METODY NUMERYCZNE – LABORATORIUM
Zadanie 1 – rozwiązywanie równań nieliniowych

Opis rozwiązania

1. Pierwszą zaimplementowaną przeze mnie metodą było rozwiązywanie równań nieliniowych przy pomocy metody bisekcji.

Polega ona na dzieleniu badanego przedziału argumentów funkcji na pół, a następnie przejściu do tego, który zawiera miejsce zerowe. W wyniku kolejnych iteracji tej czynności w końcu zostaje wyznaczone szukane rozwiązanie (chyba że tak jak w moim programie – istnieje warunek ograniczający działanie programu poprzez np. maksymalną liczbę iteracji).

Algorytm:

- wybranie określonego kryterium zatrzymania programu – określonej liczby iteracji lub warunek dokładności obliczeń i początkowe przedziału <a,b>.
- sprawdzenie warunków stopu i czy znaki na krańcach przedziału są różne,
- wyznaczenie środka przedziału *c*, sprawdzenie czy jego wartość jest równa 0,
- jeżeli nie to porównanie znaków środka przedziału z jego krańcami. Jeżeli przeciwny znak ma lewy kraniec przedziału, to nowym sprawdzanym przedziałem staje się <a,c>, a <c,b> odrzucamy,
- w przeciwnym razie nowym przedziałem staje się <c,b>, a przedział <a,c> odrzucamy,
- powrót do 2 kroku.

2. Drugą sprawdzaną metodą dla mojej grupa zadaniowej (7) był wariant nr6 – wyznaczanie minimum funkcji z zadaną dokładnością metodą złotego podziału.

Polega ona również na dzieleniu sprawdzanego przedziału na kolejne podprzedziały, wykorzystuje jednak nie ich podział na pół, lecz tzw. złotą proporcję dzielenia odcinka w ten sposób, żeby stosunek długości całego odcinka (przedziału) do dłuższej części odcinka (przedziału) był taki sam jak stosunek dłuższej części odcinka (przedziału) do krótszej.

Algorytm:

- sprawdzenie czy badany przedział funkcji jest unimodalny (czyli czy jest ciągły i warunku koniecznego istnienia ekstremum),
- wybranie określonego kryterium dokładności ε zatrzymania programu, warunków stopu,
- wyznaczenie wartości x1 i x2: x1 = b-q*(b-a) ; x2 = a+q*(b-a), gdzie: $q = \frac{\sqrt{5}-1}{2}$
- sprawdzenie czy osiągnięto warunek stopu,
- obliczenie wartości f(x1) i f(x2) (lub tylko jednej z nich - w iteracjach późniejszych niż pierwsza),
- wyeliminowanie podprzedziału <a,x1> jeżeli wartość f(x1) > f(x2) lub gdy f(x1) < f(x2) - podprzedziału <b,x2>,
- ustalenie nowych wartości a i b zależnie od wyeliminowanych wcześniej podprzedziałów,
- sprawdzenie czy |a-b| < ε, jeżeli tak to zakończenie algorytmu, w przeciwnym razie powrót do kroku nr 3.

Wyniki

1) Metoda wyznaczania rozwiązań równań nieliniowych poprzez bisekcję:

Tabela 1.1 warunek stopu – określona, maksymalna liczba iteracji (i = 50).

Parametry \ Funkcja	x^4-7x^2+6x	2cos(x)-6sin(x)	2^x-10	2^x-10cos(x-20)
Lewy kraniec przedziału *a*	-1,5	0	-2,2	1
Prawy kraniec przedziału *b*	0,8	2	7,3	5,45
Przybliżenie wartości *c* i *f(c)*	0,0001	0,0001	0,0001	0,0001
Wykonana liczba iteracji *i*	16	16	19	15
Maksymalna liczba iteracji	50	50	50	50
Czy osiągnięto warunek stopu	Nie	Nie	Nie	Nie
Wyznaczone miejsce zerowe	0,0000	0,3217	3,3219	2,2321
Wartość *f(c)*	0,0000	0,0000	0,0000	0,0000

Tabela 1.2 warunek stopu – dokładność obliczeń (ε = 0,01).

Parametry \ Funkcja	x^4-7x^2+6x	2cos(x)-6sin(x)	2^x-10	2^x-10cos(x-20)
Lewy kraniec przedziału *a*	-1,5	0	-2,2	1
Prawy kraniec przedziału *b*	0,8	2	7,3	5,45
Zadana dokładność ε	0,01	0,01	0,01	0,01
Wykonana liczba iteracji *i*	8	8	10	9
Czy osiągnięto warunek stopu	Nie	Tak	Tak	Tak
Wyznaczone miejsce zerowe	0,00	0,32	3,32	2,23
Wartość *f(c)*	0,00	0,01	-0,01	-0,08

Tabela 1.3 warunek stopu – dokładność obliczeń (ε = 0,0000001).

Parametry \ Funkcja	x^4-7x^2+6x	2cos(x)-6sin(x)	2^x-10	2^x-10cos(x-20)
Lewy kraniec przedziału *a*	-1,5	0	-2,2	1
Prawy kraniec przedziału *b*	0,8	2	7,3	5,45
Zadana dokładność ε	0,0000001	0, 000001	0, 000001	0, 000001
Wykonana liczba iteracji *i*	25	25	27	26
Czy osiągnięto warunek stopu	Tak	Tak	Tak	Tak
Wyznaczone miejsce zerowe	1,937151 x 10^{-7}	0,321751	3.321928	2,232141
Wartość *f(c)*	-0,0000001	-0,0000002	0,0000002	-0,0000007

2) <u>Metoda wyznaczania minimum funkcji w przedziale poprzez złoty podział:</u>

Tabela 2.1 warunek stopu – określona, maksymalna liczba iteracji,

Parametry \ Funkcja	x^4-7x^2+6x	2cos(x)-6sin(x)	2^x-10	2^x-10cos(x-20)
Lewy kraniec przedziału *a*	-3,2	2,2	0	-0,2
Prawy kraniec przedziału *b*	-0,75	6,5	5,5	6,8
Zadana dokładność ε	0,0001	0,0001	0,0001	0,0001
Wykonana liczba iteracji *i*	23	24	24	25
Maksymalna liczba iteracji	50	50	50	50
Czy osiągnięto warunek stopu	Nie	Nie	Nie	Nie
Wyznaczony argument minimum	-2,0565	2,2000	0,0000	1,0103
Wyznaczona wartość minimum	-24,0573	-6,0280	-9,0000	-7,8876

Tabela 2.2 warunek stopu – zawężenie przedziału poszukiwań (ε = 0,01).

Parametry \ Funkcja	x^4-7x^2+6x	2cos(x)-6sin(x)	2^x-10	2^x-10cos(x-20)
Lewy kraniec przedziału *a*	-6.4	2,2	0	-0,2
Prawy kraniec przedziału *b*	-0,2	6,5	5,5	6,8
Zadana dokładność ε	0,01	0,01	0,01	0,01
Wykonana liczba iteracji *i*	23	23	23	24
Czy osiągnięto warunek stopu	Tak	Tak	Tak	Tak
Wyznaczone miejsce zerowe	-2,06	2,20	0,00	1,01
Wyznaczona wartość minimum	-24,06	-6,03	-9,00	-7,89

Tabela 2.3 warunek stopu – zawężenie przedziału poszukiwań użytkownika (ε = 0,0000001).

Parametry \ Funkcja	x^4-7x^2+6x	2cos(x)-6sin(x)	2^x-10	2^x-10cos(x-20)
Lewy kraniec przedziału *a*	-6.4	2,2	0	-0,2
Prawy kraniec przedziału *b*	-0,2	6,5	5,5	6,8
Zadana dokładność *ε*	0,0000001	0,0000001	0,0000001	0,0000001
Wykonana liczba iteracji *i*	23	23	23	24
Czy osiągnięto warunek stopu	Tak	Tak	Tak	Tak
Wyznaczone miejsce zerowe	-2,0565612	2,2000543	0,0000694	1,0103655
Wyznaczona wartość minimum	-24,0572787	-6,0278767	-8,9999519	-7,8876289

Wnioski:

a) Ogólne – dla obu metod:

- W przypadku warunku stopu polegającego na maksymalnej liczbie iteracji program posiada odgórnie przyjętą dokładność ε = 0,0001, ponieważ w przeciwnym wypadku, przy niefortunnym doborze przedziałów początkowych kolejne szukane przedziały różniły się o wielkości rzędu 10^{-20} (lub mniejsze i program dalej szukał 'idealnego' miejsca zerowego) a ilość iteracji sięgała rzędów setek, tysięcy i wciąż rosła. W przypadku gdyby chciano zmienić przybliżenie wyniku dla modelu z określoną liczbą iteracji należałoby zmienić to ustawienie wewnątrz programu.
- Podobny problem napotkałem przy drugim warunku stopu, tutaj jednak niemożliwe było wprowadzenie dokładności 'na sztywno', ale udało mi się znaleźć sposób na to, żeby program dopasowywał liczbę miejsc po przecinku, które ma zaokrąglić, do wartości dokładności *ε*. Rozwiązanie to opiera się na zaufaniu do obsługującego program (wpisuje on samodzielnie, na którym miejscu po przecinku w jego dokładności znajduje się cyfra znacząca), ale ponieważ takie rozwiązanie nie wymagało zbytniego skomplikowania obliczeń, a program jest używany w celach typowo akademickich - zdecydowałem się je wprowadzić.

b) Dla szukania miejsc zerowych funkcji nieliniowych metodą bisekcji (tabele 1.1, 1.2 i 1.3):

- W każdym z trzech przypadków rozważanej dokładności najwięcej iteracji wymagało znalezienie miejsca zerowego funkcji wykładniczej. Być może oznacza to, że zmiana argumentu '*x*' w tej funkcji wywołuje największą zmianę wartości *f(x)*. Dla pozostałych funkcji liczba iteracji różni się o 0 lub 1, a nawet dla najbardziej skrajnych ilości iteracji, różnica ta nigdy nie jest większa niż 3. Na podstawie tej obserwacji stwierdziłbym, że metoda bisekcji działa z dosyć podobną wydajnością dla każdego rodzaju funkcji (oprócz lekkiego wydłużenia l. iteracji dla f. trygonometrycznej).
- Porównując wyznaczone argumenty i wartości funkcji dla danej funkcji w każdym z trzech przypadków można zauważyć, że wartości te są wynikiem działania tego samego algorytmu, a różnice na poszczególnych ich pozycjach wynikają z przyjętego zaokrąglenia. W przypadku określonej, maksymalnej ilości iteracji obliczone wartości przyjmują idealnie wartość '0', ponieważ w ich przypadku program zakończył się przed osiągnięciem warunku stopu, więc ostatnie przyjęte przybliżenie przed zakończeniem programu zmieniło wartość w następujący sposób: np. dla funkcji wielomianowej: f(c) = 0,0002≈0,0000

c) Dla wyznaczania minimum funkcji metodą złotego podziału (tabele 2.1, 2.2 i 2.3):

- Pochodne funkcji zostały wprowadzone 'na sztywno'. Dla kilku przypadków, w których specjalnie chciałem znaleźć minimum w przedziale nie posiadającym warunku koniecznego istnienia ekstremum, program prawidłowo zwracał informację, że dany przedział nie jest unimodalny.
- W każdym z trzech przypadków rozważanej dokładności najwięcej iteracji wymagało znalezienie miejsca zerowego funkcji złożonej, ale różnica pomiędzy iteracją dla tej funkcji, a z pozostałymi wynosiła zazwyczaj 1 iterację, więc ciężko wyciągnąć z tej korelacji szerszy wniosek. Dla pozostałych funkcji liczba iteracji różni się o 0 lub 1, a nawet po uwzględnieniu f. złożonej - różnica dla najbardziej skrajnych ilości iteracji nigdy nie jest większa niż 2. Na podstawie tej obserwacji stwierdziłbym, że metoda złotego podziału działa z dosyć podobną wydajnością dla każdego rodzaju funkcji (oprócz prawdopodobnego delikatnego zwiększenia liczby iteracji dla f. złożonej).
- Porównując wyznaczone argumenty i wartości funkcji dla danej funkcji w każdym z trzech przypadków można zauważyć, podobnie jak dla metody bisekcji, że wartości te są wynikiem działania tego samego algorytmu, a różnice na poszczególnych ich pozycjach wynikają z przyjętego zaokrąglenia. Zgodnie z teorią najdokładniejszy wynik jest dla wariantu z najmniejszych zaokrągleniem.