MNUM - Projekt 3, AiR sem-2

zadanie 3.22

1. **Proszę znaleźć wszystkie pierwiastki funkcji $f(x) = 2sinx - e^x + 5x - 1$ w przedziale [–5, 4] używając dla każdego zera:**
   a) własnego solwera z implementacją metody bisekcji.
   b) podanego na stronie przedmiotu solwera „newton.m" z implementacją metody Newtona.

Metoda bisekcji zwana inaczej metodą połowienia pozwala na wyznaczenie zer funkcji na danym przedziale. Aby ta metoda działała funkcja musi być ciągła na określonym przez nas przedziale. Aby wyznaczyć pierwiastki funkcji liczymy wartość: $c_n = \frac{a_n+b_n}{2}$, gdzie $[a_n, b_n]$ jest określonym przez nas przedziałem. Następnie sprawdzamy czy wartość c_n jest pierwiastkiem funkcji licząc $f(c_n)$. Jeśli tak to właśnie znaleźliśmy zero funkcji, jeśli nie to liczymy iloczyny funkcji: $f(a_n) \cdot f(c_n)$ i $f(b_n) \cdot f(c_n)$ i sprawdzamy, który z nich jest ujemny. Jeśli pierwszy z powyższych iloczynów jest ujemny to następnym przedziałem jaki będziemy sprawdzać jest przedział $[a_n, c_n]$, jeśli natomiast drugi iloczyn jest ujemny to przedziałem do sprawdzenia staje się $[c_n, b_n]$.

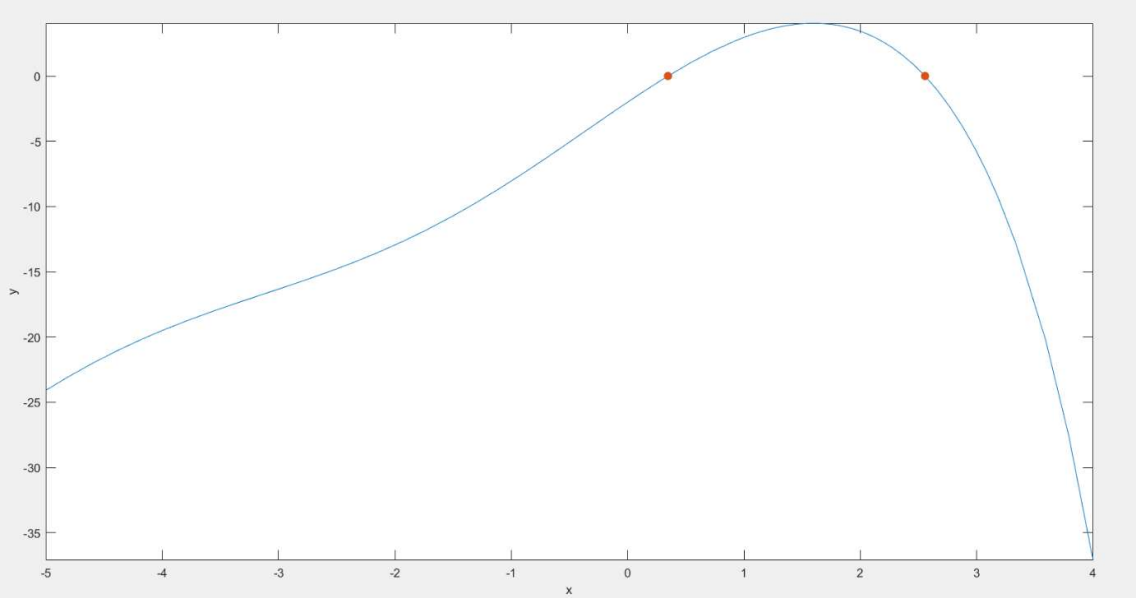
Zdarza się wówczas, że funkcja na podanym przez nas przedziale może mieć kilka zer lub nie mieć go wcale. Warto wtedy utworzyć sobie wektor argumentów o odpowiednim kroku, a następnie obliczyć wartości funkcji w tych argumentach. Następnie wziąć pierwszą wartość jako początek podprzedziału i sprawdzać, która z następnych wartości ma znak przeciwny do wartości rozpoczynającej podprzedział. Po jej znalezieniu jako koniec podprzedziału wybieramy argument odpowiadający tej wartości. Tak mamy pierwszy podprzedział, który zawiera tylko jeden pierwiastek dający się obliczyć bardzo łatwo właśnie metodą bisekcji. Następny podprzedział znajdujemy tak samo przyjmując tym razem jako początek podprzedziału koniec poprzedniego podprzedziału. W końcowym podprzedziale jest tak samo z tym wyjątkiem, że jego końcem jest koniec przedziału pierwotnego.

Solwer znajdujący zera metodą bisekcji:

```
% Deklaracja funkcji znajdującej pierwiastki funkcji metodą bisekcji:
function r = bisection(func, a, b, eps)
% func - funkcja, której pierwiastki chcemy znaleźć
% a - punkt początkowy przedziału
% b - punkt końcowy przedziału
% eps - dokładność rozwiązania
% args - argumenty, po których będziemy iterować w celu sprawdzenia ile
% pierwiastków zawiera funkcja na wskazanym przez nas przedziale:
args = [a:1:b];
j = 1;
% Podział zadeklarowanego przedziału na podprzedziały tak, aby każdy
% zawierał tylko jeden pierwiastek:
for i=1:length(args)
    if (func(a)*func(args(i)) < 0)
        range(:,j) = [a; args(i)];
        j = j + 1;
        a = args(i);
    end
end
[w, ~] = size(range);
i = 1;
% w - ilość podprzedziałów
% subr - podprzedział, w którym aktualnie znajdujemy pierwiastek
% range - zbiór podprzedziałów
subr = range(:,i);
% Pętla główna - znajdowanie pierwiastków funkcji:
while(true)
    a = subr(1,:);
    b = subr(2,:);
    % c - środek przedziału [a,b]
    c = (sum(subr))/2;
    yc = func(c);
    % Sprawdzenie czy c jest pierwiastkiem funkcji:
    % Jeśli tak to jest on dodawany do wektora rozwiązań, i następuje
    % przejście do następnego podprzedziału.
    if (yc == 0 || abs(yc) < eps)
        r(i) = c;
        i = i + 1;
        if i == w + 1
            break;
        end
        subr = range(:,i);
```

```
    % Jeśli nie to sprawdzamy podprzedział [c,b], jeśli iloczyn wartości
    % funkcji w punktach c i b jest dodatni to podzakresem, który
    % sprawdzamy staje się przedział [a,c].
    elseif func(b) * func(c) > 0
        subr = [a; c];
    % Jeśli iloczyn wartości funkcji w punktach c i b jest ujemny
    % to podzakresem, który sprawdzamy staje się przedział [c,b].
    elseif func(b) * func(c) < 0
        subr = [c;b];
    end
end
```

Wykres funkcji z zaznaczonymi zerami:



Otrzymane zera:

0.3469,

2.5561.

Wyniki otrzymane metodą bisekcji:

Punkt początkowy	Wartość w punkcie początkowym	Punkt końcowy	Wartość w punkcie końcowym	Liczba iteracji
-5	-24.0889	1	2.9647	31
1	2.9647	3	-5.8033	29

Gdzie wiersz w tabeli reprezentuje podprzedział, w którym znajduje się tylko jeden pierwiastek.

Wnioski:

Metoda bisekcji może zawieść w przypadku, gdy funkcja jest bardzo „płaska” (jej pochodna w otoczeniu zera jest bardzo mała), dlatego badając taką funkcję warto zadbać o to, aby długość

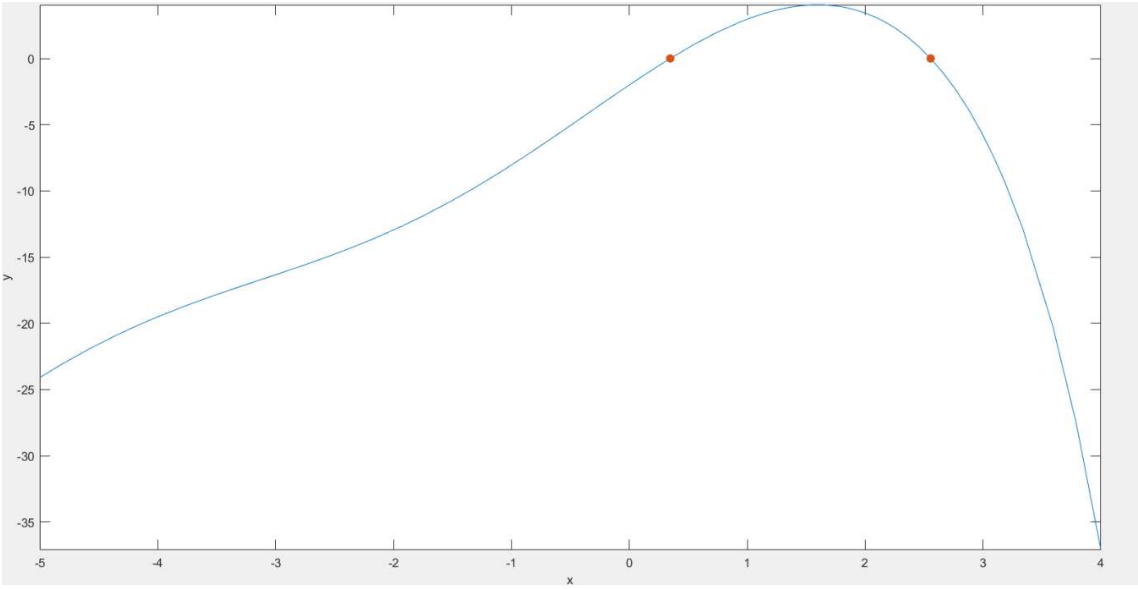
przedziału, który sprawdzamy była dostatecznie mała w celu otrzymania lepszej dokładności rozwiązania. Metoda bisekcji sprawdza się najlepiej w przypadku, gdy w określonym przez nas przedziale znajduje się tylko jedno zero funkcji. Gdy w danym przedziale jest ich więcej należy podzielić ten przedział na podprzedziały. Metoda bisekcji jest również zbieżna liniowo, a dokładność rozwiązania zależy od liczby wykonanych iteracji w celu znalezienia pierwiastka.

Metoda Newtona:

Metoda Newtona zwana inaczej metodą stycznych polega na aproksymacji funkcji jej liniowym przybliżeniem, które wynika z obcięcia rozwinięcia funkcji w szereg Taylora w aktualnym przybliżeniu pierwiastka x_n. Aby obliczyć następne przybliżenie pierwiastka x_{n+1}, przyrównujemy do zera wyrażenie: $f(x_n) + f'(x_n)(x_{n+1} - x_n) = 0$. W ten sposób otrzymamy zależność iteracyjną: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$.

Jako solwer znajdujący zera funkcji metodą Newtona wykorzystano plik „newton.m" dostępny na stronie przedmiotu.

Wykres funkcji z zaznaczonymi zerami:



Wyniki otrzymane metodą Newtona:

Punkt początkowy	Wartość funkcji w punkcie początkowym	Punkt końcowy	Wartość funkcji w punkcie końcowym	Liczba iteracji
-5	-24.0889	0.3469	-2.80711e-10	4
4	-37.1118	2.5561	-1.28431e-12	6

Wnioski:

Metoda Newtona jest bardziej efektywna w znajdowaniu zer funkcji niż metoda bisekcji. Aby uzyskać rozwiązanie o takiej samej dokładności wykonując znacznie mniej iteracji niż w metodzie

bisekcji. Metoda Newtona jest lokalnie zbieżna kwadratowo, jest bardzo efektywna, gdy funkcja jest bardzo stroma w otoczeniu pierwiastka. Niestety metoda ta może zawieść, gdy funkcja w pobliżu pierwiastka jest prawie poziomo (jej pochodna ma bardzo małą wartość). Niestety w punkcie znacznie oddalonym od rozwiązania metoda ta może stać się rozbieżna.

2. Używając metody Müllera MM1 proszę znaleźć wszystkie pierwiastki wielomianu czwartego stopnia: $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$, $[a_4, a_3, a_2, a_1, a_0]$ = [–1, –7, 7, 3, 9].

Metoda Müllera MM1 polega na aproksymacji wielomianu funkcją kwadratową w trzech wybranych przez nas punktach. Funkcja kwadratowa musi przechodzić przez te trzy punkty. Aby móc znaleźć pierwiastek wielomianu należy najpierw wybrać trzy punkty: x_0, x_1, x_2 przez które ma przechodzić funkcja kwadratowa oraz obliczyć wartości wielomianu w tych punktach: $f(x_0), f(x_1), f(x_2)$. Następnie wprowadzamy tzw. zmienne przyrostowe: $z_0 = x_0\text{-}x_2$ oraz $z_1 = x_1 - x_2$. Funkcja kwadratowa będzie mieć postać: $y(z) = az^2 + bz + c$. Następnie należy rozwiązać układ równań w celu wyznaczenia współczynników a, b, c:

$$az_0^2 + bz_0 + c = y(z_0) = f(x_0)$$
$$az_1^2 + bz_1 + c = y(z_1) = f(x_1)$$
$$c = y(0) = f(x_2)$$

Następnie mając współczynniki a, b, c, wyznaczamy dwie zmienne:

$$z_+ = \frac{-2c}{b + \sqrt{b^2 - 4ac}}$$

oraz

$$z_- = \frac{-2c}{b - \sqrt{b^2 - 4ac}}$$

następnie możemy obliczyć przybliżenie jednego z pierwiastków wielomianu korzystając ze wzoru:

$$x_3 = x_2 + z_{min}$$

gdzie:

$z_{min} = z_+$, gdy $\left|b + \sqrt{b^2 - 4ac}\right| \geq |b - \sqrt{b^2 - 4ac}|$,

$z_{min} = z_-$, w przeciwnych wypadkach.

Następnie powtarzamy ten algorytm do momentu, kiedy uznamy, że dokładność przybliżenia pierwiastka jest wystarczająco dobra. Aby wyznaczyć następne pierwiastki wielomianu wystarczy zastosować deflację wielomianu czynnikiem liniowym, czyli zwyczajnie podzielić wielomian przez $(x - \alpha)$ gdzie α to otrzymany pierwiastek na przykład stosując schemat Hornera.

Solwer znajdujący zera metodą Müllera MM1:

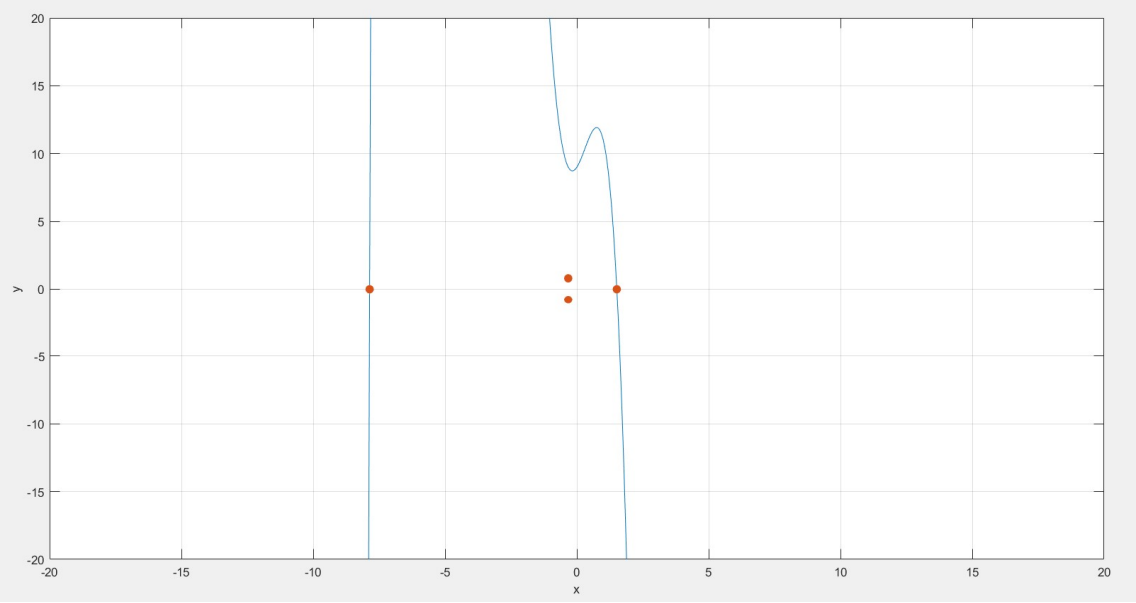
```
% Deklaracja funkcji znajdującej pierwiastki wielomianu metodą Mullera MM1:
function r = muller(w, eps, x0, x1, x2)
% w - wektor współczynników
% eps - dokładność rozwiązania
% x0 - dowolny punkt pierwszy
% x1 - dowolny punkt drugi
% x2 - dowolny punkt trzeci
% Punkty początkowe skryptu służące do tego, aby po podzieleniu wielomianu
% powrócić do tych punktów jako punktów wybranych:
x0p = x0;
x1p = x1;
x2p = x2;
p = w;
r = zeros(0);
j = 1;
% Główna pętla programu wraz z implementacją metody Mullera MM1:
while(j<=length(w)-1)
   x = [x0, x1, x2];
% Obliczenie różnic:
   z1 = x1-x0;
   z2 = x2-x1;

% Rozwiązanie układu równań tak, aby wyznaczyć współczynniki a,b,c:
   % Policzenie zmiennych pomocniczych do wyliczenia współczynników a i b:
   d1 = (polyval(p,x0)-polyval(p,x1))/z1;
   d2 = (polyval(p,x2)-polyval(p,x1))/z2;
   % Wyliczenie współczynnika a:
   a = (d2 - d1)/(z2+z1);
   % Wyliczenie współczynnika b:
   b = a * z2 + d2;
   % Wyliczenie współczynnika c:
   c = polyval(p,x2);
% Wyliczenie pierwiastków paraboli (zp - plus, zm - minus):
   zp = (-2*c/(b+sqrt(b^2-4*a*c)));
   zm = (-2*c/(b-sqrt(b^2-4*a*c)));
% Rozstrzygnięcie, który pierwiastek paraboli ma mniejszy moduł:
   if (abs(b+sqrt(b^2-4*a*c)) >= abs(b-sqrt(b^2-4*a*c)))
       zmin = zp;
   else
       zmin = zm;
   end
% Oblicznie przybliżonego rozwiązania:
   x3 = x2 + zmin;
```

```
% Usunięcie punktu położonego najdalej od przybliżonego rozwiązania:
   v = [abs(x0-x3); abs(x1-x3); abs(x2-x3)];
   i = find(max(v));
   x(i) = [];
   x(end+1) = x3;
   % Ustalenie nowych punktów:
   x0 = x(1);
   x1 = x(2);
   x2 = x(3);
   % Gdy pierwiastek osiągnie zamierzoną dokładność wektor pierwiastków
   % wielomianu jest rozszerzany o ten pierwiastek, wielomian jest
   % dzielony przez (x-"wartość pierwiastka") a punkty wybrane przyjmują
   % wartości punktów początkowych:
   if(abs(polyval(p,x3))<eps)
       r(end+1) = x3;
       j = j+1;
       p = deflation(p, x3);
       p = p(1:length(p)-1);
       x0 = x0p;
       x1 = x1p;
       x2 = x2p;
   end

end
% Transpozycja wektora z poziomego na pionowy:
r = r';
```

Wykres funkcji z zaznaczonymi pierwiastkami (także zespolonymi):



Otrzymane pierwiastki:

-0.3273 - 0.8054i,

-0.3273 + 0.8054i,

1.5151 - 0.0000i,

-7.8605 - 0.0000i.

Wyniki otrzymane za pomocą metody Müllera MM1:

Punkt początkowy	Wartość funkcji w punkcie początkowym
-5	419
0	9
4	-571

Punkt końcowy	Wartość funkcji w punkcie końcowym	Liczba iteracji
-0.3273 - 0.8054i	-0.0011 + 0.0001i	30
-0.3273 + 0.8054i	-0.0011 - 0.0001i	34
1.5151 - 0.0000i	-0.0011	24
-7.8605 - 0.0000i	0.0028	34

Wnioski:

Metoda Müllera MM1 jest zbieżna lokalnie, z rzędem zbieżności wynoszącym 1,84. Jest bardziej efektywna od metody siecznych i tylko niewiele wolniejsza od metody Newtona. Dzięki niej możemy znaleźć nie tylko pierwiastki rzeczywiste, ale także zespolone. Metody tej możemy także użyć do szukania zer nie tylko wielomianów, ale również innych funkcji nieliniowych.

Podsumowanie:

Najbardziej efektywnym i najszybszym algorytmem znajdującym zera funkcji jest metoda Newtona. Metoda Müllera MM1 jest niewiele wolniejsza od metody Newtona, jednakże pozwala nam wyznaczyć także pierwiastki zespolone. Metoda bisekcji natomiast jest najbardziej intuicyjna i zrozumiała, jednakże najmniej efektywna z tych wszystkich. Niestety metoda Newtona i metoda bisekcji mogą zawieść w przypadkach, gdy funkcja jest prawie pozioma w otoczeniu pierwiastka. Natomiast wszystkie metody dają całkiem poprawne wyniki.