OJBKQ: Objective-Joint Babai–Klein-Based Quantization*

Xinyu Wang[1], **Ziyu Zhao**[1], **Peng Lu**[2], **Yu Gu**[1] and **Xiao-Wen Chang**[1,‡]
[1]McGill University
[2]Université de Montréal
{xinyu.wang5, ziyu.zhao2}@mail.mcgill.ca, chang@cs.mcgill.ca

Abstract

Post-training quantization (PTQ) is widely used to compress large language models without retraining. However, many existing weight-only methods rely on heuristic objectives and greedy rounding strategies, leading to noticeable degradation under low-bit quantization. We propose **OJBKQ**, a layer-wise PTQ method that formulates weight quantization as a joint optimization problem over activations and weights, yielding a multiple-right-hand-side box-constrained integer least squares (BILS) problem per layer. To solve this NP-hard problem, we extend Klein's randomized variant of Babai's nearest-plane algorithm to handle box constraints and run the extended algorithm K times to generate K box-constrained Klein points. The minimum-residual candidate among these points and the box-constrained Babai point is selected as a suboptimal solution. Experiments on large language models show that OJBKQ achieves lower perplexity at 3–4 bits than existing PTQ methods, while maintaining comparable computational cost.

1 Introduction

Large Language Models (LLMs) [Achiam *et al.*, 2023; Touvron *et al.*, 2023; Comanici *et al.*, 2025] pose substantial deployment challenges due to their massive memory footprint and inference costs [Chen *et al.*, 2023]. Layer-wise post-training quantization (PTQ) [Frantar *et al.*, 2023; Lin *et al.*, 2024; Chee *et al.*, 2024] has become a widely adopted compression standard, enabling efficient deployment by sequentially quantizing parameters without retraining. In this work, we focus on standard and practically relevant 4-bit and 3-bit settings (e.g., group size 128) and study how to further improve quantization quality beyond strong PTQ baselines.

Despite its empirical success, layer-wise PTQ is a *discrete* optimization problem. At its core, each layer-wise step tries to solve an integer quadratic programming problem, typically handled using sequential rounding-type solvers. Recent theoretical analyses [Chen *et al.*, 2025] reveal that standard solvers (e.g., GPTQ) admit a lattice-decoding interpretation: they are closely related to Babai's nearest-plane algorithm [Babai, 1986], which finds the Babai point, a suboptimal solution to the closest vector problem (CVP), also known as the ordinary integer least-squares (OILS) problem. More precisely, the underlying optimization problem in PTQ is a box-constrained integer least-squares (BILS) problem. However, when the lattice basis matrix is ill-conditioned or when the dimension is large, the residual associated with the Babai point may be significantly larger than the optimal residual. Thus, it is desirable to find a better sub-optimal solution. Furthermore, minimizing a layer-local proxy loss does not necessarily lead to better end-to-end quantization due to error propagation and input distribution drift. Consequently, the *selection objective* used to compare and choose among candidate solutions is as crucial as the solver itself, yet remains inconsistent across existing methods.

In this paper we propose **OJBKQ** (Objective-Joint Babai–Klein Quantization with (K+1)-Best Sampling), a unified framework that addresses both issues. **(i) Joint Target Alignment (JTA):** To better align layer-wise decisions with end-to-end behavior under error propagation, we introduce a unified candidate *scoring* objective (Eq. (7)) with a single continuous knob. JTA interpolates between two fundamental alignment targets: matching the *runtime* quantized activations induced by the partially-quantized network, and matching the corresponding *full-precision reference* outputs [Arai and Ichikawa, 2025]. The formulated quantization problems at each layer are multiple-right-hand-side BILS problems. **(ii) Random-K quantization:** For each column of the weight matrix in a BILS problem, we apply the box-constrained Babai algorithm to obtain one sub-optimal solution (a low-bit vector candidate). For the same column, we then extend Klein's K-time randomized Babai algorithm [Klein, 2000] to the box-constrained case to generate K additional sub-optimal solutions independently. Among these $K+1$ candidates, we select the one which has the minimum residual, referred to as the best Babai–Klein point. There are two levels of parallelization. One across the columns of the weight matrix and the other across the K points produced by the box-constrained Klein algorithm. Both can be implemented ef-

*A full version with appendix is available at https://arxiv.org/pdf/2602.08376

‡Corresponding author.

ficiently.

In **OJBKQ**, Random-K expands the quality of the *candidate set*, while JTA improves *the selection of* the optimal candidate.

Our contributions are summarized as follows:

- **Quantization Formulation.** We revisit layer-wise PTQ from a CVP perspective and reformulate it as an explicit lattice-decoding problem. This enables a faithful Babai-style implementation that operates via triangular factors and back-substitution, without materializing any matrix inverse.
- **Superior Sub-optimal Solution.** We extend Klein's randomized algorithm to the box-constrained case and apply it to PTQ. By combining the Babai solution with K parallel runs of the extended Klein randomized Babai algorithm, we generate $K + 1$ integer candidates and select the one with the smallest residual, yielding low perplexity in both 4-bit and 3-bit configurations. In designing and implementation of our numerical algorithm, we take both accuracy and efficiency into account. For example, unlike GPTQ, no inverse of a matrix is computed. To reduce the overhead of multiple independent solves, we develop a GPU-efficient algorithm presented as Algorithm 2 in the appendix of the full version.
- **JTA Candidate Selection.** We propose the **Joint Target Alignment (JTA)** objective, a unified scoring criterion that subsumes commonly used PTQ objectives as special cases by varying the alignment target between runtime-quantized and full-precision references, providing a more fine-grained criterion for candidate selection under error propagation.

2 Related Work

Layer-wise post-training quantization (PTQ). Post-training quantization is a practical standard for compressing large language models without retraining. Following recent PTQ taxonomies [Zhao *et al.*, 2025], mainstream weight-only PTQ methods can be broadly grouped by their dominant mechanism. *Compensation-based* approaches, exemplified by GPTQ [Frantar *et al.*, 2023], quantize weights sequentially while updating the remaining weights to compensate for accumulated quantization errors, often leveraging second-order curvature surrogates estimated from calibration data. *Rotation-based* methods such as QuIP [Chee *et al.*, 2024] apply structured transformations to reshape weight distributions and improve quantizability under low-bit constraints. *Salience-based* methods, represented by AWQ [Lin *et al.*, 2024], reduce quantization error by selecting scaling factors according to activation/weight importance, avoiding mixed-precision deployment. In addition, *optimization-based* approaches (e.g., OmniQuant and PoTPTQ) [Shao *et al.*, 2024; Wang *et al.*, 2025] directly optimize quantization parameters with lightweight objectives while keeping pretrained weights frozen. Overall, existing PTQ pipelines mainly differ along two orthogonal axes: (i) the approximate *solver* used to handle the discrete subproblem, and (ii) the *objective* used to evaluate candidates and guide selection.

Integer least squares and Babai-type solution. CVP or OILS arises in lattice decoding [Agrell *et al.*, 2002], lattice-based cryptography [Micciancio and Goldwasser, 2002], and GPS [Teunissen, 1993; Teunissen, 1995; Chang *et al.*, 2005]. BILS mainly arises in MIMO detection [Damen *et al.*, 2003; Chang and Han, 2008]. The most practical approach for OILS is the Schnorr–Euchner algorithm [Schnorr and Euchner, 1994], which enumerates integer points in an ellipsoid to find the optimal solution and has been extended for BILS. The MATLAB package MILES provides solvers for various ILS problems [Chang, 2022]. Since OILS and BILS are NP-hard [Emde-Boas, 1981; Verdú, 1989], in some applications efficient sub-optimal solutions are often applied. A widely used method is Babai's nearest-plane algorithm [Babai, 1986], which computes the Babai point by back substitution on an upper-triangular linear system, where rounding is performed at each step. Actually, the algorithm is exactly the size-reduction procedure of the LLL algorithm [Lenstra *et al.*, 1982], and it has been extended to the box-constrained case. While computationally efficient, when the coefficient matrix is ill-conditioned, or when the dimension of the integer vector is large, the Babai point may have much larger residual than the optimal solution. There are other sub-optimal algorithms, such as those proposed in [Barbero and Thompson, 2006; Chang *et al.*, 2024], but they are not suitable for GPU computing. Fortunately, there is a randomized version of the Babai algorithm proposed in [Klein, 2000]. In each step of the Babai algorithm, instead of rounding the real solution to the nearest integer, Klein's algorithm rounds it to a nearby integer according to a predefined distribution. When the algorithm is run K times, it generates K integer points. Then the best one, which gives the minimum residual, is chosen as the final sub-optimal solution. This algorithm is for the unconstrained case and has not yet been widely used. Recently, the connection between sequential PTQ solvers and the Babai point has gained attention [Chen *et al.*, 2025]. Some similar work can be found in the GPS literature [Teunissen, 2003; Chang *et al.*, 2005]. This paper emphasizes that the PTQ problems can be modeled as BILS problems so that sub-optimal BILS algorithms can be applied.

Objectives under error propagation and distribution drift. A recurring challenge in layer-wise PTQ is that minimizing a layer-local reconstruction proxy does not necessarily translate to improved end-to-end quality, due to quantization error propagation and the resulting input distribution drift across layers. Consequently, the choice of the *alignment target*—whether to match outputs under runtime (partially-quantized) activations or to match full-precision reference outputs—plays a central role in practical PTQ performance. Recent work revisits this issue from an error-propagation viewpoint and advocates objectives that better reflect downstream behavior under partial quantization [Arai and Ichikawa, 2025]. Our JTA objective follows this perspective but provides a unified scoring criterion with a single continuous knob that interpolates between runtime-quantized

Although it is classified as PTQ in the survey, it still requires training on the calibration dataset to find scales.Then we exclude it from our baselines

activations and full-precision reference targets, enabling consistent candidate selection across layers.

3 Methodology

3.1 Layer-wise Objectives and Joint Target Alignment (JTA)

We now analyze commonly used layer-wise PTQ objectives. Consider a linear layer with full-precision weight matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$. Let $\mathbf{X} \in \mathbb{R}^{p\times m}$ be full-precision calibration activations, $\widetilde{\mathbf{X}} \in \mathbb{R}^{p\times m}$ be the corresponding *runtime* activations produced by a partially quantized network (i.e., upstream layers already quantized), $\mathbf{Y}^{\mathrm{fp}} := \mathbf{X}\mathbf{W}$ denote the full-precision reference output, and $\mathbf{Y}^{\mathrm{rt}} := \widetilde{\mathbf{X}}\mathbf{W}$ denote the runtime-consistent reference output under partially quantized activations. At runtime, a candidate $\widehat{\mathbf{W}}$ produces the output $\widehat{\mathbf{Y}} := \widetilde{\mathbf{X}}\widehat{\mathbf{W}}$.

Many existing objectives can be interpreted as choosing different alignment targets:

$$\text{(Runtime-consistent)} \quad \min_{\widehat{\mathbf{W}}} \ \|\widetilde{\mathbf{X}}\widehat{\mathbf{W}} - \widetilde{\mathbf{X}}\mathbf{W}\|_F^2 \qquad (1)$$

$$= \|\widehat{\mathbf{Y}} - \mathbf{Y}^{\mathrm{rt}}\|_F^2, \qquad (2)$$

$$\text{(Full-precision mapping)} \quad \min_{\widehat{\mathbf{W}}} \ \|\mathbf{X}\widehat{\mathbf{W}} - \mathbf{X}\mathbf{W}\|_F^2, \qquad (3)$$

$$\text{(Mismatch target)} \quad \min_{\widehat{\mathbf{W}}} \ \|\widetilde{\mathbf{X}}\widehat{\mathbf{W}} - \mathbf{X}\mathbf{W}\|_F^2 \qquad (4)$$

$$= \|\widehat{\mathbf{Y}} - \mathbf{Y}^{\mathrm{fp}}\|_F^2. \qquad (5)$$

In these optimization problems, each column of $\widehat{\mathbf{W}}$ takes values in a finite subset of a translated lattice. Specifically, $\widehat{\mathbf{W}}$ has a special form; see Eq. (9). Later we will see these are in fact BILS problems. Eq. (1) directly reflects runtime behavior but aligns to a target computed on quantized activations, which may be noisy; this objective is adopted by GPTQ and QUIP. All three formulations above admit a convenient least-squares structure in their continuous relaxation (i.e., before the integrality constraints are imposed), but suffers from train–run mismatch when earlier layers are quantized, and is optimized by AWQ. Eq. (4) partially alleviates this mismatch by evaluating candidates on runtime activations while preserving the full-precision reference; QEP [Arai and Ichikawa, 2026] employs this objective as a corrective patch within a layer-wise quantization pipeline, although it may over-penalize candidates when error drift accumulates across layers. These trade-offs motivate a unified selection objective that interpolates between runtime-consistent and full-precision references, while optionally controlling weight drift.

We propose an expressive *Joint Target Alignment (JTA)* objective with a single continuous knob to score and select candidates generated by a random decoding. We define an interpolated target

$$\mathbf{Y}^{\star}(\mu) := (1-\mu)\,\mathbf{X}\mathbf{W} + \mu\,\widetilde{\mathbf{X}}\mathbf{W}, \qquad \mu \in [0,1], \qquad (6)$$

and formulate the optimization problem:

$$\min \mathcal{S}(\widehat{\mathbf{W}}) \ := \ \|\widetilde{\mathbf{X}}\widehat{\mathbf{W}} - \mathbf{Y}^{\star}(\mu)\|_F^2 \ + \ \lambda^2\|\widehat{\mathbf{W}} - \mathbf{W}\|_F^2. \qquad (7)$$

The knob μ interpolates between matching full-precision and runtime-consistent references, while λ optionally regularizes weight displacement. In our framework, **Random-$(K+1)$ quantization** (Sec. 3.4) finds a sub-optimal solution to (7), and **JTA** (Eq. (7)) provides a consistent curvature-aware criterion for selecting the best candidate under error propagation. When μ=1 and λ=0, Eq. (7) reduces to the runtime-consistent objective in Eq. (1); when μ=0 and λ=0, it aligns to the full-precision reference in Eq. (4).

End-to-end layer-wise procedure. For each layer, we (i) form $\widetilde{\mathbf{X}}$ by a partially quantized forward pass, (ii) find a sub-optimal optimal solution to Eq. (7), to be seen in the next sections. This closes the loop between discrete candidate generation and objective-aligned selection.

3.2 From Layer-wise Quantization to Integer Least Squares

From (7) we obtain

$$\mathcal{S}(\widehat{\mathbf{W}}) \ := \ \left\| \begin{bmatrix} \widetilde{\mathbf{X}} \\ \lambda\mathbf{I} \end{bmatrix} \widehat{\mathbf{W}} - \begin{bmatrix} \mathbf{Y}^{\star}(\mu) \\ \lambda\mathbf{W} \end{bmatrix} \right\|_F^2. \qquad (8)$$

The task of finding an optimal quantized weight $\widehat{\mathbf{W}}$ to minimize the selection objective in Sec. 3.1 can be formally mapped to the classical *Integer Least Squares* (ILS) problem.

For the derivation, we first introduce additional notation. Let

$$\mathbb{B} = \{0, 1, \ldots, 2^{\mathrm{wbit}} - 1\}$$

denote the box constraint corresponding to the set of admissible integer values for a wbit-bit quantized weight. Let

$$\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_n] \in \mathbb{B}^{m\times n}$$

denote the quantized version of the full-precision weight matrix $\mathbf{W}$. We further define

$$\mathbf{S} = [\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n] \in \mathbb{R}^{m\times n}, \ \mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n] \in \mathbb{Z}^{m\times n}$$

as the scale matrix and zero-point matrix, respectively. Both $\mathbf{S}$ and $\mathbf{Z}$, which have some structures (each $\mathbf{s}_j$ and each $\mathbf{z}_j$ have several groups and the elements in the same group have the same value), are pre-computed using standard statistical calibration methods (e.g., absmax calibration).

Finally, the dequantized weight matrix is given by

$$\widehat{\mathbf{W}} = \mathbf{S} \odot (\mathbf{Q} - \mathbf{Z}), \qquad (9)$$

where $\odot$ denotes element-wise multiplication.

To simplify notation in Eq. (8), we further define

$$\mathbf{A} := \begin{bmatrix} \widetilde{\mathbf{X}} \\ \lambda\mathbf{I} \end{bmatrix}, \qquad \mathbf{T} := \begin{bmatrix} \mathbf{Y}^{\star}(\mu) \\ \lambda\mathbf{W} \end{bmatrix} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_n].$$

It is easy to see that (8) can be rewritten as

$$\mathcal{S}(\widehat{\mathbf{W}}) = \sum_{j=1}^{n} \|\mathbf{A}\mathbf{D}_j(\mathbf{q}_j - \mathbf{z}_j) - \mathbf{t}_j\|_2^2,$$

where $\mathbf{D}_j := \mathrm{diag}(\mathbf{s}_j) \in \mathbb{R}^{m\times m}$.

As a result, the optimization problem in Eq. (7) decomposes into n independent BILS problems:

$$\min_{\mathbf{q}_j \in \mathbb{B}^m} \|\mathbf{A}\mathbf{D}_j(\mathbf{q}_j - \mathbf{z}_j) - \mathbf{t}_j\|_2^2, \qquad j = 1, \ldots, n. \quad (10)$$

Since each subproblem is independent, we simplify notation by focusing on a single per-column subproblem:

$$\min_{\mathbf{q} \in \mathbb{B}^m} \|\mathbf{A}\mathbf{D}(\mathbf{q} - \mathbf{z}) - \mathbf{t}\|_2^2. \quad (11)$$

However, for computational efficiency, we need to keep in mind that $\mathbf{A}$ is shared across the n BILS problems, while $\mathbf{D}$ depends on each individual.

3.3 Babai-Based Quantization

The standard way to compute the box-constrained Babai point, a sub-optimal solution to (11), is via the QR factorization of $\mathbf{AD}$. However, in the quantization application, the dimensions of $\mathbf{A}$ are large, so we instead use the Cholesky factorization to reduce the computational cost.

Let $\bar{\mathbf{q}} \in \mathbb{R}^m$ denote the unconstrained *real* least-squares solution of (11). The corresponding normal equations are given by

$$\mathbf{D}\mathbf{A}^\top\mathbf{A}\mathbf{D}(\bar{\mathbf{q}} - \mathbf{z}) = \mathbf{D}\mathbf{A}^\top\mathbf{t},$$

or equivalently

$$\mathbf{A}^\top\mathbf{A}\mathbf{D}(\bar{\mathbf{q}} - \mathbf{z}) = \mathbf{A}^\top\mathbf{t}. \quad (12)$$

Denote $\mathbf{v} := \mathbf{D}(\bar{\mathbf{q}} - \mathbf{z})$. To obtain $\mathbf{v}$ from (12), we first compute the Cholesky factorization $\mathbf{A}^\top\mathbf{A} = \mathbf{R}^\top\mathbf{R}$, where $\mathbf{R}$ is upper triangular. Then we solve two triangular linear systems $\mathbf{R}^\top\mathbf{u} = \mathbf{A}^\top\mathbf{t}$ and $\mathbf{R}\mathbf{v} = \mathbf{u}$.

After $\mathbf{v}$ is founded, we obtain the real least-squares solution of (11):

$$\bar{\mathbf{q}} = \mathbf{D}^{-1}\mathbf{v} + \mathbf{z} = \mathbf{v} \oslash \mathbf{s} + \mathbf{z},$$

where $\oslash$ denotes element-wise division.

Denote $\bar{\mathbf{R}} := \mathbf{R}\mathbf{D}$. It is not difficult to verify that the BILS problem (11) is equivalent to

$$\min_{\mathbf{q} \in \mathbb{B}^m} \left\|\bar{\mathbf{R}}(\mathbf{q} - \bar{\mathbf{q}})\right\|_2^2. \quad (13)$$

Then, we can find the optimal solution or some sub-optimal solutions.

The algorithm which finds the box-constrained Babai point of (13) is described in Algorithm 1. In the algorithm,

$$\mathrm{clamp}(x; u, v) := \min(\max(x, u), v), \quad (14)$$

i.e., the point in the interval $[u, v]$ closest to x, Moreover, $\mathbf{c}(i)$ is the real-valued quantity computed for $\mathbf{q}(i)$ at step i, assuming that $\mathbf{q}(m), \mathbf{q}(m-1), \ldots, \mathbf{q}(i+1)$ have been fixed during back-substitution, and $\mathbf{q}(i)$ is then chosen as the integer in the constraint $\mathbb{B}$ closest to $\mathbf{c}(i)$.

3.4 $(K+1)$-Best Babai–Klein-Based Quantization

Babai's nearest plane algorithm is deterministic and makes greedy local decisions during back-substitution, so an early incorrect decision cannot be corrected later. Consequently, the algorithm may return a lattice point far from the closest lattice point, especially when the lattice basis matrix is ill-conditioned (or poorly reduced). The problem becomes more serious when the dimension of the lattice basis matrix is large. To address this limitation, Klein's sampling algorithm [Klein, 2000] introduces controlled randomness into the rounding process, enabling the algorithm to explore a richer set of lattice points while maintaining polynomial-time complexity.

Algorithm 1 Babai quantization with *JTA*

1: Compute $\mathbf{s}$ and $\mathbf{z}$
2: Compute the Cholesky factorization: $\widetilde{\mathbf{X}}^\top\widetilde{\mathbf{X}} + \lambda^2\mathbf{I} = \mathbf{R}^\top\mathbf{R}$
3: Solve the lower triangular system $\mathbf{R}^\top\mathbf{u} = \mathbf{A}^\top\mathbf{t}$ and the upper triangular system $\mathbf{R}\mathbf{v} = \mathbf{u}$.
4: Compute $\bar{\mathbf{q}} = \mathbf{v} \oslash \mathbf{s} + \mathbf{z}$
5: Compute $\bar{\mathbf{R}} = \mathbf{R}\mathbf{D}$ with $\mathbf{D} = \mathrm{diag}(\mathbf{s})$
6: $\mathbf{c}(m) = \bar{\mathbf{q}}(m)$
7: $\mathbf{q}(m) = \mathrm{clamp}(\lfloor \mathbf{c}(m) \rceil; 0, 2^{\mathrm{wbit}-1})$
8: **for** $i = m-1 : -1 : 1$ **do**
9: $\quad \mathbf{c}(i) = \bar{\mathbf{q}}(i) + \left(\sum_{j=i+1}^{m} \bar{\mathbf{R}}(i,j)(\bar{\mathbf{q}}(j) - \mathbf{q}(j))\right) / \bar{\mathbf{R}}(i,i)$
10: $\quad \mathbf{q}(i) = \mathrm{clamp}(\lfloor \mathbf{c}(i) \rceil; 0, 2^{\mathrm{wbit}} - 1)$
11: **end for**

Klein randomized rounding. Like Babai's original algorithm, Klein's original algorithm is for the unconstrained case. However, we can easily extend it to deal with the box constraint in our quantization application. In each back-substitution step, instead of deterministic rounding such as lines 7 and 10 in Algorithm 1, we sample $\mathbf{q}(i)$ from a distribution concentrated around $\mathbf{c}(i)$:

$$\Pr(\mathbf{q}(i) = \ell) = \frac{\exp\left(-\alpha\, \bar{\mathbf{R}}(i,i)\, (\mathbf{c}(i) - \ell)^2\right)}{\sum_{j=0}^{2^{\mathrm{wbit}}-1} \exp\left(-\alpha\, \bar{\mathbf{R}}(i,i)\, (\mathbf{c}(i) - j)^2\right)}, \quad \ell \in \mathbb{B}, \quad (15)$$

where $\alpha > 0$ controls the sampling temperature. Larger α approaches greedy Babai; smaller α encourages exploration. For this paper, we follow [Liu *et al.*, 2011] design choices where $\alpha = \ln(\rho)/\min_{1 \le i \le m} r_{ii}^2$ and ρ is the solution of $K = (e\rho)^{(2m/\rho)}$. We refer to the integer vector obtained by the extended Klein algorithm as a box-constrained Klein point.

By executing the extended Klein algorithm K times independently, we obtain K box-constrained Klein points. We then choose, among these K points and the box-constrained Babai point, the one with the minimum residual as a final sub-optimal solution to the BILS problem (13). We refer to sub-optimal solution as the $(K+1)$-best box-constrained Babai–Klein point.

To get some idea about how the value of K affects the residual of the $(K+1)$-best box-constrained Babai–Klein point, we give Figure 1, in which $K0$ means only the box-constrained Babai is used as the sub-optimal solution. We observe that typically larger K makes the objective smaller.

Parallel-Isolated $(K+1)$-best Klein-Babai

We propose **Parallel-Isolated $(K+1)$-best Klein-Babai (PI-Klein-Babai)**, a GPU kernel that evaluates K randomized Klein-Babai instances together with one deterministic greedy

	Method	L2-7B	L2-13B	L3-8B	Q3-0.6B	Q3-4B	Q3-8B	M-7B
	BF16	6.97/5.47	6.47/4.88	8.93/6.13	25.28/20.94	16.53/13.67	13.23/9.72	8.15/5.50
g128 W4A16	RTN	7.72/6.11	6.83/5.20	12.09/8.53	42.20/37.52	20.15/17.52	15.54/11.97	8.96/6.12
	GPTQ	**7.11**/5.62	6.56/4.99	9.41/6.54	30.79/25.20	16.73/13.66	13.51/10.08	8.29/5.63
	AWQ	7.16/**5.61**	6.56/**4.97**	9.40/6.54	27.99/25.89	17.18/18.43	13.51/10.02	8.28/5.61
	Ours(B)	7.15/5.63	6.56/4.99	9.38/6.52	28.35/23.48	18.20/13.97	13.42/9.97	8.27/5.60
	Ours(R)	7.14/5.62	**6.55**/4.98	9.35/6.50	28.07/23.64	16.44/13.84	13.40/9.95	8.27/5.59
	Ours	7.14/**5.61**	**6.55/4.97**	**9.33/6.48**	**26.84/22.80**	**16.32/13.54**	**13.39/9.94**	**8.23/5.58**
g128 W3A16	RTN	4e2/5e2	12.50/10.68	4e2/2e3	1e5/2e5	3e3/3e3	5e2/7e2	19.65/14.88
	GPTQ	7.92/6.44	6.99/5.43	12.25/8.35	43.60/41.54	18.16/15.19	14.34/11.01	9.10/6.27
	AWQ	7.90/**6.25**	6.95/**5.33**	11.63/8.19	39.60/36.88	19.75/16.83	14.97/11.23	8.89/6.07
	Ours(B)	7.88/6.33	6.98/5.41	11.52/8.19	47.23/42.75	25.46/20.36	14.44/11.11	8.87/6.06
	Ours(R)	7.87/6.34	**6.94**/5.39	11.43/8.16	44.09/40.19	19.52/14.91	14.37/11.05	8.86/**6.03**
	Ours	**7.86**/6.30	**6.94/5.33**	**11.40/7.87**	**35.03/31.75**	**17.01/12.89**	**14.30/10.98**	**8.84/6.03**
W4A16	GPTQ	7.37/5.83	6.70/5.16	10.30/7.37	37.03/33.64	17.34/14.34	13.99/10.65	8.53/5.81
	AWQ	7.34/5.83	6.68/5.06	9.44/7.10	30.07/25.89	18.83/16.35	14.29/10.77	10.24/5.88
	QUIP	12.54/10.44	6.71/5.15	9.84/6.85	51.41/47.10	182.16/126.55	19.19/14.85	9.69/6.65
	Ours(B)	7.14/5.64	6.57/4.98	9.39/6.52	28.35/23.48	17.30/14.30	13.41/9.97	8.28/5.62
	Ours(R)	7.11/5.62	**6.56**/4.97	9.38/6.50	28.11/23.04	17.30/14.30	13.41/9.96	8.27/5.60
	Ours	**7.10/5.60**	**6.56/4.96**	**9.33/6.48**	**26.94/22.61**	**17.25/14.27**	**13.38/9.92**	**8.24/5.59**
W3A16	GPTQ	9.83/8.33	8.04/6.52	29.42/23.81	90.23/90.98	23.16/21.39	18.10/16.46	10.90/8.01
	AWQ	15.62/15.51	8.17/6.45	17.60/11.84	78.52/79.44	30.31/33.19	18.36/14.97	10.27/7.65
	QUIP	27.47/28.05	7.17/5.57	11.50/8.31	121.26/112.09	23.02/19.74	30.39/19.78	9.30/6.61
	Ours(B)	7.88/6.30	6.97/5.42	11.52/8.17	47.23/42.75	22.95/19.65	25.46/20.36	8.78/6.09
	Ours(R)	7.87/6.29	6.97/5.41	11.35/8.12	43.94/39.77	22.90/19.63	18.07/16.48	8.77/6.07
	Ours	**7.75/6.22**	**6.96/5.40**	**11.38/8.04**	**35.56/32.95**	**22.87/19.56**	**18.05/14.75**	**8.72/6.06**

Table 1: Perplexity comparison across different models and quantization methods. For each entry, the left value is evaluated on C4 and the right value on WikiText-2. We denote our proposed methods as follows: **Ours(B)** represents Babai-Based, **Ours(R)** represents $(K+1)$-Best Babai–Klein-Based, and **Ours** represents $(K+1)$-Best Babai–Klein-Based with Joint Activation-Target optimization. The model families are abbreviated as: L2/L3 for Llama2/Llama 3, Q3 for the Qwen3 series, and M for Mistral. All activations are in BF16 format.

Babai instance in a single parallel pass, while keeping all $K+1$ residuals strictly isolated. Each instance owns a private residual slot along a batch dimension, and the look-ahead update from already-quantized coordinates to the remaining ones is fused across all $K+1$ instances into a single batched matrix multiplication. This recovers the throughput of a shared-buffer implementation without its correctness violation. Reserving the extra slot for the deterministic greedy Babai solution guarantees its inclusion among the $K+1$ candidates, and the final quantization is selected using the unified JTA score in Sec. 3.1. Klein-style JTA quantization, $(K+1)$-best JTA quantization, and detailed kernel design, buffer layout, and blocked update rules are shown in the full version

4 Experiments

To evaluate the effectiveness of our method, we conduct comprehensive experiments comparing it against commonly used layer-wise quantization baselines. We first describe the baseline methods, the large language models being quantized, as well as the evaluation metrics and datasets. We then present the experimental results, followed by a detailed analysis. To ensure fair comparisons, we adopt the default configurations of each baseline's repo whenever possible.

Baselines We evaluate our method against state-of-the-art quantization approaches, including AWQ [Lin *et al.*, 2024], GPTQ [Chen *et al.*, 2023], and QUIP [Chee *et al.*, 2024], and additionally report round-to-nearest as a naïve baseline for perplexity. Although theortically, the K can be any positive integer number, considering the trade-off between resource usage and performance gain Figure2, we set the K as 5 in our experiments. To ensure strong baselines, we enable activation ordering for GPTQ. Calibration is performed using 128 samples from the C4 [Raffel *et al.*, 2020a] dataset with a sequence length of 2048 tokens. For all baselines, only minimal modifications are made to accommodate the architectural structures of newer LLMs, while their original algorithms and configurations are kept unchanged.

Models We consider two major LLM families, Qwen [Yang *et al.*, 2025]and LLaMA [Touvron *et al.*, 2023; Grattafiori *et al.*, 2024], together with Mistral-7B [Jiang *et al.*, 2023]. This selection covers diverse model families, reasoning capabilities, and parameter sizes.

Metrics and Datasets To test the basic text generation ability, we use the perplexity to measure each quantized model on C4 [Raffel *et al.*, 2020a] and Wikitext2 [Raffel *et al.*, 2020b]. Then we further evaluate on the zero shot accuary on ARC [Clark *et al.*, 2018], boolq [Wang *et al.*, 2019], HellaSwag [Zellers *et al.*, 2019], PIQA [Bisk *et al.*, 2020] and WinoGrande [Sakaguchi *et al.*, 2021] For the last part, we test the reasoning ablility on GSM-8K [Cobbe *et al.*, 2021], GPQA-diamond [Rein *et al.*, 2023] and MBPP [Austin *et al.*, 2021]. Except the evalution of PPL, all other tasks are evaluated on LM-harness library [Gao *et al.*, 2024]

Perplexity Table 1 reports perplexity (PPL) under group-wise weight-only quantization with 16-bit activation. At 4-bit, GPTQ and AWQ perform reasonably on large LLaMA

Model	Method	4 bits							3 bits						
		ARC-C	ARC-E	BoolQ	Hella	PIQA	Wino	Average*	ARC-C	ARC-E	BoolQ	Hella	PIQA	Wino	Average*
Llama3-8B	BF16	51.71	80.89	81.77	60.56	78.84	73.40	71.20	51.71	80.89	81.77	60.56	78.84	73.40	71.20
	GPTQ	48.29	78.89	80.21	59.47	78.29	**73.32**	69.74	34.21	66.04	71.89	55.26	73.56	68.11	61.51
	AWQ	48.70	79.21	80.80	59.73	78.62	**73.32**	70.06	**43.86**	75.21	76.15	55.79	75.95	71.03	66.33
	QUIP	46.25	77.36	**81.93**	58.76	77.53	73.09	69.15	41.38	71.34	77.68	55.67	76.12	**71.67**	65.64
	O(B)	48.46	78.53	81.04	**60.00**	78.18	73.16	69.90	41.81	71.71	76.73	**56.71**	76.88	71.35	65.87
	O(R)	**49.32**	79.12	81.04	59.79	78.24	72.30	69.97	41.30	73.48	76.45	56.67	77.37	70.64	65.99
	O	49.15	**79.59**	81.13	59.70	**78.67**	72.69	**70.16**	42.92	**75.42**	**78.69**	55.73	**77.75**	70.32	**66.81**
Qwen3-4B	BF16	50.34	80.26	85.11	52.29	75.30	65.67	68.16	50.34	80.26	85.11	52.29	75.30	65.67	68.16
	GPTQ	47.57	77.90	82.43	50.12	**74.43**	62.43	65.81	39.33	70.92	**80.95**	**46.81**	70.92	**61.64**	61.76
	AWQ	46.29	77.32	83.23	50.37	74.10	**62.88**	65.70	40.27	68.27	81.19	46.02	70.87	60.38	61.17
	QUIP	39.25	71.09	80.73	43.64	70.24	59.91	60.81	24.15	47.81	67.86	33.72	62.79	51.14	47.91
	O(B)	41.81	74.79	83.46	47.61	73.12	58.33	63.19	35.95	58.68	69.20	40.65	67.74	50.83	53.84
	O(R)	44.28	74.03	82.97	49.28	72.80	61.17	64.09	37.79	65.35	70.58	43.39	67.25	53.99	56.39
	O	**46.65**	**78.27**	**84.31**	**49.85**	74.21	62.35	**65.94**	**41.50**	**71.41**	80.95	45.61	**72.64**	60.06	**62.03**
Qwen3-8B	BF16	55.38	83.50	86.73	57.12	76.50	67.88	71.19	55.38	83.50	86.73	57.12	76.50	67.88	71.19
	GPTQ	**54.27**	**82.74**	**86.73**	**56.62**	76.22	66.77	70.56	44.28	74.62	83.33	53.88	75.95	67.56	66.60
	AWQ	53.67	82.37	85.93	55.91	75.79	**68.75**	70.40	**50.56**	**80.93**	**84.43**	52.80	75.90	67.25	**68.65**
	QUIP	45.48	75.42	81.99	44.72	69.53	62.67	63.30	40.27	71.93	77.37	43.22	71.33	59.83	60.66
	O(B)	53.75	82.20	86.02	56.39	76.22	68.03	70.44	47.69	77.61	84.46	53.76	74.76	67.35	67.61
	O(R)	53.41	82.36	86.02	56.34	76.06	68.82	70.50	47.59	77.62	84.04	53.88	74.59	67.46	67.53
	O	54.18	82.62	86.45	55.70	**76.39**	68.59	**70.66**	47.27	78.41	84.39	**54.23**	**76.27**	**67.90**	68.08

Table 2: **Zero-shot accuracy comparison on six common sense reasoning tasks across Llama3-8B, Qwen3-4B, and Qwen3-8B.** The evaluation is conducted under **4-bit** and **3-bit** quantization settings. We denote our proposed methods as: **O(B)** represents Babai-Based, **O(R)** represents $(K+1)$-Best Babai–Klein-Based, and **Ours** represents $(K+1)$-Best Babai–Klein-Based with Joint Activation-Target optimization.The unquantized **BF16** baseline is shaded in gray. For quantized methods, **bold** indicates the best result and underlining denotes the second-best.

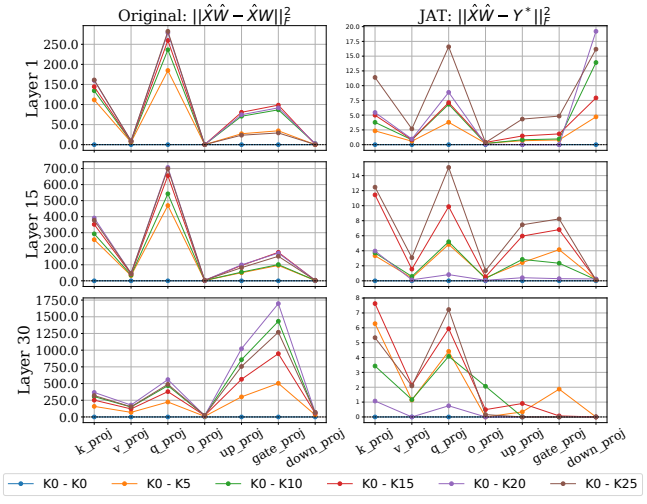

Figure 1: Layer-wise MSE between the objectives in Eq. (1) and our proposed JTA across Layers 1, 15, and 30 for all linear modules under varying K settings.

models but degrade on harder architectures such as Qwen-0.6B and LLaMA-3-8B, while RTN frequently diverges. Our method consistently achieves the lowest or near-lowest perplexity across all model families, with clear gains in challenging regimes. This advantage is more pronounced at 3-bit, where prior methods often suffer sharp drops or catastrophic failures, whereas ours remains stable with smooth degradation relative to BF16. When group quantization is disabled (group size = 0), baselines exhibit increased quantization error and numerical instability, especially on smaller models, while our method remains stable and preserves low perplexity across all architectures.

Zero-shot accuracy Table 2 compares BF16 and low-bit PTQ methods across three model families under both 4-bit and 3-bit settings. At 4 bits, all methods remain relatively close to BF16, but our approach consistently achieves the strongest or second-strongest performance across most tasks and models, yielding the highest average accuracy for all three backbones. Notably, the gains are more pronounced on reasoning-heavy benchmarks such as ARC-C, ARC-E, and Hella, indicating improved robustness beyond simple lexical or pattern-matching tasks. When moving to the more challenging 3-bit regime, performance gaps widen substantially: GPTQ and QUIP exhibit significant degradation, particularly on ARC and Hella, while our method degrades more gracefully and maintains clear advantages in average performance. This trend is consistent across model scales, from Qwen-3-4B to Qwen-3-8B, suggesting that the proposed approach scales favorably and is less sensitive to aggressive quantization. Overall, the results demonstrate that our method not only preserves accuracy under moderate compression but also provides superior stability in extreme low-bit settings, highlighting its effectiveness as a general and robust PTQ solution.

Reasoning Table 3 reports reasoning accuracy on GSM8K, GPQA, and MBPP under 4-bit quantization across three model families. Compared to GPTQ, AWQ, and QUIP, our method consistently achieves the highest average accuracy

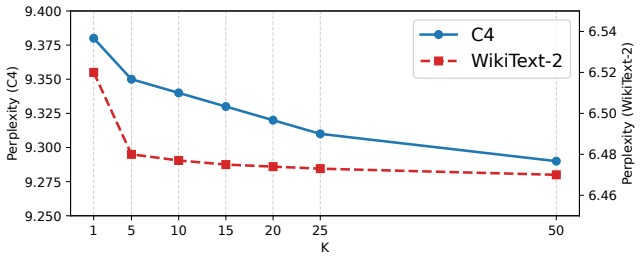

Figure 2: **Ablation study on the candidate size K.** We evaluate the perplexity on C4 and WikiText-2 datasets using Llama-3-8B with 4-bit quantization (group size 128). The results demonstrate the impact of increasing K.

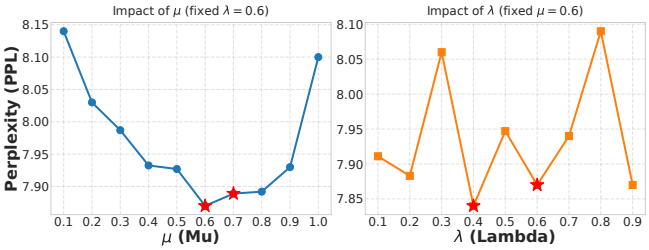

Figure 3: **Sensitivity analysis of hyperparameters μ and λ.** Evaluated on WikiText-2 (calibrated on C4), the plots show the perplexity trend when varying one parameter while fixing the other at 0.6. The U-shaped curve for μ (left) confirms the necessity of balancing the two objectives, while λ (right) shows 0.6 as a robust operating point.

for all backbones, with particularly strong gains on GSM8K and GPQA, which require multi-step numerical and factual reasoning. While AWQ remains competitive on MBPP for some models, its performance is less consistent across tasks, leading to lower overall averages. QUIP exhibits severe degradation on reasoning benchmarks, including complete failure on MBPP for Qwen models, highlighting its instability under reasoning-intensive workloads. In contrast, our approach maintains balanced performance across arithmetic, knowledge-intensive, and program synthesis tasks, closely tracking BF16 accuracy despite aggressive quantization. These results indicate that the proposed method better preserves structured reasoning capabilities under low-bit constraints, rather than overfitting to a single task type or model family.

Ablations We investigate the impact of the hyperparameter K by varying it within $\{1, 5, \ldots, 50\}$ under the setting of 4-bit Llama-3-8B with a group size of 128. As illustrated in Fig. 2, the perplexity on both WikiText-2 and C4 exhibits a **significant drop** at $K = 5$. Beyond this point, while the performance continues to improve up to $K = 50$, the marginal gains become negligible. Based on these observations, we select $K = 5$ as the default configuration for our method.

We analyze the impact of μ and λ on WikiText-2 in Fig. 3. For μ, performance is optimized at 0.6 but degrades at the boundaries $(0.1, 1.0)$. This confirms that neither Eq. 1 nor Eq. 4 alone is sufficient; a balanced combination is essential. For λ, despite higher variance, setting $\lambda = 0.6$ yields a robust minimum. Consequently, we adopt $(\mu = 0.6, \lambda = 0.6)$ as the default configuration for 3 bits setting. Similarly, we adopt $(\mu = 0.1, \lambda = 0.2)$ for 4 bits setting.

5 Conclusion

We presented a unified post-training quantization framework that reformulates layer-wise weight quantization as a structured integer optimization problem and solves it using a joint objective with Babai-type decoding. By integrating compensation-aware targets and controlled randomness, the proposed method consistently outperforms existing PTQ approaches across a wide range of models, tasks, and bit-widths. Extensive experiments on both general evaluation benchmarks and reasoning-intensive tasks demonstrate that our approach preserves accuracy more effectively under aggressive low-bit quantization, while remaining stable across model scales. These results suggest that principled optimization-based formulations provide a robust foundation for future low-bit quantization methods, bridging classical lattice decoding techniques and modern large language model compression.

Model	Method	GSM8K	GPQA	MBPP	Avg
LLaMA3 -8B	BF16	51.86	35.98	48.40	45.41
	GPTQ	45.03	29.27	45.00	39.77
	AWQ	44.42	32.32	**46.40**	41.05
	QUIP	42.30	29.87	40.40	37.52
	Ours	**48.22**	**32.93**	44.80	**41.98**
Qwen3 -4B	BF16	83.70	38.38	62.40	61.49
	GPTQ	57.71	31.81	27.60	39.77
	AWQ	80.15	33.33	**57.80**	57.09
	QUIP	5.69	25.75	0.00	10.48
	Ours	**84.29**	**36.36**	56.80	**59.15**
Qwen3 -8B	BF16	88.48	33.33	65.00	62.27
	GPTQ	86.80	**36.86**	60.20	61.29
	AWQ	85.44	32.83	**62.80**	60.36
	QUIP	64.40	31.31	0.00	31.90
	Ours	**87.87**	34.34	63.20	**61.80**

Table 3: **Reasoning accuracy (%) on GSM8K, GPQA, and MBPP.** 4-bit (g128) quantization. **Bold** = best, underline = second-best.

Limitations Our current framework does not yet incorporate weight permutation (e.g., GPTQ) or dynamic scaling, which are promising for future integration. Additionally, we presently use fixed hyperparameters (μ, λ) and candidate size K for all layers. Future work will explore layer-wise adaptive strategies, specifically assigning distinct μ, λ and varying the number of random candidates per layer to better match the sensitivity of different model components.

Contribution

Xinyu Wang and Ziyu Zhao contributed equally.

References

[Achiam *et al.*, 2023] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni

Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[Agrell *et al.*, 2002] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. Closest point search in lattices. *IEEE Trans. Inf. Theory*, 48(8):2201–2214, 2002.

[Arai and Ichikawa, 2025] Yamato Arai and Yuma Ichikawa. Quantization error propagation: Revisiting layer-wise post-training quantization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[Arai and Ichikawa, 2026] Yamato Arai and Yuma Ichikawa. Quantization error propagation: Revisiting layer-wise post-training quantization, 2026.

[Austin *et al.*, 2021] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021.

[Babai, 1986] László Babai. On Lovász'lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[Barbero and Thompson, 2006] Luis G Barbero and John S Thompson. Performance analysis of a fixed-complexity sphere decoder in high-dimensional MIMO systems. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 4, pages IV–IV. IEEE, 2006.

[Bisk *et al.*, 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, pages 7432–7439, 2020.

[Chang and Han, 2008] X.-W. Chang and Q. Han. Solving box-constrained integer least squares problems. *IEEE Trans. Wireless Commun.*, 7(1):277–287, 2008.

[Chang *et al.*, 2005] X.-W. Chang, X. Yang, and T. Zhou. MLAMBDA: A modified LAMBDA method for integer least-squares estimation. *Journal of Geodesy*, 79(9):552–565, 2005.

[Chang *et al.*, 2024] Xiao-Wen Chang, Zhilong Chen, and Jinming Wen. An extended Babai method for estimating linear model based integer parameters. *Econometrics and Statistics*, 29:238–251, 2024.

[Chang, 2022] X.-W. Chang. MILES: MATLAB package for solving mixed integer least squares problems. *School of Computer Science, McGill University, http://www.cs.mcgill.ca/~chang/software/MILES.php*, 2022.

[Chee *et al.*, 2024] Jerry Chee, Yaohui Cai, Volodymyr Kuleshov, and Christopher De Sa. QuIP: 2-bit quantization of large language models with guarantees, 2024.

[Chen *et al.*, 2023] Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

[Chen *et al.*, 2025] Jiale Chen, Yalda Shabanzadeh, Elvir Crnčević, Torsten Hoefler, and Dan Alistarh. The geometry of LLM quantization: GPTQ as Babai's nearest plane algorithm, 2025.

[Clark *et al.*, 2018] 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.

[Cobbe *et al.*, 2021] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

[Comanici *et al.*, 2025] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities, 2025.

[Damen *et al.*, 2003] M. O. Damen, H. E. Gamal, and G. Caire. On maximum likelihood detection and the search for the closest lattice point. *IEEE Trans. Inf. Theory*, 49(10):2389–2402, 2003.

[Emde-Boas, 1981] PV Emde-Boas. Another NP-complete partition problem and the complexity of computing short vectors in a lattice. *Repport 81-04, Mathematisch Institut, Amsterdam, Netherlands*, 1981.

[Frantar *et al.*, 2023] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers, 2023.

[Gao *et al.*, 2024] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024.

[Grattafiori *et al.*, 2024] 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, 2024.

[Jiang *et al.*, 2023] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.

[Klein, 2000] Philip Klein. Finding the closest lattice vector when it's unusually close. In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '00, page 937–941, USA, 2000. Society for Industrial and Applied Mathematics.

[Lenstra *et al.*, 1982] A.K. Lenstra, H.W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515–534, 1982.

[Lin *et al.*, 2024] 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 llm compression and acceleration, 2024.

[Liu *et al.*, 2011] S. Liu, C. Ling, and D. Stehle. Decoding by sampling: A randomized lattice algorithm for bounded distance decoding. *IEEE Trans. Inf. Theor.*, 57(9):5933–5945, September 2011.

[Micciancio and Goldwasser, 2002] Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, March 2002.

[Raffel *et al.*, 2020a] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[Raffel *et al.*, 2020b] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[Rein *et al.*, 2023] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark, 2023.

[Sakaguchi *et al.*, 2021] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[Schnorr and Euchner, 1994] C. Schnorr and M. Euchner. Lattice basis reduction: improved practical algorithms and solving subset sum problems. *Math Program*, 66:181–191, 1994.

[Shao *et al.*, 2024] Wenqi Shao, Mengzhao Chen, Zhaoyang Zhang, Peng Xu, Lirui Zhao, Zhiqian Li, Kaipeng Zhang, Peng Gao, Yu Qiao, and Ping Luo. OmniQuant: Omnidirectionally calibrated quantization for large language models, 2024.

[Teunissen, 1993] P. J. G Teunissen. Least-squares estimation of the integer GPS ambiguities. In *Invited lecture, Section IV theory and methodology, IAG general meeting, Beijing, China, August*, 1993.

[Teunissen, 1995] P. J. G Teunissen. The least-squares ambiguity decorrelation adjustment: a method for fast GPS integer ambiguity estimation. *Journal of Geodesy*, 70(1-2):65–82, 1995.

[Teunissen, 2003] Peter J. G. Teunissen. Towards a unified theory of gnss ambiguity resolution. *Journal of Global Positioning Systems*, 2(1):1–12, 2003.

[Touvron *et al.*, 2023] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[Verdú, 1989] Sergio Verdú. Computational complexity of optimum multiuser detection. *Algorithmica*, 4(3):303–312, 1989.

[Wang *et al.*, 2019] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[Wang *et al.*, 2025] Xinyu Wang, Vahid Partovi Nia, Peng Lu, Jerry Huang, Xiao-Wen Chang, Boxing Chen, and Yufei Cui. PoTPTQ: A two-step power-of-two post-training for llms, 2025.

[Yang *et al.*, 2025] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, et al. Qwen3 technical report, 2025.

[Zellers *et al.*, 2019] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

[Zhao *et al.*, 2025] Jiaqi Zhao, Ming Wang, Miao Zhang, Yuzhang Shang, Xuebo Liu, Yaowei Wang, Min Zhang, and Liqiang Nie. Benchmarking post-training quantization in LLMs: Comprehensive taxonomy, unified evaluation, and comparative analysis, 2025.