POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Randomized Linear Algebra and its Applications

High Performance Computing Engineering - Advanced Methods for Scientific Computing

Jasmin Spinetto, Peng Rao, Anna Paola Izzo, Cao Wu, Jiali Claudo Haung,
10734813, 11022931, ..., ..., ...

Advisor:
Prof. Luca Formaggia

Academic year:
2024-2025

Abstract: Here goes the Abstract in English of your thesis (in article format) followed by a list of keywords. The Abstract is a concise summary of the content of the thesis (single page of text) and a guide to the most important contributions included in your thesis. The Abstract is the very last thing you write. It should be a self-contained text and should be clear to someone who hasn't (yet) read the whole manuscript. The Abstract should contain the answers to the main research questions that have been addressed in your thesis. It needs to summarize the motivations and the adopted approach as well as the findings of your work and their relevance and impact. The Abstract is the part appearing in the record of your thesis inside POLITesi, the Digital Archive of PhD and Master Theses (Laurea Magistrale) of Politecnico di Milano. The Abstract will be followed by a list of four to six keywords. Keywords are a tool to help indexers and search engines to find relevant documents. To be relevant and effective, keywords must be chosen carefully. They should represent the content of your work and be specific to your field or sub-field. Keywords may be a single word or two to four words.

Key-words: Randomized Linear Algebra, RandomizedSVD, CUR Decomposition

1. Introduction

This project is a C++ implementation of the Randomized Singular Value Decomposition (rSVD) algorithm. We only used the matrix operations of the `Eigen` library to implement our algorithm. We do some benchmarks to compare the performance of our implementation with the Eigen library, the result indicates that our implementation can enhance the performance of handling large or sparse matrices.
We have already implemented the following algorithms:

- QR factorization using Given's rotation: `GivensRotation`
- Basic SVD using Power Method: `PowerMethod`
- Randomized singular value decomposition: `RandomizedSVD`

Figure 1 shows the speedup of our implementation compared to `BDCSVD` the `Eigen` library. The speedup is calculated as the ratio of the time taken by the Eigen library to the time taken by our implementation.

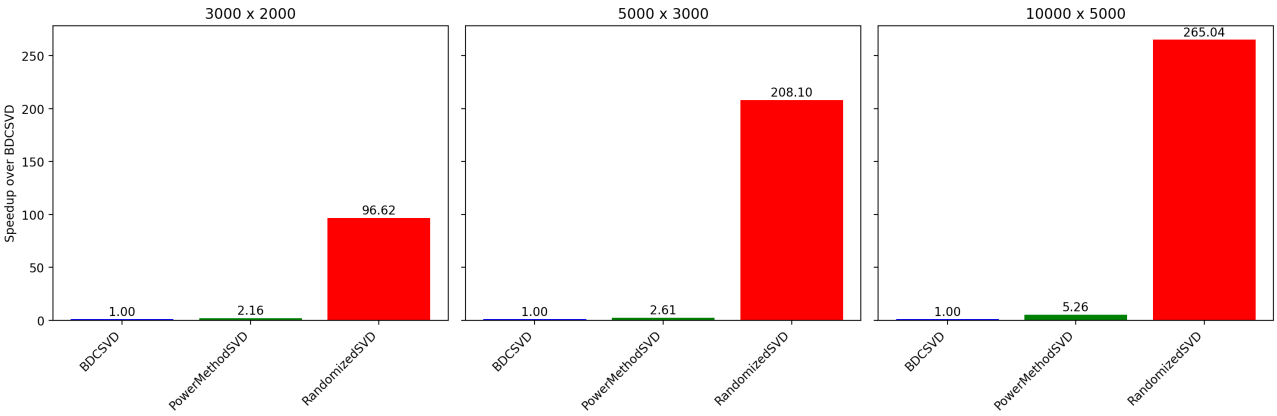

Figure 1: Randomized SVD

2. Randomized Linear Algebra

Random Linear Algebra is a branch of mathematics that combines principles from linear algebra with techniques from probability and statistics to analyze and solve problems involving large-scale or high-dimensional data. The base idea is to introduce **randomness** into computations to make algorithms faster, simpler, more efficient, more scalable, or more robust while maintaining accurate approximations of the desired results. This is needed because traditional linear algebra techniques can become computationally expensive for large datasets, such as those encountered in machine learning, data science, and numerical simulations. Randomized algorithms help mitigate these challenges by:

- Reducing computation complexity
- Lowering memory requirements
- Allowing for parallelization and distributed computing

The main techniques used are:

- **Randomized Sampling**: instead of processing an entire matrix, Randomized Linear Algebra uses random sampling to approximate matrix operations. For example by:
  - *Row/column sampling*: selecting a subset of rows or columns from a matrix based on a probability distribution to approximate its structure.
  - *Sketching*: compressing a large matrix into a smaller one ("sketch matrix") while preserving key properties, such as norms or singular values.
- **Low-Rank Approximation**: many large matrices encountered in practice are approximately low-rank, meaning their significant information can be captured by a smaller number of dimensions. Randomized techniques are used to compute these approximations efficiently:
  - *Randomized SVD (Singular Value Decomposition)*: Approximating the singular values and vectors using random projections.
  - *CUR Decomposition*: representing a matrix using a subset of its actual rows (C) and columns (R), and a smaller core matrix (U).
- **Random Projections**: high-dimensional data can be projected into a lower-dimensional space using random matrices, such as Gaussian random matrices and Sparse random matrices.
- **Montecarlo methods**: randomized algorithms often rely on Monte Carlo methods to provide probabilistic guarantees about the accuracy of approximations. For example, the probability of achieving a given approximation error is often a parameter of the algorithm.
- **Stochastic Iterative Methods**: iterative solvers for linear systems, such as gradient descent, can be randomized by incorporating stochastic components, which often improves convergence rates or reduces computational cost in large-scale problems.

2.1. Challenges

The main challenges that arise when using Randomized Linear Algebra are

3. Randomized Singular Value Decomposition

3.1. QR Factorization

The Givens Rotation QR decomposition is a method for decomposing a matrix A into an orthogonal matrix Q and an upper triangular matrix R, such that:

$$A = QR$$

A Givens rotation matrix is used to zero out specific elements of a matrix. For two elements a and b, the Givens rotation coefficients c and s are calculated such that:

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}$$

where $r = \sqrt{a^2 + b^2}$. The Givens rotation matrix is then used to zero out the element b by multiplying the matrix from the left:

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} r & 0 \\ 0 & d' \end{bmatrix}$$

where $d' = cd - sb$. The Givens rotation matrix is a simple and efficient way to perform QR decomposition, especially for sparse matrices. The pseudocode for the Givens rotation QR decomposition is shown in Algorithm **??**.

3.2. Singular Value Decompoition using Power Method

The singular value decomposition (SVD) is a fundamental matrix decomposition method that decomposes a matrix A into three matrices U, Σ, and V such that:

$$A = U\Sigma V^T$$

where U and V are orthogonal matrices and Σ is a diagonal matrix with the singular values of A. The SVD is widely used in various applications, including dimensionality reduction, data compression, and machine learning. The power method is an iterative algorithm that can be used to compute the singular values and vectors of a matrix. The power method works by repeatedly multiplying the matrix by a vector and normalizing the result. The pseudocode for the power method SVD is shown in Algorithm **??**.

3.3. Algorithm Description

Let $A \in \mathbb{R}^{m \times n}$ be a matrix of low rank, and $m \geq n$. In the following, we seek the near-optimal low-rank approximation of the form

$$A \approx U_k \Sigma_k V_k^T$$

where k denotes the target rank. Instead of computing the singular value decomposition directly, we embed the SVD into the probabilistic framework. The principal concept is sketched in Figure 2.

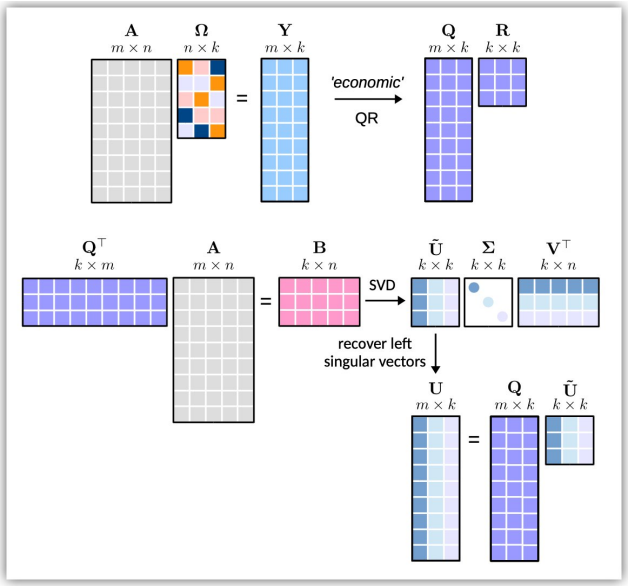

Figure 2: Randomized SVD [1]

3.4. Optimization technique

4. Applications

In this section, we present some useful applications of the RandomizedSVD algorithm.

4.1. Image Compression

4.1.1 Code Explanation

4.1.2 Results Analysis

4.2. Handwritten Numbers Recognition

We implemented a handwritten digit recognition system using the MNIST dataset and our RandomizedSVD algorithm for dimensionality reduction. The system demonstrates how dimensionality reduction can be effectively used for classification tasks while maintaining high accuracy.

4.2.1 Implementation Details

The implementation consists of several key components:

- **Data Loading**: The `MNISTLoader` class handles loading and preprocessing of the MNIST dataset, converting the binary data into Eigen matrices and normalizing pixel values to [0,1].
- **Dimensionality Reduction**: We use PCA with RandomizedSVD to reduce the dimension of the input images (784 dimensions) to a lower-dimensional space (2-256 components in our tests).
- **Classification**: The system uses a simple nearest neighbor classifier in the reduced space to predict digits.

The core algorithm workflow is:

1. Center the data by subtracting the mean
2. Project the data to a lower-dimensional space using PCA
3. For prediction, project new images into the same space and find the nearest neighbor

4.2.2 Results Analysis

We conducted extensive testing using 10,000 training images and 1,000 test images from the MNIST dataset. The results for different numbers of principal components are shown in Table **??**.

Number of Components	Accuracy
2	37.9%
4	58.1%
8	85.1%
16	92.3%
32	93.7%
64	93.3%
128	93.1%
256	92.7%

Table 1: Recognition accuracy with different numbers of principal components

Key observations from the results:

- **Rapid Improvement**: The accuracy improves dramatically from 37.9% with 2 components to 85.1% with just 8 components, showing that even a small number of carefully chosen components can capture significant discriminative information.
- **Optimal Range**: The accuracy peaks at 93.7% with 32 components, suggesting this is the optimal trade-off between dimensionality reduction and information preservation for this task.

- **Diminishing Returns**: Beyond 32 components, the accuracy slightly decreases, indicating that additional components may be introducing noise rather than useful information.

We also tested the system on individual images, successfully recognizing test digits (e.g., correctly identifying a test image of the digit "6"). This demonstrates the practical applicability of our implementation for real-world digit recognition tasks.
The results show that our RandomizedSVD-based approach can achieve high accuracy while significantly reducing the dimensionality of the input data (from 784 to just 32 dimensions in the optimal case). This reduction leads to faster computation and lower memory requirements while maintaining excellent recognition performance.

4.3. Principal Component Analysis (PCA)

4.3.1 Code Explanation

4.3.2 Results Analysis

5. CUR Decomposition

5.1. Algorithm

5.1.1 Code Explanation

5.2. Results Analysis

6. Conclusion

6.1. Possible Further Applications

Present some possible further applications or possible further development of the ones presented.

7. Bibliography and citations

8. Appendix - Group Work Organization

Here explain the contributions from each member of the group.
FROM HERE ON IT'S JUST CODE EXAMPLES

9. Equations

This section gives some examples of writing mathematical equations in your thesis.
Maxwell's equations read:

$$\begin{cases} \nabla \cdot \boldsymbol{D} = \rho, & \text{(1a)} \\ \nabla \times \boldsymbol{E} + \dfrac{\partial \boldsymbol{B}}{\partial t} = \boldsymbol{0}, & \text{(1b)} \\ \nabla \cdot \boldsymbol{B} = 0, & \text{(1c)} \\ \nabla \times \boldsymbol{H} - \dfrac{\partial \boldsymbol{D}}{\partial t} = \boldsymbol{J}. & \text{(1d)} \end{cases}$$

Equation (1) is automatically labeled by `cleveref`, as well as Equation (1a) and Equation (1c). Thanks to the `cleveref` package, there is no need to use `\eqref`. Equations have to be numbered only if they are referenced in the text.

Equations (2), (3), (4), and (5) show again Maxwell's equations without brace:

$$\nabla \cdot \boldsymbol{D} = \rho, \tag{2}$$

$$\nabla \times \boldsymbol{E} + \frac{\partial \boldsymbol{B}}{\partial t} = \boldsymbol{0}, \tag{3}$$

$$\nabla \cdot \boldsymbol{B} = 0, \tag{4}$$

$$\nabla \times \boldsymbol{H} - \frac{\partial \boldsymbol{D}}{\partial t} = \boldsymbol{J}. \tag{5}$$

Equation (6) is the same as before, but with just one label:

$$\begin{cases} \nabla \cdot \boldsymbol{D} = \rho, \\ \nabla \times \boldsymbol{E} + \dfrac{\partial \boldsymbol{B}}{\partial t} = \boldsymbol{0}, \\ \nabla \cdot \boldsymbol{B} = 0, \\ \nabla \times \boldsymbol{H} - \dfrac{\partial \boldsymbol{D}}{\partial t} = \boldsymbol{J}. \end{cases} \tag{6}$$

10. Figures, Tables and Algorithms

Figures, Tables and Algorithms have to contain a Caption that describes their content, and have to be properly referred in the text.

10.1. Figures

For including pictures in your text you can use `TikZ` for high-quality hand-made figures [**?**], or just include them with the command

`\includegraphics[options]{filename.xxx}`

Here xxx is the correct format, e.g. `.png`, `.jpg`, `.eps`, ...

Figure 3: Caption of the Figure.

Thanks to the `\subfloat` command, a single figure, such as Figure 3, can contain multiple sub-figures with their own caption and label, e.g. Figure 4a and Figure 4b.

(a) One PoliMi logo.

(b) Another one PoliMi logo.

Figure 4: Caption of the Figure.

10.2. Tables

Within the environments `table` and `tabular` you can create very fancy tables as the one shown in Table 3.

Example of Table (optional)

	column1	column2	column3
row1	1	2	3
row2	α	β	γ
row3	alpha	beta	gamma

Table 2: Caption of the Table.

You can also consider to highlight selected columns or rows in order to make tables more readable. Moreover, with the use of `table*` and the option `bp` it is possible to align them at the bottom of the page. One example is presented in Table 4.

10.3. Algorithms

Pseudo-algorithms can be written in LaTeX with the `algorithm` and `algorithmic` packages. An example is shown in Algorithm 1.

Algorithm 1 Name of the Algorithm

```
Initial instructions
for for − condition do
    Some instructions
    if if − condition then
        Some other instructions
    end if
end for
while while − condition do
    Some further instructions
end while
Final instructions
```

11. Some further useful suggestions

Theorems have to be formatted as follows:

Theorem 11.1. *Write here your theorem.*

Proof. If useful you can report here the proof.

Propositions have to be formatted as follows:

	column1	column2	column3	column4	column5	column6
row1	1	2	3	4	5	6
row2	a	b	c	d	e	f
row3	α	β	γ	δ	ϕ	ω
row4	alpha	beta	gamma	delta	phi	omega

Table 3: Highlighting the columns

Proposition 11.1. *Write here your proposition.*

How to insert itemized lists:

- first item;
- second item.

How to write numbered lists:

1. first item;
2. second item.

12. Use of copyrighted material

Each student is responsible for obtaining copyright permissions, if necessary, to include published material in the thesis. This applies typically to third-party material published by someone else.

13. Plagiarism

You have to be sure to respect the rules on Copyright and avoid an involuntary plagiarism. It is allowed to take other persons' ideas only if the author and his original work are clearly mentioned. As stated in the Code of Ethics and Conduct, Politecnico di Milano *promotes the integrity of research, condemns manipulation and the infringement of intellectual property*, and gives opportunity to all those who carry out research activities to have an adequate training on ethical conduct and integrity while doing research. To be sure to respect the copyright rules, read the guides on Copyright legislation and citation styles available at:

`https://www.biblio.polimi.it/en/tools/courses-and-tutorials`

You can also attend the courses which are periodically organized on "Bibliographic citations and bibliography management".

14. Conclusions

A final section containing the main conclusions of your research/study and possible future developments of your work have to be inserted in the section "Conclusions".

15. Bibliography and citations

Your thesis must contain a suitable Bibliography which lists all the sources consulted on developing the work. The list of references is placed at the end of the manuscript after the chapter containing the conclusions. It is suggested to use the BibTeX package and save the bibliographic references in the file `bibliography.bib`. This is indeed a database containing all the information about the references. To cite in your manuscript, use the `\cite{}` command as follows:
*Here is how you cite bibliography entries: [**?**], or multiple ones at once: [**?** **?**].*
The bibliography and list of references are generated automatically by running BibTeX [**?**].

References

[1] N. Benjamin Erichson, Sergey Voronin, Steven L. Brunton, and J. Nathan Kutz. Randomized Matrix Decompositions using R. 89(11).

A. Appendix A

If you need to include an appendix to support the research in your thesis, you can place it at the end of the manuscript. An appendix contains supplementary material (figures, tables, data, codes, mathematical proofs, surveys, ...) which supplement the main results contained in the previous sections.

B. Appendix B

It may be necessary to include another appendix to better organize the presentation of supplementary material.

Abstract in lingua italiana

Qui va l'Abstract in lingua italiana della tesi seguito dalla lista di parole chiave.

Parole chiave: qui, le parole chiave, della tesi, in italiano

Acknowledgements

Here you might want to acknowledge someone.