 \left\|\lambda\hat{\mathbf{W}}_{:,c} - \rho\mathbf{W}_{:,c}\right\|_2^2,$$

the last expression defining the injection penalty $\mathcal{R}(\hat{\mathbf{W}}) \triangleq \sum_c \sigma_c^2 \left\|\lambda\hat{\mathbf{W}}_{:,c} - \rho\mathbf{W}_{:,c}\right\|_2^2$. The first term $\mathbb{E}\|\mathbf{a}\|_2^2$ is exactly the reconstruction term of Equation 2 with $\boldsymbol{\eta} = \boldsymbol{\eta}^{\uparrow}$. Hence the objective with injection equals the objective without local injection plus $\mathcal{R}(\hat{\mathbf{W}})$.

Step 2: combining $\mathcal{R}$ with the proximity ridge. Because $\boldsymbol{\Sigma}$ is diagonal and the proximity ridge $\alpha\|\hat{\mathbf{W}} - \mathbf{W}\|_F^2 = \alpha\sum_c \|\hat{\mathbf{W}}_{:,c} - \mathbf{W}_{:,c}\|_2^2$ is column-separable, the combined penalty splits over columns:

$$\mathcal{R}(\hat{\mathbf{W}}) + \alpha\|\hat{\mathbf{W}} - \mathbf{W}\|_F^2 = \sum_c \Big(\sigma_c^2\, \|\lambda\hat{\mathbf{W}}_{:,c} - \rho\mathbf{W}_{:,c}\|_2^2 + \alpha\, \|\hat{\mathbf{W}}_{:,c} - \mathbf{W}_{:,c}\|_2^2\Big).$$

Fix a column c and let $\langle \mathbf{u}, \mathbf{v}\rangle$ denote the Euclidean inner product. Expanding the two squared norms of that column,

$$\begin{aligned}
&\sigma_c^2\, \|\lambda\hat{\mathbf{W}}_{:,c} - \rho\mathbf{W}_{:,c}\|_2^2 + \alpha\, \|\hat{\mathbf{W}}_{:,c} - \mathbf{W}_{:,c}\|_2^2 \\
&= (\alpha + \lambda^2\sigma_c^2)\, \|\hat{\mathbf{W}}_{:,c}\|_2^2 - 2(\alpha + \rho\lambda\sigma_c^2)\, \langle\hat{\mathbf{W}}_{:,c}, \mathbf{W}_{:,c}\rangle + (\alpha + \rho^2\sigma_c^2)\, \|\mathbf{W}_{:,c}\|_2^2.
\end{aligned}$$

Let $a_c \triangleq \alpha + \lambda^2\sigma_c^2$, positive whenever $\alpha > 0$ or $\lambda\sigma_c \neq 0$, and $s_c = (\alpha + \rho\lambda\sigma_c^2)/a_c$ as in the statement. Then

$$a_c\, \|\hat{\mathbf{W}}_{:,c} - s_c\mathbf{W}_{:,c}\|_2^2 = a_c\, \|\hat{\mathbf{W}}_{:,c}\|_2^2 - 2(\alpha + \rho\lambda\sigma_c^2)\, \langle\hat{\mathbf{W}}_{:,c}, \mathbf{W}_{:,c}\rangle + a_c\, s_c^2\, \|\mathbf{W}_{:,c}\|_2^2,$$

which matches the previous display in every term that depends on $\hat{\mathbf{W}}_{:,c}$. The difference is the column constant

$$\mathrm{const}_c = \Big(\alpha + \rho^2\sigma_c^2 - \frac{(\alpha + \rho\lambda\sigma_c^2)^2}{\alpha + \lambda^2\sigma_c^2}\Big)\|\mathbf{W}_{:,c}\|_2^2 = \frac{\alpha\,\sigma_c^2(\lambda - \rho)^2}{\alpha + \lambda^2\sigma_c^2}\, \|\mathbf{W}_{:,c}\|_2^2,$$

where the last equality uses $(\alpha+\rho^2\sigma_c^2)(\alpha+\lambda^2\sigma_c^2) - (\alpha+\rho\lambda\sigma_c^2)^2 = \alpha\sigma_c^2(\lambda^2 - 2\lambda\rho + \rho^2)$. Summing over c gives the stated decomposition with $\mathrm{const} = \sum_c \mathrm{const}_c$, which does not depend on $\hat{\mathbf{W}}$. □

Remark (down-scaling). For any channel with $\sigma_c > 0$ and $\lambda > 0$ we have $s_c < 1$ if and only if $\rho\lambda\sigma_c^2 < \lambda^2\sigma_c^2$, i.e. if and only if $\rho < \lambda$. In this regime the combined regularizer pulls each column $\hat{\mathbf{W}}_{:,c}$ toward the down-scaled $s_c\mathbf{W}_{:,c}$, with $s_c \to \rho/\lambda$ as $\alpha \to 0$ and $s_c \to 1$ as $\alpha \to \infty$. For $\rho = \lambda$ we obtain $s_c = 1$ and the regularizer is centered at the original weight column.