Paulina Grabowska [redacted]	Metody Numeryczne
Interpolacja Lagrange'a	

1. Wstęp

Interpolacja jest matematyczną techniką umożliwiającą estymację wartości funkcji w punktach, dla których nie są dostępne bezpośrednie pomiary. W procesie interpolacji tworzy się funkcję lub krzywą, która przechodzi przez zadane punkty danych, co umożliwia szacowanie wartości dla argumentów znajdujących się między tymi punktami.

Istnieje wiele różnych metod interpolacji: wielomianami, funkcjami wymiernymi, funkcjami trygonometrycznymi czy chociażby funkcjami sklejanymi. Interpolację stosuje się: w metodach numerycznych oraz w budowaniu funkcji na podstawie danych pomiarowych ograniczonej liczby punktów.

2. Interpolacja Lagrange'a

Jednym z popularnych algorytmów interpolacyjnych jest metoda Lagrange'a, nazwana na cześć francuskiego matematyka Josepha-Louisa Lagrange'a.

Interpolacja wielomianowa polega na przybliżaniu funkcji za pomocą wielomianów, co pozwala na przybliżenie funkcji, której wartości znane są tylko w wybranych punktach. Zakłada się, że znane są wartości funkcji w pewnych równo rozmieszczonych punktach. Wielomian Lagrange'a stopnia n, to taki wielomian, który w $n+1$ punktach przyjmuje wartości argumentów i odpowiadające im wartości funkcji.

Tworzony jest wielomian interpolacyjny, który jest kombinacją liniową wielomianów bazowych. Każdy z tych wielomianów bazowych jest związany z jednym z punktów danych i jest konstruowany w taki sposób, że przyjmuje wartość 1 w swoim punkcie bazowym i 0 we wszystkich pozostałych punktach danych.

Zadanie interpolacyjne Lagrange'a ma jednoznaczne rozwiązanie, które można przedstawić w postaci:

$$L_n(x) = \sum_{i=0}^{n} f(x_i) l_i(x) \tag{1}$$

$$l_i(x) = \prod_{j=0\ \&\ j \neq i}^{n} \frac{x - x_j}{x_i - x_j} \tag{2}$$

Nie jest to jednak metoda idealna - interpolacja Lagrange'a może prowadzić do efektu Rungego (od nazwiska Carla Davida Tolmé Rungego). Efekt ten polega na oscylacji wartości przy zbliżaniu się do krańców przedziału interpolacji, czyli coraz gorszej jakości interpolacji wraz ze wzrostem liczby n.

Interpolacja Lagrange'a znajduje zastosowanie w różnych dziedzinach, takich jak analiza danych, grafika komputerowa, nauki przyrodnicze i inżynieria. Może być używana do przybliżania funkcji na podstawie eksperymentalnych danych pomiarowych.

3. Implementacja numeryczna

Zaimplementowano algorytm interpolacji Lagrange'a w środowisku języka C++. Funkcja *lagrange_interpolation* implementuje interpolację dla zestawu danych punktów (*x[i], y[i]*). Funkcja zaczyna od utworzenia zmiennej *L*, symbolizującą wartość interpolowanej funkcji i ustawia ją na 0.0. Następnie rozpoczyna się pętla iterująca po wszystkich punktach danych (*x[i], y[i]*). Dla każdego takiego punktu, tworzona jest zmienna *l*, której wartość ustawiona jest na *y[i]* i symbolizuje iloczyn wartości funkcji w danym punkcie przez współczynniki wielomianu Lagrange'a.

Wówczas rozpoczyna się druga pętla, wewnętrzna, iterująca po wszystkich punktach danych. Ta pętla służy do obliczenia współczynnika Lagrange'a dla danego punktu. Aktualizowana jest wartość *l* poprzez pomnożenie ją przez współczynnik Lagrange'a dla danego punktu (*x[i], y[i]*).

Po zakończeniu wewnętrznej pętli następuje dodanie wartości zmiennej *l* do zmiennej *L*. Ta operacja pozwala na zsumowanie wszystkich składników interpolacji Lagrange'a dla poszczególnych punktów. Następnie kończy się zewnętrzna pętla i funkcja *lagrange_interpolation* zwraca obliczoną wartość interpolacji Lagrange'a dla punktu *input_data*.

```cpp
double lagrange_interpolation(double x[], double y[], int n, double input_data) {

    double L = 0.0;

    for (int i = 0; i < n; ++i) {
        double l = y[i];
        for (int j = 0; j < n; ++j) {
            if (j != i) {
                l *= (input_data - x[j]) / (x[i] - x[j]);
            }
        }
        L += l;
    }

    return L;
}
```

Rys. 1 Fragment kodu zawierający funkcję *lagrange_interpolation*.

Dane, na których pracuje funkcja umieszczono w pliku tekstowym, a w funkcji *main* dołączono funkcjonalność czytania z pliku. Następuje to przy pomocy inicjalizacji obiektu *ifstream*, po czym sprawdzany jest warunek otwarcia pliku, celem uniknięcia ewentualnego błędu.

Struktura pliku tekstowego wygląda następująco: pierwsza linijka zawiera jedynie liczbę punktów *n*. Każda następna linijka zawiera już dwie liczby: pierwsza z nich to zmienna *x* danego punktu, a druga jest zmienną *y*. W taki też sposób następuje odczyt z pliku - najpierw sczytywana jest liczba n, a na jej podstawie tworzone są tablice x[n] i y[n], które są później uzupełniane danymi z pliku tekstowego poprzez zastosowanie pętli.

Po zakończeniu odczytywania następuje zamknięcie pliku tekstowego, celem zwolnienia zasobów systemowych i uniknięcia późniejszych ewentualnych problemów z dostępem do pliku, gdyż z dostępu do niego mogą później korzystać różne aplikacje.

```
main.cpp | data.txt
3
1 12
3 4
5 4

```

Rys. 2 Plik tekstowy zawierający zadane dane.

Na koniec jeszcze wprowadzana z poziomu konsoli jest jedna dana wejściowa - *input_data* - która określa, dla jakiego argumentu funkcji chcemy obliczyć wartość funkcji za pomocą interpolacji Lagrange'a. Dla wprowadzonej liczby wywołujemy funkcję *lagrange_interpolation* i wynik interpolacji wyświetlamy na ekranie konsoli.

```
int main() {

    ifstream file("data.txt");

    if (!file.is_open()) {
        cout << "no such file exits" << endl;
        return 1;
    }


    int n;
    file >> n;

    double x[n];
    double y[n];

    for (int i = 0; i < n; ++i) {
        file >> x[i] >> y[i];
    }

    double input_data;
    cout << "input data x for intepolation: ";
    cin >> input_data;

    double interpolatedValue = lagrange_interpolation(x, y, n, input_data);

    cout << "interpolated data in point " << input_data << ": " << interpolatedValue << endl;

    file.close();

    return 0;
}
```

Rys. 3 Fragment kodu zawierający funkcję *main* oraz funkcjonalność czytania z pliku.

Po zaimplementowaniu algorytmu sprawdzono poprawność jego działania dla zadanej wcześniej wartości (równej 4). Wynik zgadzał się, co pozwala na stwierdzenie, że algorytm działa poprawnie. Następnie przystąpiono do napisania testów jednostkowych, które mają upewnić się, co do poprawnego działania algorytmu.

```
input data x for intepolation: 4
interpolated data in point 4: 3
```

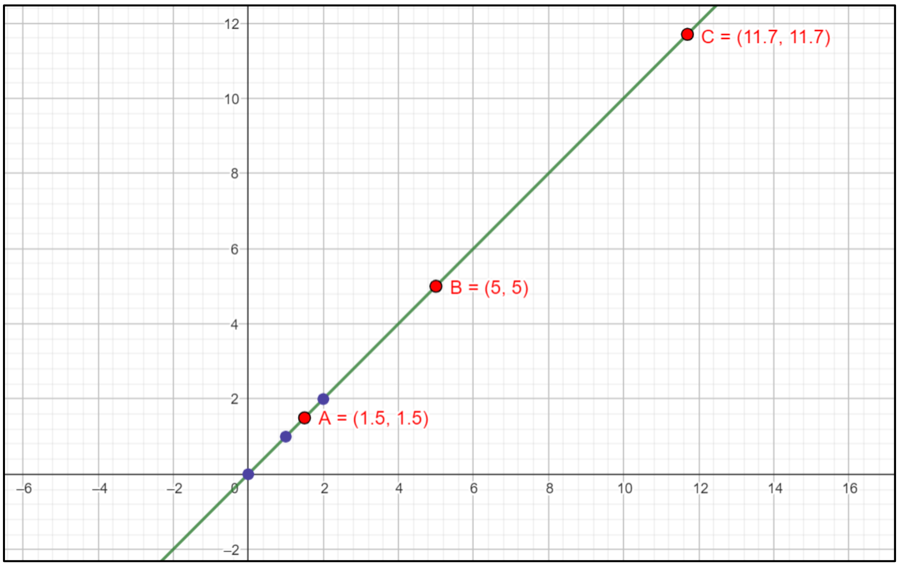
Rys. 4 Zrzut ekranu konsoli programu.

4. Wykonanie testów jednostkowych

4.1 Test jednostkowy dla danych liniowych.

Plik zmodyfikowano - wartości *x* i *y* zostały ustawione na identyczne względem siebie, równe odpowiednio 0, 1 i 2; tworząc tym samym funkcję *f(x)* = *x*. Liczbę punktów pozostawiono równą trzy.

Funkcję tę (oznaczona na zielono) zwizualizowano w programie Geogebra i uzyskano wykres przedstawiony na *Rys. 5a* (punkty wprowadzone oznaczone są na niebiesko). Na wykresie zaznaczono punkty *input_data*, dla których liczono interpolację. Wybrano wartości 1,5; 5 oraz 11,7 (punktu oznaczone na czerwono z przypisanymi wartościami), których wynik porównano z danymi otrzymanymi w PlanetCalc (*Rys. 5b*).



Rys. 5a Wykres funkcji utworzonej dla testu jednostkowego dla danych liniowych z zaznaczonymi punktami poddawanymi interpolacji.

Lagrange Polynomial
$L(x) = x$

Interpolated Points

x	1.5	5	11.7
y	1.5	5	11.7

Rys. 5b Wartości funkcji w określonych punktach policzone w programie PlanetCalc.

```
input data x for intepolation: 1.5
interpolated data in point 1.5: 1.5
```

Rys. 5c Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych liniowych dla punktu 1,5.

```
input data x for intepolation: 5
interpolated data in point 5: 5
```

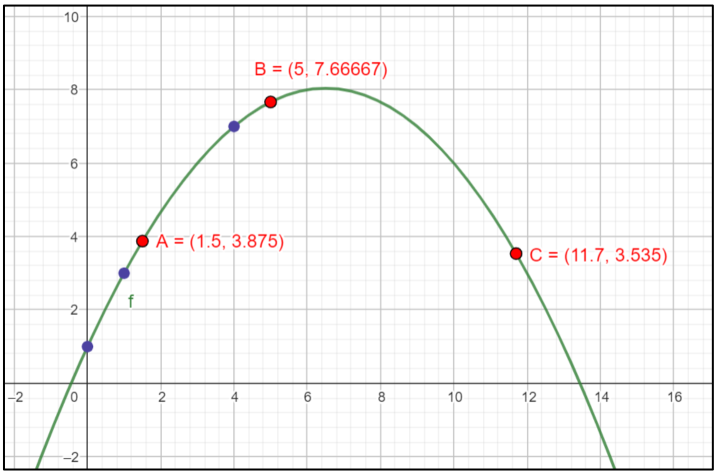
Rys. 5d Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych liniowych dla punktu 5.

```
input data x for intepolation: 11.7
interpolated data in point 11.7: 11.7
```

Rys. 5e Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych liniowych dla punktu 11,7.

4.2 Test jednostkowy dla danych nieliniowych

Zmodyfikowano plik tekstowy *data.txt* - liczba punktów *n* pozostała niezmienna i wynosi 3, natomiast zmieniono wartości *x* oraz *y*. Wprowadzono punkty (0, 1), (1, 3), (4, 7). Wykres z programu Geogebra obrazuje *Rys. 6a*. Wartości punktów, dla których liczona będzie interpolacja ustawiono na 1.5, 5 oraz 11,7 (zaznaczone na *Rys. 6a* na czerwono). Obliczenia sprawdzono w programie PlanetCalc.



Rys. 6a Wykres funkcji utworzonej dla testu jednostkowego dla danych nieliniowych z zaznaczonymi punktami poddawanymi interpolacji (na niebiesko)/

Lagrange Polynomial

$$L(x) = -\frac{1}{6}x^2 + \frac{13}{6}x + 1$$

Interpolated Points

x	1.5	5	11.7
y	3.875	7.66667	3.535

Rys. 6b Wartości funkcji w określonych punktach policzone w programie PlanetCalc.

```
input data x for intepolation: 1.5
interpolated data in point 1.5: 3.875
```

Rys. 6c Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych nieliniowych dla punktu 1,5.

```
input data x for intepolation: 5
interpolated data in point 5: 7.66667
```

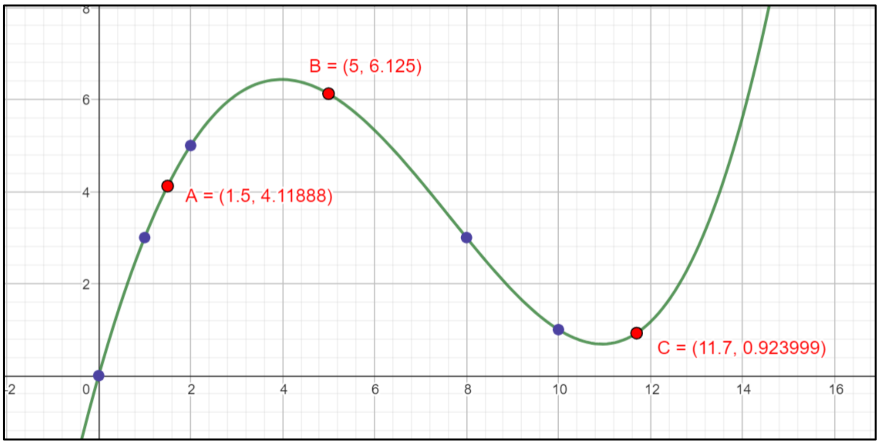
Rys. 6d Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych nieliniowych dla punktu 5.

```
input data x for intepolation: 11.7
interpolated data in point 11.7: 3.535
```

Rys. 6e Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych nieliniowych dla punktu 11,7.

4.3 Test jednostkowy dla różnych liczb punktów danych

Ponownie zmodyfikowano, tym razem zmieniając liczbę punktów *n* na 5. Współrzędne punktów ustawiono na: (0, 0), (1, 3), (2, 5), (8, 3) oraz (10, 1). Punkty wprowadzono do programu Geogebra i na ich podstawie utworzono wykres, na którym zaznaczono na czerwono punkty poddawane interpolacji (o wartościach odpowiednio o 1,5; 5; 11,7). Poprawność liczenia sprawdzono w programie PlanetClac.



Rys. 7a Wykres funkcji utworzonej dla testu jednostkowego dla różnych liczb punktów danych z zaznaczonymi punktami poddawanymi interpolacji (na niebiesko).

Lagrange Polynomial

$$L(x) = \frac{1}{1440}x^4 + \frac{19}{1440}x^3 - \frac{49}{90}x^2 + \frac{1271}{360}x$$

Interpolated Points

x	1.5	5	11.7
y	4.118880	6.125000	0.923999

Rys. 7b Wartości funkcji w określonych punktach policzone w programie PlanetCalc.

```
input data x for intepolation: 1.5
interpolated data in point 1.5: 4.11888
```

Rys. 7c Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla różnych liczb punktów danych dla punktu 1,5.

```
input data x for intepolation: 5
interpolated data in point 5: 6.125
```

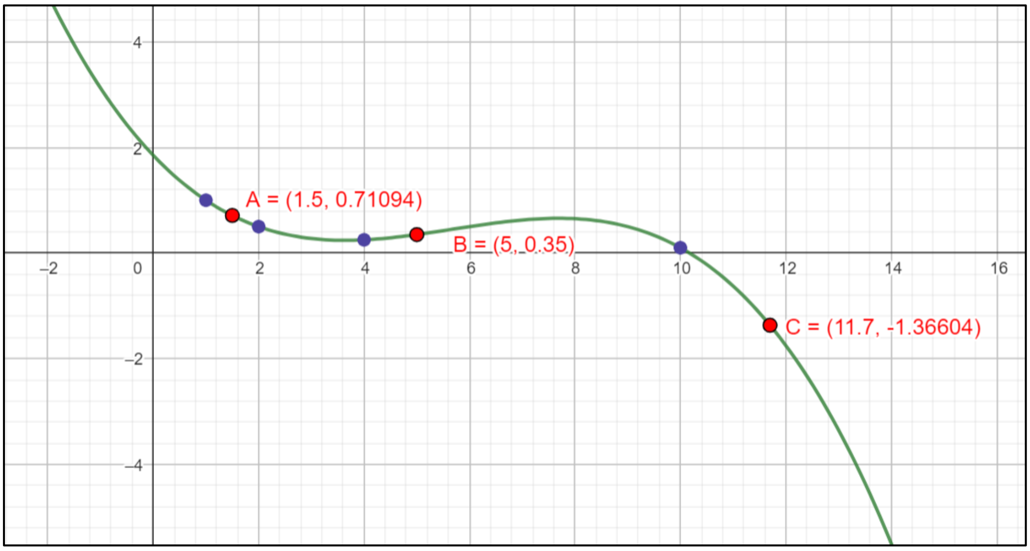
Rys. 7d Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla różnych liczb punktów danych dla punktu 5.

```
input data x for intepolation: 11.7
interpolated data in point 11.7: 0.923999
```

Rys. 7e Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla różnych liczb punktów danych dla punktu 11,7.

4.4 Test jednostkowy dla danych odwrotnie proporcjonalnie

Dla tego testu ustawiono cztery punkty testowe o parach współrzędnych odpowiednio (1, 1), (2, 0.5) oraz (4, 0.25), (10, 0.1); w programie Geogebra utworzono wykres funkcji od razu zaznaczając na nim punktu poddawane interpolacji (1,5; 5; 11,7) i na wykresie zostały zaznaczone na czerwono. Obliczenia zostały sprawdzone w programie PlanetCalc.



Rys. 8a Wykres funkcji utworzonej dla testu jednostkowego dla różnych liczb punktów danych z zaznaczonymi punktami poddawanymi interpolacji

Lagrange Polynomial

$$L(x) = -\frac{1}{80}x^3 + \frac{17}{80}x^2 - \frac{21}{20}x + \frac{37}{20}$$

Interpolated Points

x	1.5	5	11.7
y	0.710937	0.350000	-1.366038

Rys. 8b Wartości funkcji w określonych punktach policzone w programie PlanetCalc.

```
input data x for intepolation: 1.5
interpolated data in point 1.5: 0.710938
```

Rys. 8c Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych odwrotnych proporcjalnie dla punktu 1,5.

```
input data x for intepolation: 5
interpolated data in point 5: 0.35
```

Rys. 8d Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych odwrotnych proporcjalnie dla punktu 5.

```
input data x for intepolation: 11.7
interpolated data in point 11.7: -1.36604
```

Rys. 8e Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla danych odwrotnych proporcjalnie dla punktu 11,7.

4.5 Test jednostkowy dla dużych zestawów danych

Dla tego testu zrezygnowano z funkcjonalności czytania z pliku. Utworzono pętlę, która przydziela wartości tablicowej *x[n]* i *y[n]*. Liczbę punktów zwiększono do tysiąca. W przypadku tablicy *x[n]* przydzielane wartości są liczbami całkowitymi równymi liczbie iteracji pętli, natomiast dla tablicy *y[n]* wartości te są obliczane na podstawie iloczynu tejże liczby iteracji. W domyśle utworzono tym samym funkcję kwadratową rozpoczynającą się w punkcie (0,0) o tysiącu elementach. Obrano cztery punkty do zinterpolowania: 398.2, 501.3, 645, 875.

Lagrange Polynomial
$L(x) = x^2$

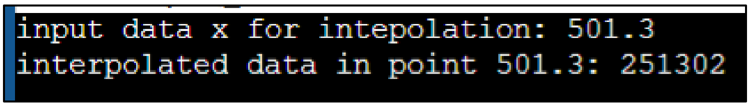
Interpolated Points

x	398.2	501.3	645	875
y	158563.24	251301.69	416025	765625

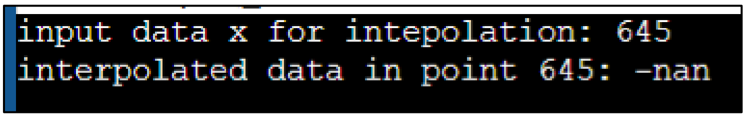
Rys. 9a Wartości funkcji w określonych punktach policzone w programie PlanetCalc.

```
input data x for intepolation: 398.2
interpolated data in point 398.2: 158563
```

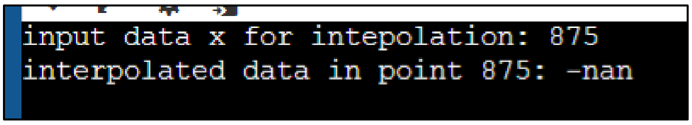
Rys. 9b Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla dużych zestawów danych dla punktu 398.2.



Rys. 9c Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla dużych zestawów danych dla punktu 501.3



Rys. 9d Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla dużych zestawów danych dla punktu 645.



Rys. 9e Wydruk z konsoli po skompilowaniu programu dla testu jednostkowego dla dużych zestawów danych dla punktu 875.

5. Opracowanie wyników

Testy jednostkowe wykonane dla mniejszej ilości danych i rozmaitych funkcji zwracają wartości poprawne i o dobrym rozszerzeniu po przecinku; co potwierdzają wykonanie w PlanetCalc obliczenia, od których otrzymane wyniki nie odbiegają. Dla danych odwrotnie proporcjonalnych można zauważyć lekkie odchylenia w przybliżeniu – nie są to odchylenia wyraźne, ale warto odnotować ich obecność.

Dopiero przy dużych zestawach danych widać niedoskonałości interpolacji Lagrange'a. W przypadku pierwszej i drugiej próby (liczb 398,2 i 501.3) można zaobserwować zaokrąglenie, którego nie powinno być w przypadku zmiennej typu *double*, która może przechowywać aż do piętnastu znaków – zaokrąglenie jest poprawne (w górę lub w dół), ale nie powinno mieć miejsca. Natomiast w przypadku jeszcze większych liczb (645 i 875) algorytm nie jest w stanie już dokonać obliczeń i wyrzuca błąd -*nan* (*not a numer*), co potwierdza występowanie efektu Rungego dla interpolacji Lagrange'a przy dużej ilości danych punktów.

6. Wnioski

Interpolacja Lagrange'a jest metodą dość uniwersalną - stanowi bardzo ważne narzędzie podczas przybliżania funkcji, przede wszystkim ze względu na łatwość implementacji. Jest to również metoda całkiem dokładna i skuteczna, szczególnie gdy punkty są rozłożone równomiernie; należy jednak pamiętać o efekcie Rungego, który może zaburzyć jakość interpolacji podczas nierównomiernego rozłożenia punktów.

Ogólna złożoność obliczeniowa tego algorytmu wynosi $O(n^2)$. Dla prostszych problemów więc jest to metoda całkiem wydajna. Interpolacja Lagrange'a znajduje dzięki temu zastosowanie z wielu dziedzinach i ponadto stanowi cenny element edukacji matematycznej. Wartościowe zastosowanie interpolacji Lagrange'a wymaga jednak świadomości jej ograniczeń i rozważenia alternatywnych metod w zależności od konkretnego kontekstu problemu.

7. Źródła

Wykłady u dr hab. Danuty Szeligi

Prezentacja dra hab. inż. Marcina Hojnego

Wikipedia

https://fizyka.p.lodz.pl/en/download/resource/12711/

Program Geogebra: https://www.geogebra.org/calculator

Program PlanetCalc: https://planetcalc.com