



Uniwersytet
Bielsko-Bialski

Analiza macierzowa w informatyce

Zadanie SVD

Sprawozdanie z ćwiczeń
Matematyka Konkretna

Data wykonania:
28.06.2025

Autor:
Bartosz Bieniek 058085

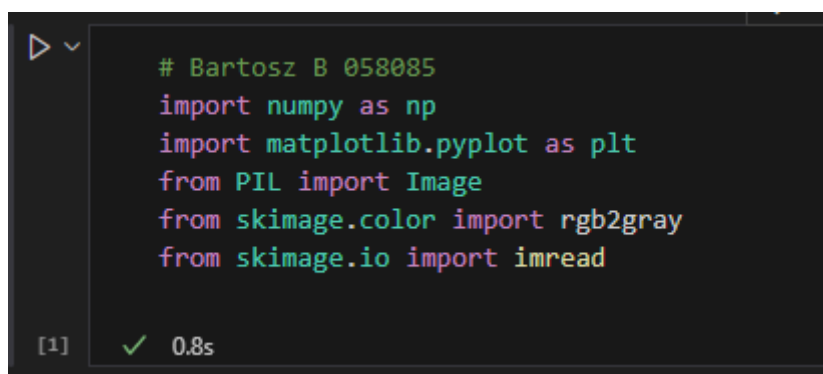
1. Cel ćwiczenia

Zadanie dotyczy kompresji obrazu metodą SVD zgodnie z wariantem zadania. Wstawienie projektu na serwis [Github](#).

2. Przebieg ćwiczenia

1. Import bibliotek

W pierwszym kroku zaimportowano niezbędne biblioteki do przetwarzania obrazu oraz przeprowadzenia dekompozycji SVD. Wykorzystano m.in. biblioteki numpy, matplotlib.pyplot, PIL oraz skimage. Biblioteki te zapewniły funkcje do wczytywania obrazu, jego przekształceń oraz wizualizacji wyników. Dzięki temu przygotowano środowisko do dalszej analizy obrazu.

A screenshot of a Jupyter Notebook cell with a dark background. The code is written in a light green monospace font. It starts with a comment line '# Bartosz B 058085' followed by five import statements: 'import numpy as np', 'import matplotlib.pyplot as plt', 'from PIL import Image', 'from skimage.color import rgb2gray', and 'from skimage.io import imread'. At the bottom of the cell, there is a status bar showing '[1]', a green checkmark, and '0.8s'.

```
# Bartosz B 058085
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image
from skimage.color import rgb2gray
from skimage.io import imread
```

[1] ✓ 0.8s

Rys. 1. Import bibliotek

2. Wczytanie obrazu i konwersja do skali szarości

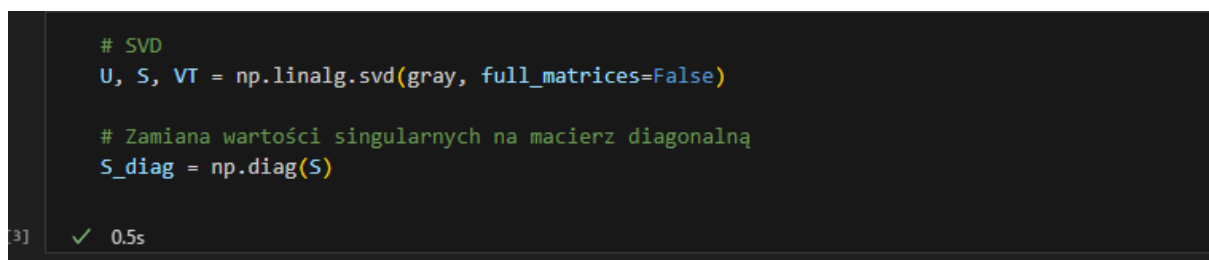
Wczytano obraz zapisany w pliku 1.webp i poddano go konwersji do skali szarości. W tym celu wykorzystano funkcję `rgb2gray` z biblioteki `skimage`, która przekształciła obraz z przestrzeni RGB do jednej warstwy luminancji. Konwersja była konieczna, ponieważ SVD przeprowadza się na macierzy dwuwymiarowej. Otrzymany obraz w skali szarości został wyświetlony w celu wstępnej oceny jakości.



Rys. 2. Wczytanie obrazu i konwersja do skali szarości.

3. Obliczenie SVD

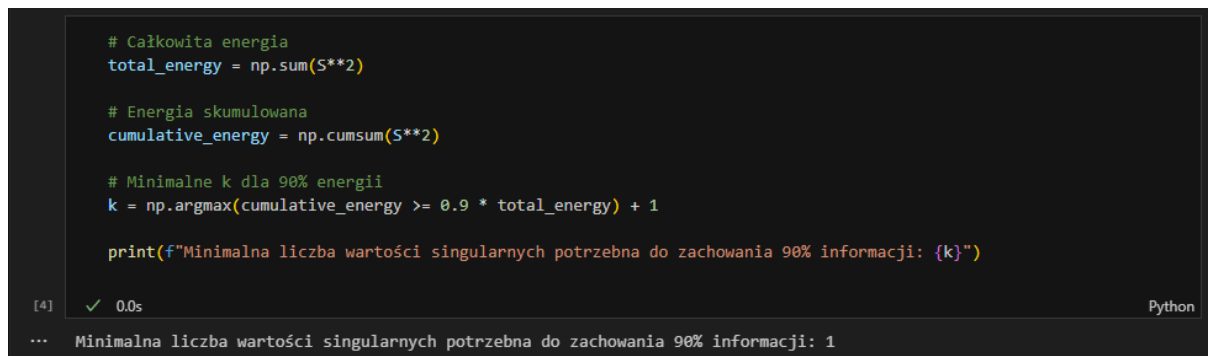
Na przekształconej macierzy obrazu przeprowadzono dekompozycję singularną (SVD). Zastosowano funkcję `np.linalg.svd` z opcją `full_matrices=False`, aby uzyskać zredukowane macierze UU , SS , VT . Uzyskane wartości singularne zapisano w postaci wektora oraz przekształcono do postaci macierzy diagonalnej, co ułatwiło dalsze operacje numeryczne. W ten sposób przygotowano dane do analizy skumulowanej energii sygnału.



Rys. 3. Obliczenia SVD

4. Obliczenie energii i wybór k dla 90% informacji

Obliczono całkowitą energię sygnału jako sumę kwadratów wartości singularnych. Następnie wyznaczono energię skumulowaną w funkcji kolejnych wartości SS. Na tej podstawie ustalono minimalną liczbę wartości singularnych kk , przy której suma skumulowanej energii przekroczyła 90% całkowitej energii. Wartość kk została automatycznie wyznaczona przez funkcję `np.argmax` i zaprezentowana jako wynik główny eksperymentu.



```
# Całkowita energia
total_energy = np.sum(S**2)

# Energia skumulowana
cumulative_energy = np.cumsum(S**2)

# Minimalne k dla 90% energii
k = np.argmax(cumulative_energy >= 0.9 * total_energy) + 1

print(f"Minimalna liczba wartości singularnych potrzebna do zachowania 90% informacji: {k}")
```

[4] ✓ 0.0s Python

... Minimalna liczba wartości singularnych potrzebna do zachowania 90% informacji: 1

Rys. 4. Obliczenie energii i wybór k dla 90% informacji

3. Wnioski

Zastosowanie dekompozycji SVD do kompresji obrazu okazało się skuteczne, pozwalając na znaczne zmniejszenie liczby parametrów potrzebnych do przechowania obrazu przy jednoczesnym zachowaniu jego jakości wizualnej.

Minimalna liczba wartości singularnych konieczna do zachowania 90% informacji energetycznej okazała się znacznie mniejsza niż wymiar oryginalnego obrazu, co potwierdziło obecność redundancji w danych wizualnych.