SPRAWOZDANIE NUM4

JAKUB KRĘCISZ

Treść zadania

Zadane jest macierz:

$$\boldsymbol{A} = \begin{pmatrix} 10 & 8 & 1 & \cdots & 1 & 1 & 1 \\ 1 & 10 & 8 & \cdots & 1 & 1 & 1 \\ 1 & 1 & 10 & \cdots & 1 & 1 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 1 & 1 & \cdots & 10 & 8 & 1 \\ 1 & 1 & 1 & \cdots & 1 & 10 & 8 \\ 1 & 1 & 1 & \cdots & 1 & 1 & 10 \end{pmatrix}$$

oraz wektor $\mathbf{b} \equiv (5, \dots, 5)^{\mathrm{T}}$. Macierz **A** ma liczby 10 na diagonali, 8 na pierwszej pozycji nad diagonalą, a pozostałe elementy są równe 1. Wymiar macierzy ustalamy na $N = 50$.

- Rozwiąż numerycznie równanie **Ay** = **b**, dobierając odpowiedni algorytm. **Uwaga:** algorytm na- leży go zaimplementować samodzielnie (mile widziane jest jednak sprawdzenie wyniku 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?

Omówienie zadania

Zacznijmy od omówienia problemu, z którym musimy się uporać. Mamy dane **A** i **b** oraz niewiadomy wektor **y**. Naszym zadaniem jest rozwiązanie równania: **Ay** = **b** numerycznie wybierając odpowiedni algorytm. Również mamy porównać wyniki z wbudowaną biblioteką stworzoną do tego typu operacji.

Jeżeli chodzi o macierz A, mamy do czynienia z macierzą, która nie jest gęsta, więc obliczenie jej standardową metodą będzie osiągało dużą złożoność a co za tym idzie zajmowało dużą ilość czasu na rozwiązanie takiej macierzy. Tak więc, pierwsze co możemy zrobić to sprowadzenie macierzy do macierzy rzadkiej, w szczególności do macierzy w postaci gdy jedyne niezerowe wartości znajdują się na jej diagonalach. W naszym przypadku, żeby sprowadzić macierz do takiej postaci wystarczy odjąć od każdego elementu wartość 1. Dzięki temu przekształceniu nasze obliczenia znacznie się skrócą. Macierz wtedy wyglądałaby tak:

$$\begin{pmatrix} 10 & 8 & 1 & \cdots & 1 & 1 & 1 \\ 1 & 10 & 8 & \cdots & 1 & 1 & 1 \\ 1 & 1 & 10 & \cdots & 1 & 1 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 1 & 1 & \cdots & 10 & 8 & 1 \\ 1 & 1 & 1 & \cdots & 1 & 10 & 8 \\ 1 & 1 & 1 & \cdots & 1 & 1 & 10 \end{pmatrix} = \begin{pmatrix} 9 & 7 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 9 & 7 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 9 & \cdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 9 & 7 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 9 & 7 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 9 \end{pmatrix} + \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}$$

Reprezentacja naszej macierzy **A** będzie przedstawiona jako suma dwóch macierzy, jak na górze. Widzimy że tak jak w przypadku zadania numerycznego **NUM3** idealnie nada się macierz wstęgowa do przechowania wartości jednej macierzy z naszej sumy:

$$\begin{pmatrix} 9 & 7 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 9 & 7 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 9 & \cdots & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 9 & 7 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 9 & 7 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 9 \end{pmatrix} \rightarrow \begin{pmatrix} 9 & 7 \\ 9 & 7 \\ 9 & 7 \\ \vdots & \vdots \\ 9 & 7 \\ 9 & 7 \\ 9 & 0 \end{pmatrix}$$

Jeżeli chodzi o macierz zbudowaną z samych jedynek, rozpiszmy ją jako iloczyn wektorów z samych jedynek, drugi oczywiście transponowany. Ułatwi nam to potem zastosowanie potrzebnych wzorów:

$$\begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \\ 1 \end{pmatrix} \times (1 \quad 1 \quad 1 \quad \cdots \quad 1 \quad 1 \quad 1)$$

Zapiszmy nasze macierze tak:

- Macierz rzadka jako B

- Iloczyn wektorów z samymi jedynkami jako $\mathbf{uv^T}$

$$Ay = b \Leftrightarrow (\mathrm{B} + \mathrm{uv^T})\mathrm{y} = \mathrm{b}$$

Zastosujemy wzór Shermana-Morrisona, ale żeby go zastosować, musimy zrobić jeszcze jedno przekształcenie naszego równania. Dokładniej to musimy pomnożyć obustronnie przez odwrotność naszej macierzy **A**, wtedy będziemy mogli zastosować ten wzór:

Nasze przekształcenie:

$$(B + uv^T)y = b \Leftrightarrow y = (B + uv^T)^{-1}b$$

Wzór Shermana-Morrisona:

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

Po zastosowaniu wzoru, nasze równanie ma postać:

$$y = (B + uv^T)^{-1}b \Leftrightarrow y = \left(B^{-1} - \frac{B^{-1}uv^T B^{-1}}{1 + \ v^T B^{-1}u}\right)b \Leftrightarrow y = \left(B^{-1}b - \frac{B^{-1}uv^T B^{-1}}{1 + \ v^T B^{-1}u}b\right)$$

Wiemy, że $B^{-1}b$ oraz $B^{-1}u$ będą wektorami, dla przejrzystości i dalszych przekształceń przyjmijmy, że:

$$B^{-1}b = x \qquad\qquad B^{-1}u = z$$

Wiemy również, że iloczyn wektoru transponowanego z wektorem daje nam skalar z sumą iloczynów ich elementów. Patrząc a iloczyn wektoru $\mathbf{v}^\mathbf{T}$ z naszymi **x** i **z**, wiemy nawet że naszym skalarem będzie po prostu suma elementów wektoru x oraz z ze względu na to, że wektor $\mathbf{v}^\mathbf{T}$ jest skonstruowany z samych jedynek. Mamy więc:

$$y = B^{-1}b - \frac{B^{-1}uv^T B^{-1}}{1 + \ v^T B^{-1}u}b \ \Leftrightarrow y = x - z(\frac{sum(x)}{1 + sum(z)})$$

Zajmijmy się jeszcze **x** i **z**. Pomnóżmy obustronnie przez macierz B, żeby dostać iloczyn B (który jest macierzą trójkątną górną) z naszymi **x** i **z.**

$$B^{-1}b = x \ \Leftrightarrow b = Bx$$

$$B^{-1}u = z \Leftrightarrow u = Bz$$

Tak jak powiedziałem wcześniej, macierz B jest trójkątnie górna. Jak dobrze wiemy (z zadania NUM3) w takim przypadku możemy zastosować backward substitution:

Dla Bx = b:

$$x_n = \frac{b_n - B_{1_n}x_{n+1}}{B_{0_n}} \qquad\qquad x_{49} = \frac{b_{49}}{B_{0_{49}}}$$

Dla n < 49

Dla Bz=u:

$$z_n = \frac{1 - B_{1_n} z_{n+1}}{B_{0_n}} \qquad\qquad z_{49} = \frac{1}{B_{0_{49}}}$$

Dla n < 49

(B_0 – diagonala środkowa, B_1 – diagonala nad środkową)

Teraz mamy już wszystko by zacząć implementację naszego programu.

Uruchomienie programu

Do uruchomienia programu wykorzystamy Makefile:

Aby uruchomić nasz program, wystarczy użyć polecenia:

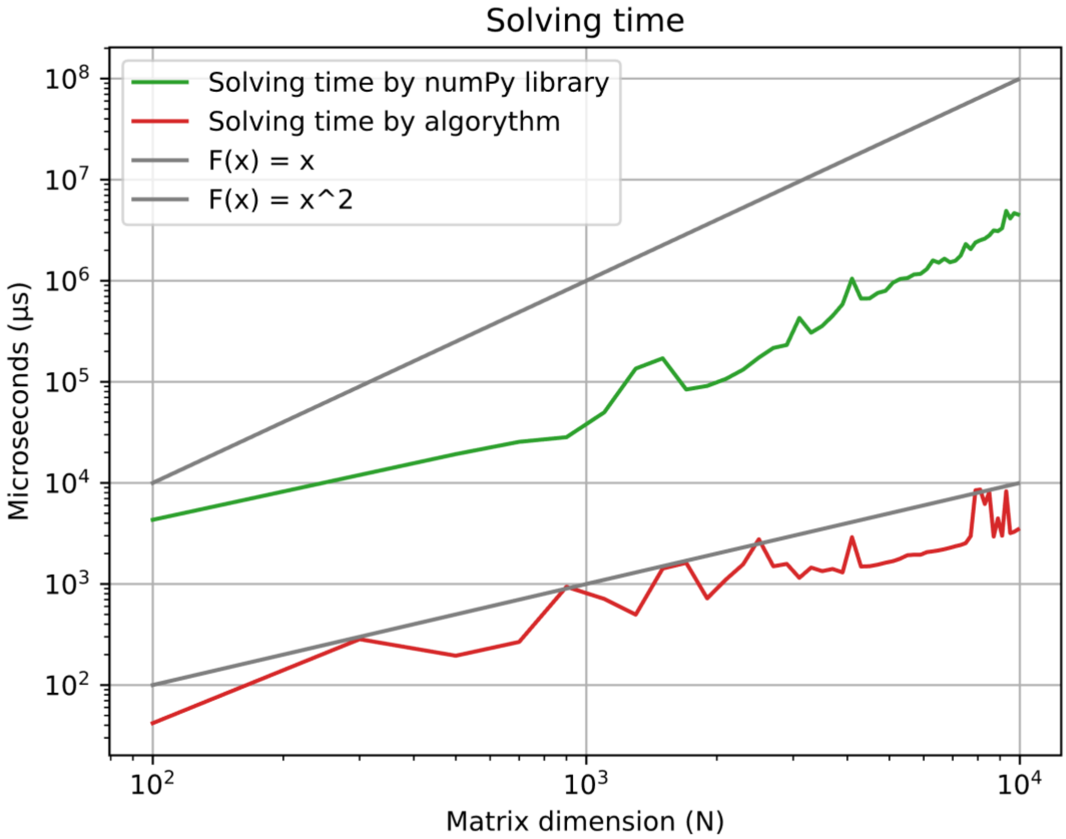
```
make run
```

Wyniki

1. Pierwszy podpunktem było policzenie dla N=50, wynik to:

y = [0.07525844089350037, 0.07525904117533852, 0.07525826938440369, 0.07525926168703423, 0.07525798586936636, 0.07525962620636797, 0.07525751720165161, 0.07526022877914401, 0.07525674246522518, 0.07526122486883524, 0.07525546177847939, 0.07526287146607977, 0.07525334472487927, 0.07526559339213704, 0.07524984510566277, 0.0752700929025582 6, 0.07524406002083556, 0.07527753086876468, 0.0752344969214272, 0.07528982628228972, 0.07521868853260927, 0.07531015135362709, 0.07519255629803279, 0.07534374994093965, 0.07514935811434514, 0.07539929046282379, 0.07507794887192268, 0.07549110234593842, 0.07495990502220382, 0.07564287300986267, 0.07476477131144413, 0.0758937592094108, 0.07444220334059656, 0.07630848945764337, 0.07390897873572605, 0.07699406394961972, 0.07302752581747077, 0.0781273605588052, 0.07157043017708939, 0.08000076923929544, 0.0691617618736019, 0.08309762848663654, 0.06518008569844908, 0.08821692642611872, 0.05859813120482912 4, 0.09667943934648726, 0.04771775745006959, 0.11066849131689238, 0.02973183348812022 4, 0.1337932506965414 7]

2. Drugim podpunktem było przyjęcie N jako zmienną, zmierzenie czasu i porzedstawienei wyników na wykresie:



Wnioski

Jeżeli chodzi o wyniki, nasz zaimplementowany algorytm oblicza macierze z takim samym rezultatem jak wbudowane biblioteki do tego stworzone (w naszym przypadku numPy).

Jeżeli chodzi o czas wykonania. Nasz zaimplementowany algorytm okazał się być naprawdę zauważalnie szybszy od metody ogólnej, którą jest w naszym przypadku, użycie funkcji z biblioteki numPy. Jak widzimy na wykresie, algorytm sobie o wiele lepiej radzi, dzięki zastosowaniu wzoru Shermana-Morrisona i backward substitution, udało nam się doprowadzić algorytm do wykonywania w czasie liniowym (O(n)), gdzie numpy te same obliczenia robi w O(n^2). Wniosek z tego płynie taki, że zastosowanie odpowiednich algorytmów do danej sytuacji, pozwala na o wiele szybsze poradzenie sobie z problemem rozwiązania takiego równania.