# Przetwarzanie danych

Joanna Klimek

Informatyka, WIET

19.03.2021r.

**Zadanie:**

Obliczyć macierz odwrotną do $A + uv^T$, wykorzystując $A^{-1}$ oraz $A + uv^T$, gdzie

A – macierz $\mathbb{R}^{N \times N}$

u, v – wektory $\mathbb{R}^{N \times 1}$

**Metody obliczeń:**

1. Funkcją **numpy.linalg.inv**
2. Wzorem Shermana-Morrisona:

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^T A^{-1}}{1 + v^T A^{-1} u}$$

3. Wzorem Shermana-Morrisona z wykorzystaniem łączności jak poniżej:

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}(uv^T)A^{-1}}{1 + v^T A^{-1} u}$$

**Pomocniczy kod w Pythonie:**

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from timeit import default_timer as timer

# wymiary macierzy (N)
sizes = [100, 200, 300, 400, 500, 600, 700, 800, 900, 1000]
index = -1

# tablica do zapisywania czasów
# times[nr_metody][rozmiar_N][nr_pomiaru]
times = [[[None for k in range(10)] for j in range(len(sizes))] for i in
range(3)]

for size in sizes:
    n = size
    index += 1

    A = np.random.rand(n, n)    # macierz A
    u = np.random.rand(n, 1)    # wektor u
    v = np.random.rand(n, 1)    # wektor v

    inversed_A = np.linalg.inv(A)    # A^(-1)
    transposed_v = v.transpose()     # v^T

    rank_one_update_A = A + (u @ transposed_v)  # A + uv^T

    # METODA PIERWSZA
    method = 0

    for i in range(10):
        start = timer()
        inverse_A_1 = np.linalg.inv(rank_one_update_A)
        end = timer()

        times[method][index][i] = (end - start)


    # METODA DRUGA
    method = 1

    for i in range(10):
        start = timer()
        numerator = (((inversed_A @ u) @ transposed_v) @ inversed_A)
        denominator = 1 + (transposed_v @ inversed_A @ u)
        inverse_A_2 = inversed_A - (numerator / denominator)
        end = timer()

        times[method][index][i] = (end - start)


    # METODA TRZECIA
    method = 2

    for i in range(10):
        start = timer()
        numerator = ((inversed_A @ (u @ transposed_v)) @ inversed_A)
        denominator = 1 + (transposed_v @ inversed_A @ u)
        inverse_A_3 = inversed_A - (numerator / denominator)
```

```
        end = timer()

        times[method][index][i] = (end - start)

#--------------- PANDAS -----------------

rozmiary = []
for i in range(len(sizes)):
    rozmiary += ([sizes[i]] * 10)

for i in range(3):
    print("Metoda", i+1)

    czasy = []
    for j in range(len(sizes)):
        czasy += times[i][j]

    dict = {'Rozmiar': rozmiary, 'Czasy': czasy}

    dataFrame = pd.DataFrame(dict)
    print(dataFrame)

    print("\nSrednie czasy:")
    print(dataFrame.groupby('Rozmiar')['Czasy'].mean())

    print("\nOdchylenia standardowe:")
    print(dataFrame.groupby('Rozmiar')['Czasy'].std())
    print("\n\n")

    #dataFrame.groupby('Rozmiar')['Czasy'].mean().plot()
    plt.errorbar(sizes, dataFrame.groupby('Rozmiar')['Czasy'].mean(), yerr
= dataFrame.groupby('Rozmiar')['Czasy'].std(), fmt = '-o')
    plt.xlabel("N", fontsize = 16)
    plt.ylabel("Mean time", fontsize = 16)
    plt.title("Method " + str(i+1), fontsize = 20)
    plt.show()
```

**Tabela** podsumowująca częściowe wyniki pomiarów czasów odwracania macierzy 3 metodami:

**Częściowe wyniki:**

Metoda 1

| | Rozmiar | Czasy |
|---|---|---|
| 0 | 100 | 0.002299 |
| 1 | 100 | 0.002006 |
| 2 | 100 | 0.002180 |
| 3 | 100 | 0.004337 |
| 4 | 100 | 0.002199 |
| .. | ... | ... |
| 95 | 1000 | 0.081295 |
| 96 | 1000 | 0.084547 |
| 97 | 1000 | 0.080304 |
| 98 | 1000 | 0.129869 |
| 99 | 1000 | 0.083439 |

[100 rows x 2 columns]

Metoda 2

| | Rozmiar | Czasy |
|---|---|---|
| 0 | 100 | 0.000438 |
| 1 | 100 | 0.000372 |
| 2 | 100 | 0.000546 |
| 3 | 100 | 0.000430 |
| 4 | 100 | 0.000417 |
| .. | ... | ... |
| 95 | 1000 | 0.043798 |
| 96 | 1000 | 0.046041 |
| 97 | 1000 | 0.047207 |
| 98 | 1000 | 0.045269 |
| 99 | 1000 | 0.046438 |

[100 rows x 2 columns]

Metoda 3

| | Rozmiar | Czasy |
|---|---|---|
| 0 | 100 | 0.001927 |
| 1 | 100 | 0.000682 |
| 2 | 100 | 0.000531 |
| 3 | 100 | 0.000485 |
| 4 | 100 | 0.000441 |
| .. | ... | ... |
| 95 | 1000 | 0.084176 |
| 96 | 1000 | 0.078787 |
| 97 | 1000 | 0.138376 |
| 98 | 1000 | 0.074722 |
| 99 | 1000 | 0.082351 |

[100 rows x 2 columns]

**Średnie czasy:**

Srednie czasy:

| Rozmiar | |
|---|---|
| 100 | 0.003027 |
| 200 | 0.006676 |
| 300 | 0.009341 |
| 400 | 0.012882 |
| 500 | 0.015441 |
| 600 | 0.021500 |
| 700 | 0.041068 |
| 800 | 0.052915 |
| 900 | 0.069809 |
| 1000 | 0.088143 |

Name: Czasy, dtype: float64

Srednie czasy:

| Rozmiar | |
|---|---|
| 100 | 0.000444 |
| 200 | 0.000570 |
| 300 | 0.002467 |
| 400 | 0.003253 |
| 500 | 0.005910 |
| 600 | 0.008580 |
| 700 | 0.017168 |
| 800 | 0.023417 |
| 900 | 0.031734 |
| 1000 | 0.044825 |

Name: Czasy, dtype: float64

Srednie czasy:

| Rozmiar | |
|---|---|
| 100 | 0.000677 |
| 200 | 0.000973 |
| 300 | 0.002821 |
| 400 | 0.004785 |
| 500 | 0.008700 |
| 600 | 0.014928 |
| 700 | 0.031123 |
| 800 | 0.040645 |
| 900 | 0.055581 |
| 1000 | 0.084233 |

Name: Czasy, dtype: float64

**Odchylenia standardowe:**

Odchylenia standardowe:

| Rozmiar | |
|---|---|
| 100 | 0.001630 |
| 200 | 0.002066 |
| 300 | 0.003742 |
| 400 | 0.003413 |
| 500 | 0.002027 |
| 600 | 0.003366 |
| 700 | 0.004539 |
| 800 | 0.004767 |
| 900 | 0.017957 |
| 1000 | 0.014861 |

Name: Czasy, dtype: float64

Odchylenia standardowe:

| Rozmiar | |
|---|---|
| 100 | 0.000047 |
| 200 | 0.000046 |
| 300 | 0.000501 |
| 400 | 0.000447 |
| 500 | 0.001014 |
| 600 | 0.000825 |
| 700 | 0.000627 |
| 800 | 0.001732 |
| 900 | 0.001400 |
| 1000 | 0.001989 |

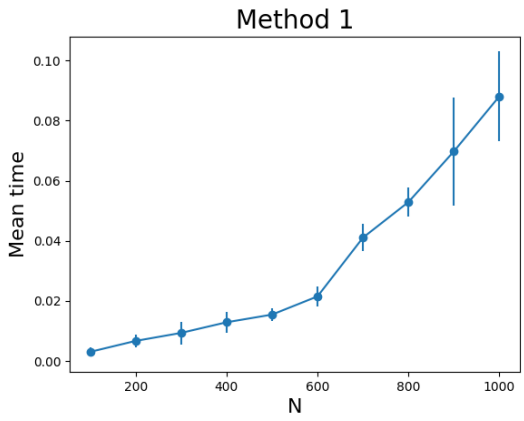
Name: Czasy, dtype: float64

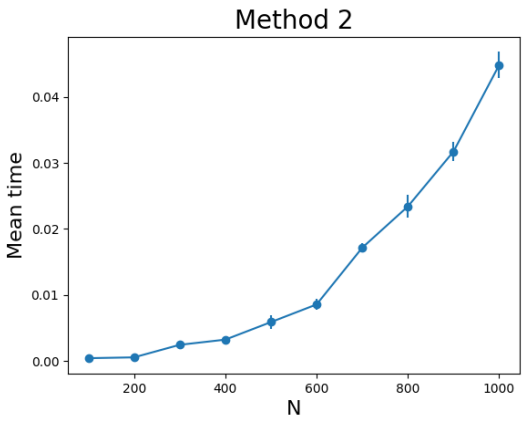
Odchylenia standardowe:

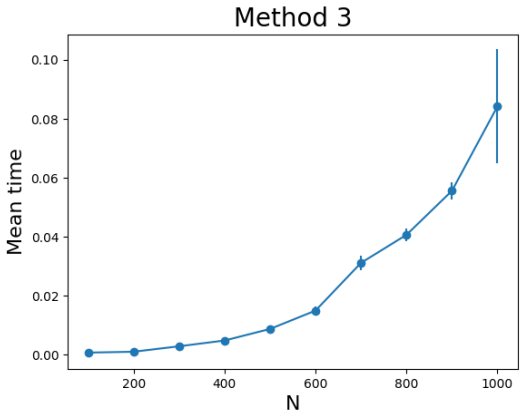
| Rozmiar | |
|---|---|
| 100 | 0.000448 |
| 200 | 0.000431 |
| 300 | 0.000403 |
| 400 | 0.000725 |
| 500 | 0.001175 |
| 600 | 0.001392 |
| 700 | 0.002381 |
| 800 | 0.002111 |
| 900 | 0.002876 |
| 1000 | 0.019332 |

Name: Czasy, dtype: float64

**Wygenerowane wykresy:**




**Złożoności obliczeniowe:**

Metoda 1: O($n^3$)

Metoda 2: O($3n^2$)

Metoda 2: O($n^3$)

## Wnioski:

Po zróżnicowaniu początkowych wielkości macierzy i poprawieniu drobnego błędu w kodzie wyniki mają dużo więcej sensu.

We wszystkich trzech metodach wraz ze wzrostem rozmiarów macierzy i wektorów wzrasta również średni czas wykonania oraz odchylenie standardowe, wzrost ten nie jest jednak liniowy. Wszystkie trzy wykresy wyglądają bardzo podobnie, ale trzecia metoda wypada najlepiej (choć różnice są minimalne).

Warto jednak podkreślić, że losowość danych miała duży wpływ na końcowe wyniki i przy każdym uruchomieniu programu wykresy wyglądały nieco inaczej.