

Pivot Gradient Scaling: Reducing Dwell Time to Stabilize Extreme Low-Bit Quantization-aware Training

Vinnam Kim¹ Junhan Kim¹ Seungjae Shin¹ Jung Hyun Lee¹
Jaeseong You¹ An Chen¹ Jungwook Choi^{1,2*}

¹Qualcomm AI Research

²Hanyang University

{vinnkim, junhank, sseungja, jungh, jaeseong, anc}@qti.qualcomm.com

*Corresponding author: choij@hanyang.ac.kr

Abstract

Extreme low-bit quantization-aware training (QAT) is often unstable due to synchronized bin transitions, which can trigger catastrophic loss spikes. While prior works rely on scale-side controls to mitigate this fragility, we demonstrate that restricting pivot movement alone is insufficient. Instead, we identify a critical weight-side state governing low-bit QAT stability: **pivot-proximal residence**, where latent weights persistently linger near quantization pivots. We formalize this duration as **dwell time** and show that long dwell times accumulate a large, vulnerable population of weights prone to synchronized mass flips. To resolve this vulnerability, we propose **Pivot Gradient Scaling (PGS)** as a principled modification to the Straight-Through Estimator (STE) that leaves the forward pass intact and selectively amplifies latent-weight gradients within a localized pivot band. Crucially, we provide a mathematical framework proving that this sign-preserving amplification increases pivot-proximal mobility and systematically shortens expected dwell time under both SGD and Adam with a local random-walk approximation. Exhaustive empirical evaluations across Llama 3.2 and Qwen3 model families under full-training regimes up to 30 billion tokens demonstrate that PGS structurally stabilizes optimization, attenuates transition bursts, and delivers substantial downstream accuracy gains.

1 Introduction

Quantization is essential for deploying large language models (LLMs) efficiently (Dettmers et al., 2022; Frantar et al., 2022), but training in the extreme low-bit regime (e.g., 2-bit) remains highly unstable (Liu et al., 2024b; Lee et al., 2025a). Coarse quantization bins interact with optimizer updates, causing latent weights to oscillate destructively or flip rapidly between adjacent levels (Nagel et al., 2022; Lee et al., 2025b). While individual bin flips are well-documented, the collective training-time

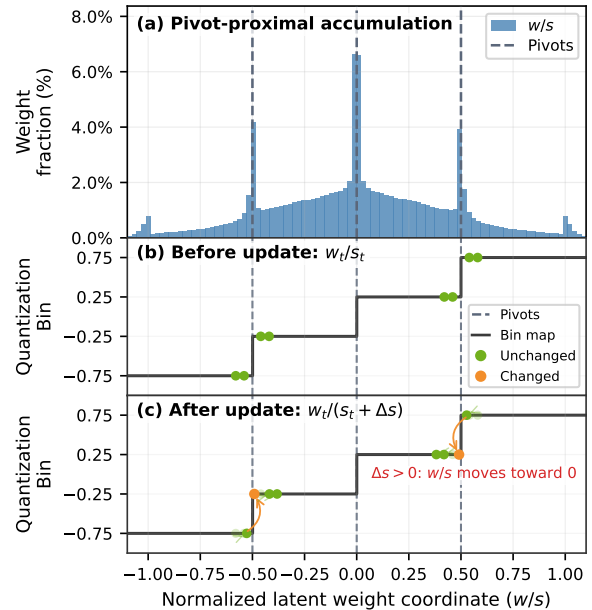


Figure 1: Pivot-proximal aggregation as an unstable state in low-bit QAT. Panel (a) shows the latent weight distribution of a Llama 3.2 1B Instruct model after 2-bit QAT, where weights concentrate near decision boundaries. Panels (b) and (c) illustrate how a small-scale update changes the normalized coordinates w/s relative to fixed decision boundaries. When many weights are already near the boundary, this coordinate shift can cause multiple weights to change their bin assignments in the same update.

dynamics that synchronize these flips and trigger catastrophic loss spikes remain underexplored.

Conventional methods attribute this fragility to quantization error or learned scale volatility, focusing heavily on scale-side controls (Esser et al., 2020; Bhalgat et al., 2020). For instance, Learned Step Size Quantization (LSQ) (Esser et al., 2020) rescales step-size gradients to balance the relative update magnitudes between scales and weights. However, limiting only scale-induced movement of the decision boundary (i.e., pivot) is fundamentally insufficient. As demonstrated by our baseline experiments with “Scale-Clip”—which

constraints realized scale updates using boundary-crossing margins—restricting boundary movement fails to prevent training collapse at high learning rates. This insufficiency motivates us to look beyond scale dynamics and examine the underlying population state of the latent weights themselves.

In this work, we identify a critical weight-side state governing low-bit QAT stability: **pivot-proximal residence**. As shown in Figure 1, latent weights frequently accumulate near the decision boundaries separating quantization bins. We formalize their persistence as **dwell time**—the number of consecutive steps a weight spends inside a fragile boundary band. Long dwell times pin a large fraction of weights near boundaries, rendering them simultaneously vulnerable to perturbations. Consequently, a minor update can trigger synchronized mass flips across widely spaced levels, surfacing as loss spikes.

To address this vulnerability, we introduce **Pivot Gradient Scaling (PGS)** as a principled modification to the standard Straight-Through Estimator (STE) (Bengio et al., 2013). PGS leaves the forward pass intact while selectively amplifying latent-weight gradients within a narrow band near pivots. Crucially, Theorem 3.1 provides a mathematical guarantee proving that this sign-preserving amplification increases pivot-proximal mobility and minimizes expected dwell time under both SGD and Adam. By anchoring our approach in this guarantee, we demonstrate that PGS is a principled framework that enhances local step variance and resolves weight-side fragility (Figure 2).

To validate this mechanism, we evaluate PGS across diverse model families—including Llama 3.2 and Qwen3—under full-training regimes up to 30 billion tokens. PGS consistently attenuates transition bursts, stabilizes optimization, and delivers substantial downstream accuracy gains. Furthermore, through meticulous locality controls (random, far-from-pivot, and norm-matched budgets), we identify pivot-local evacuation as the prime driver of stability. Finally, a case study reinterprets a prior surrogate gradient (Wang et al., 2025), revealing that its efficacy is implicitly driven by the principle of pivot-proximal gradient amplification we discovered. Our primary contributions are summarized as follows:

- We show that pivot-proximal residence and long dwell time drive low-bit QAT fragilities unresolved by scale-side controls.

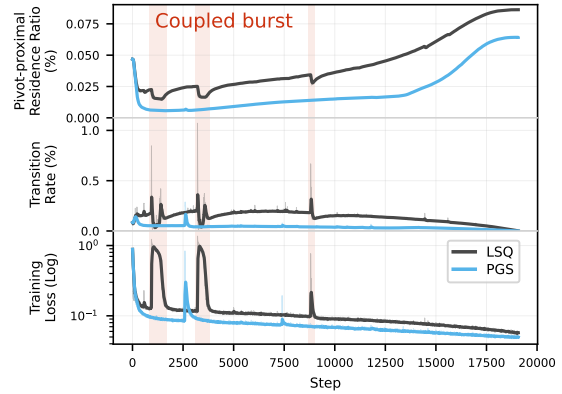


Figure 2: Residence–transition–loss dynamics in a 2-bit QAT run on Llama 3.2 1B Instruct. Compared with LSQ scale-side control baseline, PGS keeps near-boundary fraction lower, attenuates transition-rate bursts, and yields a smoother, lower loss trajectory.

- We introduce PGS as a foundational modification to STE and provide a theoretical formulation showing how localized amplification systematically minimizes expected dwell time.
- Through extensive long-run training runs (up to 30B tokens) across multiple Llama and Qwen scales, we demonstrate that PGS structurally stabilizes optimization and improves downstream performance.

2 Fragility of Pivot-Proximal Weights in Low-Bit QAT

Extreme low-bit QAT is often attributed to large quantization error, but coarse bins also make bin assignments highly sensitive to small perturbations. This section develops a pivot-centric geometric view in normalized weight coordinates and formalizes pivot margin, pivot-proximal residence, and transition rate as diagnostics of bin-assignment fragility. These definitions allow us to separate scale-induced pivot motion from weight-side residence.

2.1 Sensitivity to Bin Transitions

Table 1 probes bin-assignment fragility under small continuous perturbations of latent weights after QAT. 4-bit remains stable even at 1%, increasing only from 10.34 to 10.83. 3-bit lies between these regimes: it remains stable under smaller perturbations but collapses at 1%. By contrast, 2-bit models are extremely fragile: a perturbation strength of only 0.01% increases WikiText-2 PPL from 10.79 to a collapse-level value above 10^5 .

Table 1: WikiText-2 perplexity (PPL) under bin-assignment perturbations after Llama 3.2 3B Instruct QAT. Columns denote the perturbation magnitude (details in Section E).

Bitwidth	0%	0.01%	0.10%	1%
2-bit	10.79	3.4e5	3.4e5	3.6e5
3-bit	10.34	10.66	10.66	3.6e5
4-bit	10.34	10.44	10.44	10.83

These post-QAT perturbation results highlight the extreme fragility of bin assignments in low-bit models. During training, such perturbation effects arise continuously through optimizer updates and can affect many weights in close succession. This suggests that instability is governed by how many weights are simultaneously susceptible to bin reassignment. We therefore treat pivot-proximal residence as a key training-time state controlling this vulnerability.

2.2 Quantifying Pivot-Proximal Residence

To operationalize this training-time state, we measure residence in the scale-normalized coordinate w/s , since bin assignment is determined by this coordinate rather than by the raw latent weight w alone. In this space, both latent-weight updates and scale updates can move weights relative to fixed quantization pivots.

We denote the decision boundaries between adjacent bins as *pivots*. For the i -th weight $w_i(t)$ at step t , let $d_i(t)$ denote its distance to the nearest pivot, normalized by the distance from that pivot to the adjacent bin center. Thus, $d_i(t) = 0$ at a pivot and $d_i(t) = 1$ at the bin center, and δ specifies a relative pivot-band width. Then, the *pivot-proximal ratio* $\rho_\delta(t)$, defined as the fraction of weights that are close to a pivot, can be expressed as

$$\rho_\delta(t) = \frac{1}{N} \sum_{i=1}^N \mathbf{1}[d_i(t) \leq \delta], \quad (1)$$

where N is the total number of weights.

In addition, we track downstream bin dynamics using the transition rate (TR) (Lee et al., 2025b):

$$\text{TR}(t) = \frac{1}{N} \sum_i \mathbf{1}[z_i(t) \neq z_i(t-1)], \quad (2)$$

where $z_i(t) = Q_b(w_i(t)/s)$ and $Q_b : \mathbb{R} \rightarrow \mathcal{Z}_b$ is the quantization bin map. For 2-bit Stretched Elastic Quantization (SEQ) (Liu et al., 2026),

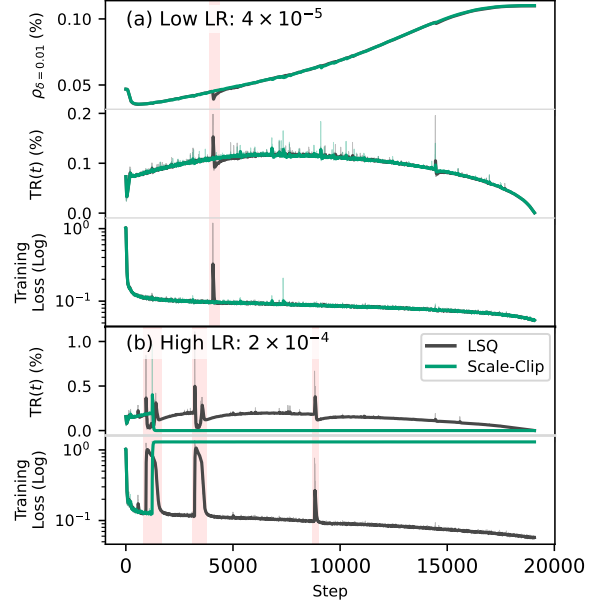


Figure 3: Scale-side control does not directly resolve pivot-proximal residence. Scale-Clip constrains realized scale updates using pivot-crossing margins. Although it can reduce transition bursts at low learning rate (Panel (a)), it leaves the pivot-proximal residence ratio close to the scale-gradient baseline and fails to prevent collapse at high learning rate (Panel (b)).

$\mathcal{Z}_2 = \{-0.75, -0.25, 0.25, 0.75\}$ with pivots at $\{-0.5, 0, 0.5\}$. Thus, $\rho_\delta(t)$ measures the simultaneous near-pivot population, whereas $\text{TR}(t)$ measures the resulting bin-reassignment activity. To capture this persistence over time, we introduce **dwelt time**—formalized in Section 3.2 as the number of consecutive steps a weight spends within a fragile pivot band—which directly drives cumulative training fragility.

2.3 Decoupling from Scale Dynamics

LSQ balances the relative update magnitudes of scales and weights with a uniform normalization factor to the scale gradient. Let $h_i(t) = \partial \mathcal{L} / \partial s_i(t)$ denote the scale gradient, then the normalized scale gradient $h'_i(t)$ is expressed as

$$h'_i(t) = \frac{1}{\sqrt{N_W Q_P}} h_i(t), \quad (3)$$

where N_W is the number of weights in the layer and $Q_P := \max z$ denotes the maximum positive quantization bin.

Although LSQ is commonly inherited by low-bit QAT methods (Liu et al., 2026; Lee et al., 2025a), the uniform rescaling in Equation (3) can be largely absorbed by adaptive optimizers such

as Adam (Kingma and Ba, 2014; Loshchilov and Hutter, 2017), as moment normalization cancels uniform rescaling factors after the statistics are adjusted (see Section C).

To avoid this effect, we consider Scale-Clip, a stronger scale-side baseline that constrains the scale update using current pivot-crossing margins (details in Section C.1). Unlike LSQ, this margin-aware step clipping is not normalized away by adaptive moments, making Scale-Clip a stronger scale-side intervention.

Despite this stronger intervention, Figure 3 shows that Scale-Clip leaves the pivot-proximal residence ratio close to LSQ at low learning rates. It can attenuate some transition bursts, but it does not directly reduce the near-pivot population; at high learning rates, training can still collapse. These results suggest that limiting scale-induced pivot motion alone is insufficient to control pivot-proximal residence. This motivates treating residence as a separate stabilization target.

3 Pivot Gradient Scaling

The analysis in the previous section suggests that controlling pivot motion alone is insufficient to reduce the pivot-proximal residence ratio. Since this ratio aggregates how long weights remain near pivots, reducing it requires shortening the dwell time. A natural way to achieve this is to increase the mobility of weights oscillating nearby pivots (Nagel et al., 2022), thereby facilitating faster escape from the pivot-proximal region. In this section, we introduce PGS, a principled weight-side modification that selectively accelerates the dynamics of pivot-proximal weights, and analyze its effect on dwell-time reduction.

3.1 Localized Gradient Amplification

Let $g_i(t) = \partial \mathcal{L} / \partial w_i(t)$ denote the base STE gradient of the latent weight w_i . To increase the mobility of weights near pivots while leaving other weights unaffected, the proposed PGS amplifies the gradients of pivot-proximal weights as follows:

$$g_i^{\text{PGS}}(t) = \begin{cases} \alpha \cdot g_i(t), & \text{if } d_i(t) \leq \delta, \\ g_i(t), & \text{otherwise.} \end{cases} \quad (4)$$

Here, $\alpha > 1$ controls the amplification strength, and $\delta > 0$ determines the pivot-band width.

This modification preserves the gradient direction while selectively increasing the step size of pivot-proximal weights. As a result, the task-loss

gradient continues to determine the update direction, ensuring that the optimization process remains aligned with the original objective. This property, in turn, enables controlled enhancement of near-pivot mobility without introducing additional directional biases or explicit repulsive forces.

3.2 Theoretical Dwell-Time Reduction

Before presenting empirical results, we develop a framework for understanding how PGS affects the behavior of pivot-proximal weights. In particular, we focus on how much PGS reduces the duration for which a pivot-proximal weight continuously remains near a pivot. To this end, we define the *dwell time* as follows.

Definition 3.1. Suppose that the weight w_i becomes pivot-proximal at time t , i.e., $d_i(t) \leq \delta$. The dwell time $\tau_i(t)$ is defined as

$$\tau_i(t) = \max \{k \mid d_i(t+u) \leq \delta \text{ for } 0 \leq u \leq k\}. \quad (5)$$

In other words, $\tau_i(t)$ measures the number of consecutive steps until the weight first exits the pivot-proximal region.

The following theorem explains the effect of PGS on pivot-proximal residence.

Theorem 3.1. Under the local random-walk approximation within the pivot-proximal region, suppose that the visit duration k_* satisfies $\beta_1^{k_*} \ll 1$ and $\beta_2^{k_*} \approx 1$. Then, the expected dwell time under SGD satisfies

$$\mathbb{E}[\tau_i^{\text{PGS,SGD}}(t)] \approx \frac{1}{\alpha^2} \mathbb{E}[\tau_i^{\text{SGD}}(t)]. \quad (6)$$

For Adam, the expected dwell time satisfies

$$\mathbb{E}[\tau_i^{\text{PGS,Adam}}(t)] \approx \frac{1}{\alpha^2} f_{\beta_2}(k_*) \mathbb{E}[\tau_i^{\text{Adam}}(t)], \quad (7)$$

where the correction factor is defined as

$$f_{\beta_2}(k) := \beta_2^k + (1 - \beta_2^k) \alpha^2,$$

and k_* denotes the effective duration of a pivot-band visit. Equivalently,

$$\frac{\mathbb{E}[\tau_i^{\text{PGS,Adam}}(t)]}{\mathbb{E}[\tau_i^{\text{Adam}}(t)]} \approx 1 - \beta_2^{k_*} \left(1 - \frac{1}{\alpha^2}\right). \quad (8)$$

Proof. See Appendix D. $\square$

Theorem 3.1 shows that PGS can reduce the dwell time by increasing the local step variance within the pivot-proximal region. Under SGD, this

increase leads to a quadratic reduction in the expected dwell time, i.e., by a factor of α^2 .

Under Adam, a similar reduction holds up to a correction factor $f_{\beta_2}(k_*)$, which arises from the slow adaptation of the second moment. In particular, with the standard settings where $\beta_1 = 0.9$ and $\beta_2 = 0.999$, the first moment more quickly adapts to the amplified gradient scale ($\beta_1^{k_*} \ll 1$), whereas the second moment retains significant memory of the pre-entry gradient scale ($\beta_2^{k_*} \approx 1$). These conditions typically hold for short pivot-band visit durations k_* . As a result, the denominator in Adam’s update remains close to its pre-entry value, while the numerator already reflects the amplified gradients. This transient mismatch prevents full normalization of the amplified gradients, allowing PGS to effectively increase the local step variance. In this regime, the correction factor $f_{\beta_2}(k_*)$ remains close to one, and PGS achieves a near- α^2 reduction in dwell time.

In Section 4, we empirically validate this theoretical prediction by showing that PGS consistently reduces dwell time, which in turn reduces transition rates and eventually leads to more stable and monotonic loss reduction.

4 Experiments

We evaluate three mechanistic predictions: (1) whether PGS shortens dwell time and lowers the pivot-proximal residence ratio; (2) whether reduced residence attenuates transition-rate bursts and loss spikes; and (3) whether smoother dynamics lead to downstream quality gains.

4.1 Experimental Setup

Model and quantization setup. We study extreme weight-only QAT across Llama3 (Grattafiori et al., 2024) and Qwen3 (Yang et al., 2025) model families, using 2-bit SEQ. Following the recent LLM QAT recipe (Lee et al., 2025a; Du et al., 2024), we train with self-distillation loss on DCLM-Edu (Allal et al., 2025) under billion-token budgets. We use LSQ as the primary baseline for the scale-side control, as it is commonly adopted in prior extreme low-bit QAT studies (Liu et al., 2026). We implement PGS by fusing the gradient amplification into the existing SEQ (details in section A). This kernel fusion incurs no measurable additional memory and latency overheads (details in section B).

Table 2: Trajectory-level residence statistics at diagnostic width $\delta = 0.01$. *In-band logs* summarizes the fraction of sampled weight states inside pivot bands, corresponding to the trajectory-level average of $\rho_{\delta=0.01}(t)$. *Entries* measures how often weights enter pivot bands, reported as contiguous visits per tracked trajectory per 1k steps. *Med.* τ_i and *P90* τ_i denote the median and 90th-percentile dwell time in steps. Bold indicates the best non-scale-side method within each block.

Setting	Method	In-band % $\downarrow$	Entry rate $\downarrow$	Med. $\tau_i \downarrow$	P90 $\tau_i \downarrow$
Llama3.2 1B Instruct, 2-bit, LR = 2e-4	LSQ	3.21	1.088	9	46
	PGS (near-pivot)	0.96	0.355	3	27
	PGS (random)	2.12	0.742	7	40
	PGS (far-from-pivot)	3.60	1.473	8	38
	PGS (norm-matched)	3.09	1.195	8	39
Llama3.2 1B Instruct, 2-bit, LR = 4e-5	LSQ	5.81	0.459	27	185
	PGS	2.82	0.209	19	166
Qwen3-0.6B, 2-bit, LR = 2e-4	LSQ	3.93	1.059	10	58
	PGS	1.27	0.271	5	54

Evaluation tasks. We report WikiText-2 perplexity (Merity et al., 2016), MMLU (Hendrycks et al., 2020), IFEval (Zhou et al., 2023), and common-sense reasoning benchmarks (CSR). CSR averages ARC-e, ARC-c (Clark et al., 2018), PIQA (Bisk et al., 2020), HellaSwag (Zellers et al., 2019), OpenBookQA (Mihaylov et al., 2018), and Winogrande (Sakaguchi et al., 2019).

Pivot-Band Scheduling. For our QAT runs, we anneal the pivot-band width. For training step $t \in [0, T]$, we set

$$\delta_t = \delta_0 \frac{1 + \cos(\pi t/T)}{2}, \quad (9)$$

where $\delta_0 \in [0, 1]$ is the initial relative pivot-band width. This schedule matches the cosine learning-rate decay used in our QAT runs, keeping the probe broader early and more localized near the end; fixed versus annealed bands are ablated in Section G. If not mentioned specifically, we setup PGS parameters as $\alpha = 5$, $\delta_0 = 0.05$.

4.2 Trajectory-Level Dwell Time

As analyzed in Section 3.2, PGS increases the optimizer-step mobility of pivot-proximal weights. To verify it at the trajectory level, we track a sampled subset of latent weights throughout QAT and measure their contiguous visits inside pivot bands, i.e., their pivot-proximal dwell time.

Table 2 compares PGS with the scale-side baseline (LSQ) across three 2-bit QAT settings using group size 128 and a 5B-token training budget: a high-LR Llama 3.2 1B Instruct run, a low-LR Llama 3.2 1B Instruct run, and a Qwen3-0.6B run. In the first high-LR diagnostic block, we additionally ablate where PGS gradient amplification is

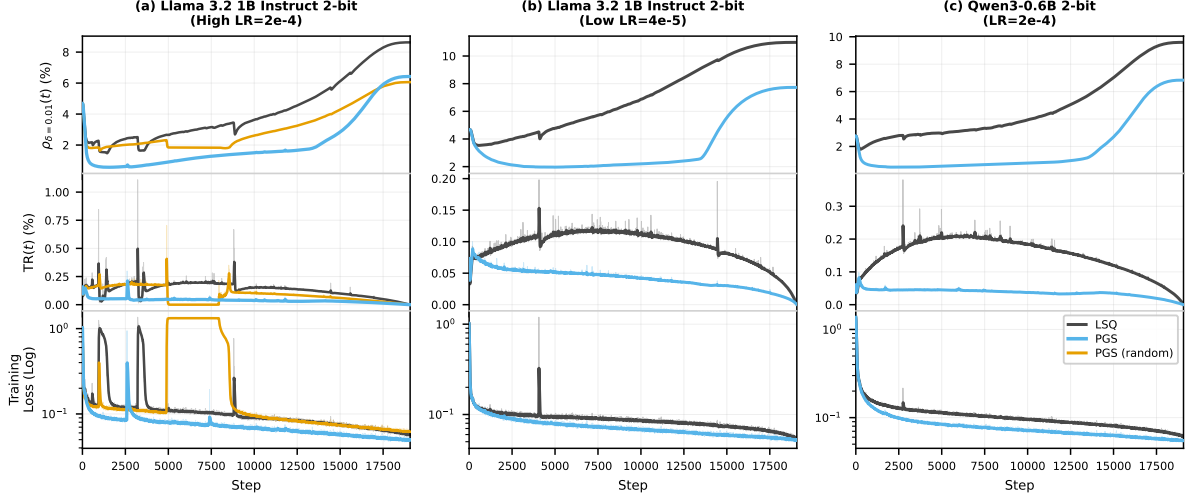


Figure 4: Lower pivot-proximal residence is accompanied by smoother transition–loss dynamics. Panel (a) compares PGS with PGS (random) to isolate the role of locality in gradient amplification. Panels (b–c) show that the same residence–transition–loss coupling persists across more stable low-LR regimes.

Table 3: Normalized weight step $\Delta_i(t) := \frac{w_i(t)}{s_i(t)} - \frac{w_i(t-1)}{s_i(t-1)}$ statistics across τ_i intervals from the Qwen3-0.6B (2-bit SEQ, LR=2e-4) run. $\text{sgn}(\Delta)$ denotes the average sign-flip rate of $\Delta_i(t)$, and $|\mu_\Delta/\sigma_\Delta|$ denotes the absolute mean of $\Delta_i(t)$ normalized by its standard deviation.

τ_i	[5, 20)	[20, 50)	[50, 100)	[100, 500)
$\text{sgn}(\Delta)$	0.21	0.29	0.32	0.34
$ \mu_\Delta/\sigma_\Delta $	0.85	0.27	0.15	0.08

applied using three locality controls: random, far-from-pivot, and norm-matched variants (details in Section H).

The first block tests this mechanism under a canonical high-LR diagnostic setting through locality ablations. Among the amplification variants in the first block, only PGS (near-pivot) substantially reduces the duration of pivot-proximal visits, as measured by Med. τ_i and P90 τ_i , relative to both the baseline and the other controls. Notably, the effect is not limited to shorter visits: weights also enter pivot bands less often, as reflected in the lower entry rate. This indicates that the observed reduction is not a generic consequence of larger gradients, but instead arises from increased mobility at pivot-proximal locations.

In the remaining blocks, we use PGS to denote the canonical near-pivot variant, and observe the same residence trends in more stable lower-LR regimes. Together, these results demonstrate that dwell-time reduction is governed by mobility at pivot-proximal locations, rather than by amplifica-

tion magnitude alone.

Table 3 provides empirical support for the local random-walk behavior assumed in Theorem 3.1. A sign-flip rate approaching 0.5 indicates increasingly random directional changes, while $|\mu_\Delta/\sigma_\Delta|$ approaching zero indicates a zero mean drift. As τ_i increases, the sign-flip rate increases from 0.21 to 0.34, while $|\mu_\Delta/\sigma_\Delta|$ decreases from 0.85 to 0.08. Although the random-walk assumption is only approximate for short visits due to the momentum in AdamW, its validity becomes more plausible for longer visits.

4.3 Optimization and Transition Dynamics

We next ask whether the residence reduction observed in the same three 2-bit QAT settings as Table 2 propagates to training dynamics: high-LR Llama 3.2 1B Instruct, low-LR Llama 3.2 1B Instruct, and Qwen3-0.6B runs.

Figure 4(a) first tests this link under the canonical high-LR diagnostic setting by comparing pivot-proximal amplification with random amplification¹. PGS (random) behaves similarly to LSQ early in training. In contrast, PGS rapidly evacuates weights from the pivot band, causing $\rho_{\delta=0.01}(t)$ to drop early and remain substantially lower for most of training. This sustained reduction yields smoother transition–loss dynamics. Thus, the stabilizing effect does not arise from amplification magnitude alone, but from pivot-localized mobility that reduces residence.

¹We show only PGS (random) for clarity; see Section H for the other locality controls.

Table 4: PGS amplification-strength and band-width sweeps on the canonical high-LR Llama 3.2 1B Instruct 2-bit setting (LR=2e-4). $\rho_{\delta=0.01}(t)$ AUC and $\text{TR}(t)$ AUC denote trapezoidal integrals over training step, reported in 10^3 step-%. PPL means WikiText-2 test perplexity. The $\alpha = 3, \delta_0 = 0.05$ and $\alpha = 5, \delta_0 = 0.05$ runs are used in the following experiments (*).

α	δ_0	Med. $\tau_i \downarrow$	P90 $\tau_i \downarrow$	$\rho_{\delta=0.01}(t)$ AUC $\downarrow$	$\text{TR}(t)$ AUC $\downarrow$	PPL $\downarrow$
2	0.05	6	33	54.05	1.44	14.04
3*	0.05	5	29	46.48	1.07	13.90
5*	0.05	3	27	40.15	0.72	13.96
7	0.05	3	29	37.31	0.58	14.03
10	0.05	3	30	34.47	0.46	14.04
5	0.05*	3	27	40.15	0.72	13.96
5	0.15	5	31	23.66	0.80	13.95
5	0.25	6	32	25.75	1.11	14.05
5	0.35	7	34	31.62	1.47	14.09

Figure 4(b-c) shows that the same residence-transition-loss relationship persists in lower-LR and cross-model settings. Even when visible loss spikes disappear, reduced transition fluctuations remain associated with better optimization, consistent with the view that lower residence reduces the population of weights simultaneously vulnerable to bin reassignment.

Amplification-strength and band-width sweeps.

We vary only α or only δ_0 respectively to test whether residence reduction changes systematically with the strength and support width of the intervention. Here, $\rho_{\delta=0.01}(t)$ area under the curve (AUC) summarizes the cumulative pivot-proximal ratio $\rho_{\delta=0.01}(t)$ over training, while $\text{TR}(t)$ AUC summarizes the cumulative transition rate over training. These AUCs thus complement Med./P90 τ_i , which measure dwell time.

Table 4 shows a strength-dependent trend in the α sweep: larger α consistently reduces both cumulative quantities, while visit-level dwell time already saturates at $\alpha = 5, \delta_0 = 0.05$. The δ_0 sweep reveals a trade-off between cumulative occupancy and visit-level sharpness: a moderately wider band lowers $\rho_{\delta=0.01}(t)$ AUC, whereas the narrow canonical band yields the shortest Med./P90 τ_i and the lowest $\text{TR}(t)$ AUC. When δ_0 becomes too wide, $\text{TR}(t)$ AUC increases, suggesting that the benefit does not arise merely from amplifying more weights indiscriminately.

Based on these observations, we select $\alpha = 5, \delta_0 = 0.05$ for Table 5. Its second-best PPL further indicates the mechanism-based selection using dwell-time statistics transfers well to downstream model quality. For the longer QAT runs in table 6, on the other hand, we use $\alpha = 3, \delta_0 = 0.05$

Table 5: Final quality after QAT. Italic: FP16 reference (upper bound, not competing). Bold: best result among quantized methods per block. Avg. = mean(CSR, MMLU, IFEval).

Setting	Method	PPL $\downarrow$	CSR $\uparrow$	MMLU $\uparrow$	IFEval $\uparrow$	Avg. $\uparrow$
Llama 3.2 1B Instruct, 2-bit, LR = 2e-4	<i>FP16</i>	<i>12.15</i>	<i>55.1</i>	<i>45.5</i>	<i>59.7</i>	<i>53.4</i>
	LSQ	14.42	52.4	31.6	31.5	38.5
	PGS	13.96	53.4	39.8	35.2	42.8
Llama 3.2 1B Instruct, 2-bit, LR = 4e-5	<i>FP16</i>	<i>12.15</i>	<i>55.1</i>	<i>45.5</i>	<i>59.7</i>	<i>53.4</i>
	LSQ	14.21	53.3	37.1	34.9	41.8
	PGS	14.10	52.9	39.6	38.8	43.8
Qwen3-0.6B, 2-bit, LR = 2e-4	<i>FP16</i>	<i>19.16</i>	<i>48.8</i>	<i>47.2</i>	<i>65.5</i>	<i>53.8</i>
	LSQ	20.20	46.7	42.7	30.4	39.9
	PGS	19.89	47.0	43.9	34.1	41.7

Table 6: 30B-token 2-bit QAT results across model families. Italic: FP16 reference (upper bound, not competing). Bold: best result among quantized methods per block. PGS uses $\alpha = 3.0$ and $\delta_0 = 0.05$.

Setting	Method	PPL $\downarrow$	CSR $\uparrow$	MMLU $\uparrow$	IFEval $\uparrow$	Avg. $\uparrow$
Llama 3.2 3B Instruct, 2-bit	<i>FP16</i>	<i>10.48</i>	<i>65.44</i>	<i>59.92</i>	<i>74.98</i>	<i>66.78</i>
	LSQ	11.49	64.28	55.88	56.64	58.93
	PGS	11.04	65.12	56.54	62.12	61.26
Qwen3-1.7B, 2-bit	<i>FP16</i>	<i>15.67</i>	<i>61.43</i>	<i>60.12</i>	<i>71.76</i>	<i>64.44</i>
	LSQ	17.52	60.14	55.41	39.42	51.66
	PGS	17.23	60.51	55.62	44.32	53.48
Qwen3-4B-Instruct-2507, 2-bit	<i>FP16</i>	<i>9.40</i>	<i>70.52</i>	<i>72.70</i>	<i>86.99</i>	<i>76.74</i>
	LSQ	12.48	67.63	67.19	74.52	69.78
	PGS	11.65	69.27	68.03	76.06	71.12

showing the best PPL.

4.4 Downstream Performance

We next evaluate whether the stabilization observed above translates into downstream quality.

Table 5 reports final quality for the same three 2-bit settings analyzed above. Across all three settings, PGS consistently improves perplexity over LSQ, with reductions of 0.46, 0.11, and 0.31 on high-LR Llama 3.2 1B Instruct, low-LR Llama 3.2 1B Instruct, and Qwen3-0.6B, respectively. The downstream results follow the same trend, with average-score gains of +4.3, +2.0, and +1.8 points.

Table 6 further evaluates 30B-token 2-bit per-channel QAT on larger ($> 1B$) Llama and Qwen models. In this longer-run setting, the gains remain consistent: PGS reduces PPL by 0.45, 0.29, and 0.83, and improves the average downstream score by +2.33, +1.82, and +1.34 points across the three models. Notably, IFEval improves by +5.48, +4.90, and +1.54 points despite the absence of instruction-tuning data in the QAT corpus.

These gains show that the residence-related stabilization is not merely diagnostic, but remains relevant in practical, longer-run low-bit QAT.

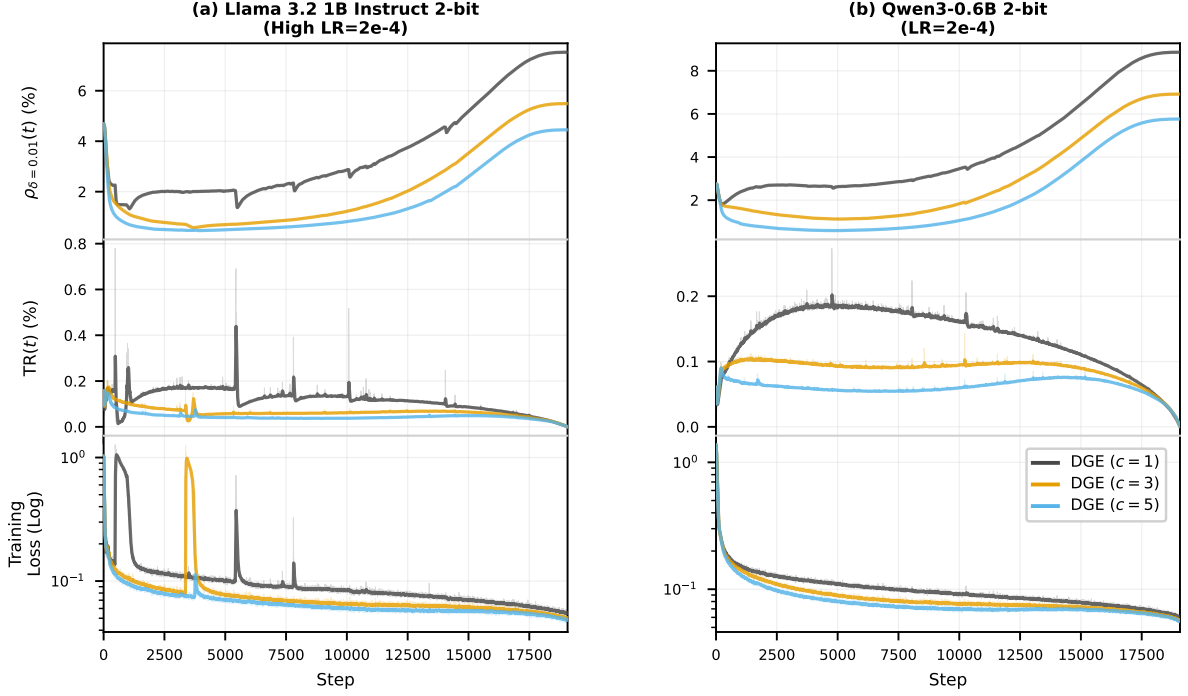


Figure 5: Ablating the amplification component of DGE on the two 2-bit settings: Llama 3.2 1B Instruct (Panel (a)) and Qwen3-0.6B (Panel (b)). We compare the default DGE ($c = 3$), a clamped variant DGE ($c = 1$) that preserves the DGE surrogate shape but removes the $\gamma_i > 1$ amplification regime, and DGE ($c = 5$) that increases the cap on pivot-proximal amplification.

Table 7: Final quality after QAT across the DGE amplification clamp $c \in \{1, 3, 5\}$.

Model	Method	PPL ↓	CSR ↑	MMLU ↑	IFEval ↑	Avg. ↑
Llama 3.2 1B Instruct, 2-bit, LR=2e-4	<i>FP16</i>	<i>12.15</i>	<i>55.1</i>	<i>45.5</i>	<i>59.7</i>	<i>53.4</i>
	DGE ($c = 1$)	14.26	52.6	33.9	31.0	39.2
	DGE ($c = 3$)	13.98	53.2	39.5	33.6	42.1
	DGE ($c = 5$)	13.89	53.2	39.6	38.1	43.6
Qwen3-0.6B, 2-bit, LR=2e-4	<i>FP16</i>	<i>19.16</i>	<i>48.8</i>	<i>47.2</i>	<i>65.5</i>	<i>53.8</i>
	DGE ($c = 1$)	20.57	46.6	42.0	31.6	40.1
	DGE ($c = 3$)	20.07	47.1	44.2	34.7	42.0
	DGE ($c = 5$)	20.59	46.9	43.8	33.5	41.4

4.5 Reinterpreting Surrogate Gradients

So far we have studied near-boundary amplification through PGS, a minimal amplifier we designed for this purpose. A stronger test of the residence account is whether the same mechanism already appears in a surrogate gradient developed for unrelated reasons. If it does, residence reduction is not an artifact of our construction, but a property that existing backward paths exploit without naming it.

We examine DGE (Wang et al., 2025), which replaces the STE pass-through with a smooth-step differentiable estimator for FP4 LLM training. DGE is motivated by differentiability of the backward pass and does not discuss decision-boundary gradients, residence, or dwell time. Viewed through our boundary-centric lens, however, its backward

multiplier

$$\gamma_i^{\text{DGE}}(t) = \min\left(\frac{d_i(t)^{1/k-1}}{k2^{b-1}}, c\right) \quad (10)$$

(with $k = 5$ by default and $d_i(t)$ numerically floored in implementation) grows as the boundary distance $d_i(t) \rightarrow 0$ and is capped by the clamp c , so DGE inherently amplifies near-boundary gradients up to c while attenuating them away from the boundary ($\gamma_i^{\text{DGE}} \leq 1$); see Section I. The clamp c therefore acts as a near-boundary amplification ceiling, the DGE analogue of PGS’s strength α : $c > 1$ permits near-boundary amplification, whereas $c = 1$ removes it and leaves only the soft-step shaping away from the boundary.

We use this structure to isolate which part of DGE produces stabilization. We compare DGE ($c = 3$), its default, with a clamped variant DGE ($c = 1$) that preserves the surrogate shape but disables the near-boundary amplification regime. As shown in Figure 5, DGE ($c = 3$) tracks PGS on the residence–transition–loss trajectories, whereas clamped DGE ($c = 1$) reverts toward the scale-side baseline. Even the higher-clamp DGE ($c = 5$) lowers $\rho_{\delta=0.01}(t)$ and attenuates $TR(t)$ further, indicating that stronger near-boundary amplification con-

sistently improves stability. This trend is most distinct in Panel (a), where the high-LR Llama 3.2 1B Instruct setting makes training prone to loss spikes; it also persists in the more stable Qwen3-0.6B 2-bit setting (Panel (b)), where the residence–transition–loss gap across clamps remains clear. Table 7 further reports downstream quality across c , where enabling near-boundary amplification ($c > 1$) consistently outperforms the disabled variant ($c = 1$), confirming that this stabilization transfers to final model quality.

Two observations bracket the effect. PGS, which applies near-boundary amplification with no soft-step shaping, already reproduces the stabilization signature, so amplification is sufficient. Removing amplification within DGE while keeping its shaping (DGE $c = 1$) loses the signature, so amplification is also the operative component inside DGE. Together, this indicates that in the extreme low-bit regime we study, DGE’s stabilization is driven by near-boundary amplification rather than by soft-step shaping away from the boundary. We do not claim that amplification is DGE’s only useful effect, nor that DGE was intended as a residence intervention; rather, our lens identifies near-boundary gradient amplification as a shared mechanism that DGE does not analyze, and reinterprets its clamp c as a control on that mechanism.

5 Related Work

Low-Bit QAT Dynamics and Scale-Side Stabilization. Extreme low-bit QAT is unstable because minor updates trigger abrupt bin transitions in ternary and 2-bit regimes. Recent works stabilize these dynamics via transition-rate control (Lee et al., 2025b), optimizer-state retention (Nielsen et al., 2025), or trust-gradient estimation (Panferov et al., 2025). Another line focuses on learnable quantization parameters, such as gradient scaling in LSQ (Essex et al., 2020) and LSQ+ (Bhalgat et al., 2020), or combines parameter tuning with block-wise training, distillation, and progressive precision schedules (Liu et al., 2024b; Du et al., 2024; Chen et al., 2025; Liu et al., 2026; Lee et al., 2025a). While these methods mitigate scale-induced pivot motion, our work uniquely targets the complementary weight-side state: the population of pivot-proximal weights and their temporal dwell time.

Surrogate Gradients and Low-Bit LLM Recipes. QAT typically relies on the non-differentiable STE,

where the choice of surrogate gradient determines optimization stability (Yin et al., 2019). Recent approaches design richer backward paths, including Fourier-based surrogates (Chen et al., 2026), differentiable estimators for FP4 training (Wang et al., 2025), and stochastic rounding analyses (Chmiel et al., 2026). Concurrently, low-bit LLM compression has advanced through distillation-guided recipes (Liu et al., 2024b; Du et al., 2024; Chen et al., 2025; Liu et al., 2026; Lee et al., 2025a), vector post-training quantization (Liu et al., 2024a), and architectural innovations in binary or ternary LLMs (Wang et al., 2023; Ma et al., 2024; Xu et al., 2024; Chen et al., 2024; Kaushal et al., 2025). Rather than replacing these methods, PGS acts as a backward-only probe that isolates pivot-proximal gradient amplification to study how near-pivot mobility reduces residence and stabilizes training.

6 Conclusion

We introduced pivot-proximal residence as a training-time fragility state in extreme low-bit QAT. Using PGS as a backward-only intervention, we isolated a weight-side mechanism: pivot-local gradient amplification increases near-pivot mobility and shortens the dwell time of latent weights in fragile pivot bands. The resulting residence-time reduction consistently attenuates transition-rate bursts, smooths optimization, and improves final quality across extreme low-bit QAT settings. This perspective provides a diagnostic lens for analyzing existing low-bit QAT methods and designing future stabilization techniques.

7 Limitations

Our analysis assumes identifiable quantization pivots and is most directly applicable to uniform or SEQ-like quantizers. Whether the pivot-proximal residence view extends to non-uniform codebooks, learned codebooks, and joint weight-activation QAT remains to be validated. Our experiments also focus on weight-only extreme low-bit QAT for Llama and Qwen model families, so broader validation across architectures and QAT recipes is needed.

PGS introduces design choices such as amplification strength α and the pivot-band schedule $\delta(t)$. Although our ablations show that locality and amplification jointly control pivot-band evacuation, optimal settings may depend on bitwidth, model scale, optimizer, and learning-rate schedule.

Due to the limited compute budget, we could not repeat QAT runs across multiple random seeds under a fixed setting. Instead, we allocated our budget toward demonstrating consistency across diverse conditions—multiple model families and scales, learning rates, and longer QAT runs up to 30B tokens—which we view as complementary evidence of robustness.

Theorem 3.1 relies on a local random-walk approximation within the pivot-proximal region. As shown in Table 3, this approximation may be less accurate for very short visits under momentum-based optimizers.

References

- Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgrén, Xuan-Son Nguyen, Clémentine Fourier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf. 2025. [SmolLM2: When smol goes big – data-centric training of a small language model](#).
- Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.
- Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. 2020. Lsq+: Improving low-bit quantization through learnable offsets and better initialization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 696–697.
- Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.
- Mengzhao Chen, Wenqi Shao, Peng Xu, Jiahao Wang, Peng Gao, Kaipeng Zhang, and Ping Luo. 2025. Efficientqat: Efficient quantization-aware training for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10081–10100.
- Tianqi Chen, Zhe Li, Weixiang Xu, Zeyu Zhu, Dong Li, Lu Tian, Emad Barsoum, Peisong Wang, and Jian Cheng. 2024. Ternaryllm: Ternarized large language model. *arXiv preprint arXiv:2406.07177*.
- Tianyi Chen, Sihan Chen, Xiaoyi Qu, Dan Zhao, Ruomei Yan, Jongwoo Ko, Luming Liang, and Pashmina Cameron. 2026. Stableqat: Stable quantization-aware training at ultra-low bitwidths. *arXiv preprint arXiv:2601.19320*.
- Brian Chmiel, Maxim Fishman, Ron Banner, and Daniel Soudry. 2026. [FP4 all the way: Fully quantized training of large language models](#). In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*.
- Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. 2022. Llm.int8(): 8-bit matrix multiplication for transformers at scale. In *Proceedings of the 36th International Conference on Neural Information Processing Systems, NIPS ’22*, Red Hook, NY, USA. Curran Associates Inc.
- Dayou Du, Yijia Zhang, Shijie Cao, Jiaqi Guo, Ting Cao, Xiaowen Chu, and Ningyi Xu. 2024. Bitdistiller: Unleashing the potential of sub-4-bit llms via self-distillation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 102–116.
- Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. 2020. [Learned step size quantization](#). In *International Conference on Learning Representations*.
- Elias Frantar, Saleh Ashkboos, Torsten Hoeftler, and Dan Alistarh. 2022. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*.
- Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.
- Ayush Kaushal, Tejas Vaidhya, Arnab Kumar Mondal, Tejas Pandey, Aaryan Bhagat, and Irina Rish. 2025. Surprising effectiveness of pretraining ternary language model at scale. In *The Thirteenth International Conference on Learning Representations*.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Jung Hyun Lee, Seungjae Shin, Vinnam Kim, Jaeseong You, and An Chen. 2025a. Unifying block-wise ptq and distillation-based qat for progressive quantization toward 2-bit instruction-tuned llms. *arXiv preprint arXiv:2506.09104*.
- Junghyup Lee, Jeimin Jeon, Dohyung Kim, and Bumsub Ham. 2025b. Scheduling weight transitions for quantization-aware training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23466–23475.
- Yifei Liu, Jicheng Wen, Yang Wang, Shengyu Ye, Li Lyna Zhang, Ting Cao, Cheng Li, and Mao Yang. 2024a. Vptq: Extreme low-bit vector post-training quantization for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8181–8196.
- Zechun Liu, Barlas Oguz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. 2024b. Llm-qat: Data-free quantization aware training for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 467–484.
- Zechun Liu, Changsheng Zhao, Hanxian Huang, Sijia Chen, Jing Zhang, Jiawei Zhao, Scott Roy, Lisa Jin, Yunyang Xiong, Yangyang Shi, et al. 2026. Paretoq: Improving scaling laws in extremely low-bit llm quantization. *Advances in Neural Information Processing Systems*, 38:91311–91336.
- Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.
- Shuming Ma, Hongyu Wang, Lingxiao Ma, Lei Wang, Wenhui Wang, Shaohan Huang, Li Dong, Ruiping Wang, Jilong Xue, and Furu Wei. 2024. The era of 1-bit llms: All large language models are in 1.58 bits. *arXiv preprint arXiv:2402.17764*.
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. [Pointer sentinel mixture models](#).
- Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.
- Markus Nagel, Marios Fournarakis, Yelysei Bondarenko, and Tijmen Blankevoort. 2022. Overcoming oscillations in quantization-aware training. In *International Conference on Machine Learning*, pages 16318–16330. PMLR.
- Jacob Nielsen, Peter Schneider-Kamp, and Lukas Galke. 2025. Continual quantization-aware pre-training: When to transition from 16-bit to 1.58-bit pre-training for bitnet language models? In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 13483–13493.
- Andrei Panferov, Jiale Chen, Soroush Tabesh, Roberto L Castro, Mahdi Nikdan, and Dan Alistarh. 2025. Quest: Stable training of llms with 1-bit weights and activations. *arXiv preprint arXiv:2502.05003*.
- Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*.
- Philippe Tillet, H. T. Kung, and David Cox. 2019. [Triton: an intermediate language and compiler for tiled neural network computations](#). In *Proceedings of the 3rd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, MAPL 2019, page 10–19, New York, NY, USA. Association for Computing Machinery.
- Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Huaijie Wang, Lingxiao Ma, Fan Yang, Ruiping Wang, Yi Wu, and Furu Wei. 2023. Bitnet: Scaling 1-bit transformers for large language models. *arXiv preprint arXiv:2310.11453*.
- Ruizhe Wang, Yeyun Gong, Xiao Liu, Guoshuai Zhao, Ziyue Yang, Baining Guo, Zheng-Jun Zha, and Peng Cheng. 2025. Optimizing large language model training using fp4 quantization. In *International Conference on Machine Learning*, pages 62937–62957. PMLR.
- Yuzhuang Xu, Xu Han, Zonghan Yang, Shuo Wang, Qingfu Zhu, Zhiyuan Liu, Weidong Liu, and Wanxiang Che. 2024. Onebit: Towards extremely low-bit large language models. *Advances in Neural Information Processing Systems*, 37:66357–66382.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- P Yin, J Lyu, S Zhang, S Osher, YY Qi, and J Xin. 2019. Understanding straight-through estimator in training activation quantized neural nets. In *International Conference on Learning Representations*.
- Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. [Instruction-following evaluation for large language models](#).

A PGS Implementation Code

We implemented PGS on top of ParetoQ (Liu et al., 2026) Stretched Elastic Quant (SEQ) PyTorch autograd implementation as follows.

```
1 import math
2 import torch
3
4 class StretchedElasticQuant(torch.autograd.Function):
5     """Stretched Elastic Quantizer with optional PGS surrogate gradient."""
6
7     @staticmethod
8     def forward(
9         ctx,
10         weight,
11         scale,
12         num_bits,
13         use_pgs: bool = False,
14         pgs_delta: float = 0.05,
15         pgs_multiplier: float = 5.0,
16     ):
17         ctx.num_bits = num_bits
18
19         eps = torch.tensor(1e-5, device=scale.device).float()
20         scale = torch.where(scale > eps, scale, eps)
21
22         clip_val = 1 - 1e-2
23         if num_bits == 0:
24             n_levels, shift = 1.5, 0.0
25         else:
26             n_levels, shift = 2 ** (num_bits - 1), 0.5
27
28         ctx.save_for_backward(weight, scale)
29         ctx.other = use_pgs, pgs_delta, pgs_multiplier
30
31         if num_bits == 1:
32             q_w = weight.sign()
33         else:
34             q_w = (torch.round(torch.clamp(weight / scale, -clip_val, clip_val) * n_levels - shift) +
35                    shift) / n_levels
36         return q_w * scale
37
38     @staticmethod
39     def backward(ctx, grad_output):
40         weight, scale = ctx.saved_tensors
41         use_pgs, pgs_delta, pgs_multiplier = ctx.other
42         num_bits = ctx.num_bits
43
44         q_w = weight / scale
45         clip_val = 1 - 1e-2
46         n_levels = 1.5 if num_bits == 0 else 2 ** (num_bits - 1)
47         shift = 0.0 if num_bits == 0 else 0.5
48
49         q_w_final = (torch.round(torch.clamp(q_w, -clip_val, clip_val) * n_levels - shift) + shift) /
50         n_levels
51         q_w_diff = q_w_final - q_w
52
53         indicate_small = (q_w < -clip_val).float()
54         indicate_big = (q_w > clip_val).float()
55         indicate_middle = 1.0 - indicate_small - indicate_big
56
57         Qp = (n_levels - shift) / n_levels
58         Qn = -Qp
59         grad_scale = 1.0 / math.sqrt(weight.numel() * Qp)
60
61         if num_bits == 1:
62             grad_scale_out = weight.sign() * grad_output * grad_scale
63         else:
64             grad_scale_out = (indicate_small * Qn + indicate_big * Qp + indicate_middle * q_w_diff) *
65             grad_output * grad_scale
66         grad_scale_out = grad_scale_out.sum(dim=-1, keepdim=True)
67
68         if use_pgs:
69             M = 3 if num_bits == 0 else 2.0 * num_bits
70             grad_multiplier = torch.where(q_w_diff.abs() > (1.0 - pgs_delta) / M, pgs_multiplier, 1.0)
71             grad_multiplier = torch.where(q_w.abs() < 1.0 + (pgs_delta / M), grad_multiplier, 1.0)
72             grad_weight = indicate_middle * grad_output
73             grad_weight = grad_weight * grad_multiplier
74
75         return grad_weight, grad_scale_out, None, None, None, None
```

Table 8: Kernel-level latency overhead of PGS for quantize–dequantize operations on selected layers of Llama 3.2 1B Instruct and Llama 3.2 3B Instruct.

Layer shape	w/o PGS (ms)	w/ PGS (ms)	Overhead
Llama 3.2 1B Instruct q_proj (2048×2048)	0.901	0.905	+0.5%
Llama 3.2 1B Instruct gate_proj (8192×2048)	0.933	0.911	-2.3%
Llama 3.2 3B Instruct q_proj (3072×3072)	0.959	0.950	-0.9%
Llama 3.2 3B Instruct gate_proj (8192×3072)	0.950	0.973	+2.5%

Table 9: End-to-end training-step latency overhead of PGS.

Model	w/o PGS (s/step)	w/ PGS (s/step)	Overhead
Llama 3.2 1B Instruct	3.14	3.13	-0.3%
Qwen3-1.7B	4.52	4.54	+0.4%
Qwen3-0.6B	2.10	2.10	-0.1%

B PGS Computational Overhead

PGS incurs no measurable additional memory overhead in our implementation. Specifically, we fuse PGS into the existing SEQ backward kernel implemented in OpenAI Triton (Tillet et al., 2019), adding only an element-wise pivot-proximity comparison and a conditional multiplication (see the backward function in Section A). This design requires neither an additional kernel launch nor an intermediate tensor. Consequently, the measured peak GPU memory is identical with and without PGS across all profiled configurations (MB).

Tables 8 and 9 report the kernel-level latency and end-to-end training time, respectively. The kernel-level latency differences range from -2.3% to +2.5%, consistent with run-to-run measurement variability, while the end-to-end step-time differences remain within $\pm 0.4\%$ across all evaluated models. This negligible overhead arises because the weight quantization–dequantization kernel is memory-bound, allowing the small amount of additional computation required for the pivot-proximity comparison and conditional scaling to be effectively hidden by memory-access latency.

C Uniform Scale-Gradient Rescaling under Adam

We analyze the effect of the LSQ-style uniform rescaling of the scale gradient under the Adam optimizer. For a scale parameter s , the Adam update is

$$\Delta_t^{\text{Adam}} = -\eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}, \quad (11)$$

where $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first and second moments of the gradient.

Suppose that the gradient of this parameter is uniformly rescaled by a fixed positive constant c , i.e., $g_\tau \mapsto cg_\tau$ for all steps $\tau \leq t$. Then both the first and second moments are rescaled accordingly:

$$\hat{m}_t \mapsto c \hat{m}_t, \quad \hat{v}_t \mapsto c^2 \hat{v}_t.$$

Substituting these into the Adam update gives

$$-\eta \frac{c \hat{m}_t}{\sqrt{c^2 \hat{v}_t} + \varepsilon} = -\eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon/c} \xrightarrow{\varepsilon \rightarrow 0} -\eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t}}. \quad (12)$$

Thus, after the moment statistics have adapted, a fixed positive rescaling of the gradient is largely absorbed by Adam’s normalization.

In our setting, LSQ multiplies $\partial \mathcal{L} / \partial s$ by the constant factor

$$c = \frac{1}{\sqrt{N_W Q_P}},$$

41524

where N_W is the number of weights sharing the step size and $Q_P := \max z$ is the maximum positive quantization bin. Therefore, under Adam, this uniform rescaling does not reliably constrain the realized scale update magnitude, and hence does not provide a robust mechanism for limiting scale-induced pivot motion.

This cancellation is approximate rather than exact. Residual effects may remain due to finite ε , transient adaptation of the moment statistics, or additional operations such as gradient clipping. This is why we use Scale-Clip as a stronger scale-side control: unlike uniform gradient rescaling, it directly constrains the realized scale update and therefore cannot be absorbed by Adam’s adaptive normalization.

C.1 Scale-Clip: State-dependent scale-update clipping

Scale-Clip is introduced as a stronger scale-side baseline than LSQ in Section 2.3. Unlike LSQ-style uniform scale-gradient rescaling, it directly constrains the realized update of the learned step size and therefore is not normalized away by Adam’s adaptive moment normalization.

For each quantization group $\mathcal{G}$ (e.g., all latent weights associated with one output channel in per-channel quantization, or the set of latent weights sharing one step size in group quantization, so that $|\mathcal{G}|$ denotes the group size), let $s_{\mathcal{G}}(t)$ be the step size before the optimizer update, and let $\tilde{s}_{\mathcal{G}}(t+1)$ be the tentative step size proposed by Adam. We define

$$\Delta s_{\mathcal{G}}(t) = \tilde{s}_{\mathcal{G}}(t+1) - s_{\mathcal{G}}(t). \quad (13)$$

For 2-bit SEQ, the nonzero pivots are located at normalized coordinates $\pm a$ with $a = 1/2$, corresponding in raw weight space to $\pm a s_{\mathcal{G}}(t)$. The zero pivot is fixed under scale updates, so it is sufficient to control the motion of these nonzero pivot magnitudes. Using symmetry, for each latent weight $w_i(t) \in \mathcal{G}$, we define

$$r_i(t) = |w_i(t)| - a s_{\mathcal{G}}(t). \quad (14)$$

This is the signed distance from the current nonzero pivot magnitude in raw weight space.

The closest inner and outer crossing margins in scale-update units are

$$\Delta s_{\mathcal{G}}^-(t) = \max_{w_i(t) \in \mathcal{G}: r_i(t) < 0} \frac{r_i(t)}{a}, \quad \Delta s_{\mathcal{G}}^+(t) = \min_{w_i(t) \in \mathcal{G}: r_i(t) \geq 0} \frac{r_i(t)}{a}, \quad (15)$$

where $\Delta s_{\mathcal{G}}^-(t) = -\infty$ or $\Delta s_{\mathcal{G}}^+(t) = +\infty$ if the corresponding set is empty.

Scale-Clip then applies

$$\Delta s_{\mathcal{G}}^{\text{S-Clip}}(t) = \text{clip}(\Delta s_{\mathcal{G}}(t), \mu \Delta s_{\mathcal{G}}^-(t), \mu \Delta s_{\mathcal{G}}^+(t)), \quad (16)$$

and updates

$$s_{\mathcal{G}}(t+1) = s_{\mathcal{G}}(t) + \Delta s_{\mathcal{G}}^{\text{S-Clip}}(t), \quad (17)$$

where $\mu \in (0, 1]$ is a safety multiplier. We use $\mu = 0.1$.

This clipping rule prevents the scale update itself from moving the nonzero pivot magnitudes across any current latent weight in the group. It does not, however, prevent bin transitions caused by updates to the latent weights. This distinction is important: Scale-Clip isolates the effect of scale-side control and shows that even strong crossing-aware pivot-motion control is insufficient to eliminate low-bit QAT instability.

D Dwell Time Analysis of PGS (Proof of Theorem 3.1)

In this appendix, we show that under a local random-walk approximation, PGS achieves a quadratic reduction in the expected dwell time.

Notation Suppose that the weight w_i enters a pivot-proximal region of a pivot p at time $t = 0$, i.e., $d_i(0) = |w_i(0) - p| \leq \delta$. Although the pivot p can evolve over time due to updates in the scaling parameters, such changes are typically small across iterations. Therefore, for the purpose of local analysis within a short pivot-band visit, we assume that the pivot remains approximately fixed. For notational convenience, we define the pivot-centered variable $\tilde{w}_i(t)$ as

$$\tilde{w}_i(t) = w_i(t) - p. \quad (18)$$

Let τ_i denote the first time that w_i exits the pivot-proximal region:

$$\tau_i = \inf\{t \geq 0 : |\tilde{w}_i(t)| > \delta\}. \quad (19)$$

Local random-walk approximation In our analysis, we assume that during a short pivot-band visit, the stochastic gradient sequence $\{g_i(t)\}$ can be approximated as

$$\mathbb{E}[g_i(t)] \approx 0, \quad \mathbb{E}[g_i^2(t)] \approx \sigma^2. \quad (20)$$

This approximation is motivated by empirical observations that near a pivot, gradients tend to exhibit oscillatory behavior with frequent sign changes rather than a consistent directional drift (as reported in (Nagel et al., 2022)), resulting in an approximately zero local mean. Under this local random-walk assumption, the increments $\Delta_i(t) := \tilde{w}_i(t+1) - \tilde{w}_i(t)$ can be viewed as a mean-zero random variable with variance σ^2 . Then, it is well known that the expected first-exit time τ_i from the interval $(-\delta, \delta)$ satisfies

$$\mathbb{E}[\tau_i] \approx \frac{\delta^2}{\sigma^2}. \quad (21)$$

Effectiveness of PGS under SGD Under SGD with learning rate η , the update increment Δ_i^{SGD} in $\tilde{w}_i$ satisfies

$$\Delta_i^{\text{SGD}}(t) := \tilde{w}_i(t+1) - \tilde{w}_i(t) = w_i(t+1) - w_i(t) = -\eta g_i(t),$$

and then we have

$$\mathbb{E}[(\Delta_i^{\text{SGD}})^2] \approx \eta^2 \sigma^2. \quad (22)$$

Combining this with (21), we obtain

$$\mathbb{E}[\tau_i^{\text{SGD}}] \approx \frac{\delta^2}{\eta^2 \sigma^2}.$$

Now, we consider the case where PGS is applied. Under PGS, the gradient is amplified as $g_i \mapsto \alpha g_i$ within the pivot-proximal region. The increment then becomes

$$\Delta_i^{\text{PGS,SGD}}(t) = -\alpha \cdot \eta g_i(t), \quad (23)$$

and therefore

$$\mathbb{E}\left[\left(\Delta_i^{\text{PGS,SGD}}(t)\right)^2\right] \approx \alpha^2 \cdot \eta^2 \sigma^2.$$

Then, by applying (21) again, we have

$$\mathbb{E}[\tau_i^{\text{PGS,SGD}}] \approx \frac{1}{\alpha^2} \cdot \frac{\delta^2}{\eta^2 \sigma^2} = \frac{1}{\alpha^2} \mathbb{E}[\tau_i^{\text{SGD}}]. \quad (24)$$

Effectiveness of PGS under Adam For Adam, exponential moving averages (EMAs) are used to estimate the first and second moments of the gradients:

$$m_i(t) = \beta_1 m_i(t-1) + (1 - \beta_1) g_i(t), \quad (25a)$$

$$v_i(t) = \beta_2 v_i(t-1) + (1 - \beta_2) g_i^2(t). \quad (25b)$$

The update increment $\Delta_i^{\text{Adam}}(t)$ of $\tilde{w}_i$ is then given by

$$\Delta_i^{\text{Adam}}(t) = -\eta \frac{\hat{m}_i(t)}{\sqrt{\hat{v}_i(t) + \varepsilon}}, \quad (26)$$

where $\hat{m}_i(t)$ and $\hat{v}_i(t)$ are the bias-corrected estimates defined as

$$\hat{m}_i(t) = \frac{m_i(t)}{1 - \beta_1^t}, \quad \hat{v}_i(t) = \frac{v_i(t)}{1 - \beta_2^t}. \quad (27)$$

To understand the local dynamics under Adam, we unroll the first-moment estimate:

$$m_i(t) = \beta_1^t m_i(0) + (1 - \beta_1) \sum_{j=1}^t \beta_1^{t-j} g_i(j). \quad (28)$$

This expression shows that the first moment consists of a decaying contribution from the pre-entry state and a weighted accumulation of recent gradients. Since $\beta_1 < 1$, the pre-entry state decays exponentially fast, which in turn implies that the drift becomes negligible for moderate values of t . Furthermore, since $\mathbb{E}[g_i(t)] \approx 0$ under the local random-walk approximation, the update $\Delta_i^{\text{Adam}}(t)$ can be approximated as a mean-zero stochastic process. Consequently, the expected first-exit time satisfies

$$\mathbb{E}[\tau_i^{\text{Adam}}] \approx \frac{\delta^2}{\sigma_{\text{Adam}}^2}, \quad (29)$$

where $\sigma_{\text{Adam}}^2 = \mathbb{E}[(\Delta_i^{\text{Adam}}(t))^2]$ denotes the effective local step variance.

When PGS is applied, the gradients are amplified as $g_i(t) \mapsto \alpha g_i(t)$ within the pivot-proximal region, leading to

$$m_i^{\text{PGS}}(t) = \beta_1 m_i^{\text{PGS}}(t-1) + \alpha \cdot (1 - \beta_1) g_i(t), \quad (30a)$$

$$v_i^{\text{PGS}}(t) = \beta_2 v_i^{\text{PGS}}(t-1) + \alpha^2 \cdot (1 - \beta_2) g_i^2(t). \quad (30b)$$

Unrolling the first moment gives

$$\begin{aligned} m_i^{\text{PGS}}(t) &= \beta_1^t m_i(0) + \alpha \cdot (1 - \beta_1) \sum_{j=1}^t \beta_1^{t-j} g_i(j) \\ &= \beta_1^t m_i(0) + \alpha (m_i(t) - \beta_1^t m_i(0)). \end{aligned} \quad (31)$$

As before, the pre-entry contribution $\beta_1^t m_i(0)$ decays rapidly, so for moderate t , we have

$$m_i^{\text{PGS}}(t) \approx \alpha m_i(t). \quad (32)$$

We now analyze the second moment under PGS. Unrolling the recursion and taking expectations yields

$$\mathbb{E}[v_i^{\text{PGS}}(t)] = \beta_2^t \mathbb{E}[v_i(0)] + \alpha^2 \cdot (1 - \beta_2^t) \sigma^2. \quad (33)$$

Assuming that the pre-entry second moment satisfies $\mathbb{E}[v_i(0)] \approx \sigma^2$, we obtain

$$\mathbb{E}[v_i^{\text{PGS}}(t)] \approx \sigma^2 [\beta_2^t + (1 - \beta_2^t) \alpha^2]. \quad (34)$$

Combining these results, the Adam update increment under PGS can be approximated as

$$\Delta_i^{\text{PGS,Adam}}(t) \approx \frac{\alpha}{\sqrt{\beta_2^t + (1 - \beta_2^t) \alpha^2}} \Delta_i^{\text{Adam}}(t). \quad (35)$$

Taking the ratio of squared step sizes, we obtain

$$A_{\text{Adam}}(t) := \frac{\mathbb{E} \left[\left(\Delta_i^{\text{PGS,Adam}}(t) \right)^2 \right]}{\mathbb{E} \left[\left(\Delta_i^{\text{Adam}}(t) \right)^2 \right]} \approx \frac{\alpha^2}{\beta_2^t + (1 - \beta_2^t)\alpha^2}. \quad (36)$$

Let k_* denote the effective duration of a pivot-band visit, i.e., a typical value of t at which a weight exits the pivot-proximal region. Then, since the expected first-exit time is inversely proportional to the local step variance, we obtain

$$\mathbb{E}[\tau_i^{\text{PGS,Adam}}] \approx \frac{1}{A_{\text{Adam}}(k_*)} \mathbb{E}[\tau_i^{\text{Adam}}], \quad (37)$$

which yields

$$\mathbb{E}[\tau_i^{\text{PGS,Adam}}] \approx \frac{\beta_2^{k_*} + (1 - \beta_2^{k_*})\alpha^2}{\alpha^2} \mathbb{E}[\tau_i^{\text{Adam}}]. \quad (38)$$

We emphasize that this behavior crucially depends on the separation of timescales between the two moments: the first moment decays rapidly since β_1^t vanishes over a short timescale, whereas the second moment adapts much more slowly due to the large value of β_2 . As a result, during typical short pivot-band visits, β_2^t remains close to one while β_1^t is already near zero, leading to a transient regime in which the gradient amplification is not fully normalized.

E Perturbation details in Table 1

We perform 2-bit per-channel QAT on the Llama 3.2 3B Instruct model. After QAT, we perturb the latent weights prior to quantization.

We use a continuous multiplicative perturbation rather than directly flipping bin assignments, because our goal is to probe the fragility of the post-QAT latent-weight geometry near quantization pivots. A direct bin-flip intervention would overwrite the trained assignment pattern with an artificial discrete reassignment. By contrast, the perturbation $w \mapsto w(1 + pu)$ more closely mirrors the training-time mechanism of bin reassignment in QAT, where small latent-weight or scale updates move weights across quantization pivots. At the same time, it preserves the post-QAT latent-weight geometry—in which weights accumulate near pivots—more faithfully than direct bin flipping, allowing us to test whether the trained near-pivot configuration is itself fragile to QAT-like continuous perturbations.

For perturbation strength $p \in [0, 1]$, we sample $u \sim \text{Uniform}(0, 1)$ independently for each weight and compute

$$q^{\text{pert}} = q(w(1 + pu); s, b). \quad (39)$$

The x-axis in Table 1 corresponds to the perturbation strength p .

F Experimental details

Table 10 summarizes the training setups for the six QAT runs used in the main analysis. The table is intended to separate run-specific differences (model, learning rate, quantization granularity, context length, and batch configuration) from recipe components that are shared across runs. In particular, it places the three 5B-token diagnostic runs and the three 30B-token longer-QAT runs in a single view for easy comparison. Our use of pretrained models and evaluation datasets is limited to research-only quantization-aware training and evaluation and follows their intended access conditions.

G Ablation for PGS cosine scheduling

All experiments in Section 4 use cosine scheduling for the pivot-band width δ_t . We compare this default choice with a fixed-band variant using the same initial width, $\delta = 0.05$, on Llama 3.2 1B Instruct 2-bit QAT at LR 4e-5. Figure 6 supports this choice: by shrinking the pivot band together with the cosine LR decay, the cosine schedule yields a more monotonic decrease in transition rate than the fixed band $\delta = 0.05$. This suggests that the fixed band keeps a broad amplification region active too late, whereas the

Table 10: Training setups for the six QAT runs used in the main analysis. *Diagnostic* denotes the three 5B-token runs in Table 5, and *Longer QAT* denotes the three 30B-token runs reported in Table 6. All runs use DCLM-Edu, JSD self-distillation loss, and AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight decay = 0). $G \times B \times A$ denotes number of GPUs $\times$ per-device batch size $\times$ gradient-accumulation steps; the value in parentheses is the effective per-step batch size. Unless otherwise noted, runs use bf16 mixed precision, ZeRO-2 (accelerate+DeepSpeed), cosine LR decay with 200 warm-up steps, and on-policy GKD distillation from the same-family instruct teacher. GPU-hours are summed across restarts and computed as wall-clock time multiplied by the number of GPUs.

	Diagnostic			Longer QAT		
	Llama 3.2-1B Instruct 2-bit (LR=2e-4)	Llama 3.2-1B Instruct 2-bit (LR=4e-5)	Qwen3-0.6B 2-bit (LR=2e-4)	Llama 3.2-3B Instruct 2-bit (LR=2e-5)	Qwen3-1.7B 2-bit (LR=2e-5)	Qwen3-4B-Instruct-2507 2-bit (LR=2e-5)
Quantization bit	2-bit			2-bit		
Dataset	DCLM-Edu			DCLM-Edu		
Loss	Jensen–Shannon divergence (JSD) loss with self-distillation (Lee et al., 2025a)			Jensen–Shannon divergence (JSD) loss with self-distillation (Lee et al., 2025a)		
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight-decay = 0)			AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, weight-decay = 0)		
LR Scheduler	Cosine scheduling			Cosine scheduling		
Quantization Granularity	group-wise (128)	group-wise (128)	group-wise (128)	per-channel	per-channel	per-channel
Training Context Length	4k	4k	4k	1k	1k	1k
$G \times B \times A$	$8 \times 4 \times 2 (=64)$	$8 \times 4 \times 2 (=64)$	$8 \times 4 \times 2 (=64)$	$8 \times 8 \times 4 (=256)$	$8 \times 8 \times 4 (=256)$	$16 \times 4 \times 4 (=256)$
Steps (Tokens)	19,073 (5.0B)	19,073 (5.0B)	19,073 (5.0B)	114,440 (30B)	114,440 (30B)	114,440 (30B)
GPU-hours	136.5	136.7	92.7	93.1	204.8	83.9

Table 11: Final-quality comparison of fixed and cosine pivot-band scheduling. The fixed band is marginally better in PPL, whereas the cosine schedule improves all downstream metrics and the average score.

Method	PPL↓	CSR↑	MMLU↑	IFEval↑	Avg.↑
PGS (fixed)	14.07	52.55	37.99	35.86	42.14
PGS (cosine)	14.10	52.88	39.60	38.78	43.75

cosine schedule makes the intervention increasingly localized as training stabilizes. Although the fixed band gives marginally better PPL, Table 11 shows that cosine scheduling improves downstream metrics, so we use it as the default throughout the main text.

H Gradient Amplification Locality Details

We design three control variants of PGS to test whether its effect depends on *where* amplification is applied, rather than on *how many* weights are amplified or on the total amplification budget. Each control preserves a complementary budget notion of canonical PGS (near-pivot) while perturbing locality.

Let

$$\pi_i(t; \delta) := \mathbf{1}[d_i(t) \leq \delta]$$

denote the canonical pivot-band selector. All controls are applied on the same backward-active weights as PGS (near-pivot), so that differences arise only from locality rather than from changes in which weights receive gradients.

PGS (random). *Count-matched random selection.* For each quantization group $\mathcal{G}$, let

$$n_{\mathcal{G}}^{PGS}(t) = \sum_{i \in \mathcal{G}} \pi_i(t; \delta)$$

be the number of weights selected by PGS (near-pivot). We then draw a uniformly random subset

$$\mathcal{S}_{\mathcal{G}}(t)$$

from the backward-active weights in $\mathcal{G}$ such that

$$|\mathcal{S}_{\mathcal{G}}(t)| = n_{\mathcal{G}}^{PGS}(t),$$

and use

$$\tilde{\pi}_i^{\text{rand}}(t) = \mathbf{1}[i \in \mathcal{S}_{\mathcal{G}(i)}(t)]. \quad (40)$$

Thus, the number of amplified weights is matched exactly, while their locations are randomized independently of the pivot distance $d_i(t)$.

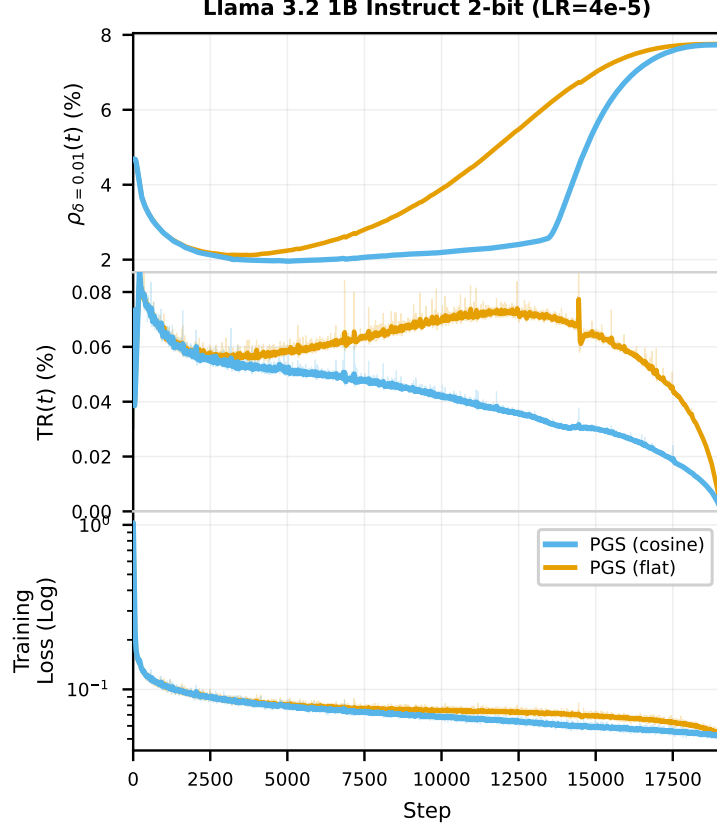


Figure 6: Fixed versus cosine pivot-band scheduling on Llama 3.2 1B Instruct 2-bit QAT at LR 4e-5. Cosine scheduling better matches the intended early-broad/late-localized intervention and yields a more steadily decreasing transition rate.

PGS (far-from-pivot). *Locality-reversed control with the same band width and multiplier.* PGS (near-pivot) amplifies weights with $d_i(t) \leq \delta$, whereas the far-from-pivot control amplifies weights in a band of the same width adjacent to the bin center:

$$\tilde{\pi}_i^{\text{far}}(t; \delta) = \mathbf{1}[d_i(t) \geq 1 - \delta]. \quad (41)$$

This preserves the band width and multiplier α of PGS (near-pivot) while reversing locality from bin boundaries to bin centers.

PGS (norm-matched). *Uniform per-group rescaling with matched multiplier energy.* For each group $\mathcal{G}$, let $n_{\mathcal{G}}^{\text{PGS}}(t)$ be the number of weights selected by PGS (near-pivot) under the same backward-active mask. PGS (near-pivot) applies multiplier α to those selected weights and multiplier 1 to the remaining backward-active weights in the group. Norm-matched replaces this spatially selective $\{\alpha, 1\}$ pattern with a single uniform multiplier $\tilde{\gamma}_{\mathcal{G}}^{\text{norm}}(t)$, chosen so that the average squared multiplier over the same backward-active weights is preserved. Equivalently, if

$$\tau_{\mathcal{G}}(t)$$

denotes the fraction of backward-active weights in group $\mathcal{G}$ that are selected by PGS (near-pivot), then

$$\tilde{\gamma}_{\mathcal{G}}^{\text{norm}}(t) = \sqrt{\tau_{\mathcal{G}}(t)\alpha^2 + (1 - \tau_{\mathcal{G}}(t))}. \quad (42)$$

Thus, norm-matched does *not* match the number of amplified weights. Instead, it matches the per-group squared multiplier budget of canonical PGS while removing locality entirely.

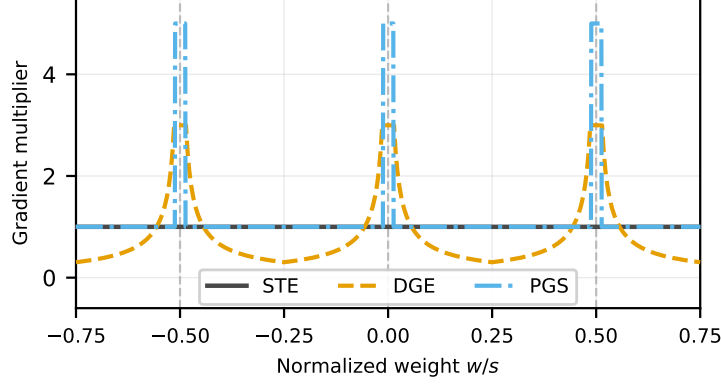


Figure 7: Backward gradient multipliers of STE, DGE ($k = 5, c = 3$), and PGS ($\alpha = 5, \delta = 0.05$). DGE yields a smooth-step derivative profile that attenuates gradients away from pivots and amplifies them near pivots, whereas PGS keeps the STE profile except for a pivot-local amplification band.

Summary. The three controls preserve either the number of amplified weights or the per-group multiplier budget of PGS (near-pivot), while changing only their locality. None reproduces the reduction in $\rho_{0.01}(t)$ or the smoothing of transition-rate and loss trajectories achieved by PGS (near-pivot), indicating that its effect is specifically tied to pivot-local amplification.

I Surrogate Multiplier Profiles

Figure 7 visualizes the backward gradient multipliers of STE, DGE, and PGS as functions of the normalized coordinate w/s . The purpose of this comparison is to make explicit which part of the surrogate profile is isolated by PGS.

STE uses a flat pass-through multiplier in the active region. By contrast, DGE induces a smooth-step derivative profile: it attenuates gradients away from pivots while amplifying them near pivots. PGS differs from both in that it preserves the STE profile almost everywhere and modifies it only through a localized amplification band near the pivots. This makes PGS a useful probe for isolating the pivot-proximal amplification component of DGE-style surrogate gradients, without introducing additional shaping away from pivots.

J PGS with CAKLD loss

Table 12: Comparison between LSQ and PGS with replacing the training objective with BitDistiller’s CAKLD loss. PPL, CSR, MMLU, IFEval, and Avg. follow the same evaluation protocols as in Table 2, while In-band %, Entry rate, Med. τ_i , and P90 τ_i are the same trajectory-level residence statistics as in Table 5.

Backward estimator	PPL ↓	CSR ↑	MMLU ↑	IFEval ↑	Avg. ↑	In-band % ↓	Entry rate ↓	Med. τ_i ↓	P90 τ_i ↓
LSQ	20.00	46.5	42.7	23.7	37.6	3.46	1.181	8	46
PGS	19.79	46.8	44.1	24.8	38.6	1.05	0.287	4	41

Table 12 extends the training objective loss from JSD to CAKLD of BitDistiller (Du et al., 2024). Same as with the JSD loss, PGS further improves PPL and downstream accuracy while reducing residence statistics compared to the LSQ baseline.