Zadanie NUM 4- Sprawozdanie

Karol Cichowski

1.Wstęp

7. **(zadanie numeryczne NUM4)** Zadana jest macierz

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

oraz wektor $\mathbf{b} \equiv (2, \dots, 2)^T$. Macierz $\mathbf{A}$ 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 $\mathbf{Ay} = \mathbf{b}$, 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?

Do rozwiąznia tego równania użyjemy algorytmu Shermana-Marrisona. Wybrałem tą metode że macierz A (którą w dalszej części będę oznaczał jako po przez A_1) możemy bardzo łatwo rozłozyć na sume macierzy:

$$\mathbf{A_1} = \mathbf{A} + \mathbf{u}*\mathbf{v}^\mathbf{T} \qquad \text{, gdzie:}$$

$$A = \begin{pmatrix} 4 & 2 & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 4 & 2 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 4 & 2 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 4 & 2 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & 4 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 4 \end{pmatrix}_{N \times N} \qquad u = v = [1, 1, 1, \dots, 1, 1]_N$$

2. Sposób rozwiązania

Otrzymana macierz A to macierz rzadka, dzięki czemu po zaprojektowaniu dla niej algorytmu możemy pracować na niej dużo wydajniej. Z wzoru Shermana-Marrisona otrzymujemy wzór:

$$y = \mathbf{A}_1^{-1}\mathbf{b} = \left(\mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{u}\mathbf{v}^T\mathbf{A}^{-1}}{1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u}}\right)\mathbf{b}$$

Aby zminimalizować złożoność obliczeniową nie będziemy obliczać jawnie odwrotności macierzy A, zamiast tego użyjemy:

$$\begin{cases} A*z=b \\ A*q=u \end{cases}$$

Z czego otrzymamy:

$$y = \mathbf{z} - \frac{\mathbf{v}^T\mathbf{z}}{1 + \mathbf{v}^T\mathbf{q}}\mathbf{q}$$

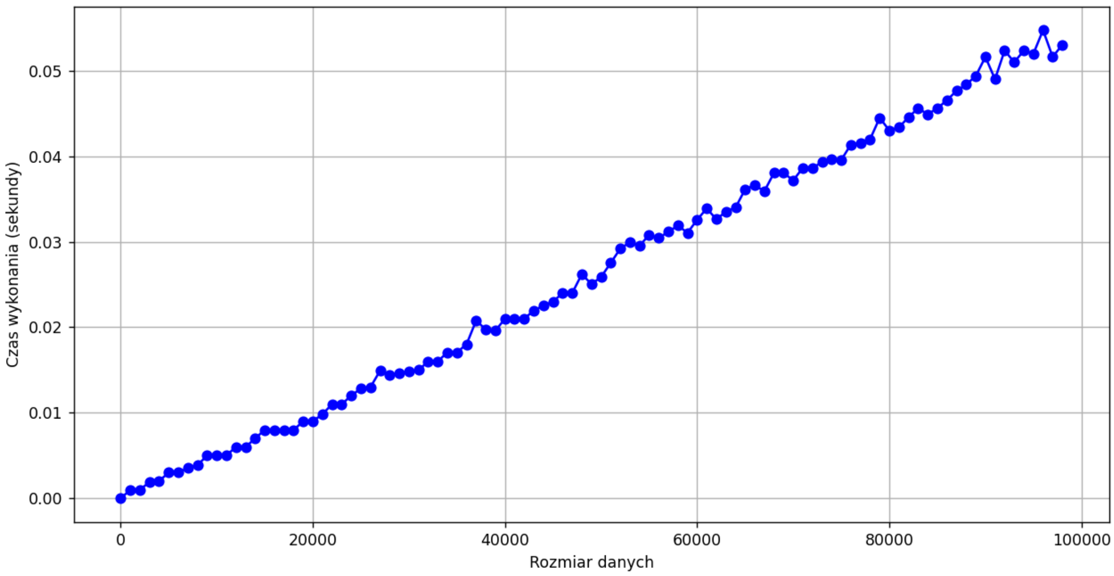
Powyższy układ możemy łatwo rozwiązać za pomocą backsubstitution z uwagi na konstrukcje macierzy A. Dzięki czemu złożoność obliczeniowa rozwiązania tego układu to **O(N) + O(N) = O(N)**. Rozwiązanie tego układu podstawiamy do powyższego wzoru i mamy nasz wynik. Rozwiązanie tego równania ma złożoność O(N), z czego wynika że **złożoność całej procedury to O(N)**, co jest dużo efektywniejsze niż O(N^3) które otrzymujemy dla niewyspecjalizowanych algorytmów.

3. Wyniki

Wynik naszego układu równań dla N=120 to:

y= [0.0158311345646438, 0.015831134564643856, 0.015831134564643745, 0.01583113456464391, 0.01583113456464358, 0.015831134564644245, 0.015831134564642912, 0.015831134564645577, 0.015831134564640192, 0.015831134564651017, 0.015831134564629423, 0.01583113456467261, 0.01583113456458618, 0.015831134564758986, 0.01583113456441343, 0.015831134565104543, 0.015831134563722316, 0.01583113456648677, 0.015831134560957805, 0.015831134572015737, 0.015831134549899928, 0.0158311345941316, 0.015831134505668254, 0.01583113468259495, 0.015831134328741503, 0.015831135036448396, 0.015831133621034665, 0.015831136451862127, 0.015831130790207204, 0.015831142113516994, 0.0158311946689736, 0.015831164760136684, 0.01583107417365809, 0.01583125534661528, 0.015830893000070084, 0.015831617692529665, 0.015830168308872017, 0.015833067076187313, 0.01582726954155672, 0.01583886461081796, 0.015815674472295538, 0.015862054749340382, 0.015769294195250638, 0.01595481530343007, 0.015583773087071262, 0.016325857519788933, 0.014841688654353535, 0.017810026385224276, 0.01187335092348285, 0.0237467018469657]

Natomiast wykres czasu od N prezentuje się następująco:



Tak jak spodziewaliśmy się jest to wykres liniowy, co potwierdza że złożoność czasowa tego algorytmu to O(N).