AKADEMIA GÓRNICZO-HUTNICZA

Project Documentation

SVD Algorithm in Python

course

Programming in Python Language

Electronics and Telecommunications, 4th year

Michał Ciągała

Group 4, 5:30 PM Friday

Lecturer: PhD Mustafa Sakhai

12.12.2025

Table of contents

Introduction

Singular Value Decomposition (SVD) is one of the most important matrix factorization techniques used in numerical linear algebra, data analysis, and machine learning. It provides a way to represent a matrix as a product of three simpler matrices, revealing its intrinsic structure and the most significant patterns present in the data.

The main goal of this project is to implement the Singular Value Decomposition algorithm from scratch, without relying on built-in SVD implementations. By doing so, the project focuses on understanding the internal mechanics of the algorithm rather than treating SVD as a black-box tool.

To evaluate the correctness of the implementation, synthetic low-rank datasets are generated. These datasets have a known underlying structure, which allows direct verification of whether the algorithm successfully recovers the essential information from noisy data. The results obtained using the custom SVD implementation are compared with those produced by the scikit-learn library.

In addition to the algorithm implementation, the project includes unit tests, numerical experiments, and visualizations that demonstrate the behavior of truncated SVD. The comparison with scikit-learn confirms that the proposed implementation produces results consistent with a well-established reference method.

Problem statement

The problem addressed in this project is the approximation of a noisy matrix that has an underlying low-rank structure. The goal is to recover the essential information contained in the data using Singular Value Decomposition and to verify whether a custom implementation produces results comparable to a reference implementation from scikit-learn.

Dataset Description

The experiments are conducted on synthetic datasets generated specifically for this project. A clean low-rank matrix is created using randomly generated factors, and Gaussian noise is added to simulate real-world data imperfections.

Since the true rank of the matrix is known, this dataset allows direct evaluation of how well the SVD algorithm recovers the underlying structure.

Algorithm Description

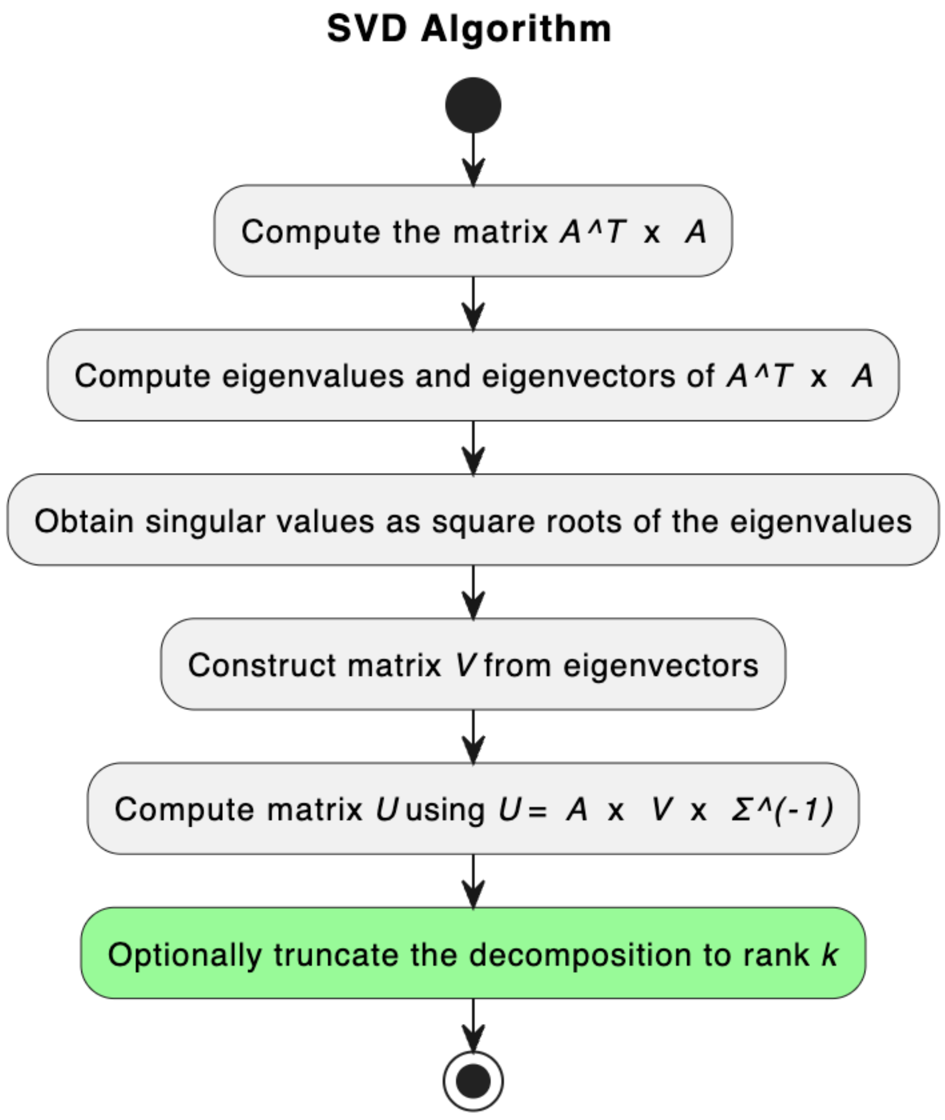
Mathematical Background

Singular Value Decomposition factorizes a matrix $\boldsymbol{A}$into $\boldsymbol{A} = \boldsymbol{U\Sigma V^T}$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal matrices and $\boldsymbol{\Sigma}$ contains the singular values.

Algorithm Steps

The algorithm proceeds as follows:

1. Compute the matrix A^T x A.
2. Compute eigenvalues and eigenvectors of A^T x A.
3. Obtain singular values as square roots of the eigenvalues.
4. Construct matrix *V* from eigenvectors.
5. Compute matrix *U* using $U = A$ x V x Σ^{-1}.
6. Optionally truncate the decomposition to rank *k*.

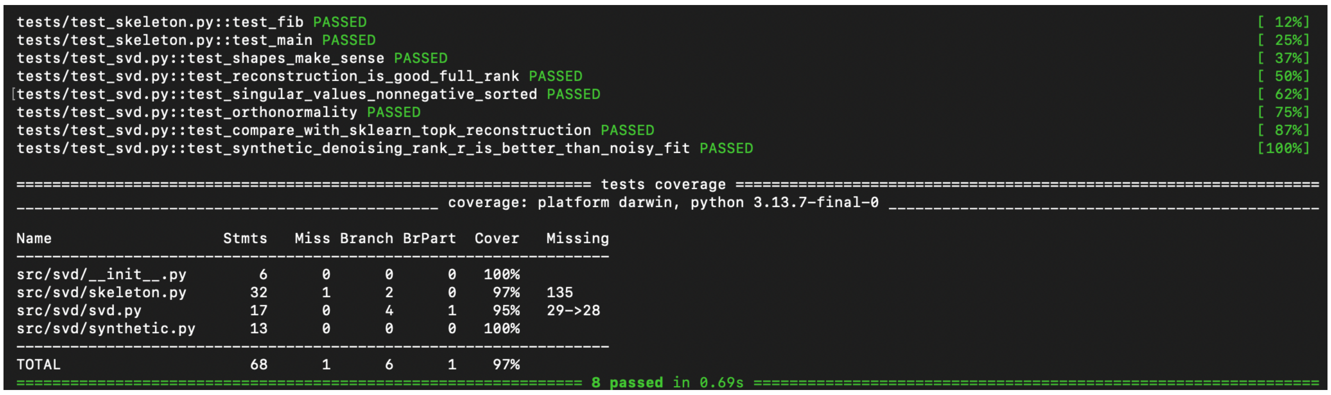


SVD Algorithm
Compute the matrix A^T x A
Compute eigenvalues and eigenvectors of A^T x A
Obtain singular values as square roots of the eigenvalues
Construct matrix V from eigenvectors
Compute matrix U using U = A x V x Σ^(-1)
Optionally truncate the decomposition to rank k

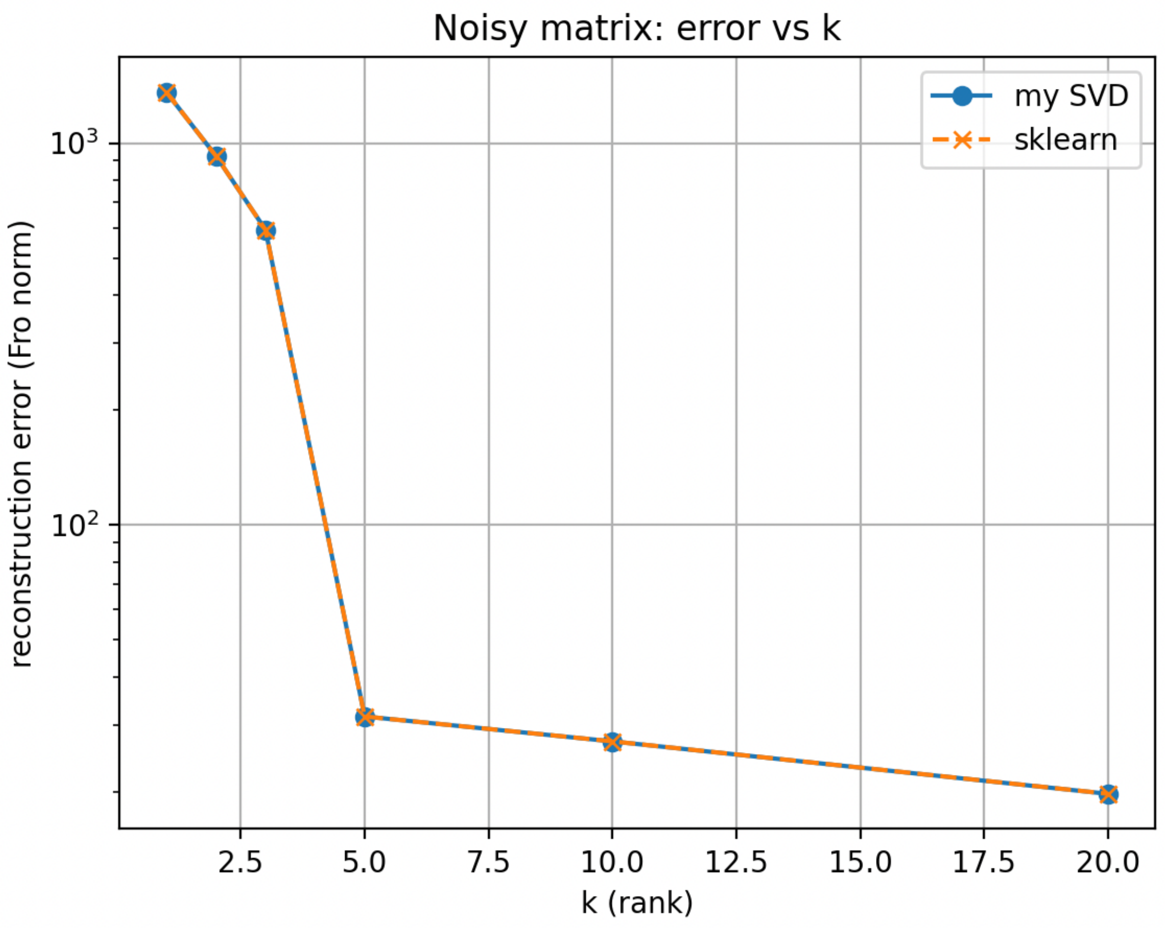
Experimental Setup

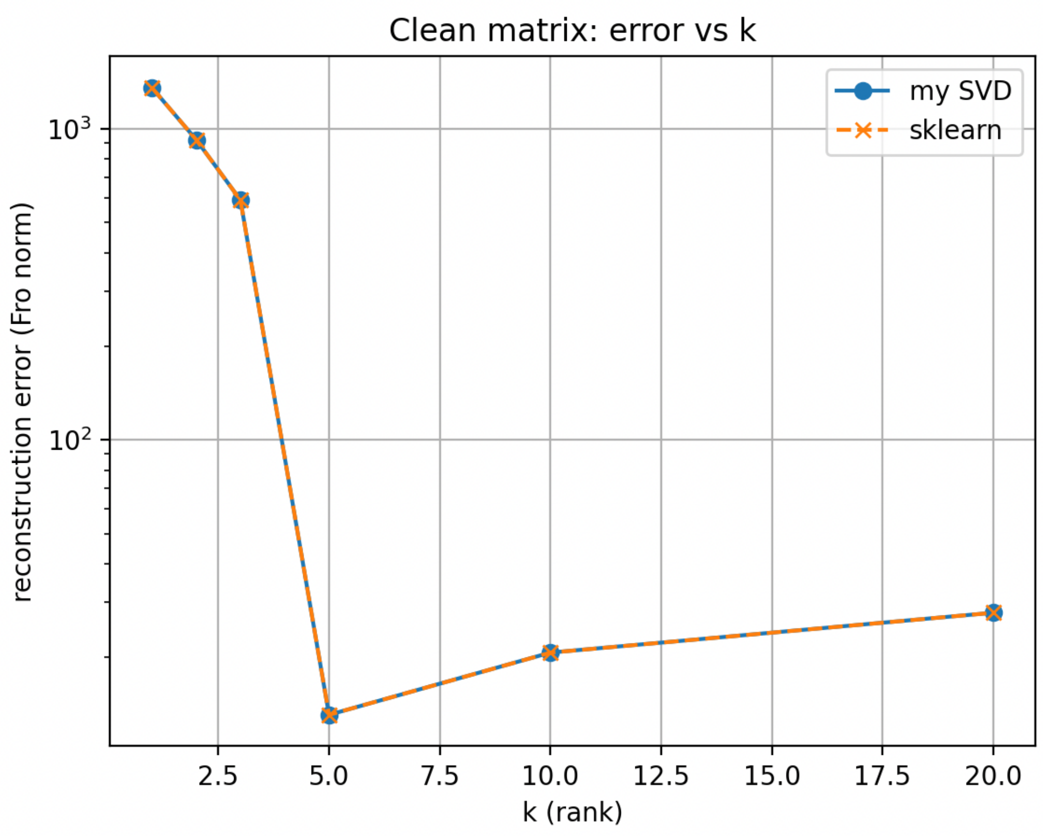
The implementation is evaluated using reconstruction error measured with the Frobenius norm. Truncated SVD is applied for different values of k, and the results are compared with the randomized SVD implementation provided by scikit-learn.

Results

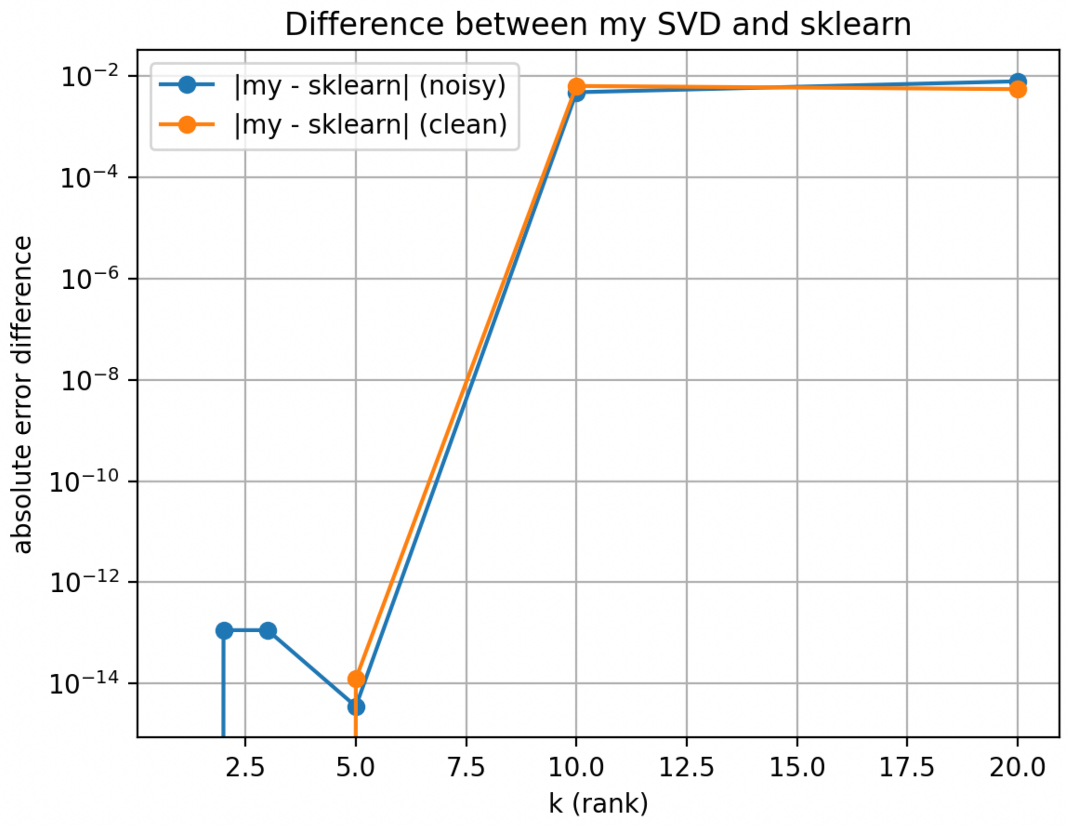


```
tests/test_skeleton.py::test_fib PASSED                                                                                                    [ 12%]
tests/test_skeleton.py::test_main PASSED                                                                                                   [ 25%]
tests/test_svd.py::test_shapes_make_sense PASSED                                                                                           [ 37%]
tests/test_svd.py::test_reconstruction_is_good_full_rank PASSED                                                                            [ 50%]
[tests/test_svd.py::test_singular_values_nonnegative_sorted PASSED                                                                         [ 62%]
tests/test_svd.py::test_orthonormality PASSED                                                                                              [ 75%]
tests/test_svd.py::test_compare_with_sklearn_topk_reconstruction PASSED                                                                    [ 87%]
tests/test_svd.py::test_synthetic_denoising_rank_r_is_better_than_noisy_fit PASSED                                                         [100%]

================================================================ tests coverage ==================================================================
_______________________________________________ coverage: platform darwin, python 3.13.7-final-0 _________________________________________________

Name                    Stmts   Miss Branch BrPart  Cover   Missing
-------------------------------------------------------------------
src/svd/__init__.py         6      0      0      0   100%
src/svd/skeleton.py        32      1      2      0    97%   135
src/svd/svd.py             17      0      4      1    95%   29->28
src/svd/synthetic.py       13      0      0      0   100%
-------------------------------------------------------------------
TOTAL                      68      1      6      1    97%
=============================================================== 8 passed in 0.69s ================================================================
```



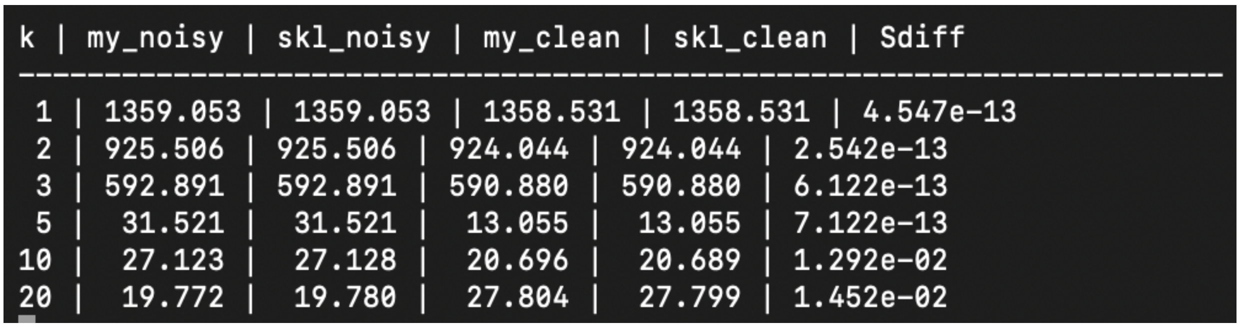


Clean matrix: error vs k
reconstruction error (Fro norm)
10^3
10^2
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
k (rank)
my SVD
sklearn



Difference between my SVD and sklearn
absolute error difference
10^{-2}
10^{-4}
10^{-6}
10^{-8}
10^{-10}
10^{-12}
10^{-14}
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
k (rank)
|my - sklearn| (noisy)
|my - sklearn| (clean)

Comparison with scikit-learn



```
k | my_noisy | skl_noisy | my_clean | skl_clean | Sdiff
----------------------------------------------------------------------
 1 | 1359.053 | 1359.053 | 1358.531 | 1358.531 | 4.547e-13
 2 | 925.506 | 925.506 | 924.044 | 924.044 | 2.542e-13
 3 | 592.891 | 592.891 | 590.880 | 590.880 | 6.122e-13
 5 |  31.521 |  31.521 |  13.055 |  13.055 | 7.122e-13
10 |  27.123 |  27.128 |  20.696 |  20.689 | 1.292e-02
20 |  19.772 |  19.780 |  27.804 |  27.799 | 1.452e-02
```

The results obtained using the custom SVD implementation closely match those produced by scikit-learn. Differences between reconstruction errors are minimal and can be attributed to numerical precision and the approximate nature of randomized SVD.

Conclusion

In this project, a custom implementation of Singular Value Decomposition was developed and evaluated. The results demonstrate that the implementation is correct and produces results comparable to scikit-learn, while providing insight into the internal workings of the SVD algorithm.