JACQUANT: STE-FREE QUANTIZATION-AWARE TRAINING VIA LEARNED JACOBIAN SURROGATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Quantization-aware training (QAT) usually propagates gradients through hard quantizers with a fixed straight-through estimator (STE). At ultra-low precision, this backward rule ignores how quantized weights respond to finite perturbations. We introduce JacQuant, a QAT framework that learns a lightweight surrogate of quantizer sensitivity while retaining the hard forward quantizer. A periodic Gaussian probe fits one scalar gain per weight group; the gains are averaged over time and reused between refreshes. We identify the population regression target through Gaussian smoothing and separate it from the clipped estimator used in practice. For a local linearized surrogate, we establish non-convex stationarity bounds and linear convergence to an error neighborhood under a Polyak–Łojasiewicz condition, with explicit bias and variance terms. Across 1B–8B LLaMA and Qwen models with 1–2-bit weights, JacQuant improves both perplexity and mean zero-shot accuracy over the ParetoQ and WinQ baselines. On LLaMA3-1B W2A16, it lowers ParetoQ perplexity by 8.9% and recovers roughly one third of the gap to FP16 in both metrics. Gains persist at all three 8B checkpoints through 90K steps. In its default configuration, JacQuant incurs only a 1.6–3.5% reduction in training throughput on 1B models and 0.76% on 8B relative to the corresponding STE baselines. It requires no additional computation at inference.

1 INTRODUCTION

Quantization is a primary route to deploy large language models (LLMs) under tight memory and latency budgets by reducing parameter precision and improving hardware utilization (Egiazarian et al., 2024; Frantar et al., 2023; Tseng et al., 2024; Liu et al., 2024; 2025b). In many practical deployments, *post-training quantization* (PTQ) remains attractive because it avoids gradient-based training and can be applied quickly to a pretrained checkpoint (Frantar et al., 2023; Tseng et al., 2024). However, when pushing to *ultra-low precision*, most notably $\leq$ 2-bit weights, PTQ often suffers sharp degradation, and *quantization-aware training* (QAT) becomes important to recover accuracy by adapting scales, clipping, and representations to the task distribution (Liu et al., 2025b; Li et al., 2026).

Most QAT methods use the *Straight-Through Estimator* (STE) or a clipped variant (Bengio et al., 2013; Esser et al., 2020): the forward pass uses a hard quantizer Q_Δ, while the backward pass substitutes a tractable derivative. For fixed quantization parameters, the hard map has derivative zero almost everywhere and is discontinuous at thresholds. An identity backward rule therefore describes neither its pointwise derivative nor its response to finite perturbations. This distinction matters at low bit-widths, where threshold crossings and saturation can produce very different responses across weight groups (Yin et al., 2019; Nagel et al., 2020).

Key idea: learn a reusable finite-scale sensitivity map. We introduce JacQuant (Jacobian learning for Quantization), which keeps the deployed hard quantizer in the forward pass and replaces its fixed backward gain with a learned map $B(W)$. Let $q = Q_\Delta(W)$ and $v = \nabla_q \ell$. The weight-gradient surrogate is $g_B = B(W)v$, with $B(W) = \mathrm{blkdiag}(b_1 I, \ldots, b_G I)$. Each b_g summarizes the response of a group to sampled weight perturbations. The implemented gains lie in $[0, 1]$, providing a compact, bounded model of finite-scale quantization sensitivity that preserves or attenuates the incoming signal group by group.

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.

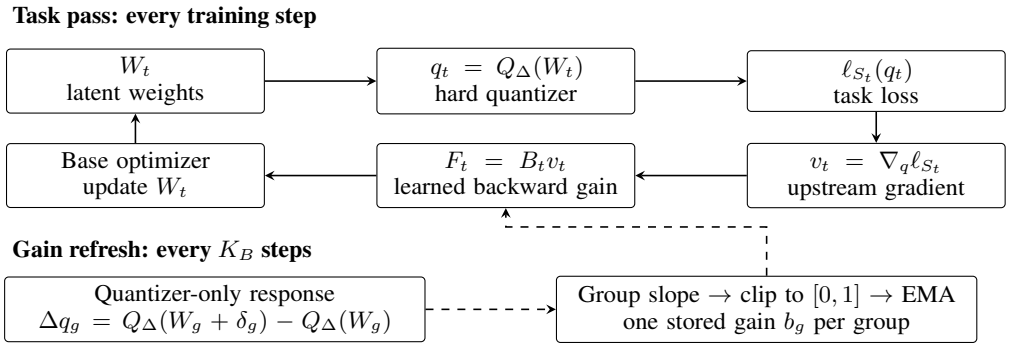

Figure 1: JacQuant separates training from sensitivity estimation. The task forward pass uses $Q_\Delta(W)$; the backward pass multiplies the upstream gradient by group gains. Quantizer-only probes refresh those gains every K_B steps. Dashed paths denote the refresh computation, which does not evaluate the task loss.

Fig. 1 summarizes the separation between the hard forward pass, the learned backward gains, and their periodic refresh. The Rotation Trick (Fifty et al., 2025) instead derives a rotation and rescaling from each input–codeword pair and treats that transform as constant during differentiation. JacQuant estimates scalar gains from perturbation responses and reuses them over time. The matched comparison in Tab. 1a evaluates these two backward designs within the ParetoQ pipeline.

The sensitivity refresh queries the quantizer, rather than the full LLM loss, and stores one scalar per group. This makes the update inexpensive enough to amortize over many QAT steps. We also describe a dither-based variant and optional control variates, but distinguish these choices from the core learned backward rule. Our analysis explains the finite-scale target and the remaining approximation error; our experiments test the resulting training configurations.

Contributions. (a) Learning the backward rule for ultra-low-bit QAT. We introduce JacQuant, which replaces the fixed STE in weight quantization with a lightweight, learned Jacobian surrogate. This makes the backward signal adaptive to quantization sensitivity while preserving the original hard forward quantizer. **(b) Theoretical foundations for sensitivity-aware optimization.** We characterize what the learned surrogate estimates and establish non-convex stationarity bounds and Polyak–Łojasiewicz convergence rates for a local smooth objective. These results quantify how estimation bias and variance affect optimization accuracy. **(c) Consistent performance gains with low training overhead.** Across the evaluated 1B–8B LLMs and 1–2-bit settings, JacQuant consistently improves both perplexity and mean zero-shot accuracy over ParetoQ and WinQ, and outperforms the matched Rotation Trick baseline. These improvements come with modest additional training overhead and no additional inference computation.

2 Related Work

Post-training quantization (PTQ). PTQ calibrates a pretrained model without extended QAT retraining and remains a strong default for rapid LLM deployment. Representative methods include SmoothQuant (Xiao et al., 2023) (activation outlier smoothing for W8A8), GPTQ (Frantar et al., 2023) (Hessian-informed weight-only quantization), ZeroQuant/ZeroQuant-V2 (Yao et al., 2022; 2023) (per-layer PTQ with hardware-friendly kernels), and AWQ (Lin et al., 2024) (activation-aware protection of salient channels for 4-bit inference).

Quantization-aware training (QAT). QAT typically improves accuracy in aggressive regimes (e.g., $\leq$2-bit weights and/or W&A quantization) by adapting scales and clipping to the task distribution rather than relying purely on calibration statistics. Classic STE-based QAT spans early work such as DoReFa-Net (Zhou et al., 2016), PACT (Choi et al., 2018), and step-size learning methods such as LSQ/LSQ+ (Esser et al., 2020; Bhalgat et al., 2020). Recent findings similarly suggest QAT is often preferable to PTQ in ultra-low-bit settings (e.g., ParetoQ (Liu et al., 2025b), WinQ (Li et al., 2026)).

Beyond the Straight-Through Estimator. QAT commonly uses an STE or a clipped surrogate (Bengio et al., 2013; Esser et al., 2020). PV-Tuning (Malinovskii et al., 2024) alternates optimization of discrete assignments and continuous quantized values. The Rotation Trick (Fifty et al., 2025) constructs an input-dependent rotation and rescaling for the backward pass. Differentiable relaxations (Agustsson et al., 2017), hybrid first/zeroth-order estimators (Yang & Aamodt, 2025), and forward-only QAT (Tan et al., 2025) offer other approaches. JacQuant instead fits a scalar response per group using quantizer-only perturbations, while retaining ordinary backpropagation through the rest of the network. Expanded comparisons appear in Sec. H.

Variance reduction. Control-variate methods such as SVRG (Johnson & Zhang, 2013), SAGA (Defazio et al., 2014), and JacSketch (Gower et al., 2021) reduce sampling noise under appropriate oracle assumptions. They complement sensitivity learning but incur their own state and computation costs. We specify these costs and the reference-gradient error in Secs. E and H. Sensitivity learning addresses the backward approximation, while control variates address minibatch sampling noise.

3 METHOD

We introduce JacQuant, a QAT framework that replaces STE with a *learned surrogate Jacobian*. Here STE-free refers to the targeted weight-quantizer backward path; other quantized operators and scale/clipping updates follow the base pipeline. Rather than treating quantization as "identity in backward", we learn how perturbations of full-precision weights affect their quantized proxies and use this sensitivity to map gradients back to weight space.

3.1 PRELIMINARIES AND SENSITIVITY GROUPS

Consider a linear layer with weights $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ and input x, producing $z = Wx$. QAT uses a quantized proxy $\widehat{W} = Q_\Delta(W)$ in the forward pass. Partition the quantized weights into sensitivity groups $\{W^{(g)}\}_{g=1}^{G}$. The sensitivity group size g_s determines how many weights share a backward gain; it need not equal the group size used for forward scales. The reported forward quantizers use per-channel scaling. For a sample (x_i, y_i), let $v_i := \nabla_{\widehat{W}} \ell(f(x_i; \widehat{W}), y_i)\big|_{\widehat{W}=Q_\Delta(W)}$. For fixed scales and thresholds, Q_Δ is piecewise constant: its derivative is zero inside cells and undefined at thresholds. We therefore learn a finite-scale sensitivity surrogate.

3.2 LEARNING A BACKWARD SENSITIVITY MAP

The deployed quantized model is evaluated using the objective

$$L(W) = \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[\ell(f(x; Q_\Delta(W)), y)\big]. \tag{1}$$

The identity STE propagates the upstream gradient unchanged. JacQuant instead learns a lightweight sensitivity map $B(W)$ and uses the backward rule

$$g_B(W) := B(W)\,\bar{v}(W), \qquad \bar{v}(W) := \mathbb{E}_{(x,y)}[\nabla_q \ell(f(x; q), y)]_{q=Q_\Delta(W)}. \tag{2}$$

Parameterization and cost. We use a block-diagonal $B(W)$ with one scalar per sensitivity group, $B(W) = \mathrm{blkdiag}(b_1 I, \ldots, b_G I)$. With group size 128 this adds one scalar per 128 weights. For N_q quantized weights and FP32 gains, the persistent gain state is $4\lceil N_q/g_s \rceil$ bytes, excluding temporary probes and optional control-variate state. We treat B_t as constant during each task backward pass; scale and clipping gradients follow the base QAT pipeline.

What a group gain represents. A gain measures how strongly the quantized weights respond along sampled perturbations. If most weights remain in their bins, the observed response is small; if perturbations cross thresholds, the fit increases. Sharing one gain pools these responses within a group. Pooling local responses into one group gain provides a compact representation of sensitivity and amortizes estimation across weights. Because the probe queries the quantizer rather than the task loss, the gain can be reused across minibatches. The upstream gradient still supplies the task-dependent direction.

Algorithm 1 JacQuant with a hard task forward pass

Require: Weights W_0, quantizer Q_Δ, refresh interval K_B, probe scale σ, EMA rate β
1: $b_g \leftarrow 1$ for every sensitivity group g
2: **for** $t = 0, 1, \ldots$ **do**
3: Sample minibatch S_t; set $q_t \leftarrow Q_\Delta(W_t)$
4: Backpropagate the task loss to obtain $v_t = \nabla_{q_t} \ell_{S_t}$
5: Form $F_t \leftarrow \mathrm{blkdiag}(b_1 I, \ldots, b_G I) v_t$
6: Set $g_t \leftarrow F_t$, or use the optional control variate in Sec. E
7: Apply the base optimizer to W_t using g_t
8: **if** $(t+1) \bmod K_B = 0$ **then**
9: **for** each sensitivity group g **do**
10: Query $\Delta q_g = Q_\Delta(W_g + \delta_g) - Q_\Delta(W_g)$, $\delta_g \sim \mathcal{N}(0, \sigma^2 I)$
11: Update b_g once using (3)
12: **end for**
13: **end if**
14: **end for**

3.3 Practical Estimation of $B(W)$

JacQuant updates $B(W)$ only on refresh steps (Alg. 1) and reuses it across many QAT iterations.

JacQuant-Probe **(default).** Hold forward scales and clipping thresholds fixed during each probe query. For each group g, draw $\delta_g \sim \mathcal{N}(0, \sigma^2 I)$ and measure $\Delta q_g = Q_\Delta(W_g + \delta_g) - Q_\Delta(W_g)$. A regularized one-probe slope fit and clipped exponential moving average (EMA) give

$$\hat{b}_g = \frac{\langle \Delta q_g, \delta_g \rangle}{\|\delta_g\|_2^2 + \epsilon}, \quad b_g \leftarrow (1-\beta) b_g + \beta \, \mathrm{clip}(\hat{b}_g, 0, 1). \tag{3}$$

Here $\epsilon > 0$ stabilizes the denominator and β is the weight of the *new* estimate. Finite probes may cross thresholds even when the unperturbed iterate stays in one cell. Small probes often yield no code changes; large probes mix responses over a wider region. Thus σ controls the sensitivity scale, rather than approximating a nonzero pointwise derivative. Clipping each estimate before the EMA also changes its population target; Sec. 4 accounts for this explicitly.

Initialization, refresh, and optimizer state. We initialize every b_g to one, so the first backward pass matches identity STE. Identity initialization preserves the baseline backward rule while the gain learner begins adapting to the quantizer's local responses. On each refresh, we update the gains once using the current weights and quantizer parameters, then reuse them for K_B task steps while weights and scales continue to change. Early gain updates therefore change the relative group contributions to the backward signal. The transient in Fig. 2(a) is consistent with this initial adaptation of the sensitivity map, followed by recovery to a lower training loss.

JacQuant-Dither**.** The Dither variant randomizes thresholds with $r_g \sim \mathcal{U}[-\Delta/2, \Delta/2]$ and the de-dithered proxy $Q_\Delta(W_g + r_g) - r_g$. We characterize the corresponding mean map in Sec. F.3. Our evaluated Dither variant uses this proxy in the task forward pass, so the comparison with Probe assesses the combined effect of the forward distribution and sensitivity estimator.

Interaction with the optimizer. With SGD, multiplying a group gradient by b_g is equivalent to using a group learning rate. With AdamW, a constant positive gain largely cancels through first/second-moment normalization, whereas time-varying gains alter the moment history (Sec. F.11). Thus the relevant distinction is the response-based, evolving gain rule. The spectral bound in Lem. 3 concerns the incoming surrogate gradient, not a proportional reduction of the AdamW parameter update.

3.4 Optional Variance Reduction

Sensitivity estimation and minibatch sampling introduce different errors. The base estimator $F_{S_t}(W_t, B_t) = |S_t|^{-1} \sum_{i \in S_t} B_t v_i$ uses the learned gains directly. An anchor-based control variate instead uses

$$g_t = F_{S_t}(W_t, B_t) - F_{S_t}(\widetilde{W}, \widetilde{B}) + \widetilde{g}, \qquad \widetilde{g} = F_{S_{\mathrm{ref}}}(\widetilde{W}, \widetilde{B}). \tag{4}$$

Using the same minibatch in the first two terms allows their sampling fluctuations to cancel when current and anchor directions are correlated. The reference term restores the anchor mean; a finite reference batch leaves a realized reference error. The anchor includes the quantizer parameters used to evaluate $\widetilde{W}$, and its gain $\widetilde{B}$ stays fixed until refresh.

Gain refresh and anchor refresh are separate operations: K_B denotes the deterministic gain interval, while p_A denotes a loopless anchor-refresh probability when that option is used. The latter has mean interval $1/p_A$; it is not interchangeable with K_B. Anchor evaluation adds computation and model-sized state. Sec. E gives the exact estimator semantics and contrasts the dataset-sized SAGA alternative. Our analysis keeps gain error, reference bias, and centered sampling variance separate.

4 Convergence Analysis of JacQuant

The analysis separates three objects: a smooth reference map, the population target of the implemented gain estimator, and the task-training direction. Smoothing a quantizer does not make its hard forward objective differentiable. Full proofs and extensions are in Sec. F.

4.1 A Finite-Scale Reference Gradient

Quantization $Q_\Delta : \mathbb{R}^d \to \mathbb{R}^d$ has derivative zero almost everywhere and undefined derivative on cell boundaries. Fix the forward scales and thresholds, and write $\mathcal{L}(q) = n^{-1} \sum_i \ell(f(x_i; q), y_i)$ for the loss as a function of dequantized weights. For Gaussian probes, define

$$m_\sigma(W) = \mathbb{E}_\delta[Q_\Delta(W + \delta)], \qquad J_\sigma(W) = Dm_\sigma(W), \quad \delta \sim \mathcal{N}(0, \sigma^2 I). \tag{5}$$

For clipped separable scalar quantizers, m_σ is smooth and J_σ is diagonal. Its entries depend on threshold distances and on σ: they can be near zero inside a cell when probes rarely cross a threshold, and can exceed one near a threshold. Deep saturation also gives small sensitivity.

A binary example makes this target concrete. For $Q(w) = a\ \mathrm{sign}(w)$, $a > 0$, Gaussian smoothing gives

$$J_\sigma(w) = \frac{2a}{\sigma}\, \varphi(w/\sigma), \tag{6}$$

where φ is the standard Gaussian density. Sensitivity peaks at the threshold and decays as $|w|/\sigma$ grows. Thus the same hard quantizer can yield very different gains across weights. The cap at one trades fidelity to a sharp threshold response for a bounded backward signal; it is an explicit design constraint.

Assumption 1 (Code preservation)**.** There exist t_0 and $T \geq 1$ such that $Q_\Delta(W_t) = Q_\Delta(W_{t_0}) \equiv q$ for all $t \in [t_0, t_0 + T)$. The quantizer parameters and all other trainable variables are held fixed for this local analysis.

Assume $\mathcal{L}$ is differentiable at q. Within the window, $\bar{v}^{(q)} = \nabla\mathcal{L}(q)$ is fixed. A well-defined local surrogate and its gradient are

$$\widetilde{L}_q(W) = \mathcal{L}(q) + \langle \bar{v}^{(q)}, m_\sigma(W) - q \rangle, \qquad g^\dagger(W) = J_\sigma(W)^\top \bar{v}^{(q)}. \tag{7}$$

This is the smoothed *linearization* of the task loss, not generally $\mathbb{E}_\delta[\mathcal{L}(Q_\Delta(W + \delta))]$. Sec. F.6 bounds their discrepancy. Code preservation is an analytical condition, not a restriction imposed during training.

Lemma 1 (Bias to the target gradient)**.** *On a code-preserving window, writing $J = J_\sigma(W)$ and $\bar{v} = \bar{v}^{(q)}$,*

$$\|g_B - g^\dagger\| \leq \|B - J^\top\|_{\mathrm{op}} \|\bar{v}\|, \qquad \|g^{\mathsf{STE}} - g^\dagger\| \leq \|I - J^\top\|_{\mathrm{op}} \|\bar{v}\|. \tag{8}$$

These are worst-case bounds, not unconditional dominance over STE. The actual error is $(B - J^\top)\bar{v}$: sensitivity mismatch matters most along directions emphasized by the task gradient. Thus a useful B must preserve task-relevant responses, rather than merely shrink the gradient norm. Cor. 1 gives the exact directional comparison.

4.2 What the Gain Estimator Learns

Lemma 2 (Population group regression). *For a group of size d_g, let $\Delta q_g = Q_\Delta(W_g+\delta_g) - Q_\Delta(W_g)$ and $\delta_g \sim \mathcal{N}(0, \sigma^2 I)$. The unrestricted scalar population least-squares solution is*

$$b_g^{\mathrm{LS}} = \frac{\mathbb{E}[\langle \Delta q_g, \delta_g \rangle]}{d_g \sigma^2} = \frac{\operatorname{tr} J_{\sigma,g}(W)}{d_g}. \tag{9}$$

The pooled empirical ratio converges almost surely to b_g^{LS} as the number of independent probes grows. Constraining the population fit to $[0,1]$ gives $b_g^\star = \Pi_{[0,1]}(b_g^{\mathrm{LS}})$.

The actual one-probe update in (3) has a different mean:

$$\bar{b}_g(W) = \mathbb{E}\left[\Pi_{[0,1]}\left(\frac{\langle \Delta q_g, \delta_g \rangle}{\|\delta_g\|^2 + \epsilon}\right)\right]. \tag{10}$$

Averaging ratios and clipping before averaging do not commute with population least squares. Let $\bar{B} = \mathrm{blkdiag}(\bar{b}_g I)$ and $B^\star = \mathrm{blkdiag}(b_g^\star I)$. Then

$$\|B_t - J_\sigma^\top\|_{\mathrm{op}} \le \underbrace{\|B_t - \bar{B}\|_{\mathrm{op}}}_{\text{tracking}} + \underbrace{\|\bar{B} - B^\star\|_{\mathrm{op}}}_{\text{estimator discrepancy}} + \underbrace{\|B^\star - J_\sigma^\top\|_{\mathrm{op}}}_{\text{group/clipping approximation}}. \tag{11}$$

The three terms answer different design questions. More observations reduce tracking noise at fixed weights, but do not remove estimator discrepancy. Finer groups can reduce within-group sensitivity differences, but cannot undo the gain cap. The probe scale changes the reference map itself. Thus these hyperparameters trade off different errors rather than serving as interchangeable ways to improve the same estimator.

Proposition 1 (EMA tracking). *At fixed W, let independent clipped estimates $s_k \in [0,1]^G$ have mean $\bar{b}$ and $\mathbb{E}[\|s_k - \bar{b}\|^2] \le V_B$. With $b_{k+1} = (1-\beta)b_k + \beta s_k$, $0 < \beta < 1$,*

$$\mathbb{E}[\|b_k - \bar{b}\|^2] \le (1-\beta)^{2k}\|b_0 - \bar{b}\|^2 + \frac{\beta}{2-\beta} V_B. \tag{12}$$

For a moving predictable target with RMS change at most D_B per refresh, the RMS tracking error is bounded by the decaying initial error plus $\sqrt{\beta V_B/(2-\beta)} + D_B/\beta$.

A constant EMA rate has a noise floor. Almost-sure convergence at fixed W requires diminishing rates under Prop. 2; it does not follow for the constant-rate implementation. Uniform dithering induces a different smoothing kernel and is treated in Sec. F.3.

4.3 Conditional Optimization Guarantees

Assumption 2 (Boundedness of B). $B(W)$ is diagonal or block-diagonal, uniformly bounded as $\|B(W)\| \le \beta_B$.

Lemma 3 (Spectral stability). *Under Assump. 2, $\|B(W)v_i\| \le \beta_B \|v_i\|$ for every per-example backpropagation signal v_i. For gains in $[0,1]$, $\beta_B = 1$. This bounds the surrogate signal, not the subsequent AdamW update.*

We analyze $W_{t+1} = W_t - \eta g_t$ for $\widetilde{L}_q$. Let $\mathcal{F}_t$ contain the current weights, gains, and any control-variate state.

Assumption 3 (Conditional oracle error). The surrogate is lower bounded by $\widetilde{L}_q^\star$ and L_s-smooth on every update segment in the window. The estimator satisfies

$$\mathbb{E}[g_t \mid \mathcal{F}_t] = \nabla \widetilde{L}_q(W_t) + e_t, \qquad \mathbb{E}[\|e_t\|^2] \le \varepsilon^2, \tag{13}$$

$$\mathbb{E}[\|g_t - \mathbb{E}[g_t \mid \mathcal{F}_t]\|^2 \mid \mathcal{F}_t] \le V. \tag{14}$$

Reference-gradient errors and any additional upstream-gradient approximation errors are included in e_t; variance reduction is reflected in V only when established for the chosen estimator.

Theorem 1 (Non-convex windowed convergence). *Under Assumps. 1 and 3, for $\eta \le 1/(4L_s)$,*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\nabla \widetilde{L}_q(W_t)\|^2] \le \frac{4(\widetilde{L}_q(W_0) - \widetilde{L}_q^\star)}{\eta T} + 3\varepsilon^2 + 2L_s \eta V. \tag{15}$$

Table 1: Main quantization results. Each pair reports WikiText-2 PPL (↓) and mean eight-task accuracy (%, ↑). Bold compares JacQuant with its own QAT baseline. PTQ rows provide context; full task scores are in Sec. B.

(a) `LLaMA3-1B`, A16.

	W1A16		W1.58A16		W2A16	
Metrics	PPL	Acc.	PPL	Acc.	PPL	Acc.
FP16 Baseline	9.6	58.5	9.6	58.5	9.6	58.5
RTN (PTQ)	4.2e8	33.7	1.8e6	36.2	1.5e6	36.2
GPTQ (PTQ)	3.3e8	32.7	7.5e4	32.7	3.8e4	32.7
SpinQuant (PTQ)	2.4e8	33.7	5.8e3	32.7	3.8e2	34.9
ParetoQ	17.9	51.9	14.8	54.3	13.5	55.7
+Rotation Trick	18.1	52.0	14.2	54.7	13.3	56.1
+JacQuant	**17.6**	**52.4**	**13.9**	**55.3**	**12.3**	**56.6**
WinQ	15.3	52.6	12.9	55.6	11.9	56.6
+JacQuant	**15.1**	**53.5**	**12.3**	**56.1**	**11.8**	**56.8**

(b) `LLaMA3-3B` and `Qwen3-1.7B`, A8.

	`LLaMA3-3B`				`Qwen3-1.7B`			
	W1A8		W1.58A8		W1A8		W2A8	
Metrics	PPL	Acc.	PPL	Acc.	PPL	Acc.	PPL	Acc.
FP16 Base	7.7	65.2	7.7	65.2	16.2	58.2	16.2	58.2
ParetoQ	15.7	54.1	13.1	56.0	46.5	42.3	22.2	47.9
+JacQuant	**15.2**	**54.9**	**12.7**	**57.1**	**45.3**	**42.7**	**21.5**	**48.1**

(c) `LLaMA3-8B`, W2A16.

	`30K`		`60K`		`90K`	
Metrics	PPL	Acc.	PPL	Acc.	PPL	Acc.
ParetoQ	11.78	58.54	10.43	60.67	9.79	61.41
+JacQuant	**11.69**	**59.10**	**10.32**	**60.98**	**9.71**	**62.04**

Assumption 4 (PŁ inequality)**.** There exists $\mu > 0$ such that $\frac{1}{2}\|\nabla \widetilde{L}(W)\|^2 \geq \mu(\widetilde{L}(W) - \widetilde{L}^\star)$ on the window. Here $\widetilde{L} = \widetilde{L}_q$.

Theorem 2 (PŁ regime)**.** *Under Assumps. 1, 3 and 4, if* $\eta \leq \min\{1/(4L_s), 1/\mu\}$,

$$\mathbb{E}[\widetilde{L}_q(W_t) - \widetilde{L}_q^\star] \leq (1 - \eta\mu/2)^t(\widetilde{L}_q(W_0) - \widetilde{L}_q^\star) + \frac{3\varepsilon^2}{2\mu} + \frac{L_s\eta V}{\mu}. \tag{16}$$

The bias includes all three errors in (11); the variance includes minibatch and reference-estimation noise. The results concern a local smooth surrogate and plain stochastic updates, rather than unconditional convergence of AdamW on the hard quantized loss. Sec. F.10 gives a composition bound with explicit accumulated objective drift when codes change.

Implications for training. The bounds favor a backward model that is accurate enough to lower bias without injecting excessive estimation noise. Faster gain refresh can reduce staleness while increasing the frequency of noisy changes; slower refresh amortizes cost but increases target drift. A control variate addresses centered sampling noise, whereas better gain estimates address the bias terms in (11). Neither mechanism guarantees the other's benefit. These distinctions motivate the separate group/interval ablation, finite-difference diagnostic, and runtime measurements below.

5 Experiments

We evaluate whether learned backward gains improve ultra-low-bit LLM QAT, how the gains transfer across base pipelines and model sizes, and what state and runtime they require. The comparisons separate three questions: final model quality, backward-signal behavior during training, and the cost of estimating that signal.

Experimental setup. We evaluate `LLaMA3-1B/3B/8B` (Grattafiori et al., 2024) and `Qwen3-1.7B` (Yang et al., 2025), using the model shorthand below. Weight precision is 1, 1.58 (ternary), or 2 bits, with A8 or A16 activations. Models undergo QAT on FineWebEdu and are evaluated by WikiText-2 perplexity (PPL; lower is better) and mean accuracy over eight zero-shot tasks (Acc.; higher is better). We augment ParetoQ (Liu et al., 2025b) and WinQ (Li et al., 2026); each pair retains its base forward-quantization and scale/clipping recipe. ParetoQ and WinQ are distinct training pipelines, so comparisons between them are not backward-rule ablations. The reported optimizer is AdamW with learning rate 2×10^{-5} and 1K warmup steps. Settings are selected by WikiText-2 validation PPL. Most runs span 240K steps; the 8B results report 30K/60K/90K checkpoints. Sec. A specifies the reported optimizer and selection protocol, and Sec. E distinguishes the optional control-variate configurations.

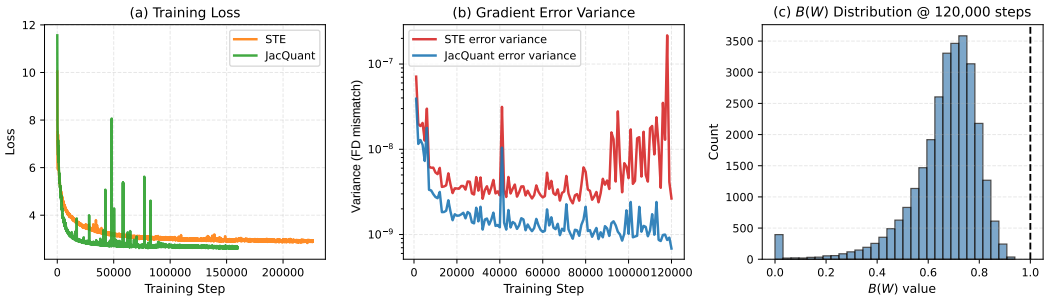

Figure 2: Training dynamics and finite-difference diagnostics. (a) **Expected gain-learning transient, not training failure.** Early loss spikes occur as $B(W)$ learns from identity; recovery to a lower loss motivates accurate, stable backward-gain estimation. (b) Variance of coordinate-wise mismatch with a finite-difference reference, which measures finite-scale response rather than a pointwise derivative. (c) Group gains at 120K steps; $b = 1$ is the identity STE.

Main QAT results. On `LLaMA3-1B`, ParetoQ +JacQuant improves PPL and mean accuracy at all three precisions (Tab. 1a). At W2A16, PPL falls from 13.5 to 12.3 (8.9%) and accuracy rises from 55.7% to 56.6%. The matched Rotation Trick result is 13.3/56.1 (PPL/Acc.). WinQ +JacQuant also improves both metrics at every tested precision, gaining 0.9, 0.5, and 0.2 accuracy points at W1, W1.58, and W2, respectively. Its W2A16 accuracy reaches 56.8%, showing that gains persist with the stronger WinQ baseline. Relative to FP16 (9.6 PPL, 58.5% accuracy), the W2A16 ParetoQ comparison recovers 30.8% of the PPL gap and 32.1% of the accuracy gap: roughly one third of the remaining gap within the same training pipeline. The benefit of backward adaptation depends on both precision and the base pipeline. ParetoQ gains 0.5, 1.0, and 0.9 accuracy points at 1, 1.58, and 2 bits, whereas WinQ gains 0.9, 0.5, and 0.2 points. Improvement across all six pairs supports learning sensitivity within the chosen quantizer and training recipe. The differing trends also motivate selecting gain settings for each pipeline and precision.

Scaling to other families and the 8B run. Tab. 1b extends the comparison to A8 on `LLaMA3-3B` and `Qwen3-1.7B`, with gains in both metrics in all four configurations. On 8B (Tab. 1c), accuracy improves by 0.56, 0.31, and 0.63 points at the matched 30K, 60K, and 90K checkpoints. At 90K, PPL decreases from 9.79 to 9.71 (0.8%). The advantage persists as both models improve, although its size varies with training. These matched checkpoints show that the benefit is sustained across the evaluated 90K-step trajectory at the larger model scale.

Task-level behavior. Across the ten smaller-model pipeline/precision pairs, 57 of 80 task scores improve, 2 tie, and 21 decrease; the three 8B checkpoints give 17 improvements and 7 decreases. Gains are distributed across tasks, with some regressions. For 1B W2A16 ParetoQ +JacQuant, ARC-e and HellaSwag gain 3.0 and 2.2 points, while BoolQ and SIQA decrease by 0.4 and 0.3. At 8B/90K, BoolQ gains 3.40 points and WinoGrande loses 1.19. These descriptive counts of rounded scores complement the mean accuracy by showing how improvements and tradeoffs are distributed across tasks.

Mechanistic diagnostic and early adaptation. For sampled coordinates, we form $\hat{J}_j^{\mathrm{FD}} = [Q_\Delta(w_j + h) - Q_\Delta(w_j - h)]/(2h)$ and compare $\hat{J}_j^{\mathrm{FD}} v_j$ with each training surrogate. JacQuant has lower coordinate-wise mismatch variance in Fig. 2(b). This measures the spread of errors relative to the quantizer response at scale h; it is neither minibatch gradient variance nor a measure of mean bias. It therefore complements, rather than replaces, the loss and downstream evaluations.

The early loss spikes are a reasonable learning transient, not evidence of training failure. In Fig. 2(a), $B(W)$ starts at identity and must learn heterogeneous quantizer responses. Early refreshes change the relative group contributions to the gradient and interact with AdamW's accumulated moments; the training loss can temporarily rise during this adjustment. The subsequent recovery to a lower loss is consistent with successful adaptation. This behavior underscores why a good sensitivity map matters: reliable gains align group scaling with the evolving quantizer response.

In Fig. 2(c), many gains differ from one: the learner has moved beyond the initial STE rule. Gains below one attenuate low-response groups at the chosen probe scale. Such groups can lie deep inside quantization cells or in saturation regions; the histogram summarizes their aggregate sensitivity.

Efficiency and ablations. The gain state requires $4\lceil N_q/g_s \rceil$ bytes in FP32: for one billion quantized weights, about 31.25 MB at $g_s = 128$ or 125 MB at $g_s = 32$. Optional anchor state and temporary buffers are additional. In the 1B runtime suite, measured throughput decreases by 1.6–5.2%; at $g_s = 128$, Probe decreases throughput by 3.5% for ParetoQ and 1.6% for WinQ. On 8B, Probe at $g_s \in \{32, 128\}$ stays within 0.8% of the STE baseline (Tabs. 7 and 8). Probe exceeds the reported Dither variant by 0.8 accuracy points at $g_s = 32, K_B = 200$ (Tab. 10).

Table 2: Group size and refresh interval on LLaMA3-1B W2A16, ParetoQ +JacQuant. Each cell pair reports PPL/Acc. The full eight-task scores are retained in Tab. 9.

	$K_B = 50$		$K_B = 100$		$K_B = 200$	
g_s	PPL	Acc.	PPL	Acc.	PPL	Acc.
8	12.6	56.6	12.6	55.7	12.5	55.7
32	12.5	55.8	12.5	56.2	**12.3**	**56.6**
128	12.5	56.1	**12.3**	56.3	12.5	56.5

Choosing a practical operating point. At $K_B = 200$, moving from $g_s = 8$ to 32 improves both PPL (12.5 to 12.3) and accuracy (55.7% to 56.6%), despite using fewer gain parameters. At $g_s = 128$, the best PPL occurs at $K_B = 100$, whereas the best accuracy occurs at 200. At the displayed precision, $g_s = 32, K_B = 200$ jointly achieves the lowest PPL and highest accuracy in the grid. The default $g_s = 128, K_B = 100$ matches this PPL with one quarter of the gain state, at a cost of 0.3 accuracy points.

Refresh cost and persistent state scale differently. Increasing K_B reduces query frequency; increasing g_s reduces gain storage. Each refresh still queries the quantized weights, so memory and runtime benefits differ. The runtime measurements and accuracy grid favor periodic estimation with moderate groups.

6 Discussion and Conclusion

JacQuant makes the quantizer's backward rule a learnable component of QAT. By fitting reusable group gains to finite quantizer responses, it adapts the backward signal while preserving the hard forward quantizer. Across the evaluated 1B–8B models, the JacQuant training configurations improve perplexity and mean zero-shot accuracy over their base pipelines with modest measured training overhead.

Training cost versus deployment cost. For the default Probe path, gains, probe buffers, and optional control-variate state are needed only during optimization. After exporting $Q_\Delta(W)$, this state can be discarded: inference uses the base pipeline's quantized forward operator, with no gain multiplication or probe queries. The measured overhead therefore concerns training; deployment speed and memory depend on the underlying quantizer, weight packing, and kernels.

Balancing sensitivity and efficiency. Group size, refresh interval, and EMA averaging provide complementary controls over gain storage, query frequency, and temporal smoothing. The ablations show that compact group gains can retain useful quantization-response information, with practical operating points that balance model quality and training cost. The early adaptation in Fig. 2(a) further motivates exploring sensitivity-adaptive refresh schedules and coordinated gain/optimizer warmup to improve tracking as training evolves.

Together, the analysis and experiments support finite-response sensitivity learning as a practical approach to ultra-low-bit LLM QAT. Future work can extend this approach to longer training at larger model scales and joint weight, activation, and KV-cache quantization. The central design principle is to learn and reuse the backward sensitivity map while retaining the deployment pipeline.

Reproducibility Statement

The gain update and training pseudocode are provided in Secs. 3 and E. Assumptions and proofs appear in Secs. 4 and F. Secs. A to D report the training recipe, per-task results, runtime measurements, and sensitivity studies. The reporting limitations are stated in Sec. G.

AI Use Statement

We used ChatGPT to improve the clarity and readability of the manuscript. The authors take full responsibility for the final content.

Ethics Statement

This work studies quantization-aware training using existing language models and datasets. Efficiency gains can broaden access to model deployment and inherit the base models' misuse, bias, privacy, and safety risks. The method does not provide a safety mitigation; use remains subject to the relevant model and data licenses and deployment evaluations.

References

Eirikur Agustsson, Fabian Mentzer, Michael Tschannen, Lukas Cavigelli, Radu Timofte, Luca Benini, and Luc V Gool. Soft-to-hard vector quantization for end-to-end learning compressible representations. *Advances in neural information processing systems*, 30, 2017.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Yash Bhalgat, Jinwon Lee, Markus Nagel, Tijmen Blankevoort, and Nojun Kwak. 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*, pp. 696–697, 2020.

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

Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I.-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018. URL https://arxiv.org/abs/1805.06085.

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

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

Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex sgd. *Advances in neural information processing systems*, 32, 2019.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014.

Vage Egiazarian, Andrei Panferov, Denis Kuznedelev, Elias Frantar, Artem Babenko, and Dan Alistarh. Extreme compression of large language models via additive quantization. *arXiv preprint arXiv:2401.06118*, 2024.

Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. Learned step size quantization. In *International Conference on Learning Representations*, 2020.

Christopher Fifty, Ronald Guenther Junkins, Dennis Duan, Aniketh Iyengar, Jerry Weihong Liu, Ehsan Amid, Sebastian Thrun, and Christopher Re. Restructuring vector quantization with the rotation trick. In *The Thirteenth International Conference on Learning Representations*, 2025.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate quantization for generative pre-trained transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

Robert M Gower, Peter Richtárik, and Francis Bach. Stochastic quasi-gradient methods: Variance reduction via jacobian sketching. *Mathematical Programming*, 188(1):135–192, 2021.

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

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems*, 26, 2013.

Dongyue Li, Zechun Liu, Kai Yi, Zhenshuo Zhang, Changsheng Zhao, Raghuraman Krishnamoorthi, Harshit Khaitan, Hongyang R. Zhang, and Steven Li. Winq: Accelerating quantization-aware training of language models around saddle points. *ICML*, 2026.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of machine learning and systems*, 6: 87–100, 2024.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025a.

Yifei Liu, Jicheng Wen, Yang Wang, Shengyu Ye, Li Lyna Zhang, Ting Cao, Cheng Li, and Mao Yang. 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*, pp. 8181–8196, 2024.

Zechun Liu, Changsheng Zhao, Hanxian Huang, Sijia Chen, Jing Zhang, Jiawei Zhao, Scott Roy, Lisa Jin, Yunyang Xiong, Yangyang Shi, Lin Xiao, Yuandong Tian, Bilge Soran, Raghuraman Krishnamoorthi, Tijmen Blankevoort, and Vikas Chandra. Paretoq: Improving scaling laws in extremely low-bit llm quantization. In *Advances in Neural Information Processing Systems*, 2025b.

Vladimir Malinovskii, Denis Mazur, Ivan Ilin, Denis Kuznedelev, Konstantin Burlachenko, Kai Yi, Dan Alistarh, and Peter Richtarik. Pv-tuning: Beyond straight-through estimation for extreme llm compression. *Advances in Neural Information Processing Systems*, 37:5074–5121, 2024.

Grigory Malinovsky, Kai Yi, and Peter Richtárik. Variance reduced proxskip: Algorithm, theory and application to federated learning. *Advances in Neural Information Processing Systems*, 35: 15176–15189, 2022.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 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 *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2381–2391, 2018.

Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International conference on machine learning*, pp. 7197–7206. PMLR, 2020.

Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Sarah: A novel method for machine learning problems using stochastic recursive gradient. In *International conference on machine learning*, pp. 2613–2621. PMLR, 2017.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849, 2024.

Herbert Robbins and David Siegmund. A convergence theorem for non negative almost supermartingales and some applications. In *Optimizing Methods in Statistics*, pp. 233–257. Academic Press, 1971. doi: 10.1016/B978-0-12-604550-5.50015-8.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4463–4473, 2019.

Qitao Tan, Xiaoying Song, Jin Lu, Guoming Li, Jun Liu, Lingzi Hong, Caiwen Ding, Jundong Li, Xiaoming Zhai, Shaoyi Huang, Wei Niu, and Geng Yuan. End-to-end on-device quantization-aware training for llms at inference cost. *arXiv preprint arXiv:2509.00031*, 2025.

Albert Tseng, Jerry Chee, Qingyao Sun, Volodymyr Kuleshov, and Christopher De Sa. Quip #: Even better llm quantization with hadamard incoherence and lattice codebooks. In *International Conference on Machine Learning*, pp. 48630–48656. PMLR, 2024.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pp. 38087–38099. PMLR, 2023.

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

Ningfeng Yang and Tor M. Aamodt. Improving the straight-through estimator with zeroth-order information. *arXiv preprint arXiv:2510.23926*, 2025.

Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in neural information processing systems*, 35:27168–27183, 2022.

Zhewei Yao, Xiaoxia Wu, Cheng Li, Stephen Youn, and Yuxiong He. Zeroquant-v2: Exploring post-training quantization in llms from comprehensive study to low rank compensation. *arXiv preprint arXiv:2303.08302*, 2023.

P Yin, J Lyu, S Zhang, S Osher, YY Qi, and J Xin. Understanding straight-through estimator in training activation quantized neural nets. In *International Conference on Learning Representations*, 2019.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, 2019.

Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016. URL https://arxiv.org/abs/1606.06160.

CONTENTS

1 Introduction **1**

2 Related Work **2**

3 Method **3**
3.1 Preliminaries and Sensitivity Groups . . . 3
3.2 Learning a Backward Sensitivity Map . . . 3
3.3 Practical Estimation of $B(W)$ . . . 4
3.4 Optional Variance Reduction . . . 4

4 Convergence Analysis of JacQuant **5**
4.1 A Finite-Scale Reference Gradient . . . 5
4.2 What the Gain Estimator Learns . . . 6
4.3 Conditional Optimization Guarantees . . . 6

5 Experiments **7**

6 Discussion and Conclusion **9**

A Detailed Experimental Setup and Baselines **14**

B Full Per-Task Results **14**

C Runtime Overhead **16**

D Ablation Studies **17**
D.1 Sensitivity Group Size and Refresh Interval . . . 17
D.2 Probe vs. Dither Jacobian Learners . . . 18

E Implementation and Configuration Semantics **18**

F Proofs and Scope of the Theoretical Results **19**
F.1 Gaussian Smoothing and the Regression Target . . . 19
F.2 Proof of Lem. 2 . . . 19
F.3 The Dither Target and Its Distinction from Gaussian Probing . . . 20
F.4 Proof of Prop. 1 and Diminishing-Rate Convergence . . . 21
F.5 Bias, Stability, and Directional Comparison . . . 21
F.6 Relation to the Smoothed Task Objective . . . 22
F.7 Conditional Bias and Variance of Control Variates . . . 22
F.8 Proof of Thm. 1 . . . 22
F.9 Proof of Thm. 2 . . . 23
F.10 Composition Across Changing Surrogates . . . 23
F.11 Interaction with SGD and AdamW . . . 24

G Limitations and Extensions **24**

H Focused Comparisons with Related Methods **24**

The supplement gives experimental settings and all per-task results, followed by runtime and ablations (Secs. A to D), implementation semantics (Sec. E), and proofs (Sec. F).

A Detailed Experimental Setup and Baselines

Models and data. We evaluate on the following LLM checkpoints: `LLaMA3-1B`, `LLaMA3-3B`, `LLaMA3-8B` (Grattafiori et al., 2024), and `Qwen3-1.7B` (Yang et al., 2025). All models are trained on `FineWebEdu` (Penedo et al., 2024) for pre-training QAT and evaluated on `WikiText-2` (Merity et al., 2017) for perplexity, plus eight standard zero-shot benchmarks including `ARC-easy/challenge` (Clark et al., 2018), `BoolQ` (Clark et al., 2019), `PIQA` (Bisk et al., 2020), `SIQA` (Sap et al., 2019), `HellaSwag` (Zellers et al., 2019), `OBQA` (Mihaylov et al., 2018), and `WinoGrande` (Sakaguchi et al., 2020).

Quantization configuration. We apply symmetric uniform quantization with per-channel scaling to all linear layers (attention Q/K/V projections, MLP up/down projections, output head). Bit-widths range from 1 to 2 bits for weights (`W1`, `W1.58`, `W2`) and 8 to 16 bits for activations (`A8`, `A16`). The default sensitivity group contains $g_s = 128$ weights; the reported runtime and accuracy sweeps include $g_s \in \{8, 32, 128\}$. Sensitivity groups share a backward gain, not necessarily a forward quantization scale. W1.58 denotes three quantization levels ($\log_2 3 \approx 1.585$ nominal bits), not a measured packed model size.

Training Details. Within each base-pipeline pair, the forward quantizer and scale/clipping recipe are retained. ParetoQ and WinQ have different training mechanisms; optional VR and dither choices must also be distinguished from the backward gain. We train with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\varepsilon = 10^{-8}$), learning rate $\eta = 2 \times 10^{-5}$, constant schedule with 1k warmup steps, weight decay 0, batch size 8 per device, and gradient accumulation 1. Training runs for 240k steps with evaluation every 5k steps. For `LLaMA3-8B`, we report intermediate checkpoints at 30K/60K/90K due to the substantially higher end-to-end QAT cost. The default gain refresh occurs every $K_B = 100$ steps and EMA coefficient $\beta = 0.9$. In (3), β weights the new estimate.

The hyperparameter search considers both the probe-based (JacQuant-Probe) and dither-based (JacQuant-Dither) variants, exploring probe noise levels $\sigma \in \{10^{-4}, 10^{-3}, 10^{-2}\}$ (Gaussian queries only), EMA coefficients $\beta \in \{0.8, 0.9, 0.95\}$, update intervals $K_B \in \{50, 100, 200\}$, and group sizes $g_s \in \{32, 64, 128, 256\}$. Settings are selected by WikiText-2 validation perplexity, so the reported PPL is a model-selection metric. Zero-shot accuracy provides a complementary downstream outcome. The listed search space denotes candidate settings, not a count of completed runs.

Computational Overhead. For N_q quantized weights, FP32 gains occupy $4\lceil N_q/g_s \rceil$ bytes. Relative to two FP32 Adam moment tensors, this is approximately $1/(2g_s)$: 0.39% at $g_s = 128$ and 1.56% at $g_s = 32$. At eight billion quantized weights, the gain-only state is approximately 250 MB or 1 GB, respectively. This accounting excludes master weights, temporary probes, gradients, and optional VR state. Runtime is reported directly in Tabs. 7 and 8; measured 1B throughput reductions range from 1.6% to 5.2%.

Baselines The paired training baselines are ParetoQ (Liu et al., 2025b) and WinQ (Li et al., 2026). ParetoQ provides a low-bit training/quantizer recipe; WinQ uses weight interpolation and noise-injected gradients to improve QAT optimization. The main table also includes Rotation Trick under ParetoQ. RTN, GPTQ, and SpinQuant are retained as contextual PTQ results in the detailed tables; they are not matched-training controls for the backward rule.

The main JacQuant results are VR-enabled and evaluate the combined training configurations; Sec. E states the estimator choices and attribution limit.

B Full Per-Task Results

Tab. 1 summarizes these results. Below, each row gives WikiText-2 PPL (↓), eight zero-shot task accuracies, and their reported mean (%, ↑). Models are evaluated without task-specific fine-tuning. Bold compares each JacQuant row with its own base pipeline.

Table 3: LLaMA3-1B, A16: full per-task results.

Method	**Wiki2** (↓)	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	Wino.	**Avg. Acc.** (↑)
FP16 Baseline	9.6	64.8	42.5	64.8	74.8	44.8	64.4	50.2	61.5	58.5
`LLaMA3-1B, W1A16`										
RTN	4.2e8	25.0	22.5	37.6	49.5	32.9	25.0	27.1	49.6	33.7
GPTQ	3.3e8	26.9	21.7	37.6	51.8	33.5	25.5	14.8	49.7	32.7
SpinQuant	2.4e8	25.0	22.5	37.6	49.5	32.9	25.0	27.1	49.6	33.7
ParetoQ	17.9	58.7	39.8	62.6	67.4	42.4	46.9	43.9	53.5	51.9
+JacQuant	**17.6**	60.6	38.9	64.1	67.2	42.6	47.8	44.1	53.9	**52.4**
WinQ	15.3	59.7	37.0	61.3	69.5	42.6	49.8	46.1	54.4	52.6
+JacQuant	**15.1**	61.1	37.9	63.1	70.2	43.2	50.9	44.9	56.3	**53.5**
`LLaMA3-1B, W1.58A16`										
RTN	1.8e6	24.5	22.3	62.4	52.7	33.4	25.4	18.4	50.2	36.2
GPTQ	7.5e4	24.8	22.2	38.0	52.2	32.5	25.4	17.0	49.4	32.7
SpinQuant	5.8e3	25.3	22.5	37.6	51.6	33.3	25.3	17.6	48.5	32.7
ParetoQ	14.8	61.4	39.7	63.1	71.0	42.6	53.1	47.5	55.7	54.3
+JacQuant	**13.9**	64.2	38.2	62.2	71.6	44.4	55.0	47.7	59.1	**55.3**
WinQ	12.9	65.0	39.7	62.1	72.9	44.3	56.2	47.1	57.4	55.6
+JacQuant	**12.3**	66.1	39.2	63.4	73.6	44.8	56.9	47.5	57.1	**56.1**
`LLaMA3-1B, W2A16`										
RTN	1.5e6	24.5	23.1	62.4	52.3	33.6	25.4	17.6	50.3	36.2
GPTQ	3.8e4	26.9	21.7	37.6	51.8	33.5	25.5	14.8	49.7	32.7
SpinQuant	3.8e2	31.0	20.1	45.5	54.6	33.8	26.7	16.2	50.9	34.9
ParetoQ	13.5	63.1	41.9	62.5	72.8	43.6	55.2	49.2	57.3	55.7
+JacQuant	**12.3**	66.1	42.0	62.1	72.8	43.3	57.4	49.8	59.1	**56.6**
WinQ	11.9	65.0	42.5	62.5	73.8	43.2	58.4	48.1	59.1	56.6
+JacQuant	**11.8**	66.4	39.8	64.7	73.3	43.8	59.2	47.8	59.4	**56.8**

Table 4: LLaMA3-3B, A8: full per-task results.

Method	**Wiki2** (↓)	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	Wino.	**Avg. Acc.** (↑)
FP16 Baseline	7.7	72.6	50.7	74.6	78.2	48.5	74.3	53.7	69.2	65.2
`LLaMA3-3B, W1A8`										
RTN	7.3e7	25.0	22.5	37.6	49.5	32.9	25.0	27.1	49.6	33.7
GPTQ	5.9e7	26.6	21.9	37.6	52.5	33.7	25.6	15.0	49.4	32.8
SpinQuant	4.5e7	27.0	22.0	37.6	52.0	33.4	25.5	14.5	50.2	32.8
ParetoQ	15.7	63.1	39.1	63.0	70.9	42.6	50.6	46.7	56.5	54.1
+JacQuant	**15.2**	64.3	38.9	60.6	72.2	43.2	50.7	49.8	59.1	**54.9**
`LLaMA3-3B, W1.58A8`										
RTN	7.9e5	24.7	23.4	37.6	52.9	33.8	25.5	17.4	50.3	33.2
GPTQ	2.7e5	25.3	22.9	37.6	53.7	34.4	25.3	15.6	49.5	33.0
SpinQuant	3.1e3	26.6	23.4	39.1	52.7	34.7	25.3	16.4	48.5	33.3
ParetoQ	13.1	67.9	42.0	53.9	72.0	43.9	58.2	49.6	60.2	56.0
+JacQuant	**12.7**	67.3	42.0	57.5	73.4	43.8	59.1	51.8	61.3	**57.1**

Table 5: Qwen3-1.7B, A8: full per-task results.

Method	**Wiki2 (↓)**	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	Wino.	**Avg. Acc. (↑)**
FP16 Baseline	16.2	68.9	41.0	78.9	71.7	45.1	59.6	38.5	61.6	58.2
`Qwen3-1.7B, W1A8`										
ParetoQ	46.5	42.0	26.2	61.4	59.8	40.0	29.1	28.9	50.8	42.3
+JacQuant	**45.3**	42.5	26.5	61.8	59.6	40.6	29.5	29.4	51.7	**42.7**
`Qwen3-1.7B, W2A8`										
ParetoQ	22.2	52.9	32.6	62.4	65.1	42.7	41.2	32.8	53.1	47.9
+JacQuant	**21.5**	54.1	31.1	64.4	64.1	43.5	40.3	34.1	52.8	**48.1**

Table 6: LLaMA3-8B, W2A16: results at matched training checkpoints.

Method	**Wiki2 (↓)**	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	Wino.	**Avg. Acc. (↑)**
`30K`										
ParetoQ	11.78	70.79	42.54	67.37	74.80	45.41	66.12	37.89	63.39	58.54
+JacQuant	**11.69**	69.65	44.95	67.55	74.88	44.86	66.65	39.94	64.32	**59.10**
`60K`										
ParetoQ	10.43	72.50	46.58	69.30	75.56	46.88	69.09	40.72	64.70	60.67
+JacQuant	**10.32**	72.74	46.21	70.73	75.42	46.92	68.91	41.12	65.80	**60.98**
`90K`										
ParetoQ	9.79	75.60	48.36	66.91	76.79	47.13	69.36	40.03	67.07	61.41
+JacQuant	**9.71**	75.48	49.42	70.31	76.93	47.53	69.44	41.31	65.88	**62.04**

C Runtime Overhead

Table 7 reports end-to-end training throughput (steps/second) for `LLaMA3-1B` W2A16 when augmenting two strong QAT baselines (ParetoQ and WinQ) with our Jacobian-learning backpropagation (JacQuant-Probe/ JacQuant-Dither).

Group size and throughput. Larger g_s reduces gain-state traffic, while end-to-end throughput also reflects the base pipeline and refresh computation. WinQ+Probe increases from 1.7223 steps/s at $g_s = 8$ to 1.7447 at 128. WinQ+Dither gives similar measured throughput at 32 and 128: 1.7411 versus 1.7308 steps/s, a difference of about 0.6%.

Measured overhead. Relative throughput reductions range from 1.6% to 5.2% in this 1B W2A16 suite. At $g_s = 128$, Probe has reductions of 1.6% with WinQ and 3.5% with ParetoQ. Throughput reduction is $100(1 - r/r_0)$; the corresponding step-time increase is $100(r_0/r - 1)$, so the two percentages should not be interchanged. These measurements describe the reported training configurations; they are not an isolated measurement of the gain-refresh kernel.

Table 7: **Training throughput** (steps/s; higher is better) for `LLaMA3-1B` W2A16 QAT. g_s is the number of weights that share one backward-gain entry. Parentheses report relative throughput change vs. STE.

Base	STE	JacQuant-Probe (g_s)			JacQuant-Dither (g_s)		
		8	32	128	8	32	128
ParetoQ Liu et al. (2025b)	1.8074	1.7391 (-3.8%)	1.7135 (-5.2%)	1.7435 (-3.5%)	1.7194 (-4.9%)	1.7347 (-4.0%)	1.7334 (-4.1%)
WinQ Li et al. (2026)	1.7738	1.7223 (-2.9%)	1.7392 (-2.0%)	1.7447 (-1.6%)	1.7229 (-2.9%)	1.7411 (-1.8%)	1.7308 (-2.4%)

Runtime on `LLaMA3-8B`. Tab. 8 gives raw step times for `LLaMA3-8B` under ParetoQ W2A16. The STE baseline is 1297.01 ms/step. Probe adds 10.17 ms at $g_s = 32$ and 9.98 ms at $g_s = 128$, corresponding to throughput reductions of 0.78% and 0.76%. Dither also stays close to baseline throughput: the largest reduction is 1.62%, with a measured change of $+0.29\%$ at $g_s = 32$.

Table 8: **`LLaMA3-8B` runtime under ParetoQ W2A16.** Step time is wall-clock milliseconds per training step; steps/s is recomputed from step time. The final column gives the relative throughput change from STE.

Configuration	Step time (ms)	Steps/s	Relative to STE
ParetoQ STE baseline	1297.01	0.7710	–
+JacQuant-Probe, $g_s = 8$	1335.81	0.7486	-2.90%
+JacQuant-Probe, $g_s = 32$	1307.18	0.7650	-0.78%
+JacQuant-Probe, $g_s = 128$	1306.99	0.7651	-0.76%
+JacQuant-Dither, $g_s = 8$	1300.75	0.7688	-0.29%
+JacQuant-Dither, $g_s = 32$	1293.20	0.7733	+0.29%
+JacQuant-Dither, $g_s = 128$	1318.38	0.7585	-1.62%

D Ablation Studies

Setup. The ablations use LLaMA3-1B W2A16 with ParetoQ +JacQuant and report the same PPL and eight-task metrics as the main text. The reported Probe noise is $\sigma = 10^{-4}$. The tables vary the deterministic gain-refresh interval K_B over 50/100/200 steps. Gain refresh and any separately scheduled anchor refresh are distinct operations.

D.1 Sensitivity Group Size and Refresh Interval

Group size g_s controls how many weights share a gain; the interval K_B controls how often that gain changes. We vary them independently to test representation granularity and refresh frequency.

Table 9: Sensitivity to group size g_s and gain-refresh interval K_B (steps), LLaMA3-1B W2A16 with ParetoQ +JacQuant.

g_s	K_B	**Wiki2** (↓)	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	Wino.	**Avg. Acc.** (↑)
8	50	12.6	65.2	41.0	63.3	73.9	44.8	56.8	50.4	57.3	56.6
8	100	12.6	64.9	40.0	60.2	73.2	43.8	56.8	47.7	58.9	55.7
8	200	12.5	65.5	38.4	61.0	73.1	43.1	57.2	49.2	58.0	55.7
32	50	12.5	64.2	42.3	61.2	72.1	43.6	57.4	49.4	56.3	55.8
32	100	12.5	64.8	40.5	61.9	72.6	43.9	56.8	51.4	57.8	56.2
32	200	**12.3**	66.0	42.0	62.1	72.8	43.3	57.4	49.8	59.1	**56.6**
128	50	12.5	65.3	41.2	62.1	73.1	42.7	57.3	49.4	57.6	56.1
128	100	12.3	64.4	41.7	62.6	73.1	43.3	57.0	50.2	58.5	56.3
128	200	12.5	65.1	42.8	62.5	74.0	43.4	57.4	48.4	58.4	56.5

Findings. **(1) Several settings are competitive.** The grid spans 12.3–12.6 PPL and 55.7–56.6% mean accuracy. The $g_s = 32, K_B = 200$ configuration combines the lowest reported PPL with a highest reported mean accuracy, but $g_s = 128, K_B = 100$ also reaches 12.3 PPL and $g_s = 8, K_B = 50$ also reaches 56.6% accuracy. The table supports a range of useful choices, rather than a unique optimal group size.

(2) Less frequent refresh can help, but the trend depends on the metric. At $g_s = 32$, accuracy improves from 55.8% to 56.6% as the interval increases from 50 to 200 steps. At $g_s = 128$, accuracy also increases, while PPL is 12.5, 12.3, and 12.5. Thus the PPL trend is not monotonic. Refresh cost, target drift, and estimator noise offer a plausible tradeoff, but the table does not isolate their individual effects.

D.2 Probe vs. Dither Jacobian Learners

The reported Probe–Dither comparison uses $g_s = 32$ and $K_B = 200$. Probe uses Gaussian response queries; Dither uses the dithered-forward variant described in Sec. E.

Table 10: Probe and Dither on LLaMA3-1B W2A16, $g_s = 32$, $K_B = 200$.

Method	**Wiki2** (↓)	ARC-e	ARC-c	BoolQ	PIQA	SIQA	HellaS.	OBQA	Wino.	**Avg. Acc.** (↑)
JacQuant-Dither	12.5	64.2	42.3	61.2	72.1	43.6	57.4	49.4	56.3	55.8
JacQuant-Probe	**12.3**	66.0	42.0	62.1	72.8	43.3	57.4	49.8	59.1	**56.6**

Probe improves PPL by 0.2 and mean accuracy by 0.8 points in this comparison, motivating its use as the default learner. Because Dither changes the task forward distribution as well as the gain-estimation mechanism, the result does not isolate the estimator alone.

E Implementation and Configuration Semantics

The core Probe procedure is fully specified by Alg. 1. Optional control variates change the gradient estimator; Dither can also change the task forward distribution. This section states these choices separately so that their state and cost are not attributed to the gain learner.

Probe: one scalar fit and one EMA update. For each sensitivity group, form $A_g = \langle Q_\Delta(W_g + \delta_g) - Q_\Delta(W_g), \delta_g \rangle$ and $D_g = \|\delta_g\|^2$ using the same frozen scales and thresholds for both queries. Compute $s_g = \Pi_{[0,1]}(A_g/(D_g + \epsilon))$ and apply $b_g \leftarrow (1-\beta)b_g + \beta s_g$ once. This is a scalar group fit, not a coordinate-wise ratio and not an EMA of an already averaged gain. The updated gain is held constant during subsequent task differentiation. Chunking groups limits temporary probe storage; the persistent gain state remains $4G$ bytes in FP32.

Base estimator. Without a control variate, use $g_t = F_{S_t}(W_t, B_t)$. Uniform minibatch sampling makes this conditionally unbiased for $B_t \bar{v}_t$, the current mean surrogate direction. It does not make that direction the derivative of the hard objective. No state beyond the gains and the base optimizer is required.

Anchor-based (L-SVRG-style) estimator. Equation (4) uses one current and one anchor evaluation on the same minibatch. The anchor freezes the weights, gains, quantizer scales/clipping parameters, and any other trainable state needed by its forward map. For the exact finite-sum reference, set $\widetilde{g} = n^{-1} \sum_i F_i(\widetilde{W}, \widetilde{B})$; for a reference batch, use the mean over S_{ref}. An independently sampled batch reused between refreshes leaves conditional bias $\widetilde{g} - \overline{F}(\widetilde{W}, \widetilde{B})$, as shown in Lem. 6. In a loopless schedule, refresh the complete anchor with probability p_A after a task update, using the current gain state. This anchor schedule is separate from the gain interval K_B.

The anchor-based design uses model-sized state rather than a dataset-sized table, but still evaluates the anchor on each task minibatch. Sparse anchor refresh amortizes the reference calculation, not this per-step evaluation. Its wall-clock cost therefore depends on which parameters are covered, reference-batch size, and how the anchor evaluation is implemented. Gain-only memory and query costs exclude this additional work.

SAGA-style alternative. For a finite indexed dataset, store Y_i and the exact mean $\bar{Y} = n^{-1} \sum_i Y_i$, then use $g_t = \bar{Y} + |S_t|^{-1} \sum_{i \in S_t} (F_i(W_t, B_t) - Y_i)$. Form this estimator from the old table before replacing sampled entries and adding their changes divided by n to $\bar{Y}$. The estimator is conditionally unbiased for the current mean surrogate direction under uniform sampling. A full table costs $O(nN_q)$ numbers, so this literal formulation is not a low-memory LLM implementation. A compressed or sampled table requires its own estimator specification; it cannot inherit the full-table guarantee automatically.

Dither: forward map and sensitivity target. The dithered-forward variant samples $r \sim \mathcal{U}[-\Delta/2, \Delta/2]$ and feeds $q_d = Q_\Delta(W + r) - r$ to the task. The interval is set by the quantization step Δ, not by the Gaussian Probe scale σ. Its gain update has the form $b_g \leftarrow (1-\beta)b_g + \beta\phi_g(W; r)$; the mean of the actual clipped statistic ϕ_g determines the gain target. The uniform-dither identity in Sec. F.3 supplies an exact reference sensitivity; relating a particular update to that reference requires an explicit choice of ϕ_g. A query-only dither that retains $Q_\Delta(W)$ for the task is a different variant. The reported Probe–Dither comparison therefore concerns the complete variants.

Evaluated configurations. The main JacQuant configurations are reported as VR-enabled, so their results assess gain learning and control variates together. A matched four-way comparison of STE/JacQuant with and without the same control variate would isolate their respective contributions. The anchor-based and SAGA-style designs have distinct state and computation costs; runtime attribution depends on the implemented configuration.

F Proofs and Scope of the Theoretical Results

All vector norms below are Euclidean and matrix norms are operator norms unless otherwise specified. The quantizer is fixed during a sensitivity query. We assume a clipped, coordinate-separable scalar quantizer when identifying scalar group targets. The loss as a function of dequantized parameters is denoted by $\mathcal{L}$, avoiding confusion with the model $f(x; q)$. A local window fixes the unperturbed code q; probes may leave its cell.

F.1 Gaussian Smoothing and the Regression Target

Lemma 4 (Gaussian score identity)**.** *Let $Q : \mathbb{R}^d \to \mathbb{R}^d$ be measurable and bounded, and $\delta \sim \mathcal{N}(0, \sigma^2 I)$, $\sigma > 0$. Then $m_\sigma(W) = \mathbb{E}[Q(W + \delta)]$ is smooth, and*

$$Dm_\sigma(W) = \frac{1}{\sigma^2}\mathbb{E}[Q(W+\delta)\delta^\top] = \frac{1}{\sigma^2}\mathbb{E}[(Q(W+\delta) - Q(W))\delta^\top]. \tag{17}$$

Proof. Write $m_\sigma(W) = \int Q(z)\varphi_\sigma(z - W)\, dz$, where φ_σ is the Gaussian density. Differentiating the density gives $\nabla_W \varphi_\sigma(z - W) = \frac{z-W}{\sigma^2}\varphi_\sigma(z - W)$. The boundedness of Q and integrability of Gaussian density derivatives justify differentiating under the integral locally in W. Substituting $z = W + \delta$ gives the first identity. The second uses $\mathbb{E}[\delta] = 0$. Higher density derivatives establish smoothness without differentiating the discontinuous quantizer. □

Threshold interpretation. A scalar quantizer with levels $a_0, \ldots, a_M$ and thresholds $\tau_1, \ldots, \tau_M$ can be written $Q(w) = a_0 + \sum_{j=1}^{M}(a_j - a_{j-1})\mathbf{1}\{w \geq \tau_j\}$. Consequently,

$$J_\sigma(w) = \sum_{j=1}^{M} \frac{a_j - a_{j-1}}{\sigma}\varphi\left(\frac{w - \tau_j}{\sigma}\right), \tag{18}$$

where φ is the standard Gaussian density. For a monotone quantizer, $J_\sigma(w) \geq 0$, but it need not be at most one. Near an isolated threshold with jump Δ, $J_\sigma(\tau) \approx \Delta/(\sqrt{2\pi}\sigma)$; far from every threshold it can be very small. Thus clipping gains to $[0, 1]$ is a deliberate stability constraint with approximation cost. Identity sensitivity throughout bin interiors applies to a suitable uniform-dither mean map, not to arbitrary Gaussian probe scales.

F.2 Proof of Lem. 2

Proof. For a fixed group, abbreviate $A = \langle \Delta q_g, \delta_g \rangle$ and $D = \|\delta_g\|^2$. The population scalar regression loss expands as $R(b) = \mathbb{E}[\|\Delta q_g\|^2] - 2b\,\mathbb{E}[A] + b^2\,\mathbb{E}[D]$. Since $\mathbb{E}[D] = d_g\sigma^2 > 0$, this is strictly convex and its unique unconstrained minimizer is $\mathbb{E}[A]/(d_g\sigma^2)$. Taking the trace in (17) gives (9). The constrained minimizer is its scalar projection onto $[0, 1]$.

For independent samples (A_k, D_k), the empirical minimizer is $\widehat{b}_m = \frac{\sum_{k=1}^m A_k}{\sum_{k=1}^m D_k}$. Bounded quantizer outputs and Gaussian moments imply $\mathbb{E}|A| < \infty$ and $\mathbb{E}D < \infty$. The strong law of large numbers

applies to numerator and denominator separately, so $\widehat{b}_m \to b_g^{\mathrm{LS}}$ almost surely. The denominator limit is strictly positive. Continuity of scalar clipping also gives $\Pi_{[0,1]}(\widehat{b}_m) \to b_g^\star$. □

A finite-sample bound with explicit assumptions. Let $\nu_g^2 = \mathrm{Var}(A - b_g^{\mathrm{LS}} D) < \infty$. For any $\alpha \in (0,1)$,

$$\Pr\left(|\widehat{b}_m - b_g^{\mathrm{LS}}| \le \frac{2\nu_g}{d_g \sigma^2 \sqrt{m\alpha}} \right) \ge 1 - \alpha - \exp(-m d_g/16). \tag{19}$$

The same bound holds between the scalar-clipped estimate and $b_g^\star$. The bound applies to the scalar group class used by the algorithm.

Proof. The centered variable $Z = A - b_g^{\mathrm{LS}} D$ has mean zero. By Chebyshev's inequality, $|\frac{1}{m}\sum_k Z_k| \le \nu_g/\sqrt{m\alpha}$ with probability at least $1-\alpha$. Also $\sum_k D_k/\sigma^2 \sim \chi^2_{md_g}$. Its lower-tail bound gives $\Pr(\frac{1}{m}\sum_k D_k < d_g\sigma^2/2) \le \exp(-md_g/16)$. On the intersection of these events,

$$|\widehat{b}_m - b_g^{\mathrm{LS}}| = \frac{|\frac{1}{m}\sum_k Z_k|}{\frac{1}{m}\sum_k D_k} \le \frac{2\nu_g}{d_g\sigma^2\sqrt{m\alpha}}.$$

A union bound proves the claim; scalar clipping is nonexpansive. □

Practical EMA versus pooled regression. The pooled ratio differs from $\frac{1}{m}\sum_k \Pi_{[0,1]}(A_k/(D_k + \epsilon))$. The latter converges to $\bar{b}_g(W)$ in (10). Even without clipping, in general $\mathbb{E}[A/(D+\epsilon)] \neq \mathbb{E}[A]/\mathbb{E}[D]$. A fixed regularizer changes the target of pooled ridge regression to $\mathbb{E}[A]/(\mathbb{E}[D] + \epsilon)$ when the ridge term is $m\epsilon b^2$. These effects are included in the estimator-discrepancy term.

Group approximation. For a diagonal $J_{\sigma,g} = \mathrm{diag}(j_1, \ldots, j_{d_g})$, minimizing $\|bI - J_{\sigma,g}\|_F^2$ gives $d_g^{-1}\sum_j j_j$. However, the gain minimizing the directional discrepancy $\|(bI - J_{\sigma,g})v_g\|^2$ is $b_g^{\mathrm{dir}} = \frac{\sum_j j_j v_{g,j}^2}{\sum_j v_{g,j}^2}$ when $v_g \neq 0$, followed by clipping if constrained. The quantizer-only estimator does not use these loss-dependent weights. Finer grouping can reduce heterogeneity, but does not guarantee a better trained model or remove finite-probe noise.

F.3 The Dither Target and Its Distinction from Gaussian Probing

Lemma 5 (Uniform-dither mean sensitivity)**.** *For scalar $r \sim \mathcal{U}[-h,h]$ and locally integrable Q, define $m_h^{\mathrm{U}}(w) = \mathbb{E}_r[Q(w+r) - r]$. Then*

$$m_h^{\mathrm{U}}(w) = \frac{1}{2h}\int_{w-h}^{w+h} Q(z)\,dz, \qquad J_h^{\mathrm{U}}(w) = \frac{Q(w+h) - Q(w-h)}{2h} \tag{20}$$

almost everywhere. For an unclipped uniform quantizer of step Δ with $h = \Delta/2$, $J_h^{\mathrm{U}} = 1$ almost everywhere. Clipping can reduce this sensitivity near saturation.

Proof. The dither has mean zero, yielding the first equality after a change of variables. The fundamental theorem for integrals of locally integrable functions gives the endpoint formula almost everywhere. Away from tie points, an unclipped uniform quantizer satisfies $Q(w+\Delta/2) - Q(w-\Delta/2) = \Delta$. With equally spaced clipped levels, a length-Δ interval crosses at most one active threshold almost everywhere, so the derivative is zero or one. Boundary points need not have a classical derivative. □

Finite differences after dithering. If independent Gaussian probes are applied to the dithered map using the same r in both queries, the cross moment is

$$\frac{1}{\sigma^2}\mathbb{E}_{\delta,r}[(Q(W+\delta+r) - Q(W+r))\delta^\top] = D_W\,\mathbb{E}_\delta[m_h^{\mathrm{U}}(W+\delta)].$$

This is a Gaussian-smoothed uniform-dither target, not generally $J_h^{\mathrm{U}}(W)$ at finite σ. The Gaussian score identity applies to the bounded dithered map; the discontinuous quantizer is not differentiated pointwise.

The gain statistic used in a Dither implementation must have the stated mean for this identity to describe its fixed point. A dithered task forward also changes the expected task loss (Sec. E).

F.4 Proof of Prop. 1 and Diminishing-Rate Convergence

Proof. At fixed W, write $e_k = b_k - \bar{b}$ and $\xi_k = s_k - \bar{b}$. The recursion is $e_{k+1} = (1-\beta)e_k + \beta\xi_k$. Conditional centering eliminates the cross term:

$$\mathbb{E}[\|e_{k+1}\|^2 \mid \mathcal{F}_k] = (1-\beta)^2\|e_k\|^2 + \beta^2\mathbb{E}[\|\xi_k\|^2 \mid \mathcal{F}_k].$$

Iterating and summing the geometric series yields

$$\mathbb{E}[\|e_k\|^2] \le (1-\beta)^{2k}\|e_0\|^2 + \frac{\beta}{2-\beta}\big(1-(1-\beta)^{2k}\big)V_B,$$

which implies (12).

For moving weights, let $\bar{b}_k = \mathbb{E}[s_k \mid \mathcal{F}_k]$ and $d_k = \bar{b}_{k+1} - \bar{b}_k$. Then

$$e_{k+1} = (1-\beta)e_k + \beta\xi_k - d_k. \tag{21}$$

Unrolling separates the initial condition, a weighted sum of martingale differences, and a weighted drift sum. If $\mathbb{E}[\|\xi_k\|^2 \mid \mathcal{F}_k] \le V_B$ and $(\mathbb{E}[\|d_k\|^2])^{1/2} \le D_B$, orthogonality of martingale differences and the L^2 triangle inequality give

$$(\mathbb{E}[\|e_k\|^2])^{1/2} \le (1-\beta)^k(\mathbb{E}[\|e_0\|^2])^{1/2} + \sqrt{\frac{\beta V_B}{2-\beta}} + \sum_{j=0}^{k-1}(1-\beta)^j D_B.$$

The last sum is at most D_B/β. No independence between target drift and gain noise is required for this triangle bound. □

Proposition 2 (Fixed-weight diminishing-rate convergence)**.** *Fix W. Suppose $\mathbb{E}[s_k \mid \mathcal{F}_k] = \bar{b}(W)$ and the conditional second moments are bounded. If $0 < \beta_k \le 1$, $\sum_k \beta_k = \infty$, and $\sum_k \beta_k^2 < \infty$, then the EMA converges to $\bar{b}(W)$ almost surely.*

Proof. The conditional second-moment identity gives $\mathbb{E}[\|e_{k+1}\|^2 \mid \mathcal{F}_k] \le (1-\beta_k)\|e_k\|^2 + \beta_k^2 V_B$. The almost-supermartingale convergence lemma implies that $\|e_k\|^2$ has an almost-sure limit and $\sum_k \beta_k\|e_k\|^2 < \infty$ (Robbins & Siegmund, 1971). Since $\sum_k \beta_k = \infty$, the limit must be zero. This establishes convergence to the implemented estimator's mean; it does not identify that mean with J_σ or extend the result to constant β. □

Intermittent refresh. Here k counts sensitivity refreshes, rather than task-training steps. A larger K_B reduces refresh cost but may increase D_B, the target change between observations. The bound formalizes this tradeoff without assuming that a fixed code assignment implies fixed W or a fixed Gaussian sensitivity.

F.5 Bias, Stability, and Directional Comparison

Proof of Lems. 1 and 3. Equation (7) gives $g_B - g^\dagger = (B - J_\sigma^\top)\bar{v}$ and $g^{\mathsf{STE}} - g^\dagger = (I - J_\sigma^\top)\bar{v}$. Applying $\|Av\| \le \|A\|_{\mathrm{op}}\|v\|$ proves the bias bounds. The same inequality gives $\|Bv_i\| \le \beta_B\|v_i\|$. A block-scalar matrix with gains in $[0,1]$ has operator norm at most one. □

Corollary 1 (Directional bias comparison)**.** *For diagonal J and B, the learned rule has no larger target-gradient error for a particular v precisely when*

$$\sum_j \big[(1-J_{jj})^2 - (B_{jj}-J_{jj})^2\big]v_j^2 \ge 0. \tag{22}$$

In particular, $|B_{jj} - J_{jj}| \le |1 - J_{jj}|$ for every coordinate is sufficient for all v.

Proof. Subtract $\|(B-J)v\|^2$ from $\|(I-J)v\|^2$ and expand coordinatewise. □

Why an operator-norm comparison alone is insufficient. Take $J = \mathrm{diag}(0,1)$, $B = \mathrm{diag}(0,1/2)$, and $v = (0,1)^\top$. Then $\|B-J\|_{\mathrm{op}} = 1/2 < 1 = \|I-J\|_{\mathrm{op}}$, while STE has zero directional error and the learned rule has error $1/2$. Thus the two upper bounds in Lem. 1 cannot be replaced by equalities to prove unconditional dominance.

Regularity and the objective. For fixed q and $v_i^{(q)}$, a locally Lipschitz deterministic B gives $\|F_i(W) - F_i(W')\| \le L_B\|W - W'\|\|v_i^{(q)}\|$. This establishes regularity of the field but not the cross-partial symmetry needed for a potential. The optimization proof therefore uses the explicit $\widetilde{L}_q$ in (7) and bounds the error of g_t relative to its gradient.

F.6 Relation to the Smoothed Task Objective

Let $F_\sigma(W) = \mathbb{E}_\delta[\mathcal{L}(Q(W+\delta))]$ and fix the reference code q. Suppose $\mathcal{L}$ has an $L_{\mathcal{L}}$-Lipschitz gradient on every segment from q to $Q(W+\delta)$. Then

$$|F_\sigma(W) - \widetilde{L}_q(W)| \le \frac{L_{\mathcal{L}}}{2}\mathbb{E}[\|Q(W+\delta) - q\|^2], \tag{23}$$

$$\|\nabla F_\sigma(W) - \nabla\widetilde{L}_q(W)\| \le \frac{L_{\mathcal{L}}}{2\sigma^2}\mathbb{E}[\|Q(W+\delta) - q\|^2\|\delta\|]. \tag{24}$$

Proof. Taylor's inequality at q bounds

$$R_q(z) = \mathcal{L}(Q(z)) - \mathcal{L}(q) - \langle\nabla\mathcal{L}(q), Q(z) - q\rangle$$

by $|R_q(z)| \le L_{\mathcal{L}}\|Q(z) - q\|^2/2$. Taking expectations proves the value bound. The Gaussian score identity applied to the scalar residual yields $\nabla(F_\sigma - \widetilde{L}_q) = \sigma^{-2}\mathbb{E}[R_q(W+\delta)\delta]$. Its norm is bounded by the expected norm of its integrand. The fixed reference q is not differentiated. Bounded quantizer outputs and the smoothness conditions ensure integrability. □

These inequalities expose the additional error needed to interpret the local target as an approximation to the full smoothed task gradient. Small Jacobian-tracking error alone does not control task-linearization error. Neither bound identifies the gradient of the discontinuous hard objective.

F.7 Conditional Bias and Variance of Control Variates

Lemma 6 (Exact oracle decomposition)**.** *Let $a_t = \nabla\widetilde{L}_q(W_t)$, $e_t = \mathbb{E}[g_t \mid \mathcal{F}_t] - a_t$, and $\xi_t = g_t - \mathbb{E}[g_t \mid \mathcal{F}_t]$. Then*

$$\mathbb{E}[\|g_t - a_t\|^2 \mid \mathcal{F}_t] = \|e_t\|^2 + \mathbb{E}[\|\xi_t\|^2 \mid \mathcal{F}_t].$$

For an SVRG-style estimator $g_t = F_{S_t}(W_t, B_t) - F_{S_t}(\widetilde{W}, \widetilde{B}) + \widetilde{g}$ with uniformly sampled S_t and an anchor fixed before sampling,

$$e_t = (B_t - J_\sigma(W_t)^\top)\bar{v}^{(q)} + \widetilde{g} - \overline{F}(\widetilde{W}, \widetilde{B}).$$

Proof. The first equality follows by expanding the square and using $\mathbb{E}[\xi_t \mid \mathcal{F}_t] = 0$. For the second, take the conditional expectation of the three terms in g_t. The current mean is $B_t\bar{v}^{(q)}$ on the window, and the expected anchor minibatch equals its population mean $\overline{F}(\widetilde{W}, \widetilde{B})$. □

A reference gradient estimated once and then reused is part of $\mathcal{F}_t$; its realized error is conditional bias. It cannot automatically be absorbed into fresh zero-mean minibatch noise. An SVRG difference can have smaller variance when current and anchor gradients are sufficiently correlated, but an arbitrary stale anchor does not guarantee a universal factor $\rho < 1$. The descent result below uses the stated oracle bounds directly.

F.8 Proof of Thm. 1

Proof. Let $H = \widetilde{L}_q$, $a_t = \nabla H(W_t)$, and $g_t = a_t + e_t + \xi_t$, where $\mathbb{E}[\xi_t \mid \mathcal{F}_t] = 0$ and conditional centered variance is at most V. Smoothness on the update segment gives

$$H(W_{t+1}) \le H(W_t) - \eta\langle a_t, g_t\rangle + \frac{L_s\eta^2}{2}\|g_t\|^2.$$

Take conditional expectations and use $-\langle a_t, e_t\rangle \le \frac{1}{2}\|a_t\|^2 + \frac{1}{2}\|e_t\|^2$ and $\|a_t + e_t\|^2 \le 2\|a_t\|^2 + 2\|e_t\|^2$. Then

$$\mathbb{E}[H(W_{t+1}) \mid \mathcal{F}_t] \le H(W_t) - \left(\frac{\eta}{2} - L_s\eta^2\right)\|a_t\|^2 + \left(\frac{\eta}{2} + L_s\eta^2\right)\|e_t\|^2 + \frac{L_s\eta^2}{2}V$$
$$\le H(W_t) - \frac{\eta}{4}\|a_t\|^2 + \frac{3\eta}{4}\|e_t\|^2 + \frac{L_s\eta^2}{2}V,$$

where the last line uses $\eta \le 1/(4L_s)$. Take total expectation, sum T steps, use the lower bound $H^\star$, and divide by $\eta T/4$ to obtain (15). □

For fixed nonzero variance, $\eta = O(T^{-1/2})$ gives an $O(T^{-1/2})$ stationarity term plus the bias floor. With fixed η, the transient decays as $1/T$ toward a neighborhood. These differ from a noise-free $O(1/T)$ convergence guarantee.

F.9 Proof of Thm. 2

Proof. Let $D_t = \mathbb{E}[H(W_t) - H^\star]$. The one-step bound and the PL inequality imply

$$D_{t+1} \le (1 - \eta\mu/2)D_t + \frac{3\eta}{4}\varepsilon^2 + \frac{L_s\eta^2}{2}V.$$

The stepsize ensures $r = 1 - \eta\mu/2 \in [0, 1)$. Iterating and summing the geometric series gives

$$D_t \le r^t D_0 + \frac{(3\eta/4)\varepsilon^2 + (L_s\eta^2/2)V}{1 - r} = r^t D_0 + \frac{3\varepsilon^2}{2\mu} + \frac{L_s\eta V}{\mu}.$$

□

F.10 Composition Across Changing Surrogates

Code or scale changes produce a sequence of local surrogate objectives H_t. To compose descent bounds, one must control changes in objective values (or optimality gaps), not only differences between gradients.

Assumption 5 (Objective drift)**.** Each H_t is $\mathcal{F}_t$-measurable, L_s-smooth on its update segment, and lower bounded by a common finite $H_{\inf}$. Define

$$d_t = \mathbb{E}[H_{t+1}(W_{t+1}) - H_t(W_{t+1})], \qquad S_T = \sum_{t=0}^{T-1} \max\{d_t, 0\}.$$

The conditional bias and variance bounds of Assump. 3 hold relative to $\nabla H_t(W_t)$.

Theorem 3 (Composition with accumulated objective drift)**.** *Under Assump. 5 and $\eta \le 1/(4L_s)$,*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}[\|\nabla H_t(W_t)\|^2] \le \frac{4(\mathbb{E}[H_0(W_0)] - H_{\inf} + S_T)}{\eta T} + 3\varepsilon^2 + 2L_s\eta V. \tag{25}$$

Proof. Apply the one-step descent inequality to H_t, then add d_t to replace $H_t(W_{t+1})$ by $H_{t+1}(W_{t+1})$ in expectation. Sum over t. The objective terms telescope, the final one is bounded below by $H_{\inf}$, and the sum of drift terms is at most S_T. Division by $\eta T/4$ gives the result. □

PL composition. Let $D_t = \mathbb{E}[H_t(W_t) - H_t^\star]$ and assume each local surrogate satisfies PL with the same μ. If the expected positive change in the optimality gap at W_{t+1} is bounded by $\kappa_t \ge 0$, then

$$D_{t+1} \le rD_t + c + \kappa_t, \quad r = 1 - \eta\mu/2, \quad c = \frac{3\eta}{4}\varepsilon^2 + \frac{L_s\eta^2}{2}V.$$

Therefore

$$D_T \le r^T D_0 + \frac{c}{1 - r} + \sum_{t=0}^{T-1} r^{T-1-t}\kappa_t. \tag{26}$$

If changes cease, the remaining trajectory contracts geometrically to the bias/variance floor. If $\kappa_t \to 0$, its convolution with the geometric weights vanishes, but the bias/variance floor remains unless those errors also vanish. A gradient-drift condition alone does not justify omitting this floor or claiming global linear convergence of the hard QAT objective.

F.11 Interaction with SGD and AdamW

A scalar group gain has a simple interpretation under SGD, but a different one under a moment-normalizing optimizer. This distinction prevents interpreting bounded input gradients as a general bound on optimizer steps.

SGD. For a group with upstream gradient $v_{g,t}$, the update $W_{g,t+1} = W_{g,t} - \eta b_{g,t} v_{g,t}$ is equivalent to an SGD step with group learning rate $\eta b_{g,t}$. The defining choice is to obtain those learning-rate multipliers from finite quantizer responses.

Constant positive gains under Adam. Consider one coordinate, zero-initialized moments, a constant $c > 0$, and the same upstream sequence v_t. If $g_t = cv_t$ for every step, Adam's moment recursions give $m_t^{(g)} = c\, m_t^{(v)}, \qquad u_t^{(g)} = c^2 u_t^{(v)}$, where u_t is the second-moment estimate. The same identities hold after bias correction. With zero denominator offset, $\frac{\widehat{m}_t^{(g)}}{\sqrt{\widehat{u}_t^{(g)}}} = \frac{\widehat{m}_t^{(v)}}{\sqrt{\widehat{u}_t^{(v)}}}$. For a nonzero Adam offset ϵ_A, the ratio becomes $\widehat{m}_t^{(v)} / (\sqrt{\widehat{u}_t^{(v)}} + \epsilon_A / c)$. A gain of zero is excluded from the cancellation argument. Decoupled weight decay remains a separate update. These identities compare prescribed gradient sequences; if trajectories change, future upstream gradients can also change.

Time-varying gains. For $g_t = b_t v_t$ with varying b_t, the histories are

$$m_t^{(g)} = (1-\beta_1) \sum_{s=1}^{t} \beta_1^{t-s} b_s v_s, \qquad u_t^{(g)} = (1-\beta_2) \sum_{s=1}^{t} \beta_2^{t-s} b_s^2 v_s^2.$$

There is generally no common factor that cancels. Refreshes, clipping, and changes in quantizer response can therefore affect the adaptive update through its history. Fixed- or shuffled-gain controls would distinguish response-dependent gain learning from other effects of time-varying gradient scaling.

G Limitations and Extensions

Our evaluation covers 1–2-bit weight QAT on 1B–8B models, including an 8B trajectory through 90K steps. The tables report point estimates without multi-seed uncertainty intervals and assess VR-enabled JacQuant as a combined training configuration. Multi-seed studies and matched component controls would further characterize robustness and the respective contributions of gain learning and control variates. Exact reproduction also requires checkpoint/tokenizer versions, sequence and token budgets, evaluation settings, and the implemented VR/Dither configuration.

The theory characterizes a local smooth surrogate with fixed quantizer parameters. Extending it to joint scale/weight updates and AdamW would connect the guarantees more closely to end-to-end training. Beyond the evaluated A8/A16 settings, low-bit activation and KV-cache quantization offer further applications of sensitivity learning. Non-uniform scalar quantizers can use the same response fit, while vector/codebook quantizers motivate richer block surrogates and corresponding analysis.

H Focused Comparisons with Related Methods

Discrete optimization and geometric transforms. PV-Tuning (Malinovskii et al., 2024) alternates discrete assignments and continuous quantized values. Its optimization parameterization differs from JacQuant, which retains the base QAT weights and changes their backward signal. The Rotation Trick (Fifty et al., 2025) constructs an input-dependent rotation and rescaling, held constant within differentiation. JacQuant instead fits scalar group gains from finite responses and reuses them across steps. Scalars cannot represent arbitrary rotations; the matched comparison in Tab. 1a tests these two designs directly.

Relaxations and zeroth-order estimators. Soft-to-Hard VQ (Agustsson et al., 2017) anneals a differentiable relaxation toward hard assignments. JacQuant-Probe keeps the hard task forward

and changes the backward rule. FOGZO (Yang & Aamodt, 2025) combines first- and zeroth-order information, whereas ZeroQAT (Tan et al., 2025) uses forward-only estimation to address backpropagation cost. JacQuant queries the quantizer response rather than task-loss differences and retains first-order backpropagation through the network. These distinctions concern the information queried and the forward map; direct efficiency comparisons with these methods are outside the reported experiments.

Optimizer geometry and variance reduction. Muon (Liu et al., 2025a) changes matrix update geometry; JacQuant changes the signal delivered to an optimizer through the quantizer. Control-variate methods, including SVRG (Johnson & Zhang, 2013; Malinovsky et al., 2022), SAGA (Defazio et al., 2014), SARAH (Nguyen et al., 2017), and STORM (Cutkosky & Orabona, 2019), target sampling noise. JacSketch (Gower et al., 2021) interprets certain control variates as sketches of per-example loss Jacobians. JacQuant estimates the quantizer's sensitivity, a different Jacobian. These components are compatible in principle, but their individual cost and benefit must be evaluated separately.