WIMiIP Inżynieria Obliczeniowa Grupa 4	Data wykonania: 14.03.2024 r.	Przedmiot: Metody numeryczne
Temat ćwiczenia: **Interpolacja Lagrange'a**		
Imię i nazwisko: Wojciech Matys	Ocena i uwagi:	

Cel ćwiczenia

Celem laboratorium było zapoznanie się z metodą interpolacji Lagrange'a oraz implentacja jej w wybranym przez siebie języku, ja zdecydowałem się na C++.

Wstęp teoretyczny

Interpolacja w metodach numerycznych odnosi się do procesu przybliżania nieznanej funkcji za pomocą zestawu znanych danych punktowych. W skrócie, interpolacja polega na konstrukcji funkcji, która "przechodzi przez" dane punkty, pozwalając na szacowanie wartości funkcji w punktach pomiędzy tymi znanymi. Jest to jedna z podstawowych technik analizy danych numerycznych i jest szeroko stosowana w różnych dziedzinach nauki, inżynierii i informatyki.

Interpolacja w metodach numerycznych jest kluczowym narzędziem w analizie danych, wizualizacji, modelowaniu matematycznym oraz w praktycznych zastosowaniach, takich jak przetwarzanie obrazów, grafika komputerowa czy symulacje fizyczne. Jednak zawsze warto pamiętać, że dobór odpowiedniej metody interpolacji oraz odpowiednia analiza uzyskanych wyników są kluczowe dla uzyskania wiarygodnych i użytecznych wyników.

Istnieje wiele różnych metod interpolacji, w tym sprawozdaniu przeanalizuję metodę interpolacji Lagrange'a.

Interpolacja Lagrange'a

Interpolacja Lagrange'a to technika interpolacji wykorzystywana do przybliżania funkcji za pomocą wielomianu, który przechodzi przez zestaw danych punktów. Jest to jedna z popularnych metod interpolacji, która opiera się na wykorzystaniu wielomianów Lagrange'a do utworzenia wielomianu interpolacyjnego.

Załóżmy, że mamy zestaw n punktów danych: (x1, y1), (x2, y2),...,(xn, yn). Wielomian interpolacyjny L(x) utworzony za pomocą interpolacji Lagrange'a jest sumą iloczynów Lagrange'owych dla każdego punktu danych.

$$L(x) = \sum_{i=1}^{n} y_i l_i(x)$$

Gdzie:

li – współczynnik,

yi - wartość funkcji odpowiadająca argumentowi xi

Współczynnik li jest zdefiniowany jako iloczyny:

$$l_i(x) = \prod_{0<j \wedge j \neq i} \frac{x - x_j}{x_i - x_i}$$

Interpolacja Lagrange'a jest stosunkowo łatwa do zrozumienia i implementacji, co sprawia, że jest popularną techniką w praktycznych zastosowaniach numerycznych. Jednakże dla dużych zestawów danych może być bardziej efektywna w użyciu czasu obliczeń niż metody oparte na interpolacji funkcjami sklejanymi.

Implementacja

Interpolacje Lagrange'a implementuję w języku programowania c++, korzystając z środowiska Visual Studio.

Pierwszy fragment programu

```cpp
#include <iostream>
#include <fstream>

using namespace std;

int main()
{
    int rozmiar;
    fstream czytaj;
    czytaj.open("MN-1.txt");
    czytaj >> rozmiar;
    cout << rozmiar;

    int** M = new int* [rozmiar];

    for (int i = 0; i < rozmiar; i++)
        M[i] = new int[rozmiar];
    cout << endl;

    for (int i = 0; i < rozmiar; i++)
    {
        for (int j = 0; j < 2; j++)
        {
            czytaj >> M[i][j];
            cout << M[i][j] << " ";
        }
        cout << endl;
    }
```

Ta część kodu odpowiada za odczytanie danych z pliku tekstowego i zapisanie ich w tablicy dwuwymiarowej. Rozmiar tablicy jest odczytywany z pierwszej linii pliku, a pozostałe dane są odczytywane wiersz po wierszu. Odczytane dane są wyświetlane na ekranie.

Drugi fragment programu

```
    float argument;
    cout << "Podaj argument wielomianu dla, ktorego chesz znalezc wartosc: " << endl;
    cin >> argument;

    float wynik = 0;

    for (int i = 0; i < rozmiar; i++)
    {
        float wielomian = 1;
        for (int j = 0; j < rozmiar; j++)
        {
            if (j != i)
            {
                wielomian = wielomian * (argument - M[j][0]) / (M[i][0] - M[j][0]);
            }
        }
        wynik = wynik + M[i][1] * wielomian;
    }
    cout << wynik << endl;

    for (int i = 0; i < rozmiar; i++) {   // Zwolnienie miejsca w pamięci
        delete[] M[i];
    }
    delete[] M;

    return 0;
}
```

Opis działania zaimplementowanego algorytmu

1. Wczytanie argumentu:

- Użytkownik jest proszony o podanie argumentu, dla którego chce obliczyć wartość wielomianu.

2. Inicjalizacja zmiennych:

- Zmienna `wynik` jest inicjalizowana jako 0. Będzie przechowywać obliczoną wartość wielomianu dla danego argumentu.

3. Iteracja po punktach interpolacji:

- Algorytm przechodzi przez wszystkie punkty interpolacji, które zostały wcześniej wczytane z pliku.

- Dla każdego punktu interpolacji, który reprezentowany jest przez `M[i][0]` (pierwsza kolumna macierzy `M`), wykonuje się następujące kroki:

4. Obliczenie wielomianu Lagrange'a:

- Inicjalizuje się zmienną `wielomian` jako 1. Ta zmienna będzie reprezentować obecny wielomian Lagrange'a dla danego punktu interpolacji.

- Następnie wykonuje się pętlę wewnętrzną, która iteruje po wszystkich punktach interpolacji:

  - Warunek `if (j != i)` zapewnia, że aktualny punkt interpolacji nie będzie brany pod uwagę w obliczeniach, aby uniknąć dzielenia przez zero.

  - Obliczana jest różnica argumentu (`argument - M[j][0]`) oraz różnica między argumentem a `x` danego punktu interpolacji (`M[i][0] - M[j][0]`).

  - Wartość obliczona dla każdego punktu interpolacji jest mnożona przez dotychczasowy `wielomian`.

5. Sumowanie wartości wielomianu:

- Obliczona wartość `wynik` dla danego punktu interpolacji jest sumowana z poprzednimi wartościami pomnożonymi przez współczynniki `M[i][1]` (drugą kolumną macierzy `M`).

6. Wyświetlenie wyniku:

- Obliczona wartość wielomianu dla podanego argumentu jest wyświetlana na standardowym wyjściu.

7. Zwolnienie pamięci:

- Na koniec program zwalnia pamięć zajmowaną przez dynamicznie alokowaną macierz `M`.

Algorytm ten wykorzystuje interpolację Lagrange'a do obliczenia wartości wielomianu dla danego argumentu, bazując na punktach interpolacji wczytanych z pliku. Każdy punkt interpolacji przyczynia się do obliczenia wartości wielomianu poprzez obliczenie odpowiadającego mu wielomianu Lagrange'a.

Testy jednostkowe

Test 1 (Przykład z instrukcji, sprawdzamy czy nasz program działa poprawnie)

Chcemy wyznaczyć wartość funkcji w punkcie **x = 4**

Dane punkty (1, 12), (3, 4), (5, 4)

Wzór funkcji f(x) = $x^2 - 8x + 19$

Dane wejściowe programu:

3

1 12

3 4

5 4

Wynik programu :

```
3
1 12
3 4
5 4
Podaj argument wielomianu dla, ktorego chesz znalezc wartosc:
4
Wynik = 3
```

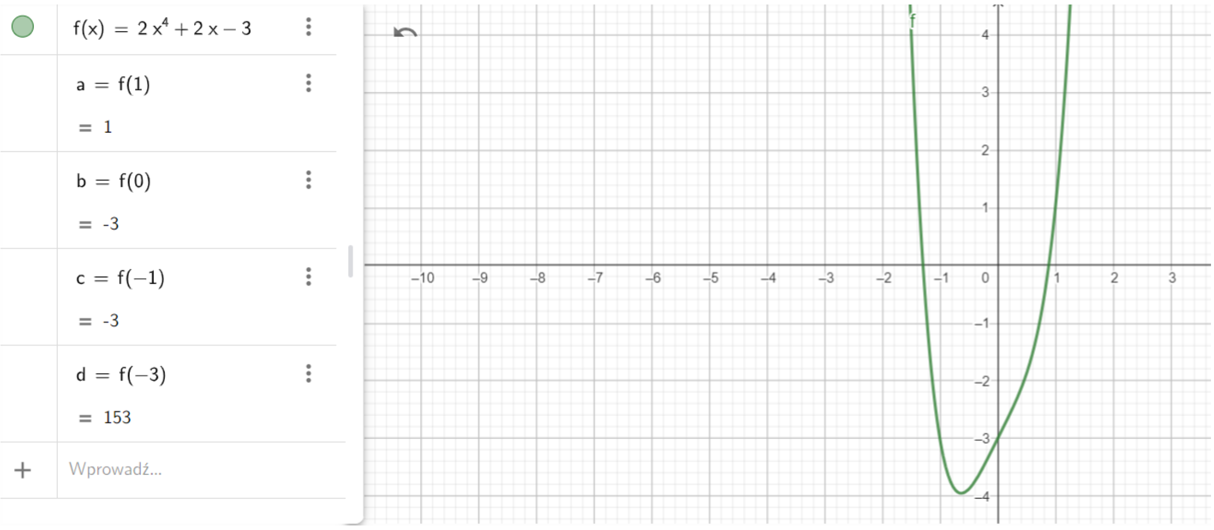
Wynik obliczony ze wzoru = 3

Wniosek

Wynik obliczony ze wzoru = Wynik programu

Test 2

Na załączonym obrazie, mamy wykres badanej funkcji, dane punkty oraz jej wzór



Dane wejściowe programu:

5

12	41493
1	1
0	-3
-1	-3
-3	153

Chcemy wyznaczyć wartość funkcji w punkcie **x = 11**

Wynik programu :

```
5
12 41493
1 1
0 -3
-1 -3
-3 153
Podaj argument wielomianu dla, ktorego chesz znalezc wartosc:
11
Wynik = 29301
```

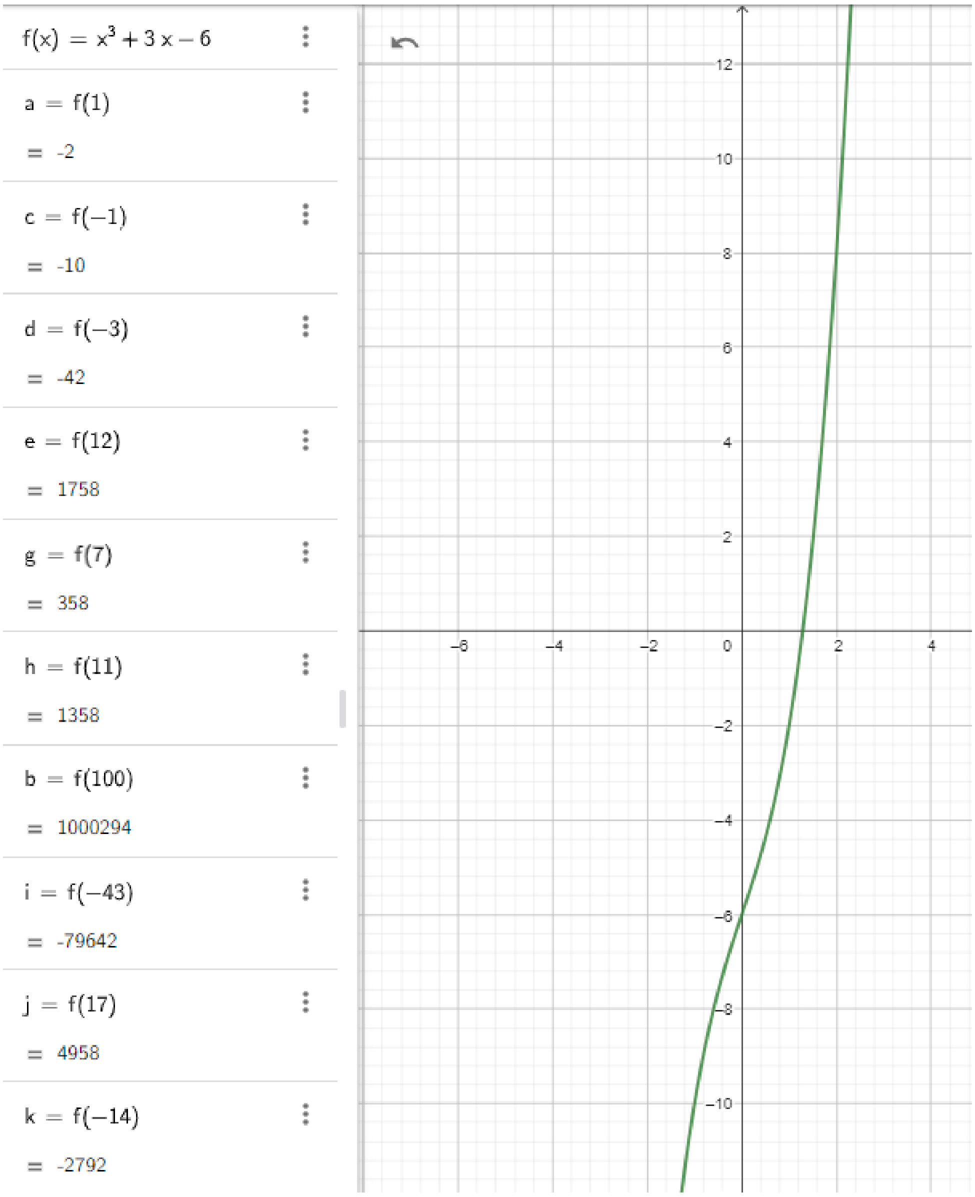
Wynik obliczony ze wzoru = 29301

Wniosek

Wynik obliczony ze wzoru = Wynik programu

Test 3

Na załączonym obrazie, mamy wykres badanej funkcji, dane punkty oraz jej wzór



Dane wejściowe programu:

10

1	-2
-1	-10
-3	-42
12	1758
7	358
11	1358
100	1000294
-43	-79642

17 4958

-14 -2792

Chcemy wyznaczyć wartość funkcji w punkcie **x = 71**

Wynik programu :

```
10
1 -2
-1 -10
-3 -42
12 1758
7 358
11 1358
100 1000294
-43 -79642
17 4958
-14 -2792
Podaj argument wielomianu dla, ktorego chesz znalezc wartosc:
71
Wynik = 359473
```

Wynik obliczony ze wzoru = 358118

Wniosek

Wynik obliczony ze wzoru != Wynik programu w tym przypadku testowym wykryliśmy błąd, spróbujmy dodać kolejne węzły interpolacji bliżej argumentu 71 i zobaczyć czy wynik będzie bardziej dokładny.

Na ten moment skala błędu to około 0,3% więc uważam ten wynik mimo wszystko za dosyć dokładny.

Dodajemy dwa punkty :

m = f(70)

= 343204

n = f(72)

= 373458

Dane wejściowe programu:

12

1 -2

-1 -10

-3 -42

12 1758

7	358
11	1358
100	1000294
-43	-79642
17	4958
-14	-2792
70	343204
72	373458

Ponownie chcemy wyznaczyć wartość funkcji w punkcie **x = 71**

Wynik programu :

```
12
1 -2
-1 -10
-3 -42
12 1758
7 358
11 1358
100 1000294
-43 -79642
17 4958
-14 -2792
70 343204
72 373458
Podaj argument wielomianu dla, ktorego chesz znalezc wartosc:
71
Wynik = 358118
```

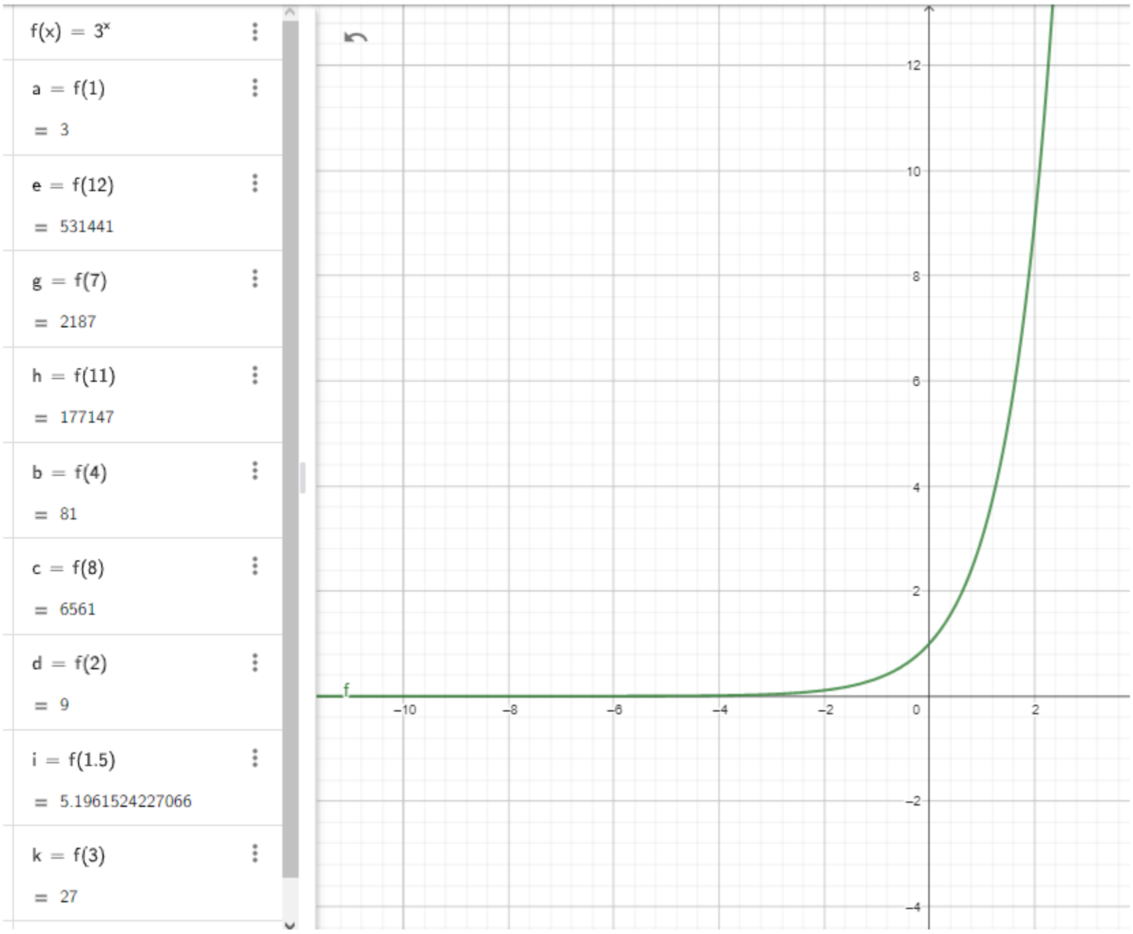
Dzięki dodaniu dwóch dodatkowych punktów program policzył tym razem wartość poprawnie.

Wynik obliczony ze wzoru = Wynik programu

Test 4

Na załączonym obrazie, mamy wykres badanej funkcji, dane punkty oraz jej wzór

$$f(x) = 3^x$$



Dane wejściowe programu:

9

1	3
2	9
3	27
4	81
7	2187
8	6561
11	177147
12	531441
17	129140163

Chcemy wyznaczyć wartość funkcji w punkcie **x = 11.5**

Wynik programu :

```
9
1 3
2 9
3 27
4 81
7 2187
8 6561
11 177147
12 531441
17 129140163
Podaj argument wielomianu dla, ktorego chesz znalezc wartosc:
11.5
Wynik = 289917
```

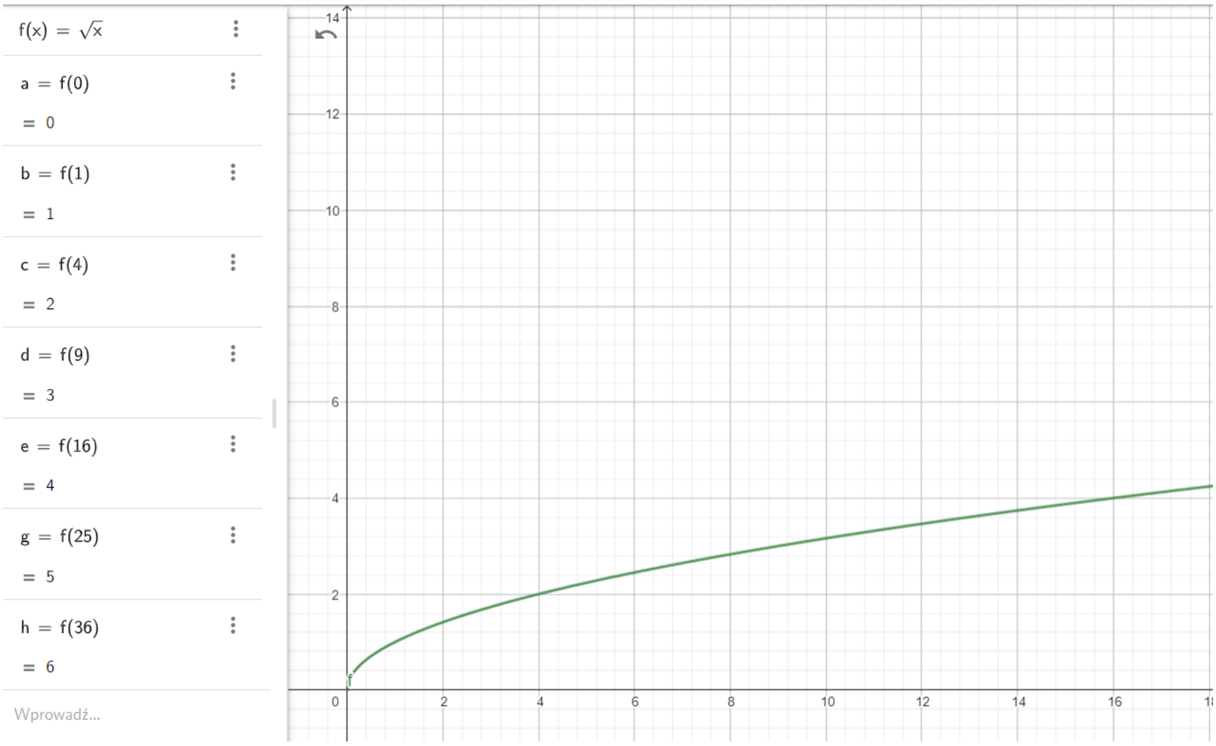
Wynik obliczony ze wzoru: ≈ 306827.6

Wynik obliczony ze wzoru != Wynik programu

Skala błędu około 5,5%

Test 5

Na załączonym obrazie, mamy wykres badanej funkcji, dane punkty oraz jej wzór



Dane wejściowe programu:

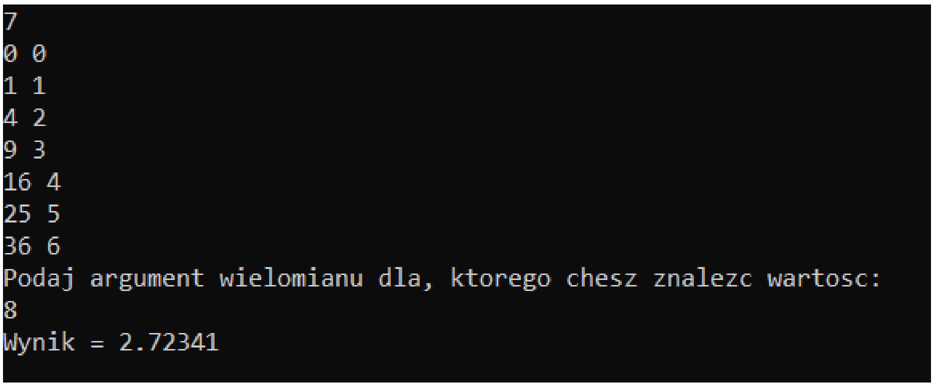
7

0	0
1	1
4	2
9	3

16	4
25	5
36	6

Chcemy wyznaczyć wartość funkcji w punkcie **x = 8**

Wynik programu :



```
7
0 0
1 1
4 2
9 3
16 4
25 5
36 6
Podaj argument wielomianu dla, ktorego chesz znalezc wartosc:
8
Wynik = 2.72341
```

Wynik obliczony ze wzoru ≈ 2.829

Wynik obliczony ze wzoru != Wynik programu

Skala błędu około 2%

Wnioski:

1. Wybór odpowiednich punktów interpolacji ma kluczowe znaczenie dla precyzji interpolacji. Gdy punkty są równomiernie rozłożone w obszarze, gdzie funkcja interpolowana ulega istotnym zmianom, interpolacja będzie bardziej dokładna. W przypadku funkcji, które są trudne do dokładnego obliczenia, jak np. funkcje pierwiastkowe lub potęgowe, istotne jest odpowiednie dobranie punktów, aby uniknąć nietypowych wyników.

2. Dla funkcji, które sprawiają trudności w interpolacji za pomocą metody Lagrange'a, warto rozważyć zastosowanie alternatywnych metod, takich jak metoda Newtona-Raphsona. Te metody są lepiej dostosowane do obliczeń numerycznych, szczególnie jeśli dokładność jest kluczowa.

3. Przed zastosowaniem interpolacji Lagrange'a ważne jest przetestowanie kodu z różnymi zestawami danych, aby ocenić jego skuteczność i dokładność. W przypadku funkcji, które są trudne do interpolacji, może być konieczne eksperymentalne dostosowanie punktów interpolacji w celu uzyskania lepszych wyników.

Podsumowując, wybór odpowiednich punktów interpolacji, rozważenie alternatywnych metod oraz eksperymentalne testowanie kodu są kluczowymi krokami w zapewnieniu skutecznej i dokładnej interpolacji funkcji.

Źrodła

-wykresy funkcji pobrane ze strony https://www.geogebra.org/?lang=pl