SPRAWOZDANIE METODY NUMERYCZNE

LABORATORIUM NR 5 – METODA POTĘGOWA Z ORTOGONALIZACJĄ GRAMMA-SCHMIDTA

Aleksandra Krzemińska,

Informatyka Stosowana II rok WFiIS, I stopień,

rok 2021

WSTĘP TEORETYCZNY

Tematem przewodnim bieżących zajęć laboratoryjnych była metoda potęgowa wykorzystana w celu wyznaczenia wektorów własnych dla danej maicerzy. Ta iteracyjna metoda wyznaczania wektorów własnych jest wysoce skuteczna przy wyznaczaniu dominującej wartości własnej i wektora do niej przynależnego, dlatego też stosowana jest w algorytmie PageRank - algorytmie pozycjonującym, który wykorzystywany jest przez popularną wyszukiwarkę internetową Google. Metoda implementowana w ramach zajęć została lekko zmodyfikowana - wykorzystuje ortogonalizację Gramma-Schmidta, jest to operacja mająca na celu przekształcenie wektorów unitarnych w układ wektorów ortogonalncyh (tj. kiedy iloczyn skalarny wektorów jest równy 0). Dzięki temu macierz wejściowa nie będzie modyfikowana – zastąpione zostanie to jedynie operacją ortogonalizacji wektorów. W przypadku obliczania pozostałych wartości własnych i wektorów własnych (niedominujących), metoda ta jest już mniej dokładna.

ZADANIE DO WYKONANIA

Zadaniem laboratoryjnym było zaimplementowanie ww. metody potęgowej, z wykorzystaniem ortogonalizacji Gramma-Schmidta, wyliczenie wartości własnych i wektorów własnych dla macierzy symetrycznej **A** (rozmiar n = 7), zdefiniowanej jako:

$$A_{ij} = \frac{1}{\sqrt{2 + |i - j|}} \ ,$$

zgodnie z poniższym algorytmem:

$$
\begin{aligned}
&for(k = 0;\ k < K_{val};\ k++)\{ \\
&\quad \boldsymbol{x}_k^0 = [1, 1, \ldots, 1] \qquad \text{(inicjalizacja wektora startowego)} \\
&\quad for(i = 1; i <= IT\_MAX; i++)\{ \\
&\qquad \boldsymbol{x}_k^{i+1} = A\boldsymbol{x}_k^i \\
&\qquad for(j = 0;\ j < k;\ j++)\{ \text{ (ortogonalizacja G-S)} \\
&\qquad\quad \boldsymbol{x}_k^{i+1} = \boldsymbol{x}_k^{i+1} - \left[\left(\boldsymbol{x}_k^{i+1}\right)^T \boldsymbol{x}_j\right] \boldsymbol{x}_j \\
&\qquad \} \\
&\qquad \lambda_k^i = \frac{\left(\boldsymbol{x}_k^{i+1}\right)^T \boldsymbol{x}_k^i}{\left(\boldsymbol{x}_k^i\right)^T \boldsymbol{x}_k^i} \\
&\qquad \boldsymbol{x}_k^i = \frac{\boldsymbol{x}_k^{i+1}}{\|\boldsymbol{x}_k^{i+1}\|_2} \\
&\quad \} \\
&\}
\end{aligned}
$$

gdzie: k – to numer wyznaczanej wartości własnej, **A** – macierz wejściowa, λ^i_k – przybliżenie k-tej wartości w i-tej iteracji, x^i_k – przybliżenie k-tego wektora własnego, IT_MAX – maksymalna liczba iteracji dla każdego k (przyjęta jako 12).

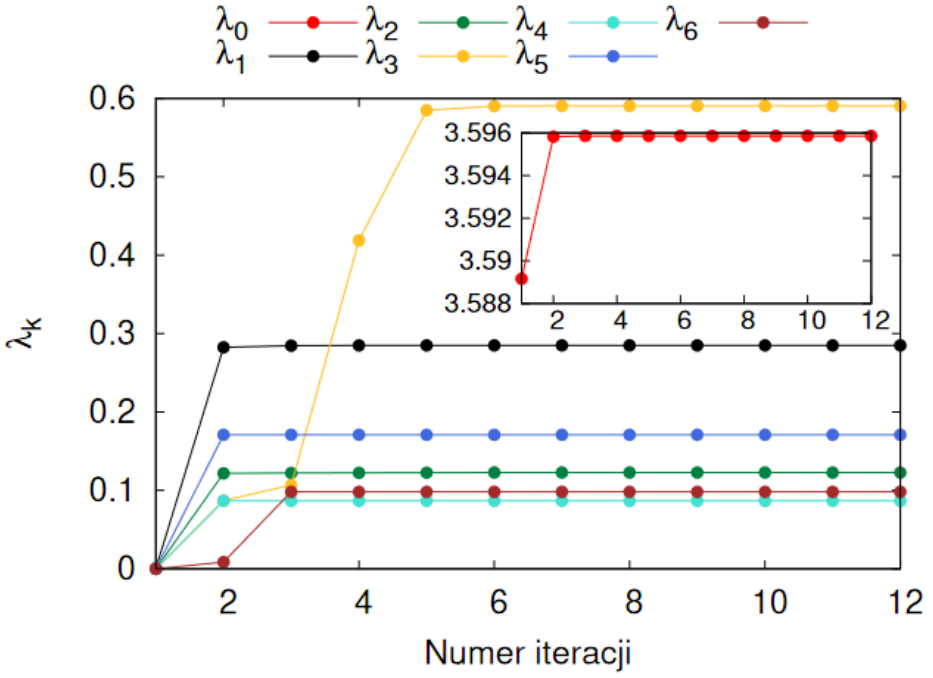
Następnie należało utworzyć macierz **X** zawierającą zbiór wszystkich wyliczonych wektorów własnych oraz wyliczyć macierz **D**, zdefiniowaną jako iloczyn macierzowy $\mathbf{D} = \mathbf{X}^T\mathbf{A}\mathbf{X}$.

METODA

W celu poprawienia czytelności kodu zaimplementowane zostały samodzielnie funkcje pomocnicze, m.in. mnożenie macierzy, mnożenie wektorów, mnożenie macierzy z wektorem, mnożnie przez skalar, czy funkcja transpozycji macierzy. Wyniki własności własnych zostały wpisane do osobnego pliku tekstowego, tak samo wynik mnożenia macierzowego – macierz D. W ramach ułatwienia alokacji pamięci dla macierzy i wektorów wykorzystana została, już wcześniej wykorzystywana na laboratoriach, biblioteka GSL.

WYNIKI

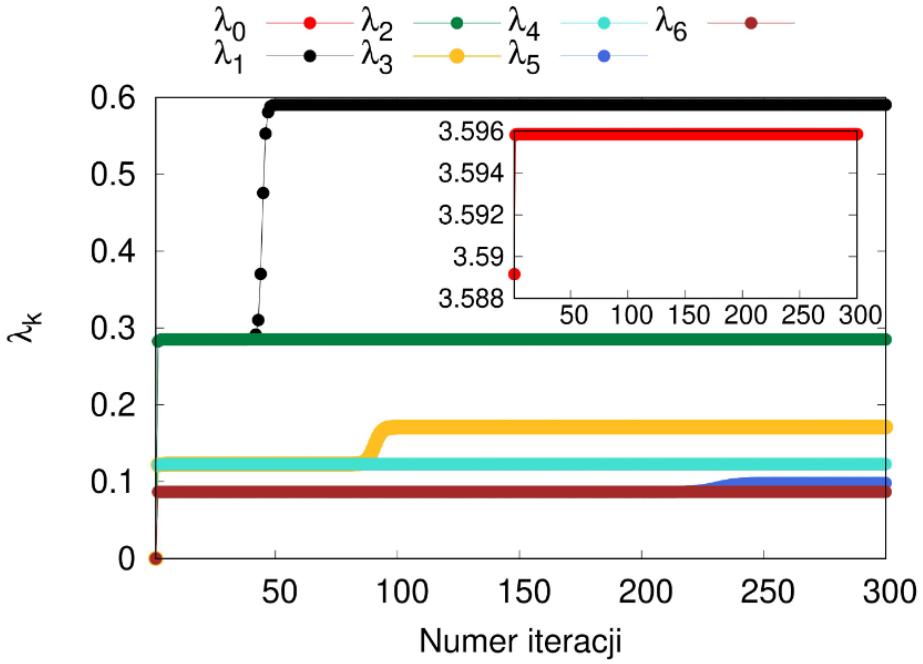
Wyniki zostały zaprezentowane w sposób graficzny przy pomocy skryptu w *GnuPlot,* jak poniżej:



Wykres 1: Graficzna prezentacja kolejnych przybliżeń znalezionych wartości własnych λ_k w kolejnych iteracjach algorytmu

Przedstawiony powyżej *Wykres 1* pokazuje wcześniej już wspomniane właściwości – widać, że wartość własna λ_0 jest znacząco większa od pozostałych (dominująca). Dodatkowo jej wartość już przy drugiej iteracji była określona na tyle dokładnie, że kolejne iteracje nie zmieniły wyniku, w przeciwieństwie do λ_3 czy λ_6, gdzie widać, iż te wartości ustabilizowały się dopiero przy większej ilości iteracji.

W ramach weryfikacji rozszerzona została dziedzina (liczba iteracji) dla algorytmu, tak aby zweryfikować, czy dokładność wartości nie zmieni się przy większej ilości przybliżeń. Poniższy *Wykres 2* pokazuje tę samą prezentację przybliżeń wartości własnych, ale tym razem dla 300 iteracji.



Wykres 2: Graficzna prezentacja kolejnych przybliżeń znalezionych wartości własnych λ_k dla 300 iteracji

Jak widać powyżej, dla wartości dominującej wartość nawet przy 300 iteracjach nie uległa zmianie, w przeciwieństwie do niektórych z pozostałych wartości własnych. Co za tym idzie – jest to swego rodzaju dowód, na mniejszą dokładność wyników wartości własnych niedominujących i im odpowiadających wektorów własnych i mniejszą efektywność algorytmu przy ich wyliczaniu.

Wartości dla macierzy **D** z kolei prezentują się następująco:

$$\mathbf{D} = \begin{pmatrix} \mathbf{3.59586} & -8.99281e{-15} & -1.66533e{-16} & -5.55112e{-16} & -3.33067e{-16} & 1.11022e{-16} & -2.77556e{-16} \\ -8.89566e{-15} & \mathbf{0.284988} & -1.88273e{-06} & -9.6575e{-09} & -1.57912e{-09} & 3.46945e{-17} & -6.93889e{-18} \\ -2.28983e{-16} & -1.88273e{-06} & \mathbf{0.122798} & -0.00240002 & -0.000136113 & -2.50361e{-12} & -2.94903e{-17} \\ -6.93889e{-17} & -9.6575e{-09} & -0.00240002 & \mathbf{0.590378} & -1.13699e{-08} & 2.77556e{-17} & -5.55112e{-17} \\ -2.71484e{-16} & -1.57912e{-09} & -0.000136113 & -1.13699e{-08} & \mathbf{0.0865952} & -1.60842e{-09} & -7.29798e{-15} \\ 2.70617e{-16} & -4.85723e{-17} & -2.50361e{-12} & -1.38778e{-17} & -1.60842e{-09} & \mathbf{0.170974} & -4.06949e{-09} \\ -5.55112e{-17} & -8.67362e{-18} & -7.45931e{-17} & -9.19403e{-17} & -7.26112e{-15} & -4.06949e{-09} & \mathbf{0.0981544} \end{pmatrix}$$

Na pierwszy rzut oka widać, że wartości na diagonali są znacząco większe od pozostałych – można stwierdzić, że wartości macierzy poza diagonalą są równe prawie zero. Zaobserwować można też, iż wartości na diagonali są tożsame z wcześniej obliczonymi, kolejnymi wartościami własnymi dla macierzy **A**.

WNIOSKI

Metoda potęgowa zaimplementowana w ramach laboratorium jest bardzo efektywna. W przypadku obliczania dominującej wartości własnej wystarczyło zaledwie kilka iteracji do znalezienia bardzo dokładnego wyniku, z czego wynika bardzo szybki czas wykonania algorytmu. Mankamentem jednak jest mniejsza dokładność obliczeniowa dla wartości własnych, które nie są dominujące – wówczas algorytm ten nie jest już tak niezawodny i wymaga dłuższego czasu (większej ilości iteracji) do znalezienia akceptowalnego rezultatu.