Obliczenia Naukowe

Lista 3

Jakub Gogola
236412

20 listopada 2018

1 Zadanie I

1.1 Problem

W zadaniu 1. należało zaimplementować w języku `Julia` algorytm wyznaczający miejsca zerowe danej funkcji za pomocą metody bisekcji, znanej również pod nazwą metody połowienia przedziału.

1.2 Algorytm

```
1:  function MBISEKCJI(f, a, b, delta, epsilon)
2:      r ← 0
3:      v ← 0
4:      it ← 0
5:      err ← 0
6:      f_a ← f(a)
7:      f_b ← f(b)
8:      if SGN(f_a) = SGN(f_b) then
9:          err ← 1
10:         return (r, v, it, err)
11:     end if
12:     mid ← b − a
13:     while mid > epsilon do
14:         it ← it + 1
15:         mid ← mid/2
16:         r ← a + mid
17:         v ← f(r)
18:         if |mid| < delta or |v| < epsilon then
19:             return (r, v, it, err)
20:         end if
21:         if SGN(v) ≠ SGN(f_a) then
22:             b ← r
23:             f_b ← v
24:         else
25:             a ← r
26:             f_a ← v
27:         end if
28:     end while
29: end function
```

Opis parametrów:
`f` - funkcja $f(x)$ zadana jako anonimowa funkcja,
`a`, `b` - końce przedziału początkowego,
`delta`, `epsilon` - dokładność obliczeń,

Dane wynikowe:
`r` - przybliżenie pierwiastka równania $f(x) = 0$,

`v` - wartość $f(r)$,
`it` - liczba wykonanych iteracji,
`err` - sygnalizacja błędu: 0 - brak błędu, 1 - funkcja nie zmienia znaku na przedziale $[a, b]$.

1.3 Opis algorytmu

Ważnym założeniem, pozwalającym stosować metodę bisekcji do znalezienia rozwiązania równania nieliniowego, jest fakt, że zadana funkcja f jest ciągła na przedziale $[a, b]$ oraz $f(a)f(b) < 0$ (funkcja f zmienia na tym przedziale znak). Na początku działania algorytmu sprawdzane jest, czy funkcja zmienia znak na zadanym przedziale (linie 8-11). Jeżeli nie, to algorytm kończy działanie z kodem błędu 1. W przeciwnym wypadku rozpoczyna się wykonywanie pętli `while`, gdzie warunkiem końca jest osiągnięcie przedziału mniejszego od `epsilon` (o szerokości mniejszej od `epsilon`). W każdym przebiegu pętli jest wyznaczany środek bieżącego przedziału (linie 15-16) oraz liczona wartość funkcji f dla wyznaczonego środka (linia 17). W tym miejscu należy zwrócić uwagę na sposób wyznaczania środka przedziału. W celu uniknięcia błędów, które mogłyby powstać przy wyliczaniu przedziału za pomocą wzoru $mid \leftarrow \frac{1}{2}(b - a)$, najpierw jest wyliczana wartość mid, a następnie ta wartość jest dodawana do lewego końca bieżącego przedziału. W przypadku, jeżeli osiągnięto zadaną precyzję dla szerokości przedziału lub dla wartości funkcji (wartość f jest wystarczająco bliska 0) to wtedy zwracamy rezultat obliczeń (linie 18 - 20). Jeżeli algorytm nie kończy jeszcze działania to sprawdzamy, na którym z dwóch podprzedziałów funkcja f zmienia znak (linie 21-27) i ten podprzedział przyjmujemy jako nowy przedział, w którym będziemy szukać przybliżenia miejsca zerowego.

Jeszcze jedną rzeczą wartą uwagi, poza sposobem obliczania środka przedziału, jest to, jak sprawdzany jest znak funckji. W przedstawionym algorytmie zastosowano następujący sposób:

---

if $\text{SGN}(f_a) = (\neq)\ \text{SGN}(f_b)$ **then**
    ...
end if

Taka, a nie inna konstrukcja powyższego warunku, wynika z faktu, że badanie znaku funkcji za pomocą nierówności $f_a f_b < 0$ mogłoby doprowadzić do powstania niedomiaru lub niedomiaru podczas wykonywania zbędnej operacji mnożenia, czego unikamy w wypadku sposobu zastosowanego w prezentowanej implementacji algorytmu bisekcji.

2 Zadanie II

2.1 Problem

W zadaniu 2. należało zaimplementować w języku `Julia` algorytm wyznaczający miejsce zerowe danej funkcji za pomocą metody stycznych (metody Newtona). Metoda ta jest zdefiniowana następująco:

2.2 Algorytm

Opis parametrów:
`f`, `pf` - funkcja $f(x)$ oraz $f'(x)$ zadane jako anonimowe funkcje,
`x0` - przybliżenie początkowe,
`delta`, `epsilon` - dokładność obliczeń,
`maxit` - maksymalna dopuszczalna liczba iteracji

Dane wynikowe:
`r` - przybliżenie pierwiastka równania $f(x) = 0$,
`v` - wartość $f(r)$,
`it` - liczba wykonanych iteracji,
`err` - sygnalizacja błędu: 0 - brak błędu, 1 - nie osiągnięto wymaganej dokładności w `maxit` iteracji, 2 - pochodna bliska 0.

2.3 Opis algorytmu

Metoda Newtona, znana również pod nazwą metody stycznych (ta nazwa została użyta przy implementacji), wykorzystuje do obliczenia miejsca zerowego funkcji f tak zwaną *metodę linearyzacji*,

```
1: function MSTYCZNYCH(f, pf, x_0, delta, epsilon, maxit)        ▷ Oznaczenie: pf ← f'
2:     r ← x_0
3:     v ← f(r)
4:     it ← 0
5:     err ← 0
6:     if |f'(r)| < epsilon then
7:         err ← 2
8:         return (r, v, it, err)
9:     end if
10:    for it ← 1 to maxit do
11:        x_1 ← (r − v) / f'(r)
12:        v ← f(x_1)
13:        if |x_1 − r| < delta or |v| < epsilon then
14:            r ← x_1
15:            return (r, v, it, err)
16:        end if
17:        r ← x_1
18:    end for
19:    err ← 1
20:    return (r, v, it, err)
21: end function
```

czyli zastąpienie funkcji f funkcją liniową. Jest nią suma dwóch początkowych składników we wzorze Taylora dla funkcji f. Jeżeli

$$f(x) = f(c) + f^{'}(c)(x-c) + \tfrac{1}{2!}f^{''}(c)(x-c)^2 + \dots$$

to w wyniku zastosowania metody linearyzacji otrzymujemy funkcję liniową l:

$$l(x) = f(c) + f^{'}(c)(x-c).$$

Jest to dobre przybliżenie funkcji f w pobliżu argumentu c.

Do obliczania kolejnych przybliżeń pierwiastka r zadanej funkcji f wykorzystywany jest następujący wzór:

$$x_{n+1} = x_n - \tfrac{f(x_n)}{f'(x_n)}$$

gdzie $n \leqslant 0$.

Algorytm rozpoczyna swoje działanie od sprawdzenia czy pochodna funkcji f nie jest bliska 0 (czyli, na potrzeby tej metody numerycznej, mniejsza od wartości *epsilon*). Jeżeli jest, to algorytm kończy działanie z kodem błędu 2. W przeciwnym przypadku algorytm próbuje w zadanej liczbie iteracji *maxit* (linie 10-18) wyznaczyć miejsce zerowe zadanej funkcji f. Stosowany jest tutaj wzór wynikający z samej definicji. W liniach 11-12 wyliczana jest wartość x_{n+1} oraz wartość f dla tego argumentu. Następnie (linie 13-15), jeżeli otrzymane przybliżenie miejsca zerowego jest już wystarczająco dokładne, to algorytm kończy działanie zwracając otrzymane wyniki. W przeciwnym wypadku algorytm wykonuje kolejną iterację. Jeżeli nie osiągnięto wymaganej dokładności w zadanej liczbie iteracji *maxit*, to algorytm kończy działanie z kodem błędu 1.

3 Zadanie III

3.1 Problem

W zadaniu 3. należało zaimplementować w języku `Julia` algorytm znajdujący miejsce zerowe funkcji f za pomocą metody siecznych - metody iteracyjnej, pewnej modyfikacji metody Newtona.

3.2 Algorytm

Opis parametrów:
`f` - funkcja $f(x)$ zadana jako anonimowa funkcja,
`x0`, `x1` - przybliżenia początkowe,
`delta`, `epsilon` - dokładność obliczeń,

`maxit` - maksymalna dopuszczalna liczba iteracji

Dane wynikowe:
`r` - przybliżenie pierwiastka równania $f(x) = 0$,
`v` - wartość $f(r)$,
`it` - liczba wykonanych iteracji,
`err` - sygnalizacja błędu: 0 - brak błędu, 1 - nie osiągnięto wymaganej dokładności w `maxit` iteracji

```
1:  function MSIECZNYCH(f, x_0, x_1, delta, epsilon, maxit)
2:      r ← 0
3:      v ← 0
4:      it ← 0
5:      err ← 0
6:      f_{x_0} ← f(x_0)
7:      f_{x_1} ← f(x_1)
8:      for it ← 1 to maxit do
9:          if |f_{x_0}| > |f_{x_1}| then
10:             SWAP(x_0, x_1)
11:             SWAP(f_{x_0}, f_{x_1})
12:         end if
13:         s ← (x_1 − x_0) / (f_{x_1} − f_{x_0})
14:         x_1 ← x_0
15:         f_{x_1} ← f_{x_0}
16:         x_0 ← x_0 − f_{x_0} · s
17:         f_{x_0} ← f(x_0)
18:         if |x_1 − x_0| < delta or |f_{x_0}| < epsilon then
19:             r ← x_0
20:             v ← f_{x_0}
21:             return (r, v, it, err)
22:         end if
23:     end for
24:     err ← 1
25:     r ← x_0
26:     v ← f_{x_0}
27:     return (r, v, it, err)
28: end function
```

3.3 Opis algorytmu

Metoda siecznych, podobnie jak metoda Newtona, jest również metodą iteracyjną. Jednak, w odróżnieniu od odpisanej w poprzednim punkcie metody stycznych, nie wymaga on obliczania wartości pochodnej podczas każdej z iteracji wykonywanych przez algorytm. Dokonano tutaj zastąpienia pochodnej funkcji f *ilorazem różnicowym* mającym następującą postać:

$$f^{'}(x_n) \approx \frac{f(x_n)-f(x_{n-1})}{x_n-x_{n-1}}$$

Stąd, metoda siecznych, jest opisana przez następujący wzór:

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

gdzie $n \geqslant 1$. Ze względu, że iloraz różnicowy jest wyrażony przez x_n oraz x_{n-1}, jako argument algorytm obliczania miejsca zerowego metodą siecznych, otrzymuje na wejściu dwa punkty początkowe: x_0 i x_1. W przypadku tej metody styczna, wykorzystywana w metodzie Newtona, zostaje zastąpiona sieczną wykresu funkcji f.

Algorytm rozpoczyna działanie od wyliczenia wartości funkcji f dla zadanych x_0 i x_1 (linie 6-7). Następnie algorytm przechodzi do próby znalezienia przybliżenia pierwiastka f w zadanej liczbie iteracji *maxit*. Na początku każdego przebiegu pętli (linie 8-23) sprawdzany jest warunek, czy $|f_{x_0}| > |f_{x_1}|$ (linie 9-12). Jeżeli nierówność zachodzi, to następuje zamiana wartości zmiennych. Ma to na celu zapewnienie, że moduły funkcji f w kolejnych punktach x_n nie rosną. W następnej kolejności (linie 13-17) liczony jest iloraz różnicowy oraz wartość dla kolejnego x_{n+1} (linia 16). Jeżeli przybliżenie pierwiastka jest dokładne (odległość punktów przecięcia siecznej z wykresem

funkcji jest mniejsza od *delta*) lub wartość funkcji dla bieżącego argumentu jest bliska 0 (mniejsza od *epsilon*) to algorytm zwraca wynik (linie 18-22). Jeżeli nie - kontynuuje swoje działania. Jeśli nie otrzymano przybliżenia pierwiastka funckji f w zadanej liczbie iteracji *maxit*, to algorytm kończy działanie z kodem błędu 1.

4 Zadanie IV

4.1 Problem

Zadanie 4. polegało na wyznaczeniu pierwiastka równania $\sin x - (\frac{1}{2}x)^2 = 0$ wykorzystując zaimplementowanie wcześniej metody bisekcji, siecznych i stycznych (zadania 1-3).

4.2 Algorytm

Dla następujących danych początkowych zostały uruchomione zaimplementowane wcześniej algorytmy:

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

Ponadto, dla metod siecznych i stycznych została ustalona maksymalna liczba iteracji równa 50.

4.3 Wyniki

W wyniku działania programów otrzymano następujące wyniki dla poszczególnych implementacji metod obliczenia miejsc zerowych funkcji (x_0 to obliczone przybliżenie miejsca zerowego funkcji f:

Metoda	x_0	$f(x_0)$	Iteracje	Kod błędu
bisekcji	1.9337539672851562	$-2.7027680138402843 \cdot 10^{-7}$	16	0
Newtona	1.933753779789742	$-2.2423316314856834 \cdot 10^{-8}$	4	0
siecznych	1.933753644474301	$1.564525129449379 \cdot 10^{-7}$	4	0

Obliczenia zostały wykonane przy użyciu arytmetyki `Float64`.

4.4 Obserwacje

Najwięcej iteracji do znalezienia przybliżenia pierwiastka funkcji f potrzebowała metoda bisekcji (16 iteracji), a najmniej pozostałe dwie metody (4 iteracje). Metoda bisekcji zwróciła wartości najbliższe zadanej dokładności, a pozostałe dwie metody wyliczyły pierwiastki najbliższe pierwiastkowi rzeczywistemu.

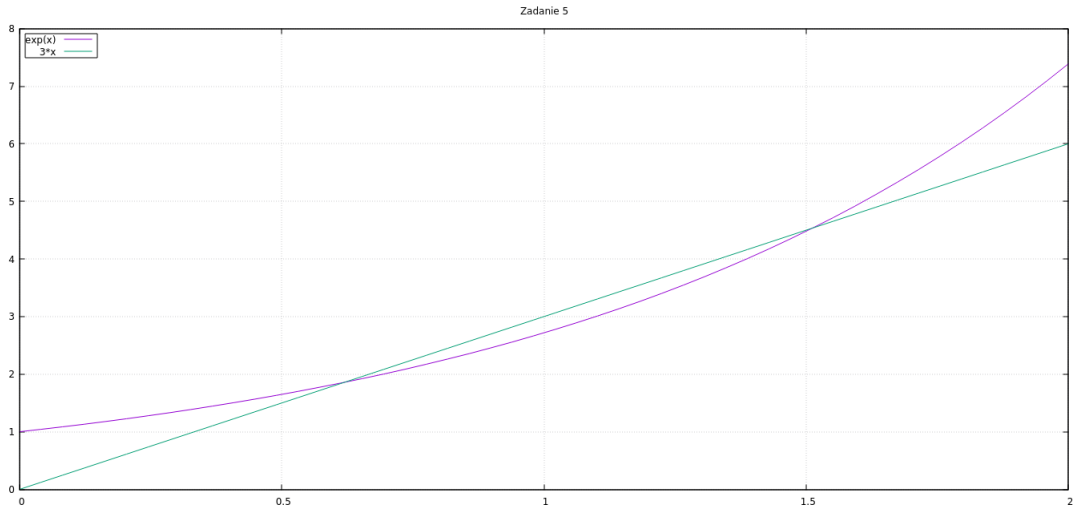
4.5 Wnioski

Uzyskane wyniki potwierdzają zbieżność każdej z metod determinowaną przez *współczynnik zbieżności* α. Dla metody bisekcji otrzymano zbieżność kwadratową ($\alpha = 1$), dla metody Newtona zbieżność liniową ($\alpha = \frac{1+\sqrt{5}}{2}$) i dla metody siecznych zbieżność nadliniową ($\alpha = 2$). Metoda bisekcji, mimo że zbiega najwolniej spośród wszystkich trzech metod, to jest to metoda najstabilniejsza, gdyż uzyskaliśmy w jej przypadku wyniki najbardziej zbliżone do zadanej na wejściu dokładności. Metody Newtona oraz siecznych są najszybciej zbieżne.

5 Zadanie V

5.1 Problem

Używając zaimplementowanej w zadaniu 1. metody bisekcji należało znaleźć wartości zmiennej x, dla których wykresy funkcji $y = 3x$ i $y = e^x$ przecinają się. Zadana dokładność: $\delta = 10^{-4}$, $\epsilon = 10^{-4}$.



Rysunek 1: Wykres funkcji $y = 3x$ i $y = e^x$ wygenerowany za pomocą pakietu `GNUPlot`

5.2 Algorytm

Na podstawie przedstawionego poniżej wykresu stwierdzamy, iż takich wartości zmiennej x należy szukać na przedziale $[0, 2]$. Problem znalezienia żądanych wartości zmiennej x sprowadzono do znalezienia miejsc zerowych funkcji $y = 3x - e^x$, które są jednocześnie miejscami przecięcia się wspomnianych funkcji. W celu znalezienia każdego z dwóch punktów przecięcia rozpatrzymy osobno dwa podprzedziały: $[0, 1]$ oraz $[1, 2]$ (wywołanie metody bisekcji dla przedziału $[0, 2]$ będzie skutkować zakończeniem wykonywania algorytmu z kodem błędu 1 (funkcja nie zmienia znaku na zadanym przedziale).

Do obliczenia wartości zmiennej x wykorzystano zaimplementowaną metodę bisekcji w zadaniu 1. Obliczenia wykonano w arytmetyce `Float64`.

5.3 Wyniki

W wyniku działania programu dla powyższych danych uzyskano następujące wyniki:

Przedział	x	Iteracje
$[0, 1]$	0.619140625	9
$[1, 2]$	1.5120849609375	13

5.4 Obserwacje

Wyniku działania programu otrzymano pierwiastki funkcji $y = 3x - e^x$, które są zgodne z punktami przecięcia widocznymi na wykresie.

5.5 Wnioski

Na podstawie analizy otrzymanych wyników i wykresu funkcji dochodzimy do wniosku, że otrzymano poprawne wyniki. Dla poprawnego wybrania podprzedziałów, na których wywołano metodę bisekcji dla zadanej funkcji, była konieczna analiza wykresu obu funkcji, o których mowa w poleceniu zadania.

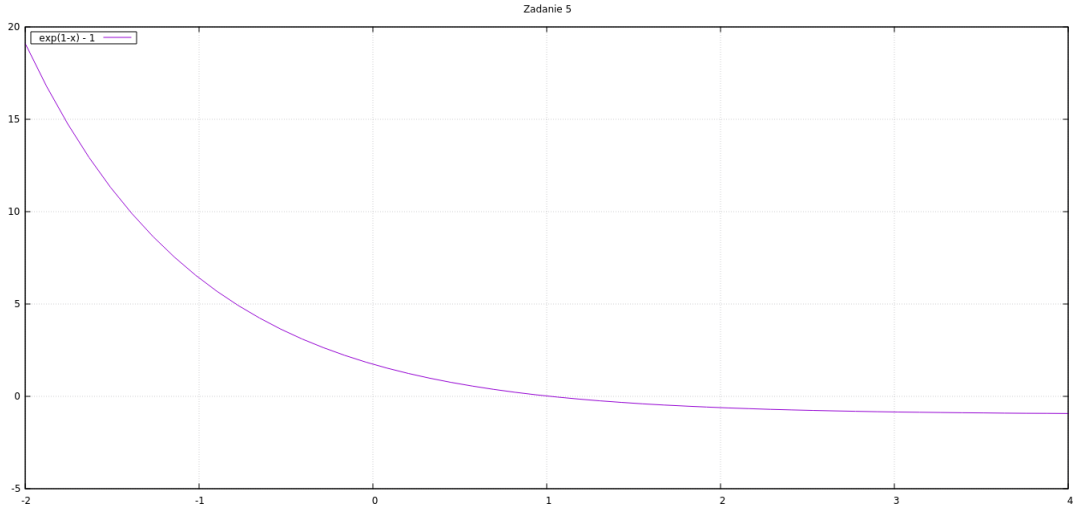
6 Zadanie VI

6.1 Problem

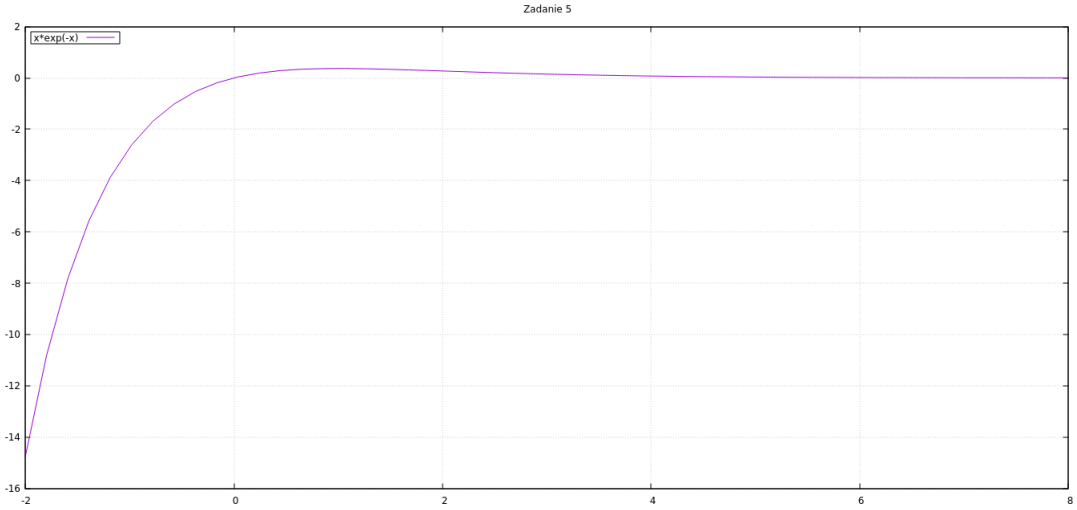
W zadaniu 6. należało znaleźć miejsca zerowe funkcji $f_1(x) = e^{1-x} - 1$ oraz $f_2(x) = xe^{-x}$ za pomocą metod bisekcji, Newtona i siecznych.

6.2 Algorytm

Poniżej znajdują się wykresy funkcji f_1 oraz f_2. Wykresy wykonano za pomocą pakietu `GNUPlot`. Na podstawie analizy przebiegu funkcji f_1 i f_2 przedstawionej na wykresie zostały wykonane



Rysunek 2: Wykres funkcji $f_1(x) = e^{1-x} - 1$



Rysunek 3: Wykres funkcji $f_2(x) = xe^{-x}$

eksperymenty dla różnych danych testowych.

6.3 Wyniki

Wyniki otrzymane dla każdej metody dla funkcji f_1:

Przedział	r	$f_1(r)$	Iteracje
$[0.0, 2.0]$	1.0	0.0	1
$[-2.0, 2.0]$	1.0	0.0	2
$[0.1, 1.8]$	0.9999992370605468	$7.6293974426988 \cdot 10^{-7}$	17
$[0.1, 1.2]$	0.9999938964843748	$6.10353425178 \cdot 10^{-6}$	14
$[-0.2, 1.8]$	0.9999969482421875	$3.051762469175045 \cdot 10^{-6}$	17
$[-5.0, 500.0]$	0.9999921917915344	$7.80823894963589 \cdot 10^{-6}$	24

Wyniki dla metody bisekcji

x_0	r	$f(r)$	Iteracje	Kod błędu
-1.0	0.9999922654776594	$7.73455225200336 \cdot 10^{-6}$	5	0
0.0	0.9999984358892101	$1.564112013019425 \cdot 10^{-6}$	4	0
1.1	0.99999999991094	$8.90598705893808 \cdot 10^{-11}$	3	0
2.0	0.9999999810061002	$1.899390000836831 \cdot 10^{-8}$	5	0
6.0	0.9999999573590406	$4.26409603182520 \cdot 10^{-8}$	147	0
8.0	NaN	NaN	1000	1
15.0	15.0	-0.9999991684712809	0	2

Wyniki dla metody Newtona

x_0	x_1	r	$f(r)$	Iteracje	Kod błędu
-2.0	2.0	1.0000000080618678	$-8.06186783997020 \cdot 10^{-9}$	8	0
-0.3	1.8	1.0000003464708873	$-3.464708272504779 \cdot 10^{-7}$	6	0
0.1	1.3	0.9999999935820667	$6.4179332959213 \cdot 10^{-9}$	5	0

Wyniki dla metody siecznych

Wyniki otrzymane dla każdej metody dla funkcji f_2:

Przedział	r	$f_1(r)$	Iteracje
$[-0.5, 0.5]$	0.0	0.0	1
$[-0.7, 0.4]$	$4.5776367188509 \cdot 10^{-6}$	$4.577615764140961 \cdot 10^{-6}$	16
$[-10.0, 100.0]$	45.0	$1.288133361247227 \cdot 10^{-18}$	1

Wyniki dla metody bisekcji

x_0	r	$f(r)$	Iteracje	Kod błędu
-1.0	$-3.064249341646176 \cdot 10^{-7}$	$-3.064250280608723 \cdot 10^{-7}$	5	0
-0.4	$-1.844031330942592 \cdot 10^{-8}$	$-1.84403136494710 \cdot 10^{-8}$	4	0
0.0	0.0	0.0	1	0
1.0	1.0	0.36787944117144233	0	2
6.0	14.97432014974184	$4.69983382720811 \cdot 10^{-6}$	8	0
8.0	14.636807965014	$6.43815521984328 \cdot 10^{-6}$	6	0
15.0	15.0	$4.58853480752738 \cdot 10^{-6}$	0	2

Wyniki dla metody Newtona

x_0	x_1	r	$f(r)$	Iteracje	Kod błędu
-2.0	2.0	14.294924723787231	$8.8506454983386 \cdot 10^{-6}$	15	0
-0.3	1.8	14.661140570698615	$6.29383436678240 \cdot 10^{-6}$	14	0
0.1	1.3	$2.482998842799128 \cdot 10^{-7}$	$2.4829982262708 \cdot 10^{-7}$	5	0
-2.0	6.0	14.812321857602736	$5.46655212223931 \cdot 10^{-6}$	12	0
-10.0	10.0	9.999999958776927	0.0004539993144685704	1	0
10.0	100.0	100.0	$3.720075976020836 \cdot 10^{-42}$	1	0

Wyniki dla metody siecznych

6.4 Obserwacje

Zauważamy, że metoda bisekcji, niezależnie od poprawnie wybranego przedziału (czyli takiego, na którym funkcja zmienia znak), zawsze znajduje miejsce zerowe z określoną dokładnością. Wybranie przedziału, dla którego miejsce zerowe leży w jego środku, powoduje, że metoda znajduje je niemal natychmiastowo (w 1-2 iteracjach), a znalezione rozwiązanie jest dokładne. Można zauważyć, że dla f_1 zawsze otrzymano dobre przybliżenie (zmieniała się jedynie liczba iteracji), natomiast dla f_2 metoda ta, dla niektórych przedziałów, dawała kompletnie niepoprawną wartość już po jednej iteracji i kończyła działanie.

W przypadku metody Newtona zawuważalny jest znaczny wzrost szybkości wyznaczania przybliżenia pierwiastka r dla danej funkcji w porównaniu z metodą bisekcji. Dla funkcji f_1 przekazanie algorytmowi wartości początkowej $x_0 \in (1, \infty]$ powoduje drastyczny wzrost liczby potrzebnych iteracji do wyznaczenia przybliżenia miejsca zerowego. W przypadku funkcji f_2 przekazanie do algorytmu $x_0 = 1$ powoduje natychmiastowe zakończenie działa programu, gdyż $f_2^{'}(x_0)$ jest bliska wartości 0. Wzrost wartości x_0 powoduje zmniejszanie się dokładności wyznaczonego przybliżenia. Dla wartości przybliżeń zbyt odległych od faktycznego pierwiastka, metoda stycznych nie jest w stanie osiągnąć poprawnego przybliżenia.

W przypadku metody siecznych wybranie początkowych wartości x_0, x_1 zbyt odległych od faktycznej wartości pierwiastka funkcji powoduje, że metoda nie jest w stanie osiągnąć przybliżenia

o wymaganej dokładności i zwraca wartość bardzo odległą od rzeczywistej (analogicznie jak w przypadku metody stycznych). Im bardziej ścisły przedział wokół rzeczywisstego miejsca zerowego, tym mniej potrzeba iteracji do obliczenia przybliżenia z zadaną dokładnością.

6.5 Wnioski

Na podstawie analizy otrzymanych wyników można stwierdzić, że metoda bisekcji jest zbieżna globalnie i jej zbieżność nie zależy od wybranego przedziału.

Fakt, iż metoda Newtona jest szybsza od metody bisekcji wynika wynika z jej lepszej zbieżności - metoda bisekcji ma zbieżność kwadratową, a stycznych - liniową (zostało do bardziej szczegółowo opisane w zadaniu 4.).

Podobnie, jak metoda Newtona, metoda siecznych jest o wiele szybsza od metody bisekcji, ponieważ posiada ona złożoność nadliniową.