Paulina Grabowska [redacted]	Metody numeryczne
Sprawozdanie 11 Optymalizacja jednowymiarowa	

1. Wstęp

Optymalizacja to proces znajdowania najlepszego możliwego rozwiązania w danej sytuacji, zgodnie z określonymi kryteriami i ograniczeniami. Celem optymalizacji jest maksymalizacja lub minimalizacja wybranej funkcji celu, takiej jak zysk, koszty, wydajność czy czas.

Optymalizacja wymaga precyzyjnego definiowania problemu, w tym określenia zmiennych decyzyjnych, funkcji celu oraz ograniczeń. Narzędzia matematyczne i algorytmy, takie jak programowanie liniowe, nieliniowe, dynamiczne czy algorytmy genetyczne, są często używane do rozwiązywania problemów optymalizacyjnych.

2. Opis metody numerycznej

Metoda złotego podziału jest techniką optymalizacji jednowymiarowej, która pozwala na znalezienie minimum (lub maksimum) funkcji w przedziale bez konieczności znajomości jej pochodnej. Nazwa tej metody pochodzi od złotej proporcji, która jest używana do efektywnego podziału przedziału poszukiwań i wynosi około 0,61803398 (czyli odwrotność ułamka $\frac{1+\sqrt{5}}{2}$). W zależności od źródeł można też spotkać informację, że liczba ta jest równa temu ułamkowi, a nie jego odwrotności, jednak w metodach optymalizacji zwykle stosuje się wcześniej wypisaną wartość.

Metoda złotego podziału rozpoczyna się od określenia przedziału $[a, b]$, w którym szukane będzie minimum funkcji $f(x)$. Następnie dwa punkty wewnętrzne x_1 oraz x_2 są wybierane w taki sposób, że:

$$x_1 = b - \phi(b - a)$$
$$x_2 = a + \phi(b - a) \quad (1)$$

Gdzie liczba ϕ jest wcześniej wspominaną złotą proporcją i wynosi około 0,61803398. Następnie obliczane są wartości funkcji w punktach $f(x_1)$ oraz $f(x_2)$, a ich wartości są ze sobą porównywane. Jeżeli:

- $f(x_1) < f(x_2)$, oznacza to, że minimum leży w przedziale *[a, x_2],* wówczas przestawiana jest wartość $b = x_2$
- $f(x_1) \geq f(x_2)$, oznacza to, że minimum leży w przedziale *[x_1, b],* wówczas przestawiana jest wartość $a = x_1$

Powyższe kroki od obliczania wartości punktów wewnętrznych są powtarzane, do momentu, aż długość przedziału *[a, b]* nie będzie dostatecznie mała (a więc równa zadanej tolerancji), co oznacza, że zostało odnalezione przybliżone położenie minimum.

Metoda złotego podziału jest szczególnie użyteczna, ponieważ jest efektywna i nie wymaga znajomości pochodnej funkcji. Dzięki specyficznemu sposobowi wybierania punktów wewnętrznych, zapewnia szybkie zbieganie do minimum.

3. Implementacja metody numerycznej

3.1. Funkcja *f* (*Fragment kodu 1*)

Funkcja ta pozwala na zadeklarowanie dla jakiej funkcji jednej zmiennej szukane będzie minimum funkcji na zadanym przedziale.

```
1. double f(double x) {
2.     return (1/x) ;
3. }
```

Fragment kodu 1: Funkcja *f*

3.2. Funkcja *optimazation* (*Fragment kodu 2*)

W tej funkcji deklarowana jest zmienna *phi* typu *const double*, która jest wartością złotej proporcji zadanej odwrotnością ułamka $\frac{1+\sqrt{5}}{2}$. Następnie deklarowane są punkty *x1* oraz *x2* – punkty wewnętrzne które podzielą przedział *[a, b]* zgodnie ze złotym podziałem według wzorów (1). Następnie obliczane są wartości funkcji w obu tych punktach.

W zaimplementowanej potem pętli *while* zadeklarowano warunewk stopu jako długość przedziału *[a*, b] mieszący się w zadanej tolerancji *tol*. W pętli porównywane są wartości *f1* oraz *f2*. Jeżeli wartość *f1* jest zmniejsza od *f2* oznacza to, że minimum funkcji znajduje się w przedziale *[a, x2],* a więc następują niezbędne aktualizacje wszystkich zmiennych – czyli wartość *x2* jest przypisywana do *b*, a *x2* jest przesuwane do *x1*, a następnie obliczane są nowe wartości *x1* oraz *f1*.

W przeciwnym przypadku minimum będzie znajdować w przedziale *[x1, b].* Ponownie następują niezbędne aktualizacje – *a* do *x1* oraz przesunięcie *x1* do *x2*, a następnie ponowienie obliczenie nowych wartości *x2* oraz *f2*.

Po zakończeniu pętli zwracana jest wartości będąca średnią z *a* i *b*, która to jest przybliżeniem położenia minimum funkcji.

```
 1. double optimization(double a, double b, double tol) {
 2.     const double phi = (1/((1 + sqrt(5.0)) / 2.0));
 3.
 4.     double x1 = b - phi * (b - a);
 5.     double x2 = a + phi * (b - a);
 6.     double f1 = f(x1);
 7.     double f2 = f(x2);
 8.
 9.     while (abs(b - a) > tol) {
10.         if (f1 < f2) {
11.             b = x2;
12.             x2 = x1;
13.             f2 = f1;
14.             x1 = b - phi * (b - a);
15.             f1 = f(x1);
16.         } else {
17.             a = x1;
18.             x1 = x2;
19.             f1 = f2;
20.             x2 = a + phi * (b - a);
21.             f2 = f(x2);
22.         }
23.     }
24.
25.     return (a + b) / 2.0;
26. }
```

Fragment kodu 2: Funkcja *optimaztion*

3.3. Funkcja *main* (*Fragment kodu 3*)

W funkcji main następuje deklaracja początku przedziału a jego końca b oraz zmienna tol, czyli tolerancja dokładności poszukiwanego minimum. Następnie wywoływana jest funkcji optymalizacyjna ze wszystkimi trzema parametrami. Wynik, czyli przybliżone minimum jest przypisywany do zmiennej do zmiennej *min_x* i następnie obliczana jest wartość funkcji w tym

punkcie minimum *min_y*. Wyświetlany jest wynik – punkt, w którym istnieje minimum na przedziale *[a, b]*.

```
1. int main() {
2.     double a = -4.0;
3.     double b = -0.5;
4.     double tol = 1e-6;
5.
6.
7.     double min_x = optimization(a, b, tol);
8.     double min_y = f(min_x);
9.
10.    cout << "\n\n\tminimum of the function on the interval [" << a << ", " << b << "] \n\tis
       in point: (" << min_x << "; " << min_y << ")" << endl;
11.
12.
13.    return 0;
14. }
```

Fragment kodu 3: funkcja *main*

4. Testy jednostkowe

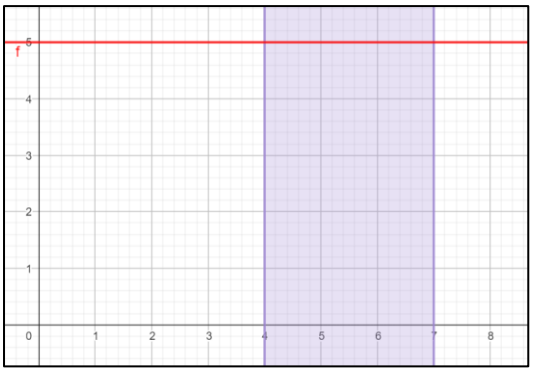
Dla wszystkich testów ustawiona wartość tolerancji wynosi 10^{-6}.

4.1. Test dla funkcji stałej

Wprowadzone dane:

- f(x) = 5;
- a = 4;
- b = 7.

Dokonano wizualizacji w programie Geogebra (*Rys.1a*) oraz wydruk z konsoli przedstawiony na rysunku *Rys. 1b*.



Rys. 1a: Wizualizacja wyniku w programie Geogebra

```
minimum of the function on the interval [4, 7]
is in point: (7; 5)
```

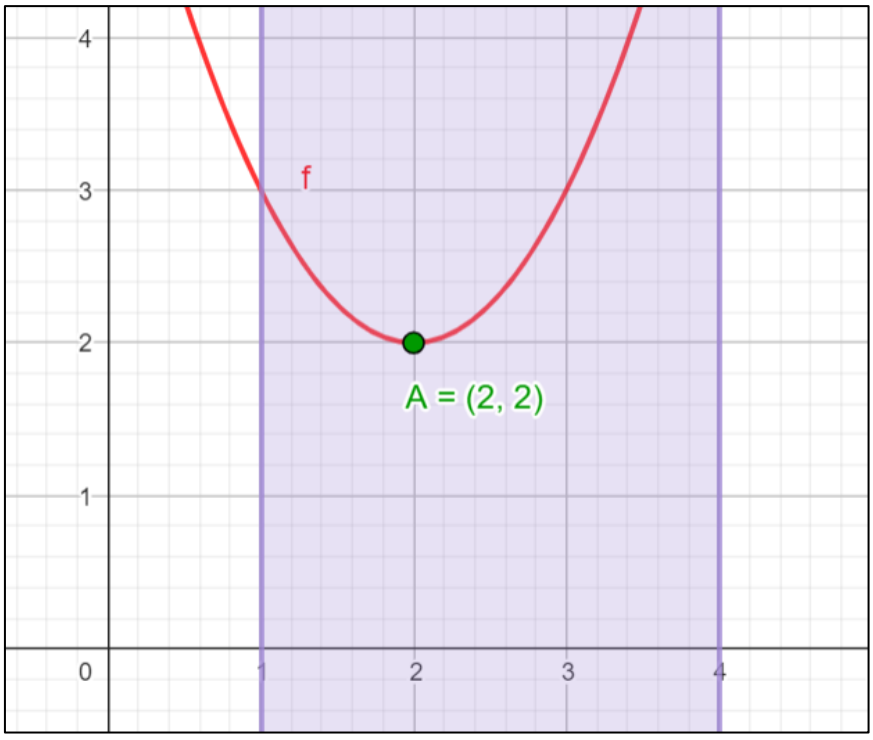
Rys. 1b: Wydruk z konsoli

4.2. Test dla funkcji potęgowej

Wprowadzone dane:

- f(x) = $(x-2)^2 + 2$;
- a = 1;
- b = 4.

Dokonano wizualizacji w programie Geogebra na którym zwizualizowano również oczekiwany wynik, który obliczono za pomocą programu WolframAlpha (*Rys.2a*) oraz wydruk z konsoli przedstawiony na rysunku *Rys. 2b*.



Rys. 2a: Wizualizacja wyniku w programie Geogebra

```
minimum of the function on the interval [1, 4]
is in point: (2; 2)
```

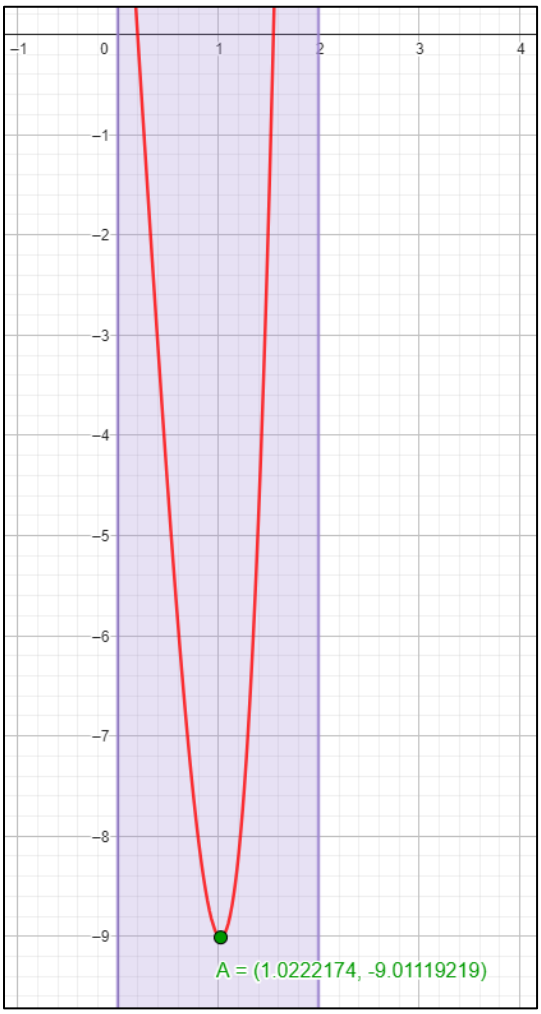
Rys. 2b: Wydruk z konsoli

4.3. Test dla funkcji wielomianowej

Wprowadzone dane:

- f(x) = $x^5 + 5x^3 - 3x^2 - 15x + 3$;
- a = 0;
- b = 2.

Dokonano wizualizacji w programie Geogebra na którym zwizualizowano również oczekiwany wynik, który obliczono za pomocą programu WolframAlpha (*Rys. 3a*) oraz wydruk z konsoli przedstawiony na rysunku *Rys. 3b*.



Rys. 3a: Wizualizacja wyniku w programie Geogebra

```
minimum of the function on the interval [0, 2]
is in point: (1.02222; -9.01119)
```

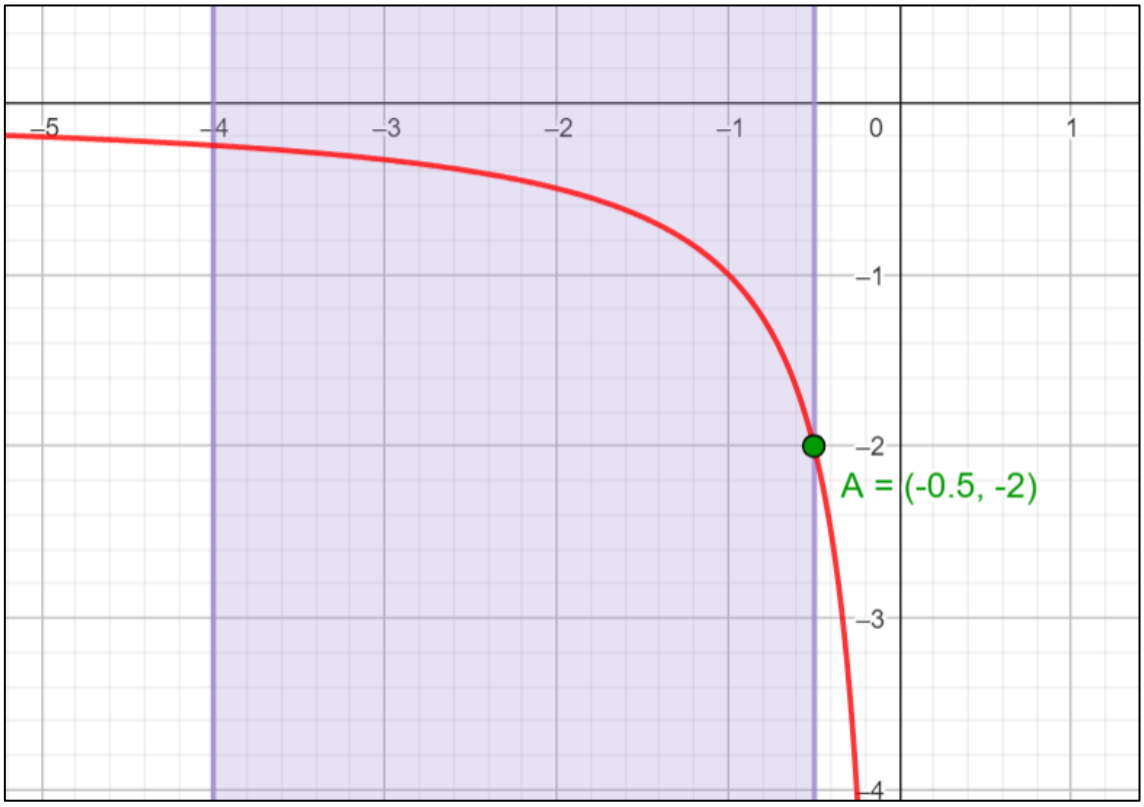
Rys. 3b: Wydruk z konsoli

4.4. Test dla funkcji homograficznej (z istniejącym minimum)

Wprowadzone dane:

- f(x) = $\frac{1}{x}$;
- a = - 4;
- b = - 0,5.

Dokonano wizualizacji w programie Geogebra na którym zwizualizowano również oczekiwany wynik, który obliczono za pomocą programu WolframAlpha (*Rys. 4a*) oraz wydruk z konsoli przedstawiony na rysunku *Rys. 4b*.



Rys. 4a: Wizualizacja wyniku w programie Geogebra

```
minimum of the function on the interval [-4, -0.5]
is in point: (-0.5; -2)
```

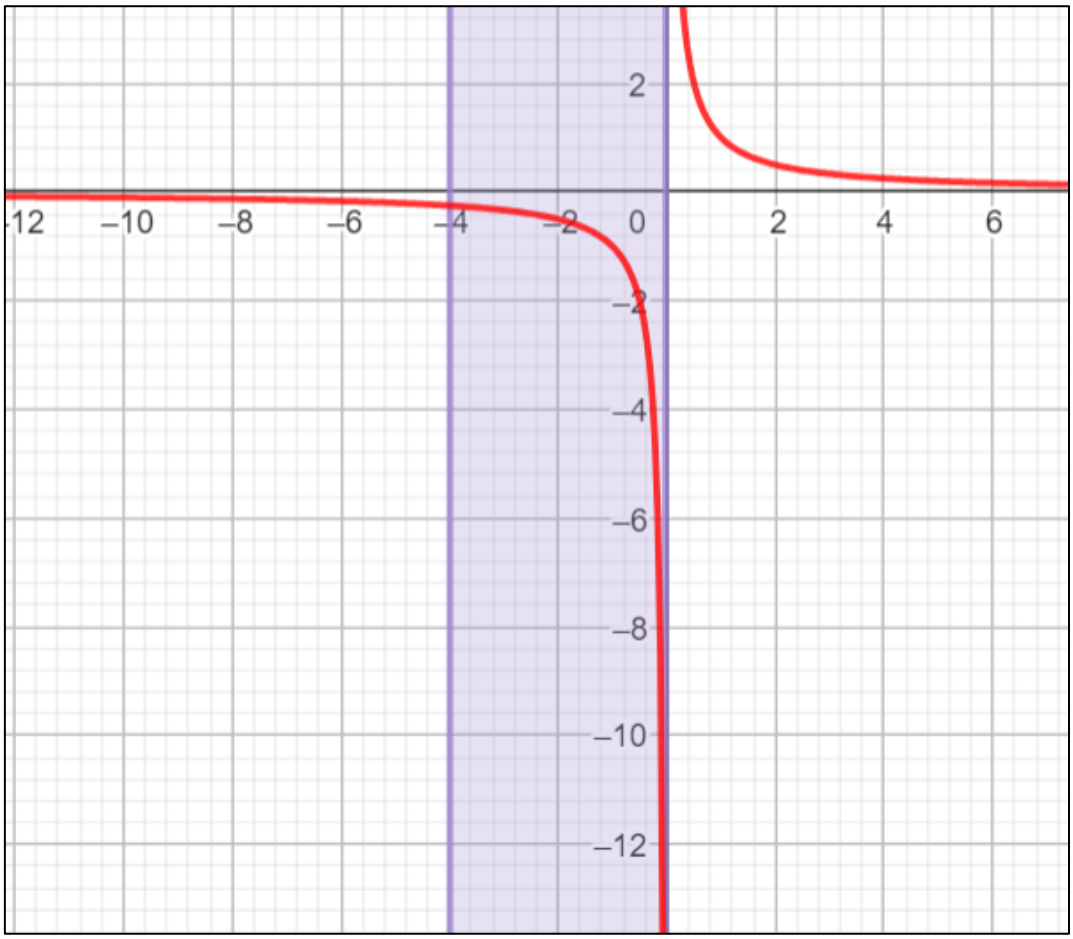
Rys. 4b: Wydruk z konsoli

4.5. Test dla funkcji homograficznej (bez istniejącego minimum)

Wprowadzone dane:

- f(x) = $\frac{1}{x}$;
- a = - 4;
- b = 0.

Dokonano wizualizacji w programie Geogebra (*Rys.5a*) oraz wydruk z konsoli przedstawiony na rysunku *Rys. 5b*.



Rys. 5a: Wizualizacja wyniku w programie Geogebra

```
minimum of the function on the interval [-4, 0]
is in point: (-4.10606e-07; -2.43542e+06)
```

Rys. 5b: Wydruk z konsoli

5. Opracowanie wyników

5.1. Test dla funkcji stałej

Testowanie metody złotego podziału na funkcji stałej nie ma sensu, ponieważ niezależnie od wyboru punktów, wartość funkcji będzie zawsze taka sama. To oznacza, że metoda nie będzie w stanie znaleźć jednego, wyróżnionego punktu, który jest minimum lub maksimum, ponieważ wszystkie punkty mają tę samą wartość. Jednak wartym zauważenia jest, jak algorytm zachowuje się w obliczu takiej funkcji – algorytm zwrócił punkt (7; 5) (*Tabela 1*), który to jest punktem końca przedziału.

Tabela 1: Zestawienie wyniku dla testu dla funkcji stałej.

Spodziewany wynik	Otrzymany wynik
Brak minimum	Punkt (7; 5)

5.2. Test dla funkcji potęgowej

Testowanie metody złotego podziału na funkcji potęgowej, jest doskonałym przykładem skuteczności tego algorytmu. Funkcja potęgowa użyta w teście ma wyraźne minimum w punkcie (2; 2), co sprawia, że jest idealna do testowania metod optymalizacyjnych (*Tabela 2*). Wynik zwrócony przez algorytm zgadza się z wynikiem oczekiwanym.

Tabela 2: Zestawienie wyniku dla testu dla funkcji potęgowej.

Spodziewany wynik	Otrzymany wynik
Punkt (2; 2)	Punkt (2; 2)

5.3. Test dla funkcji wielomianowej

Testowanie optymalizacji jednowymiarowej na funkcji wielomianowej, jest świetnym przykładem, jak algorytm radzi sobie z bardziej złożonymi funkcjami, które mają jednoznaczne minima w określonych przedziałach. Otrzymany wynik (Tabela 3) pokrywa się z oczekiwanym.

Tabela 3: Zestawienie wyniku dla testu dla funkcji wielomianowej.

Spodziewany wynik	Otrzymany wynik
Punkt (1.0222174, -9.01119219)	Punkt (1.02222; -9.01119)

5.4. Test dla funkcji homograficznej (z istniejącym minimum)

Ten test został przeprowadzony celem sprawdzenia, czy zaimplementowany algorytm poradzi sobie z funkcją homograficzną. Na przedziale zawężonym do takiego, gdzie minimum istnieje funkcja zwraca poprawny i spodziewany wynik (*Tabela 4*).

Tabela 4: Zestawienie wyniku dla testu dla funkcji homograficznej z istniejącym minimum.

Spodziewany wynik	Otrzymany wynik
Punkt (-0.5; -2)	Punkt (-0.5; -2)

5.5. Test dla funkcji homograficznej (bez istniejącego minimum)

Ten test miał na celu sprawdzenie jaki wynik zwróci algorytm w obliczu funkcji posiadającą asymptotę pionową w końcowej wartości przedziału. Zadane warunki testu nie mają więc minimum, ponieważ z racji niemożności dzielenia przez zero, jest to liczba nie należąca do dziedziny tej funkcji. Zaimplementowany algorytm nie obsługuje takich błędów, więc test miał na celu sprawdzenie jak algorytm się zachowa. Wynik otrzymany z metody złotego podziału w postaci punktu (*Tabela 5*) sugeruje punkt blisko zera jako potencjalne minimum, jednakże wartość funkcji w tym punkcie jest bardzo duża.

Tabela 5: Zestawienie wyniku dla testu dla funkcji homograficznej bez istniejącego minimum.

Spodziewany wynik	Otrzymany wynik
Brak minimum	Punkt (-4,10606e-07; -2,43542e+06)

6. Wnioski

Analiza otrzymanych wyników w kontekście spodziewanych rezultatów pozwala na lepsze zrozumienie działania metody optymalizacyjnej. W przypadku niespójności między oczekiwanymi a otrzymanymi wynikami, istotne jest zidentyfikowanie potencjalnych przyczyn takiego zachowania i ewentualne dostosowanie procedury testowej – w obu takich przypadkach wiadomo było skąd biorą się błędy i miało to na celu ukazanie zachowania algorytmu wobec przypadków szczególnych.

Parametry takie jak przedział poszukiwań czy tolerancja mogą mieć istotny wpływ na wyniki optymalizacji. Konieczne jest staranne dostosowanie tych parametrów do charakterystyki badanej funkcji w celu uzyskania wiarygodnych wyników. Testowanie metod optymalizacyjnych pozwala także na zidentyfikowanie ewentualnych ograniczeń tych algorytmów – w przypadku zaimplementowanego algorytmu szczególnie w przypadku funkcji, które nie posiadają minimum.

7. Źródła

Wykłady z Metod Numerycznych autorstwa dr hab. Danuty Szeligi

Prezentacja Optymalizacji – metoda złotego podziału autorstwa dr. hab. Inż. Marcina Hojnego.

Wikipedia – artykuły o optymalizacji, złotym podziale

program WolframAlpha: https://www.wolframalpha.com

program Geogebra: https://www.geogebra.org/graphing