Paweł Maśluch

[MNUM] Metody Numeryczne

Semestr 2021L

Projekt 3

Sprawozdanie

Treść zadania.

1. Proszę znaleźć wszystkie zera funkcji $f(x) = 1.4 \cdot \sin(x) - e^x + 6 \cdot x - 0.5$ w przedziale $[-5, 5]$, używając dla każdego zera programu z implementacją:
   a) metody bisekcji
   b) metody Newtona

2. Używając metody Müllera MM2, proszę znaleźć wszystkie pierwiastki wielomianu czwartego stopnia

$$f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0, \quad [a_4\ a_3\ a_2\ a_1\ a_0] = [1\ \ 3\ -8\ \ 4\ \ 2].$$

Uwagi:

I. Implementacje algorytmów mają być w postaci solwerów;
II. Podczas testów należy wybierać szerokie przedziały startowe (lub punkty startowe znacznie oddalone od zer funkcji), dopiero w razie potrzeby należy te przedziały odpowiednio modyfikować;
III. Żeby znaleźć wszystkie pierwiastki wielomianu, zastosować deflację czynnikiem liniowym (bez tego ocena niższa o 1p.);
IV. We wnioskach dotyczących p. 1 powinna znaleźć się odpowiedź na pytanie: czy i kiedy dana metoda może zawieść i dlaczego?

Sprawozdanie powinno zawierać:
- krótki opis zastosowanych algorytmów (w tym najważniejsze wzory),
- listing dobrze skomentowanych programów w Matlabie z implementacją użytych algorytmów,
- przybliżony wykres funkcji z zaznaczonymi zerami i punktami (lub przedziałami) startowymi,
- porównanie wyników otrzymanych przy użyciu poszczególnych metod, zawierające tablicę, a w niej: punkt (przedział) początkowy (pp), wartość funkcji w (na krańcach) pp, punkt końcowy (pk), wartość funkcji w pk, liczba iteracji dla wszystkich metod,
- komentarz do otrzymanych wyników oraz wnioski eksperymentów (ocena poprawności wyników, dokładności, efektywności algorytmów, itd.).

Rozwiązanie.

Zadanie 1.

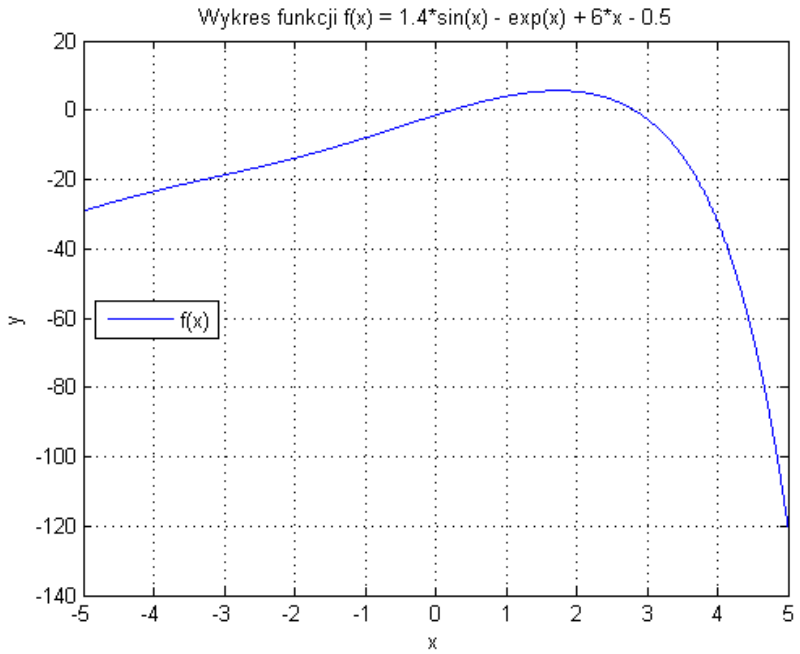
W tym zadaniu naszym celem jest znalezienie wszystkich miejsc zerowych funkcji $f(x) = 1.4 \cdot \sin(x) - e^x + 6 \cdot x - 0.5$ w przedziale $[-5, 5]$, przy użyciu wyspecyfikowanych metod (tj. metody bisekcji oraz metody Newtona).

Na początku sporządzimy wykres funkcji $f(x)$.
Taki zabieg pomoże nam w określeniu odpowiednich przedziałów izolacji pierwiastka lub punktów startowych (zależnie od zastosowanej metody).

Listing skryptu sporządzającego wykres funkcji $f(x)$, dla $x \in [-5, 5]$ (Zad1_wykres.m):

```
% Zadanie 1.
% Sporzadzenie wykresu funkcji f(x) = 1.4*sin(x) - exp(x) + 6*x - 0.5
% dla x nalezacego do przedzialu [-5,5]


clear;
clc;


% Uchwyt do funkcji f(x)
f = @(x) ( 1.4*sin(x) - exp(x) + 6*x - 0.5 );


x = [-5:0.0001:5]; % argumenty
y = f(x); % wartosci funkcji


% Wykres funkcji f
figure(1);

plot(x,y);
grid on;

title('Wykres funkcji f(x) = 1.4*sin(x) - exp(x) + 6*x - 0.5');
xlabel('x');
ylabel('y');
legend('f(x)', 'Location', 'West');
```

Wykres:



Wykres 1. Wykres funkcji $f(x) = 1.4 \cdot sin(x) - e^x + 6 \cdot x - 0.5$, *dla* $x \in [-5, 5]$.

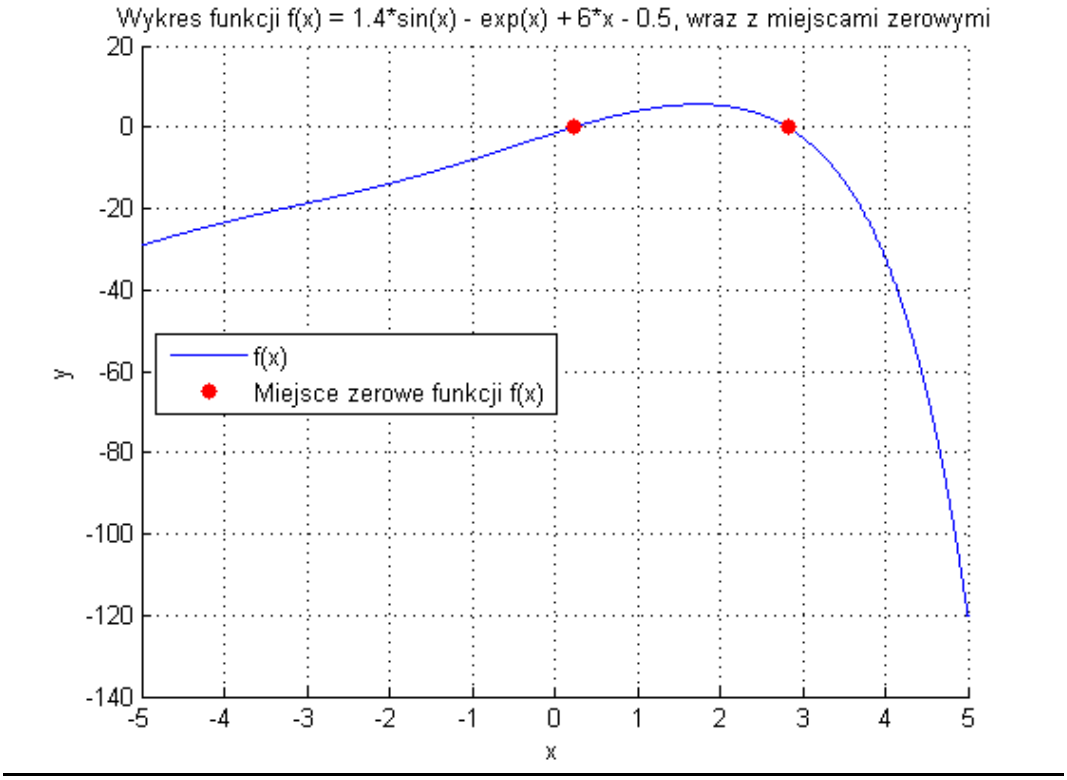
Z analizy wykresu 1. wynika, funkcja $f(x)$ ma 2 miejsca zerowe w przedziale $[-5,5]$.
Jedno z miejsc zerowych znajduje się w przedziale $[0, 0.5]$,
natomiast drugie w przedziale $[2.5, 3]$.

Jak się okaże w dalszej części sprawozdania, miejscami zerowymi
(z dokładnością do 4. miejsca po przecinku) funkcji $f(x)$ są liczby 0.2397 oraz 2.8270.

Listing skryptu sporządzającego wykres funkcji $f(x)$ (dla $x \in [-5, 5]$)
wraz z przybliżonymi miejscami zerowych funkcji $f(x)$ (Zad1_wykres_i_zera.m):

```
% Zadanie 1.
% Sporzadzenie wykresu funkcji f(x) = 1.4*sin(x) - exp(x) + 6*x - 0.5
% dla x nalezacego do przedzialu [-5,5],
% wraz z przyblizonymi wartosciami miejsc zerowych


clear;
clc;


% Uchwyt do funkcji f(x)
f = @(x) ( 1.4*sin(x) - exp(x) + 6*x - 0.5 );


x = [-5:0.0001:5]; % argumenty
y = f(x); % wartosci funkcji
roots = [0.2397 2.8270]; % miejsca zerowe funkcji f
values = f(roots); % wartosci funkcji f dla jej miejsc zerowych


% Wykres funkcji f oraz jej miejsca zerowe
figure(1);
hold on;

plot(x,y); % wygenerowanie wykresu
plot(roots,values, '.r', 'MarkerSize', 20); % wyroznienie pierwiastkow na
wykresie
grid on;

title('Wykres funkcji f(x) = 1.4*sin(x) - exp(x) + 6*x - 0.5, wraz z
miejscami zerowymi');
xlabel('x');
ylabel('y');
legend('f(x)', 'Miejsce zerowe funkcji f(x)', 'Location', 'West');

hold off;
```

Wykres:



Wykres 2. Wykres funkcji $f(x) = 1.4 \cdot sin(x) - e^x + 6 \cdot x - 0.5$, *dla* $x \in [-5, 5]$, *wraz z zaznaczonymi miejscami zerowymi funkcji* $f(x)$.

a) Metoda bisekcji

Metoda bisekcji jest jedną z metod iteracyjnych służących do znajdowania pierwiastka pojedynczego równania nieliniowego $f(x) = 0$.
Metoda bisekcji jest metodą zbieżną globalnie.
Dodatkowo, metoda ta jest zbieżna liniowo (rząd metody $p = 1$),
z ilorazem zbieżności $k = 0.5$ [1].

Metoda bisekcji bazuje na twierdzeniu Darboux: jeżeli funkcja f jest ciągła
i ma na końcach przedziału $[a, b]$ przeciwne znaki, tzn. $f(a) \cdot f(b) < 0$,
to w przedziale $[a, b]$ leży co najmniej jeden jej pierwiastek.

Opis metody

W kroku bazowym metody ustalamy początkowy przedział $[a_0, b_0]$ izolacji pierwiastka, wyznaczamy przybliżenie startowe $x_0 := \frac{a_0+b_0}{2}$ oraz ustalamy numer iteracji $n := 0$. Oczywiście, zakładamy, że $f(a_0) \cdot f(b_0) < 0$.

Dopóki zachodzi warunek $|f(x_n)| > \varepsilon$ (gdzie ε oznacza założoną dokładność rozwiązania, np. $\varepsilon = 10^{-6}$) oraz $n < \max\_iter$ (gdzie max_iter oznacza maksymalną liczbę iteracji, np. $max\_iter = 200$), dopóty wykonujemy, co następuje:

1) dzielimy przedział bieżący $[a_n, b_n]$ na dwie połowy za pomocą punktu środkowego $x_n = \frac{a_n+b_n}{2}$.
2) W zależności od tego, czy $f(a_n) \cdot f(x_n) < 0$, czy $f(x_n) \cdot f(b_n) < 0$, przyjmujemy (odpowiednio) $[a_{n+1}, b_{n+1}] = [a_n, x_n]$ lub $[a_{n+1}, b_{n+1}] = [x_n, b_n]$.
3) Inkrementujemy licznik iteracji: $n := n + 1$.

W opisie algorytmu w postaci liczby kroków, ε oznacza założoną dokładność rozwiązania (np. $\varepsilon = 10^{-6}$), natomiast max_iter oznacza maksymalną dopuszczalną liczbę iteracji algorytmu (np. $max\_iter = 200$).

Algorytm w postaci listy kroków:

1) Przyjmij początkowy przedział izolacji pierwiastka $[a_0, b_0]$
2) $x_0 := \frac{a_0+b_0}{2}$
3) $n := 0$
4) Dopóki $|f(x_n)| > \varepsilon$ oraz $n < max\_iter$, wykonuj:

    4.1) Jeśli $f(a_n) \cdot f(x_n) < 0$, to wykonuj:

        4.1.1) $[a_{n+1}, b_{n+1}] := [a_n, x_n]$

    4.2) W przeciwnym przypadku (tj. jeśli nie zachodzi warunek z p. 4.1), wykonuj:

        4.2.1)) $[a_{n+1}, b_{n+1}] := [x_n, b_n]$

    4.3) $x_{n+1} := \frac{a_{n+1}+b_{n+1}}{2}$

    4.4) $n := n + 1$

Listing solwera (bisection.m):

```
function [x, n] = bisection(f, a, b, eps, max_iter)
% Solwer sluzacy do znajdowania zera funkcji
% nieliniowej za pomoca metody bisekcji.

% x - znalezione przyblizenie rozwiazania
% n - liczba iteracji potrzebnych do znalezienia rozwiazania
% f - uchwyt do funkcji, dla ktorej szukamy pierwiastka
% [a, b] - przedzial izolacji pierwiastka (zakladamy, ze f(a)*f(b) < 0)
% eps - dokladnosc rozwiazania
% max_iter - maksymalna liczba iteracji


% Poczatkowe przyblizenie rozwiazania
x = (a+b)/2;

% Licznik iteracji
n = 0;


% Poszukiwanie rozwiazania
while abs(f(x)) > eps  &&  n < max_iter
    if f(a)*f(x) < 0
        b = x; % nowym przedzialem izolacji jest [a, x]
    else
        a = x; % nowym przedzialem izolacji jest [x, b]
    end

    x = (a+b)/2; % kolejne przyblizenie rozwiazania
    n = n + 1; % kolejna iteracja
end



end
```

Listing skryptu wywołującego (Zad1a.m):

```
% Zadanie 1a) - metoda bisekcji


clear;
clc;


% Uchwyt do funkcji, dla ktorej poszukujemy miejsc zerowych
f = @(x) ( 1.4*sin(x) - exp(x) + 6*x - 0.5 );


% Przedzialy [a(i), b(i)] izolacji pierwiastka
a = [-5, 4, 0.5, -5, -1,   -5,   -5, -1, 2.5, 2.5, 0.5]'; % poczatki
przedzialow
b = [-1, 5, 2.5,  5,  4,  0.5,  2.5,  1,   3,   5,   5]'; % konce
przedzialow


% Wartosci funkcji f dla poczatkow przedzialow izolacji pierwiastka
fa = f(a);


% Wartosci funkcji f dla koncow przedzialow izolacji pierwiastka
fb = f(b);


% Ilosc przedzialow izolacji pierwiastka
k = length(a);


% Numery przypadkow testowych
nr = [1:k]';


% Przyblizenia pierwiastka
x = zeros(k,1);


% Wartosci funkcji f dla znalezionych przyblizen pierwiastka
fx = zeros(k,1);


% Liczby iteracji potrzebne do znalezienia przyblizenia pierwiastka
n = zeros(k,1);


% Dokladnosc rozwiazania
eps = 1e-6;


% Maksymalna liczba iteracji
max_iter = 200;
```

```
% Czasy dzialania solwera
T = zeros(k,1);


% Znajdowanie przyblizen pierwiastka
for i=1:k
    tic;
    [x(i), n(i)] = bisection( f, a(i), b(i), eps, max_iter );
    T(i) = toc;

    fx(i) = f( x(i) );
end


% Przeskalowanie czasow do ms
T = T .* 1000;


% Wykres czasu dzialania solwera w zaleznosci od numeru przypadku testowego
figure(1);
plot(nr, T, '-o', 'MarkerSize', 10);
grid on;
title('Wykres czasu działania solwera w zależności od numeru przypadku
testowego');
xlabel('Numer przypadku testowego');
ylabel('Czas działania solwera [ms]');
```

Poniższa tabela przedstawia wyniki eksperymentów z metodą bisekcji.

Objaśnienie oznaczeń:

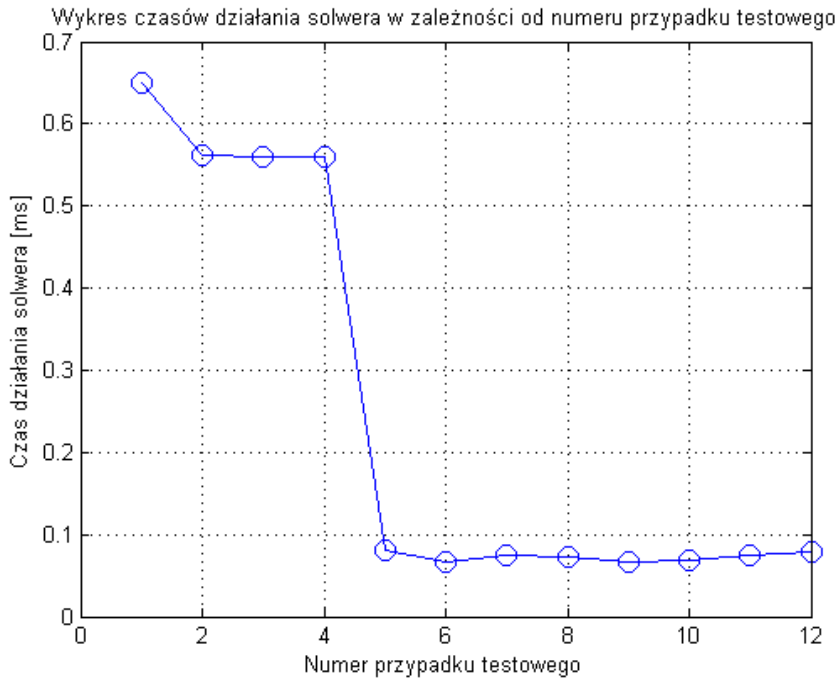
- nr – numer przypadku testowego
- a_0 – początek przedziału izolacji pierwiastka
- b_0 – koniec przedziału izolacji pierwiastka
- $f(a_0)$ – wartość funkcji f dla argumentu a_0 (tj. początku przedziału izolacji pierwiastka)
- $f(b_0)$ – wartość funkcji f dla argumentu b_0 (tj. końca przedziału izolacji pierwiastka)
- x – znalezione rozwiązanie, tj. przybliżenie pierwiastka
- $f(x)$ - wartość funkcji f dla argumentu x (tj. znalezionego przybliżenia pierwiastka)
- n – liczba iteracji potrzebnych do znalezienia rozwiązania, tj. przybliżenia pierwiastka

Uwaga 1. Przedział izolacji pierwiastka jest przedziałem obustronnie domkniętym $[a_0, b_0]$.
Uwaga 2. Wartości funkcji oraz znalezione przybliżenia pierwiastka są zaokrąglone do 4. miejsca po przecinku.

nr	a_0	b_0	$f(a_0)$	$f(b_0)$	x	$f(x)$	n
1	-5	-1	-29,1642	-8,0459	-1	-8,0459	200
2	4	5	-32,1577	-120,2557	5	-120,2557	200
3	0,5	2,5	1,5225	3,1554	2,5	3,1554	200
4	0	10	-1.5	-2.1968e+04	10	-2.1968e+04	200
5	-5	5	-29,1642	-120,2556	0,2397	5,1212e-07	24
6	-1	4	-8,0459	-32,1577	0,2397	1,4919e-07	18
7	-5	0,5	-29,1642	1,5225	0,2397	-9,3961e-07	22
8	-5	2,5	-29,1642	3,1554	0,2397	5,1212e-07	22
9	-1	1	-8,0459	3,9598	0,2397	1,4919e-07	19
10	2,5	3	3,1554	-2,3880	2,8270	-1,0236e-08	20
11	2,5	5	3,1554	-120,2557	2,8270	7,1853e-07	22
12	0,5	5	1,5225	-120,2557	2,8270	-5,5681e-07	24

Tabela 1. Wyniki eksperymentów z metodą bisekcji.

Poniżej przedstawiam wykres czasów działania solwera
w zależności od numeru przypadku testowego.



Wykres 3. Wykres czasów działania solwera w zależności od numeru przypadku testowego.

Komentarz:

Jeśli chodzi o tabelę 1., to możemy zauważyć, iż dla przypadków testowych 1-4, znalezione przybliżenia pierwiastka x nie spełniają warunku $|f(x)| \leq \varepsilon = 10^{-6}$. Fakt ten implikuje, że udało się znaleźć zadowalającego przybliżenia pierwiastka w założonej liczbie iteracji.
Z kolei, dla przypadków testowych 5-12 obserwujemy, że znalezione przybliżenia pierwiastka x spełniają warunek $|f(x)| \leq \varepsilon = 10^{-6}$. Oznacza to, że udało się znaleźć zadowalające przybliżenia pierwiastka w założonej liczbie iteracji.
Liczby iteracji dla przypadków testowych 5-12 są zbliżone do siebie – najmniejsza ich liczba to 18, a największa to 24.
Warto zauważyć, że dla przypadków testowych 1-6, przedział izolacji pierwiastka $[a_0, b_0]$ nie spełnia warunku $f(a_0) \cdot f(b_0) < 0$, a mimo to dla przypadków testowych 5-6 udało się znaleźć zadowalające przybliżenia pierwiastka. Dodatkowo, analiza tabeli 1. oraz wykresu 2. pozwala stwierdzić, że przedziały izolacji pierwiastka z przypadków testowych 1-3 nie zawierają w sobie żadnego pierwiastka funkcji f, natomiast przedziały izolacji pierwiastka z przypadków testowych 4-6 zawierają w sobie 2 pierwiastki.

Reasumując:

- przypadki testowe 1-3 pokazują, że jeśli nie zachodzi warunek $f(a_0) \cdot f(b_0) < 0$ oraz w przedziale $[a_0, b_0]$ nie ma pierwiastka, to metoda bisekcji słusznie nie znajdzie pierwiastka
- przypadki testowe 4-6 pokazują, że jeśli nie zachodzi warunek $f(a_0) \cdot f(b_0) < 0$ oraz w przedziale $[a_0, b_0]$ istnieje pierwiastek, to metoda bisekcji może znaleźć zadowalające przybliżenie pierwiastka (przypadki testowe 5-6) albo nie (przypadek testowy nr 4)
- przypadki testowe 7-12 pokazują, że jeśli zachodzi warunek $f(a_0) \cdot f(b_0) < 0$ oraz w przedziale $[a_0, b_0]$ istnieje pierwiastek, to metoda bisekcji znajdzie zadowalające przybliżenie pierwiastka

Zaleca się zatem dobieranie takich przedziałów izolacji pierwiastka $[a_0, b_0]$, dla których zachodzi $f(a_0) \cdot f(b_0) < 0$ – wtedy mamy gwarancję znalezienia zadowalającego przybliżenia pierwiastka (oczywiście, jeśli funkcja $f(x)$ ma wiele różnych pierwiastków, to dobór przedziału izolacji pierwiastka może wpłynąć na znalezione rozwiązanie).

Jeśli chodzi o wykres 3., to możemy zauważyć, iż dla przypadków testowych 1-3 mamy istotnie większy czas działania solwera niż dla pozostałych testów (tj. przypadków testowych 4-11). Wiąże się to z faktem nie znalezienia zadowalającego przybliżenia pierwiastka w założonej liczbie iteracji. Z kolei, dla przypadków testowych 4-11 obserwujemy zbliżony czas działania solwera (maksymalny czas działania nie przekracza 0.1 ms). Wynika to ze zbliżonej liczby iteracji potrzebnych do znalezienia zadowalającego przybliżenia pierwiastka. Pamiętamy, że dla przypadków 4-5, przybliżenia startowe $[a_0, b_0]$ nie spełniały warunku $f(a_0) \cdot f(b_0) < 0$.

b) Metoda Newtona

Metoda Newtona (inaczej metoda stycznych) jest jedną z metod iteracyjnych służących do znajdowania pierwiastka pojedynczego równania nieliniowego $f(x) = 0$. Jest zbieżna lokalnie, a ponadto (lokalnie, asymptotycznie) bardzo szybka; zbieżność metody Newtona jest kwadratowa (tzn. z rzędem zbieżności $p = 2$) [1].

Wyprowadzenie zależności iteracyjnej dla tej metody opiera się na rozwinięciu funkcji $f(x)$ w szereg Taylora w punkcie x_n (aktualnym przybliżeniu pierwiastka) i pozostawieniu części liniowej tego rozwinięcia: $f(x) \approx f(x_n) + f'(x_n) \cdot (x - x_n)$ [1]. (1)

Następne przybliżenie pierwiastka, x_{n+1}, wynika z przyrównania do zera prawej strony zależności (1) oraz podstawienia $x := x_{n+1}$: $f(x_n) + f'(x_n) \cdot (x_{n+1} - x_n) = 0$ [1]. (2)

Z zależności (2) wyznaczamy wzór iteracyjny metody Newtona: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$ [1]. (3)

Algorytm oparty na tej metodzie sprowadza się do ustalenia przybliżenia startowego x_0, a następnie stosowaniu wzoru (3) tak długo, aż nie zachodzi warunek stopu (np. tak długo, aż zachodzą warunki $|f(x_n)| > \varepsilon$ oraz $n < max\_iter$, gdzie ε oznacza założoną dokładność rozwiązania, natomiast max_iter oznacza maksymalną dopuszczalną liczbę iteracji algorytmu; przykładowo, $\varepsilon = 10^{-6}$ oraz $max\_iter = 200$).

W opisie algorytmu w postaci liczby kroków, ε oznacza założoną dokładność rozwiązania (np. $\varepsilon = 10^{-6}$), natomiast max_iter oznacza maksymalną dopuszczalną liczbę iteracji algorytmu (np. $max\_iter = 200$).

Algorytm w postaci listy kroków:

1) Przyjmij przybliżenie początkowe x_0
2) $n := 0$
3) Dopóki $|f(x_n)| > \varepsilon$ oraz $n < max\_iter$, wykonuj:
    3.1) Wyznacz kolejne przybliżenie rozwiązania: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$
    3.2) $n := n + 1$

Listing solwera (Newton.m):

```
function [x, n] = Newton(f, f_prim, x0, eps, max_iter)
% Solwer sluzacy do znajdowania zera funkcji
% nieliniowej za pomoca metody Newtona.

% x - znalezione przyblizenie rozwiazania
% n - liczba iteracji potrzebnych do znalezienia rozwiazania
% f - uchwyt do funkcji, dla ktorej szukamy pierwiastka
% f_prim - uchwyt do pochodnej funkcji f
% x0 - przyblizenie startowe
% eps - dokladnosc rozwiazania
% max_iter - maksymalna liczba iteracji


% Poczatkowe przyblizenie rozwiazania
x = x0;


% Licznik iteracji
n = 0;


% Poszukiwanie rozwiazania
while abs(f(x)) > eps  &&  n < max_iter
    x = x - f(x)/f_prim(x); % kolejne przyblizenie rozwiazania
    n = n + 1; % kolejna iteracja
end



end
```

Listing skryptu wywołującego (Zad1b.m):

```
% Zadanie 1b) - metoda Newtona


clear;
clc;


% Uchwyt do funkcji, dla ktorej poszukujemy miejsc zerowych
f = @(x) ( 1.4*sin(x) - exp(x) + 6*x - 0.5 );


% Uchwyt do pochodnej funkcji f
f_prim = @(x) ( 1.4*cos(x) - exp(x) + 6 );


% Przyblizenia startowe
x0 = [-100 -50 -10 -5 0 0.8 1.75 1.7506 3 5 10 50 100]';


% Wartosci funkcji f dla przyblizen startowych
fx0 = f(x0);


% Ilosc przyblizen startowych
k = length(x0);


% Numery przypadkow testowych
nr = [1:k]';


% Przyblizenia pierwiastka
x = zeros(k,1);


% Wartosci funkcji f dla znalezionych przyblizen pierwiastka
fx = zeros(k,1);


% Liczby iteracji potrzebne do znalezienia przyblizenia pierwiastka
n = zeros(k,1);


% Dokladnosc rozwiazania
eps = 1e-6;


% Maksymalna liczba iteracji
max_iter = 200;


% Czasy dzialania solwerow
T = zeros(k,1);
```

```
% Znajdowanie przyblizen pierwiastka
for i=1:k
    tic;
    [x(i), n(i)] = Newton( f, f_prim, x0(i), eps, max_iter );
    T(i) = toc;

    fx(i) = f( x(i) );
end


% Przeskalowanie czasow do ms
T = T .* 1000;


% Wykres czasu dzialania solwera w zaleznosci od numeru przypadku testowego
figure(1);
plot(nr, T, '-o', 'MarkerSize', 10);
grid on;
title('Wykres czasu działania solwera w zależności od numeru przypadku
testowego');
xlabel('Numer przypadku testowego');
ylabel('Czas działania solwera [ms]');
```

Poniższa tabela przedstawia wyniki eksperymentów z metodą Newtona.

Objaśnienie oznaczeń:

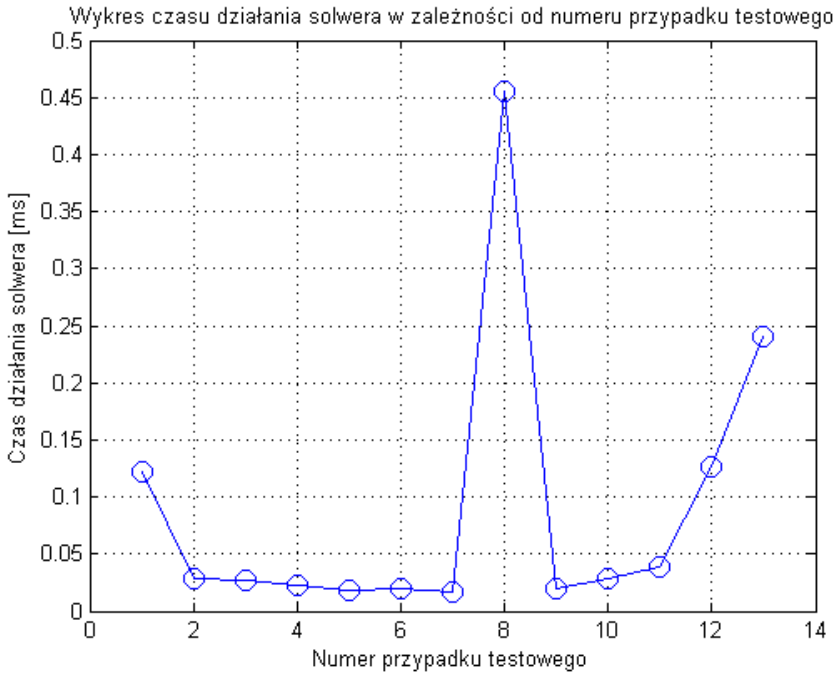
- nr – numer przypadku testowego
- x_0 – przybliżenie startowe
- $f(x_0)$ – wartość funkcji f dla argumentu x_0 (tj. przybliżenia startowego)
- x – znalezione rozwiązanie, tj. przybliżenie pierwiastka
- $f(x)$ - wartość funkcji f dla argumentu x (tj. znalezionego przybliżenia pierwiastka)
- n – liczba iteracji potrzebnych do znalezienia rozwiązania, tj. przybliżenia pierwiastka

Uwaga. Wartości funkcji oraz znalezione przybliżenia pierwiastka są zaokrąglone do 4. miejsca po przecinku.

nr	x_0	$f(x_0)$	x	$f(x)$	n
1	-100	-599,7911	2,8270	-5,5778e-13	10
2	-50	-300,1327	2,8270	-1,1404e-12	5
3	-10	-59,7384	2,8270	-9,9121e-13	6
4	-5	-29,1642	0,2397	-1,4845e-09	4
5	0	-1,5	0,2397	-1,1414e-11	3
6	0,8	3,0788	0,2397	-6,5113e-07	3
7	1,75	5,6230	NaN	NaN	2
8	1,7506	5,6230	469,9715	-1,2765e+204	200
9	3	-2,3880	2,8270	-5,06589e-07	3
10	5	-120,2557	2,8270	-7,1054e-14	7
11	10	-21967,7274	2,8270	-1,3358e-12	12
12	50	-5,1847e+21	2,8270	-1,3820e-12	52
13	100	-2,6881e+43	2,8270	-1,3820e-12	102

Tabela 2. Wyniki eksperymentów z metodą Newtona.

Poniżej przedstawiam wykres czasu działania solwera
w zależności od numeru przypadku testowego.



Wykres 4. Wykres czasu działania solwera w zależności od numeru przypadku testowego.

Komentarz:

Jeśli chodzi o tabelę 2. to możemy zauważyć, iż dla przypadków testowych 1-6 oraz 9-13 metoda Newtona znalazła zadowalające przybliżenie pierwiastka w założonej liczbie iteracji. Dla testów 2-6 oraz 9-10 metoda znalazła rozwiązanie w kilku iteracjach, dla testów 1 oraz 11 znaleziono rozwiązanie (odpowiednio) w 10 oraz 12 iteracjach, z kolei dla testów 12 i 13 metoda Newtona zbiega w istotnie większej liczbie iteracji – odpowiednio, w 52 oraz 102 iteracjach. Warto także zauważyć, że dla testów 1-3 oraz 9-13 znaleziono przybliżenie pierwiastka równe 2.8270, natomiast dla testów 4-6 znaleziono przybliżenie pierwiastka równe 0.2397.
Jeśli chodzi o testy 7-8, to metoda Newtona nie znalazła rozwiązania – punkty startowe zostały przemyślnie dobrane, gdyż znajdują się one w pobliżu maksimum lokalnego funkcji $f(x)$ (patrz wykres 2.).

Reasumując:

- w zależności od przybliżenia startowego, metoda Newtona może zbiegać bardzo szybko (tj. w kilku iteracjach), bardzo wolno (tj. w kilkudziesięciu lub więcej iteracjach) lub wcale
- dodatkowo, jeśli funkcja $f(x)$ ma wiele różnych pierwiastków, to w zależności od przybliżenia startowego możemy uzyskać różne przybliżenia pierwiastka

Jeśli chodzi o wykres 4., to możemy zauważyć, że dla większości testów
(tj. testów 2-7 oraz 9-11) osiągamy czasu poniżej 0.05 ms.
Z kolei wykonanie testów 1, 8, 12, 13 zajęło istotnie więcej czasu.
Test 1 potrzebował 10 iteracji, a dłuższe wykonanie może wynikać z uruchamiania skryptu.
Z kolei wykonanie testów 8, 12, 13 zajęło sporą liczbę iteracji, zatem czas wykonywania był dłuższy.

Pytanie do zadania 1.
Czy i kiedy dana metoda może zawieść i dlaczego?

Odpowiedź:
Zaimplementowane w zadaniu 1. metody bisekcji oraz Newtona mogą zawieść (tzn. nie znaleźć dostatecznego dobrego przybliżenia miejsca zerowego w założonej liczbie iteracji).
Jeśli chodzi o metodę bisekcji, metoda ta może zawieść, jeśli początkowy przedział izolacji pierwiastka $[a_0, b_0]$ nie spełnia warunku $f(a_0) \cdot f(b_0) < 0$
(w czasie eksperymentów z metodą bisekcji uwzględniłem taką właśnie sytuację).
Z kolei, jeśli początkowy przedział izolacji pierwiastka $[a_0, b_0]$ spełnia warunek $f(a_0) \cdot f(b_0) < 0$, to metoda bisekcji znajdzie pierwiastek w przedziale $[a_0, b_0]$.

Jeśli chodzi o metodę Newtona, metoda ta może zawieść, jeśli przybliżenie startowe jest poza obszarem atrakcji jakiegokolwiek pierwiastka α (*obszarem atrakcji* rozwiązania α nazywamy zbiór wszystkich punktów początkowych x_0, dla których metoda jest zbieżna do α [1]). Oczywiście, w eksperymentach z metodą Newtona uwzględniłem taką sytuację.

Zadanie 2.

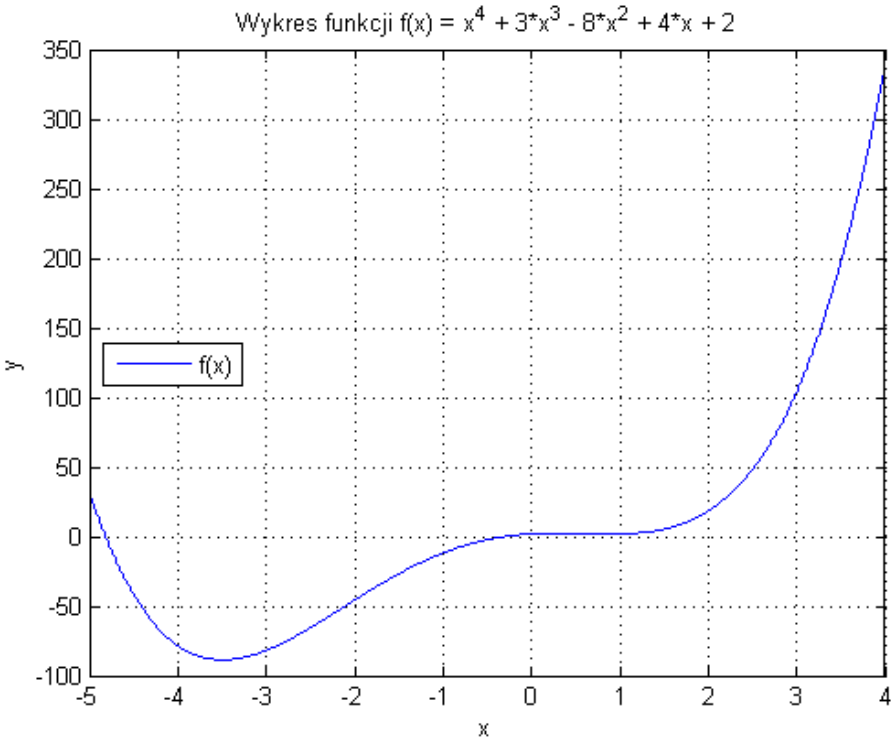
W tym zadaniu celem będzie znalezienie miejsc zerowych wielomianu $f(x) = x^4 + 3 \cdot x^3 - 8 \cdot x^2 + 4 \cdot x + 2$.

Na początku zaprezentuję wykres funkcji $f(x)$.

Listing skryptu generującego wykres funkcji $f(x)$ (Zad2_wykres.m):

```
% Zadanie 2.
% Sporzadzenie wykresu wielomianu f(x) = x^4 + 3*x^3 - 8*x^2 + 4*x + 2
% dla x nalezacego do przedzialu [-5,5]


clear;
clc;


% Uchwyt do funkcji f(x)
f = @(x) ( x.^4 + 3*x.^3 - 8*x.^2 + 4.*x + 2 );


x = [-5:0.0001:4]; % argumenty
y = f(x); % wartosci funkcji


% Wykres funkcji f
figure(1);

plot(x,y);
grid on;

title('Wykres funkcji f(x) = x^4 + 3*x^3 - 8*x^2 + 4*x + 2');
xlabel('x');
ylabel('y');
legend('f(x)', 'Location', 'West');
```

Wykres:



Wykres 5. Wykres funkcji $f(x) = x^4 + 3 \cdot x^3 - 8 \cdot x^2 + 4 \cdot x + 2$, *dla* $x \in [-5, 4]$.

Z analizy wykresu 5. wynika, że funkcja $f(x)$ ma 2 miejsca zerowe rzeczywiste – jedno z nich należy do przedziału $[-5, -4]$, natomiast drugie z nich należy do przedziału $[-1, 0]$.

Jak się okaże w dalszej części sprawozdania, funkcja $f(x)$ posiada 2 pierwiastki rzeczywiste oraz 2 pierwiastki zespolone (nie będące liczbami rzeczywistymi) sprzężone (gdyż współczynniki wielomianu $f(x)$ są rzeczywiste).

Metoda Müllera MM2

Metoda Müllera MM2 jest jedną z metod iteracyjnych, służących do znajdowania miejsc zerowych (zarówno rzeczywistych, jak i zespolonych) wielomianów [1].

Metoda ta wykorzystuje informację o wartości wielomianu i jego pochodnych, pierwszego i drugiego rzędu w aktualnym punkcie (przybliżeniu miejsca zerowego) x_k [1]. Metoda Mullera jest zbieżna lokalnie, z rzędem zbieżności $p = 1.84$ [1].

Opis metody

Naszym celem będzie znalezienie miejsca zerowego funkcji wielomianowej $f(x) = a_k \cdot x^k + a_{k-1} \cdot x^{k-1} + \cdots + a_1 \cdot x + a_0$.

Na początku dobieramy przybliżenie startowe x_0 i ustalamy licznik iteracji $n \coloneqq 0$.

Dopóki zachodzi warunek $|f(x_n)| > \varepsilon$ (gdzie ε oznacza założoną dokładność rozwiązania, np. $\varepsilon = 10^{-6}$) oraz $n < \max\_iter$ (gdzie max_iter oznacza maksymalną liczbę iteracji, np. $max\_iter = 200$), dopóty wykonujemy, co następuje:

- obliczamy $z_+ \coloneqq \frac{-2 \cdot f(x_n)}{f'(x_n) + \sqrt{(f'(x_n))^2 - 2 \cdot f(x_n) \cdot f''(x_n)}}$
- obliczamy $z_- \coloneqq \frac{-2 \cdot f(x_n)}{f'(x_n) - \sqrt{(f'(x_n))^2 - 2 \cdot f(x_n) \cdot f''(x_n)}}$
- jeśli $\left| f'(x_n) + \sqrt{(f'(x_n))^2 - 2 \cdot f(x_n) \cdot f''(x_n)} \right| \geq \left| f'(x_n) - \sqrt{(f'(x_n))^2 - 2 \cdot f(x_n) \cdot f''(x_n)} \right|$, to przyjmujemy $z_{min} \coloneqq z_+$; w przeciwnym przypadku przyjmujemy $z_{min} \coloneqq z_-$
- obliczamy nowe przybliżenie pierwiastka: $x_{n+1} \coloneqq x_n + z_{min}$
- inkrementujemy licznik iteracji: $n \coloneqq n + 1$

W opisie algorytmu w postaci liczby kroków, ε oznacza założoną dokładność rozwiązania (np. $\varepsilon = 10^{-6}$), natomiast max_iter oznacza maksymalną dopuszczalną liczbę iteracji algorytmu (np. $max\_iter = 200$).

Algorytm w postaci listy kroków

1) Przyjmij przybliżenie startowe x_0.
2) Ustaw licznik iteracji $n := 0$.

3) Dopóki $|f(x_n)| > \varepsilon$ oraz $n < \max\_iter$, wykonuj:

   3.1) Oblicz $z_+ := \frac{-2 \cdot f(x_n)}{f'(x_n) + \sqrt{(f'(x_n))^2 - 2 \cdot f(x_n) \cdot f''(x_n)}}$.

   3.2) Oblicz $z_- := \frac{-2 \cdot f(x_n)}{f'(x_n) - \sqrt{(f'(x_n))^2 - 2 \cdot f(x_n) \cdot f''(x_n)}}$.

   3.3) Jeśli $\left| f'(x_n) + \sqrt{(f'(x_n))^2 - 2 \cdot f(x_n) \cdot f''(x_n)} \right| \geq \left| f'(x_n) - \sqrt{(f'(x_n))^2 - 2 \cdot f(x_n) \cdot f''(x_n)} \right|$,
   to ustaw $z_{min} := z_+$; w przeciwnym przypadku ustaw $z_{min} := z_-$.

   3.4) Oblicz nowe przybliżenie pierwiastka: $x_{n+1} := x_n + z_{min}$.
   3.5) Inkrementuj licznik iteracji: $n := n + 1$.

Listing solwera (MullerMM2.m):

```
function [x, n] = MullerMM2(a, x0, eps, max_iter)
% Solwer sluzacy do znajdowania zera funkcji
% wielomianowej f(t) za pomoca metody Mullera MM2.

% x - znalezione przyblizenie rozwiazania
% n - liczba iteracji potrzebnych do znalezienia rozwiazania
% a - wspolczynniki wielomianu f(t) (od najwyszej do najnizszej potegi
zmiennej t)
% x0 - przyblizenie startowe
% eps - dokladnosc rozwiazania
% max_iter - maksymalna liczba iteracji



% Wspolczynniki wielomianu f'(t) (od najwyszej do najnizszej potegi
zmiennej t)
a_prim = polyder(a);
```

```
% Wspolczynniki wielomianu f''(t) (od najwyszej do najnizszej potegi
zmiennej t)
a_bis = polyder(a_prim);


% Poczatkowe przyblizenie rozwiazania
x = x0;

% Licznik iteracji
n = 0;

% Wartosc wielomianu dla przyblizenia startowego
f = polyval(a,x);


% Poszukiwanie rozwiazania
while abs(f) > eps  &&  n < max_iter

    % Wartosc 1. pochodnej wielomianu dla aktualnego przyblizenia
rozwiazania
    f_prim = polyval(a_prim,x);

    % Wartosc 2. pochodnej wielomianu dla aktualnego przyblizenia
rozwiazania
    f_bis = polyval(a_bis,x);

    % Mianowniki wartosci z_plus oraz z_minus (odpowiednio)
    den_z_plus =  f_prim + sqrt(f_prim^2 - 2*f*f_bis);
    den_z_minus = f_prim - sqrt(f_prim^2 - 2*f*f_bis);

    % Wartosci z_plus oraz z_minus
    z_plus =  -2*f/den_z_plus;
    z_minus = -2*f/den_z_plus;

    if abs( den_z_plus ) >= abs( den_z_minus )
        x = x + z_plus; % kolejne przyblizenie rozwiazania
    else
        x = x + z_minus; % kolejne przyblizenie rozwiazania
    end

    n = n + 1; % kolejna iteracja

    f = polyval(a,x); % wartosc funkcji dla kolejnej iteracji
end



end
```

Deflacja czynnikiem liniowym

Po znalezieniu pojedynczego pierwiastka α (np. za pomocą metody Müllera MM2) należy, przed wyznaczaniem następnego, uprościć wielomian dzieląc go przez czynnik $(x - \alpha)$, zawierający znaleziony pierwiastek. Jest to tzw. *deflacja czynnikiem liniowym*. Uzyskany wielomian niższego rzędu jest już prostszy i nie wyznaczymy ponownie tego samego pierwiastka (chyba że jest wielokrotny) [1]. Wykonując wielokrotnie (tj. k razy, gdzie k – stopień wielomianu) deflację czynnikiem liniowym, uzyskamy wszystkie pierwiastki wielomianu. Aby dokonać deflacji czynnikiem liniowym, można posłużyć się funkcją *deconv* języka MATLAB (więcej szczegółów można znaleźć w dokumentacji MATLAB-a).

Schemat metody Müllera MM2 z deflacją czynnikiem liniowym

Niech $f(x)$ będzie wielomianem k-tego stopnia, dla którego będziemy poszukiwali pierwiastków. Załóżmy ponadto, że za każdym razem będziemy startować z ustalonego przybliżenia początkowego x_0.
Wówczas znajdowanie wszystkich pierwiastków wielomianu $f(x)$ za pomocą metody Müllera MM2 z deflacją czynnikiem liniowym wygląda następująco:

1) Wykonuj k razy:
   1.1) Wyznacz za pomocą metody Müllera MM2 pierwiastek α wielomianu $f(x)$ (startując z ustalonego przybliżenia początkowego x_0).
   1.2) Wyznacz (np. za pomocą funkcji *deconv* w MATLAB-ie) wielomian $q(x)$ oraz stałą r, takie że $f(x) = (x - \alpha) \cdot q(x) + r$ oraz przypisz $f(x) \coloneqq q(x)$ (w ten sposób obniżamy stopień wielomianu $f(x)$ o 1).

Listing skryptu wywołującego metodę Müllera MM2 z deflacją czynnikiem liniowym, dla różnych przypadków testowych (Zad2.m):

```
% Zadanie 1b) - metoda Mullera MM2 z deflacja czynnikiem liniowym


clear;
clc;


% Wspolczynniki wielomianu f(x)
% (od najwyzszej do najnizszej potegi zmiennej x)
a = [1 3 -8 4 2];


% Stopien wielomianu f(x)
k = length(a)-1;


% Przyblizenia startowe
x0 = [-50 -10 -5 -2 -1 0 1 2 5 10 50]';


% Wartosci funkcji f dla przyblizen startowych
fx0 = polyval(a, x0);


% Ilosc przyblizen startowych (i przypadkow testowych zarazem)
m = length(x0);


% Dokladnosc rozwiazania
eps = 1e-6;


% Maksymalna liczba iteracji
max_iter = 200;


% Numery przypadkow testowych
nr = [1:m]';


% Przyblizenia pierwiastka
x = zeros(m,k);


% Wartosci funkcji f dla znalezionych przyblizen pierwiastka
fx = zeros(m,k);


% Liczby iteracji potrzebne do znalezienia przyblizenia pierwiastka
% (dla kazdego przypadku testowego)
n = zeros(m,k);
```

```
% Sumaryczne liczby iteracji dla przypadkow testowych
sum_n = zeros(m,1);


% Dokladnosc rozwiazania
eps = 1e-6;


% Maksymalna liczba iteracji
max_iter = 200;


% Czasy dzialania solwerow dla poszczegolnych przyblizen pierwiastka
% (w ramach kazdego przypadku testowego)
T = zeros(m,k);


% Sumaryczne czasy dzialania solwerow dla przypadkow testowych
sum_T = zeros(m,1);


% Metoda Mullera MM2 z deflacja czynnikiem liniowym
for i=1:m % dla kazdego przypadku testowego

    % Wspolczynniki wielomianu biezacego;
    % poczatkowo oryginalne wspolczynniki wielomianu f(x)
    b = a;

    for j=1:k % dla kazdego pierwiastka

        tic;

        % Znalezione przyblizenie pierwiastka
        [x(i,j), n(i,j)] = MullerMM2(b, x0(i), eps, max_iter);

        T(i,j) = toc;


        % Przeskalowanie czasu do ms
        T(i,j) = T(i,j) * 1000;

        % Obliczenie wartosci wielomianu f(x)
        % dla znalezionego przyblizenia pierwiastka
        fx(i,j) = polyval( a, x(i,j) );

        % Deflacja czynnikiem liniowym
        [b, r] = deconv( b, [1 -x(i,j)] );
    end

    sum_n(i) = sum( n(i,:) );
    sum_T(i) = sum( T(i,:) );
end
```

```
% Wykres sumarycznego czasu dzialania solwera w zaleznosci od przypadku
testowego
figure(1);
plot(nr, sum_T, '-o', 'MarkerSize', 10);
grid on;
title('Wykres sumarycznego czasu działania solwera w zależności od
przypadku testowego');
xlabel('Numer przypadku testowego');
ylabel('Sumaryczny czas działania solwera [ms]');
```

Poniższa tabela przedstawia wyniki eksperymentów z metodą Newtona.

Objaśnienie oznaczeń:

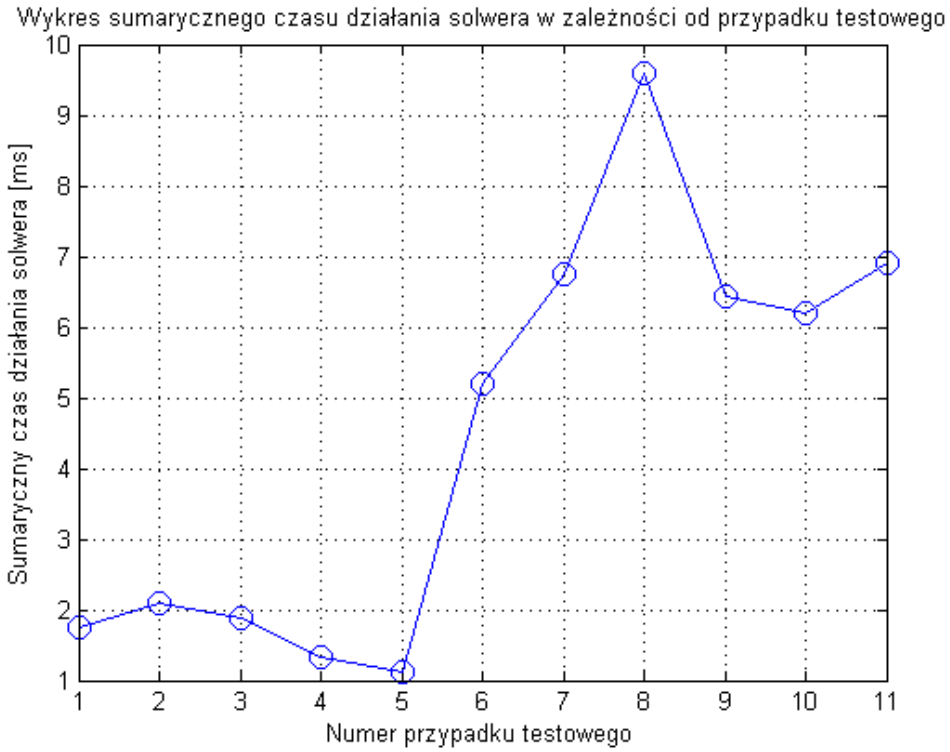
- nr – numer przypadku testowego
- x_0 – przybliżenie startowe
- $f(x_0)$ – wartość funkcji f dla argumentu x_0 (tj. przybliżenia startowego)
- $[x_1, x_2, x_3, x_4]$ – znalezione rozwiązania, tj. przybliżenia pierwiastków wielomianu $f(x)$
- $[f(x_1),\ f(x_2),\ f(x_3),\ f(x_4)]$ - wartości wielomianu $f(x)$ dla znalezionych przybliżeń pierwiastków
- sum_n – sumaryczna liczba iteracji potrzebnych do znalezienia rozwiązania, tj. przybliżeń pierwiastków wielomianu $f(x)$

Uwaga. Wartości funkcji oraz znalezione przybliżenia pierwiastka są zaokrąglone do 4. miejsca po przecinku.

nr	x_0	$f(x_0)$	$[x_1;\ x_2;\ x_3;\ x_4]$	$[f(x_1);\ f(x_2);\ f(x_3);\ f(x_4)]$	sum_n
1	-50	5854802	[-0,3007 + 8,5312e-17i; -4,8158 + 2,6511e-09i; 1,0583 + 0,5109i; 1,0583 - 0,5109i]	[2,2204e-16 + 8,1193e-16i; -1,3517e-07 - 4,1614e-07i; -6,3979e-09 + 1,4055e-07i; 8,7779e-08 + 1,0996e-07i]	22
2	-10	6162	[-0,3007 - 3,8116e-21i; -4,8158 - 1,4475e-12i; 1,0583 - 0,5109i; 1,0583 + 0,5109i]	[2,2204e-16 - 3,6276e-20i; 1,3706e-10 + 2,2721e-10i; -6,6789e-11 - 5,2975e-11i; -1,5362e-11 - 8,3826e-11i]	14
3	-5	32	[-0,3007 + 0,0000i; -4,8158 + 0,0000i; 1,0583 + 0,5109i; 1,0583 - 0,5109i]	[-1,2575e-08 + 0,0000i; -1,2548e-08 + 0,0000i; -1,2583e-08 - 3,0147e-12i; -1,2583e-08 + 3,0084e-12i]	9
4	-2	-46	[-0,3007 + 0,0000i; -4,81578 + 0,0000i 1,0583 + 0,5109i; 1,0583 - 0,5109i]	[-1,5244e-07 + 0,0000i; -1,5244e-07 + 0,0000i; -1,5244e-07 + 2,7423e-14i; -1,5244e-07 - 2,7423e-14i]	9
5	-1	-12	[-0,3007 + 0,0000i; -4,8158 + 2,8376e-20i; 1,0583 + 0,5109i; 1,0583 - 0,5109i]	[-4,4409e-16 + 0,0000i; -4,8850e-15 - 4,4540e-18i; -4,4409e-16 + 1,9984e-15i; 4,4409e-16 + 1,1102e-15i]	13
6	0	2	[-0,3007 + 0,0000i; 0,8088 + 0,5955i; 1,3085 - 0,5469i; -4,8166 - 0,0486i]	[-4,4409e-16 + 0,0000i; 1,0062 - 1,8748i; -0,9241 - 2,7189i; -0,0821 + 7,6299i]	205
7	1	2	[-0,3007 + 1,0712e-11i; 0,8088 - 0,5955i; 1,3085 + 0,5469i; -4,8166 + 0,0486i]	[-9,6992e-11 + 1,0195e-10i; 1,006 + 1,8748i; -0,9241 + 2,7189i; -0,0821 - 7,6299i]	208
8	2	18	[-0,3007 - 1,8275e-19i; 0,8088 + 0,5955i; 1,3085 - 0,5469i; -4,8166 - 0,0486i]	[2,2204e-16 - 1,7392e-18i; 1,0062 - 1,8748i; -0,9241 - 2,7189i; -0,0821 + 7,6299i]	210
9	5	822	[-0,3007 + 5,1335e-10i; 0,8088 - 0,5955i; 1,3085 + 0,5469i; -4,8166 + 0,0486i]	[2,5649e-09 + 4,8857e-09i; 1,0062 + 1,8748i; -0,9241 + 2,7189i; -0,0821 - 7,6299i]	212
10	10	12242	[-0,3007 - 3,4905e-11i; 0,8088 + 0,5955i; 1,3085 - 0,5469i; -4,8166 - 0,0486i]	[6,9501e-10 - 3,3220e-10i; 1,0062 - 1,8748i; -0,9241 - 2,7189i; -0,0821 + 7,6299i]	214
11	50	6605202	[-0,3007 - 2,1716e-19i; 0,8088 + 0,5955i; 1,3085 - 0,5469i; -4,8166 - 0,0486i]	[2,2204e-16 - 2,0667e-18i; 1,0062 - 1,8748i; -0,9241 - 2,7189i; -0,0821 + 7,6299i]	219

Tabela 3. Wyniki eksperymentów z metodą Müllera MM2 z deflacją czynnikiem liniowym

Poniżej prezentuję wykres sumarycznego czasu działania solwera
w zależności od przypadku testowego.



Wykres 6. Wykres sumarycznego czasu działania solwera w zależności od przypadku testowego

Komentarz:

Jeżeli części rzeczywiste lub urojone (w tabeli 5.) dla przybliżeń pierwiastków
lub odpowiadających im wartości funkcji $f(x)$ wyszły bardzo bliskie zera,
to możemy przyjąć (bez dużej straty), że są one zerowe.
Analizując tabelę 3., możemy wywnioskować, iż dla przypadków testowych 1-5
metoda Müllera MM2 z deflacją czynnikiem liniowym znalazła zadowalające przybliżenia
pierwiastków wielomianu $f(x)$ (świadczą o tym praktycznie zerowe wartości
wielomianu $f(x)$ dla uzyskanych przybliżeń pierwiastków).

Pierwiastkami wielomianu $f(x) = x^4 + 3 \cdot x^3 - 8 \cdot x^2 + 4 \cdot x + 2$
są zatem (z dokładnością do 4. miejsca po przecinku):
$x_1 = -0.3007,$
$x_2 = -4.8158,$
$x_3 = 1.0583 + 0.5109\mathrm{i},$
$x_4 = 1.0583 - 0.5109\mathrm{i}.$

Ponadto, dla przypadków testowych 1-5, łączna liczba iteracji jest pomiędzy 9 a 22 – daje to średnio kilka iteracji na znalezienie pojedynczego pierwiastka wielomianu $f(x)$.

Dla przypadków testowych 6-11, metoda nie znalazła zadowalających przybliżeń wszystkich pierwiastków wielomianu $f(x)$ (świadczą o tym wartości wielomianu $f(x)$ dla uzyskanych przybliżeń pierwiastków). Niemniej, należy zaznaczyć, że w każdym z tych testów został znaleziony jeden z pierwiastków, tj. $x_1 = -0.3007$. Ponadto, łączna liczba iteracji dla testów 6-11 jest spora – za każdym razem przekracza 200.

Reasumując, metoda Müllera MM2 z deflacją czynnikiem liniowym potrafi znaleźć wszystkie pierwiastki wielomianu, niemniej jest to uzależnione od przybliżenia startowego (warto przypomnieć założenie, w myśl którego w miarę zmniejszania stopnia wielomianu poprzez deflację czynnikiem liniowym, używamy tego samego przybliżenia startowego x_0 do wyznaczenia kolejnych pierwiastków). Jest to zatem metoda lokalna i szybko zbieżna, jeśli dobrze dobierzemy przybliżenie startowe.

Jeśli chodzi o wykres 6., to możemy zauważyć, że testy 1-5 zajmują co najwyżej ok. 2 ms, natomiast pozostałe testy (tj. testy 6-11) wykonują się znacznie dłużej – od ok. 5 ms aż do ok. 9.5 ms. Różnice czasowe wynikają z łącznej liczby iteracji dla naszych testów – dla testów 1-5 jest to co najwyżej 22 (a przeważnie kilka do kilkunastu iteracji), natomiast dla testów 6-11 jest to zawsze ponad 200 iteracji łącznie.

Bibliografia:

[1] Piotr Tatjewski „Metody Numeryczne”, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 2013