Introduction to randomization and sketching techniques

Laura Grigori

EPFL and PSI

October 29, 2024

EPFL

PAUL SCHERRER INSTITUT PSI

Plan

Some background

Random sketching

Randomization for least-squares problem

Plan

Some background

Random sketching

Randomization for least-squares problem

Singular value decomposition

For any given $A \in \mathbb{R}^{m \times n}$, $m \geq n$ its singular value decomposition is

$$A = U\Sigma V^T = \begin{pmatrix} U_1 & U_2 & U_3 \end{pmatrix} \cdot \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \\ 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} V_1 & V_2 \end{pmatrix}^T$$

where for a given k,

- $U \in \mathbb{R}^{m \times m}$ is orthogonal matrix, the left singular vectors of A ,
  U_1 is $m \times k$, U_2 is $m \times n - k$, U_3 is $m \times m - n$
- $\Sigma \in \mathbb{R}^{m \times n}$, its diagonal is formed by $\sigma_1(A) \geq \ldots \geq \sigma_n(A) \geq 0$
  Σ_1 is $k \times k$, Σ_2 is $n - k \times n - k$
- $V \in \mathbb{R}^{n \times n}$ is orthogonal matrix, the right singular vectors of A,
  V_1 is $n \times k$, V_2 is $n \times n - k$

Min-max principle for singular values

Courant-Fischer Min-max Theorem

$$\sigma_i(A) = \min_{\substack{\mathcal{V} \text{ subspace of } \mathbb{R}^n \\ dim(\mathcal{V})=n+1-i}} \max_{\substack{x \in \mathcal{V} \\ \|x\|_2=1}} \|Ax\|_2. \tag{1}$$

Properties of SVD

Given $A = U\Sigma V^T$, we have

- $A^T A = V\Sigma^T \Sigma V^T$,
  the right singular vectors of A are a set of orthonormal eigenvectors of $A^T A$.
- $AA^T = U\Sigma^T \Sigma U^T$,
  the left singular vectors of A are a set of orthonormal eigenvectors of AA^T.
- The non-negative singular values of A are the square roots of the non-negative eigenvalues of $A^T A$ and AA^T.
- If $\sigma_k \neq 0$ and $\sigma_{k+1}, \ldots, \sigma_n = 0$, then
  $Range(A) = span(U_1)$, $Null(A) = span(V_2)$,
  $Range(A^T) = span(V_1)$, $Null(A) = span(U_2\ U_3)$.

Norms and condition number

$$
\begin{aligned}
||A||_2 &= \sigma_{max}(A) = \sigma_1(A) \\
||A||_F &= \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2} = \sqrt{\sigma_1^2(A) + \ldots \sigma_n^2(A)} \\
||A||_* &= \sigma_1(A) + \ldots \sigma_n(A) \\
\kappa(A) &= \frac{\sigma_{max}(A)}{\sigma_{min}(A)} = \sqrt{\|A^T A\|_2 \|(A^T A)^{-1}\|_2}
\end{aligned}
$$

Some properties:

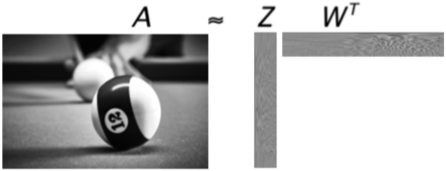
$$
\begin{aligned}
\max_{i,j}|A(i,j)| &\leq ||A||_2 \leq \sqrt{mn}\max_{i,j}|A(i,j)| \\
||A||_2 &\leq ||A||_F \leq \sqrt{min(m,n)}||A||_2
\end{aligned}
$$

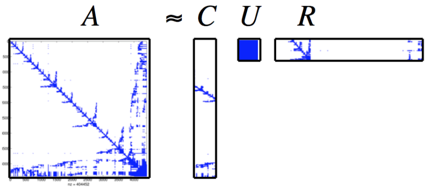
Orthogonal Invariance: If $Q \in \mathbb{R}^{m\times m}$ and $Z \in \mathbb{R}^{n\times n}$ are orthogonal, then

$$||QAZ||_F = ||A||_F$$
$$||QAZ||_2 = ||A||_2$$

Low rank matrix approximation

- Problem: given $A \in \mathbb{R}^{m \times n}$, compute rank-k approximation ZW^T, where Z is $m \times k$ and W^T is $k \times n$.





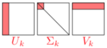
- Problem with diverse applications
  - from scientific computing: fast solvers for integral equations, H-matrices
  - to data analytics: principal component analysis, image processing, ...

$$Ax \to ZW^T x$$

$$\textit{Flops} \quad 2mn \to 2(m+n)k$$

Low rank matrix approximation

- Best rank-k approximation $[\![A]\!]_k = U_k \Sigma_k V_k^T$ is rank-k truncated SVD of A [Eckart and Young, 1936]



$$\min_{rank(A_k) \leq k} ||A - A_k||_2 \quad = \quad ||A - [\![A]\!]_k||_2 = \sigma_{k+1}(A) \tag{2}$$

$$\min_{rank(A_k) \leq k} ||A - A_k||_F \quad = \quad ||A - [\![A]\!]_k||_F = \sqrt{\sum_{j=k+1}^{n} \sigma_j^2(A)} \tag{3}$$

Image, size 1190×1920

Rank-10 approximation, SVD

Rank-50 approximation, SVD

- Image source: `https://pixabay.com/photos/billiards-ball-play-number-half-4345870/`

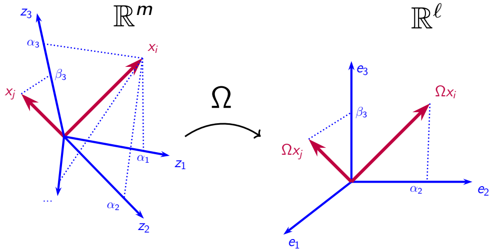
Plan

Some background

Random sketching

Randomization for least-squares problem

Random sketching



Sketching: embedding of a high dimensional subspace into a low dimensional one, while preserving some geometry, with high probability

Applications: least squares problems, low rank matrix approximation, data compression, column subset selection, orthogonalization of set of vectors, Krylov subspace methods, ...

References: [Johnson and Lindenstrauss, 1984, Dasgupta and Gupta, 2003], [Martinsson and Tropp, 2020]

Image courtesy of O. Balabanov

RandBLAS and RandLAPACK

Ongoing effort to define standards similar to BLAS/LAPACK, organized as

- drivers: few and simple
- computational routines: building blocks for the drivers

RandBLAS - data-oblivious sketching routines

- generate a sketching operator
- apply a sketching operator to a matrix

RandLAPACK: linear algebra problems solved through randomization, e.g.

- least squares
- low rank approximation
- linear solvers
- advanced sketching: leverage scores, sketching operators with tensor product structures

RandBLAS and RandLAPACK

Randomized Numerical Linear Algebra: A Perspective on the Field With an Eye to Software, R. Murray et al, describes:

- basic sketching: dense and sparse sketching operators
- least squares and optimization
- low rank approximation
- full rank matrix decompositions
- kernel methods as arising in machine learning models
- linear solvers and trace estimation
- advanced sketching: leverage scores, sketching operators with tensor product structures

References on RandNLA

Many references available, as:

- Sketching As a Tool for Numerical Linear Algebra [Woodruff, 2014]
- Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions [Halko et al., 2011]
- Randomized Numerical Linear Algebra: Foundations and Algorithms [Martinsson and Tropp, 2020]
- Randomized Numerical Linear Algebra: A Perspective on the Field With an Eye to Software [Murray et al., 2023]

ε-subspace embedding property

For a given subspace $\mathcal{V} \subset \mathbb{R}^m$ and $\varepsilon \in (0,1)$, a sketching matrix $\Omega \in \mathbb{R}^{l \times m}$ is an ε-embedding for $\mathcal{V}$ if for all $x_i, x_j \in \mathcal{V}$, we have

$$|\langle \Omega x_i, \Omega x_j \rangle - \langle x_i, x_j \rangle| \leq \epsilon \|x_i\|_2 \|x_j\|_2 \tag{4}$$

- If $x_i = x_j$ we obtain

$$(1-\varepsilon)\|x_i\|_2^2 \leq \|\Omega x_i\|_2^2 \leq (1+\varepsilon)\|x_i\|_2^2. \tag{5}$$

ε-subspace embedding property

For a given subspace $\mathcal{V} \subset \mathbb{R}^m$ and $\varepsilon \in (0,1)$, a sketching matrix $\Omega \in \mathbb{R}^{l \times m}$ is an ε-embedding for $\mathcal{V}$ if for all $x_i, x_j \in \mathcal{V}$, we have

$$|\langle \Omega x_i, \Omega x_j \rangle - \langle x_i, x_j \rangle| \leq \epsilon \|x_i\|_2 \|x_j\|_2 \tag{6}$$

- It can also be expressed as: given all vectors $x_i, x_j \in V$ are rescaled to be unit vectors, then for all $x_i, x_j \in V$ we require to hold:

$$(1-\epsilon)\|x_i + x_j\|_2^2 \leq \|\Omega(x_i + x_j)\|_2^2 \leq (1+\epsilon)\|x_i + x_j\|_2^2 \tag{7}$$

Proof that we obtain relation (6):

$$\begin{aligned}
\langle \Omega x_i, \Omega x_j \rangle &= \left(\|\Omega(x_i + x_j)\|_2^2 - \|\Omega x_i\|_2^2 - \|\Omega x_j\|_2^2\right)/2 \\
&= \left((1 \pm \epsilon)\|x_i + x_j\|_2^2 - (1 \pm \epsilon)\|x_i\|_2^2 - (1 \pm \epsilon)\|x_j\|_2^2\right)/2 \\
&= \langle x_i, x_j \rangle \pm O(\epsilon)
\end{aligned}$$

ε-subspace embedding property

Let A be a matrix whose columns form a basis for $\mathcal{V}$. For simplicity, we refer to an ε-subspace embedding for $\mathcal{V}$ as an ε-embedding for A.

Corollary 1

If $\Omega \in \mathbb{R}^{l \times m}$ is an ε-embedding for A, then the singular values of A are bounded by

$$(1+\varepsilon)^{-1/2}\sigma_{min}(\Omega A) \ \leq \sigma_{min}(A) \leq \sigma_{max}(A) \leq (1-\varepsilon)^{-1/2}\sigma_{max}(\Omega A).$$

Proof.

By min-max principle we have: $\sigma_i(A) = \min\limits_{\substack{\mathcal{V} \text{ subspace of } \mathbb{R}^n \\ dim(\mathcal{V})=n+1-i}} \max\limits_{\substack{x \in \mathcal{V} \\ \|x\|_2=1}} \|Ax\|_2$. *We obtain:*

$$\sigma_i(\Omega A) = \min_{\substack{\mathcal{V} \text{ subspace of } \mathbb{R}^n \\ dim(\mathcal{V})=n+1-i}} \max_{\substack{x \in \mathcal{V} \\ \|x\|_2=1}} \|\Omega Ax\|_2 \leq \min_{\substack{\mathcal{V} \text{ subspace of } \mathbb{R}^n \\ dim(\mathcal{V})=n+1-i}} \max_{\substack{x \in \mathcal{V} \\ \|x\|_2=1}} \sqrt{1+\varepsilon}\, \|Ax\|_2 = \sqrt{1+\varepsilon}\, \sigma_i(A). \tag{8}$$

Proceed similarly for the other bound. □

Oblivious subspace embedding

Aim: construct Ω such that for any n-dimensional subspace $\mathcal{V} \subset \mathbb{R}^m$

$$\mathbb{P}(\Omega \text{ is } \varepsilon\text{-embedding for } \mathcal{V}) \geq 1 - \delta$$

Definition: oblivious subspace embedding

A random matrix $\Omega \in \mathbb{R}^{l \times m}$ is an oblivious subspace embedding with parameters OSE(n, ϵ, δ) if with probability at least $1 - \delta$ for any n-dimensional subspace $\mathcal{V} \subset \mathbb{R}^m$, for all $x_i, x_j \in \mathcal{V}$, we have

$$|\langle \Omega x_i, \Omega x_j \rangle - \langle x_i, x_j \rangle| \leq \epsilon \|x_i\|_2 \|x_j\|_2 \tag{9}$$

Random sketching matrices

- $\Omega \in \mathbb{R}^{l \times m}$ whose entries are independent standard normal random variables, multiplied by $1/\sqrt{l}$
  - Ω is OSE(n, ϵ, δ) with $l = \mathcal{O}(\epsilon^{-2}(n + \log \frac{1}{\delta}))$
  - Cost of computing ΩA, $A \in \mathbb{R}^{m \times n}$: $2mnl$ flops
  - Relies on BLAS3 operations when A is dense
- Easy to parallelize, $\Omega A = \sum_{i=1}^{P} \Omega_i A_i$

$$\Omega A = \begin{pmatrix} \Omega_1 & \dots & \Omega_P \end{pmatrix} \begin{pmatrix} A_1 \\ \vdots \\ A_P \end{pmatrix} = \sum_{i=1}^{P} \Omega_i A_i$$

  - Each processor i owns a block $\Omega_i \in \mathbb{R}^{l \times m/P}$ and a block $A_i \in \mathbb{R}^{m/P \times n}$
  - Each processor computes $\Omega_i A_i \in \mathbb{R}^{l \times n}$
  - Sum-Reduce among all processors to compute $\Omega A = \sum_{i=1}^{P} \Omega_i A_i$
  - Cost of the algorithm

$$(2mnl/P)\gamma + \log_2 P\alpha + ln \log_2 P\beta$$

Fast Johnson-Lindenstrauss transform

Find sparse or structured Ω such that computing ΩA is cheap, e.g. a subsampled random Hadamard transform (SRHT).
Given $m = 2^q, l < m$, the SRHT ensemble embedding $\mathbb{R}^m$ into $\mathbb{R}^l$ is defined as

$$\Omega = \sqrt{\frac{m}{l}} \cdot P \cdot H \cdot D, \text{ where} \tag{10}$$

- $D \in \mathbb{R}^{m \times m}$ is diagonal matrix of uniformly random signs, random variables uniformly distributed on ± 1
- $H \in \mathbb{R}^{m \times m}$ is the normalized Walsh-Hadamard transform
- $P \in \mathbb{R}^{l \times m}$ formed by subset of l rows of the identity, chosen uniformly at random (draws l rows at random from HD).

References: Sarlos'06, Ailon and Chazelle'06, Liberty, Rokhlin, Tygert and Woolfe'06.

Fast Johnson-Lindenstrauss transform (contd)

Definition of Normalized Walsh–Hadamard Matrix
For given $m = 2^q$, $H_m \in \mathbb{R}^{m \times m}$ is the non-normalized Walsh-Hadamard transform defined recursively as,

$$H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad H_m = \begin{pmatrix} H_{m/2} & H_{m/2} \\ H_{m/2} & -H_{m/2} \end{pmatrix}. \tag{11}$$

The normalized Walsh-Hadamard transform is $H = m^{-1/2} H_m$.

Cost of matrix vector multiplication:
For $w \in \mathbb{R}^m$ and $\Omega \in \mathbb{R}^{l \times m}$, computing Ωw costs $2m \log_2 m$ flops.

Random sketching matrices - SRHT

- $\Omega \in \mathbb{R}^{l \times m}$ is a fast Johnson-Lindenstrauss transform, e.g. a subsampled randomized Hadamard transform (SRHT)[1]

$$\Omega = \sqrt{\frac{m}{l}} \cdot R \cdot H \cdot D, \text{ where} \tag{12}$$

$D \in \mathbb{R}^{m \times m}$ is diagonal with independent random signs, $H \in \mathbb{R}^{m \times m}$ is normalized Walsh-Hadamard matrix, $R \in \mathbb{R}^{l \times m}$ draws l rows uniformly at random from HD.

  - Ω is OSE(n, ϵ, δ) with $l = \mathcal{O}(\epsilon^{-2} \left(n + \ln \frac{m}{\delta}\right) \ln \frac{n}{\delta})$
  - Cost of computing ΩA, $A \in \mathbb{R}^{m \times n}$ on P processors:

$$\frac{2mn \log_2 m}{P} \gamma + \log_2 P \alpha + \frac{mn}{P} \log_2 P \beta$$

1. Ailon and Chazelle'06, Liberty, Rokhlin, Tygert and Woolfe'06, Sarlos'06.

Block SRHT for parallelization on P processors

- Ω as in (13) is OSE(n, ϵ, δ) with $l = \mathcal{O}(\epsilon^{-2}\left(n + \ln \frac{m}{\delta}\right) \ln \frac{n}{\delta})$

$$\Omega = [\Omega_1 \quad \Omega_2 \quad \ldots \quad \Omega_P] = \sqrt{\frac{m}{Pl}} \cdot [D_{L1} \quad \ldots \quad D_{LP}] \begin{bmatrix} RH & & \\ & \ddots & \\ & & RH \end{bmatrix} \begin{bmatrix} D_{R1} & & \\ & \ddots & \\ & & D_{RP} \end{bmatrix}, \tag{13}$$

where $\Omega_i = \sqrt{\frac{m}{Pl}} D_{Li} RHD_{Ri}$, $D_{Li} \in \mathbb{R}^{l \times l}$, $D_{Ri} \in \mathbb{R}^{m/P \times m/P}$ are diagonal with independent random signs, $H \in \mathbb{R}^{m/P \times m/P}$ is normalized Walsh-Hadamard matrix, $R \in \mathbb{R}^{l \times m/P}$ is uniform sampling matrix.

- Parallelize as [Balabanov et al., 2022]:

$$\Omega A = \sqrt{\frac{m}{Pl}} \sum_{i=1}^{P} D_{Li} RHD_{Ri} A_i$$

Parallelization of block SRHT

Considering each processor i owns a block $A_i \in \mathbb{R}^{m/P \times n}$, parallelize as:

$$
\begin{aligned}
\Omega A &= \begin{pmatrix} \Omega_1 & \dots & \Omega_P \end{pmatrix} \begin{pmatrix} A_1 \\ \vdots \\ A_P \end{pmatrix} = \begin{pmatrix} \sqrt{\frac{m}{Pl}} D_{L1} RHD_{R1} & \dots & \sqrt{\frac{m}{Pl}} D_{LP} RHD_{RP} A_P \end{pmatrix} \begin{pmatrix} A_1 \\ \vdots \\ A_P \end{pmatrix} \\
&= \sum_{i=1}^{P} \sqrt{\frac{m}{Pl}} D_{Li} RHD_{Ri} A_i
\end{aligned}
$$

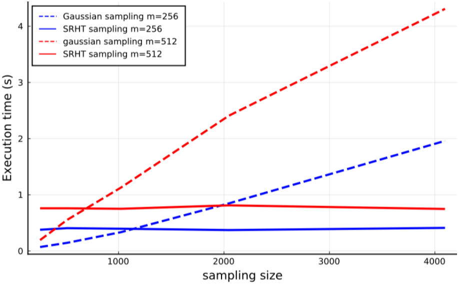
- Root processor broadcasts seed of R
- Each processor i draws R, D_{Li}, D_{Ri}
- Each processor computes $\Omega_i A_i = \sqrt{\frac{m}{Pl}} D_{Li} RHD_{Ri} A_i$, $\Omega_i A_i \in \mathbb{R}^{l \times n}$
- Sum-Reduce among all processors to compute $\Omega A = \sum_{i=1}^{P} \Omega_i A_i$

Cost of the algorithm (some lower order terms ignored)

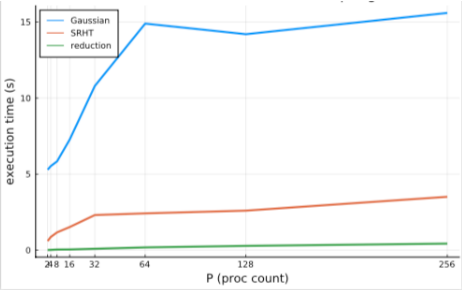
$$2mn \log_2 m/P\gamma + \log_2 P\alpha + ln \log_2 P\beta$$

Performance of Gaussian vs block SRHT

$A \in \mathbb{R}^{n \times m}, n = 64000, P = 1$



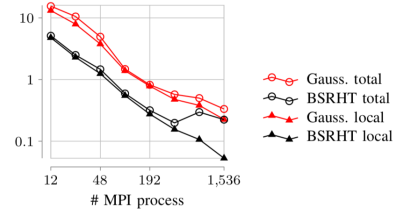
$A \in \mathbb{R}^{nP \times m}, n = 65536, m = 256, k = 4096$



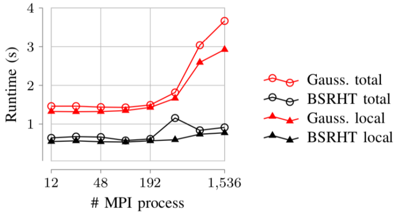
- Results obtained in Julia on nodes formed by 2 Cascade Lake Intel Xeon 5218, 16 cores each, 2.4GHz/core

Performance of Gaussian vs SRHT

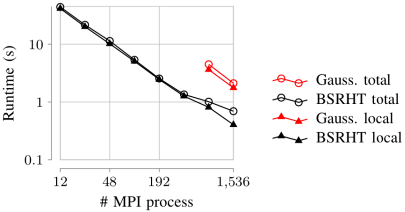
$A \in \mathbb{R}^{m \times n}, m = 10^7, n = 200, l = 2000$



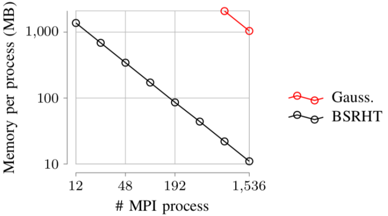
$A \in \mathbb{R}^{m \times n}, m = 10^5 \times P, n = 200, l = 2000$



$A \in \mathbb{R}^{m \times n}, m = 10^8, n = 200, l = 2000$



Memory per processor



Machine: Intel Skylake 2.7GHz (AVX512), 48 cores per node

Plan

Some background

Random sketching

Randomization for least-squares problem

Solving least squares problems

Given $A \in \mathbb{R}^{m \times n}$ full-rank and $b \in \mathbb{R}^m$, with $n \ll m$, solve

$$y := arg \min_{x \in \mathbb{R}^n} \|Ax - b\|_2$$

The unique solution is

$$y = A^+ b, \qquad A^+ = (A^T A)^{-1} A^T$$

Solve by using the QR factorization of A (see lecture on dense QR)

Least squares problems

Solve by using the normal equations,

$$A^T A x = A^T b$$

1. with direct methods
   multiply $A^T A$ and compute the Cholesky factorization of the result
2. or with iterative methods without computing explicitely $A^T A$
   use a Krylov subspace solver and at each iteration multiply $A^T A$ with a vector

Randomized least squares - sketch and solve

Solve by using randomization, with $\Omega \in \mathbb{R}^{l \times m}$ being OSE$(n+1, \epsilon, \delta)$ for $\mathcal{V} = range([A, b])$

$$y_s := arg \min_{x \in \mathbb{R}^n} \|\Omega(Ax - b)\|_2$$

or $y_s = (\Omega A)^{\dagger}(\Omega b)$
We obtain with probability $1 - \delta$:

$$\frac{1}{\sqrt{1+\varepsilon}}\|\Omega(Ay_s - b)\|_2 \leq \|Ay - b\|_2 \leq \frac{1}{\sqrt{1-\varepsilon}}\|\Omega(Ay_s - b)\|_2$$

Randomized least squares - sketch and solve

$$\frac{1}{\sqrt{1+\varepsilon}}\|\Omega(Ay_s - b)\|_2 \le \|Ay - b\|_2 \le \frac{1}{\sqrt{1-\varepsilon}}\|\Omega(Ay_s - b)\|_2$$

Proof

Since y is the minimizer of the original least squares problem, using (5), we obtain

$$\|Ay - b\|_2 = \min_{x\in\mathbb{R}^n} \|Ax - b\|_2 \le \|Ay_s - b\|_2 \le \frac{1}{\sqrt{1-\varepsilon}}\|\Omega(Ay_s - b)\|_2 \quad (14)$$

Similarly, since y_s is the minimizer of the sketched least square problem, we have

$$\|\Omega(Ay_s - b)\|_2 = \min_{x\in\mathbb{R}^n} \|\Omega(Ax - b)\|_2 \le \|\Omega(Ay - b)\|_2 \le \sqrt{1+\varepsilon}\|Ay - b\|_2 \quad (15)$$

References (1)

Balabanov, O., Beaupere, M., Grigori, L., and Lederer, V. (2022).
Block subsampled randomized hadamard transform for low-rank approximation on distributed architectures.

Dasgupta, S. and Gupta, A. (2003).
An elementary proof of a theorem of johnson and lindenstrauss.
Random Structures & Algorithms, 22(1):60–65.

Eckart, C. and Young, G. (1936).
The approximation of one matrix by another of lower rank.
Psychometrika, 1:211–218.

Halko, N., Martinsson, P. G., and Tropp, J. A. (2011).
Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions.
SIAM Rev., 53(2):217–288.

Johnson, W. and Lindenstrauss, J. (1984).
Extensions of Lipschitz mappings into a hilbert space.
Contemp. Math., 26:189–206.

Martinsson, P.-G. and Tropp, J. (2020).
Randomized numerical linear algebra: Foundations and algorithms.

Murray, R., Demmel, J., Mahoney, M. W., Erichson, N. B., Melnichenko, M., Malik, O. A., Grigori, L., Luszczek, P., Dereziński, M., Lopes, M. E., Liang, T., Luo, H., and Dongarra, J. (2023).
Randomized numerical linear algebra : A perspective on the field with an eye to software.

Woodruff, D. P. (2014).
Sketching as a tool for numerical linear algebra.
Found. Trends Theor. Comput. Sci., 10(1–2):1–157.