Sprawozdanie MNUM
Projekt nr.3
Zadanie 3.32

Kajetan Kaczmarek

2 maja 2018

1. Opis zastosowanych algorytmów :

   (a) W pierwszym zadaniu, tj. znalezienie zer dla funkcji

   $$f(x) = 0.5xcos(x) - ln(x)$$

   użyłem dwóch metod. Założeniami dla obywdu metod była a) ciągłość, co jest oczywiste dla ww. funkcji, oraz b) różne znaki na krańcach przedziału, do czego odnoszę się poniżej. Zastosowane metody :

   - Metoda bisekcji
     W metodzie bisekcji na początek liczony jest punkt wypadający pomiędzy podanymi wejściowymi punktami, tj. $x = \frac{a+b}{2}$ dla p. wejściowych a i b. Następnie sprawdzamy czy punkt ten jest naszym zerem z podaną dokładnością eps, czyli czy $|f(x)| < eps$. Jeśli tak jest kończymy wykonywanie algorytmu, jeśli nie to sprawdzamy warunek $f(a)f(b) < 0$ i w zależności od wyniku zastępujemy lewy lub prawy koniec przedziału w którym szukamy wyliczonym x, tak, aby krańce przedziału nadal miały przeciwne znaki. Alternatywnym warunkiem wyjściowym z pętli jest $|a - b| < eps$, czyli zbliżenie się do siebie punktów a i b tak, że dalsze obliczenia są niemożliwe.
   - Metoda Siecznych
     Metoda siecznych jest podobna do metody bisekcji - szukamy zer przez zawężanie zakresu poszukiwań, warunki końcowe są więc takie same.Różny jest jednak algorytm wyznaczania kolejnego punktu : tutaj kolejne punkty wyznaczamy ze wzoru

     $$x_i = x_{i-1} - f(x_{i-1})\frac{x_{i-1} - x_{i-2}}{f(x_{i-1}) - f(x_{i-2})}$$

Tak że łączenie kolejnych punktów daje nam sieczne naszej funkcji f(x) i przybliża jej zera.

- Uwaga techniczna - założeniami obydwu metod są różne znaki funkcji na krańcach przedziału. Jako że warunek ten nie jest spełniony dla zadanego przedziału w mojej funkcji, a do tego ww. metody znajdują tylko jedno zero, podzieliłem zadany przedział [2,11] na dwa mniejsze , tj. [2,7] i [7,11] tak aby w każdym znajdowało się jedno zero, i aby spełniały one założenia metod.

2. Kod moich programów

- Funkcja main dla pierwszego zadania

```
xb1 = bisect(11,7,10e-5); %Pierwsze zero funkcji metoda bisekcji
xb2 = bisect(7,2,10e-5); %Drugie zero funkcji metoda bisekcji

xs1 = secants(11,7,10e-5); %Pierwsze zero funkcji metoda siecznych
xs2 = secants(7,2,10e-5); %Drugie zero funkcji metoda siecznych

Ans = [xb1,xb2,xs1,xs2]; % Wektor z rozwiazaniami dla ulatwienia
    rysowania

X = 2:0.1:11; % Nasz przedzial z probkowaniem co 0.1
Y = arrayfun(@(x) fzad(x),X); %Wyliczenie wartosci naszej funkcji do
    narysowania wykresu

figure; % Ponizszy kod rysuje wykres
plot(X,Y);% Rysowanie wykresu funkcji
hold on;
plot(Ans,0, 'o','MarkerEdgeColor','b','MarkerFaceColor'
    ,[0.5,0.5,0.5]); % Rysowanie miejsc zerowych
hold off;
```

- Pomocnicza funkcja licząca wartości naszej funkcji podanej dla zadania

```
function f = fzad(x) % Funkcja pomocnicza obliczajaca wartosci funkcji
    podanej w zadaniu
f = 0.5*x*cos(x) - log(x);
```

- Funkcja licząca zera funkcji metodą bisekcji

```
function [x]=bisect(a,b,eps) % Funkcja realizujaca metode bisekcji

while abs( a - b ) > eps % Warunek koncowy - punkty pomiedzy ktorymi
    liczymy zblizyly sie na odleglosc abs
    x = ( a + b ) / 2 ; % Policzenie punktu pomiedzy p. wejsciowymi
    if( abs( fzad(x) ) < eps) % Warunek koncowy - policzenie zera z
        dokladnoscia eps
        break;
    else
        if(fzad(x) * fzad(a) <0) % W zaleznosci od tego z ktorej
            strony wypadl punkt x podmieniamy wartosc a lub b na x
            b = x;
        else
            a = x;
        end
    end
end
```

- Funkcja licząca zera funkcji metodą siecznych

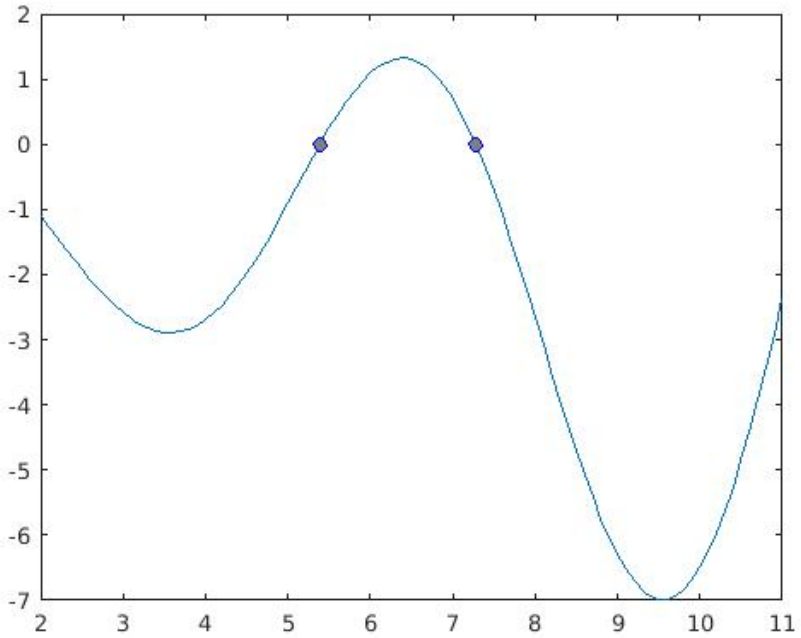
```
function x = secants(x1,x2,eps) % Funkcja realizujaca metode siecznych

```

```
3   if(abs(fzad(x2)-fzad(x1))<eps) % Sprawdzenie czy punkty wejsciowe nie
        leza zbyt blisko
4       error("Zle punkty wyjsciowe");
5       return;
6   end
7
8   f1 = fzad(x1); % Policzenie wartosci funkcji w p. wejsciowych
9   f2 = fzad(x2);
10
11  if(f1<f2) % Zamienienie miejscami punktow jesli wartosc w x1 jest
        mniejsza niz w x2
12      a = x1;
13      x1= x2;
14      x2 = a;
15      f1 = fzad(x1);
16      f2 = fzad(x2);
17  end
18
19  while abs(x2-x1)>eps % Warunek koncowy - jesli punkty pomiedzy ktorymi
        liczymy zbliza sie do odleglosci eps
20      x = x1 - (f1*(x1-x2))/(f1-f2); % Obliczenie nastepnego punkty
21      f = fzad(x);
22
23      if(abs(f) < eps) % Warunek koncowy -  znalezienie zera z
            dokladnoscia do eps
24          break;
25      end
26
27      x2 = x1; % Zmiana na punkty dla nastepnej iteracji
28      f2 = f1;
29      x1 = x;
30      f1 = f;
31  end
```

3. Wyniki :

   - Dla zadania pierwszego obydwie metody zwróciły zbliżone wyniki, tj.

Metoda	Zero nr.1	Zero nr. 2
Metoda Bisekcji	7.27703857421875	5.38775634765625
Metoda Siecznych	7.27702154631274	5.38773923503257



4. Wnioski :