Sprawozdanie z zadania numerycznego 6

1. Problematyka: W zadaniu mamy porównać dwa algorytmy ze sobą, algorytm QR i metodę potęgową, pod względem liczby wykonanych iteracji podczas wyliczania wartości własnych macierzy.
   a) **Metoda potęgowa** jest jednym z najprostszych algorytmów do wyznaczania największej wartości własnej (co do modułu) macierzy oraz odpowiadającego jej wektora własnego.
   b) **Algorytm QR** jest bardziej zaawansowaną techniką, która pozwala na wyznaczenie wszystkich wartości własnych macierzy.
   c) Dokonujemy oceny zbieżności obu algorytmów oraz analizy szybkości obu algorytmów.

2. Teoria:

Podpunkt (a) odnosi się do metody potęgowej (ang. power method), która jest iteracyjną techniką numeryczną stosowaną do wyznaczania dominującej wartości własnej macierzy M*M* oraz odpowiadającego jej wektora własnego. Metoda ta opiera się na zasadzie, że wielokrotne mnożenie macierzy przez dowolny wektor x*x* (zazwyczaj zaczynamy od wektora jednostkowego) prowadzi do uzyskania wektora zbieżnego do wektora własnego odpowiadającego największej wartości własnej. W każdej iteracji k*k*, wektor xk*xk* jest aktualizowany według wzoru:

$$x = \frac{\mathrm{Mx_k}}{||\mathrm{Mx_k}a||}$$

gdzie ‖Mxkjest normą euklidesową wektora Mxk. Zbieżność metody jest sprawdzana przez porównanie różnicy między wartościami własnymi w kolejnych iteracjach z określoną tolerancją ϵ. Jeżeli różnica ta jest mniejsza niż ϵ, przyjmuje się, że algorytm osiągnął zbieżność.

- Podpunkt (b) dotyczy algorytmu QR, który jest uogólnieniem metody potęgowej i pozwala na wyznaczenie wszystkich wartości własnych macierzy, nie tylko tej dominującej. Algorytm ten w każdej iteracji dokonuje rozkładu macierzy M*M* na iloczyn macierzy ortogonalnej Q*Q* i macierzy trójkątnej górnej R*R*, a następnie mnoży je w odwrotnej kolejności:

$$M_k = Q_k R_k$$
$$M_{k+1} = R_k Q_k$$

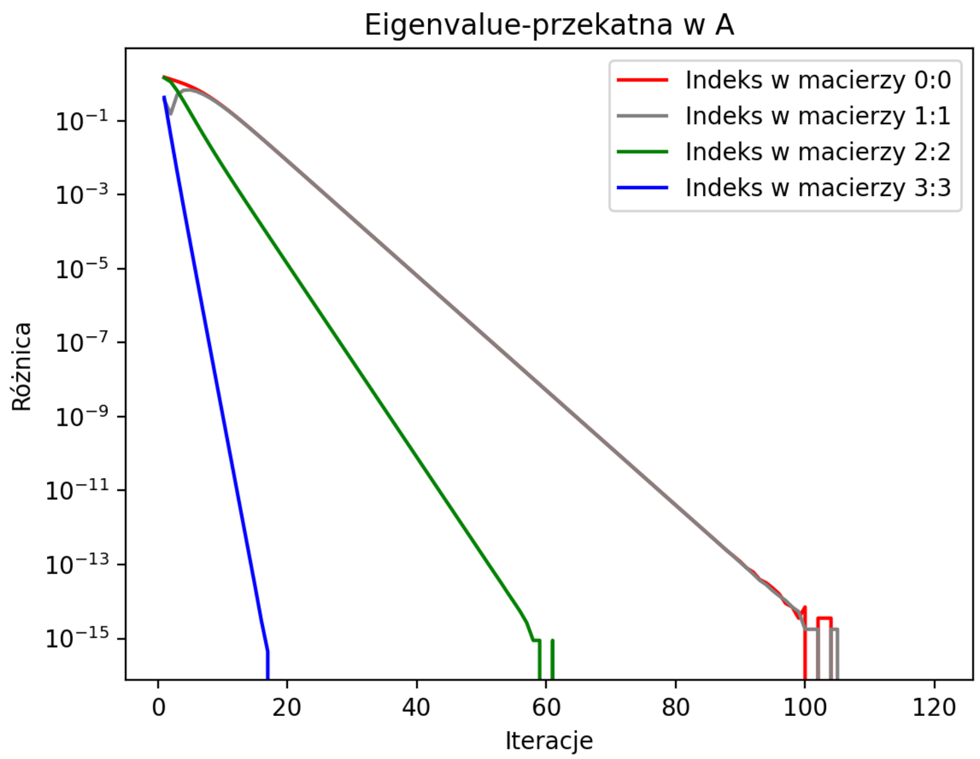
gdzie Mk*Mk* to macierz z k*k*-tej iteracji. Proces ten prowadzi do powstania nowej macierzy Mk+1*Mk*+1, która jest coraz bliższa macierzy trójkątnej górnej. Wartości własne macierzy M*M* są aproksymowane przez elementy na głównej przekątnej macierzy Mk*Mk* w kolejnych iteracjach. Zbieżność algorytmu QR jest zazwyczaj szybsza niż w metodzie potęgowej, ale nadal może wymagać dużej liczby iteracji w zależności od właściwości macierzy M*M*.

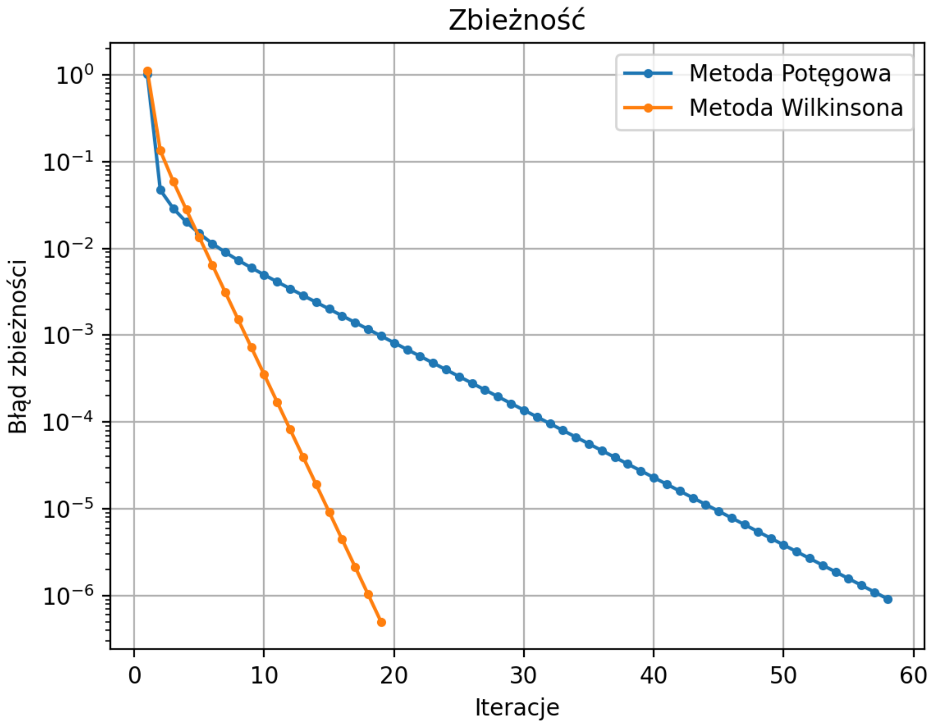
3. Rozwiązanie: Program zawiera trzy funkcje do obliczania wartości własnych macierzy:

- iter_power(mat, tol=1e-6): Implementuje metodę potęgową, która iteracyjnie znajduje dominującą wartość własną i odpowiadający jej wektor własny.
- QR_algo(mat): Implementuje algorytm QR, który iteracyjnie przekształca macierz za pomocą rozkładu QR, zwracając listę błędów i obliczone wartości własne.
- Wilkinson_algo(mat, tol=1e-6): Jest modyfikacją metody potęgowej, która wykorzystuje algorytm Wilkinsona do obliczenia wartości własnych, wykonując przesunięcie spektralne.

Program definiuje macierz mat_A i wykonuje obliczenia przy użyciu powyższych funkcji. Wyniki są ilustrowane na wykresach generowanych za pomocą biblioteki Matplotlib. Funkcje pomocnicze, takie jak mat_mult i normalize, są używane do uproszczenia głównych funkcji algorytmów.

W przykładzie a) nasza dominująca wartość własna= 9.742393758842107, której towarzyszy taki wektor własny: [0.3327265892506157, 0.579734285789243, 0.628566072807256, 0.3976252844095202]
Dla b wartości własne mają postać : Wartości własne: [9.74239376 8.14771771 6.03172371 2.07816482].





Jak widać metoda Wilkinsona radzi sobie z naszym przykładem o wiele lepiej, niż metoda potęgowa. Jest trzykrotnie szybsza.

4. Wnioski: W obu przypadkach, zarówno dla metody potęgowej, jak i algorytmu QR, ważnym aspektem jest analiza szybkości zbieżności oraz dokładności wyników i jak możemy nietrudno zobaczyć metoda Wilkinsona radzi sobie znacznie lepiej. Jak widać na rysunku „Zbieżność" pomimo tego, że obydwie metody są linowe w ilości iteracji, widzimy, że między prawdziwą złożonością a O(n) jest olbrzymia różnica.