

Lectures 8 & 9: Randomized Linear Algebra in LLMs

Instructors: *Alex Andoni, Edo Liberty*Lecturer: *Edo Liberty*

Lecture Overview & Pedagogical Roadmap

Modern AI and Large Language Models (LLMs) are bottlenecked by large-scale matrix operations: searching across millions of high-dimensional embeddings in $\mathbb{R}^d$, storing and attending over 128k-token KV-caches, comparing multi-billion-parameter gradients, and computing low-rank approximations of massive weight matrices. Exact deterministic linear algebra hits quadratic and cubic walls ($\Theta(nd)$ distance scans, $\Theta(mdn)$ matrix multiplications, and $\Theta(mn^2)$ Singular Value Decompositions).

In these two 90-minute sessions, we develop **Randomized Linear Algebra** from first principles:

1. **Session 1 — Random Projections (Sections 1–3):** After motivating fast approximate matrix operations in AI (1.a), we use the rotational invariance of Gaussians and χ_k^2 concentration to prove that projecting a single unit vector with a random Gaussian matrix $\frac{1}{\sqrt{k}}R$ preserves its norm within $(1 \pm \varepsilon)$, and then apply this to $u = \frac{x-y}{\|x-y\|_2}$ across all pairs to establish the **Johnson-Lindenstrauss (JL) Lemma** [3, 1] (1.b). We conclude with concrete AI applications of JL projections (1.c).
2. **Session 2 — Singular Value Decomposition & Power Methods (Sections 4–6):** Moving from data-oblivious projections to **data-aware** low-rank structure, we define the **SVD** ($A = U\Sigma V^T$), explain its geometric meaning, and state its optimal rank- k approximation properties (2.1). We briefly outline the classical deterministic $\Theta(mn^2)$ approach (2.2), explain the **Randomized Power Method** intuitively, and then prove that the **Block Power Method (Simultaneous Iteration)** [7, 2, 5, 4] achieves a $(1 + \varepsilon)\sigma_{k+1}$ spectral-norm guarantee **completely independent of any spectral gap** (2.3).

Session 1: Random Projections

1 Motivating Fast and Approximate Matrix Operations in AI

In modern AI and LLM pipelines, three fundamental linear algebra bottlenecks recur at massive scale (hidden dimension $d = 4,096$ – $16,384$, sequence length $n = 128,000$, corpus size $N = 10^7$ – 10^9 , parameter count $P = 7 \times 10^{10}$):

1. **High-Dimensional Distance and Inner-Product Computations (Xq and QK^T):**
 - *Vector Search & RAG:* Given n embedding vectors $x_1, \dots, x_n \in \mathbb{R}^d$ and a query $q \in \mathbb{R}^d$, computing exact distances $\|q - x_i\|_2$ or inner products $\langle q, x_i \rangle$ requires $\Theta(nd)$ time and memory.

- *Gradient Attribution (TRAK [6])*: Tracing how training example z_i influences a test prediction requires computing inner products of parameter gradients $\nabla_{\theta}\mathcal{L}(z_i; \theta) \in \mathbb{R}^P$. At $P = 7 \times 10^{10}$, a single gradient vector in FP16 occupies 140 GB, making storage and inner products across n examples impossible unless we project $\mathbb{R}^P \rightarrow \mathbb{R}^k$ ($k \ll P$).
2. **Dense Matrix-Matrix Multiplication (AB)**: Multiplying $A \in \mathbb{R}^{m \times d}$ by $B \in \mathbb{R}^{d \times n}$ takes $\Theta(mdn)$ operations. If a random projection $\Pi \in \mathbb{R}^{k \times d}$ ($k \ll d$) approximately preserves inner products, we can approximate $AB \approx (A\Pi^T)(\Pi B)$ in $\mathcal{O}(mkn)$ time.
 3. **Low-Rank Approximation & Spectral Decomposition** ($A \approx U_k \Sigma_k V_k^T$): Compressing weight matrices, initializing adapters (**LoRA**), or projecting gradients onto low-rank subspaces (**GaLore** [9]) requires the top- k singular vectors of $m \times n$ matrices, where deterministic SVD costs $\Theta(mn^2)$ even for $k \ll n$.

Why Randomness Is Necessary. Any *deterministic* matrix $\Pi \in \mathbb{R}^{k \times d}$ ($k < d$) has a null space of dimension at least $d - k > 0$, so an adversarial vector $x \in \text{null}(\Pi)$ is mapped to $\Pi x = 0$ (100% distortion). By drawing Π **at random** independently of the data, every fixed vector x has its norm preserved within $(1 \pm \varepsilon)$ with overwhelmingly high probability.

2 Proving Dense Gaussian Random Projections (JL) from Scratch

We now prove the **Johnson-Lindenstrauss (JL) Lemma** [3, 1] using a dense matrix of i.i.d. standard Gaussians.

Theorem 2.1 (Johnson-Lindenstrauss Lemma via Gaussian Projections [3, 1]). *Let $x_1, \dots, x_n \in \mathbb{R}^d$ be any set of n points in $\mathbb{R}^d$ and let $\varepsilon \in (0, 1/2)$. Let $R \in \mathbb{R}^{k \times d}$ be a random matrix whose entries are drawn i.i.d. from the standard normal distribution, $R(i, j) \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$, with target dimension:*

$$k = \mathcal{O}\left(\frac{\log n}{\varepsilon^2}\right).$$

Then the linear projection $\Pi = \frac{1}{\sqrt{k}}R : \mathbb{R}^d \rightarrow \mathbb{R}^k$ preserves all $\binom{n}{2}$ pairwise Euclidean distances within $(1 \pm \varepsilon)$ simultaneously:

$$(1 - \varepsilon)\|x_i - x_j\|_2 \leq \left\| \frac{1}{\sqrt{k}}Rx_i - \frac{1}{\sqrt{k}}Rx_j \right\|_2 \leq (1 + \varepsilon)\|x_i - x_j\|_2 \quad \forall 1 \leq i < j \leq n, \quad (1)$$

with probability at least $1/2$.

Proof. Step 1 (Rotational Invariance / 2-Stability of Gaussians): Recall the **2-stability** (rotational invariance) of the normal distribution: if $z_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$ and $z_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$ are independent, their sum is also normal with $z_1 + z_2 \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$. In particular, if $r(1), \dots, r(d) \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$, then for any fixed vector $u \in \mathbb{R}^d$, each term $r(j)u(j) \sim \mathcal{N}(0, u(j)^2)$, and hence their sum is distributed as:

$$z = \sum_{j=1}^d r(j)u(j) \sim \mathcal{N}\left(0, \sum_{j=1}^d u(j)^2\right) = \mathcal{N}(0, \|u\|_2^2). \quad (2)$$

Step 2 (Projecting a Single Unit Vector $u \in \mathbb{S}^{d-1}$): Fix any unit vector $u \in \mathbb{S}^{d-1}$ ($\|u\|_2 = 1$). Consider the i -th coordinate of the projected vector $Ru \in \mathbb{R}^k$:

$$z_i \triangleq (Ru)_i = \sum_{j=1}^d R(i, j)u(j).$$

By (2) and $\|u\|_2 = 1$, each coordinate is distributed as a standard normal $z_i \sim \mathcal{N}(0, 1)$. Moreover, because the k rows of R are mutually independent, $z_1, \dots, z_k \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$ are independent standard normal variables. Therefore, the squared norm of $\frac{1}{\sqrt{k}}Ru$ is distributed identically to the average of k independent squared standard Gaussians:

$$\left\| \frac{1}{\sqrt{k}}Ru \right\|_2^2 = \frac{1}{k} \sum_{i=1}^k z_i^2, \quad (3)$$

where $\sum_{i=1}^k z_i^2 \sim \chi_k^2$ follows the **chi-squared distribution with k degrees of freedom**, and $\mathbb{E}[\left\| \frac{1}{\sqrt{k}}Ru \right\|_2^2] = \frac{1}{k} \sum_{i=1}^k \mathbb{E}[z_i^2] = 1$.

Step 3 (Concentration of χ_k^2): By the standard tail concentration of the χ_k^2 distribution around its mean k (stated here without proof):

$$\Pr \left[\left| \sqrt{\frac{1}{k} \sum_{i=1}^k z_i^2} - 1 \right| > \varepsilon \right] \leq \Pr \left[\left| \frac{1}{k} \sum_{i=1}^k z_i^2 - 1 \right| > \varepsilon \right] \leq 2e^{-k\varepsilon^2/8}. \quad (4)$$

Choosing $k = \left\lceil \frac{8 \ln(2n^2)}{\varepsilon^2} \right\rceil = \mathcal{O}\left(\frac{\log n}{\varepsilon^2}\right)$ ensures $2e^{-k\varepsilon^2/8} \leq \frac{1}{n^2}$. Thus, for **any single unit vector** $u \in \mathbb{S}^{d-1}$:

$$\Pr \left[\left| \left\| \frac{1}{\sqrt{k}}Ru \right\|_2 - 1 \right| > \varepsilon \right] \leq \frac{1}{n^2}. \quad (5)$$

Step 4 (From a Single Unit Vector to Metric Preservation via $u = \frac{x-y}{\|x-y\|_2}$): Now consider any two distinct points x, y in our dataset $\{x_1, \dots, x_n\} \subset \mathbb{R}^d$, and define the unit vector:

$$u = \frac{x - y}{\|x - y\|_2} \in \mathbb{S}^{d-1}.$$

By linearity of matrix multiplication, $\frac{1}{\sqrt{k}}Rx - \frac{1}{\sqrt{k}}Ry = \frac{1}{\sqrt{k}}R(x - y) = \|x - y\|_2 \cdot \left(\frac{1}{\sqrt{k}}Ru\right)$. Therefore:

$$\left\| \frac{1}{\sqrt{k}}Rx - \frac{1}{\sqrt{k}}Ry \right\|_2 = \|x - y\|_2 \cdot \left\| \frac{1}{\sqrt{k}}Ru \right\|_2.$$

By (5), the event that this pair (x, y) violates $(1 - \varepsilon)\|x - y\|_2 \leq \left\| \frac{1}{\sqrt{k}}Rx - \frac{1}{\sqrt{k}}Ry \right\|_2 \leq (1 + \varepsilon)\|x - y\|_2$ occurs with probability at most $\frac{1}{n^2}$. Taking a union bound over all $\binom{n}{2}$ pairs $\{x_i, x_j\}$ ($1 \leq i < j \leq n$), the probability that *at least one* pair fails is at most:

$$\binom{n}{2} \cdot \frac{1}{n^2} = \frac{n(n-1)}{2n^2} < \frac{1}{2}.$$

Hence, with probability at least $1/2$, all $\binom{n}{2}$ pairwise distances are preserved within $(1 \pm \varepsilon)$. $\square$

3 Use Cases of Random Projections in AI

3.1 Preserving Inner Products via Polarization

In AI, we often care about **inner products** $\langle x, y \rangle$ (cosine similarity in RAG, attention logits $q^T k$, or matrix multiplication AB) as much as Euclidean distances $\|x - y\|_2$. Because inner products can be written in terms of Euclidean norms via the polarization identity:

$$\langle u, v \rangle = \frac{\|u + v\|_2^2 - \|u - v\|_2^2}{4},$$

preserving the norms of $u + v$ and $u - v$ within $(1 \pm \varepsilon)$ for unit vectors $u, v \in \mathbb{S}^{d-1}$ immediately guarantees that $\Pi = \frac{1}{\sqrt{k}}R$ preserves their inner product up to additive error ε :

$$|\langle \Pi u, \Pi v \rangle - \langle u, v \rangle| = \frac{1}{4} \left| (\|\Pi(u+v)\|_2^2 - \|u+v\|_2^2) - (\|\Pi(u-v)\|_2^2 - \|u-v\|_2^2) \right| \leq \frac{\varepsilon}{4} (\|u+v\|_2^2 + \|u-v\|_2^2) = \varepsilon. \quad (6)$$

3.2 Concrete Applications in Modern AI

1. **Compressed Vector Search & RAG Indexing:** Given $n = 10^7$ document embeddings in $\mathbb{R}^d$ ($d = 4,096$), projecting each vector to $\tilde{x}_i = \Pi x_i \in \mathbb{R}^k$ ($k \approx 256$ – 512) reduces both index memory and distance evaluation time from $\Theta(nd)$ to $\mathcal{O}(nk)$ while preserving nearest-neighbor rankings within $(1 \pm \varepsilon)$.
2. **Tractable Gradient Attribution for LLMs (TRAK [6]):** To attribute a model’s prediction on z_{test} to training examples $z_1, \dots, z_n$, influence functions compute gradient inner products $g_i^T g_{\text{test}}$ where $g_i = \nabla_{\theta} \mathcal{L}(z_i; \theta) \in \mathbb{R}^P$. Because the JL dimension $k = \mathcal{O}(\varepsilon^{-2} \log n)$ is **completely independent of the ambient dimension P** , TRAK projects billion-dimensional gradients ($P \sim 10^{10}$) down to $k \sim 4,096$ dimensions on the fly, compressing gradient storage by $> 10^6 \times$ while preserving all pairwise gradient inner products.
3. **Approximate Matrix Multiplication (“Sketch-and-Multiply”):** To approximate the matrix product AB for $A \in \mathbb{R}^{m \times d}$ and $B \in \mathbb{R}^{d \times n}$, notice that the (i, j) entry of AB is the inner product $\langle a_{i,:}, b_{:,j} \rangle$ in $\mathbb{R}^d$. Drawing $\Pi = \frac{1}{\sqrt{k}}R \in \mathbb{R}^{k \times d}$ with $k = \mathcal{O}(\varepsilon^{-2} \log(m+n))$ and computing

$$\tilde{C} = (A\Pi^T)(\Pi B)$$

approximates every entry $(AB)_{ij}$ within $\pm \varepsilon \|a_{i,:}\|_2 \|b_{:,j}\|_2$ while reducing the matrix multiplication cost from $\Theta(mdn)$ to $\mathcal{O}((m+n)dk + mkn)$.

Session 2: Singular Value Decomposition (SVD) & Randomized Power Methods

In Session 1, random projections Π were **data-oblivious**, requiring dimension $k = \Theta(\varepsilon^{-2} \log n)$ to preserve geometry across all directions. In Session 2, we study **data-aware** dimensionality reduction via the **Singular Value Decomposition (SVD)**: finding the exact k -dimensional subspace that best captures a given matrix $A \in \mathbb{R}^{m \times n}$, and using randomness to compute it rapidly.

4 Defining the SVD ($A = U\Sigma V^T$) — Intuition and Optimal Low-Rank Properties

4.1 Definition of the Singular Value Decomposition

Definition 4.1 (Singular Value Decomposition). *Any matrix $A \in \mathbb{R}^{m \times n}$ (assume w.l.o.g. $m \leq n$) of rank $r \leq m$ can be decomposed as:*

$$A = \sum_{\ell=1}^r \sigma_{\ell} u_{\ell} v_{\ell}^T = U \Sigma V^T, \quad (7)$$

where:

1. $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > 0$ are the **singular values** of A (with $\sigma_{\ell} = \sqrt{\lambda_{\ell}(AA^T)}$).
2. $U = [u_1, \dots, u_r] \in \mathbb{R}^{m \times r}$ has orthonormal columns ($\langle u_{\ell}, u_{\ell'} \rangle = \delta_{\ell, \ell'}$, so $U^T U = I_r$), called the **left singular vectors** (eigenvectors of AA^T).
3. $V = [v_1, \dots, v_r] \in \mathbb{R}^{n \times r}$ has orthonormal columns ($\langle v_{\ell}, v_{\ell'} \rangle = \delta_{\ell, \ell'}$, so $V^T V = I_r$), called the **right singular vectors** (eigenvectors of $A^T A$).
4. $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r) \in \mathbb{R}^{r \times r}$ is diagonal.

4.2 Intuitive Meaning of $A = U\Sigma V^T$

1. **Geometric Meaning (Rotate $\rightarrow$ Stretch $\rightarrow$ Rotate):** For any input vector $x \in \mathbb{R}^n$, multiplying by A acts in three transparent steps:

$$Ax = U \Sigma V^T x = \sum_{\ell=1}^r \sigma_{\ell} \langle v_{\ell}, x \rangle u_{\ell}. \quad (8)$$

- $V^T x$ (**Input Rotation**): Express x in the orthonormal basis of right singular vectors $v_1, \dots, v_r$ by taking inner products $\langle v_{\ell}, x \rangle$.
 - $\Sigma(V^T x)$ (**Axis Scaling**): Stretch the ℓ -th coordinate by $\sigma_{\ell} \geq 0$, mapping the unit sphere in $\mathbb{R}^n$ into a hyper-ellipsoid with semi-axes $\sigma_1, \dots, \sigma_r$.
 - $U(\Sigma V^T x)$ (**Output Rotation**): Reconstruct the output vector along the orthonormal left singular directions $u_1, \dots, u_r \in \mathbb{R}^m$.
2. **Sum of Rank-1 Outer Products & Signal Compression:** Writing $A = \sum_{\ell=1}^r \sigma_{\ell} u_{\ell} v_{\ell}^T$ decomposes A into a sum of orthogonal rank-1 matrices weighted by σ_{ℓ} . The spectral norm is $\|A\|_2 = \sigma_1$, and the

Frobenius norm is $\|A\|_F = \sqrt{\sum_{\ell=1}^r \sigma_\ell^2}$. If the tail singular values $\sigma_{k+1}, \dots, \sigma_r$ are small, discarding them yields a compressed rank- k representation using $(m+n)k$ parameters instead of mn .

3. SVD vs. Eigendecomposition:

- Every rectangular matrix has an SVD; not every square matrix has an eigendecomposition.
- In $A = X\Lambda X^{-1}$, the eigenbasis X is generally non-orthogonal and can be ill-conditioned; in $A = U\Sigma V^T$, the singular bases U and V are **always orthonormal**.

4.3 Optimal Low-Rank Approximation (Eckart-Young) & PCA

Truncating the SVD to its top k terms $A_k \triangleq \sum_{j=1}^k \sigma_j u_j v_j^T = U_k \Sigma_k V_k^T$ gives the **best possible rank- k approximation** to A in both spectral and Frobenius norms (stated here without proof):

Theorem 4.2 (Eckart-Young-Mirsky: Best Rank- k Approximation). *For any matrix $A \in \mathbb{R}^{m \times n}$ and target rank $k < \text{rank}(A)$, let $A_k = \sum_{j=1}^k \sigma_j u_j v_j^T$. Then A_k simultaneously minimizes the approximation error over all rank- k matrices $B \in \mathbb{R}^{m \times n}$ in both the spectral norm and the Frobenius norm:*

$$\min_{\substack{B \in \mathbb{R}^{m \times n} \\ \text{rank}(B) \leq k}} \|A - B\|_2 = \|A - A_k\|_2 = \sigma_{k+1}, \quad \text{and} \quad \min_{\substack{B \in \mathbb{R}^{m \times n} \\ \text{rank}(B) \leq k}} \|A - B\|_F = \|A - A_k\|_F = \sqrt{\sum_{i=k+1}^m \sigma_i^2}. \quad (9)$$

Principal Component Analysis (PCA). Given n points $x_1, \dots, x_n \in \mathbb{R}^m$ arranged as columns of $A \in \mathbb{R}^{m \times n}$, the rank- k projection matrix $P \in \mathbb{R}^{m \times m}$ minimizing $\sum_{i=1}^n \|Px_i - x_i\|_2^2 = \|PA - A\|_F^2$ is $P = U_k U_k^T$ (since $PA = A_k$). Thus $x_i \mapsto U_k^T x_i \in \mathbb{R}^k$ is the optimal k -dimensional linear representation of $x_1, \dots, x_n$ in squared loss.

5 The Deterministic Slow $\mathcal{O}(mn^2)$ Approach (Without Proofs)

How do classical deterministic algorithms (LAPACK `dgesvd`, torch.`linalg.svd`) compute the SVD of $A \in \mathbb{R}^{m \times n}$ ($m \geq n$), and why do they require $\Theta(mn^2)$ operations?

1. **Why Not Form $A^T A$ Directly?** Forming the Gram matrix $A^T A \in \mathbb{R}^{n \times n}$ already takes $\Theta(mn^2)$ FLOPs, and worse, it **squares the condition number** $\kappa(A^T A) = (\sigma_1/\sigma_n)^2$, wiping out any singular value $\sigma_j \leq \sqrt{\epsilon_{\text{mach}}} \sigma_1$ in floating-point arithmetic.
2. **The Classical Two-Stage Golub-Kahan-Reinsch Algorithm ($\Theta(mn^2)$):**
 - *Stage 1 — Householder Bidiagonalization ($\Theta(mn^2)$ FLOPs):* Apply a sequence of n deterministic left and right **Householder reflections** $H_i = I - 2u_i u_i^T$ and $G_i = I - 2v_i v_i^T$ to zero out entries below the diagonal and to the right of the superdiagonal, reducing A to an upper bidiagonal matrix $B = U_B^T A V_B \in \mathbb{R}^{n \times n}$. Updating the trailing $(m-i) \times (n-i)$ submatrix at each step $i = 1, \dots, n$ costs $\sum_{i=1}^n 4(m-i)(n-i) \approx 4mn^2 - \frac{4}{3}n^3 = \Theta(mn^2)$ operations.
 - *Stage 2 — Implicitly Shifted QR Iteration on B ($\mathcal{O}(n^2)$ to $\mathcal{O}(mn^2)$ FLOPs):* Apply iterative Givens rotations to chase off the superdiagonal entries of B , converging rapidly to $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_n)$.

3. **Why This Is Too Slow for AI:** Even if we only want a **low-rank approximation** of rank $k \ll n$ (e.g., $k = 16$ for LoRA or GaLore on a $16,384 \times 16,384$ matrix), deterministic bidiagonalization must reduce the **entire** matrix A in Stage 1 first, paying the full $\Theta(mn^2)$ upfront cost and destroying any sparsity in A .

6 Randomized SVD — The Power Method and Block Power Method

To avoid the $\Theta(mn^2)$ deterministic bottleneck, we interact with A **only by multiplying it with random vectors or thin random matrices**, reducing the work to $\tilde{O}(mnk)$ operations.

6.1 Intuition for the Standard Power Method

Suppose we want to find the top singular value σ_1 and right singular vector v_1 of $A \in \mathbb{R}^{m \times n}$. We start with a random Gaussian vector $x_0 \sim \mathcal{N}(0, I_n)$ and repeatedly multiply by $A^T A$:

$$x_i \leftarrow A^T(Ax_{i-1}) \quad \text{for } i = 1, \dots, s,$$

where each step costs only $\mathcal{O}(mn)$ operations via two matrix-vector products Ax_{i-1} and $A^T(Ax_{i-1})$.

Why It Works Intuitively. Write the starting vector x_0 in the orthonormal basis of right singular vectors $\{v_1, \dots, v_n\}$:

$$x_0 = \sum_{j=1}^n \alpha_j^0 v_j.$$

Because v_j is an eigenvector of $A^T A$ with eigenvalue σ_j^2 ($A^T A v_j = \sigma_j^2 v_j$), applying $A^T A$ a total of s times scales the j -th coordinate by σ_j^{2s} :

$$x_s = (A^T A)^s x_0 = \sum_{j=1}^n \alpha_j^0 \sigma_j^{2s} v_j = \sigma_1^{2s} \left(\alpha_1^0 v_1 + \sum_{j=2}^n \alpha_j^0 \left(\frac{\sigma_j}{\sigma_1} \right)^{2s} v_j \right). \quad (10)$$

1. **Role of Randomness:** Because $x_0 \sim \mathcal{N}(0, I_n)$ is chosen at random, its initial projection $\alpha_1^0 = \langle x_0, v_1 \rangle \sim \mathcal{N}(0, 1)$ onto the unknown top direction v_1 is non-zero with high probability (whereas a deterministic vector x_0 could accidentally be orthogonal to v_1).
2. **Exponential Amplification of the Top Direction:** Whenever there is a spectral gap $\sigma_1 > \sigma_2$, every ratio $(\sigma_j/\sigma_1)^{2s}$ for $j \geq 2$ decays **exponentially** to 0 as s grows, leaving $x_s/\|x_s\|_2 \approx \pm v_1$. Once (σ_1, u_1, v_1) is found, one can deflate $A \leftarrow A - \sigma_1 u_1 v_1^T$ and repeat k times.
3. **The Spectral Gap Limitation:** If $\sigma_1 \approx \sigma_2$ (or $\sigma_k \approx \sigma_{k+1}$), the ratio $\sigma_j/\sigma_1 \approx 1$ decays very slowly, and sequential deflation stalls. However, this is an artifact of trying to isolate individual singular vectors one at a time rather than finding a **rank- k subspace** simultaneously.

6.2 The Block Power Method (Simultaneous Iteration) & Gap-Independent Analysis

In the **Block Power Method (Simultaneous Iteration)** [7, 2, 5, 4], we multiply $A \in \mathbb{R}^{n \times m}$ by a block of k random Gaussian vectors $G \in \mathbb{R}^{m \times k}$ simultaneously, run t power iterations, and extract an

orthonormal basis $Z \in \mathbb{R}^{n \times k}$ of the resulting subspace:

Algorithm BLOCK-POWER-METHOD($A \in \mathbb{R}^{n \times m}$, target rank k , accuracy ε)

1. Draw $G \in \mathbb{R}^{m \times k}$ with i.i.d. entries $G(i, j) \sim \mathcal{N}(0, 1)$.
2. Set $t \leftarrow \lceil c \cdot \log(n/\varepsilon)/\varepsilon \rceil$.
3. Compute $Y \leftarrow (AA^T)^t AG \in \mathbb{R}^{n \times k}$ via $2t + 1$ matrix multiplications. $// \mathcal{O}(mnkt)$
4. Orthonormalize columns of Y via QR decomposition to obtain $Z \in \mathbb{R}^{n \times k}$ ($Z^T Z = I_k$).
5. **Return** rank- k approximation $ZZ^T A$.

Lemma 6.1 (Gap-Independent Block Power Method [4]). *Let $A \in \mathbb{R}^{n \times m}$ be an arbitrary matrix and let $G \in \mathbb{R}^{m \times k}$ be a matrix of i.i.d. standard Gaussian entries. Let $t = c \cdot \frac{\log(n/\varepsilon)}{\varepsilon}$ and let Z be an orthonormal basis for $\text{span}((AA^T)^t AG)$. Then:*

$$\|A - ZZ^T A\|_2 \leq (1 + \varepsilon)\sigma_{k+1} \quad (11)$$

with high probability depending only on the universal constant c (completely independent of any spectral gap).

Proof. Step 1 (Variational Characterization & Change of Basis): By definition of orthogonal projection onto Z , we have:

$$\|A - ZZ^T A\|_2 = \max_{\substack{x: \|x\|_2=1 \\ \|x^T Z\|_2=0}} \|x^T A\|_2.$$

Substitute the SVD $A = USV^T$ and change variables by setting $x = Uy$ (so $\|y\|_2 = 1$) and $G' = V^T G$. Because V is orthogonal, rotational invariance of the Gaussian distribution guarantees that G' is **also a matrix of i.i.d. standard Gaussian entries**. Since $x^T Z = 0 \iff x^T (AA^T)^t AG = 0 \iff y^T S^{2t+1} G' = 0$, this reduces to:

$$\|A - ZZ^T A\|_2 = \max_{\substack{y: \|y\|_2=1 \\ y^T S^{2t+1} G'=0}} \|y^T S\|_2. \quad (12)$$

Step 2 (Block Partitioning & Coordinate Decay Bound): Break y , S , and G' into two blocks each at index k :

$$y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \quad S = \left(\begin{array}{c|c} S_1 & 0 \\ \hline 0 & S_2 \end{array} \right), \quad G' = \begin{pmatrix} G'_1 \\ G'_2 \end{pmatrix},$$

where $y_1 \in \mathbb{R}^k$, $y_2 \in \mathbb{R}^{n-k}$, $S_1 \in \mathbb{R}^{k \times k}$, $S_2 \in \mathbb{R}^{(n-k) \times (n-k)}$, $G'_1 \in \mathbb{R}^{k \times k}$, and $G'_2 \in \mathbb{R}^{(n-k) \times k}$. Using

$$y^T S^{2t+1} G' = 0:$$

$$\begin{aligned}
0 &= \|y^T S^{2t+1} G'\|_2 = \|y_1^T S_1^{2t+1} G'_1 + y_2^T S_2^{2t+1} G'_2\|_2 \\
&\geq \|y_1^T S_1^{2t+1} G'_1\|_2 - \|y_2^T S_2^{2t+1} G'_2\|_2 \\
&\geq \frac{\|y_1^T S_1^{2t+1}\|_2}{\|(G'_1)^{-1}\|_2} - \|y_2^T\|_2 \cdot \|S_2^{2t+1}\|_2 \cdot \|G'_2\|_2 \\
&\geq \frac{|y_1(i)| \sigma_i^{2t+1}}{\|(G'_1)^{-1}\|_2} - \sigma_{k+1}^{2t+1} \cdot \|G'_2\|_2.
\end{aligned}$$

Rearranging gives, for every coordinate $i \in \{1, \dots, k\}$:

$$|y_1(i)| \leq \left(\frac{\sigma_{k+1}}{\sigma_i} \right)^{2t+1} \|G'_2\|_2 \|(G'_1)^{-1}\|_2. \quad (13)$$

Step 3 (Splitting Singular Values at $(1 + \varepsilon)\sigma_{k+1}$): Let $k' \leq k$ be the index such that $\sigma_{k'} \geq (1 + \varepsilon)\sigma_{k+1}$ and $\sigma_{k'+1} < (1 + \varepsilon)\sigma_{k+1}$. Using (13) on the first k' coordinates and $\sigma_i < (1 + \varepsilon)\sigma_{k+1}$ on the remaining coordinates:

$$\begin{aligned}
\|A - ZZ^T A\|_2^2 &= \|y^T S\|_2^2 = \sum_{i=1}^{k'} y_i^2 \sigma_i^2 + \sum_{i=k'+1}^n y_i^2 \sigma_i^2 \\
&\leq \left(\|G'_2\|_2^2 \|(G'_1)^{-1}\|_2^2 \sum_{i=1}^{k'} \left(\frac{\sigma_{k+1}}{\sigma_i} \right)^{4t} \sigma_{k+1}^2 \right) + (1 + \varepsilon) \sigma_{k+1}^2 \\
&\leq \left[\|G'_2\|_2^2 \|(G'_1)^{-1}\|_2^2 k \left(\frac{1}{1 + \varepsilon} \right)^{4t} + (1 + \varepsilon) \right] \sigma_{k+1}^2 \leq (1 + 2\varepsilon) \sigma_{k+1}^2. \quad (14)
\end{aligned}$$

The last inequality holds as long as $\|G'_2\|_2^2 \|(G'_1)^{-1}\|_2^2 k (1 + \varepsilon)^{-4t} \leq \varepsilon$, which holds for:

$$t \geq \frac{\log(\|G'_2\|_2^2 \|(G'_1)^{-1}\|_2^2 k / \varepsilon)}{4 \log(1 + \varepsilon)} = \mathcal{O}\left(\frac{\log(n/\varepsilon)}{\varepsilon}\right),$$

since G'_1 and G'_2 are i.i.d. Gaussian matrices and therefore satisfy $\|G'_2\|_2^2 \|(G'_1)^{-1}\|_2^2 = \mathcal{O}(\text{poly}(n))$ with high probability [8]. Finally, $\|A - ZZ^T A\|_2 \leq \sqrt{1 + 2\varepsilon} \sigma_{k+1} \leq (1 + \varepsilon) \sigma_{k+1}$. $\square$

Summary & Synthesis

Session & Topic	Core Algorithm / Primitive	Key Guarantee	Complexity
1.b Gaussian JL [3, 1]	$\Pi = \frac{1}{\sqrt{k}}R, R(i, j) \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$	$(1 - \varepsilon)\ x_i - x_j\ _2 \leq \ \Pi x_i - \Pi x_j\ _2 \leq (1 + \varepsilon)\ x_i - x_j\ _2$	$k = \mathcal{O}\left(\frac{\log n}{\varepsilon^2}\right), \mathcal{O}(kd)$ time/vector
2.1 Exact SVD	$A = \sum_{\ell=1}^r \sigma_{\ell} u_{\ell} v_{\ell}^T = U \Sigma V^T$	$\min_{\text{rank}(B) \leq k} \ A - B\ _2 = \ A - A_k\ _2 = \sigma_{k+1}$	—
2.2 Deterministic SVD	Householder Bidiagonalization + QR	Exact (up to ϵ_{mach})	$\Theta(mn^2)$ FLOPs
2.3.1 Power Method	$x_s = (A^T A)^s x_0, x_0 \sim \mathcal{N}(0, I_n)$	Amplifies top singular direction by $(\sigma_1/\sigma_2)^{2s}$	Depends on spectral gap σ_1/σ_2
2.3.2 Block Power Method [2, 4]	$Z = \text{orth}((AA^T)^t AG),$ $G \sim \mathcal{N}(0, 1)^{m \times k}$	$\ A - ZZ^T A\ _2 \leq (1 + \varepsilon)\sigma_{k+1}$ (gap-independent)	$\mathcal{O}\left(\frac{mnk \log(n/\varepsilon)}{\varepsilon}\right)$ FLOPs

References

- [1] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of the Johnson-Lindenstrauss lemma. *International Computer Science Institute, Technical Report*, 99-006, 1999. (Published in *Random Structures & Algorithms*, 22(1):60–65, 2003.)
- [2] Nathan Halko, Per-Gunnar Martinsson, and Joel A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.
- [3] William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26:189–206, 1984.
- [4] Edo Liberty. A short proof of a spectral gap independent property of simultaneous iterations. *Manuscript*, 2016.
- [5] Cameron Musco and Christopher Musco. Randomized block Krylov methods for stronger and faster approximate singular value decomposition. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, 2015.
- [6] Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. TRAK: Attributing model behavior at scale. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*, pages 27074–27113, 2023.
- [7] Vladimir Rokhlin, Arthur Szlam, and Mark Tygert. A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1100–1124, 2009.
- [8] Mark Rudelson and Roman Vershynin. The Littlewood-Offord problem and invertibility of random matrices. *Advances in Mathematics*, 218(2):600–633, 2008.

- [9] Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. GaLore: Memory-efficient LLM training by gradient low-rank projection. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.