Imię i nazwisko Meg Paskowski	Temat laboratorium Optymalizacja – metoda złotego podziału	Data oddania 13.06.2024r.	Data ćwiczeń 07.06.2024r.
Prowadzący dr hab. inż. Marcin Hojny		Grupa laboratoryjna 4	

1. Cel ćwiczenia

Celem laboratorium nr. 13 było zapoznanie się z pojęciem optymalizacji równań metodą złotego podziału zaimplementowanie tego algorytmu w wybranym języku programowania.

2. Wstęp teoretyczny

Metoda złotego podziału (złota proporcja, boska proporcja) → jest to metoda polegająca na podzieleniu odcinka na dwie części tak, aby stosunek długości dłuższej części do krótszej był taki sam jak całego odcinka do części dłuższej. Przy czym długość dłuższej części jest średnią geometryczną długości krótszej części i całego odcinka.

$$\frac{a+b}{a} = \frac{a}{b} = \varphi \quad (1)$$

Gdzie:

- φ to stała nazywana złotym współczynnikiem i wynosi ok. 1.61803398875

Głównym celem takiej optymalizacji jednowymiarowej jest znalezienie jej minimum na podanym przedziale [a, b].

Wybranie punktów początkowych x_1 i x_2 wewnątrz przedziału [a,b] zgodne jest ze wzorami (2).

$$X_1 = b - \varphi(b-a) \quad (2)$$
$$X_2 = a + \varphi(b-a)$$

Po ustaleniu punktów początkowych obliczane są wartości funkcji w tych punktach. Następnie dopóki różnica między dolną i górną granicą przedziału jest większa niż ustalone przez nas przybliżenie następuję porównanie wartości funkcji w tych punktach i przesuwanie dolnej lub górnej granicy przedziału [a,b] w zależności od tego, która wartość funkcji jest mniejsza. Po kolei następuję zaktualizowanie wartości x_1 i x_2.

$$\text{Gdy } f(x_1) > f(x_2) \text{ to } a = x_1,\ x_1 = x_2,\quad x_2 = a + \varphi\cdot(b-a) \quad (3)$$
$$\text{Gdy } f(x_2) > f(x_1) \text{ to } b = x_2,\ x_2 = x_1,\quad x_1 = b - \varphi\cdot(b-a) \quad (4)$$

3. Implementacja

W ramach ćwiczeń zaimplementowano program w języku C++, który implementuje metodę złotego podziału.

Na samym początku określane są zmienne stałe i globalne potrzebna do rozwiązania zadania. Liczba „PHI” przybliżona wartość złotego podziału, używana w tej metodzie optymalizacji natomiast „Elipson” określa dokładność, do której algorytm ma dążyć.

```
const double PHI = 0.61803398875;  // Golden ratio approximation
const double epsilon = 1e-5; // Desired precision
```

Funkcja „*functionToMinimize()”* to funkcja celu w której określamy funkcje, którą algorytm próbuje zminimalizować. Przyjmuje liczbę „x” dla której jest liczona wartość „y”.

```
// Function to minimize
double functionToMinimize(double x) {
```

```
    return (x - 2) * (x - 2);
};
```

Kolejna metoda „*goldenSectionSearch()”* to główna funkcja wykonująca algorytm wyszukiwania metodą złotego podziału w celu znalezienia minimum funkcji. Na początku działania funkcji zachodzi inicjalizacja zmiennych lokalnych „x1”, x2” „f1” „f2”. Zmienne „x1” i „x2” są początkowymi punktami wewnątrz przedziału [a, b], które dzielą ten przedział zgodnie z proporcją złotego podziału. „f1” i „f2” to wartości funkcji celu w tych punktach. Algorytm działa dopóki różnica między górną i dolną granicą przedziału [a, b] jest większa od zdefiniowanej precyzji „epsilon”. W pętli „while” obliczana jest wartość minimum funkcji za pomocą odpowiednich wzorów (2-4).

```
double goldenSectionSearch(double a, double b) {
    double x1 = b - PHI * (b - a);
    double x2 = a + PHI * (b - a);
    double f1 = functionToMinimize(x1);
    double f2 = functionToMinimize(x2);

    while ((b - a) > epsilon) {
        if (f1 > f2) {
            a = x1;
            x1 = x2;
            f1 = f2;
            x2 = a + PHI * (b - a);
            f2 = functionToMinimize(x2);
        }
        else {
            b = x2;
            x2 = x1;
            f2 = f1;
            x1 = b - PHI * (b - a);
            f1 = functionToMinimize(x1);
        }
    }
    return (a + b) / 2;

}
```

W głównej funkcji „main()” programu inicjalizowane są wartości graniczne „a” oraz „b” i wywoływana jest funkcja „*goldenSectionSearch*()”. Wynik wypisywany jest w konsoli z poprzedzającym komentarzem.

```
double a = 0; // Lower bound of search interval
double b = 4; // Upper bound of search interval

cout << "The minimum value is located at x = " << goldenSectionSearch(a, b) << endl;
```

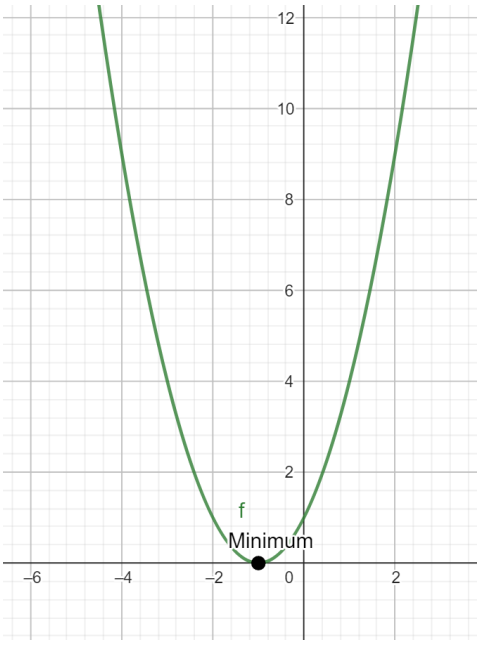
5. Testy jednostkowe i opracowanie wyników

Testy zostały przeprowadzone dla różnych funkcji następnie sprawdzone z wynikami uzyskanaymi na stronie „Wolframalpha.com” oraz przedstawione na„Geogebra.com”.

I. Test 1 dla funkcji $f(x) = (x+1)^2$
Dla przedziału [-5, 5]
Oczekiwany wynik programu: -1
Wynik funkcji „*goldenSectionSearch()”*: -0.999999

```
The minimum value is located at x = -0.999999
Press any key to continue . . .
```

Rysunek 1 Wyniki testu 1

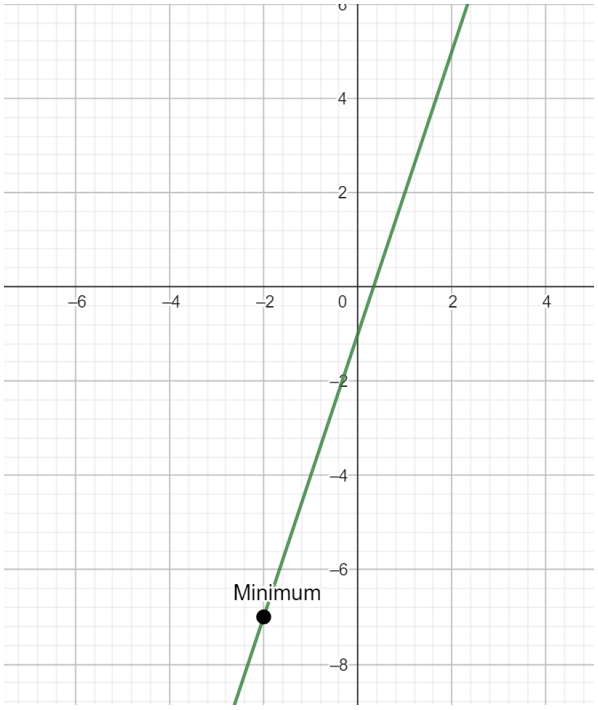


Rysunek 2 Wyniki przedstawione w programie "Geogebra" dla testu 1

II. Test 2 dla funkcji f(x) = 3x -1
Dla przedziału [-2; 3]
Oczekiwany wynik programu: -2
Wynik funkcji „*goldenSectionSearch()*”: -2

```
The minimum value is located at x = -2
Press any key to continue . . .
```

Rysunek 3 Wyniki testu 2



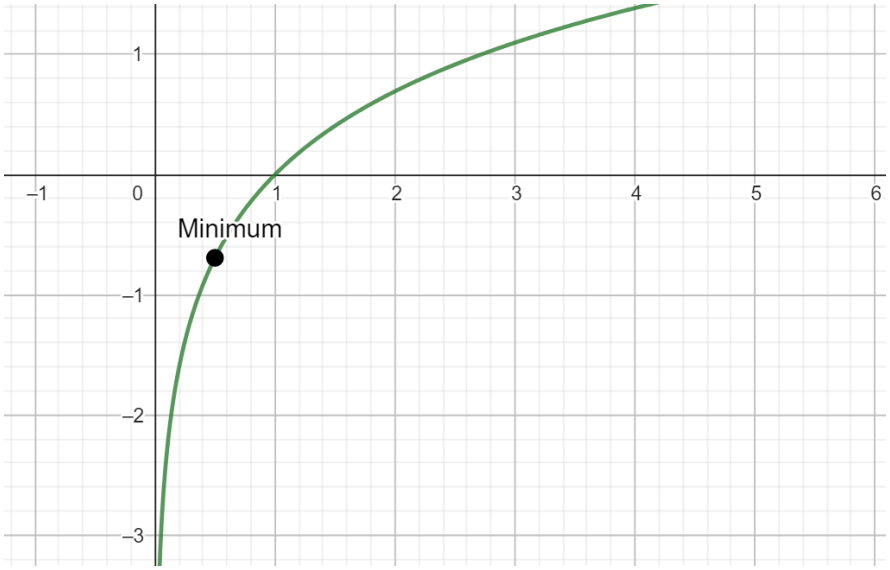
Rysunek 4 Wyniki przedstawione w programie "Geogebra" dla testu 2

III. Test 3 dla funkcji f(x) = lnx
Dla przedziału [0.5; 4]

Oczekiwany wynik programu: 0.5
Wynik funkcji „*goldenSectionSearch()*”: 0.500004
Wyniki uzyskane różnią się z uwagi na przyjętą prezyzję.

```
The minimum value is located at x = 0.500004
Press any key to continue . . .
```

Rysunek 5 Wyniki testu 3

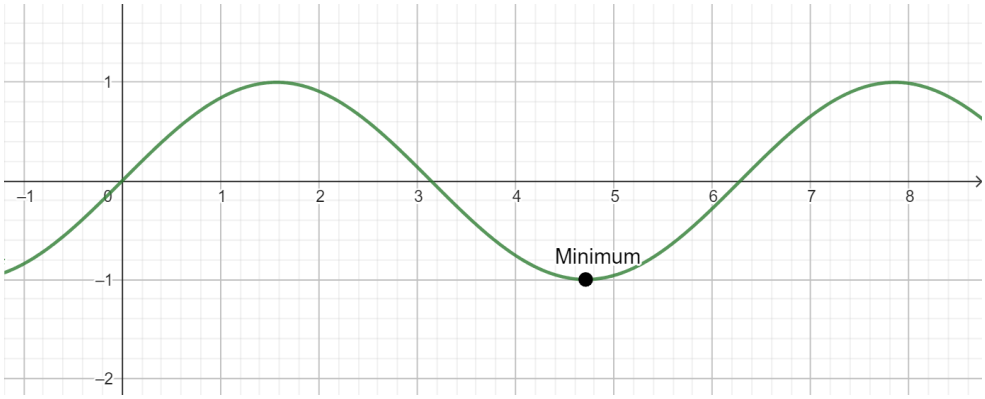


Rysunek 6 Wyniki przedstawione w programie "Geogebra" dla testu 3

IV. Test 4 dla funkcji f(x) = sinx
Dla przedziału [0; 2π]
Oczekiwany wynik programu: $x = \frac{3}{2}\pi \approx 4.71239$
Wynik funkcji „*goldenSectionSearch()*” $x = 4.71239$

```
The minimum value is located at x = 4.71239
Press any key to continue . . .
```

Rysunek 7 Wyniki testu 4



Rysunek 8 Wyniki przedstawione w programie "Geogebra" dla testu 4

6. Analiza wyników

Wyniki uzyskane z testów (Tabela 1) przeprowadzonych przy użyciu metody złotego podziału potwierdzają skuteczność tego algorytmu w znajdowaniu minimów funkcji w zadanych przedziałach. Porównanie wyników uzyskanych za pomocą programu napisanego w języku

C++ z oczekiwanymi wartościami oraz wynikami przedstawionymi na stronach takich jak „*WolframAlpha*” i w programie „*GeoGebra*” wykazało wysoką zgodność, co dowodzi poprawności implementacji.

Tabela 1 Wyniki przeprowadzonych testów

Numer testu	Uzyskany wynik	Oczekiwany wynik
1	-0.999999	-1
2	-2	-2
3	0.500004	0.5
4	4.71239	4.71239

Dla każdej testowanej funkcji uzyskano wynik bardzo zbliżony lub identyczny do oczekiwanego, co świadczy o dokładności i precyzji metody złotego podziału, nawet przy przyjęciu określonej tolerancji błędu (epsilon = 1e-5). Wartości minimalne funkcji zostały znalezione z bardzo małym błędem, co jest szczególnie widoczne w przypadkach funkcji bardziej skomplikowanych, takich jak funkcja logarytmiczna czy sinusoidalna.

7. Wnioski

Metoda złotego podziału okazała się być przydatna w praktyce, mogąc być stosowana w różnorodnych zadaniach optymalizacyjnych jednowymiarowych wymagających precyzyjnej lokalizacji minimum funkcji w określonych przedziałach. Przeprowadzone testy jednostkowe potwierdzają jej wysoką użyteczność i zgodność z teoretycznymi podstawami metody.

8. Źródła

- Prezentacja autorstwa dr hab. inż. Marcina Hojnego „Optymalizacja – metoda złotego podziału”.