Sprawozdanie z projektu nr 5,
Metody Numeryczne II

Piotr Onyszczuk, Maciej Pióro, gr. F3

12.12.2018 r.

1 Temat 2.

Metoda Jacobiego poszukiwania wszystkich wartości własnych rzeczywistej macierzy symetrycznej.

2 Opis metody.

Niech $A \in \mathbb{R}^{n \times n}$. Metoda Jacobiego produkuje ciąg maierzy podobnych A_k, $k = 0, 1, 2, 3, \ldots$, $(A_0 = A)$, który zbiega do macierzy diagonalnej, której wyrazami są wartości własne macierzy A. Kolejne macierze w ciągu wyznaczamy przy pomocy obrotów Givensa. Miarą odchylenia macierzy A_k od macierzy diagonalnej będzie połowa sumy kwadratów elementów nieleżących na przekątnej ($\omega(A_k) = \sum_{i<j} (a_{i,j}^{(k)})^2$). Będziemy dokonywać obrotów o kąt ϕ, wykonując je tylko gdy $a_{p,q}^2 \geq \frac{\omega(A)}{N}$, gdzie $N = \frac{n(n-1)}{2}$.

3 Implementacja.

1. *rotation_matrix* - funkcja zwracająca macierz obrotu ($O_{p,q}$) macierzy$A$ o odpowiedni kąt ϕ. Oblicza ona $c = cos(\phi)$ oraz $s = sin(\phi)$ z następujących zależności[1]:

$$\beta = \frac{a_{q,q} - a_{p,p}}{2a_{p,q}},$$

$$t = tg(\phi) = \frac{sgn(\beta)}{|\beta| + \sqrt{1+\beta^2}}$$

$$c = \frac{1}{\sqrt{1+t^2}}, \quad s = tc$$

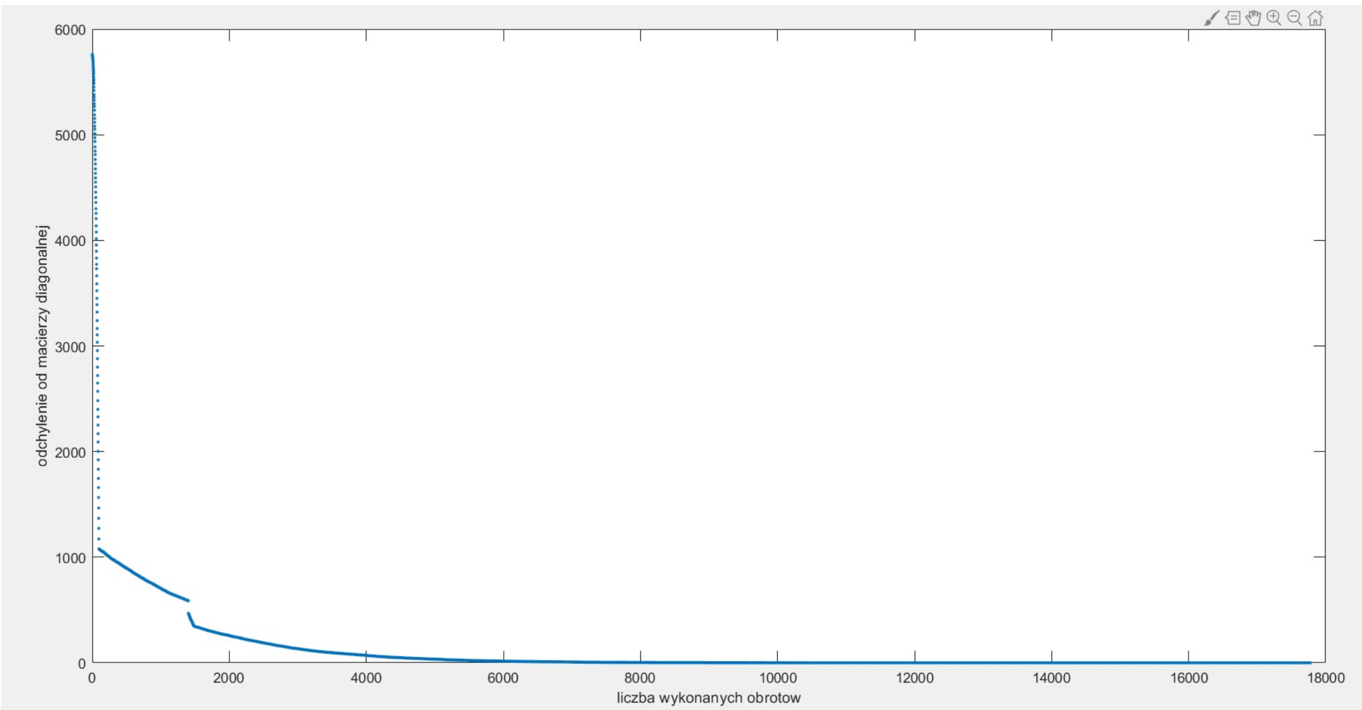
Macierz obrotu Givensa (wyróżnione wiersze to wiersze o indeksach p i q):

$$G_{p,q} = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & -s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix},$$

2. *calculate_deviation* - funkcja obliczająca miarę odchylenia macierzy A od macierzy diagonalnej jako połowę sumy kwadratów elementów pozadiagonalnych,

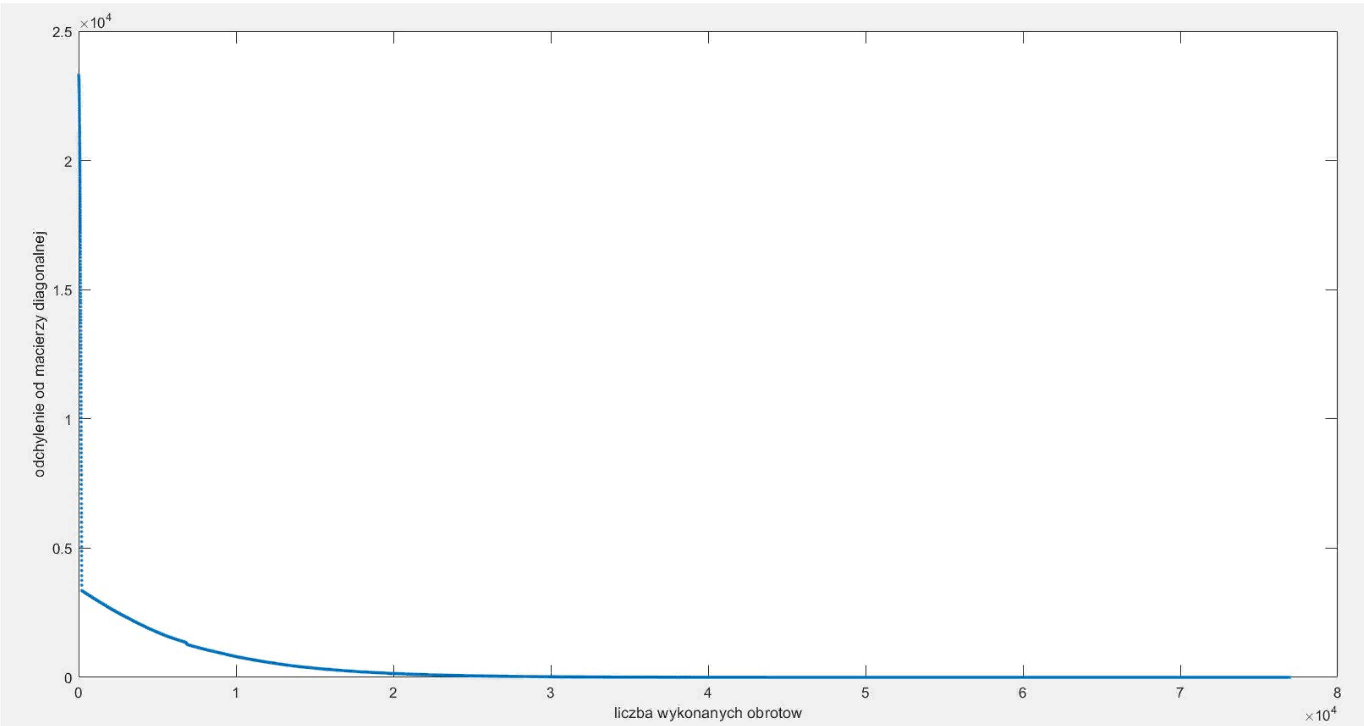
3. $Jacobi$ - funkcja zwracająca wektor eigenvalues wartości własnych macierzy A obliczony metodą Jacobiego. Działa dopóki $\omega(A_k) \geq eps$. Wykonuje obrót $O_{p,q}$ macierzy A o ile $a_{p,q} \geq \omega(A_k)$. Funkcja może się zakończyć również po wykonaniu $iter$ iteracji.

4 Przykłady




```
>> Jacobi([],100,[],[],true);
Wykonano 17747 obrotow
oraz 11 przejsc po calej macierzy.
Funkcja wykonywala sie 1.207327e+00 sekund.
Matlab liczy to samo 8.248871e-04 sekund.
Norma z roznicy miedzy wektorami wynikowymi wynosi 3.340220e-09.
```

Figure 1: Losowo generowana macierz $A \in \mathbb{R}^{100\times 100}$




```
>> Jacobi([],200,[],[],true);
Wykonano 77563 obrotow
oraz 12 przejsc po calej macierzy.
Funkcja wykonywala sie 2.912976e+01 sekund.
Matlab liczy to samo 4.148746e-02 sekund.
Norma z roznicy miedzy wektorami wynikowymi wynosi 1.227172e-09.
```

Figure 2: Losowo generowana macierz $A \in \mathbb{R}^{200\times 200}$

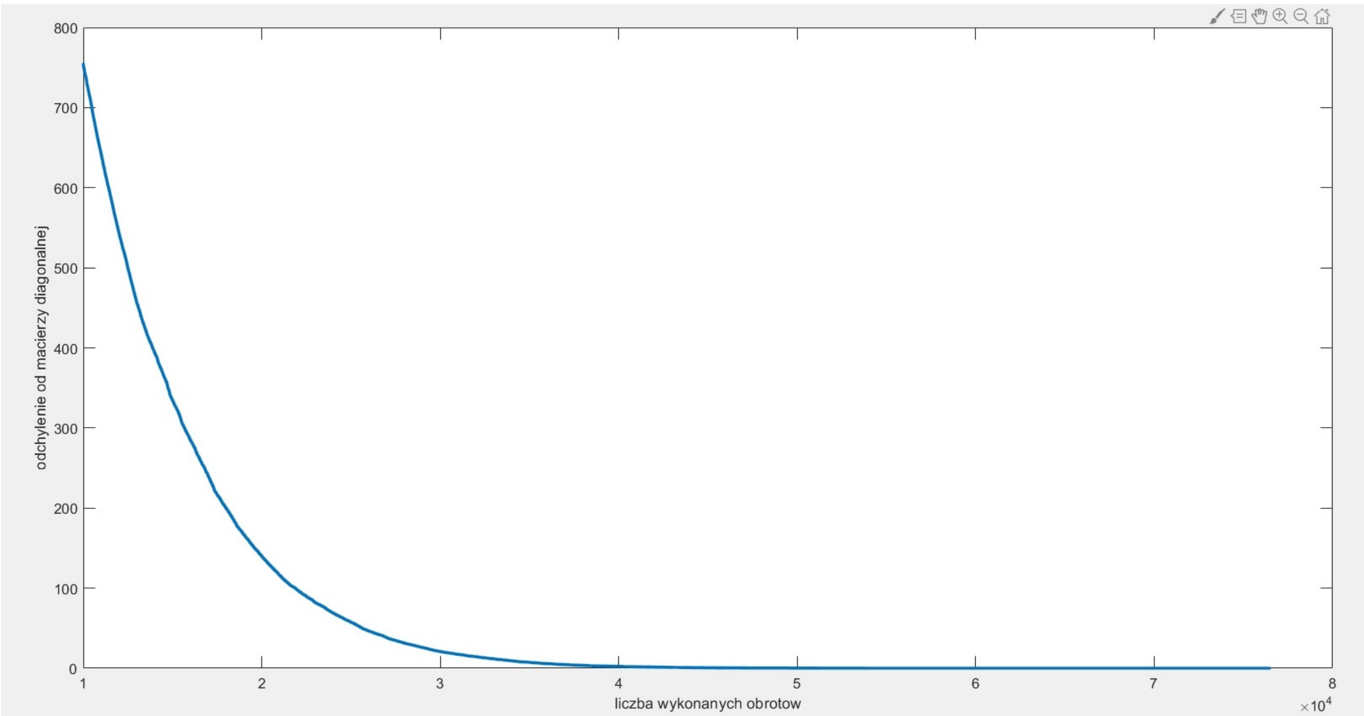


Figure 3: Wykres "Figure 2" ograniczony do argumentów ≥ 1000

5 Wnioski

Można zauważyć, że funkcja Jacobi jest znacznie wolniejsza od porównywanej z nią funkcji *eig*, będącej częścią programu MATLAB. Ponadto, w przykładzie drugim, dla czterokrotnie większej macierzy niż w przykładzie pierwszym, czas wykonywania funkcji był ponad 20 razy większy. Można wnioskować stąd że złożoność czasowa tej funkcji jest większa niż kwadratowa. Norma z różnicy między wynikami tych funkcji ($Jacobi$ oraz eig) jest rzędu 10^{-9}. Można powiedzieć zatem że funkcje w obu przypadkach testowych zwracają te same wyniki, co świadczy o poprawności implementacji funkcji $Jacobi$. Analizując wykresy odchylenia macierzy od macierzy diagonalnej w zależności od liczby wykonanych obrotów, zauważamy macierz bardzo szybko zbiega do macierzy diagonalnej. Widać to również po usunięciu z wykresu 1000 pierwszych argumentów.

References

[1] `http://mst.mimuw.edu.pl/lecture.php?lecture=mo2&part=Ch4`