Zadanie numeryczne 4

Joanna Szewczyk

19.11.2024r.

- ## Treść zadania:

Zadana jest macierz:

$$A = \begin{pmatrix} 5 & 3 & 1 & 1 & \dots & 1 & 1 & 1 & 1 \\ 1 & 5 & 3 & 1 & \dots & 1 & 1 & 1 & 1 \\ 1 & 1 & 5 & 3 & \dots & 1 & 1 & 1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & 1 & \dots & 1 & 5 & 3 & 1 \\ 1 & 1 & 1 & 1 & \dots & 1 & 1 & 5 & 3 \\ 1 & 1 & 1 & 1 & \dots & 1 & 1 & 1 & 5 \end{pmatrix}$$

Oraz wektor b≡$(2, \dots, 2)^T$. Macierz A ma liczby 5 na diagonali, 3 na pierwszej pozycji nad diagonalą, a pozostałe elementy są równe 1. Wymiar macierzy ustalamy na N=120.

Rozwiąż numerycznie równanie Ay=b, stosując odpowiednią metodę. Uwaga: algorytm należy go zaimplementować samodzielnie.

• Sprawdź swój wynik przy użyciu procedur bibliotecznych lub pakietów algebry komputerowej.

• Potraktuj N jako zmienną i zmierz czas potrzebny do uzyskania rozwiązania w funkcji N. Wynik przedstaw na wykresie. Czy wykres jest zgodny z oczekiwaniami?

- ## Wstęp:

Na początku zajmijmy się wyjaśnieniem problemu przedstawionego w zadaniu. Rozważamy macierz, która nie jest gęsta. Obliczenia przy użyciu standardowych metod byłyby bardzo złożone i czasochłonne. Naszym celem jest natomiast opracowanie rozwiązania, które będzie działać efektywnie w czasie liniowym.

Jeżeli uda nam się uprościć macierz A, przekształcając ją w macierz rzadką, a najlepiej w taką, która zawiera niezerowe elementy jedynie na diagonalach, czas wykonywania obliczeń znacząco się zmniejszy. Możemy to osiągnąć, ponieważ nasza macierz składa się niemal wyłącznie z wartości 1, poza elementami znajdującymi się na głównych przekątnych. Wystarczy więc zmniejszyć wartość każdego elementu macierzy o 1, co pozwoli przekształcić ją w macierz rzadką z dwiema niezerowymi przekątnymi.

$$A = \begin{pmatrix} 5 & 3 & 1 & 1 & \dots & 1 & 1 & 1 & 1 \\ 1 & 5 & 3 & 1 & \dots & 1 & 1 & 1 & 1 \\ 1 & 1 & 5 & 3 & \dots & 1 & 1 & 1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & 1 & \dots & 1 & 5 & 3 & 1 \\ 1 & 1 & 1 & 1 & \dots & 1 & 1 & 5 & 3 \\ 1 & 1 & 1 & 1 & \dots & 1 & 1 & 1 & 5 \end{pmatrix}_{NxN} = \begin{pmatrix} 4 & 2 & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ 0 & 4 & 2 & 0 & \dots & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 2 & \dots & 0 & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & 0 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & 4 & 2 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 4 \end{pmatrix}_{NxN} + \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}_{NxN}$$

$$Dla\ macierzy\ (A\ \ można\ zastosować\ wzór\ Shermana - Morrisona:$$

$$(A' + wv^T)^{-1} = A'^{-1} - \frac{A'^{-1}wv^TA'^{-1}}{1 + v^TA'^{-1}w}$$

gdzie z = u , a u =v. Używając tego wzoru, można rozwiązać równanie, wykonując następujące kroki:

$1.\ Rozkład\ A' = LU$

$2.\ Rozwiązanie\ równania(LU\vec{y} = \vec{b})$

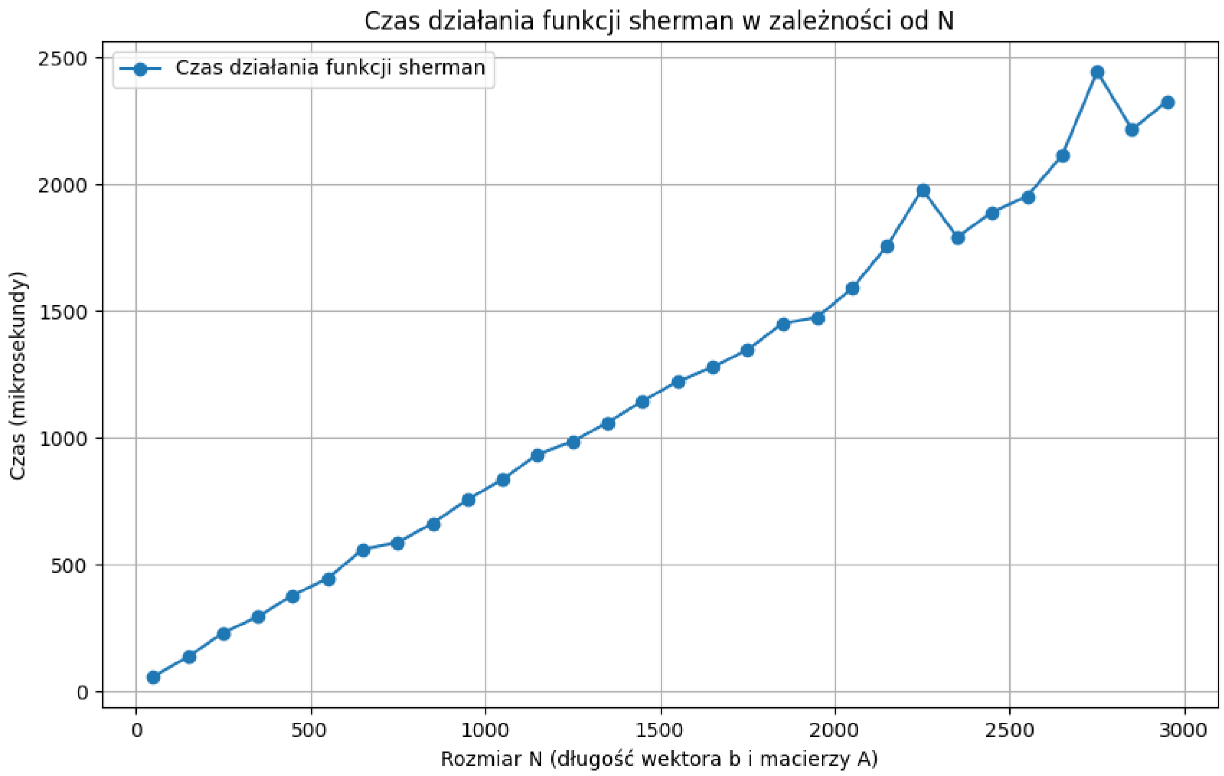
$3.\ Rozwiązanie\ równania\ LU\vec{z} = \vec{u})$

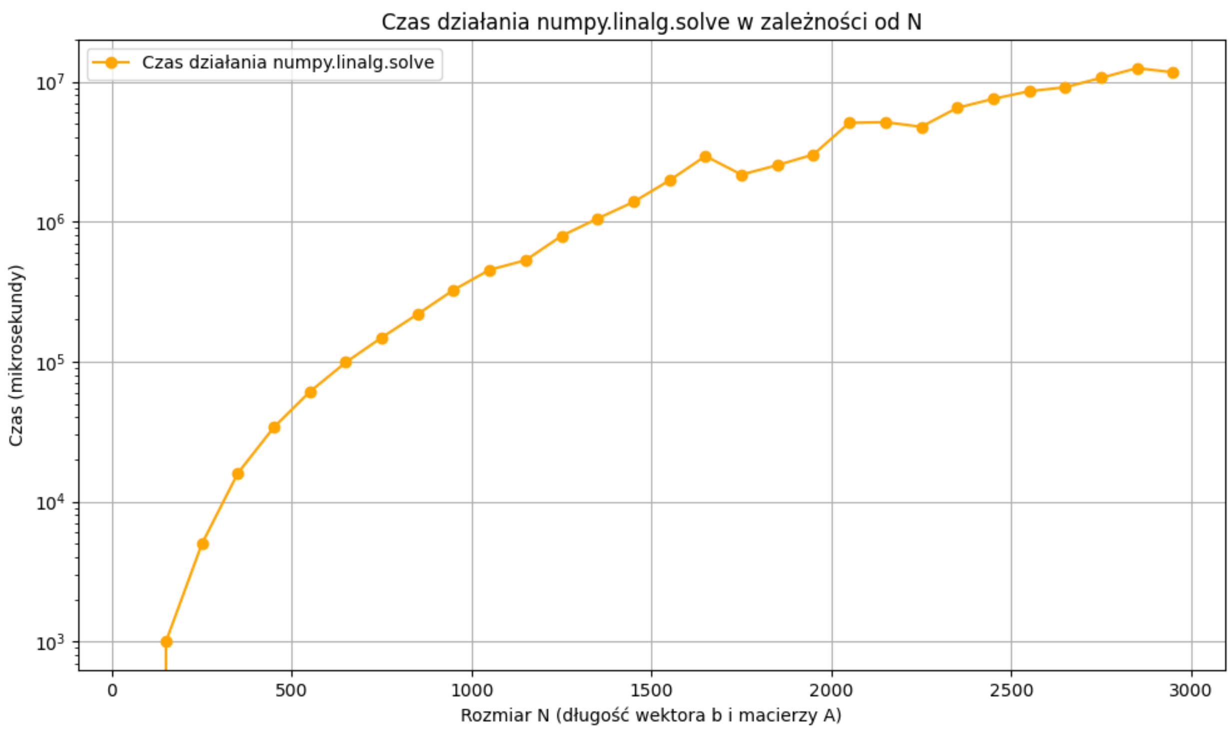
$4.\ Obliczenie\ (\vec{x} = \vec{y} - \frac{\vec{z}\vec{u}^T\vec{y}}{1 + \vec{u}^T\vec{z}}\vec{z}),\qquad gdzie:$

- $(\vec{z}\vec{u}^T\vec{y} = \sum_{i=1}^{N} \vec{y}_i\vec{z}_i)$
- $(\vec{u}^T\vec{z} = \sum_{i=1}^{N} \vec{z}_i)$

- ## Wyniki:

[0.0395778364116095, 0.03957783641160961, 0.03957783641160961, 0.03957783641160961, 0.0395778364116095, 0.03957783641160961, 0.03957783641160939, 0.039577836411609835, 0.03957783641160906, 0.03957783641161072, 0.03957783641160728, 0.039577836411614054, 0.03957783641160062, 0.0395778364116276, 0.03957783641157353, 0.039577836411681555, 0.03957783641146562, 0.03957783641189749, 0.03957783641103363, 0.03957783641276147, 0.03957783640930579, 0.03957783641621704, 0.03957783640239465, 0.039577836430039426, 0.039577836374749875, 0.03957783648532909, 0.03957783626417066, 0.0395778367064874, 0.03957783582185381, 0.03957783759112099, 0.03957783405258675, 0.039577841129655345, 0.03957782697551804, 0.03957785528379265, 0.039577798667243536, 0.03957791190034177, 0.03957768543414519, 0.03957813836653823, 0.03957723250175227, 0.0395790442313243, 0.039575420772180125, 0.039582667690468365, 0.039568173853891886, 0.039597161527044955, 0.03953918618073882, 0.03965513687335098, 0.03942323548812676, 0.039887038258575314, 0.03895943271767821, 0.040814643799472416, 0.037104221635884005, 0.04452506596306072, 0.02968337730870718, 0.05936675461741436
$]^T$


Czas działania funkcji sherman w zależności od N
Czas działania funkcji sherman
Czas (mikrosekundy)
Rozmiar N (długość wektora b i macierzy A)
0
500
1000
1500
2000
2500
3000


Czas działania numpy.linalg.solve w zależności od N
Czas działania numpy.linalg.solve
Czas (mikrosekundy)
Rozmiar N (długość wektora b i macierzy A)
10^3
10^4
10^5
10^6
10^7
0
500
1000
1500
2000
2500
3000

- Wnioski:

Wyniki uzyskane za pomocą zaimplementowanego algorytmu są niemal identyczne w porównaniu do tych uzyskanych z wykorzystaniem funkcji biblioteki Numpy. Różnice pojawiają się dopiero na piętnastym miejscu po przecinku, co najprawdopodobniej wynika z ograniczonej precyzji obliczeń numerycznych. Dzięki zastosowaniu wzoru Shermana-Morrisona oraz metody podstawiania wstecznego, udało się obliczyć równanie macierzowe w czasie liniowym O(n). Własnoręcznie zaimplementowany algorytm działa więc znacznie szybciej niż funkcje z biblioteki Numpy. Różnica w czasie działania jest zauważalna na wykresach. Na przykład przy N=3000 możemy zaobserwować,
że nasz algorytm wykonuje obliczenia w mniej niż 2500 milisekund, podczas gdy metodzie z biblioteki Numpy zajmuje ponad 10^7 milisekund.