Obliczenia naukowe - Lista nr 3

Paweł Narolski

23 listopada 2018 r.

1 Rozwiązywanie równań nieliniowych

W ramach niniejszej listy laboratoryjnej będziemy zajmować się rozwiązywaniem równań nieliniowych, czyli obliczaniem takiej wartości liczby x, dla której

$$f(x) = 0. \tag{1}$$

Omówimy trzy metody rozwiązywania równań nieliniowych: *metodę bisekcji (połowienia przedziału)*, *metodę Newtona* oraz *metodę siecznych*, których implementacje w języku *Julia* zostały przygotowane na potrzeby niniejszej listy laboratoryjnej.

1.1 Metoda bisekcji (połowienia przedziału)

Metoda bisekcji, zwana też metodą połowienia przedziału, korzysta z konsekwencji własności Darboux funkcji ciągłych: jeżeli f jest funkcją ciągłą w przedziale $[a, b]$ i jeśli $f(a) \cdot f(b) < 0$, a więc f zmienia swój znak w przedziale $[a, b]$, to tak funkcja musi mieć zero w (a, b).

Z powyższej własności korzystamy w następujący sposób:

Jeśli $f(a) \cdot f(b) < 0$, to obliczamy $c = \frac{1}{2} \cdot (a + b)$i sprawdzamy, czy $f(a) \cdot f(c) < 0$

1. Jeśli tak, to f ma zero w $[a, c]$; wówczas podstawiamy $b := c$
2. W przeciwnym razie mamy $f(c) \cdot f(b) < 0$; wówczas podstawiamy $a := c$

W obu przypadkach otrzymujemy nowy przedział $[a, b]$ dwa razy krótszy od poprzedniego, który zawiera zero funkcji f. Powyższe postępowanie możemy zatem powtórzyć. Jeśli $f(a) \cdot f(c) = 0$, to $f(c) = 0$ i tym samym znaleźliśmy zero naszej funkcji.

Warto zauważyć, że korzystając z metody połowienia przedziału jesteśmy w stanie otrzymać jedynie jedno, a nie wszystkie zera funkcji.

Co więcej, dokonując obliczeń w arytmetyce zmiennopozycyjnej *Float32* czy *Float64* musimy zdefiniować inny warunek zakończenia obliczeń, bowiem błędy przybliżeń mogą sprawić, że zdefiniowany warunek końcowy nie zostanie osiągnięty. Aby temu zaradzić, definiujemy wartości δ oraz ϵ, które kolejno definiują zadowalającą nas, dostatecznie małą wartość błędu przybliżeń wykonywanych obliczeń oraz pożądaną bliskość otrzymanej wartości iloczynu $f(a) \cdot f(c)$ do zera.

W implementacji algorytmu w języku Julia korzystamy z jeszcze dwóch poprawek:

1. Punkt środkowy c obliczamy za pomocą instrukcji $c = a+(b-a)/2$ zamiast instrukcji $c = (a+b)/2$.
2. Zmianę znaku badamy za pomocą nierówności $sgn(w) \neq sgn(u)$ zamiast $wu < 0$

Pierwszej zmiany dokonujemy na podstawie wniosków z przykładu przedstawionego przez Forsythe, Malcolm i Moler'a (1977), w którym punkt środkowy obliczany drugą instrukcją leży poza granicami

przedziału $[a, b]$ - w obliczeniach numerycznych lepiej jest obliczać nową wielkość, dodając do poprzedniej małą poprawkę.

Drugiej zmiany zaś dokonaliśmy, aby uniknąć zbędnego mnożenia, które może spowodować wystąpienie błędów niedomiaru bądź nadmiaru.

Kod źródłowy algorytmu zaimplementowanego w języku *Julia* prezentuje się następująco:

```
"""
Solves the formula f(x) = 0 using the bisection method.
    f - f(x) given as anonymous function
    a, b - starting range
    delta, epsilon - accuracy of computations
"""
function mbisekcji(f, a::Float64, b::Float64, delta::Float64, epsilon::Float64)
    u = f(a)
    v = f(b)
    e = b - a

    iterations = 0

    # print("a=$a, b=$b, u=$u, v=$v")

    if sign(u) == sign(v)
        # If function does not change its' sign in [a, b] return error
        return NaN, NaN, NaN, 1
    end

    while e > epsilon
        iterations += 1

        e = e/2          # Half of the length of the interval
        c = a + e        # The middle point of the interval
        w = f(c)         # The value of f in middle of the interval

        # The "quit" condition
        if (abs(e) < delta || abs(w) < epsilon)
            return c, w, it, 0
        end

        if sign(w) != sign(u)
            b = c        # Set the computed middle as the end of new interval
            v = w
        else
            a = c        # Set the computed middle as the beginning
            u = w
        end
    end
end
```

1.2 Metoda Newtona (stycznych)

Omówimy teraz działanie *metody Newtona*, a właściwie jej szczególnego wariantu odnoszącego się do lokalizacji miejsc zerowych funkcji rzeczywistych, nazywanego też *metodą Newtona-Raphsona*.

Dla danej funkcji f przyjmujemy następujące założenia:

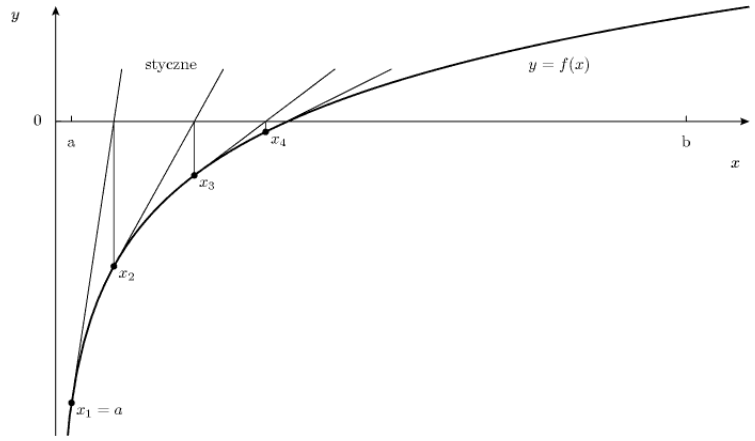
1. W przedziale $[a, b]$ znajduje się dokładnie jeden pierwiastek
2. Funkcja ma różne znaki na krańcach przedziału, tj. $f(a) \cdot f(b) < 0$
3. Pierwsza i druga pochodna f mają stały znak w przedziale $[a, b]$

W pierwszym kroku metody wybieramy punkt startowy x_0. Następnie z tego punktu wyprowadzamy prostą styczną w $f(x_0)$. Odcięta punktu przecięcia tej stycznej z osią OX układu współrzędnych, czyli pierwsza współrzędna tego przecięcia jest pierwszym przybliżeniem rozwiązania równania, które oznaczamy jako x_1.

Aby osiągnąć satysfakcjonujące nas przybliżenie, powyższe czynności powtarzamy, obierając kolejne otrzymane przybliżenia $x_1, x_2, \ldots$ jako kolejne punkty startowe dla metody. Kolejne przybliżenia dane są rekurencyjnym wzorem:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \tag{2}$$

Wizualizacja działania metody dla pierwszych czterech kroków prezentuje się następująco[1]:



Przygotowana na potrzeby zadania implementacja algorytmu w języku *Julia* przed przystąpieniem do właściwych obliczeń dokonuje sprawdzenia, czy możliwe jest zastosowanie metody (poprzez sprawdzenie, czy pochodna $f'(x)$ nie jest odpowiednio bliska zeru), oraz czy wartość funkcji dla przybliżenia początkowego jest dostatecznie bliska zeru.

Obliczenia zostają zakończone, kiedy znalezione zero spełnia założoną przez nas dokładność przeprowadzanych obliczeń, lub gdy kolejność kolejnych przeprowadzanych przybliżeń jest odpowiednio mała. W przypadku, gdy w zadanej liczbie iteracji nie jesteśmy w stanie otrzymać wyniku zwracamy błąd.

```
"""
Solves the formula f(x) = 0 using the Newton method.
    f, pf - f(x) and f'(x) given as anonymous functions
    x0 - starting approximation
    delta, epsilon - accuracy of computations
    maxit - maximum number of iterations
"""
function mstycznych(f, pf, x0::Float64, delta::Float64, epsilon::Float64, maxit::Int)
    v = f(x0)

    # If value of f in x0 is close to zero, return
```

[1] Na podstawie artykułu "Metoda Newtona", https://pl.wikipedia.org/wiki/Metoda_Newtona

```
    if abs(v) < epsilon
        return x0, v, 0, 0
    end

    # If f' is close to zero, return
    if abs(pf(x0) < epsilon)
        return NaN, NaN, NaN, 2
    end

    for it = 1:maxit
        x1 = x0 - (v/pf(x0))    # Next approximation of the solution
        v = f(x1)               # f value in next approximation

        # "End" condition
        if (abs(x1 - x0) < delta || abs(v) < epsilon)
            return x1, v, it, 0
        end

        x0 = x1                 # Set the new starting point
    end

    # Return error if result was not acheived in maxit
    return NaN, NaN, NaN, 1
end
```

Ponieważ zbieżność metody Newtona jest kwadratowa, jest ona zazwyczaj szybsza w działaniu od metody *bisekcji* i metody *stycznych*. Kiedy obliczone przybliżenia pierwiastka są dostatecznie bliskie faktycznej wartości pierwiastka wystarczy wykonać stosunkowo niewielką ilość iteracji metody, aby otrzymać maksymalną możliwą dokładność wyniku. Dzieje się tak, ponieważ przy spełnionych założeniach błąd maleje kwadratowo ze wzrostem ilości iteracji.

Niestety, zbieżność metody Newtona nie zawsze musi zachodzić. W wielu przypadkach, szczególnie, gdy punkt startowy jest daleko od szukanego pierwiastka metoda bywa rozbieżna.

W profesjonalnych zastosowaniach stosuje się najpierw stabilniejsze, lecz mniej wydajne metody (np. metodę bisekcji), aby znaleźć obszary zbieżności w dziedzinie funkcji, a następnie używa się metody Newtona-Raphsona do szybkiego i dokładniejszego obliczenia lokalnego pierwiastka równania.

Dodatkowo wprowadza się zabezpieczenie przed nadmierną ilością wykonywanych iteracji, których przekroczenie wiąże się z niepowodzeniem obliczeń.

1.3 Metoda siecznych

Jedną z wad *metody Newtona* jest fakt, iż wymaga ona obliczania wartości pochodnej funkcji f, której zera szukamy. Aby uporać się z tym problemem, zaproponowano *metodę siecznych* (w polskiej literaturze nazywaną także *metodą cięciw*).

W *metodzie siecznych* zastępujemy pochodną $f'(x)$ funkcji ilorazem różnicowym

$$f'(x_n) \approx' \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}. \tag{3}$$

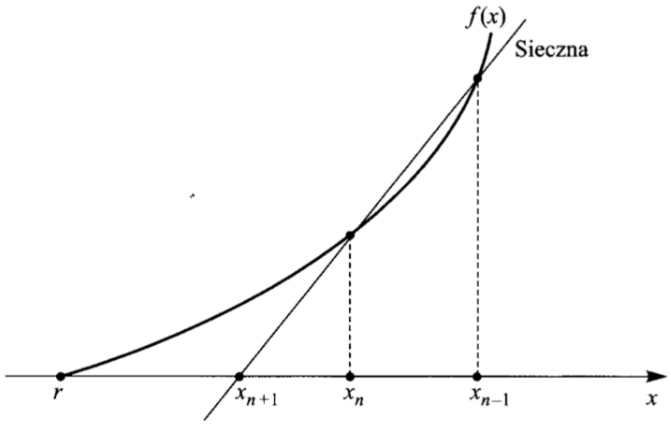
To przybliżenie wynikające wprost z definicji pochodnej funkcji pozwala zdefiniować *metodę siecznych* opisaną wzorem

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \tag{4}$$

dla n$\geqslant$ 1.

Taka definicja wymaga zdefiniowania dwóch punktów początkowych: x_n oraz x_{n-1}. W każdej kolejnej iteracji musimy jednak obliczyć tylko jedną nową wartość funkcji f, w przeciwieństwie do metody *Newtona*, dla której w każdym kroku oprócz $f(x)$ należało także obliczyć $f'(x)$.

W graficznym porównaniu do metody *Newtona* metoda *siecznych* różni się jedynie zastąpieniem stycznej do krzywej sieczną do krzywej, co ilustruje poniższy rysunek[2]:



W przygotowanej implementacji metody w języku *Julia* program "przestawia" końce a i b przedziału, kiedy wymaga tego utrzymanie nierówności $|f(a)| \leqslant |f(b)|$. Dzięki zastosowaniu tego zabiegu począwszy od drugiego kroku metody moduły wartości funkcji w punktach x_n nie rosną.

Program kończy obliczenia x_{n+1}, kiedy osiągnięte zostało zdefiniowane przybliżenie zera funkcji, bądź gdy w kolejnych iteracjach odległość kolejnych przybliżeń jest dostatecznie mała. Jeśli w zadanej liczbie iteracji nie udało się wyznaczyć wyniku, zwracany jest błąd.

```
"""
Solves the formula f(x) = 0 using the secant method.
    f - f(x) and given as anonymous function
    x0, x1 - starting approximations
    delta, epsilon - accuracy of computations
    maxit - maximum number of iterations
"""
function msiecznych(f, x0::Float64, x1::Float64, delta::Float64, epsilon::Float64,
maxit::Int)
    fx0 = f(x0)
    fx1 = f(x1)

    for it=1:maxit
        if abs(fx0) > abs(fx1)
            x0, x1, = x1, x0         # Perform swaps
            fx0, fx1 = fx1, fx0      # fx0 is greater than fx1
        end

        s = (x1 - x0)/(fx1 - fx0)
        x1 = x0
        fx1 = fx0
        x0 = x0 - (fx0 * s)          # new x0 where secant meets OX
        fx0 = f(x0)                  # value of f in new x0
```

[2]Na podstawie "Analiza Numeryczna" Kincaid, Cheney

```
24
25          # "End" condition
26          if (abs(x1 - x0) < delta || abs(fx0) < epsilon)
27              return x0, fx0, it, 0
28          end
29      end
30
31      # Error occurs if no result found in maxit
32      return NaN, NaN, NaN, 1
33  end
```

1.4 Porównanie działania metod wyznaczania pierwiastków równań nieliniowych

Dokonamy teraz porównania działania metod wyznaczania pierwiastków równań nieliniowych, aby na podstawie konkretnych danych móc dostrzec różnice w działaniu poszczególnych omówionych algorytmów.

Dokonamy wyznaczenia pierwiastka równania:

$$sinx - \frac{1}{2}x^2 = 0 \tag{5}$$

za pomocą metody:

1. bisekcji z przedziałem początkowym $[1.5, 2]$ i $\delta = \frac{1}{2} \cdot 10^{-5}, \epsilon = \frac{1}{2} \cdot 10^{-5}$
2. Newtona z przybliżeniem początkowym $x_0 = 1.5$ i $\delta = \frac{1}{2} \cdot 10^{-5}, \epsilon = \frac{1}{2} \cdot 10^{-5}$
3. siecznych z przybliżeniami początkowymi $x_0 = 1, x_1 = 2$ i $\delta = \frac{1}{2} \cdot 10^{-5}, \epsilon = \frac{1}{2} \cdot 10^{-5}$.

Wyniki przeprowadzonych obliczeń zaprezentowano w poniższej tabeli:

Metoda	Miejsce zerowe x_0	Wartość funkcji $f(x_0)$	Liczba iteracji	Błąd
Metoda bisekcji	1.9337539672851562	-2.7027680138402843e-7	16	0
Metoda stycznych	1.933753779789742	-2.2423316314856834e-8	4	0
Metoda siecznych	1.933753644474301	1.564525129449379e-7	4	0

Zauważamy, że metoda bisekcji potrzebowała 16 iteracji, aby znaleźć pierwiastek równania (5), podczas gdy metody stycznych i siecznych odnalazły poszukiwaną wartość już po 4 iteracjach. Dostrzegamy zatem wpływ kwadratowej zbieżności dwóch pozostałych metod na liczbę kroków potrzebnych do osiągnięcia poszukiwanego rozwiązania równania.

Choć w przypadku równania (5) metody stycznych i siecznych okazały się być bardziej skuteczne dla aproksymacji rozwiązania, warto pamiętać o tym, że takie zachowanie nie zostanie osiągnięte dla każdego możliwego przypadku.

1.5 Wyznaczanie wartości x dla której przecinają się wykresy danych funkcji

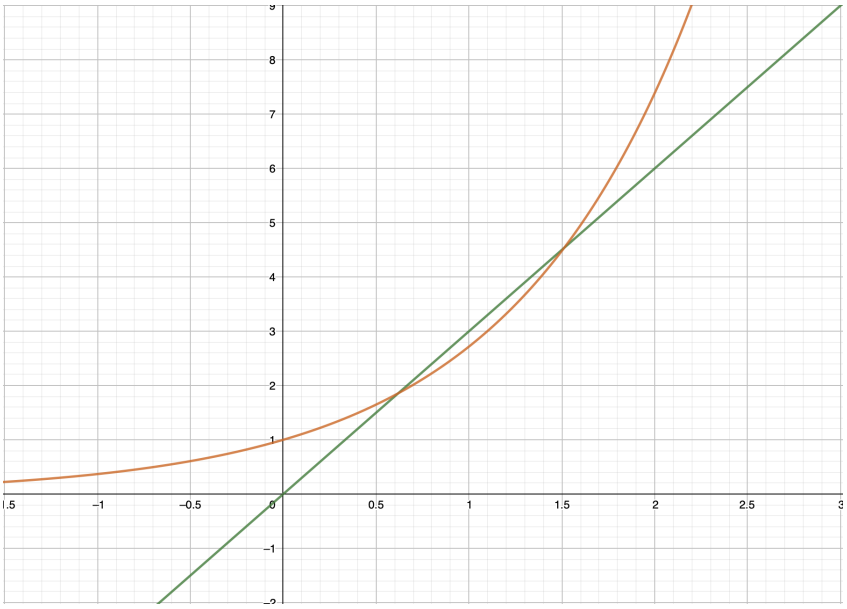
Opisane metody rozwiązywania równań nieliniowych możemy wykorzystać także do wyznaczania punktów przecięcia wykresów funkcji spełniających odpowiednie dla obranej metody założenia. W naszym przypadku dokonamy tego dla funkcji $f(x) = 3x$ oraz $g(x) = e^x$ przy użyciu metody bisekcji.

Co oczywiste, powyższe zadanie sprowadza się do rozwiązania równania

$$e^x - 3x = 0. \tag{6}$$

Pierwiastków równania będziemy poszukiwać z precyzją $\delta = 10^{-4}$ i $\epsilon = 10^{-4}$.

Aby wyznaczyć wartości x dla których przecinają się wykresy funkcji $f(x)$ oraz $g(x)$ na podstawie wykresu obu funkcji wybrane zostały dwa przedziały, dla których zostaną przeprowadzone obliczenia: przedział $[0, 1]$ oraz $[1, 2]$.



Wyniki przeprowadzonych obliczeń prezentuje poniższa tabelka:

	Przedział	Wartość	Liczba iteracji
x_0	$[0, 1]$	0.6190643310546875	16
x_1	$[1, 2]$	1.5121345520019531	18

Zauważamy, że aby móc prawidłowo wyznaczyć punkty przecięcia danych funkcji, a także miejsca zerowe dowolnych innych badanych funkcji za pomocą *metody bisekcji* konieczna jest znajomość ich przebiegu zmienności.

W przypadku braku możliwości skorzystania np. z kalkulatora graficznego bądź w wypadku szczególnie złośliwych funkcji określenie przedziałów, w których mają znajdować się miejsca zerowe może okazać się bardzo trudne. Na przykładzie tego zadania możemy się przekonać, że obranie przedziału $[0, 2]$ doprowadzi do niemożności uzyskania prawidłowego wyniku przy użyciu metody *bisekcji*, ponieważ na jego krańcach nasza funkcja nie zmienia swojego znaku.

Spostrzegamy także, że dla powyższego zadania moglibyśmy zastosować metody *stycznych* bądź *siecznych*, które dla adekwatnie obranych przybliżeń okazałyby się o tyle prostsze w zastosowaniu, że nie wymagałyby znajomości dokładnego przebiegu funkcji $f(x)$ i $g(x)$.

1.6 Dobieranie przedziałów i przybliżeń początkowych w celu obliczenia miejsc zerowych funkcji

Zastosujemy teraz omówione metody *bisekcji, Newtona i siecznych* do wyznaczenia miejsc zerowych funkcji

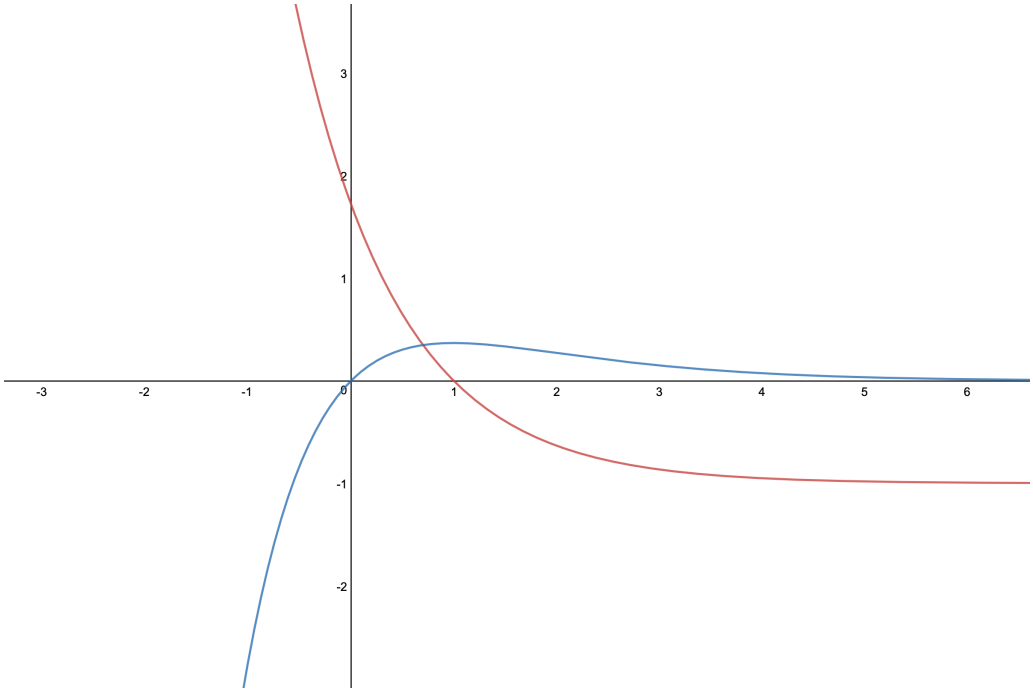
$$f(x) = e^{1-x} - 1 \tag{7}$$

oraz

$$g(x) = xe^{-x}. \tag{8}$$

Dla metody *bisekcji* wyznaczymy przedział początkowy, w ramach którego będziemy poszukiwać miejsca zerowego, zaś dla metod *stycznych i siecznych* dobierzemy adekwatne przybliżenie początkowe.

Rozwiązanie zadania rozpoczniemy od analizy wykresów funkcji (7) i (8).



Wykres funkcji $f(x) = e^{1-x} - 1$ *oraz* $g(x) = xe^{-x}$

Zauważamy, że już na podstawie powyższych wykresów jesteśmy w stanie prawidłowo wskazać miejsca zerowe obu funkcji, które dla $f(x)$ wynosi 1, a dla $g(x)$ - 0. Będziemy jednak chcieli przeprowadzić eksperymenty dla różnych przedziałów oraz wartości początkowych, aby sprawdzić, jak z tym zadaniem poradzą sobie poszczególne metody.

Nasze badania rozpoczniemy od sprawdzenia, jakie wyniki otrzymamy obierając różne przedziały początkowe dla metody *bisekcji* dla obu funkcji $f(x)$ i $g(x)$. Obrane krańce przedziałów oraz rezultaty przeprowadzonych dla nich obliczeń prezentują się następująco:

Przedział	Miejsce zerowe	Wartość $f(x)$	Liczba iteracji	Błąd
$[-0.0, 2.0]$	1.0	0.0	1	0
$[-4.0, 4.0]$	1.0	0.0	3	0
$[-0.0, 1.5]$	1.0000076293945312	-7.6293654275305656e-6	16	0
$[-1.5, 125.0]$	1.0000035166740417	-3.516667858249889e-6	23	0

Otrzymane wyniki obliczania miejsca zerowego $f(x)$ *metodą bisekcji*

Przedział	Miejsce zerowe	Wartość $g(x)$	Liczba iteracji	Błąd
$[-0.5, 0.5]$	0.0	0.0	1	0
$[-0.5, 0.4]$	1.5258789062623358e-6	1.5258765779576755e-6	16	0
$[-1.0, 5.0]$	7.62939453125e-6	7.62933632381113e-6	18	0
$[-42.7, 13.25]$	2.0146369927351243e-6	2.014632933977e-6	22	0

Otrzymane wyniki obliczania miejsca zerowego $g(x)$ *metodą bisekcji*

Zauważamy, że nieumiejętne dobranie przedziału początkowego będzie wiązało się z koniecznością wykonania znacznie większej ilości iteracji *metody bisekcji*.

Następnie przebadaliśmy wyniki otrzymywane po zastosowaniu metody *stycznych* dla różnych obranych przybliżeń początkowych. Ustalmy maksymalną liczbę iteracji $maxit = 30$.

Przybliżenie początkowe	Miejsce zerowe	Wartość $g(x)$	Liczba iteracji	Błąd
0.99	0.9999999987583194	1.2416805361681327e-9	2	0
0.5	0.9999999998878352	1.1216494399945987e-10	4	0
-1.0	0.9999922654776594	7.734552252003368e-6	5	0
-10.0	0.9999999998781014	1.2189849130095354e-10	15	0
10.0	0.9999999999251376	7.48623385504743e-11	7	0

Otrzymane wyniki obliczania miejsca zerowego $f(x)$ metodą stycznych

Przybliżenie początkowe	Miejsce zerowe	Wartość $g(x)$	Liczba iteracji	Błąd
-0.01	-9.801990000010284e-9	-9.801990096089293e-9	2	0
-1.0	-3.0642493416461764e-7	-3.0642502806087233e-7	5	0
-10.0	-3.784424932490619e-7	-3.784426364678097e-7	16	0
0.1	-1.4906619716777104e-8	-1.490661993898442e-8	3	0
1.0	NaN	NaN	NaN	2

Otrzymane wyniki obliczania miejsca zerowego $g(x)$ metodą stycznych

Podobnie jak dla *metody bisekcji* zauważamy, że odpowiednie dobranie przybliżenia początkowego pozwala znacznie ograniczyć liczbę wykonanych iteracji potrzebnych do znalezienia miejsca zerowego.

Wyraźnie widać, że choć dla funkcji $f(x)$ możemy ustalić przybliżenie początkowe x_0 leżące po obu stronach faktycznego miejsca zerowego funkcji, dokonując tego samego dla $g(x)$ i odpowiednio dużego przybliżenia x_0 nie będziemy w stanie otrzymać oczekiwanego wyniku, ponieważ wartości tej funkcji zbiegają do zera, co obrazuje jej wykres. Metoda *Newtona* będzie próbowała znaleźć miejsce zerowe, podążając w kierunku zbiegania funkcji do zera, zamiast prawidłowo znaleźć miejsce zerowe, jeśli niepoprawnie obierzemy punkt początkowy.

Finalnie przeprowadziliśmy eksperymenty dla *metody siecznych* dla różnych obranych przybliżeń początkowych. Ponownie ustalmy maksymalną liczbę iteracji $maxit = 30$.

x_0	x_1	Miejsce zerowe	Wartość $g(x)$	Liczba iteracji	Błąd
0.8	0.99	0.9999951828950254	4.817116576738556e-6	2	0
0.0	0.5	0.9999998133327657	1.8666725165594755e-7	5	0
-1.0	-0.5	0.9999999251013865	7.489861642007156e-8	7	0
-10.0	-5.5	0.9999998310288395	1.6897117482983504e-7	15	0
2.0	1.5	1.0000034269838276	-3.4269779555229363e-6	5	0
3.0	3.1	0.9999999938646222	6.135377761395944e-9	20	0
3.0	4.0	1.0000001383119406	-1.383119310194303e-7	21	0

Otrzymane wyniki obliczania miejsca zerowego $f(x)$ metodą siecznych

x_0	x_1	Miejsce zerowe	Wartość $g(x)$	Liczba iteracji	Błąd
-0.1	-0.01	-9.42098181330557e-6	-9.421070568621977e-6	2	0
-1.0	-0.5	-1.2229958402039555e-7	-1.2229959897758473e-7	6	0
-1.0	-0.9	-3.304447675535891e-8	-3.304447784729637e-8	7	0
-10.0	-6.0	-1.8112061236315096e-6	-1.8112094041021026e-6	17	0
0.0	0.5	0.0	0.0	1	0
5.0	10.0	14.836777722716038	5.3432918524983584e-6	7	0
132.5	132.25	133.3877754930358	1.568704635593728e-56	1	0

Otrzymane wyniki obliczania miejsca zerowego $g(x)$ metodą siecznych

Zauważamy wpływ dobrania punktów początkowych x_0 i x_1 na liczbę iteracji koniecznych do wykonania w celu wyznaczenia prawidłowego wyniku. Ponownie dla funkcji $g(x)$ zauważamy, że jej przebieg sprzyja niepoprawnemu określeniu prawidłowego miejsca zerowego w wypadku nieumiejętnego dobrania x_0 i x_1.

Podobnie jak w przypadku *metody Newtona* odpowiednio duże przybliżenie startowe sprawi, że metoda uniemożliwi otrzymanie prawidłowych wyników.

Reasumując, spostrzegamy, że aby móc otrzymać prawidłowe wyniki przy użyciu *metody siecznych i stycznych* konieczne jest ustalenie początkowych przybliżeń x_0 lub x_0, x_1 możliwe mało oddalonych od faktycznego miejsca zerowego badanej funkcji.

Także w wypadku *metody bisekcji* konieczne jest prawidłowe dobranie przedziału początkowego, w ramach którego będą dokonywane obliczenia. Podobnie najlepsze efekty przeprowadzonych obliczeń uzyskamy, dobierając możliwie wąski przedział, w którym miejsce zerowe funkcji znajduje się możliwie blisko środka tego przedziału.

1.7 Podsumowanie badań

Aby otrzymać prawidłowy wynik podczas obliczania miejsca zerowego danej funkcji nieliniowej musimy posiadać pewną wiedzę na temat jej przebiegu. Dotyczy to zarówno *metody bisekcji*, jak i *metody stycznych* czy *metody siecznych*.

Jak bowiem zauważyliśmy, istnieje ryzyko, że dla pewnych funkcji nieliniowych nieprawidłowe określenie przedziału początkowego (dla *metody bisekcji*) bądź przybliżeń początkowych (dla metod *stycznych i siecznych*) może skutkować skrajnie niepoprawnymi wynikami obliczeń.

Aby skutecznie wyznaczyć miejsce zerowe funkcji nieliniowej przy skorzystaniu z powyższych metod należy określić, w którym miejscu funkcji może wystąpić miejsce zerowe, a następnie możliwe najbardziej zawęzić przedziały, w których poszukujemy miejsca zerowego (dla *metody bisekcji*) bądź też obrać odpowiednio mało oddalone od miejsca zerowego przybliżenia początkowe (dla metod *stycznych i siecznych*).