Zadanie numeryczne NUM3 - efektywne rozwiązywanie układu równań, wykorzystujące rzadką strukturę macierzy

Aleksander Pugowski

08-11-2022

1 Wprowadzenie

Zadanie polega na rozwiązaniu szeregu równań liniowych danych równaniem $Ay = x$, gdzie A jest macierzą wstęgową zawierającą niezerowe elementy jedynie bezpośrednio pod diagonalą i w obu przekątnych nad diagonalą oraz na znalezieniu wyznacznika macierzy A. Dodatkowo, należy znaleźć sposób na efektywne przechowywanie tak zadanej macierzy, aby nie przechowywać niepotrzebnych do niczego zer.

Do efektywnego rozwiązania danego układu równań zdecydowano się na faktoryzację macierzy metodą LU (bez pivotingu). Metodę wybrano z dwóch względów: faktoryzacja LU jest tutaj wyjątkowo szybka, ze względu na dużą ilość i korzystne ułożenie zer w macierzy, co powoduje 'kasowanie się' wielu wyrazów przy wyznaczaniu kolejnych elementów macierzy L czy macierzy U. Po znalezeniu sposobu na efektywne reprezentowanie takiej macierzy w pamięci, działania z zerami można w ogóle pominąć. Drugim powodem, jest pozostała część polecenia, czyli wyznaczenie wyznacznika macierzy A. Jak wiadomo, $det(A) = det(LU) = det(L) * det(U)$, gdzie L i U są macierzami trojkątnymi, więc wyliczenie ich wyznacznika sprowadza się do wyliczenia iloczynu elementów na diagonali. Do zadania wykorzystano bibliotekę NumPy (tworzenie macierzy o danym kształcie - efektywniejsze niż implementacja Pythona oraz walidacja wyników).

2 Wyniki

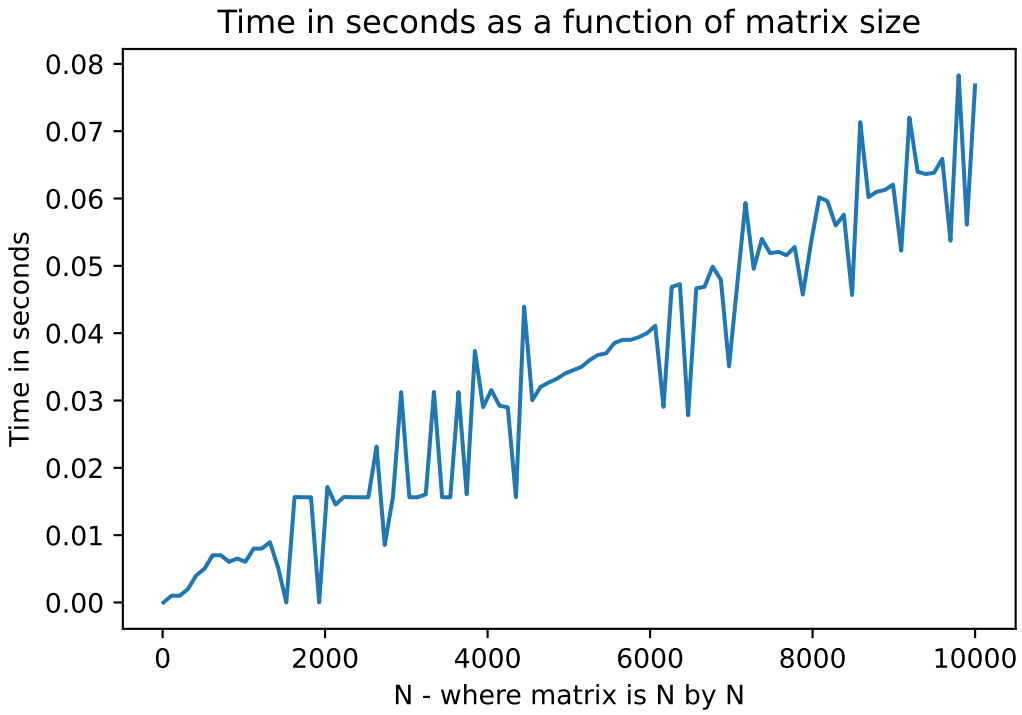
Dla macierzy A o wymiarach 100 na 100, wynikiem jest wektor o 100 współrzędnych, których wypisywanie tutaj mija się z celem. Dokładne dane można znaleźć w kodzie dołączonym do tego raportu, natomiast dwie pierwsze i dwie ostatnie współrzędne wektora y wynoszą:

$$y = \begin{bmatrix} 0.03287133486041395 \\ 1.3396227980963753 \\ \vdots \\ 70.7650588638003 \\ 71.53915685603329 \end{bmatrix}$$

Natomiast wyznacznik macierzy A wyniósł $det = 78240161.00959387$. Wyniki zostały potwierdzone tradycyjnym rozwiązaniem równania i wyliczeniem macierzy, przy pomocy biblioteki numerycznej NumPy.

Dodatkowo, zmierzono czas jaki zajmuje rozwiązanie równania tą metodą, a standardową metodą oferowaną przez NumPy (czyli, zgodnie z dokumentacją "computed using an LU decomposition [R40] with partial pivoting and row interchanges."). De facto, różnica czasu działania wynika więc głównie z wykorzystania rzadkiej struktury macierzy. Dla macierzy 100x100 przy 10-krotnym rozwiązaniu takiego równania funkcja biblioteczna potrzebowała 0.024 sekundy,

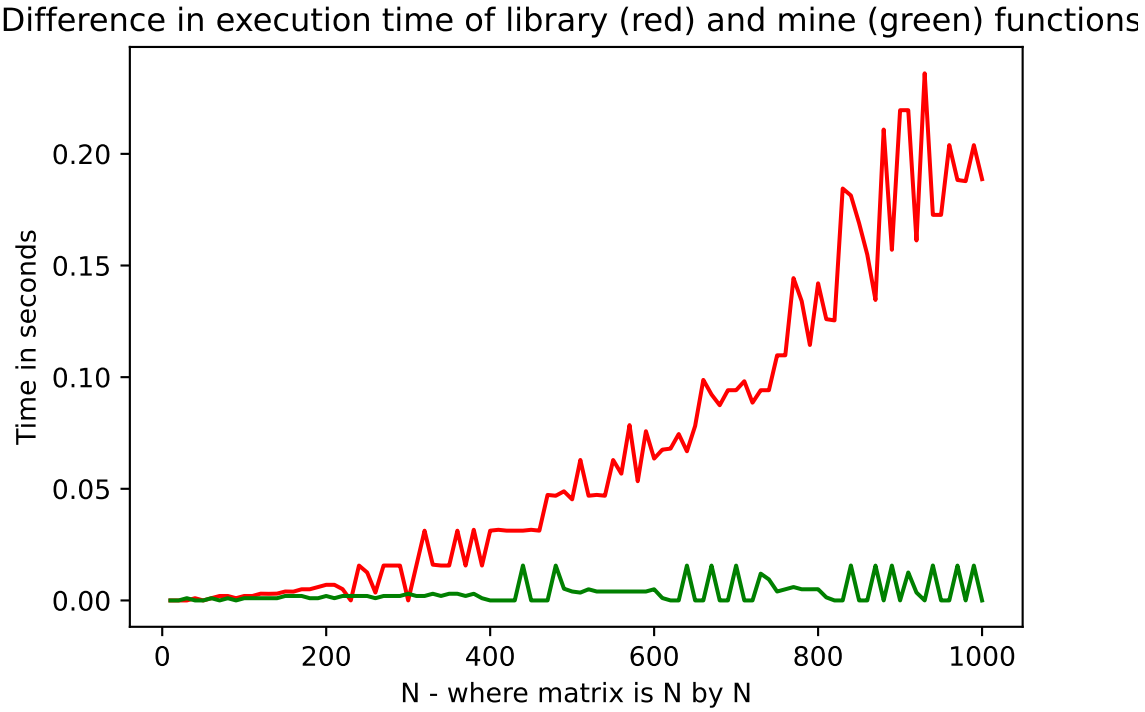
a opisywana tutaj funkcja 0.01 sekundy, co jest czasem ponad dwukrotnie krótszym.
Zmierzono również czas działania programu w funkcji rozmiaru macierzy, co zostało przedstawione na poniższym wykresie.



3 Dyskusja

Dzięki charakterystycznej strukturze macierzy, możliwe jest jej przechowywanie w pamięci w rozmiarze rzędu $O(n)$ w postaci zwykłej tablicy dwuwymiarowej. Jest to znaczna oszczędność, bo dla standardowego sposobu przechowywania potrzebujemy $O(n^2)$. Ta forma pozwala również w prostszy sposób pomijać zera w obliczeniach - zwyczajnie nie ma ich w pamięci.

Warto zwrócić uwagę, jak wykorzystanie rzadkiej struktury macierzy może poprawić wydajność obliczeń. Udało się trzykrotnie przyśpieszyć czas wyliczenia układu równań i to jedynie dla macierzy 100x100 - efekt ten będzie się potęgował wraz ze wzrostem ilości elementów w macierzy. Dla macierzy 1000 na 1000 różnica czasu wynosi już $\Delta = 2.97 - 0.11 = 2.86$. Poniższy wykres przedstawia różnicę czasu wykonania w funkcji rozmiaru macierzy.



Na wykresie zależności czasu działania programu od rozmiaru macierzy możemy zaobserwować liniowy trend, co nie stanowi zaskoczenia. To, co może jednak dziwić, to nieregularności pojawiające się po drodze, gwałtowne spadki czasu rozwiązania dla poszczególnych wartości N. Wydaje mi się, że to zachowanie nie wynika z przyczyn numerycznych, a z języka programowania w jakim rozwiązanie zostało zaimplementowane. Python ma zaawansowane i skomplikowane systemy zarządzania pamięcią w czasie wykonywania programu, co może przekładać się właśnie na tego typu nieregularności, gdzie (numerycznie) trend powinien być jednolicie liniowy.