// Autor: Daniel Szarek
//====================

Zawartość folderu z zadaniem: main.cpp – Kod z rozwiązaniem zadania napisanym w języku C++
funkcje.h – Biblioteka z własnymi funkcjami wykorzystywanymi do rozwiązania zadania N8
opracowanie.pdf – Ten dokument z opracowaniem zadania N8 funkcje.cpp – Plik zawierający implementację funkcji podanych w własnej bibliotece funkcje.h Makefile – Do uruchomienia programu main.cpp (Makefile oferuje możliwość uruchomienia programu za pomocą komendy 'make run', usunięcia plików po uruchomieniu programu komendą 'make clean' oraz możliwość utworzenia paczki .tar.gz z zawartością foldera z zadaniem za pomocą komendy 'make tar'
wyniki.txt – Plik tekstowy z wynikami programu main.cpp

Do zrealizowania zadania wykorzystałem GSL - GNU Scientific Library, w Makefile zawarłem komendy do uruchomienia zadania na swojej maszynie w systemie Ubuntu, w celu uruchomienia zadania na swoim sprzęcie należy w prawidłowy sposób zainstalować bibliotekę GSL oraz zmienić scieżkę INCLUDEPATH1 dla swojej maszyny.

W swoich plikach źródłowych zamieszczam liczne komentarze, które na bieżąco tłumaczą działanie oraz funkcjonalności mojego kodu. W tym dokumencie przedstawię omówienie metod wykorzystanych do rozwiązania zadania.

Treść Zadania:

N8 *Zadanie numeryczne*
Znaleźć wartości własne macierzy z dokładnością 10^{-8}

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & -1 \end{pmatrix}, \tag{8}$$

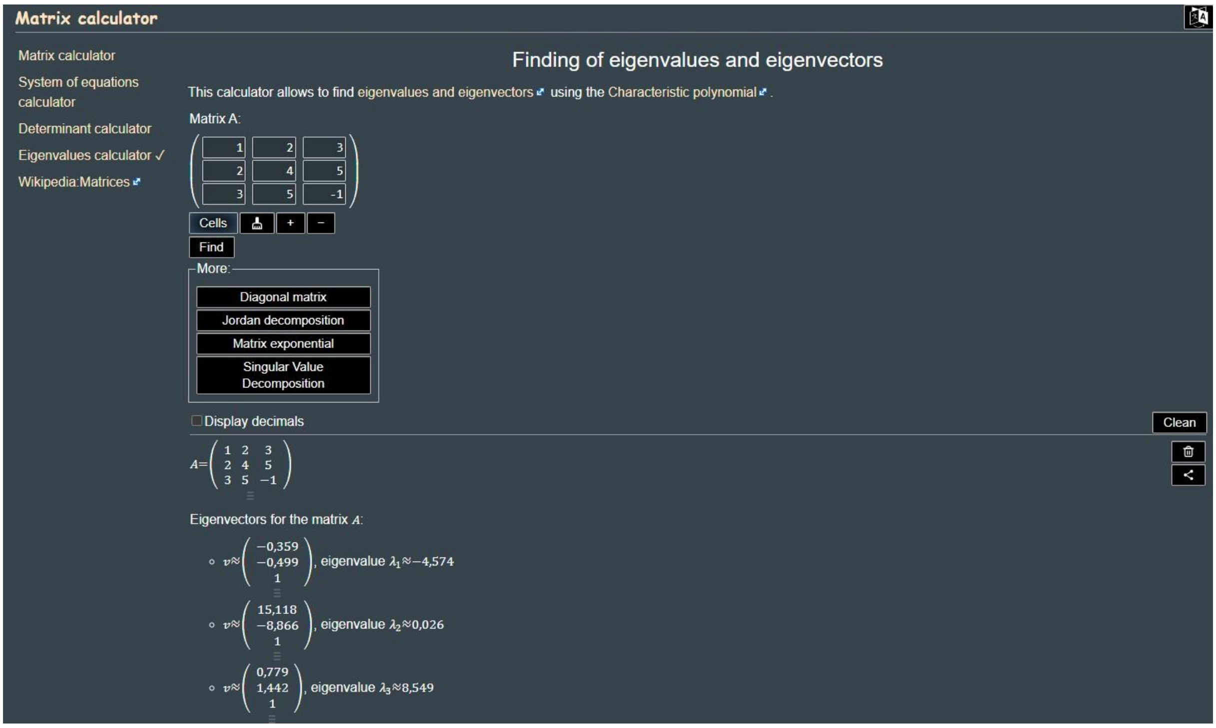
korzystając z metody potęgowej, Rayleigha i metody iteracyjnej QR:

$$\begin{aligned} B^{(0)} &= A, \\ Q^{(n)}R^{(n)} &= B^{(n)}, \\ B^{(n+1)} &:= R^{(n)}Q^{(n)}. \end{aligned} \tag{9}$$

Uwaga: Metoda Rayleigha będzie omówiona na zajęciach, ale można też oprzeć się o sekcję 3 https://mathreview.uwaterloo.ca/archive/voli/1/panju.pdf.

Rozwiązanie zadania:

```
vboxuser@Ubuntu:~/Desktop/MetodyNumeryczne/Zadanie8$ make run
g++ -Wall -std=c++11 -I/home/vboxuser/Desktop/gsl/include -c main.cpp -o main.o
g++ -Wall -std=c++11 -I/home/vboxuser/Desktop/gsl/include -c funkcje.cpp -o funkcje.o
ar rsv libMojeFunkcje.a funkcje.o
ar: creating libMojeFunkcje.a
a - funkcje.o
mkdir -p ./lib
mv libMojeFunkcje.a ./lib
g++ -o main.x main.o -Wall -std=c++11 -L./lib -lMojeFunkcje -lgsl -lgslcblas
./main.x
Wartosci wlasne macierzy A uzyskane dzieki metodzie potegowej:
Î»[0] = 8.5485128497
Î»[1] = 4.5740872277
Î»[2] = 0.0255743726
Wartosci wlasne macierzy A uzyskane dzieki metodzie Rayleigha:
Î»[0] = 8.5485128532
Î»[1] = -4.5740872259
Î»[2] = 0.0255743726
Wartosci wlasne macierzy A uzyskane dzieki metodzie iteracyjnej QR:
Î»[0] = 8.5485128520
Î»[1] = -4.5740872246
Î»[2] = 0.0255743726
vboxuser@Ubuntu:~/Desktop/MetodyNumeryczne/Zadanie8$
```



Wyniki dla metod: Rayleigha oraz Iteracyjnej QR powielają się z tym wskazanym przed stronę https://matrixcalc.org/ powyżej. Metoda potęgowa zwróciła inny wynik przy wartości własnej na minusie, zwróciła dokładnie jej przeciwieństwo, dlaczego tak jest powiem dalej w opracowaniu.

Omówienie:

Swoją implementację metod podanych w poleceniu przygotowałem na podstawie źródła podanego w poleceniu poniżej wstawiam zdjęcia algorytmów na, których bazowałem.

Metoda Potęgowa:

The method of power iteration can then be stated as follows:

Pick a starting vector $v^{(0)}$ with $\|v^{(0)}\| = 1$
For $k = 1, 2, \ldots$
    Let $w = Av^{(k-1)}$
    Let $v^{(k)} = w / \|w\|$

Metoda potęgowa służy do znalezienia największej wartości własnej macierzy oraz odpowiadającego jej wektorowi własnemu. Poprzez zastosowanie deflacji, można również odnaleźć kolejne największe wartości własne, co uczyniłem w swoim programie. Wadą tej metody jest brak informacji o znaku wartości własnej, ponieważ operujemy jedynie na modułach. W każdej iteracji metoda zbliża się coraz bardziej do poszukiwanej wartości własnej. Warto zaznaczyć, że zbieżność tej metody jest zagwarantowana tylko w przypadku unikalnych wartości własnych. Metoda ta skutecznie działa dla dużych macierzy.

Metoda potęgowa jest szczególnie skuteczna, gdy dominująca wartość własna ma większy moduł niż pozostałe. Jednak może być wolna w zbieżności i nie zawsze działa dobrze w przypadku macierzy, które są bliskie macierzom osobliwym. Warto jednak zaznaczyć, że metoda ta może być wrażliwa na wybór początkowego wektora estymacyjnego, a zbieżność może być wolniejsza w przypadku blisko siebie leżących wartości własnych.

Metoda Rayleigha:

Pick a starting vector $v^{(0)}$ with $\|v^{(0)}\| = 1$
Let $\lambda^{(0)} = r(v^{(0)}) = (v^{(0)})^T A(v^{(0)})$
For $k = 1, 2, \ldots$
    Solve $(A - \lambda^{(k-1)} I)w = v^{(k-1)}$ for w
    Let $v^{(k)} = w / \|w\|$
    Let $\lambda^{(k)} = r(v^{(k)}) = (v^{(k)})^T A(v^{(k)})$

Metoda Rayleigha to rodzaj ulepszenia odwróconej metody potęgowej z przesunięciem. Zamiast stosować stałe przesunięcie, używa się przybliżonej wartości własnej jako przesunięcia. Ta metoda umożliwia znalezienie dowolnej wartości własnej, jeśli dostarczymy wystarczająco bliskie przybliżenie. Co istotne, metoda ta charakteryzuje się znacznie szybszą zbieżnością, która postępuje w przybliżeniu w tempie kwadratowym. Dlatego metoda Rayleigha jest efektywna, zwłaszcza gdy łączymy ją z innymi metodami: najpierw znajdujemy przybliżenie, a następnie szybko szukamy dokładnej wartości własnej. Wykorzystałem to w rozwiązaniu zadania N9 łącząc metodę Rayleigha z odwróconą metodą potęgową z deflacją wektorową.

W swojej implementacji metody Rayleigha użyłem deflacji na poziomie macierzy. Jestem świadom, że nie jest to rozwiązanie uniwersalne i może ono nie działać w przypadku niektórych macierzy, niemniej jednak na potrzebę naszego zadania i małej macierzy 3 na 3 rozwiązanie to sprawdza się dobrze.

Metoda Rayleigha jest skuteczna, gdy chcemy uzyskać przybliżone wartości własne macierzy, zwłaszcza dominującej wartości własnej. Warto jednak zaznaczyć, że metoda ta może być wrażliwa na wybór początkowego wektora estymacyjnego, a zbieżność może być wolniejsza w przypadku blisko siebie leżących wartości własnych.

Metoda Iteracyjna QR:

```
Let A^(0) = A
For k = 1, 2, ...
    Obtain the factors Q^(k) R^(k) = A^(k-1)
    Let A^(k) = R^(k) Q^(k)
```

Metoda iteracyjna QR różni się od wcześniejszych metod, ponieważ pozwala jednocześnie znaleźć wszystkie wartości własne. Nazwa sugeruje, że metoda korzysta z rozkładu QR do tworzenia pomocniczych macierzy Q i R, a następnie mnoży je w odwrotnej kolejności. Taka kombinacja sprawia, że po pewnej liczbie iteracji macierz Q zbiegnie do macierzy jednostkowej, a na diagonali macierzy R pojawią się wartości własne. Można także otrzymać wektory własne, mnożąc macierz Q z każdej iteracji przez siebie.

Metoda iteracyjna QR pozwala na znalezienie przybliżonych wartości własnych macierzy, jednocześnie radząc sobie z wszystkimi wartościami własnymi. W praktyce jest stosowana, gdy zależy nam na efektywności i przybliżeniu wartości własnych, a nie na dokładności.