A CONVERGENCE ANALYSIS: PROOF OF THEOREM 1

Let $F(x) = f(x) + \lambda\phi(x)$ be the overall regularized loss with smoothness constant $L = L + \lambda L_\phi$. Then, due to smoothness inequality, we have

$$\begin{aligned} F(x_{t+1}) &\leq F(x_t) + \langle \nabla F(x_t), x_{t+1} - x_t\rangle + \frac{L}{2}\|x_{t+1} - x_t\|^2 \\ &= F(x_t) - \alpha\langle \nabla f(x_t) + \lambda(x_t - Q(x_t)), \widetilde{\nabla} f(x_t) + \lambda(x_t - Q(x_t))\rangle \\ &\qquad + \frac{L\alpha^2}{2}\|\widetilde{\nabla} f(x_t) + \lambda(x_t - Q(x_t))\|^2. \end{aligned}$$

Applying conditional expectation $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot \mid x_t]$, using unbiasedness of stochastic gradient and boundedness of variance, we get

$$\begin{aligned} \mathbb{E}_t[F(x_{t+1}) - F(x_t)] &\leq -\alpha\|\nabla f(x_t) + \lambda(x_t - Q(x_t))\|^2 + \frac{L\alpha^2}{2}\mathbb{E}_t\left[\|\widetilde{\nabla} f(x_t) + \lambda(x_t - Q(x_t))\|^2\right] \\ &= -\alpha\|\nabla_{\lambda \mathrm{P}} f(Q(x_t))\|^2 + \frac{L\alpha^2}{2}\left(\|\nabla_{\lambda \mathrm{P}} f(Q(x_t))\|^2 + \sigma^2\right) \\ &= -\alpha\left(1 - \frac{L\alpha}{2}\right)\|\nabla_{\lambda \mathrm{P}} f(Q(x_t))\|^2 + \frac{L\alpha^2}{2}\sigma^2 \\ &\leq -\frac{\alpha}{2}\|\nabla_{\lambda \mathrm{P}} f(Q(x_t))\|^2 + \frac{L\alpha^2}{2}\sigma^2, \end{aligned}$$

where we enforced $\alpha \leq \frac{1}{L}$ step-size restriction. Applying full expectation and summing the obtained inequalities from $t = 0, \ldots, T-1$, we get

$$\begin{aligned} \mathbb{E}\|\nabla_{\lambda \mathrm{P}} f(Q(\hat{x}))\|^2 &\leq \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla_{\lambda \mathrm{P}} f(Q(x_t))\|^2 \leq \frac{2(F(x_0) - F(x_T))}{\alpha T} + \alpha L\sigma^2 \\ &= \frac{2(f(x_0) - f(x_T))}{\alpha T} + \frac{2\lambda(\phi(x_0) - \phi(x_T))}{\alpha T} + \alpha L\sigma^2 \\ &\leq \frac{1}{\sqrt{T}}\left(2(f(x_0) - f^*) + 2\lambda(\phi(x_0) - \phi(x_T)) + L\sigma^2\right)\max\left(1, \frac{L}{\sqrt{T}}\right) \end{aligned}$$

with $\alpha = \min(\frac{1}{L}, \frac{1}{\sqrt{T}})$. It remains to bound the term $\phi(x_0) - \phi(x_T)$ stemming from the quantization.

Due to the assumption 3 on quantization we can represent the scalar function ϕ as path-independent line integral:

$$\phi(x) = \int_{x_0}^{x} (I - Q) \cdot d\mathbf{r},$$

where I is the identity map, i.e. $I(x) = x$ for any $x \in \mathbb{R}^d$. Since $I - Q$ is a gradient field (i.e., $x - Q(x) = \nabla\phi(x)$), the line integral above does not depend on how the endpoints x_0 and x are connected. Therefore, we choose the direct path $r(t) = x_0 + (x - x_0)t$ for $t \in [0, 1]$ and simplify the integral into usual Riemannian integral as

$$\phi(x) = \int_{x_0}^{x} (I - Q) \cdot d\mathbf{r} = \int_0^1 \langle r(t) - Q(r(t)), r'(t)\rangle\, dt.$$

Using this relation, we can bound the term as follows

$$\begin{aligned} \phi(x_0) - \phi(x_T) &\leq \left|\int_0^1 \langle r(t) - Q(r(t)), r'(t)\rangle\, dt\right| \\ &\leq \int_0^1 \|r(t) - Q(r(t))\| \cdot \|r'(t)\|\, dt \leq \max_{x \in [x_0, x_T]} \|x - Q(x)\| \cdot \|x_0 - x_T\|. \end{aligned}$$

Table 2. Hyperparameters for Llama-style pretraining runs (per model size). Token budgets follow a 100 tokens-per-parameter (TPP) rule.

Model (N)	Layers	Heads	d_{model}	Base LR	Tokens (D)
30M	6	5	640	1.2×10^{-3}	3B
50M	7	6	768	1.2×10^{-3}	5B
100M	8	8	1024	6.0×10^{-4}	10B
200M	10	10	1280	3.0×10^{-4}	20B
430M	13	13	1664	1.5×10^{-4}	43B
800M	16	16	2048	7.5×10^{-5}	80B

B Hyperparameters and Reproducibility

Shared settings. Sequence length $L = 512$; batch size 64 with gradient accumulation 8 (effective tokens/step fixed across sizes). Optimizer AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, $\varepsilon = 10^{-8}$, weight decay 0.1, gradient clip 1.0. Cosine LR schedule with warmup 10% of total steps. FP32 master weights and optimizer states; bfloat16 compute. Hardware: 8× H100 (NCCL, `compile` enabled).

Quantization. QuEST row-wise Hadamard quantizer for weights *and* activations; bit-widths $b \in \{2, 3, 4\}$ (symmetric, MSE-optimal Gaussian clipping). Trust-masked STE in transform domain.

CAGE settings. Throughout the experiments, unless otherwise stated, we use the decoupled variant, with silence ratio $s = 0.9$, and regularization parameter $\lambda = 2.0$.

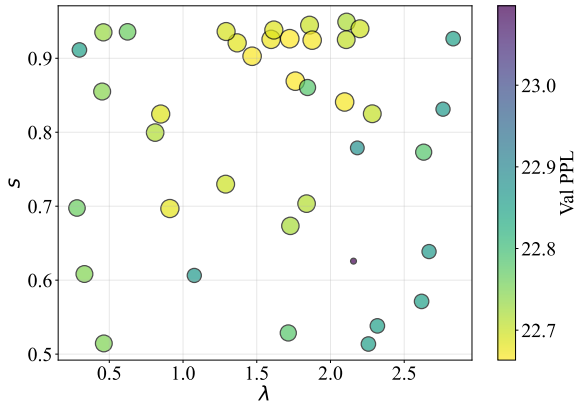

Figure 9. Validation perplexity for CAGE under different choices of silence ratio s and regularization coefficient λ on the 50M Llama model with W4A4 quantization.

C Ablation of CAGE Hyperparameters

We conduct a two-dimensional sweep over the CAGE hyperparameters; the silence ratio s and the regularization coefficient λ. Figure 9 reports validation perplexity for runs on a 50M Llama model trained on C4 in the W4A4 setting. We find that $s \in [0.8, 0.95]$ and $\lambda \in [1, 2.5]$ form a reliable operating region. Within this range, CAGE consistently improves over the baseline.

D Decoupled vs coupled CAGE

We extend Table 1 with a direct comparison between the coupled and decoupled formulations of CAGE. Table 3 shows that CAGE_C (coupled) and CAGE_D (decoupled) achieve nearly identical final validation perplexity across model sizes, both with and without Hadamard transforms (HT), and consistently outperform the QuEST baseline.

This behavior is expected: CAGE_C follows directly from the Pareto-stationary condition in our analysis, while CAGE_D is

a practical variant that decouples the correction from curvature/variance scaling to improve compatibility with a broader set of optimizers and training stacks. The empirical equivalence under AdamW indicates that the primary gains come from the Pareto-derived correction direction, and the decoupled formulation preserves these gains while simplifying integration.

Table 3. Pretraining results (final validation perplexity; lower is better) for W4A4 across model sizes. CAGE$_C$ is the coupled variant and CAGE$_D$ is the decoupled variant.

Method	30M	50M	100M
CAGE$_D$ + HT	26.277	22.747	18.944
CAGE$_C$ + HT	26.271	22.752	18.948
QuEST + HT	26.475	23.062	19.123
CAGE$_D$ (no HT)	27.287	23.781	19.630
CAGE$_C$ (no HT)	27.285	23.786	19.625
QuEST (no HT)	27.401	23.991	19.799
BF16	24.715	21.491	17.923

E ALGORITHM OVERHEADS

We quantify the runtime (and qualitative memory) overhead of CAGE relative to the QuEST baseline under identical training configurations: same model, sequence length, batch size, optimizer, quantization settings (W4A4), and hardware (single NVIDIA H100). CAGE adds one extra computation of the quantization residual $e_t = x_t - Q(x_t)$ (i.e., one additional quantization call) and a small number of elementwise operations (subtraction and an extra update add). Since the dominant cost in LLM training is matrix multiplications (forward/backward), we expect the end-to-end overhead to be small.

Implementation note. All quantization operations in these measurements are implemented in PyTorch for simplicity and reproducibility (i.e., not using a hand-tuned low-bit kernel). While the absolute iteration times therefore do not reflect fully optimized production implementations, the comparison between QuEST and CAGE is fair because both methods use the same quantization implementation and training stack. When Hadamard transforms (HT) are enabled, the runtime is additionally influenced by the (naive) Walsh–Hadamard transform used in our implementation; an optimized WHT implementation would reduce this component for both methods.

Measurement protocol. We measure wall-clock iteration time using CUDA-synchronized timing over a steady-state window of N consecutive iterations after an initial warmup period (excluding compilation and startup transients), and report mean $\pm$ standard deviation. Measurements use sequence length 2048 and batch size 16.

Results. Table 4 reports iteration time for two model sizes. Without HT, QuEST and CAGE have statistically indistinguishable iteration times (differences are within run-to-run variance). With HT enabled, iteration times are more sensitive to the WHT implementation; we observe that CAGE remains comparable to QuEST, with modest variability across model sizes. Overall, these results support that CAGE introduces negligible end-to-end overhead relative to the baseline in our training setup.

Table 4. Runtime per iteration (ms; mean $\pm$ std) for W4A4 training on a single H100 (sequence length 2048, batch size 16).

Method	100M (ms/iter)	430M (ms/iter)
QuEST (no HT)	101.6 ± 1.7	282.7 ± 3.1
CAGE$_D$ (no HT)	101.1 ± 1.8	283.1 ± 3.0
QuEST (+HT)	117.5 ± 1.8	286.6 ± 2.3
CAGE$_D$ (+HT)	105.8 ± 1.2	316.1 ± 3.7

F LONG-CONTEXT EVALUATION

We evaluate long-context behavior using RULER (Hsieh et al., 2024) on a Llama 3.1-8B model fine-tuned with QAT.

Setup. We fine-tune Llama 3.1-8B on TULU-SFT under W4A16 quantization and compare CAGE against QuEST and additional common post-training quantization baselines (RTN, GPTQ). We report the average RULER score (higher is better). All evaluated models share the same base architecture and are compared under the same evaluation protocol.

Results. Table 5 shows that CAGE improves long-context performance over the QuEST baseline under W4A16, and substantially outperforms RTN and GPTQ in this setting. While quantization reduces performance relative to the full-precision base model, CAGE narrows this gap compared to prior QAT baselines, indicating that the benefits of our correction extend to long-context behavior.

Table 5. RULER average score (higher is better) for Llama 3.1-8B under W4A16 after fine-tuning on TULU-SFT.

Model	Avg. RULER score
Llama 3.1-8B (base, full precision)	88.3
Llama 3.1-8B (CAGE, W4A16)	73.2
Llama 3.1-8B (QuEST, W4A16)	68.7
Llama 3.1-8B (GPTQ, W4A16)	65.1
Llama 3.1-8B (RTN, W4A16)	41.5

G Formal Justification for λ Scheduler

Recall that the decoupled CAGE update applies the instantaneous quantization error

$$e_t := x_t - Q(x_t) \tag{4}$$

as a post-step correction:

$$\tilde{x}_{t+1} = \textsc{OptStep}(x_t; g_t), \qquad x_{t+1} = \tilde{x}_{t+1} - \alpha\lambda_t e_t. \tag{5}$$

Empirically (and as stated in §3.3), turning on the correction too early can over-constrain the dynamics; we therefore use

$$r_t := t/T, \qquad \lambda_t = \begin{cases} 0, & r_t \le s, \\ \lambda \cdot \dfrac{r_t - s}{1 - s}, & r_t > s, \end{cases} \tag{6}$$

for a silence ratio $s \in [0, 1)$.

We provide a complementary formal perspective to show that during the silence period, applying CAGE is ineffective and may also undermine training stability.

G.1 Quantization as a random walk

Let $\mathcal{Q}$ be the (discrete) quantized support and $Q : \mathbb{R}^d \to \mathcal{Q}$ the quantizer. For any $q \in \mathcal{Q}$ define the cell $\mathcal{C}(q) := \{x \in \mathbb{R}^d : Q(x) = q\}$. Let $\{\mathcal{F}_t\}_{t\ge 0}$ be the filtration generated by the algorithm up to iteration t. Define the *cell-switch* times

$$\tau_0 := 0, \qquad \tau_{k+1} := \min\{t > \tau_k : \ Q(x_t) \neq Q(x_{\tau_k})\}. \tag{7}$$

Within a visit $t \in [\tau_k, \tau_{k+1})$ the quantized anchor $Q(x_t)$ is constant.

The residuals e_t are correlated inside a cell (since the same anchor $Q(x_t)$ is reused), but they can "reset" when the iterate crosses to a new cell. We capture this by grouping residuals per-visit:

$$E_k := \sum_{t=\tau_k}^{\tau_{k+1}-1} e_t. \tag{8}$$

G.2 Assumptions

The following conditions formalize the intuition used in the rebuttal.

Assumption A (bounded residual). There exists $B > 0$ such that $\|e_t\| \le B$ for all t almost surely. For standard uniform (or clipped) quantizers this holds since each coordinate error is bounded by half a bin width.

Assumption B (bounded dwell time during exploration). During the *exploration phase* (early training, larger effective steps), cell visits are not extremely long: there exists $L_{\max} \ge 1$ such that for all visits occurring in the exploration phase,

$$\tau_{k+1} - \tau_k \le L_{\max}. \tag{9}$$

This reflects the empirical regime where x_t traverses many nearby quantization points.

Assumption C (reset symmetry / mean-zero block residual). During the exploration phase, the per-visit residual block satisfies an (approximate) martingale-difference property:

$$\mathbb{E}[E_k \mid \mathcal{F}_{\tau_k}] = 0 \quad \text{or more generally} \quad \|\mathbb{E}[E_k \mid \mathcal{F}_{\tau_k}]\| \le \mu \tag{10}$$

for some small bias $\mu \ge 0$. Intuitively, when the iterate enters a new cell, the offset from the cell center is roughly symmetric due to stochasticity, so the visit-integrated residual has a small conditional mean.

G.3 Cancellation bound

Assumptions A–C yield a quantitative "random-walk cancellation" statement.

Lemma 2 (Bounded block increments) *Under Assumptions A–B, for each exploration-phase visit,*

$$\|E_k\| \le \sum_{t=\tau_k}^{\tau_{k+1}-1} \|e_t\| \le B(\tau_{k+1} - \tau_k) \le BL_{\max}. \tag{11}$$

Lemma 3 (concentration across visits) *Assume $\mu = 0$ in (10). Then for any unit vector $u \in \mathbb{S}^{d-1}$ and any $\varepsilon > 0$, by Azuma concentration bound,*

$$\mathbb{P}\left(\left|u^\top \sum_{k=0}^{K-1} E_k\right| \ge \varepsilon\right) \;\le\; 2\exp\left(-\frac{\varepsilon^2}{2K(BL_{\max})^2}\right), \tag{12}$$

And consequently $\left\|\sum_{k=0}^{K-1} E_k\right\| = O_{\mathbb{P}}(BL_{\max}\sqrt{K})$ via a standard ε-net argument.

Let the exploration phase span $T_{\exp}$ iterations and $K(T_{\exp})$ visits. When switching is frequent, $K(T_{\exp}) = \Theta(T_{\exp})$ and Lemma 3 implies

$$\left\|\frac{1}{T_{\exp}} \sum_{t=0}^{T_{\exp}-1} e_t\right\| = \left\|\frac{1}{T_{\exp}} \sum_{k=0}^{K(T_{\exp})-1} E_k\right\| = O_{\mathbb{P}}\left(\frac{BL_{\max}}{\sqrt{T_{\exp}}}\right). \tag{13}$$

Thus, *early* residuals behave like bounded, approximately mean-zero perturbations: they cancel in aggregate but have $\sqrt{T}$-scale fluctuations.

G.4 Why large constant λ early can be harmful

If $\lambda_t \equiv \lambda$ were active from the start, the cumulative correction displacement over the exploration phase is

$$\left\|\sum_{t=0}^{T_{\exp}-1} \alpha\lambda e_t\right\| = \alpha\lambda \left\|\sum_{k=0}^{K(T_{\exp})-1} E_k\right\| = O_{\mathbb{P}}\left(\alpha\lambda BL_{\max}\sqrt{K(T_{\exp})}\right). \tag{14}$$

Even if the mean is near zero, the *variance* grows with time like a random walk; a large early λ therefore injects a nontrivial noise-like forcing term into the dynamics, potentially destabilizing training before the iterate has settled into a basin. This motivates *silencing* the correction during early exploration (setting $\lambda_t \simeq 0$ for $t \le sT$).

G.5 Why λ_t should increase late

Near convergence, the iterate typically oscillates among a small set of adjacent cells for long periods, so the "reset" condition (10) weakens. In the extreme, if $Q(x_t)$ is constant for $t \in [t_0, t_0 + M)$, then e_t no longer cancels and $\left\|\sum_{t=t_0}^{t_0+M-1} e_t\right\|$ can scale as $\Theta(M)$. In this regime the correction is *coherent* and effectively pushes x_t toward the quantized support, which is precisely when enforcing Pareto-stationarity becomes beneficial. Hence λ_t should be *larger* late in training.

G.6 Dynamic quantizers where $L \approx 1$

Code-based visits. Define visits by the *integer code*, not by the reconstructed value.

Write the dynamic quantizer as

$$c_t := C(x_t, s_t) := \mathrm{clip}\left(\mathrm{round}\left(\frac{x_t}{s_t}\right), [-M, M]\right) \in \mathbb{Z}^d, \qquad q_t := Q_{s_t}(x_t) = s_t\, c_t.$$

The residual is then

$$e_t = x_t - q_t = x_t - s_t c_t = s_t\left(\frac{x_t}{s_t} - c_t\right) =: s_t\, r_t,$$

where

$$r_t := y_t - \mathrm{round}(y_t), \qquad y_t := \frac{x_t}{s_t}.$$

Visit definition. Define "visits" by *code stability*:

$$\tau_{k+1} := \min\{t > \tau_k : c_t \neq c_{\tau_k}\}.$$

The sequence (c_t) can remain constant for many steps even if (s_t) changes at every step. Thus, the "visit time $= 1$" issue disappears.

Assumption. A defensible assumption is the following:

> During the exploration phase, conditional on s_t (or more generally on $\mathcal{F}_{t-1}$), the normalized fractional part
>
> $$r_t = y_t - \mathrm{round}(y_t)$$
>
> is approximately symmetric around 0 and mixes across time. Consequently,
>
> $$\mathbb{E}[r_t \mid \mathcal{F}_{t-1}] \approx 0.$$

It follows immediately that

$$\mathbb{E}[e_t \mid \mathcal{F}_{t-1}] = \mathbb{E}[s_t r_t \mid \mathcal{F}_{t-1}] = s_t\, \mathbb{E}[r_t \mid \mathcal{F}_{t-1}] \approx 0.$$

Block sums over code visits. For block sums over *code visits*,

$$E_k := \sum_{t=\tau_k}^{\tau_{k+1}-1} e_t = \sum_{t=\tau_k}^{\tau_{k+1}-1} s_t r_t,$$

the same martingale and concentration arguments as before apply. Now, however, the "sticking" correlation arises from the code remaining constant, which is the appropriate notion of a visit.