SPRAWOZDANIE
Z TRZECIEGO ZADANIA NUMERYCZNEGO

MACIEJ WÓJCIK

Spis treści:

1. Polecenie zadania

. **(zadanie numeryczne NUM3)** Wyznacz $\mathbf{y} = \mathbf{A}^{-1}\mathbf{x}$ dla

$$\mathbf{A} = \begin{pmatrix} 1.2 & \frac{0.1}{1} & \frac{0.4}{1^2} & & & & & & \\ 0.2 & 1.2 & \frac{0.1}{2} & \frac{0.4}{2^2} & & & & & \\ & 0.2 & 1.2 & \frac{0.1}{3} & \frac{0.4}{3^2} & & & & \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ & & & & & 0.2 & 1.2 & \frac{0.1}{N-2} & \frac{0.4}{(N-2)^2} \\ & & & & & & 0.2 & 1.2 & \frac{0.1}{N-1} \\ & & & & & & & 0.2 & 1.2 \end{pmatrix}$$

oraz $\mathbf{x} = (1, 2, \dots, N)^T$. Ustalamy $N = 100$. Oblicz również wyznacznik macierzy $\mathbf{A}$. Zadanie rozwiąż właściwą metodą (uzasadnij wybór) i wykorzystaj strukturę macierzy (w przeciwnym wypadku zadanie nie będzie zaliczone). Algorytm proszę zaprogramować samodzielnie – wyjątkowo nie należy stosować procedur bibliotecznych z zakresu algebry liniowej ani pakietów algebry komputerowej (chyba, że do sprawdzenia swojego rozwiązania, co zawsze jest mile widziane).

Dla ambitnych: potraktuj N jako zmienną i zmierz czas działania swojego programu w funkcji N. Wynik przedstaw na wykresie.

W zadaniu mamy daną macierz A, wektor x i niewiadomy wektor y. Naszym celem jest rozwiązanie równania $y = A^{-1}x$. Warto zwrócić uwagę na to, że jawne obliczanie odwrotności macierzy jest bardzo kosztowną operacją, więc chcemy się jej pozbyć.

Dodatkowo można przyjąć N jako zmienną i zmierzyć czas działania programu. Wyniki należy przedstawić na wykresie.

2. Instrukcja uruchomienia

Aby uruchomić program należy skorzystać z polecenia:

make run

Lub

python3 NUM3.py

3. Struktura macierzy

Możemy przekształcić nasze równanie mnożąc je lewostronnie przez A do postaci:

$$Ay = x$$

Macierz A jest macierzą wstęgową, większość jej elementów, poza tymi na diagonali i w jej pobliżu są równe 0.

Pierwszą oszczędnością jaką możemy zrobić, to przechowywać tylko elementy różne od zera. Pozwoli nam to na użycie dużo mniejszej ilości pamięci.

4. Algorytm Doolittle'a

W tym zadaniu skorzystam z Algorytmu Doolittle'a do rozłożenia macierzy do postaci

$$A = LU.$$

Wzory do tego algorytmu mają postać:

$$U_{ij} = A_{ij} - \sum_{k<i} L_{ik}U_{ki}$$

$$L_{ij} = \frac{A_{ij} - \sum_{k<i} L_{ik}U_{ki}}{U_{jj}}$$

W naszym przypadku, ze względu na istniejące elementy zerowe w macierzy A, wiele współczynników z powyższych wzorów się wyzeruje.

Kolejnym krokiem w celu rozwiązania równania jest przygotowanie równania do rozwiązania przez metody forward i backward substitution.

Wiemy już, że $A = LU$ i $Ax = y$, więc możemy uzyskać z tego:

$$LUx = y$$

Powyższe równanie rozbijemy na dwa, żeby skorzystać z metody forward i backward substitution:

$$Lb = x \qquad \text{i} \qquad Uy = b$$

Przejdziemy do Forward Substitution:

$$b_0 = x_0$$
$$b_n = x_n - L_n b_{n-1}$$

I Backward Substitution:

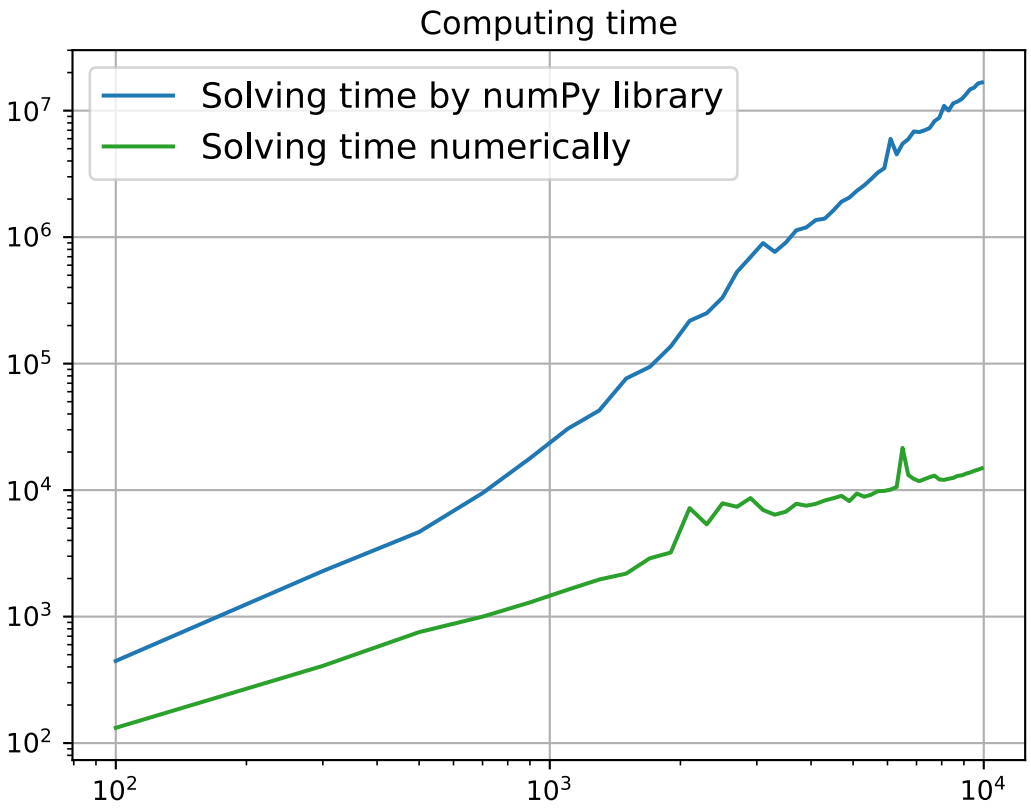
$$y_{99} = \frac{z_{99}}{Diag_{99}}$$

$$y_n = \frac{b_n - U_{1n} y_{n+1} - U_{2n} y_{n+2}}{U_{0n}} \quad \text{dla n < 98}$$

5. Wyniki

Wyznacznik macierzy A: 78240161.00959387

y =
[0.03287133486041395, 1.3396227980963753, 2.066480295894664, 2.825543605175336, 3.557571715528883, 4.284492868897645, 5.00721018451999, 5.727664002754518, 6.446615582748809, 7.164554400995276, 7.881773878242026, 8.598465868371878, 9.314759799907844, 10.030746230199034, 10.74649032115277, 11.462040127963592, 12.177431844626687, 12.892693237901542, 13.60784595684208, 14.322907124390252, 15.03789045794619, 15.75280707355121, 16.467666073000725, 17.182474979167374, 17.897240063340146, 18.611966594532937, 19.32665903159678, 20.041321172855753, 20.75595627381683, 21.47056714061568, 22.185156204831525, 22.899725583859315, 23.61427712998635, 24.328812470561147, 25.043333041083297, 25.757840112626393, 26.472334814693667, 27.186818154368854, 27.901291032443737, 28.615754257064278, 29.33020855532933, 30.04465458319117, 30.75909293394065, 31.473524145507586, 32.18794870676451, 32.902367062989086, 33.61677962061327, 34.331186751365145, 35.04558879589254, 35.75998606694211, 36.47437885215638, 37.18876741654113, 37.90315200464761, 38.617532842507245, 39.331910139350974, 40.04628408914067, 40.7606548719361, 41.47502265511775, 42.189387594482916, 42.90374983523002, 43.6181095128443, 44.33246675389621, 45.04682167676243, 45.76117439227791, 46.47552500432681, 47.189873610378676, 47.904220301975755, 48.61856516517662, 49.33290828096055, 50.047249725596565, 50.761589570980924, 51.47592788494589, 52.19026473154275, 52.904600171301595, 53.61893426146981, 54.33326705623165, 55.04759860691019, 55.761928962153874, 56.47625816810818, 57.19058626857465, 57.90491330515779, 58.61923931740096, 59.33356434291259, 60.04788841748285, 60.76221157519233, 61.47653384851288, 62.1908552684013, 62.9051758643867, 63.61949566465193, 64.33381469610926, 65.04813298447127, 65.76245055431694, 66.47676742915336, 67.19108363147355, 67.9053991828134, 68.61971410401006, 69.33402833257784, 70.04833794418792, 70.7650588638003, 71.53915685603329]



6. Wyniki

Wyniki obliczone za pomocą zaimplementowanego algorytmu są takie same jak wyniki obliczone za pomocą algorytmu NumPy. Dzięki temu, że zaimplementowany algorytm bierze pod uwagę strukturę macierzy, jego czas wykonania obliczeń jest znacząco szybszy, co tylko potwierdzają wykresy.

Wniosek z tego jest taki, że zastosowanie odpowiedniego algorytmu pozwala nam zaoszczędzić dużo czasu.