Pierwszy projekt z labu - metoda Steffensena 2019-20

Zaprogramować funkcję octave'a z testami ***metody Steffensena*** zdefiniowanej wzorem

$$x_{n+1} = x_n - \frac{f(x_n)^2}{f(x_n + f(x_n)) - f(x_n)}$$

- która ma znaleźć przybliżenie x^* rozwiązania $f(x^*) = 0$ oraz drukować na ekranie $|e_{n+1}|/|e_n|^p$ dla $p = 1, 2, 3$ i $e_n = x_n - x^*$. Jako warunek stopu przyjąć $|f(x_n)| < tola + tolr * |f(x_0)|$ lub to, że ilość iteracji przekroczyła zadaną liczbę $Nmax$.

Dokładnie napisać funkcję octave'a w m-pliku steff.m:

function [y,it,fx,e]=steff(FCN, x0, sol, tolr, tola, Nmax)

z parametrami wejściowymi:

- FCN - 'wskaźnik' do funkcji f (function handle),
- $x0$ - przybliżenie startowe,
- sol - rozwiązanie równania x^*
- $tola, tolr$ - tolerancje błędu
- $Nmax$ - maksymalna ilość iteracji.

Funkcja ma zwracać:

- y - obliczone pierwiastek
- it - ilość iteracji.
- fx - wektor zawierający kolejne $f(x_n)$.
- e - wektor zawierający błędy tj. $e(n) = e_n$.

W razie gdyby ilość iteracji przekroczyła $Nmax$ funkcja powinna zwrócić komunikat o tym na ekran.

Funkcja ma działać również przy podaniu tylko trzech lub czterech zmiennych, wtedy odpowiednie parametry mają przyjąć następujące wartości : $tola = tolr * 100 = 10^{-9}$ i $Nmax = 100$.

Przetestować rząd zbieżności tej metody dla następujących równań:

1. $f_1(x) = x^k - 2^k = 0$ dla $k = 2, 8$ (zero jednokrotne)
2. $f_2(x) = (2 + \sin(x)) * (x - 2)^p = 0$ dla $p = 2, 8$ i $x_0 = 2.1, 1.8, 3$ (zero wielokrotne)
3. $f_3(x) = \sin(x^* - 2)$ dla $x_0 = 2.1, 3, 1.99$. ($f_3'(x^*) <> 0 = f_3''(x^*)$ - metoda Newtona ma wtedy zbieżność kubiczną)

Przy testowaniu mogę poprosić o inne x_0 lub równanie.