Kacper Kula

Zadanie numeryczne nr 6

Omówienie

Problem polega na:

(a) użyciu metody potęgowej i znalezieniu największej co do modułu wartość własną macierzy M oraz odpowiadającemu jej wektora własnego i zilustrowaniu zbieżności metody w funkcji ilości wykonanych iteracji na wykresie

(b) użyciu algorytmu QR i znalezieniu wszystkich wartości własnych macierzy M oraz zademonstrowaniu, że macierze A_i upodabniają się do macierzy trójkątnej górnej w kolejnych iteracjach

(c) znalezieniu metody na usprawnienie zbieżności algorytmów

Macierz M jest zdefiniowana następująco:

$$M = \begin{bmatrix} 8 & 1 & 0 & 0 \\ 1 & 7 & 2 & 0 \\ 0 & 2 & 6 & 3 \\ 0 & 0 & 3 & 5 \end{bmatrix}$$

Narzędziem do napisania programu rozwiązującego zadanie jest język Python oraz biblioteka pythona służąca do tworzenia wykresów – matplotlib.
Do sprawdzenia poprawności programu wykorzystana została biblioteka z zakresu algebry liniowej - numpy.

Metoda potęgowa

Opis

Metoda Potęgowa stosowana jest do znajdywania wartości własnej o największym module i odpowiadającego jej wektora własnego.

Algorytm

Algorytm metody potęgowej ma następującą postać:

1. Wybranie wektora początkowego x_0 i znormalizowanie go $y_0 = \frac{x_0}{\|x_0\|}$.
2. Pomnożenie y_0 przez macierz M, otrzymany wektor x_1 normalizujemy $y_1 = \frac{x_1}{\|x_1\|}$
3. Powtarzamy ostatnią operację m razy, dopóki $y_m \approx y_{m-1}$, czyli $|y_m - y_{m-1}| \approx 0$
4. Wektor y_m jest naszym wektorem własnym, a wartość własną uzyskujemy z zależności $M * y_m = \lambda * y_m$

Algorytm QR

Opis

Algorytm QR służy do znajdywania wszystkich wartości własnych macierzy.
Generuje nam on ciąg macierzy A_k.
Jeżeli k zbiega do nieskończoności, to A_k zbiega do macierzy trójkątnej górnej

Algorytm

Algorytm QR ma następującą postać:

1. Wybieramy $A_1 = M = Q_1 * R_1$
2. Obliczamy następny element ciągu z wzoru: $A_2 = R_1 * Q_1$
3. Obliczamy $A_2 = Q_2 * R_2$.
4. Powtarzamy poprzednie kroki, dopóki macierz A_k nie jest podobna do macierzy trójkątnej górnej.
5. Elementy na diagonali powstałego A_k są naszymi wartościami własnymi.

Porównanie i sprawdzenie wyników

Przy pomocy funkcji linalg.eig() z biblioteki numpy obliczyłem wartości własne i wektory własne macierzy M, a następnie porównałem wyniki z tymi, które otrzymałem stosując opisane metody, z dokładnością do sześciu miejsc po przecinku.

Dla metody potęgowej wyszukałem największą co do modułu wartość własną z wartości własnych obliczonych przez numpy i porównałem z otrzymaną wartością przy użyciu metody potęgowej.
<u>Wartości własne okazały się równe.</u>
Znalazłem również wektor własny odpowiadający tej wartości i po odpowiednim przekształceniu porównałem z wektorem otrzymanym przy użyciu metody potęgowej.
<u>Wektory okazały się równe.</u>

Dla rozkładu QR porównałem wartości własne obliczone przez numpy z moimi wartościami otrzymanymi w ostatniej iteracji algorytmu.
<u>Wartości okazały się równe.</u>

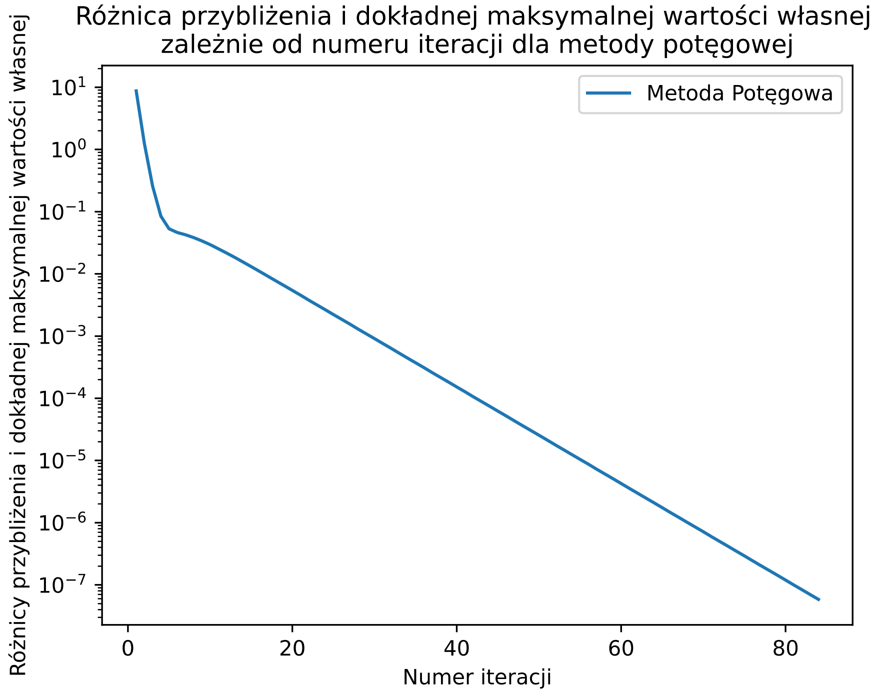
Finałowe wartości własne otrzymane z rozkładu QR mają postać:
$\lambda_1 = 9.742393758887875$
$\lambda_2 = 8.147717711696744$,
$\lambda_3 = 6.0317237108666495$
$\lambda_4 = 2.078164818548759$

Maksymalna wartość własna otrzymana z metody potęgowej wynosi $\lambda_{max} = 9.742393817254953$

Wartość $\lambda_{max} = 9.742393817254953$ odpowiada wektorowi własnemu postaci:
$v_{max} = (0.529334, 0.9223097, 1.0, 0.632591)^T$

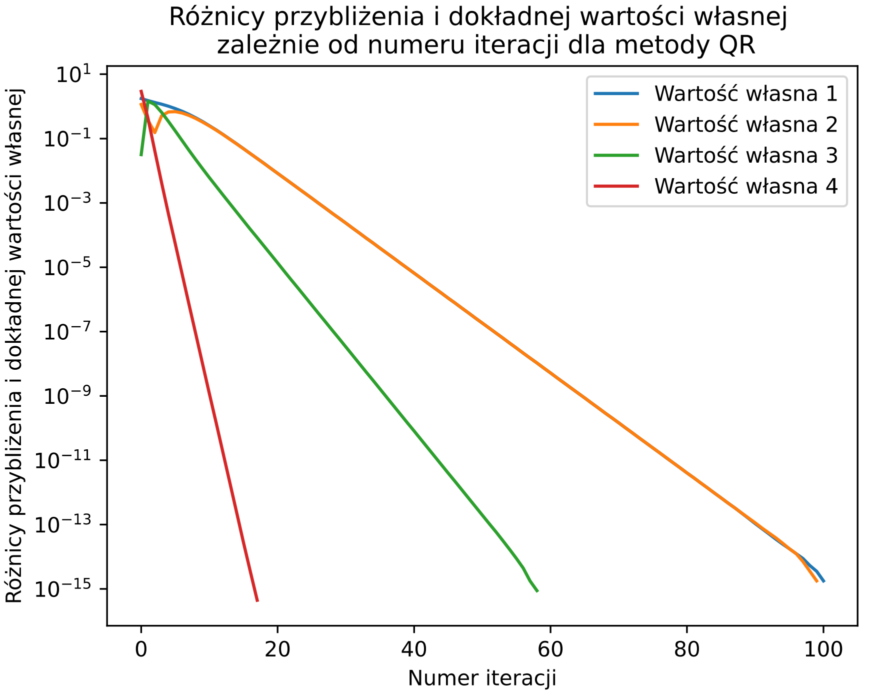
Wykresy

Wykres nr 1



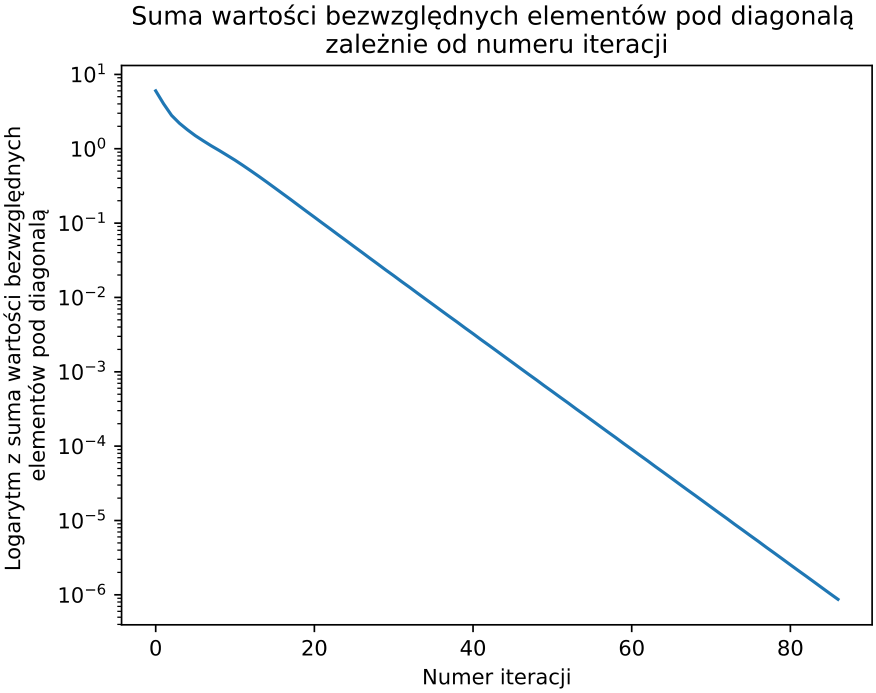
Wykres ilustruje zbieżność metody w funkcji ilości wykonanych iteracji dla metody potęgowej.

Wykres nr 2



Wykres ilustruje zbieżność każdej wartości własnych w funkcji ilości wykonanych iteracji dla rozkładu QR.

Wykres nr 3



Wykres ilustruje zbieżność sumy wartości absolutnych elementów macierzy A_k pod diagonalą do 0 w funkcji ilości wykonanych iteracji, co obrazuje zbieżność macierzy A_k do macierzy trójkątnej górnej.

Zbieżność algorytmu (a)

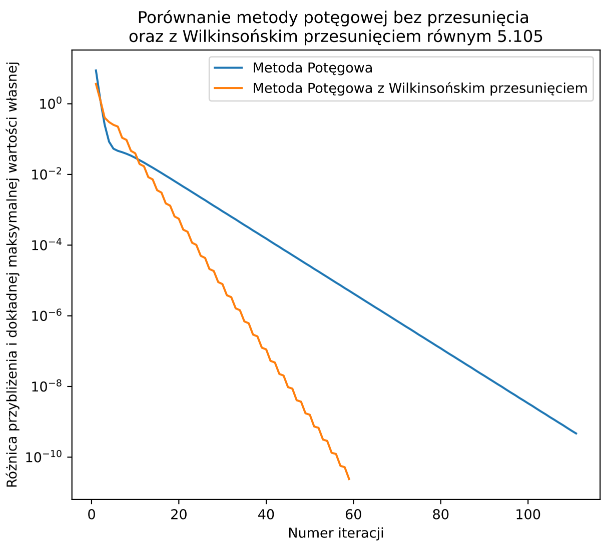
Algorytm metody potęgowej jest nieefektywny i jego zbieżność jest niezadowalająca.

Możemy w łatwy sposób usprawnić zbieżność naszej metody stosując Wilkinsońskie przesunięcie $A=p*I$, gdzie I to macierz jednostkowa, a p to wartość przesunięcia.

Należy jednak zwrócić uwagę na dobór odpowiedniej wartości p, gdyż wybór jej wpływa na zbieżność naszej metody – zarówno pozytywnie, jak i negatywnie.

Jak więc dobrać wartość p? Można pokazać, że najkorzystniejsza zbieżność osiągana jest dla: $p=\frac{1}{2}*(\lambda_2*\lambda_n)$, gdzie λ_2, λ_n to kolejno druga i ostatnia wartość własna macierzy.

Poprawnie dobrana wartość $p=\frac{1}{2}*(8.14*2.07)$ poprawiła naszą zbieżność.



Złe przesunięcie dla $p=-5$ prowadzi jednak do niekorzystnej zbieżności.