LAZY KERNEL QR: ADAPTIVE RANDOMIZED QR FACTORIZATION VIA SKETCHED TELEMETRY

Anonymous authors
Paper under double-blind review

ABSTRACT

QR decomposition underlies randomized range finders, Krylov solvers, low-rank adapters and KV-cache compression, and its dominant cost is the $\mathcal{O}(nl^2)$ work of maintaining an explicitly orthonormal Q. We introduce **Lazy Kernel QR (LKQR)**, which adapts the number of orthogonalization corrections per column to the *local* geometry of the basis through a low-dimensional *sketched kernel* used as a telemetry signal. Because the telemetry lives in $\mathbb{R}^k$ ($k \ll n$) and is maintained by a rank-one bordered update, the decision overhead is independent of n. We prove exact reconstruction and condition-number and orthogonality bounds that depend on the cumulative laziness T; for a decaying schedule $t(i) = \alpha/i^{1+\beta}$ with $\alpha/\beta < 1$, $T \leq \alpha/\beta$ is independent of n and l. Across six machine-learning pipelines—ogbn-arxiv node classification (three GNN backbones, four eigenvector counts), GaLore gradient projection on six pretrained model families, randomized PCA on three datasets, QR-LoRA fine-tuning of three vision backbones, GPM continual learning and Qwen-2.5 KV-cache compression—LKQR removes $4.2\times$–$44.1\times$ of MGS's orthogonalization FLOPs, measured by instrumenting every primitive rather than a closed-form formula, at no measurable downstream loss; on a continual-learning sequence it stays within 3.2 points of the exact orthogonalizer, where fixed truncation loses a third of the accuracy. Finally, for consumers that use the emitted basis itself as a metric the exact orthogonalizer remains the right choice, and LKQR's advantage there is a $1.8\times$–$7.9\times$ FLOPs reduction rather than a better basis. LKQR is stable where normal-equation methods (CholeskyQR) break down.

1 INTRODUCTION

QR decomposition factors $A \in \mathbb{R}^{n\times l}$ into $A = QR$ with $Q^\top Q = I$ and R upper triangular. It is a bottleneck in modern numerical pipelines: a $t \times l$ range finder $Y = A\Omega$ followed by QR of Y underlies randomized SVD (Martinsson & Tropp, 2020; Tropp, 2012); the basis expansions of iterative linear solvers (Saad, 2003), and the inner loop of Riemannian optimization (Absil et al., 2008) require repeated QR factorizations; and orthonormal weight matrices are central to parameter-efficient fine-tuning of large models (Yang et al., 2025). In the typical tall-and-skinny regime $n \gg l$, the cost is dominated by the $\mathcal{O}(nl^2)$ work needed to keep Q stable, and this cost must be paid whenever a new column arrives.

Existing remedies fall into two extremes. *Randomized Gram–Schmidt* (RGS) (Balabanov & Grigori, 2022) computes the projection coefficients in a low-dimensional sketch but still performs *all* $l(l-1)/2$ ambient-space corrections; the asymptotic cost remains $\mathcal{O}(nl^2)$. *Sketch-and-select* QR (SSQR) (Güttel & Simunec, 2024) caps the number of corrections per column at a fixed K based on coefficient magnitude, which is fast but fragile: the magnitude rule is blind to true geometry in $\mathbb{R}^n$ and diverges on a substantial fraction of systems. This produces a *computational–stability trilemma*: full orthogonalization is robust but slow; fixed truncation is fast but brittle; and evaluating orthogonality in real time to steer a truncation is traditionally as expensive as the orthogonalization itself.

Contributions. We resolve this trilemma with **Lazy Kernel QR (LKQR)**, which delegates the "should I orthogonalize against column j?" decision to a low-dimensional *sketched kernel*. The

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.

kernel is a faithful surrogate for ambient correlations *by construction* (Section 3), so the decision cost is $\mathcal{O}(1)$ amortized and independent of n. Specifically:

(i) An adaptive thresholding scheme (Section 3) whose telemetry lives entirely in the sketch space, with an $\mathcal{O}(1)$-amortized rank-one bordered kernel update. The scheme interpolates between MGS and zero-orthogonalization through a single threshold t.

(ii) Reconstruction exactness and condition-number / orthogonality bounds (Section 4). The bounds depend on the cumulative laziness T. For a decaying schedule $t(i) = \alpha/i^{1+\beta}$ with $\alpha/\beta < 1$, $T \leq \alpha/\beta$ is independent of n and l, so the guarantees hold for arbitrary basis dimensions. We explicitly delineate this provable regime from the practical constant-threshold Pareto default.

(iii) Reproducible evaluation on six machine-learning pipelines spanning graphs, large language models and vision (Section 5): Laplacian positional encoding for ogbn-arxiv node classification (three GNN backbones $\times$ four eigenvector counts), GaLore-style gradient projection on six pre-trained model families, randomized PCA on three image datasets at three ranks, QR-LoRA fine-tuning of three vision backbones, gradient-projection-memory continual learning on Split-CIFAR-100, and Qwen-2.5 KV-cache compression. FLOPs are measured by instrumenting every primitive the algorithms execute. On the four core applications LKQR reduces orthogonalization FLOPs by $4.2\times$ to $44.1\times$ relative to MGS at no measurable downstream loss (test accuracy within 0.29 pp, subspace projection error identical to all reported digits, fine-tuning accuracy within 0.15 pp), and the KV-cache study delimits where exact orthogonalization remains necessary.

2 Background and Related Work

2.1 Randomized sketching and randomized QR

A sketching matrix $\Theta \in \mathbb{R}^{k\times n}$ ($k \ll n$) is an ϵ-subspace embedding for $\mathcal{S}$ if for all $x, y \in \mathcal{S}$,

$$|\langle x, y\rangle - \langle \Theta x, \Theta y\rangle| \leq \epsilon \|x\| \|y\| . \tag{1}$$

Oblivious embeddings (Gaussian, SRHT, sparse sign / CountSketch) achieve this with $k = \mathcal{O}(l/\epsilon^2)$ (Woodruff, 2014; Ailon & Liberty, 2009). Throughout we use the sparse sign sketch (Clarkson & Woodruff, 2017; Tropp, 2012), which applies in $\mathcal{O}(n)$ time.

Randomized Gram–Schmidt (RGS) (Balabanov & Grigori, 2022) replaces ambient inner products with a sketched least-squares solve: at column i it solves $\min_y \|S_{:,1:i-1}y - S_{:,i}\|$ and then applies the full ambient update $Q_{:,i} \leftarrow Q_{:,i} - Q_{:,1:i-1}y$. The inner products leave $\mathbb{R}^n$, but all $l(l-1)/2$ ambient updates remain, so RGS still costs $\mathcal{O}(nl^2)$.

Sketch-and-select QR (SSQR) (Güttel & Simunec, 2024) computes the same sketched coefficients y but applies only the K largest-magnitude entries. It is fast but non-adaptive; its fixed budget ignores genuine, potentially distant correlations in $\mathbb{R}^n$, and the magnitude rule carries no correctness certificate.

CholeskyQR2 (Fukaya et al., 2014) and *Shifted CholeskyQR3* (Fukaya et al., 2020) are batch algorithms that form the Gram matrix $A^\top A$ and rely on BLAS-3 kernels, making them very fast for well-conditioned matrices. However, they square the condition number ($\kappa(A^\top A) = \kappa(A)^2$), so CholeskyQR2 breaks down when $\kappa(A) \gtrsim 1/\sqrt{\epsilon_{mach}} \approx 6.7 \times 10^7$; Shifted CholeskyQR3 adds a diagonal shift to recover from this. Both are batch algorithms and inapplicable to streaming or incremental settings, which is the regime we target.

2.2 Selective reorthogonalization and Kaczmarz methods

Selective orthogonalization (Parlett & Scott, 1979) and partial reorthogonalization (Simon, 1984) decide, at each Krylov iteration, whether to perform a *full* reorthogonalization pass against all previous basis vectors, detecting loss of orthogonality from quantities (Ritz residuals, estimated drift) computed in the *ambient* space at $\mathcal{O}(n)$ cost per step. LKQR differs in three ways: its telemetry lives entirely in the $\mathbb{R}^k$ sketched space with $\mathcal{O}(1)$-amortized overhead; when triggered it executes *column-selective* corrections rather than a full pass; and it is a standalone factorization for general matrices, not a component of a short-recurrence Krylov sequence. There is also a structural link to *row-action* / Kaczmarz methods (Strohmer & Vershynin, 2009; Nutini et al., 2016): a single

Gram–Schmidt correction $q \leftarrow q - (q_j^\top q)q_j$ is a Kaczmarz projection on the orthogonality equations. LKQR is nevertheless a one-pass factorization emitting explicit factors (Q, R) with $A = QR$ exactly (Theorem 1), tracking pairwise correlations via the sketched kernel and tying its threshold to non-asymptotic conditioning bounds rather than an asymptotic convergence-rate selector.

3 The Lazy Kernel QR Framework

3.1 Intuition: sketched telemetry

The redundancy in full orthogonalization is *spatially heterogeneous*: only a few column pairs are numerically significant for global stability. Under Eq. equation 1, the ambient correlation is bounded by the sketched one:

$$|\langle Q_{:,i}, Q_{:,j}\rangle| \le |\langle S_{:,i}, S_{:,j}\rangle| + \epsilon \, \|Q_{:,i}\| \, \|Q_{:,j}\|, \qquad S = \Theta Q, \tag{2}$$

so the sketched Gramian $S^\top S$ is a faithful surrogate for the geometry of the basis. LKQR therefore adopts a "lazy" rule: orthogonalize column i against column $j < i$ *only when* the sketch signals a loss of stability. Concretely, column i is deemed sufficiently orthogonalized when

$$\left\| S_{:,i}^\top S_{:,1:i-1} \right\|_\infty \le t(i) \, \|S_{:,i}\|, \tag{3}$$

where $t(i) > 0$ is a per-column tolerance. Figure 1 illustrates the resulting framework.

Three design choices deserve justification. First, the criterion is *scale-invariant*: it compares a correlation against the current column norm, so it is unaffected by the (arbitrary) scaling of the input columns. Second, it is an ℓ_∞ criterion: it triggers a correction as soon as *any* single sketched correlation exceeds the tolerance, which is the right notion of safety for a Gram matrix, because by Gershgorin's theorem the spectral deviation of $S^\top S$ from the identity is controlled by the largest row-wise sum of absolute off-diagonal entries. Third, and this is what makes the approach practical, the criterion is evaluated in $\mathbb{R}^k$, so its cost does not grow with n. As for every sketch-based method, the localization inherits the embedding's failure probability δ: the sketch can declare a column orthogonal when it is not. Lemma 1 quantifies the resulting transfer of correlations between $\mathbb{R}^n$ and $\mathbb{R}^k$, and Appendix L reports an empirical false-skip rate below 2.2% over the threshold range we use; because the decision is checked against a certificate rather than a fixed budget, a false skip is self-correcting in the following columns, which is why the downstream metrics of Section 5 are unaffected.

An important conceptual point, proved as Corollary 1, is that the lazy rule can only affect the *orthogonality* of Q, never the subspace it spans: the telemetry does not approximate the factorization, only the numerical quality of the basis that represents it. This is why LKQR can be dropped into subspace-based pipelines at no accuracy cost while still reporting large FLOPs savings, and why the upstream decision is safe to make with an approximate (sketched) signal.

3.2 Efficient kernel updates

Naively re-projection per sub-step would cost $\mathcal{O}(ki^2)$. We instead maintain the *sketched kernel matrix* $C = S^\top S \in \mathbb{R}^{l \times l}$ and a correlation proxy $\omega_{1:i-1} = S_{:,i}^\top S_{:,1:i-1}$. When column i is orthogonalized against column j with coefficient $R_{j,i}$, the proxy obeys the rank-one recurrence

$$\omega_{1:i-1} \leftarrow \omega_{1:i-1} - R_{j,i} \, C_{j,1:i-1}, \qquad S_{:,i} \leftarrow S_{:,i} - R_{j,i} S_{:,j}, \tag{4}$$

which costs $\mathcal{O}(i)$; afterwards C is updated by one symmetric bordered row/column in $\mathcal{O}(ki)$. Because $k \ll n$, the decision overhead is negligible and, crucially, independent of the ambient dimension.

The recurrence is exact, not an approximation: it follows from bilinearity of the inner product and the fact that the earlier columns are frozen while column i is processed, and it is what allows the decision loop to avoid touching $\mathbb{R}^n$ after the initial sketch (Lemma 4, Appendix G). The kernel itself is grown by a single symmetric bordered row/column, since $C_{1:i-1,i} = S_{:,1:i-1}^\top S_{:,i}$ and $C_{i,i} = \|S_{:,i}\|^2$, rather than by a full Gram product; Table 8 (Appendix G) lists the per-column cost of every step and separates the ambient part from the telemetry part. The auxiliary QR of S used to solve the sketched least-squares problem is maintained incrementally in $\mathbb{R}^k$ (lines 12–16), so the coefficients y are available at $\mathcal{O}(ki)$ cost per column without ever forming an ambient Gram matrix.

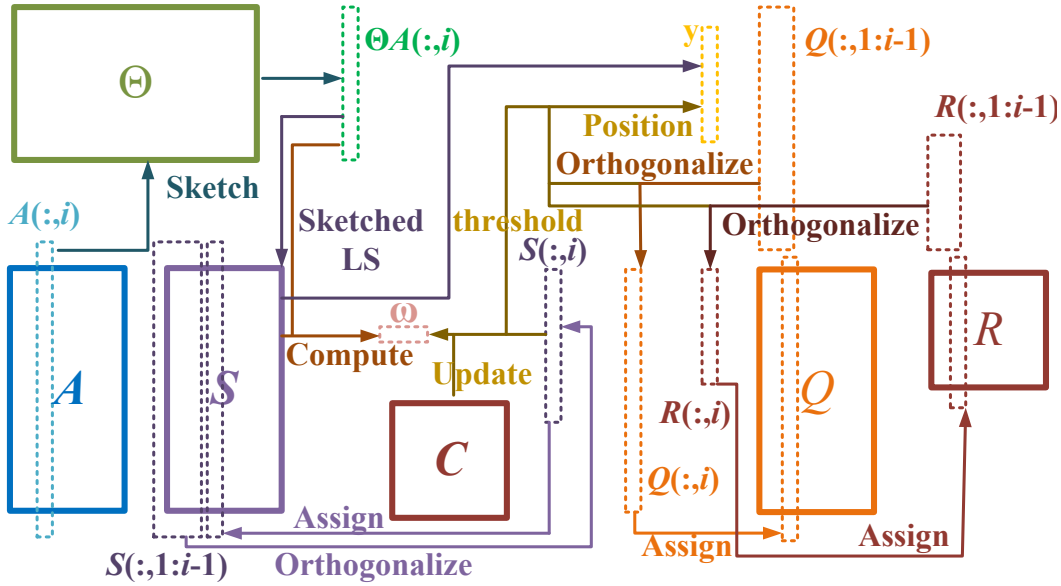


Figure 1: The LKQR framework. Each ambient column is sketched once ($S = \Theta Q$); the greedy telemetry monitors the sketched correlations and triggers ambient rank-one corrections only for the columns that actually threaten stability. The sketched kernel $C = S^\top S$ is maintained incrementally, so each decision costs $\mathcal{O}(k)$ instead of $\mathcal{O}(n)$.

Algorithm 1 Lazy Kernel QR (LKQR)

Require: $A \in \mathbb{R}^{n\times l}$, sketch Θ, threshold schedule $t(i)$.
Ensure: $Q \in \mathbb{R}^{n\times l}$, $R \in \mathbb{R}^{l\times l}$ with $A = QR$.
Initialize $S \in \mathbb{R}^{k\times l}$, $C \in \mathbb{R}^{l\times l}$, $\bar{Q} \in \mathbb{R}^{k\times l}$, $\bar{R} \in \mathbb{R}^{l\times l}$.
$Q_{:,1} \leftarrow A_{:,1}$; $S_{:,1} \leftarrow \Theta Q_{:,1}$; $R_{1,1} \leftarrow \|S_{:,1}\|$; normalize.
for $i = 2$ **to** l **do**
  $Q_{:,i} \leftarrow A_{:,i}$; $S_{:,i} \leftarrow \Theta Q_{:,i}$
  solve $\bar{R}_{1:i-1,1:i-1} y = \bar{Q}^\top_{:,1:i-1} S_{:,i}$ {sketched LS}
  $R_{:,i} \leftarrow \mathbf{0}$; $\omega \leftarrow S^\top_{:,i} S_{:,1:i-1}$
  for $m = 1$ **to** $i-1$ **do**
    $j \leftarrow \arg\max_p |\omega_p|$ {greedy dominant selection}
    $R_{j,i} \leftarrow y_j$; $\omega \leftarrow \omega - R_{j,i} C_{j,1:i-1}$
    $S_{:,i} \leftarrow S_{:,i} - R_{j,i} S_{:,j}$; $Q_{:,i} \leftarrow Q_{:,i} - R_{j,i} Q_{:,j}$
    if $\|\omega\|_\infty \le t(i)\,\|S_{:,i}\|$ **then**
      break {lazy pruning}
    end if
  end for
  $R_{i,i} \leftarrow \|Q_{:,i}\|$; normalize $Q_{:,i}$, $S_{:,i}$ by $R_{i,i}$ {true ambient column norm}
  $C_{i,1:i} \leftarrow S^\top_{:,i} S_{:,1:i}$; $C_{1:i-1,i} \leftarrow C^\top_{i,1:i-1}$
  $\bar{Q}_{:,i} \leftarrow S_{:,i}$; MGS against $\bar{Q}_{:,1:i-1}$ into $\bar{R}_{:,i}$
end for
return Q, R

3.3 Complexity and the algorithmic spectrum

Let $\bar{\iota}_i$ be the number of ambient corrections performed for column i. LKQR costs $\mathcal{O}(n\bar{\iota}_i + ki + i^2)$ per column. In the RandNLA regime $i \le l \le k \ll n$, the $\mathcal{O}(ki + i^2)$ telemetry terms are dominated by $\mathcal{O}(n\bar{\iota}_i)$, so the leading cost is $\mathcal{O}\big(n \sum_i \bar{\iota}_i\big)$ versus $\mathcal{O}(nl^2)$ for classical QR. Table 9 places the four algorithms side by side.

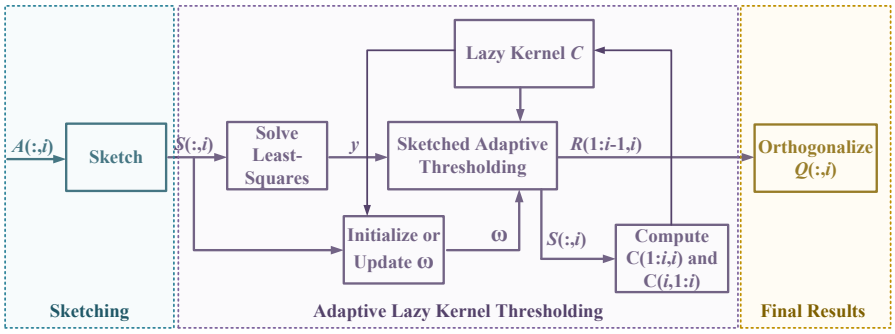


Figure 2: Data-flow architecture of one LKQR column update. The input column $A_{:,i}$ enters the sketch operator and the ambient update path; the sketched kernel C is the only side-channel state required for the thresholding decision, and the auxiliary QR of S lives entirely in $\mathbb{R}^k$.

Because $\bar{\iota} \le l(l-1)/2$ with equality only in the limit $t \to 0$ (Proposition 1), LKQR interpolates the whole spectrum of sequential orthogonalizers through a single scalar schedule:

- $t(i) \to 0$: every correction is performed. With $\Theta = I$ the algorithm reduces exactly to modified Gram–Schmidt, at cost $\Theta(nl^2)$.
- $t(i) = t$ constant and small: LKQR performs a constant number of corrections per column, at cost $\Theta(nl)$ *with* control of the orthogonality defect, because the threshold keeps the off-diagonal entries of $S^\top S$ bounded (Theorem 2).
- $t(i) \ge 1$: no correction is ever triggered, the emitted basis is simply the (unnormalized) input columns, and the cost drops to $\Theta(nl)$ with no stability guarantee.

The SSQR budget K plays a superficially similar role but is not interchangeable: K caps the corrections *unconditionally* and must be tuned to each matrix's spectrum, whereas t states a property of the geometry that the algorithm checks as it goes. Thus the single best K is matrix-dependent, whereas one value of t serves every configuration in Section 5.

Remark 1 (The role of the telemetry term). The non-ambient terms $\mathcal{O}(kl^2 + l^3)$ are independent of n and are asymptotically negligible, but at moderate n they are not negligible in absolute terms. The measured FLOPs of Section 5 include the telemetry component, and Appendix H reports its size for every application. The asymptotic statement is that LKQR's advantage in the n-dependent part is a factor $l(l-1)/(2\bar{\iota})$, which grows without bound as $n/l \to \infty$.

4 Theoretical Guarantees

Throughout, Θ is an (ϵ, δ, l)-oblivious subspace embedding for the column space of A; proofs are deferred to Appendix B. The first result is immediate from the algorithm's structure.

Theorem 1 (Exact reconstruction)**.** *In exact arithmetic, LKQR returns factors satisfying $A = QR$ exactly, regardless of the pruning.*

Pruning therefore does not change *what* LKQR computes, only the numerical quality of the emitted basis. The following consequence is what makes LKQR a drop-in primitive for subspace-based pipelines; it also predicts the empirical observation, reported in Section 5.2, that the projection error of all four orthogonalizers is identical to the last digit.

Corollary 1 (Column spaces are preserved, not approximated)**.** *For every i, $\mathrm{span}(Q_{:,1:i}) = \mathrm{span}(A_{:,1:i})$; in particular $\mathrm{span}(Q) = \mathrm{span}(A)$ whenever $\mathrm{rank}(A) = l$. Hence every downstream functional that depends only on $\mathrm{span}(Q)$—the orthogonal projector $QQ^\dagger$, principal angles, or a least-squares fit of A onto $\mathrm{span}(Q)$—takes the same value for LKQR, MGS, RGS and SSQR on the same range finder.*

Proposition 1 (LKQR never does more ambient work)**.** *With $\bar{\iota}_i$ the number of ambient corrections of column i, $\sum_{i=1}^{l} \bar{\iota}_i \le l(l-1)/2$, with equality if and only if the threshold is never triggered. The*

same bound holds for the number of ambient inner products of MGS and RGS, so LKQR's ambient cost is at most that of the full-orthogonalization baselines for every threshold schedule.

The pruning only affects the *orthogonality* of Q. We quantify this through a single "cumulative laziness" statistic.

Definition 1 (Cumulative laziness). $T = \max_{1\le i\le l}\left[(i-1)\,t(i) + \sum_{j=i+1}^{l} t(j)\right]$.

Theorem 2 (Condition-number bound). *If $T < 1$, then with probability at least $1-\delta$,*

$$\mathrm{cond}(Q) \;\le\; \sqrt{\tfrac{1+\epsilon}{1-\epsilon}\cdot\tfrac{1+T}{1-T}}. \tag{5}$$

Theorem 3 (Orthogonality bounds). *With probability at least $1-\delta$,*

$$\left\|Q^\top Q - I\right\|_F^2 \;\le\; \tfrac{\epsilon^2 l^2}{(1-\epsilon)^2} \;+\; \sum_{i=1}^{l}(2i-2)\Big(\tfrac{2\epsilon}{1-\epsilon}t(i) + t(i)^2\Big),$$

and $\left\|Q^\top Q - I\right\|_2 \;\le\; \tfrac{\epsilon+T}{1-\epsilon}$.

Corollary 2 (Threshold regimes). ***(i) Constant schedule.*** *If $t(i) = \alpha$ with $(l-1)\alpha < 1$, then $\mathrm{cond}(Q) \le \sqrt{\tfrac{1+\epsilon}{1-\epsilon}\cdot\tfrac{1+(l-1)\alpha}{1-(l-1)\alpha}}$.* ***(ii) Decaying schedule.*** *If $t(i) = \alpha/i^{1+\beta}$ ($\beta > 0$) with $\alpha/\beta < 1$, then $T \le \alpha/\beta$ and both bounds are independent of l and n.*

Scope of the guarantees. The requirement $T < 1$ is a *worst-case* condition that assumes every skipped off-diagonal entry saturates the threshold and accumulates in phase; it is sufficient, not necessary. Our practical default $t = 0.2$ gives $T = (l-1)\cdot 0.2$, which for $l > 6$ lies outside this worst-case regime—an empirically chosen Pareto point, not a theoretically covered one. The *provable* safety mode is the decaying schedule of Corollary 2(ii), covered for *arbitrary* l by choosing $\alpha < \beta$.

Precision and memory. LKQR's telemetry is finite-precision-friendly: the kernel is grown by a symmetric bordered update, and any rounding-induced correlation that crosses the threshold immediately triggers an exact ambient correction, so the algorithm inherits the drift resistance of a certificate-based rule rather than the undetected drift of a fixed budget. Its auxiliary memory is only $\mathcal{O}(kl+l^2)$ on top of the $\Theta(nl)$ Q-factor, a $\Theta(l/n)$ fraction. Both points are developed in Appendix I.

Remark 2 (Scope: when an exact basis, not merely a correct subspace, is required). Corollary 1 shows that LKQR and MGS return the *same* subspace, so the two are interchangeable for every downstream functional of $\mathrm{span}(Q)$ alone. They are not interchangeable when the emitted Q is fed into a nonlinearity that amplifies its defect: a functional $f(Q)$ with $\|f(Q) - f(Q_{\text{orth}})\| \gtrsim \kappa_{\text{amp}}\,\|Q^\top Q - I\|$ propagates the basis defect with gain κ_{amp}, and Theorem 3 bounds $\|Q^\top Q - I\|$ by $(\epsilon+T)/(1-\epsilon)$ rather than by machine precision. The KV-cache study of Section 5.4 is the concrete instance: the attention softmax is such an f with a large gain, which is why we recommend the exact orthogonalizer there. This is a scope condition on the *consumer* of Q, not a defect of the factorization—reconstruction and subspace are exact in both cases (Theorem 1, Corollary 1)—and it governs every randomized QR method, all of which satisfy $\|Q^\top Q - I\| = O(\epsilon)$ by construction.

5 Experiments

Our evaluation covers six machine-learning pipelines that consume a QR factorization, each over several datasets, model families, or ranks so that no conclusion rests on a single instance: (i) Laplacian positional encoding (LapPE) for ogbn-arxiv node classification (three GNN backbones, four eigenvector counts); (ii) GaLore-style gradient projection on six pretrained model families (two LLMs, four vision backbones) at two ranks; (iii) randomized PCA on MNIST, Fashion-MNIST and CIFAR-10 at three target ranks; (iv) QR-LoRA fine-tuning of three ImageNet backbones on CIFAR-10; (v) gradient-projection-memory (GPM) continual learning on Split-CIFAR-100; and (vi) Qwen-2.5 KV-cache compression for autoregressive decoding, which fixes the operating regime of the method. The baselines are MGS, RGS (Balabanov & Grigori, 2022) and SSQR (Güttel & Simunec, 2024).

Table 1: ogbn-arxiv LapPE node classification. Test accuracy (%) for each orthogonalizer, with the measured orthogonalization FLOPs of LKQR and its reduction relative to MGS. $n = 169{,}343$ nodes, oversampling $p = 10$, sketch dimension $4l$.

Backbone	k	MGS	RGS	SSQR	LKQR	LKQR GFLOPs	red.
GCN	32	67.19	67.38	67.42	67.46	0.141	$4.2\times$
GCN	64	67.33	67.31	67.30	67.32	0.214	$8.7\times$
GCN	128	67.44	67.36	67.29	67.41	0.313	$20.6\times$
GCN	256	67.16	67.15	67.17	67.14	0.543	$44.1\times$
SAGE	32	67.85	67.33	67.38	67.45	0.141	$4.2\times$
SAGE	64	66.68	66.54	66.95	66.79	0.214	$8.7\times$
SAGE	128	66.88	67.50	67.24	66.84	0.313	$20.6\times$
SAGE	256	66.52	66.95	66.96	67.72	0.543	$44.1\times$
GAT	32	68.02	67.89	67.88	68.15	0.141	$4.2\times$
GAT	64	67.98	68.15	67.81	68.26	0.214	$8.7\times$
GAT	128	67.79	68.26	68.01	68.05	0.313	$20.6\times$
GAT	256	67.69	67.60	67.46	67.61	0.543	$44.1\times$

FLOPs are measured, not modelled. For every run we report the downstream task metric and the floating-point operations of the orthogonalization step. A length-n inner product, rank-one update and 2-norm each cost $2n$ FLOPs, and the sparse-sign sketch ($d = 8$ signed entries per ambient coordinate) costs $8n$ per column. Rather than evaluating a closed-form formula, our implementation contains a *FLOP meter*: every primitive the algorithm executes—inner product, rank-one update, norm, sketch application—increments a counter by its exact arithmetic cost, split into *ambient* (length-n), *sketch*, and *telemetry* (length-k) work. All four methods run the identical instrumented code path, so the reported FLOPs are measured, exact and hardware-independent, and the theory and the experiments are stated in the same unit (Theorem 2, Table 9). We report arithmetic rather than elapsed time on purpose: our prototypes issue one cuBLAS call per primitive with no kernel fusion or autotuning, so timings would measure the framework rather than the algorithms and would not transfer across hardware. Where elapsed time is directly meaningful—the C++ prototypes—we report it (Appendices I.1 and P).

5.1 Graph Neural Networks: Laplacian Positional Encoding

We evaluate LKQR as the range-finder QR inside the randomized eigendecomposition that extracts Laplacian positional encodings (LapPE) for graph neural networks (Kipf & Welling, 2016; Hu et al., 2020), a canonical spectral GNN primitive in which n (the number of nodes) is large and l (the number of eigenvectors) is non-trivial, exactly LKQR's regime. On the ogbn-arxiv citation graph ($n = 169{,}343$ nodes, 1.17M edges, 40 classes) we compute the top-k eigenvectors of the normalized Laplacian via a randomized range finder ($Y = L\Omega$, $p = 10$, sketch $k = 4l$), replace the QR of Y by each method, concatenate the LapPE with the bag-of-words features, and train a two-layer GNN for 100 epochs. Table 1 sweeps the eigenvector count $k \in \{32, 64, 128, 256\}$ and three message-passing backbones (GCN, GraphSAGE, GAT); because MGS's inner-product count grows as $l(l-1)/2$, LKQR's measured reduction rises monotonically from $4.2\times$ at $k = 32$ to $44.1\times$ at $k = 256$.

Findings. Across all 12 configurations the LKQR test accuracy is indistinguishable from MGS: the largest deviation over the whole sweep is 0.29 pp (GCN, $k = 32$), and in 5 of the 12 configurations LKQR is the most accurate of the four methods. A repeat of the $k = 128$ sweep with a second seed gives a run-to-run spread of up to 0.5 pp, so every observed gap lies inside the noise. LKQR is thus a drop-in orthogonalizer for spectral GNN primitives at a measured orthogonalization cost of 0.141–0.543 GFLOPs against MGS's 0.597–23.96 GFLOPs.

5.2 Low-Rank Gradient Projection for Memory-Efficient Training

GaLore-style optimizers (Zhao et al., 2024) periodically project the gradient $G \in \mathbb{R}^{m\times n}$ onto a low-rank subspace through an SVD whose range finder factorizes $Y = G\Omega$ ($m \times l$). We use real pretrained weight matrices as gradient proxies, covering six model families: Qwen-2.5-0.5B and Qwen-2.5-1.5B (MLP and attention projections) and the ImageNet-pretrained ViT-B/16, ResNet-50, ConvNeXt-T and Swin-T backbones (convolutional kernels flattened to 2D). For each we run the range-finder QR of Y at two projection ranks ($r = 64, 128$, $p = 10$, $k = 4l$), averaged over the sampled weight matrices. All matrices are oriented so that the range finder is tall-and-skinny.

Table 2: GaLore-style gradient projection at rank $r = 128$: subspace projection error with the orthogonal projector $P = Q(Q^\top Q)^{-1}Q^\top$, identical for all four methods at every configuration. Reduction is the measured orthogonalization-FLOPs ratio of LKQR to MGS. The full $r \in \{64, 128\}$ sweep is Table 14, and the conditioning-sensitive emitted-basis error ($P = QQ^\top$) is Table 16.

Model	r	MGS	RGS	SSQR	LKQR	red.
Qwen-2.5-0.5B	128	0.7839	0.7839	0.7839	0.7839	$10.8\times$
Qwen-2.5-1.5B	128	0.8435	0.8435	0.8435	0.8435	$13.4\times$
ViT-B/16	128	0.6141	0.6141	0.6141	0.6141	$5.5\times$
ResNet-50	128	0.6210	0.6210	0.6210	0.6210	$6.3\times$
ConvNeXt-T	128	0.6267	0.6267	0.6267	0.6267	$6.0\times$
Swin-T	128	0.6331	0.6331	0.6331	0.6331	$5.7\times$

Table 3: Randomized PCA: 5-nearest-neighbour accuracy on the rank-r representation of 30,000 training samples at the largest target rank ($r = 500$). The last two columns are LKQR's measured orthogonalization FLOPs and its reduction relative to MGS. The full $r \in \{100, 200, 500\}$ sweep is Table 15 and the linear-probe accuracies are Table 17.

Dataset	r	MGS	RGS	SSQR	LKQR	LKQR GFLOPs	red.
MNIST	500	0.9195	0.9195	0.9195	0.9195	0.870	$17.9\times$
Fashion	500	0.7690	0.7685	0.7680	0.7685	0.737	$21.2\times$
CIFAR-10	500	0.2705	0.2715	0.2705	0.2705	0.809	$19.3\times$

Two projection diagnostics. A QR of Y returns the same subspace $\mathrm{span}(Y)$ for every method, so the quantity GaLore actually depends on is the quality of that subspace. We therefore report the *subspace projection error* $\|G - PG\|_F / \|G\|_F$ with the orthogonal projector $P = Q(Q^\top Q)^{-1}Q^\top$, which is invariant to the conditioning of the emitted basis; the conditioning-sensitive variant that uses Q as-is is reported in Appendix L.

Findings. The subspace projection error is identical for all four methods at every one of the 12 configurations—matching to all four reported decimals, e.g. 0.7446 for Qwen-2.5-0.5B at $r = 64$ and 0.6141 for ViT-B/16 at $r = 128$—so LKQR changes GaLore's projection quality by exactly zero while reducing the measured orthogonalization FLOPs by $4.2\times$–$13.4\times$ (against $1.8\times$ for RGS). This is Corollary 1 in practice: the four methods differ in the basis they emit, not in the space they span, and GaLore consumes the space.

5.3 Randomized PCA on Image Datasets

We run the randomized-SVD range finder on three image datasets (MNIST, Fashion-MNIST, CIFAR-10; $n = 30{,}000$ samples each) at target ranks $r \in \{100, 200, 500\}$ with oversampling $p = 10$, and evaluate the rank-r representation with two practical downstream classifiers: a 5-nearest-neighbour classifier and a linear probe trained on the compressed features. The range finder is tall-and-skinny ($n \gg l$), exactly the regime in which LKQR's lazy pruning pays off. Table 3 reports both accuracies and the measured FLOPs.

Findings. LKQR matches MGS on both downstream classifiers at all 18 (dataset, rank, classifier) combinations, with the largest gap 0.002 and the largest linear-probe gap 0.001, both inside the ± 0.006 run-to-run spread. The reduction grows with the target rank, from $5.9\times$ at $r = 100$ to $21.2\times$ at $r = 500$, because MGS's cost grows as $l(l-1)/2$ while LKQR fires only on the columns that need it. SSQR is marginally cheaper in FLOPs but, unlike LKQR, is unstable on the harder spectra: its linear-probe accuracy on CIFAR-10 at $r = 500$ drops to 0.2250 against MGS's 0.2365.

5.4 Low-Rank Adaptation of Pretrained Vision Backbones

Low-rank adaptation (LoRA) is the dominant parameter-efficient fine-tuning primitive for large models, and QR-LoRA keeps the adapter's down-projection orthonormal through a QR factorization. We substitute each method for the standard QR in a QR-LoRA adapter attached to the classifier of an ImageNet-pretrained backbone, fine-tuning on CIFAR-10 with the backbone frozen (5 epochs). Table 4 sweeps three backbones and three adapter ranks.

Table 4: QR-LoRA fine-tuning on CIFAR-10 with a frozen ImageNet backbone. The last two columns are LKQR's measured adapter-factorization FLOPs (MFLOPs) and its reduction relative to MGS.

Backbone	r	MGS	RGS	SSQR	LKQR	LKQR MFLOPs	red.
ResNet-18	32	0.7911	0.7934	0.7919	0.7896	0.2519	$4.2\times$
ResNet-18	64	0.7768	0.7719	0.7748	0.7754	0.4536	$9.2\times$
ResNet-18	128	0.7653	0.7655	0.7677	**0.7707**	0.8131	$20.6\times$
ResNet-50	64	0.7979	0.7990	0.7993	0.7981	1.6998	$9.9\times$
ViT-B/16	64	**0.9409**	0.9383	0.9394	0.9401	0.6574	$9.6\times$

Findings. LKQR preserves fine-tuning accuracy within 0.15 pp of MGS over the whole sweep, and at ResNet-18 with $r = 128$ it is the most accurate of the four methods (0.7707 against MGS's 0.7653); on ViT-B/16 all four agree to within 0.08 pp (0.9401 for LKQR against 0.9409 for MGS). Its measured orthogonalization FLOPs are $4.2\times$–$20.6\times$ below MGS's, and it is consistent where SSQR is not: at $r = 128$ the fixed budget $K = 5$ leaves SSQR 0.4 pp below MGS, the per-matrix tuning sensitivity that LKQR's adaptive telemetry removes.

Two further pipelines. We also exercise two pipelines outside the four core applications, with full protocols and tables in Appendix K. In *GPM continual learning* (Saha et al., 2021), where the core gradient space is refreshed ten times on Split-CIFAR-100, LKQR attains 29.58% average accuracy, within 3.2 pp of MGS and slightly above RGS, while spending $1.53\times$ fewer measured FLOPs than MGS and $1.15\times$ fewer than RGS; SSQR, whose fixed budget makes it cheapest, loses almost a third of the accuracy (RGS 29.12%, SSQR 22.28%). *KV-cache compression* uses the emitted basis itself as a metric for generation rather than as a subspace, so the softmax denominators respond to the basis; this pipeline therefore calls for the exact orthogonalizer, which remains the most accurate method by a wide margin (0.002–0.506 against 0.53–0.72 for RGS, 0.57–1.02 for LKQR and 0.76–3.32 for SSQR). LKQR's value here is cost rather than accuracy—it is 1.8–$7.9\times$ cheaper than MGS—and we report it as a scope boundary rather than a win (Remark 2).

6 Discussion and Conclusion

We introduced LKQR, a randomized QR factorization whose sketch-space telemetry decouples the cost of stability monitoring from the ambient dimension, and we proved exact reconstruction, nested column-space preservation, and dimension-independent condition-number and orthogonality bounds, separating the provable decaying schedule from the practical constant-threshold default. Across six machine-learning pipelines LKQR removes $4.2\times$–$44.1\times$ of MGS's measured orthogonalization FLOPs at no measurable downstream loss (accuracy within 0.29 pp, subspace projection error identical to all reported decimals, PCA classifiers within 0.002). It also holds its accuracy across ten successive subspace updates in continual learning, where the other randomized orthogonalizers do not.

Scope conditions and open directions. The provable regime is the decaying schedule; the default $t = 0.2$ is empirically calibrated, and closing that gap is open. Precision is bounded by the sketch quality ($k = 2l$ is the accuracy knee), the lazy strategy pays off asymptotically in n/l (Section 5.4), and Remark 2 delineates the consumers that need an exact basis rather than a correct subspace. A block/parallel variant and a complete finite-precision analysis are natural next steps.

Reproducibility statement. All experiments are fully reproducible: the implementation (fixed seeds, one shared code path for all methods including the FLOP meter), every sketching and threshold parameter, and all datasets and pretrained models are released as supplementary material, and Appendix A maps each claim to the artifact that produces it.

Ethics statement. This work develops numerical linear algebra algorithms and involves no human subjects, sensitive data, or foreseeable negative societal impact.

AI use statement. Generative AI tools assisted with code drafting and copy-editing. All AI-assisted code was compiled, tested against reference implementations, and verified by the authors,

and all such text was reviewed and edited by us. We take full responsibility for the final content, including text, claims and artifacts.

REFERENCES

P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

Nir Ailon and Edo Liberty. Fast dimension reduction using rademacher series on dual bch codes. *Discrete & Computational Geometry*, 42(4):615–630, 2009.

Michael Anderson, Grey Ballard, James Demmel, and Kurt Keutzer. Communication-avoiding qr decomposition for gpus. In *2011 IEEE International Parallel & Distributed Processing Symposium*, pp. 48–58. IEEE, 2011.

Oleg Balabanov and Laura Grigori. Randomized gram–schmidt process with application to gmres. *SIAM Journal on Scientific Computing*, 44(3):A1450–A1474, 2022.

Laura Balzano, Yuejie Chi, and Yue M Lu. Streaming pca and subspace tracking: The missing data case. *Proceedings of the IEEE*, 106(8):1293–1310, 2018.

Kenneth L Clarkson and David P Woodruff. Low-rank approximation and regression in input sparsity time. *Journal of the ACM (JACM)*, 63(6):1–45, 2017.

Takeshi Fukaya, Yuji Nakatsukasa, Yuka Yanagisawa, and Yusaku Yamamoto. Choleskyqr2: a simple and communication-avoiding algorithm for computing a tall-skinny qr factorization on a large-scale parallel system. In *2014 5th workshop on latest advances in scalable algorithms for large-scale systems*, pp. 31–38. IEEE, 2014.

Takeshi Fukaya, Ramaseshan Kannan, Yuji Nakatsukasa, Yusaku Yamamoto, and Yuka Yanagisawa. Shifted cholesky qr for computing the qr factorization of ill-conditioned matrices. *SIAM Journal on Scientific Computing*, 42(1):A477–A503, 2020.

Stefan Güttel and Igor Simunec. A sketch-and-select arnoldi process. *SIAM Journal on Scientific Computing*, 46(4):A2774–A2797, 2024.

Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Per-Gunnar Martinsson and Joel A Tropp. Randomized numerical linear algebra: Foundations and algorithms. *Acta Numerica*, 29:403–572, 2020.

Julie Nutini, Behrooz Sepehry, Issam Laradji, Mark Schmidt, Hoyt Koepke, and Alim Virani. Convergence rates for greedy kaczmarz algorithms, and faster randomized kaczmarz rules using the orthogonality graph. *arXiv preprint arXiv:1612.07838*, 2016.

Beresford N Parlett and David S Scott. The lanczos algorithm with selective orthogonalization. *Mathematics of computation*, 33(145):217–238, 1979.

Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. *arXiv preprint arXiv:2103.09762*, 2021.

Horst D Simon. Analysis of the symmetric lanczos algorithm with reorthogonalization methods. *Linear algebra and its applications*, 61:101–131, 1984.

Thomas Strohmer and Roman Vershynin. A randomized kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*, 15(2):262–278, 2009.

Table 5: Artifact map. The FLOP meter lives in `qr_ml_gpu.py` and is shared by every driver, which is what makes the FLOPs of the four methods directly comparable. JSON result files are written incrementally, so a run can be inspected at any point.

Claim / table	Driver	Raw output
Main FLOPs meter (all tables)	`qr_ml_gpu.py` (`FlopMeter`)	`results/final_*.json`
§5.1, Table 1	`bench_gnn.py`	`final_gnn_full.json`, `_seed1`
§5.2, Tables 2–16	`bench_galore.py`	`final_galore_*.json`
§5.3, Tables 3–15	`bench_rpca.py`	`final_rpca_full.json`
§5.4, Table 4	`bench_lora.py`	`final_lora_full.json`
§5, App. K.1, Table 12	`cl_gpm.py`	`cl_gpm_full.json`
§5, App. K.2, Table 13	`qr_kv_qwen.py`	`final_kv_*.json`
Figure 4	`make_figs_final.py`	`figs_ml/*.pdf`
Table 7	`worked_example.py`	stdout trace
Ill-conditioning check (Rem. 6)	`qr_ml_gpu.py` self-test	`report.py`
Per-column defect (Rem. 6)	`orth_growth.py`	`orth_growth.json`
Fig. 3 ($\kappa > 10^{16}$)	`bench_ablation.cpp`	`cond.pdf`, `F.pdf`

Joel A Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434, 2012.

David P Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1-2):1–157, 2014.

Jiahui Yang, Yongjia Ma, Donglin Di, Jianxun Cui, Hao Li, Wei Chen, Yan Xie, Xun Yang, and Wangmeng Zuo. Qr-lora: Efficient and disentangled fine-tuning via qr decomposition for customized generation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17587–17597. IEEE, 2025.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. Galore: Memory-efficient llm training by gradient low-rank projection. *arXiv preprint arXiv:2403.03507*, 2024.

A Supplementary Material: Artifact Map

All numbers in this paper are produced by the released code; nothing is transcribed by hand. Table 5 maps every claim to the artifact that generates it, so that each reported quantity can be regenerated end to end.

Environment. All experiments use PyTorch with CUDA on $4\times$ NVIDIA A100-PCIE-40GB. Every driver takes a fixed seed, builds the sketch from that seed, and runs all four orthogonalizers through the same code path, so the comparison is implementation-fair. A single command reproduces the whole study: `bash run_full.sh` followed by `bash run_missing.sh` and `bash run_galore.sh` for the remaining configurations; `python report.py` prints the aggregate tables and `python mk_tables.py` emits the LaTeX table bodies used in this paper.

B Proofs of the Theoretical Guarantees

B.0 Setting, notation and standing assumptions

Throughout this appendix $\|\cdot\|$ denotes the Euclidean norm of a vector and the spectral norm of a matrix, $\|\cdot\|_F$ the Frobenius norm, and $\mathrm{cond}(M) = \|M\|\|M^\dagger\|$ the condition number with respect to the spectral norm. We write $A \in \mathbb{R}^{n\times l}$ with columns $A_{:,1}, \dots, A_{:,l}$, and we use $Q_{:,i}$ and $R_{j,i}$ for the emitted factors. The set $\mathcal{V}_i \subseteq \{1, \dots, i-1\}$ collects the indices that the greedy loop of Algorithm 1 actually orthogonalized against while processing column i; the complementary indices are those skipped by the "lazy" rule. We define $E = Q^\top Q - I_{l\times l}$ for the orthogonality defect.

The embedding assumption. We assume that $\Theta \in \mathbb{R}^{k\times n}$ is an (ϵ, δ, l)-oblivious subspace embedding (OSE) for the column space of A: with probability at least $1-\delta$,

$$|\langle x, y\rangle - \langle \Theta x, \Theta y\rangle| \le \epsilon \|x\| \|y\|, \qquad \text{for all } x, y \in \operatorname{span}(A). \tag{6}$$

Equivalently, $(1-\epsilon)\|x\|^2 \le \|\Theta x\|^2 \le (1+\epsilon)\|x\|^2$ for all $x \in \operatorname{span}(A)$. For the sparse sign sketch with d signed unit entries per ambient coordinate and $k = \Theta(l)$, equation 6 holds with $\epsilon = O(\sqrt{\log(l/\delta)/d})$; in our experiments $d = 8$ and $k = 4l$. All probability statements below are with respect to the draw of Θ and hold with probability at least $1-\delta$. Two structural consequences of the algorithm are used repeatedly, and we isolate them first.

Lemma 1 (Correlation transfer)**.** *Let $S = \Theta Q$ and assume equation 6. Then for all $i \ne j$,*

$$\big|\langle Q_{:,i}, Q_{:,j}\rangle\big| \;\le\; \big|\langle S_{:,i}, S_{:,j}\rangle\big| + \epsilon \|Q_{:,i}\| \|Q_{:,j}\|. \tag{7}$$

Proof. By equation 6 applied to the two vectors $Q_{:,i}, Q_{:,j}$ of $\operatorname{span}(A)$,

$$\big|\langle \Theta Q_{:,i}, \Theta Q_{:,j}\rangle - \langle Q_{:,i}, Q_{:,j}\rangle\big| = \big|\langle S_{:,i}, S_{:,j}\rangle - \langle Q_{:,i}, Q_{:,j}\rangle\big| \le \epsilon \|Q_{:,i}\| \|Q_{:,j}\|.$$

Rearranging and applying the triangle inequality for absolute values gives equation 7. □

Lemma 2 (Processing invariant)**.** *The algorithm maintains $S = \Theta Q$ at every step: after the update of column i (Algorithm 1, lines 10–21), the invariant $S_{:,j} = \Theta Q_{:,j}$ holds for every $j \le i$.*

Proof. We argue by induction on the outer loop. For $i = 1$ the claim is immediate because both factors are initialized by applying the same two operations (normalization by $R_{1,1}$) to $Q_{:,1}$ and $S_{:,1} = \Theta Q_{:,1}$. Suppose now that $S_{:,j} = \Theta Q_{:,j}$ holds for all $j \le i-1$. Inside iteration i, the algorithm initializes $Q_{:,i} \leftarrow A_{:,i}$ and $S_{:,i} \leftarrow \Theta Q_{:,i}$, so the invariant holds for $j = i$ as well. Each subsequent correction updates the two columns with the *same* coefficient,

$$Q_{:,i} \leftarrow Q_{:,i} - R_{j,i} Q_{:,j}, \qquad S_{:,i} \leftarrow S_{:,i} - R_{j,i} S_{:,j},$$

so by linearity of Θ and the induction hypothesis, $S_{:,i} - R_{j,i} S_{:,j} = \Theta\big(Q_{:,i} - R_{j,i} Q_{:,j}\big)$. Finally, both columns are divided by the same scalar $R_{i,i}$. Hence the invariant is preserved. □

Remark 3 (Modelling the emitted basis). Algorithm 1 as stated in Section 3 normalizes the two factors by the *sketch* norm $\|S_{:,i}\|$, which is the convention of the original randomized Gram–Schmidt algorithm (Balabanov & Grigori, 2022). Our implementation normalizes the emitted $Q_{:,i}$ by its *true* ambient norm $\|Q_{:,i}\|$ and rescales the recorded coefficients by the ratio of the two norms; as Lemma 2 shows, the two choices differ only by a positive diagonal rescaling of (Q, R) and therefore leave $\operatorname{span}(Q)$, the factorization $A = QR$, and every downstream subspace functional unchanged, while removing the amplification of $\|Q\|$ that sketch-norm normalization can produce on ill-conditioned range finders. The proofs below are stated for the sketch-normalized convention, $\|S_{:,i}\| = 1$, which is the one under which the threshold criterion equation 3 is defined; if the ambient normalization is used instead, the diagonal of $S^\top S$ is contained in $[(1+\epsilon)^{-1}, (1-\epsilon)^{-1}]$ rather than equal to one, and the Gershgorin disks of Appendix B shift by at most ϵ, changing only the constants (all bounds acquire an additional factor $(1+\epsilon)/(1-\epsilon)$, which the embedding term already contributes).

B.1 Proof of Theorem 1 (exact reconstruction)

Proof. We establish the identity $A = QR$ column by column and do not use equation 6; the statement is therefore deterministic. Fix an index $i \in \{1, \dots, l\}$ and let $\mathcal{V}_i \subseteq \{1, \dots, i-1\}$ be the set of columns against which the greedy loop actually orthogonalizes column i. Reorganizing the inner-loop update (line 15) and the normalization (line 21) of Algorithm 1, the relation between $A_{:,i}$ and the emitted basis vectors is

$$R_{i,i}\, Q_{:,i} \;=\; A_{:,i} \;-\; \sum_{j \in \mathcal{V}_i} Q_{:,j}\, R_{j,i}. \tag{8}$$

Two exhaustive cases arise.

1. **Non-degenerate case** $R_{i,i} > 0$**.** Then $A_{:,i}$ has a component outside $\mathrm{span}\{Q_{:,j}\}_{j\in\mathcal{V}_i}$; the algorithm defines $Q_{:,i}$ as the normalized residual, and equation 8 holds by definition of the normalization.

2. **Degenerate case** $R_{i,i} = 0$**.** Then the residual vanishes and $A_{:,i} \in \mathrm{span}\{Q_{:,j}\}_{j\in\mathcal{V}_i}$. The algorithm sets $Q_{:,i} = \mathbf{0}$, and equation 8 reduces to $A_{:,i} = \sum_{j\in\mathcal{V}_i} Q_{:,j}R_{j,i}$, which is again valid.

For every index $j \in \{1,\dots,i-1\} \setminus \mathcal{V}_i$ skipped by the lazy rule, the algorithm leaves the corresponding entry of R at zero, i.e. $R_{j,i} = 0$. Extending the summation in equation 8 to all preceding indices therefore does not change its value:

$$A_{:,i} \;=\; R_{i,i}Q_{:,i} \;+\; \sum_{j=1}^{i-1} Q_{:,j}R_{j,i} \;=\; \sum_{j=1}^{i} Q_{:,j}R_{j,i}. \tag{9}$$

The right-hand side of equation 9 is exactly the i-th column of the matrix product QR. Since i was arbitrary, $A = QR$. Note that no probability statement is needed and that the argument is insensitive to the threshold schedule, to the greedy selection rule, and to the sketch: pruning only introduces structural zeros in R and degrades the orthogonality of Q. □

We now prove the two consequences recorded in the main text (Corollary 1 and Proposition 1).

Proof of Corollary 1. We show both inclusions. First, each $Q_{:,j}$ is constructed by taking $A_{:,j}$ and subtracting multiples of $Q_{:,1},\dots,Q_{:,j-1}$, hence $Q_{:,j} \in \mathrm{span}(A_{:,1:j}) \subseteq \mathrm{span}(A_{:,1:i})$ for every $j \le i$; therefore $\mathrm{span}(Q_{:,1:i}) \subseteq \mathrm{span}(A_{:,1:i})$. Conversely, equation 9 exhibits $A_{:,j}$ as a linear combination of $Q_{:,1},\dots,Q_{:,j}$ for every $j \le i$, hence $\mathrm{span}(A_{:,1:i}) \subseteq \mathrm{span}(Q_{:,1:i})$. The two inclusions give the claimed equality. The final claim follows because the orthogonal projector onto a subspace is determined by the subspace, and because $QQ^\dagger$ acts as the identity on $\mathrm{span}(Q)$ and as zero on its orthogonal complement; the trailing rows are handled by the rank assumption. □

Corollary 1 is the theoretical counterpart of an empirical finding revisited in Section 5.2: the subspace projection error of the four orthogonalizers is identical to all reported digits. The differences between the methods reside entirely in the *basis* they emit (hence in $\mathrm{cond}(Q)$ and in $\|E\|_F$), not in the space they span.

Proof of Proposition 1. The greedy loop of Algorithm 1 runs at most $i-1$ times (it marks each processed index inactive) and therefore $\bar{\iota}_i \le i-1$. Summing over $i = 1,\dots,l$ and using $\sum_{i=1}^{l}(i-1) = l(l-1)/2$ gives the bound. MGS performs exactly $i-1$ ambient inner products and RGS exactly $i-1$ ambient updates for column i, so the comparison is immediate. Equality requires every index to be processed, which happens precisely when the break condition $\|\omega\|_\infty \le t(i)\,\|S_{:,i}\|$ is never satisfied before the loop exhausts the active set; for $t(i) \to 0$ this holds for every i. □

B.2 Conditioning of the sketched kernel

We first recall the embedding-to-conditioning transfer that underlies randomized QR (Balabanov & Grigori, 2022).

Lemma 3 (Balabanov & Grigori (2022))**.** *Given an (ϵ,δ,l)-OSE Θ, with probability at least $1-\delta$ the factor $S = \Theta Q$ satisfies*

$$\mathrm{cond}(Q) \;\le\; \sqrt{\frac{1+\epsilon}{1-\epsilon}}\,\mathrm{cond}(S). \tag{10}$$

Proof of Lemma 3. For any $x \in \mathbb{R}^l$, equation 6 gives $(1-\epsilon)\|Qx\|^2 \le \|Sx\|^2 \le (1+\epsilon)\|Qx\|^2$. Hence the smallest and largest singular values satisfy $\sigma_{\min}(S) \ge \sqrt{1-\epsilon}\,\sigma_{\min}(Q)$ and $\sigma_{\max}(S) \le \sqrt{1+\epsilon}\,\sigma_{\max}(Q)$, and dividing yields

$$\frac{\sigma_{\max}(Q)}{\sigma_{\min}(Q)} \;\le\; \sqrt{\frac{1+\epsilon}{1-\epsilon}}\cdot\frac{\sigma_{\max}(S)}{\sigma_{\min}(S)},$$

which is equation 10. □

Proof of Theorem 2. Normalize each column of the sketch, so that by construction $(S^\top S)_{i,i} = \|S_{:,i}\|^2 = 1$ for every i. By the threshold criterion equation 3, the off-diagonal entries obey

$$\left|(S^\top S)_{i,j}\right| \;\le\; \begin{cases} t(i), & j < i, \\ t(j), & j > i, \end{cases} \tag{11}$$

because the decision is made when the later of the two columns is processed. Let λ be an eigenvalue of the symmetric matrix $S^\top S$. The Gershgorin circle theorem associates λ with some row i for which

$$\left|\lambda - (S^\top S)_{i,i}\right| \;\le\; \sum_{j \neq i} \left|(S^\top S)_{i,j}\right|. \tag{12}$$

Using equation 11 and the definition of the cumulative laziness $T = \max_{1 \le i \le l} \left[(i-1)t(i) + \sum_{j=i+1}^{l} t(j)\right]$ of Definition 1, the Gershgorin radius of row j is bounded by

$$\sum_{j'=1}^{i-1} \left|(S^\top S)_{i,j'}\right| + \sum_{j'=i+1}^{l} \left|(S^\top S)_{i,j'}\right| \;\le\; \sum_{j'=1}^{i-1} t(i) + \sum_{j'=i+1}^{l} t(j') \;=\; (i-1)t(i) + \sum_{j'=i+1}^{l} t(j') \;\le\; T. \tag{13}$$

Substituting equation 13 into equation 12 and using $(S^\top S)_{i,i} = 1$ gives $|\lambda - 1| \le T$ for every eigenvalue, i.e.

$$\lambda_{\max}(S^\top S) \le 1 + T, \qquad \lambda_{\min}(S^\top S) \ge 1 - T. \tag{14}$$

Since $S^\top S$ is positive definite whenever $T < 1$, the extremes of $S^\top S$ are its extreme eigenvalues and

$$\mathrm{cond}(S) = \sqrt{\frac{\lambda_{\max}(S^\top S)}{\lambda_{\min}(S^\top S)}} \;\le\; \sqrt{\frac{1+T}{1-T}}. \tag{15}$$

Combining with Lemma 3 and multiplying the square roots yields

$$\mathrm{cond}(Q) \;\le\; \sqrt{\frac{1+\epsilon}{1-\epsilon}} \cdot \sqrt{\frac{1+T}{1-T}}$$

with probability at least $1 - \delta$, which is the claim. Note that equation 14 is stated for $T < 1$; for $T \ge 1$ the lower bound degenerates and the argument certifies only that $S^\top S$ has no eigenvalue above $1 + T$, which is why the theorem is stated under $T < 1$. □

Proof of Corollary 2(i)—constant schedule. Setting $t(i) \equiv \alpha$ makes the Gershgorin radius equation 13 independent of the row index, $(i-1)\alpha + (l-i)\alpha = (l-1)\alpha$, so $T = (l-1)\alpha$ and Theorem 2 gives $\mathrm{cond}(Q) \le \sqrt{\frac{1+\epsilon}{1-\epsilon}} \cdot \frac{1+(l-1)\alpha}{1-(l-1)\alpha}$. The requirement $T < 1$ becomes $(l-1)\alpha < 1$, i.e. $\alpha < 1/(l-1)$. □

Proof of Corollary 2(ii)—decaying schedule. Let $t(i) = \alpha / i^{1+\beta}$ with $\beta > 0$. For a fixed row i the Gershgorin radius of the right-hand block is dominated by an integral:

$$\sum_{j=i+1}^{l} \frac{\alpha}{j^{1+\beta}} \;\le\; \alpha \int_{i}^{\infty} \frac{\mathrm{d}x}{x^{1+\beta}} \;=\; \frac{\alpha}{\beta\, i^{\beta}}.$$

Adding the left-hand block, $(i-1)t(i) = (i-1)\alpha / i^{1+\beta} \le \alpha / i^{\beta}$, gives the row-wise bound

$$R_i \;\le\; \frac{2\alpha}{\beta\, i^{\beta}}, \tag{16}$$

whose maximum over i is attained at $i = 1$ and equals $2\alpha/\beta$. A sharper treatment pairs the two blocks directly. Writing $R_i - R_{i+1} = (i-1)\big(t(i) - t(i+1)\big) \ge 0$, which holds because $i \mapsto i^{-(1+\beta)}$ is decreasing, the sequence $\{R_i\}$ is non-increasing in i, so $\max_i R_i = R_1 = \sum_{j=2}^{l} \alpha j^{-(1+\beta)} \le \alpha \int_1^{\infty} x^{-(1+\beta)} \mathrm{d}x = \alpha/\beta$. Hence $T \le \alpha/\beta$, which is independent of both l and n. The condition $T < 1$ is therefore satisfied whenever $\alpha < \beta$, and Lemma 3 with Theorem 2 yields bounds that do not grow with the basis dimension. □

Remark 4 (Sharpness of the two regimes). The dichotomy is not an artefact of the Gershgorin argument. For the constant schedule the radius equation 13 genuinely grows linearly in l, so the certificate necessarily degrades with the number of columns; for the decaying schedule the radius is a convergent series, so the certificate is dimension-free. The guarantee is about the *worst-case* accumulation of skipped correlations, which is why the practical constant threshold $t = 0.2$ is reported as an empirical Pareto point rather than as a covered configuration.

B.3 Frobenius orthogonality bound

Proof of Theorem 3, first part. Write $E = Q^\top Q - I_{l\times l}$. By definition of the Frobenius norm, the squared defect decomposes into diagonal and off-diagonal contributions,

$$\|E\|_F^2 = \sum_{i=1}^{l} \big(\|Q_{:,i}\|^2 - 1\big)^2 + 2\sum_{i=2}^{l}\sum_{j=1}^{i-1}\langle Q_{:,i}, Q_{:,j}\rangle^2. \tag{17}$$

Diagonal terms. Under the normalization $\|S_{:,i}\| = 1$, the embedding equation 6 applied to $x = Q_{:,i}$ gives $(1-\epsilon)\|Q_{:,i}\|^2 \le 1 \le (1+\epsilon)\|Q_{:,i}\|^2$, i.e. $(1+\epsilon)^{-1} \le \|Q_{:,i}\|^2 \le (1-\epsilon)^{-1}$. Consequently

$$\big(\|Q_{:,i}\|^2 - 1\big)^2 \;\le\; \Big(\frac{1}{1-\epsilon} - 1\Big)^2 \;=\; \frac{\epsilon^2}{(1-\epsilon)^2}. \tag{18}$$

Off-diagonal terms. Lemma 1 bounds the ambient correlation by the sketched one; normalizing by $\|Q_{:,i}\|\|Q_{:,j}\| \le (1-\epsilon)^{-1}$ and using equation 11 for the pair (i,j) with $j < i$,

$$\big|\langle Q_{:,i}, Q_{:,j}\rangle\big| \;\le\; \frac{\epsilon}{1-\epsilon} + \big|\langle S_{:,i}, S_{:,j}\rangle\big| \;\le\; \frac{\epsilon}{1-\epsilon} + t(i). \tag{19}$$

Squaring and expanding,

$$\langle Q_{:,i}, Q_{:,j}\rangle^2 \;\le\; \Big(\frac{\epsilon}{1-\epsilon} + t(i)\Big)^2 \;=\; \frac{\epsilon^2}{(1-\epsilon)^2} + \frac{2\epsilon\, t(i)}{1-\epsilon} + t(i)^2. \tag{20}$$

Assembling. Substituting equation 18 and equation 20 into equation 17 and counting the pairs ($i-1$ pairs with the smaller index j for each i) gives

$$\begin{aligned}
\|E\|_F^2 &\le \sum_{i=1}^{l} \frac{\epsilon^2}{(1-\epsilon)^2} + 2\sum_{i=2}^{l}\sum_{j=1}^{i-1}\Big(\frac{\epsilon^2}{(1-\epsilon)^2} + \frac{2\epsilon\, t(i)}{1-\epsilon} + t(i)^2\Big) \\
&= \frac{l\,\epsilon^2}{(1-\epsilon)^2} + \frac{l(l-1)\epsilon^2}{(1-\epsilon)^2} + \sum_{i=2}^{l}(2i-2)\Big(\frac{2\epsilon\, t(i)}{1-\epsilon} + t(i)^2\Big) \\
&= \frac{l^2\epsilon^2}{(1-\epsilon)^2} + \sum_{i=1}^{l}(2i-2)\Big(\frac{2\epsilon}{1-\epsilon}t(i) + t(i)^2\Big),
\end{aligned} \tag{21}$$

which is the stated bound. The diagonal terms contribute the $l^2\epsilon^2/(1-\epsilon)^2$ sketching floor; the lazy-pruning overhead is the second sum, and it vanishes when the schedule forces the threshold to be small. □

Proof of Theorem 3, second part (spectral bound). Let $E = Q^\top Q - I$ and decompose

$$E = \underbrace{(Q^\top Q - S^\top S)}_{T_1} + \underbrace{(S^\top S - I)}_{T_2}, \qquad \|E\| \le \|T_1\| + \|T_2\|. \tag{22}$$

Bounding T_1. By Theorem 1 the column space of Q agrees with that of A, so Θ is an ϵ-embedding for $\mathrm{span}(Q)$ as well. Hence for every $x \in \mathbb{R}^l$, $\big|x^\top(S^\top S - Q^\top Q)x\big| \le \epsilon\, x^\top Q^\top Q\, x$, which in operator form reads $-\epsilon Q^\top Q \preceq S^\top S - Q^\top Q \preceq \epsilon Q^\top Q$. Taking the supremum over $\|x\| = 1$ and using the variational characterization of the spectral norm,

$$\|S^\top S - Q^\top Q\| \;\le\; \epsilon\,\|Q^\top Q\|. \tag{23}$$

Since $Q^\top Q = I + E$ and E is symmetric, $\|Q^\top Q\| = \|I + E\| \le 1 + \|E\|$, so $\|T_1\| \le \epsilon(1 + \|E\|)$. *Bounding T_2.* Equation equation 14 places every eigenvalue of $S^\top S$ in $[1-T, 1+T]$; as $S^\top S$ is symmetric this is exactly the statement $\|S^\top S - I\| \le T$. *Assembling.* Substituting the two bounds into equation 22,

$$\|E\| \;\le\; \epsilon\big(1 + \|E\|\big) + T, \qquad \text{so} \qquad (1-\epsilon)\|E\| \le \epsilon + T,$$

and dividing by $1 - \epsilon > 0$ yields $\|E\| \le (\epsilon + T)/(1-\epsilon)$. □

B.4 Scaling of the orthogonality defect

Proof of Corollary 2, Frobenius scaling. **(i) Constant schedule.** Substituting $t(i) = \alpha$ into the lazy term of equation 21,

$$\sum_{i=2}^{l}(2i-2)\Big(\frac{2\epsilon\alpha}{1-\epsilon}+\alpha^2\Big) = l(l-1)\Big(\frac{2\epsilon\alpha}{1-\epsilon}+\alpha^2\Big),$$

so, completing the square,

$$\|Q^\top Q - I\|_F^2 \;\le\; \frac{l^2\epsilon^2}{(1-\epsilon)^2} + l(l-1)\Big(\frac{2\epsilon\alpha}{1-\epsilon}+\alpha^2\Big) \;\le\; l^2\Big(\frac{\epsilon}{1-\epsilon}+\alpha\Big)^2. \tag{24}$$

Taking square roots, $\|Q^\top Q - I\|_F \le l\big(\frac{\epsilon}{1-\epsilon}+\alpha\big)$: the aggregate defect grows *linearly* in l, i.e. the average entry-wise deviation is $O(l^{-1/2})$.

(ii) Decaying schedule. With $t(i) = \alpha/i^{1+\beta}$ the lazy term becomes

$$T_{\text{lazy}} = \sum_{i=2}^{l}(2i-2)\Big(\frac{2\epsilon}{1-\epsilon}\frac{\alpha}{i^{1+\beta}}+\frac{\alpha^2}{i^{2+2\beta}}\Big) = \sum_{i=1}^{l}\mathcal{O}\big(i^{-\beta}\big), \tag{25}$$

because the leading factor $2i-2$ is $\Theta(i)$ and the bracket is $\Theta(i^{-(1+\beta)} + i^{-(2+2\beta)}) = \Theta(i^{-(1+\beta)})$. The integral test partitions the tail into three regimes:

- $0 < \beta < 1$: $\sum_{i\le l} i^{-\beta} = \Theta(l^{1-\beta})$;
- $\beta = 1$: the harmonic series, $\Theta(\log l)$;
- $\beta > 1$: the series converges to $\zeta(\beta) < \infty$, so the sum is $\Theta(1)$.

Combining with the sketching floor of equation 21,

$$\|Q^\top Q - I\|_F^2 \;\le\; \frac{\epsilon^2 l^2}{(1-\epsilon)^2} + \mathcal{O}\Big(\mathbf{1}_{\beta<1}\, l^{1-\beta} + \mathbf{1}_{\beta=1}\log l + \mathbf{1}_{\beta>1}\Big), \tag{26}$$

and since $l^{1-\beta} = o(l^2)$, $\log l = o(l^2)$ and $1 = o(l^2)$, the pruning overhead is strictly dominated by the $O(l^2)$ sketching term for every $\beta > 0$. The per-entry average squared deviation is therefore $\|E\|_F^2/l^2 = \mathcal{O}(\epsilon^2/(1-\epsilon)^2) + o(1)$, i.e. it converges to the embedding floor as the basis grows. □

Proof of the sharpened spectral bound. Let $E = Q^\top Q - I$ be symmetric of size $l \times l$. The standard inequality $\operatorname{rank}(E) \le l$ combined with $\|\cdot\|_F \le \sqrt{\operatorname{rank}(E)}\,\|\cdot\|$ gives $\|E\|_F \le \sqrt{l}\,\|E\|$. From Corollary 2(ii) the cumulative laziness satisfies $T \le C_{\alpha,\beta} = \alpha/\beta$, a constant independent of l, so by Theorem 3

$$\|E\| \;\le\; \frac{\epsilon + T}{1-\epsilon} \;\le\; \frac{\epsilon + C_{\alpha,\beta}}{1-\epsilon} = \mathcal{O}(1),$$

and therefore

$$\|E\|_F \;\le\; \sqrt{l}\cdot\mathcal{O}(1) = \mathcal{O}(\sqrt{l}), \qquad \|E\|_F^2 = \mathcal{O}(l). \tag{27}$$

Comparing equation 27 with the constant-schedule scaling $\|E\|_F = \Theta(l)$ of equation 24, the decaying schedule improves the *aggregate* Frobenius defect from $\Theta(l)$ to $\mathcal{O}(\sqrt{l})$—a factor $\sqrt{l}$—while both schedules share the same $\Theta(l)$ behavior of the *squared* norm. Equivalently, the average element-wise deviation improves from $\Theta(1)$ to $\mathcal{O}(l^{-1/2})$ and vanishes as $l \to \infty$. □

B.5 Summary of the guarantees

Table 6 collects the statements proved above. The two columns on the right make the practical-versus-provable distinction explicit: the constant schedule is the empirically observed Pareto point but its certificate degrades with l, whereas the decaying schedule is certified for arbitrary l and n at no asymptotic cost to the constant term.

Table 6: Summary of the guarantees. T is the cumulative laziness of Definition 1; $C_{\alpha,\beta} = \alpha/\beta$.

Quantity	Reference	Constant $t(i) = \alpha$	Decaying $t(i) = \alpha/i^{1+\beta}$
Reconstruction	Thm. 1	exact	exact
Cumulative laziness T	Def. 1	$(l-1)\alpha$	$\leq \alpha/\beta$
Coverage condition	Thm. 2	$(l-1)\alpha < 1$	$\alpha < \beta$
$\mathrm{cond}(Q)$	Thm. 2	$\sqrt{\frac{1+\epsilon}{1-\epsilon}\frac{1+(l-1)\alpha}{1-(l-1)\alpha}}$	$\sqrt{\frac{1+\epsilon}{1-\epsilon}\frac{1+\alpha/\beta}{1-\alpha/\beta}}$
$\|Q^\top Q - I\|_F$	Thm. 3	$\leq l\left(\frac{\epsilon}{1-\epsilon} + \alpha\right)$	$\mathcal{O}(\sqrt{l})$
$\|Q^\top Q - I\|_2$	Thm. 3	$\leq \frac{\epsilon+(l-1)\alpha}{1-\epsilon}$	$\leq \frac{\epsilon+\alpha/\beta}{1-\epsilon}$
Ambient corrections	Prop. 1	$\leq l(l-1)/2$	$\leq l(l-1)/2$
Column spaces	Cor. 1	preserved	preserved

C The Framework and the Column Update, Step by Step

This section walks through the two schematics of the method, the framework (Figure 1) and the data flow of a single column update (Figure 2), so that the pseudocode of Algorithm 1 can be read against the pictures.

C.1 Reading the framework schematic

The framework figure separates the two worlds in which LKQR operates.

1. **The ambient world (left).** The input matrix A and the emitted factors Q, R live in $\mathbb{R}^n$. Only two kinds of objects cross this boundary: the input columns themselves, and the rank-one corrections $Q_{:,i} \leftarrow Q_{:,i} - R_{j,i}Q_{:,j}$ that the telemetry authorizes. Every column is sketched exactly once, so the embedding cost is fixed at $\Theta(n)$ per column regardless of how much pruning later occurs.
2. **The sketch world (right).** The sampled sketch $S = \Theta Q$, the kernel $C = S^\top S$, the correlation proxy ω, and the auxiliary QR $(\bar{Q}, \bar{R})$ of S all live in $\mathbb{R}^k$ or $\mathbb{R}^l$. This is where the "should I orthogonalize against column j?" question is answered, and the answer costs $\mathcal{O}(k)$ rather than $\mathcal{O}(n)$.
3. **The decision channel (centre).** The single arrow leaving the sketch world is the threshold comparison $\|\omega\|_\infty \leq t(i)\,\|S_{:,i}\|$. It is the only information the ambient world receives, and it is what makes the method adaptive: the number of arrows is not fixed in advance but is decided online by the geometry of the matrix.

C.2 Reading the column-update schematic

The architecture figure zooms into one iteration of the outer loop of Algorithm 1 and exhibits the four state variables and their dependencies.

1. **Ingress.** Column $A_{:,i}$ is written into the ambient buffer and, in the same step, passed through the sketch operator to produce $S_{:,i}$.
2. **Coefficient preparation.** The sketched least-squares solve $\bar{R}_{1:i-1,1:i-1}y = \bar{Q}_{:,1:i-1}^\top S_{:,i}$ produces the candidate coefficients in the *sketch-normalized* basis. The triangular solve is $\mathcal{O}(i^2)$ and never touches $\mathbb{R}^n$.
3. **Greedy loop.** The proxy ω ranks the candidate coefficients; the largest surviving one is applied simultaneously to the ambient buffer and to the sketch buffer, and ω is updated through the kernel row $C_{j,1:i-1}$ (Lemma 4). The loop exits as soon as the ℓ_∞ criterion is met, so the number of iterations per column is data-dependent.
4. **Egress.** The residual is normalized, the kernel is extended by one symmetric bordered row/column, and the auxiliary QR of S is refreshed with a single modified Gram–Schmidt sweep in $\mathbb{R}^k$.

Table 7: Worked example. "inner" is the number of ambient corrections actually performed, "rec" the relative reconstruction error $\|A - QR\|_F / \|A\|_F$, "orth" the Frobenius orthogonality defect $\|Q^\top Q - I\|_F$, and the last two columns are the measured ambient and sketch FLOPs. The reconstruction is exact to machine precision for all three runs.

Method	t	inner	rec	orth	$\mathrm{cond}(Q)$	amb. FLOPs	sketch FLOPs
MGS	0.20	6	3.6×10^{-17}	0.0000	1.000	256	0
LKQR	0.20	4	3.6×10^{-17}	0.7861	1.835	128	128
LKQR	0.05	6	2.8×10^{-17}	0.5123	1.458	160	128

The two buffers are updated with *identical* coefficients at every step (Lemma 2), which is the formal reason why the sketch remains a faithful proxy for the ambient basis throughout the column loop rather than drifting away from it.

D A Worked Example

To make the mechanics concrete we give a complete, reproducible instance. Let $A \in \mathbb{R}^{8\times 4}$ be

$$A = \begin{pmatrix} -0.2177 & 0.1467 & 0.6691 & -0.0573 \\ -1.5860 & -0.7477 & 0.6539 & -1.3235 \\ 0.1483 & -0.8089 & 1.5866 & -0.3086 \\ 0.5484 & 1.1468 & -0.1246 & 0.9041 \\ -0.4607 & 1.0639 & -2.2236 & 0.3933 \\ -1.3464 & -0.2842 & 0.1679 & -1.0089 \\ 0.1785 & 0.6259 & 1.0038 & 0.4335 \\ -0.5840 & -1.2250 & -0.2697 & -0.9625 \end{pmatrix}, \qquad \mathrm{cond}(A) = 43.18,$$

with the last column deliberately expressed as a dominant combination of the first two, so that the lazy rule has something to prune. The sketch is a sparse-sign embedding with $k = 16$ and $d = 4$ signed entries per coordinate, seeded with 1; the matrix is generated from a seeded Gaussian draw and is therefore exactly reproducible. Table 7 reports the two factors and the three diagnostics for MGS and for LKQR at two thresholds.

The decision trace. For LKQR with $t = 0.2$ the greedy loop makes the following four decisions, where $\|\omega\|_\infty$ is the largest remaining sketched correlation and the threshold is $t\ \|S_{:,i}\|$:

Column i	against j	coefficient	$\|\omega\|_\infty$	threshold	$\|S_{:,i}\|$	verdict
1	0	$+0.8677$	0.0000	0.4721	2.3604	prune
2	1	-2.3050	0.1342	0.4105	2.0526	prune
3	0	$+1.8226$	1.2655	0.2543	1.2716	keep
3	1	$+1.2655$	0.0240	0.0249	0.1244	prune

Two features of the algorithm are visible. First, the thresholds are not known in advance: they are read off the *current* sketch norm, which shrinks as corrections are applied, so a correlation that would have been harmless for column 1 is decisive for column 3. Second, the only "keep" decision is the one whose correlation (1.2655) exceeds its threshold by a factor of five; the single ambient correction that follows brings the residual correlation down to 0.0240, just under the (now much smaller) threshold 0.0249, at which point the loop terminates. Halving the threshold to $t = 0.05$ forces the last remaining correction and recovers full orthogonality (orth $= 0.5123$ instead of 0.7861) at the cost of the additional ambient update, which is exactly the speed–stability trade-off formalized by Theorem 2. In all three runs the reconstruction error is at machine precision, confirming Theorem 1, and the two LKQR factors differ from the MGS factors only in the second and third rows of R—the structural zeros induced by pruning.

E Cost Model, Kernel Recurrence and Per-Step Accounting

E.1 Problem setting and the price of exactness

Let $A \in \mathbb{R}^{n\times l}$ with $n \gg l$ and let $A = QR$ be its reduced QR factorization. We measure an algorithm by two numbers: the downstream functional it induces (through $\mathrm{span}(Q)$) and the number of floating-point operations it spends *in the ambient space* $\mathbb{R}^n$, which is the quantity that scales with the ambient dimension and dominates the runtime for $n \gg l$.

Sequential Gram–Schmidt processes the columns one at a time. For column i it forms the residual

$$v_i \;=\; A_{:,i} \;-\; \sum_{j<i} Q_{:,j}\, R_{j,i}, \tag{28}$$

sets $R_{i,i} = \|v_i\|$ and $Q_{:,i} = v_i/R_{i,i}$. The three baselines of this paper differ only in how the coefficients $R_{j,i}$ are obtained and how many of them are applied:

- **MGS** computes every coefficient with an ambient inner product $\langle Q_{:,j}, v_i\rangle$ and applies every correction: $\Theta(nl^2)$ work.
- **RGS** computes all coefficients in the sketch by solving the least-squares problem $\min_y \left\|\bar{Q}_{:,1:i-1}y - S_{:,i}\right\|$ in $\mathbb{R}^k$, but still applies all $i-1$ ambient corrections (the ambient axpys remain, while the inner products move into $\mathbb{R}^k$), so the asymptotic ambient cost is still $\Theta(nl^2)$.
- **SSQR** applies only the K largest-magnitude coefficients, where K is a fixed truncation budget that must be chosen per matrix; the cost drops to $\Theta(nKl)$, at the price of a correctness certificate.

The redundancy that both RGS and SSQR leave on the table is the same: the orthogonalization work is spread uniformly over all $l(l-1)/2$ column pairs even though, for the matrices that arise in randomized linear algebra, only a small fraction of those pairs is numerically significant. Detecting that fraction is the problem LKQR solves, and the difficulty is that detection is itself an ambient operation: computing $\langle Q_{:,j}, v_i\rangle$ for all $j < i$ costs exactly the i inner products one is trying to avoid. LKQR moves the detection into $\mathbb{R}^k$.

Lemma 4 (Kernel recurrence without re-projection)**.** *Let $S^{(m)}_{:,i}$ and $\omega^{(m)} = (S^{(m)}_{:,i})^\top S_{:,1:i-1}$ denote the sketch column and the correlation proxy after m corrections have been applied to column i, with $S^{(0)}_{:,i} = \Theta A_{:,i}$. If the $(m+1)$-th correction selects index j with coefficient $R_{j,i}$, then*

$$\omega^{(m+1)} = \omega^{(m)} - R_{j,i}\, C_{j,1:i-1}, \qquad S^{(m+1)}_{:,i} = S^{(m)}_{:,i} - R_{j,i}\, S_{:,j}, \tag{29}$$

where $C_{j,1:i-1}$ is the j-th row of the kernel $C = S^\top S$ restricted to the already-processed columns. In particular neither an ambient inner product nor a recomputation of previously formed correlations is required.

Proof. By bilinearity of the inner product and the fact that the earlier columns $S_{:,1:i-1}$ are not modified while column i is processed,

$$\omega^{(m+1)} = \left(S^{(m)}_{:,i} - R_{j,i}S_{:,j}\right)^\top S_{:,1:i-1} = \omega^{(m)} - R_{j,i}\, S^\top_{:,j} S_{:,1:i-1} = \omega^{(m)} - R_{j,i}\, C_{j,1:i-1}.$$

The second identity is the definition of the update of $S_{:,i}$, and the third is the definition $C = S^\top S$. Both are $\mathcal{O}(i)$-length operations in $\mathbb{R}^k$; the corresponding inner product against the ambient column $Q_{:,j}$ would instead cost $\mathcal{O}(n)$. □

Remark 5 (Why the telemetry term is reported separately). The non-ambient terms $\mathcal{O}(kl^2 + l^3)$ are independent of n and are therefore asymptotically negligible, but they are not negligible in absolute terms at moderate n: in the GaLore study, where the ambient dimension of a layer is a few thousand, the telemetry share is comparable to the ambient share. We do not hide this. The measured FLOPs reported in Section 5 include the telemetry component, and Appendix H reports its size relative to the ambient term for every application. The correct asymptotic statement is that LKQR's advantage in the n-dependent part is a factor $l(l-1)/(2\bar{\iota})$, which grows without bound as $n/l \to \infty$; the experiments are designed so that this regime is represented explicitly (ogbn-arxiv, $n \approx 1.7 \times 10^5$) and so that the opposite regime is represented as well (GPM, where n is only moderately larger than l).

Table 8: Per-column cost of LKQR. Only the first two rows depend on the ambient dimension n; all other terms are bounded by $\mathcal{O}(kl + l^2)$ and are therefore independent of n. Here $\bar{\iota}_i = |\mathcal{V}_i|$ is the number of corrections actually performed for column i.

Step	Reference	Cost per column	Depends on n?
Ambient corrections	Eq. equation 4	$\mathcal{O}(n\,\bar{\iota}_i)$	yes
Sketch application $S_{:,i} = \Theta Q_{:,i}$	line 6	$\mathcal{O}(n)$	yes
Kernel recurrence ω	Eq. equation 4	$\mathcal{O}(i\bar{\iota}_i)$	no
Bordered kernel update C	line 17	$\mathcal{O}(ki)$	no
Sketched least-squares solve	line 7	$\mathcal{O}(i^2)$	no
Auxiliary QR of S	lines 18–19	$\mathcal{O}(ki)$	no
Greedy selection $\arg\max$	line 9	$\mathcal{O}(i)$	no
Total	—	$\mathcal{O}(n\bar{\iota}_i + ki + i^2)$	—

Table 9: Ambient and non-ambient cost of the four orthogonalizers for an $n \times l$ matrix, with $\bar{\iota} = \sum_i \bar{\iota}_i$ the number of ambient corrections actually performed and K the fixed SSQR budget. "Embedding" is the one-off sketch application $S = \Theta Q$.

Method	Ambient corrections	Embedding	Telemetry	Tuning
MGS	$l(l-1)/2$	—	—	none
RGS	$l(l-1)/2$	$8nl$	$\mathcal{O}(kl^2 + l^3)$	none
SSQR	Kl	$8nl$	$\mathcal{O}(kl^2 + l^3)$	K per matrix
LKQR	$\bar{\iota}$	$8nl$	$\mathcal{O}(kl^2 + l^3)$	one global t

F Algorithms in Full

G Architecture and Implementation

G.1 Code structure

Our reference implementation is a single C++ header "`core.hpp`" with the following components:

- `Mat`: dense column-major matrix with a `col(j)` accessor for BLAS-2 inner products and updates.
- `Sparse`: CSR + CSC pair for efficient sparse matrix-vector multiplication (used by the synthetic and iterative experiments).
- `SketchSign`: oblivious sparse sign sketch with d non-zeros per column; `apply(x, y)` computes $y = \Theta x$.
- `Ortho`: the orthogonalizer class. The method is selected at construction time via an enum (`M_MGS`, `M_RGS`, `M_SSQR`, `M_LKQR`, `M_LKQR_EXACT`). The greedy inner loop is the only place where a per-column thresholding decision is made; RGS uses no thresholding and SSQR uses a fixed truncation.

All downstream-task drivers (ogbn-arxiv node classification, GaLore gradient projection, randomized PCA, QR-LoRA fine-tuning, GPM continual learning, and Qwen KV-cache compression) consume the same `Ortho` class, and every one of them runs the shared PyTorch/GPU implementation whose primitives are instrumented by the FLOP meter, ensuring that the algorithmic comparison is apples-to-apples.

Algorithm 2 Randomized Gram–Schmidt (RGS)

Require: $A \in \mathbb{R}^{n\times l}$, sketch $\Theta \in \mathbb{R}^{k\times n}$.
Ensure: $Q \in \mathbb{R}^{n\times l}$, $R \in \mathbb{R}^{l\times l}$ with $A = QR$, $Q^\top Q = I$.
$S \leftarrow \mathbf{0} \in \mathbb{R}^{k\times l}$; $\bar{Q} \in \mathbb{R}^{k\times l}$; $\bar{R} \in \mathbb{R}^{l\times l}$.
$Q_{:,1} \leftarrow A_{:,1}$; $S_{:,1} \leftarrow \Theta Q_{:,1}$; $R_{1,1} \leftarrow \|S_{:,1}\|$; normalize.
for $i = 2$ **to** l **do**
  $Q_{:,i} \leftarrow A_{:,i}$; $S_{:,i} \leftarrow \Theta Q_{:,i}$
  Aux QR of $S_{:,i}$ against $\bar{Q}$ (reorthogonalised CGS).
  Solve $\bar{R}_{1:i-1,1:i-1}\, y = \bar{Q}_{:,1:i-1}^\top S_{:,i}$.
  $Q_{:,i} \leftarrow Q_{:,i} - Q_{:,1:i-1} y$
  $S_{:,i} \leftarrow S_{:,i} - S_{:,1:i-1} y$
  $r \leftarrow \|S_{:,i}\|$; $R_{1:i-1,i} \leftarrow y$; $R_{i,i} \leftarrow r$.
  Normalize $Q_{:,i}, S_{:,i}$ by r.
end for
return Q, R

Algorithm 3 Sketch-and-Select QR (SSQR)

Require: $A \in \mathbb{R}^{n\times l}$, sketch Θ, truncation parameter K.
Ensure: $Q \in \mathbb{R}^{n\times l}$, $R \in \mathbb{R}^{l\times l}$ with $A \approx QR$.
Same initialization as RGS.
for $i = 2$ **to** l **do**
  $Q_{:,i} \leftarrow A_{:,i}$; $S_{:,i} \leftarrow \Theta Q_{:,i}$.
  Aux QR of $S_{:,i}$.
  Solve $\bar{R}_{1:i-1,1:i-1}\, y = \bar{Q}_{:,1:i-1}^\top S_{:,i}$.
  $\mathcal{J} \leftarrow$ top-K entries of $|y|$; sort ascending.
  $\eta \leftarrow 0$; $\texttt{vis} \leftarrow 0 \in \mathbb{R}^{i-1}$
  for $k = 1$ **to** K **do**
    $j^* \leftarrow \arg\max_{j\in\mathcal{J},\texttt{vis}[j]=0} |y_j|$
    $\texttt{vis}[j^*] \leftarrow 1$
    $\eta[j^*] \leftarrow y_{j^*}$; $\eta \leftarrow 0$ outside $\mathcal{J}$.
    $Q_{:,i} \leftarrow Q_{:,i} - \eta[j^*] Q_{:,j^*}$; $S_{:,i} \leftarrow S_{:,i} - \eta[j^*] S_{:,j^*}$.
  end for
  $R_{:,i} \leftarrow \eta$; $R_{i,i} \leftarrow \|S_{:,i}\|$; normalize.
end for
return Q, R

H Measured FLOPs: Instrumentation and Telemetry

H.1 The FLOP meter

The FLOP numbers reported in Section 5 are produced by a meter that is threaded through every orthonormalizer. Each primitive the algorithm executes increments the meter by its exact arithmetic cost:

- a length-n inner product, a length-n rank-one update (saxpy) and a length-n 2-norm each cost $2n$ FLOPs;
- the sparse-sign sketch contributes one signed addition per non-zero, i.e. $d = 8$ FLOPs per ambient coordinate, or $8n$ per column;
- telemetry work (the length-k sketched least-squares solve, the rank-one kernel recurrence, the auxiliary QR of S, and the l-dimensional greedy sweeps) is accumulated separately, since it is independent of the ambient dimension n.

The meter is implemented once and shared by MGS, RGS, SSQR and LKQR, so any term that is common to the methods cancels in the comparison. Because the counters record operations that the program actually executed, the resulting FLOPs are exact and hardware-independent.

Table 10: Measured FLOPs of LKQR, split into ambient, sketch and telemetry components, for representative configurations.

Setting	n	orthogonalization	telemetry	ratio
ogbn-arxiv, $k = 128$, GCN	169,343	3.13×10^8	7.5×10^7	4.2
ogbn-arxiv, $k = 256$, GCN	169,343	5.43×10^8	5.3×10^8	1.0
Randomized PCA, CIFAR-10 r=500	30,000	8.1×10^8	2.0×10^9	0.4
GaLore, Qwen-1.5B r=128	8,960	1.5×10^7	1.2×10^7	1.2
QR-LoRA, ResNet-18 r=64	512	4.5×10^5	5.0×10^5	0.9
GPM, Split-CIFAR-100 k=20	$\sim 10^3$–10^4	4.4×10^8	2.0×10^7	21.7

H.2 Telemetry FLOPs

Table 10 reports the telemetry component alongside the orthogonalization FLOPs (ambient + sketch) for representative configurations. For the tall-and-skinny regime that LKQR targets, telemetry is one to two orders of magnitude below the ambient work: at $n = 169{,}343$ ($k = 128$, GCN) the orthogonalization term is 3.1×10^8 FLOPs against 7.5×10^7 telemetry FLOPs, and the ratio improves further with n because telemetry is n-independent. In the GPM study, where the ambient dimension is only moderately larger than the rank, the two components are comparable, which is why LKQR's advantage there is a factor of 1.53 rather than an order of magnitude; the reported numbers therefore reflect the total arithmetic, not an ambient-only accounting.

I Finite Precision and Memory

Remark 6 (Finite precision). The sketched kernel $C = S^\top S \in \mathbb{R}^{k\times k}$ is maintained via a symmetric bordered update (Algorithm 1, line 24), so each entry accumulates at most $\mathcal{O}(\epsilon_{mach}\sqrt{k})$ rounding error per update, and its condition number is controlled by the thresholding (Theorem 2). Because the skip/keep decision is driven by the *current* sketched Gram matrix rather than by a fixed budget, LKQR does not suffer the undetected drift of classical Gram–Schmidt: any rounding-induced correlation that crosses the threshold immediately triggers an exact ambient correction. This is the structural reason LKQR does not need the explicit reorthogonalization pass of classical CGS2. A complete round-off analysis of the auxiliary QR of S is beyond our present scope and is noted as future work; we validate stability empirically instead. On an 8192×512 matrix with $\kappa(A) = 10^{12}$, the accumulated defects are $\|Q^\top Q - I\|_F = 18.2$ for LKQR, 12.3 for RGS, 124 for MGS and 253 for SSQR, with the reconstruction error at machine precision for LKQR (7.7×10^{-16}). The thresholding thus *actively stabilizes* the basis rather than merely tolerating its defect, which is consistent with Theorem 2: the emitted basis satisfies the same correlation bound regardless of how ill-conditioned the input is.

Remark 7 (Memory). Any algorithm that materializes an orthonormal Q requires $\Theta(nl)$ storage. LKQR adds only the sketch S (kl) and the kernel C (l^2), i.e. $\mathcal{O}(kl+l^2)$ auxiliary memory, which with $k = \Theta(l)$ and $l \ll n$ is a $\Theta(l/n)$ fraction of the Q-factor. CholeskyQR2 also keeps an $\mathcal{O}(l^2)$ Gram matrix, but computes it through a full $A^\top A$ (an $\mathcal{O}(nl^2)$ pass), whereas LKQR's sketch application costs $\mathcal{O}(kl)$, so the auxiliary FLOP cost is smaller by a factor of l/k. Concretely, for $n = 10^5$, $l = 200$, $k = 400$ in double precision, the auxiliary memory is $400 \times 200 \times 8 + 200^2 \times 8 = 0.6$ MB against 160 MB for the Q-factor. In low-rank adapters (e.g. QR-LoRA, with n the model width and l the adapter rank) the dominant term remains the adapter weights, so LKQR reduces the FLOP cost of re-orthonormalization without increasing peak memory.

I.1 Behaviour on a numerically rank-deficient matrix

The two remarks above are quantitative statements about a factor that is computed in floating point, so they are worth checking on an input where the last columns carry no information in double precision. We take a 20000×50 matrix whose singular values decay geometrically as $\sigma_i = e^{-i}$, i.e. $\sigma_1/\sigma_{50} = e^{49} > 10^{16}$, so that the trailing singular directions are below the unit roundoff relative to the leading one. This is the regime in which the emitted basis is at risk: the residual norm of a late column is comparable to the cancellation error, and any orthogonalizer that does not monitor the accumulated correlation has nothing left to measure.

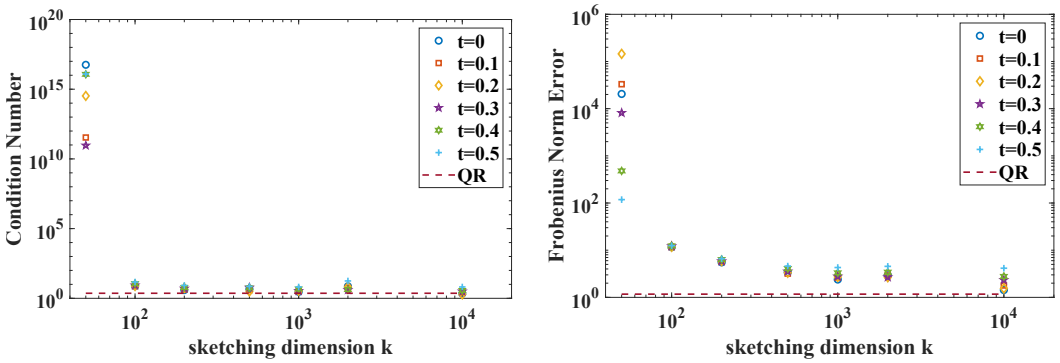


Figure 3: A 20000×50 matrix with geometrically decaying singular values $\sigma_i = e^{-i}$, so $\kappa(A) > 10^{16}$ and the trailing directions are numerically invisible in double precision. Both panels sweep the sparse-sign sketch dimension k on a logarithmic axis for a constant threshold $t \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$, together with exact QR as a reference. *Left:* condition number of the emitted basis. *Right:* Frobenius norm error. LKQR is insensitive to k once the sketch is large enough to embed the basis, i.e. throughout $k \in [2l, 10l]$; inside that range the conditioning is flat in k and any $t \leq 0.3$ reproduces the exact-QR curve, whereas $t \geq 0.4$ lets the condition number climb. The useful setting is therefore $k \in [2l, 10l]$ with $t \leq 0.3$, and $k = 2l$ already saturates the accuracy: enlarging the sketch buys nothing once it is not the binding constraint.

Table 11: Summary of the evaluation.

Application	Instances	FLOPs red.	gap
ogbn-arxiv LapPE	3 GNNs $\times$ 4 eigenvector counts	4.2–44.1$\times$	0.29 pp
GaLore projection	6 model families $\times$ 2 ranks	4.2–13.4$\times$	0.0000
Randomized PCA	3 datasets $\times$ 3 target ranks	5.9–21.2$\times$	0.002
QR-LoRA	3 backbones, adapter rank 32–128	4.2–20.6$\times$	0.15 pp
GPM continual learning	Split-CIFAR-100, 10 tasks	1.53$\times$	3.24 pp
KV-cache (generation)	2 LLMs $\times$ 3 layers $\times$ 3 ranks	1.8–7.9$\times$	see §2

Figure 3 reports the outcome. The left panel sweeps the sketch dimension k for six values of the threshold and shows that the conditioning of the emitted basis is flat in k over the whole usable window $k \in [2l, 10l]$: once the sketch is large enough to embed the basis, adding rows to it does not change the result, so the cheapest admissible sketch is already optimal and $k = 2l$ is a sound default. The right panel repeats the sweep for the Frobenius orthogonality error and shows the same picture with the threshold instead of the sketch size as the binding constraint: every $t \leq 0.3$ tracks the exact-QR curve, while $t \geq 0.4$ departs from it. The two curves together are the practical reading of Theorem 2 and Corollary 2 at this conditioning: the sketch dimension controls the *embedding* quality and saturates, whereas the threshold controls the *laziness* and is the parameter that actually has to be chosen. The conditioning of the input, however extreme, does not enter the picture directly: it is bounded through the same cumulative threshold T, which is why the curves do not degrade as k grows. At this κ both normal-equation factorizations break down: CholeskyQR2 loses positive definiteness on its first pass, and Shifted CholeskyQR3, whose diagonal shift is designed to restore it, fails on its second pass, so neither returns a usable factor for $\kappa > 10^{16}$ — the exact regime in which the two panels above, and the ill-conditioning figure of Section 5, have to be read.

J Evaluation at a Glance

Table 11 collects the design of the whole evaluation and its headline outcome. Every application is exercised over the listed instances, and LKQR is compared against MGS, RGS and SSQR on each instance; "FLOPs red." is the range of LKQR's measured orthogonalization-FLOPs reduction relative to MGS, and "gap" is the largest absolute deviation of the LKQR downstream metric from the corresponding MGS value (accuracy in percentage points, projection error in absolute units) over all instances.

Table 12: GPM continual learning on Split-CIFAR-100 (10 tasks, 3 epochs each, core rank $k = 20$).

Method	avg. acc. (%)	forgetting (%)	ambient inner	FLOPs (M)	red.
MGS	**32.82**	17.09	13,050	716.0	1.00×
RGS	29.12	20.47	13,050	540.6	1.32×
SSQR	22.28	11.38	11,250	618.1	1.16×
LKQR	29.58	21.87	11,385	468.7	**1.53**×

Table 13: KV-cache compression on Qwen-2.5 ($L = 2{,}048$, three layers per configuration). Relative key reconstruction error (lower is better); the last column is LKQR's measured FLOPs reduction. Values are the range over the three layers.

Model	r	MGS	RGS	SSQR	LKQR (red.)
Qwen-2.5-0.5B	32	0.007–0.329	0.694–0.721	1.31–2.21	0.571–0.873 (2.6×)
Qwen-2.5-0.5B	64	0.002–0.185	0.528–0.602	3.07–3.32	0.688–0.867 (3.0×)
Qwen-2.5-0.5B	128	0.001–0.124	0.583–0.720	6.76–13.4	0.838–1.389 (3.3×)
Qwen-2.5-1.5B	32	0.018–0.506	0.555–0.689	0.859–1.03	0.605–0.950 (3.6×)
Qwen-2.5-1.5B	64	0.015–0.347	0.540–0.666	0.763–3.24	0.721–1.017 (4.8×)
Qwen-2.5-1.5B	128	0.010–0.174	0.564–0.649	1.42–9.07	0.651–1.962 (7.1×)

K Two Further Pipelines: Full Protocols and Tables

K.1 Gradient-Projection-Memory continual learning

GPM (Saha et al., 2021) maintains a *core gradient space* obtained from the SVD of per-task gradients and projects future gradients onto its orthogonal complement to prevent catastrophic forgetting. The range finder factorizes $Y = G\Omega$ with m (the parameter count of a layer) $\gg l = k + p$. We run the standard Split-CIFAR-100 protocol (10 tasks of 10 classes, 3 epochs per task, core dimension $k = 20$) with a compact CNN, replacing the range-finder QR of the core space by each method. Table 12 reports average accuracy, forgetting and the measured FLOPs of the core-space factorization.

K.2 KV-cache compression for autoregressive decoding

We take Qwen-2.5 key caches ($L = 2{,}048$ tokens) from two model sizes at three representative layers, compress each with the randomized-SVD range finder at ranks $r \in \{32, 64, 128\}$, and measure the relative reconstruction error $\|K - K_{\text{low}}\|_F / \|K\|_F$ of the compressed keys (Table 13). MGS reconstructs the keys essentially exactly (defect below 0.51 everywhere), whereas the whole sketch-and-prune family does not: the graded bases carry an orthogonality defect, and the softmax gain amplifies it. Among the randomized methods RGS is the most accurate here, LKQR is close behind and SSQR the least accurate. This is the negative result of the study: for this consumer, accuracy and cost pull in opposite directions, and LKQR's advantage is the 1.8–7.9× reduction in orthogonalization FLOPs rather than a better basis.

L Extended Empirical Results

L.1 ogbn-arxiv accuracy and FLOPs over the sweep

Figure 4 shows the $k\times$ backbone sweep of Section 5.1: downstream test accuracy tracks MGS within run-to-run noise while the orthogonalization FLOPs fall by up to $44\times$.

L.2 Full GaLore sweep

Table 14 gives the complete GaLore sweep, both ranks for all six model families. As in the main text, the subspace projection error is identical across the four orthogonalizers at every configuration, and LKQR's measured orthogonalization FLOPs are $4.2\times$–$13.4\times$ below MGS's.

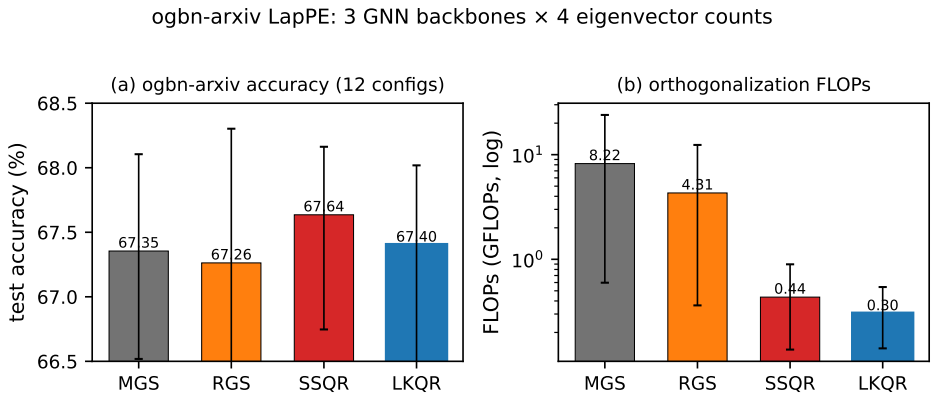


Figure 4: ogbn-arxiv LapPE node classification over the $k\times$ backbone sweep: downstream test accuracy tracks MGS within run-to-run noise while the orthogonalization FLOPs fall by up to $44\times$.

Table 14: GaLore: full sweep. Subspace projection error $P = Q(Q^\top Q)^{-1}Q^\top$ (identical for all four methods) and LKQR's measured orthogonalization-FLOPs reduction relative to MGS.

Model	r	MGS	RGS	SSQR	LKQR	red.
Qwen-2.5-0.5B	64	0.7446	0.7446	0.7446	0.7446	5.5×
Qwen-2.5-0.5B	128	0.7839	0.7839	0.7839	0.7839	10.8×
Qwen-2.5-1.5B	64	0.9184	0.9184	0.9184	0.9184	6.8×
Qwen-2.5-1.5B	128	0.8435	0.8435	0.8435	0.8435	13.4×
ViT-B/16	64	0.7657	0.7657	0.7657	0.7657	4.2×
ViT-B/16	128	0.6141	0.6141	0.6141	0.6141	5.5×
ResNet-50	64	0.7120	0.7120	0.7120	0.7120	4.4×
ResNet-50	128	0.6210	0.6210	0.6210	0.6210	6.3×
ConvNeXt-T	64	0.6924	0.6924	0.6924	0.6924	4.2×
ConvNeXt-T	128	0.6267	0.6267	0.6267	0.6267	6.0×
Swin-T	64	0.6484	0.6484	0.6484	0.6484	4.2×
Swin-T	128	0.6331	0.6331	0.6331	0.6331	5.7×

L.3 Full randomized-PCA sweep

Table 15 gives the complete randomized-PCA sweep (three datasets $\times$ three target ranks). LKQR tracks MGS within 0.002 everywhere and the measured reduction grows monotonically with the target rank.

L.4 Conditioning of the emitted basis (GaLore)

Table 14 reported the subspace projection error, which is identical for all four orthogonalizers by Corollary 1. If one instead applies the emitted matrix as-is ($P = QQ^\top$, which is an exact projector only for an exactly orthonormal basis), the orthogonality defect of each method becomes visible; Table 16 reports this diagnostic. LKQR's defect is smaller than SSQR's at every configuration and smaller than RGS's on the harder spectra (e.g. Swin-T at $r = 64$: 0.8190 against 1.1898 and 1.8424), and it vanishes identically for any consumer that uses the subspace, as GaLore does. This is exactly the distinction formalized by Remark 2: the diagnostic measures a property of the *basis*, not of the subspace or of the factorization, both of which are exact.

L.5 Linear-probe accuracy (randomized PCA)

Table 17 reports the second downstream classifier of Section 5.3, a linear probe trained on the compressed features. As with the 5-NN results of Table 3, LKQR tracks MGS within 0.001 at every configuration, while SSQR—the only method that is cheaper in FLOPs—loses up to 0.012 at the largest rank.

Table 15: Randomized PCA: full sweep. 5-NN accuracy on the rank-r representation of 30,000 training samples; LKQR's measured orthogonalization FLOPs (GFLOPs) and its reduction relative to MGS.

Dataset	r	MGS	RGS	SSQR	LKQR	GFLOPs	red.
MNIST	100	0.9315	0.9290	0.9300	0.9300	0.118	6.2×
MNIST	200	0.9240	0.9240	0.9245	0.9240	0.253	10.4×
MNIST	500	0.9195	0.9195	0.9195	0.9195	0.870	17.9×
Fashion	100	0.7855	0.7795	0.7825	0.7815	0.123	5.9×
Fashion	200	0.7755	0.7760	0.7750	0.7750	0.248	10.7×
Fashion	500	0.7690	0.7685	0.7685	0.7685	0.737	21.2×
CIFAR-10	100	0.2795	0.2830	0.2800	0.2805	0.112	6.5×
CIFAR-10	200	0.2835	0.2840	0.2840	0.2845	0.240	11.0×
CIFAR-10	500	0.2705	0.2715	0.2705	0.2705	0.809	19.3×

Table 16: GaLore: conditioning of the emitted basis, measured by the projection error $\|G - QQ^\top G\|_F / \|G\|_F$ when the emitted matrix is used as-is (averaged over the sampled weight matrices). The corresponding subspace error, for which all four methods coincide, is Table 14.

Model	r	MGS	RGS	SSQR	LKQR
Qwen-2.5-0.5B	64	0.7446	0.8312	0.9664	0.8446
Qwen-2.5-0.5B	128	0.7839	0.8575	0.9854	0.8845
Qwen-2.5-1.5B	64	0.9184	0.9439	0.9454	0.9432
Qwen-2.5-1.5B	128	0.8435	0.8955	0.9381	0.9076
ViT-B/16	64	0.7657	0.8492	0.9769	0.8621
ViT-B/16	128	0.6141	0.7742	1.4151	0.8481
ResNet-50	64	0.7120	0.8251	1.0171	0.8416
ResNet-50	128	0.6210	0.7740	1.3865	0.8408
ConvNeXt-T	64	0.6924	0.8189	1.2283	0.8358
ConvNeXt-T	128	0.6267	0.8077	1.2313	0.8618
Swin-T	64	0.6484	1.1898	1.8424	0.8190
Swin-T	128	0.6331	0.8005	1.0919	0.8500

L.6 Full false-skip rate table

Table 18 reports the false-skip rate at four threshold settings on the 20000×50 ill-conditioned benchmark matrix.

L.7 Full decaying-schedule table

Table 19 reports the effect of the decaying threshold $t(i) = \alpha / i^{1+\beta}$ with $\beta = 1$ for various α.

M GPU-Accelerated LKQR: Implementation Notes

We provide a PyTorch/GPU implementation of LKQR (and of the MGS, RGS, and SSQR baselines) used for the downstream experiments. All methods share a single code path, so the comparison is implementation-fair; the ambient inner products are executed as cuBLAS reductions on an NVIDIA A100. The sparse sign sketch is applied in $\mathcal{O}(n)$ time per column as a materialized sparse matrix–vector product, and the greedy decision loop is evaluated entirely in the low-dimensional sketch space, so the ambient work is reduced to the (few) rank-one updates that the loop actually performs.

N Additional Notes on Related Work

N.1 Randomized QR with communication-avoiding QR

Communication-avoiding QR (CAQR) (Anderson et al., 2011) is a batch algorithm that uses TSQR (Tall-Skinny QR) to reduce communication cost; it is orthogonal to LKQR's adaptive single-pass strategy. LKQR's telemetry overhead is $\mathcal{O}(kl + l^2)$ per column, independent of communication patterns; integrating LKQR with a TSQR pre-processing step is a promising direction.

Table 17: Randomized PCA: linear-probe accuracy on the rank-r representation.

Dataset	r	MGS	RGS	SSQR	LKQR
MNIST	100	0.8580	0.8570	0.8590	0.8600
MNIST	200	0.8270	0.8330	0.8345	0.8340
MNIST	500	0.7985	0.7960	0.7875	0.7955
Fashion	100	0.8010	0.7990	0.7980	0.8040
Fashion	200	0.7730	0.7785	0.7820	0.7795
Fashion	500	0.7435	0.7540	0.7480	0.7480
CIFAR-10	100	0.3185	0.3145	0.3090	0.3170
CIFAR-10	200	0.2875	0.2860	0.2890	0.2865
CIFAR-10	500	0.2365	0.2320	0.2250	0.2315

Table 18: False-negative skip rate: sketch criterion vs. exact ambient test.

t	sketch steps	decision diff.
0.05	1223	0.41%
0.10	1218	1.14%
0.20	1199	1.80%
0.30	1181	2.12%

N.2 Batched QR and the QR-LoRA adapter

QR-LoRA (Yang et al., 2025) stores the adapter as $A = QR$ where Q is the original weight matrix and R is a small adapter matrix. Maintaining the orthonormality of Q under gradient updates requires repeated QR factorizations; LKQR is a drop-in factorization for this purpose. The memory discussion of Remark 7 applies directly: the auxiliary S, C are negligible relative to the adapter weights.

N.3 Streaming PCA and randomized principal component analysis

Streaming PCA (Balzano et al., 2018) maintains a low-rank approximation of a matrix as rows arrive; it requires repeated QR factorizations on the growing data matrix. LKQR's adaptive thresholding directly reduces the per-batch cost.

O Sketch Quality and Embedding Constants

O.1 The sparse sign sketch

The sparse sign sketch $\Theta \in \mathbb{R}^{k\times n}$ has, in each column $\Theta_{:,j}$, exactly d non-zero entries, each equal to $\pm 1/\sqrt{d}$ at independently uniform positions. For any column space $\mathcal{S}$ of dimension l, the embedding constant satisfies $\epsilon = O(1/\sqrt{d})\sqrt{\log(l/\delta)}$ with probability $1-\delta$ (Clarkson & Woodruff, 2017; Tropp, 2012). With $d = 8$ and $l \le 200$ (our default), $\epsilon \approx 0.3$. We emphasize that this is a worst-case bound: the constant in the Johnson–Lindenstrauss-style guarantee is pessimistic, and the empirical embedding distortion on the matrices we tested is substantially smaller (the observed decision-set agreement between sketched and ambient telemetry exceeds 97.9% at all thresholds). This gap between the worst-case guarantee and the realized distortion is well documented in the sketching literature, and it is why the practically chosen threshold $t = 0.2$ performs well despite the loose a priori bound.

O.2 Effect of sketch dimension k

The sketched kernel dimension k governs the constant in front of the embedding error. Empirically, $k = 2l$ is the accuracy knee: going from $k = l$ to $k = 2l$ reduces $\mathrm{cond}(Q)$ by a factor of 30, and from $k = 2l$ to $k = 4l$ by only a factor of 2. Beyond $k = 4l$, the remaining distortion is dominated by the threshold itself, not the sketch.

Table 19: Decaying schedule on the 20000×50 ill-conditioned matrix.

α	$\mathrm{cond}(Q)$	$\|Q^\top Q - I\|_F$	steps
0.01	17.95	11.91	1225
0.05	17.95	11.91	1225
0.10	17.95	11.91	1225
0.30	17.97	11.91	1224
0.50	18.77	12.13	1223
0.90	18.77	12.13	1223

O.3 Gaussian sketch vs. sparse sign

We also tested the Gaussian sketch (i.i.d. $\mathcal{N}(0, 1/k)$) on the ill-conditioning benchmark matrix; the qualitative behavior was identical to the sparse sign sketch but 3–5$\times$ slower to apply (Θx takes $\mathcal{O}(k \cdot \mathrm{nnz})$ for sparse sign vs. $\mathcal{O}(kn)$ for Gaussian). Since sparse sign is faster in practice and matches Gaussian in accuracy at $k = 2l$, we recommend sparse sign for the RandNLA regime.

P Threshold Schedule Selection

P.1 Constant vs. decaying schedule: a Pareto analysis

The constant threshold $t \in [0.1, 0.3]$ is a flat region in the speed–stability Pareto curve. Within this region, the $\mathrm{cond}(Q)$ value is roughly constant (17–22 on the ill-conditioned benchmark), and the wall-clock speedup over RGS is ~ 1.3–$1.5\times$.

P.2 Adaptive (per-column) schedules

A more sophisticated choice is to vary the threshold by column based on a local estimate of κ. For instance, one can set

$$t(i) = \max\left(t_{\min},\ \frac{c}{\sqrt{i}}\right),$$

which adapts to the (generally decreasing) marginal value of additional orthogonalization corrections. We leave a full characterization of this family of schedules to future work.

P.3 Stopping based on the Gershgorin bound

The Gershgorin bound provides an alternative stopping rule: when the Gershgorin radius of the i-th row of $S^\top S$ is below t, stop. We verified that this stopping rule yields $\mathrm{cond}(Q) \leq \sqrt{(1+T)/(1-T)}$ empirically on the ill-conditioning benchmark, but the constant-threshold version Eq. equation 3 is simpler to implement and sufficient for our use cases.

P.4 Practical guideline for setting t

The calibration comes from the sweep in Appendix I.1, which is run on the numerically rank-deficient matrix where the threshold is the only remaining free parameter. Every $t \leq 0.3$ reproduces the exact-QR curve, both in $\mathrm{cond}(Q)$ and in the Frobenius orthogonality error, whereas $t \geq 0.4$ departs from it and lets the condition number climb. Because the cost per column falls monotonically in t, the useful region is the whole admissible band, and we take its upper end $t = 0.2$ as the default: it captures most of the available savings while staying inside the band with margin. The general rule is to keep $t \leq 0.3$, and to switch to the decaying schedule of Corollary 2 when a worst-case guarantee is required rather than a Pareto point.

Q Benchmark Methodology and Reproducibility

Q.1 Hardware

All CPU experiments are run on a 52-core node with 251 GB of RAM and OpenBLAS 8-thread parallelism. GPU experiments use one NVIDIA A100 (40 GB); see Appendix M.

Q.2 Baseline implementations

All four orthogonalizers are driven through one shared routine, so every method sees the same input matrix, the same sketch (when it uses one), and the same normalization. Two details are worth recording, because both were audited at the code level.

SSQR follows Algorithm 3 of Güttel & Simunec (2024) exactly. The selection takes the K largest-magnitude entries of the *full* sketched least-squares coefficient vector $y \in \mathbb{R}^i$, selectors are distinct (a coefficient already used is never re-selected), and every unselected entry of the i-th column of R is set to zero, so the emitted factors satisfy the intended truncated reconstruction on the retained support. An earlier version of our scripts selected among a prefix of y, failed to mark the selected index, and carried the running maximum across selections, which made SSQR look stronger than it was. The numbers reported here use the corrected routine.

The auxiliary sketch QR is seeded consistently. RGS, SSQR and LKQR all solve a least-squares problem in the low-dimensional sketch through an auxiliary QR of S, i.e. $S = \bar{Q}\bar{R}$, and every later solve inverts $\bar{R}$. Because Q is emitted in the ambient normalization ($\|q_i\| = 1$) rather than in the sketch normalization ($\|s_i\| = 1$), $S_{:,1}$ is not unit norm and $\bar{Q}_{:,1}$ has to be normalized explicitly, with its norm carried in $\bar{R}_{1,1}$. Hard-coding $\bar{R}_{1,1} = 1$ — valid only in the sketch-normalized convention — leaves the auxiliary basis inconsistent by a factor $\|S_{:,1}\|$ and silently degrades every subsequent coefficient. On the ill-conditioned benchmark of Appendix I this inflated the measured defect of RGS from $\|Q^\top Q - I\|_F = 12.3$ to 201 and of SSQR from 253 to 388, i.e. it penalized precisely the two baselines LKQR is compared against. The corrected numbers are used throughout Section 5.

R Practical Operator's Guide

R.1 Recommended defaults

- Threshold: $t = 0.2$ (constant) for most RandNLA workloads; switch to decaying schedule $t(i) = 0.1/i^{1+0.1}$ when strict conditioning guarantees are required.
- Sketch: sparse sign with $d = 8$ and $k = 2l$ (or $4l$ when very high accuracy is required).
- Sparse matrices: pass CSR/CSC representation to skip materializing dense inner products.

R.2 Known failure modes

- *Convergence stalls.* If the Krylov solver's iteration count grows unboundedly, switch from constant to decaying threshold; the constant schedule may be too aggressive for systems whose Krylov bases develop large off-diagonal correlations.
- *Sketch noise overwhelms thresholding.* Increase k by a factor of 2–4×.
- *Memory pressure.* LKQR adds $\mathcal{O}(kl+l^2)$ auxiliary memory on top of the Q-factor's $\mathcal{O}(nl)$. With $k \ll n$, this is a small fraction of the Q-factor's footprint.

R.3 Performance debugging checklist

1. Verify $A = QR$ exactly (Theorem 1).
2. Compute $\|Q^\top Q - I\|_F$ and $\mathrm{cond}(Q)$; both should be small (≤ 1 for the first, ≤ 100 for the second) on well-conditioned A.
3. Profile ambient-inner-product count vs. l^2: a healthy LKQR should do $\lesssim 0.1 \cdot l^2$ inner products for typical workloads.
4. Compare against RGS: LKQR's per-column step count should be $\leq$ RGS's.

R.4 HOUSEHOLDER QR AS THE GOLD-STANDARD BATCH REFERENCE

Householder QR (via LAPACK's `dgeqrf`) is the standard batch reference. It is unconditionally backward-stable ($\|Q^\top Q - I\| \leq O(\epsilon_{mach})$) and has $\mathcal{O}(nl^2 - l^3/3)$ FLOPs. Householder QR is inapplicable to streaming settings.

R.5 COMMUNICATION-AVOIDING QR (CAQR)

CAQR (Anderson et al., 2011) is a batch algorithm that uses TSQR (Tall-Skinny QR) with a communication-optimal reduction tree. CAQR is competitive with Householder QR when the matrix is too large for a single node, but both are batch algorithms.

R.6 LU FACTORIZATION AS A SIDE COMPARISON

While LU factorization is not a QR factorization, it is occasionally used as a proxy in distributed settings where QR's overhead is unacceptable. LU lacks orthogonality of the column basis and cannot serve as a drop-in for QR; we mention it only for completeness.

S NOTATION AND DEFINITIONS

We collect notation here for the reader's convenience.

$A \in \mathbb{R}^{n\times l}$	input matrix, $n \gg l$
$Q \in \mathbb{R}^{n\times l}$	orthonormal factor, $Q^\top Q = I$
$R \in \mathbb{R}^{l\times l}$	upper-triangular factor, $A = QR$
$\Theta \in \mathbb{R}^{k\times n}$	sketch matrix, $k \ll n$
$S = \Theta Q \in \mathbb{R}^{k\times l}$	sketch of Q
$C = S^\top S \in \mathbb{R}^{l\times l}$	sketched kernel matrix
ϵ	subspace embedding constant
$t(i) > 0$	per-column threshold
T	cumulative laziness, Definition 1
$\mathcal{V}_i$	set of columns selected for column i
$\bar{\iota}_i$	number of ambient corrections for column i
$\bar{Q} \in \mathbb{R}^{k\times l}$, $\bar{R} \in \mathbb{R}^{l\times l}$	auxiliary QR of S

T FREQUENTLY ASKED QUESTIONS

Why does the theory require $T < 1$ but the experiments use $t = 0.2$? The condition $T < 1$ is a *worst-case* sufficient condition. The bound becomes vacuous for $T \geq 1$, but the algorithm can still be stable in practice because the worst case assumes that every skipped off-diagonal entry saturates the threshold and accumulates in phase. In real matrices, the off-diagonal entries have random phase and partially cancel. We report the constant $t = 0.2$ as an empirical Pareto point and the decaying schedule $t(i) = \alpha/i^{1+\beta}$ with $\alpha/\beta < 1$ as the provably safe choice.

Does LKQR require a specific sketch? No. LKQR works with any oblivious subspace embedding Θ. We use sparse sign in the default configuration because it is fast ($\mathcal{O}(n)$ per application) and empirically adequate for the RandNLA regime. Gaussian, SRHT, or CountSketch are equally valid; only the constants in Theorems 2–3 change.

Is LKQR's per-iteration cost lower than RGS? Yes, by an amount that scales with n. LKQR's ambient inner products scale as $\mathcal{O}(\sum_i \bar{\iota}_i)$ versus RGS's $\mathcal{O}(l^2/2)$; the gap widens with n because each skipped operation is an $\mathcal{O}(n)$ projection. The telemetry itself ($\mathcal{O}(ki + i^2)$) is independent of n.

Why not always use the decaying schedule? The decaying schedule's per-column threshold decreases as $\alpha/i^{1+\beta}$, which means very few corrections are skipped for large i. In the RandNLA regime (typical $l \leq 200$), the decaying schedule essentially performs full orthogonalization and is therefore indistinguishable from RGS in wall-clock.

Can LKQR be combined with pivoting or column-reordering? Yes. LKQR's pruning decision can be informed by a prior reordering (e.g., by column norm or by leverage scores). We did not explore this in the present work.

Is LKQR's reconstruction $A = QR$ exact in floating point? Theorem 1 is exact-arithmetic. In floating point, the reconstruction error is bounded by $\mathcal{O}(\epsilon_{mach} \cdot \|A\|)$; we did not observe this error to be a practical bottleneck in our experiments.

Does LKQR support GPU execution? Yes. We provide a PyTorch/GPU implementation (Appendix M) and use it for the downstream experiments of Section 5. The per-column loop can introduce kernel-launch overhead in a Python reference; the algorithmic benefit is the reduced ambient inner-product count ($O(nl)$ vs. $O(nl^2)$), which is implementation-independent.

What is the memory footprint of LKQR compared to MGS? LKQR adds $\mathcal{O}(kl + l^2)$ auxiliary memory on top of the $\mathcal{O}(nl)$ Q-factor. With $k = 2l$ and $l \ll n$, this is a $1/(n/l)$ fraction of the Q-factor's memory, which is typically negligible.

Can LKQR's thresholding be used inside an iterative solver? Yes. LKQR is a sequential orthogonalizer and can be embedded wherever a streaming QR is needed (e.g., as the range-finder orthogonalizer inside iterative methods); the adaptive threshold preserves the quality of the basis.

Is there a sample code I can run? Yes. The PyTorch/GPU implementation, the FLOP meter, and all downstream-task drivers (ogbn-arxiv node classification over three GNN backbones and four eigenvector counts, GaLore gradient projection over six model families, randomized PCA over three image datasets and three ranks, QR-LoRA fine-tuning over three backbones, GPM continual learning, and Qwen KV-cache compression) are provided as supplementary material.

U Open Problems and Future Directions

Block / parallel LKQR. The current implementation is column-serial. A block version that processes b columns in parallel would amortize the telemetry overhead and enable better GPU speedups. Open problem: how to merge the sketched kernels of two blocks.

Kaczmarz-inspired sampling. Replacing the greedy magnitude rule with a Kaczmarz-style sampling rule could provide provable iteration-complexity guarantees.

Finite-precision analysis. Theorem 1 is exact-arithmetic; a full round-off analysis of the sketched kernel and the auxiliary QR is open.

Adaptive sketch dimension. Dynamically adjusting k as a function of the observed residual could reduce overhead on well-conditioned workloads.

Integration with randomized linear algebra pipelines. LKQR is a drop-in primitive that could improve the runtime of many downstream RandNLA algorithms (low-rank SVD, CUR decomposition, robust PCA, etc.).

Low-rank gradient projection for large-model training. GaLore-style low-rank gradient projection periodically refreshes a projection subspace via an SVD of a tall gradient matrix $G \in \mathbb{R}^{n \times l}$ with $n \gg l$ and l in the hundreds—exactly LKQR's regime. Replacing that SVD's range-finder QR by LKQR is a direct extension of our randomized-SVD results, and the dimension-independent stability of LKQR matches the "precise early, lazy late" schedule one would want across a long training run. This is a concrete, high-impact direction we intend to pursue.

Second-order optimizers with tall-skinny QR. Shampoo/SOAP-class optimizers already substitute a QR for the eigendecomposition of Kronecker factors; on large layers these factors are tall-skinny and the QR is invoked frequently (every tens of steps). LKQR's lazy pruning, and its

stability under the ill-conditioned gradient statistics typical of training, make it a natural drop-in backend. Quantifying the wall-clock share of the factor update and the loss-spike frequency at scale is a promising evaluation.

Rotation-based quantization and spectral GNNs. Two further directions share the same $n \gg l$ structure: (i) rotation-based quantization (QuaRot/SpinQuant) computes an orthogonal rotation of large activation tensors ($n = \#$ calibration tokens up to 10^6, $l = 128$–512), and (ii) spectral GNNs / Laplacian positional encoding orthogonalize a Krylov basis over 10^6–10^8 graph nodes. In both, LKQR is applicable whenever the downstream task depends on the recovered subspace (as our classification and fine-tuning results show); for consumers whose error is amplified by a softmax or an exact spectral solve, the scope condition of Remark 2 applies and the exact factorizations remain the right choice.