Metody Numeryczne

Projekt 3 – Metody aproksymacji interpolacyjnej

Damian Strojek, s184407

Wstęp

Aproksymacja jest działaniem mającym na celu wyznaczenie przybliżenia nieznanej wartości na podstawie innych, znanych wartości, które są jej wystarczająco bliskie. Często wykorzystuje się ją do wyznaczenia nie jednej wartości, ale postaci całej funkcji ciągłej, na podstawie zbioru znanych wartości, jakie funkcja ta przyjmuje w pewnych punktach swojej dziedziny.

Szczególnym rodzajem aproksymacji jest interpolacja, która wymaga, aby wyznaczona, przybliżona funkcja przyjmowała w punktach pomiarowych, nazywanych węzłami interpolacji, dokładnie takie wartości, jak te będące wynikiem pomiarów w tychże punktach. Do popularnych, prostych metod interpolacji należą interpolacja Lagrange'a oraz interpolacja funkcjami sklejanymi, których efektywność porównano poprzez zastosowanie każdej z nich do aproksymowania profili wysokościowych czterech tras. Trasy, które wybrałem pochodzą z dwóch polskich miast, jednego miasta włoskiego, oraz jedna trasa to trasa stworzona odręcznie (pobrana wcześniej z platformy enauczanie). Liczbę próbek – danych pomiarowych – na każdej z nich ustalono na 500, spośród których wybierane były równoodległe węzły interpolacyjne.

```
// Reference samples - each .txt has more than 500 lines
#define SAMPLES 500

// Samples are defined as distance of the route from point 0
// and elevation above sea level
struct sample{
    double x;
    double y;
};
```

Zrzut ekranu nr 1 – Fragment kodu definujący próbki

Dla obu badanych technik interpolacji wykonano serię 28 testów: po 7 testów na jedną trasę. Każdy z 7 testów różnił się liczbą węzłów interpolacji, wynoszącą kolejno: 4, 5, 6, 8, 12, 21 i 31. Zdecydowałem się na taką ilość testów, aby potem móc jak najlepiej przedstawić różnicę wynikającą z ilości węzłów.

Interpolacja Lagrange'a

Ta metoda interpolacji, zwana także *interpolacją wielomianową*, opiera się na twierdzeniu Stone'a-Weierstrassa, które mówi, że każdą funkcję ciągłą przyjmującą na przedziale [a, b] wartości rzeczywiste można przybliżyć z dowolną dokładnością za pomocą funkcji wielomianowej odpowiednio wysokiego stopnia. Implementacja tej techniki jest bardzo łatwa, gdyż wymaga jedynie wyznaczenia postaci wielomianu Lagrange'a przy wybranej liczbie węzłów interpolacji, a następnie obliczenia wartości tej funkcji w dowolnie wybranym punkcie

przedziału. Z łatwością implementacji wiąże się również krótki czas wykonania algorytmu – jest on zależny od tego ile próbek weźmiemy oraz ile węzłów wybierzemy i jest zdecydowanie krótszy od interpolacji funkcjami sklejanymi. Czasy wykonania dla tej metody jak i część kodu odpowiedzialna za tą metodę widać na poniższych zrzutach ekranu.

```
Average duration(Lagrange method) :
        31 nodes: 41.818 ms
        12 nodes: 30.306 ms
        8 nodes: 27.9091 ms
        6 nodes: 26.4575 ms
```

Zrzut ekranu nr 2 – Czas wykonania interpolacji wielomianowej

```cpp
// For each point we calculate polynomial
double polynomialLagrange(const sample* samples, const double distance, const int n){
    double temp = 0;

    for (int i = 0; i < n; ++i){
        double a = 1;

        for (int j = 0; j < n; ++j)
            if (i != j) a *= (distance - samples[j].x) / (samples[i].x - samples[j].x);

        temp += a * samples[i].y;
    }

    return temp;
};
```

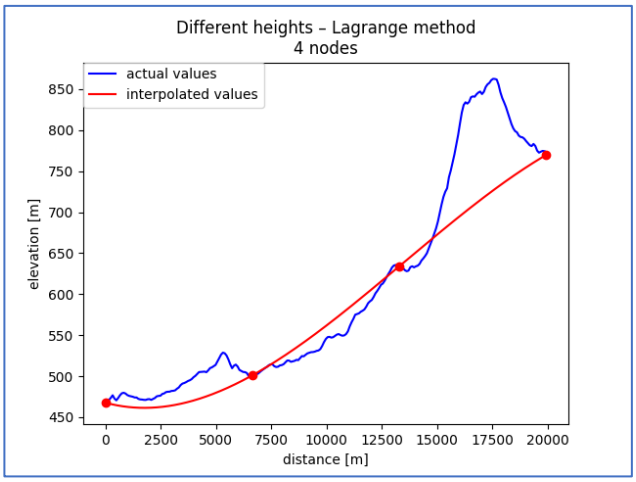
Zrzut ekranu nr 3 – Kod odpowiedzialny za obliczenie funkcji wielomianowej

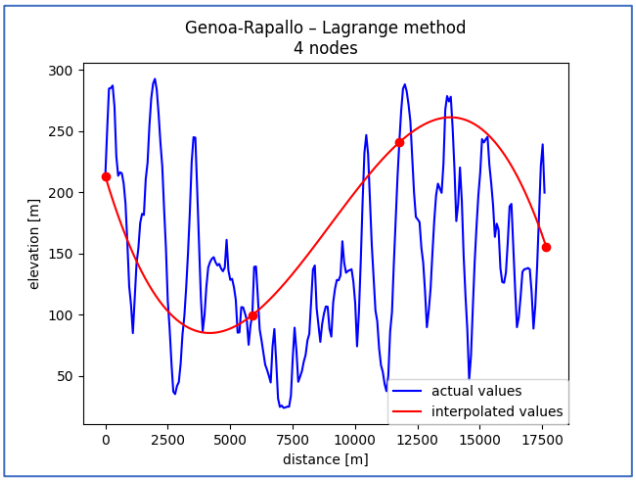
Rezultaty otrzymane po zastosowaniu metody interpolacji wielomianowej dla wybranych danych przedstawione zostały na poniższych wykresach (niebieskie linie przedstawiają rzeczywiste profile wysokościowe wzięte z plików źródłowych, zaś czerwone – aproksymowane; czerwonymi punktami zaznaczono węzły interpolacji).

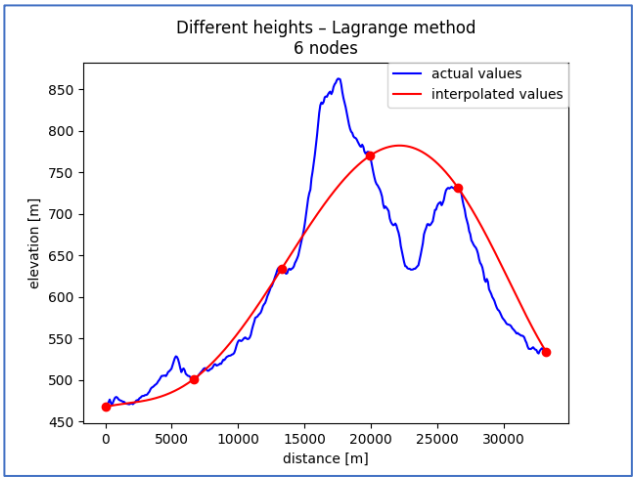
Warto jest wspomnieć o tym, że algorytmy wykorzystane do interpolacji zostały napisane w języku programowania cpp, a wizualizacja rezultatów jest wykonana w języku python.

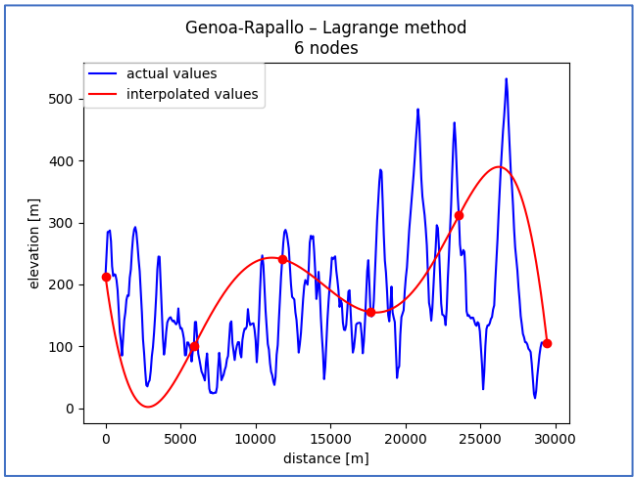
```python
# results of interpolation
x = []
y = []
with open('C:\\Users\\Administrator\\source\\repos\\profilWysokosciowy'
          '\\results\\{}\\{}_{}.txt'.format(method, filename, nodes_number), 'r') as file:
    reader = csv.reader(file, delimiter=' ')
    for row in reader:
        x.append(float(row[0]))
        y.append(float(row[1]))

# reference values
x_ref = []
y_ref = []
with open('C:\\Users\\Administrator\\source\\repos\\profilWysokosciowy'
          '\\data\\{}.txt'.format(filename), 'r') as file:
    reader = csv.reader(file, delimiter=' ')
    for row in reader:
        if float(row[0]) <= x[len(x) - 1]:
            x_ref.append(float(row[0]))
            y_ref.append(float(row[1]))
```

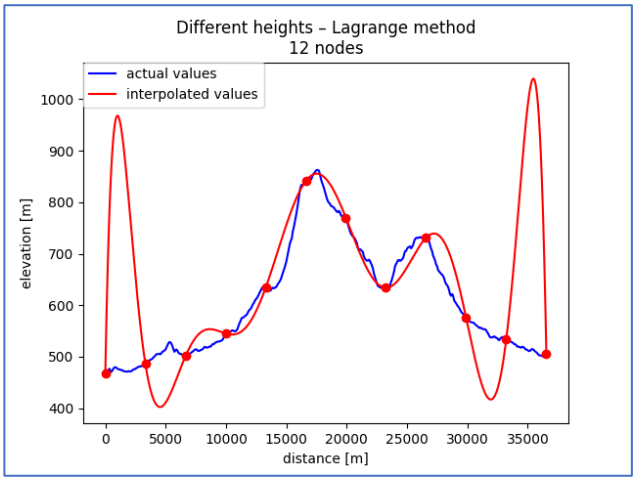
Zrzut ekranu nr 4 – Główna część algorytmu odpowiedzialnego za wizualizacje wyników

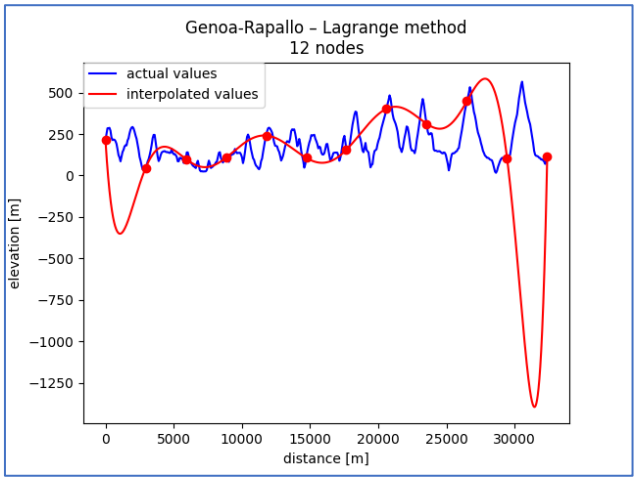

Different heights – Lagrange method
4 nodes
actual values
interpolated values
elevation [m]
distance [m]

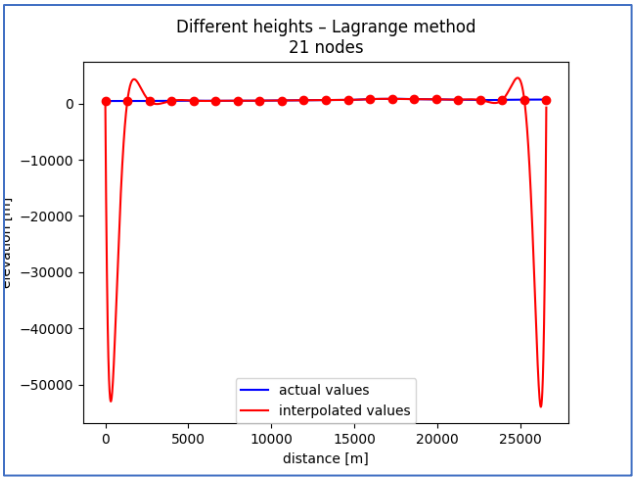

Genoa-Rapallo – Lagrange method
4 nodes
actual values
interpolated values
elevation [m]
distance [m]

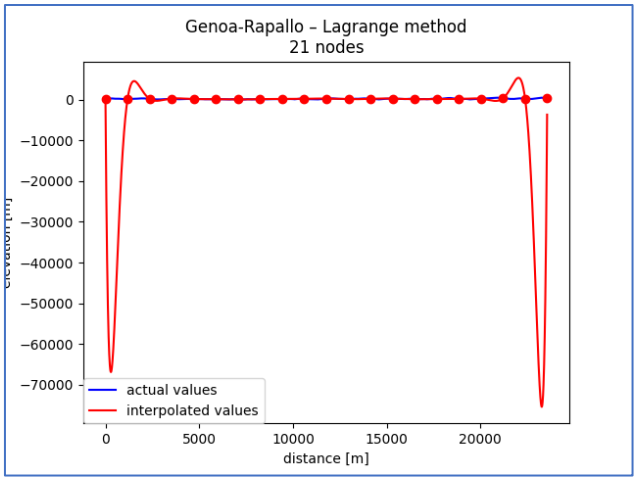

Different heights – Lagrange method
6 nodes
actual values
interpolated values
elevation [m]
distance [m]

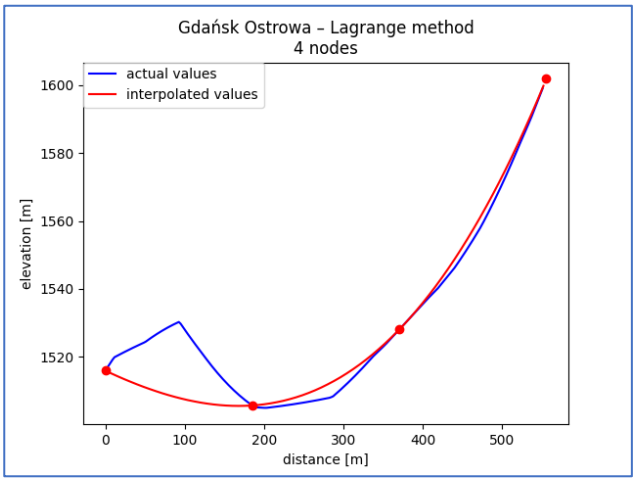

Genoa-Rapallo – Lagrange method
6 nodes
actual values
interpolated values
elevation [m]
distance [m]

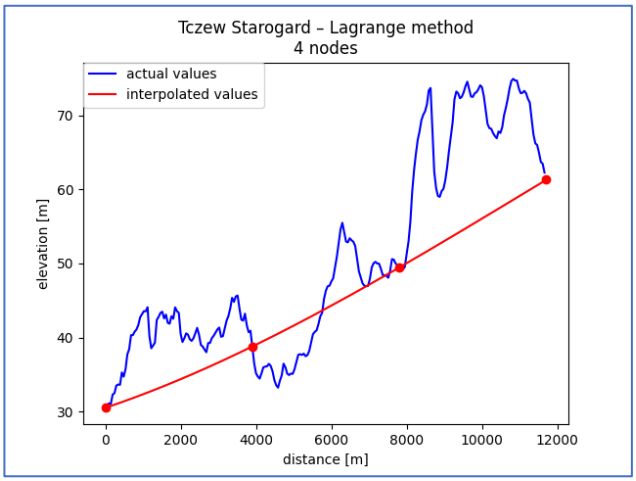

Different heights – Lagrange method
12 nodes
actual values
interpolated values
elevation [m]
distance [m]

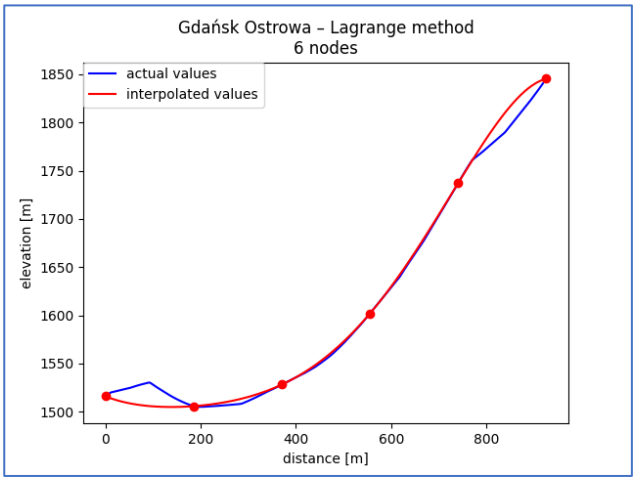

Genoa-Rapallo – Lagrange method
12 nodes
actual values
interpolated values
elevation [m]
distance [m]

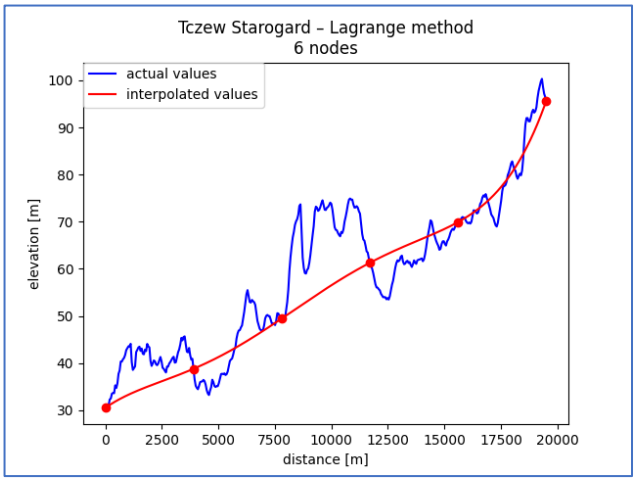

Different heights – Lagrange method
21 nodes
actual values
interpolated values
distance [m]

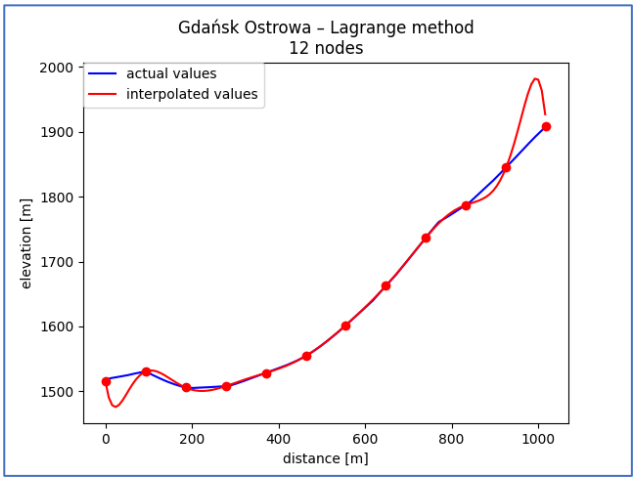

Genoa-Rapallo – Lagrange method
21 nodes
actual values
interpolated values
distance [m]

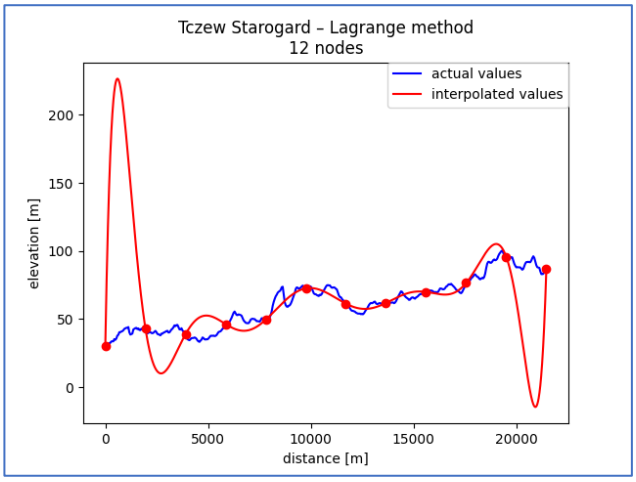

Gdańsk Ostrowa – Lagrange method
4 nodes
actual values
interpolated values
elevation [m]
distance [m]

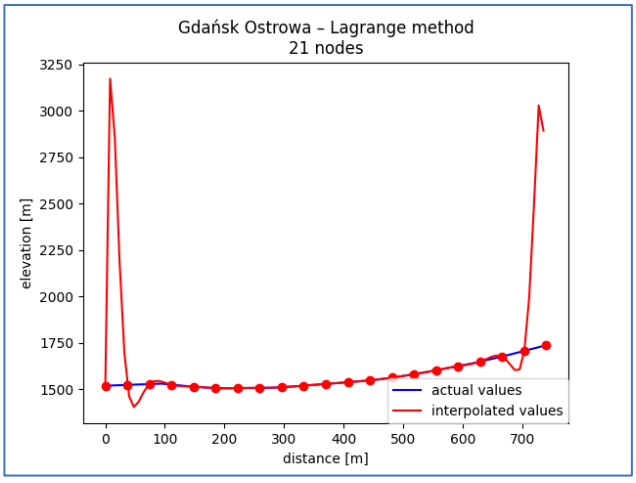

Tczew Starogard – Lagrange method
4 nodes
actual values
interpolated values
elevation [m]
distance [m]

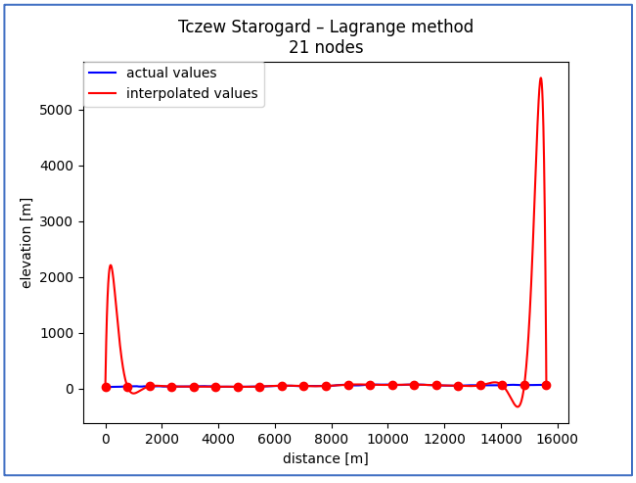

Gdańsk Ostrowa – Lagrange method
6 nodes
actual values
interpolated values
elevation [m]
distance [m]


Tczew Starogard – Lagrange method
6 nodes
actual values
interpolated values
elevation [m]
distance [m]


Gdańsk Ostrowa – Lagrange method
12 nodes
actual values
interpolated values
elevation [m]
distance [m]


Tczew Starogard – Lagrange method
12 nodes
actual values
interpolated values
elevation [m]
distance [m]


Gdańsk Ostrowa – Lagrange method
21 nodes
actual values
interpolated values
elevation [m]
distance [m]

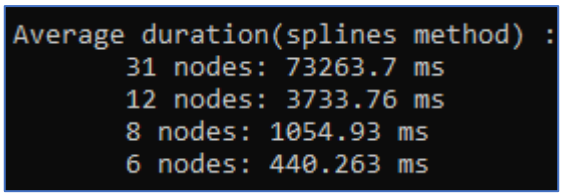

Tczew Starogard – Lagrange method
21 nodes
actual values
interpolated values
elevation [m]
distance [m]

Pierwszą rzeczą, która zwraca uwagę, są znaczne oscylacje pojawiające się na krańcach interpolowanego przedziału. Zwiększają się one wraz ze wzrostem liczby węzłów interpolacji i powodują, że charakter aproksymowanej trasy zanika przy próbie analizy wykresów w całości. Obserwowane odchylenia to tzw. *efekt Rungego*, czyli pogarszanie się wyników interpolacji na krańcach przedziału w miarę zwiększania rzędu aproksymacji wielomianowej (zwiększania stopnia wielomianu aproksymującego). W obszarach tych wartości wyznaczonej funkcji coraz bardziej odbiegają od trendu, jaki wyznaczają zebrane dane. Jest to zjawisko charakterystyczne dla interpolacji wykorzystującej wielomiany wysokich stopni.

Pomijając efekt Rungego i analizując otrzymane wykresy po odrzuceniu ich krańcowych regionów można stwierdzić, że interpolacja Lagrange'a skutecznie przybliża jedynie funkcje monotoniczne w całej dziedzinie lub monotoniczne przedziałami, jeśli przedziały monotoniczności są stosunkowo duże (brak naprzemiennie występujących wzniesień i obniżeń terenu – Gdańsk Ostrowa), ponieważ wystarcza do tego stosunkowo niewielka liczba węzłów interpolacji, a zatem wielomian interpolacyjny jest względnie niskiego stopnia. W przypadku funkcji posiadających wiele blisko siebie położonych ekstremów lokalnych (teren zróżnicowany, górzysty lub z licznymi wzniesieniami i obniżeniami – Genoa-Rapallo, Tczew Starogard) uzyskanie przybliżenia, które można uznać za wystarczające, wymagałoby zwiększenia rzędu aproksymacji, co jednak skutkowałoby pojawieniem się bardzo dużych zaburzeń wynikających z występowania efektu Rungego.

Interpolacja funkcjami sklejanymi

W przeciwieństwie do interpolacji wielomianowej, ta technika wykorzystuje nie jedną, ale szereg funkcji, z których każda aproksymuje tylko jeden z podprzedziałów wyznaczonych przez wybrane węzły interpolacji. Przy liczbie węzłów równej n funkcji tych (tzw. splajnów) jest zatem n – 1. Implementacja jest bardziej skomplikowana niż w przypadku wcześniej omawianej metody. Algorytm ten jest także dużo bardziej kosztowny obliczeniowo, gdyż konieczne jest rozwiązanie układu $4 \cdot n$ równań liniowych. Przewagą interpolacji funkcjami sklejanymi nad interpolacją wielomianem Lagrange'a jest jednak to, że nie wykorzystuje ona wielomianów wysokich stopni – w praktyce zadowalające rezultaty uzyskuje się stosując wielomiany stopnia 3. Ten sposób aproksymacji funkcji pozwala zatem uniknąć niekorzystnego efektu Rungego, co widać na pierwszy rzut oka na poniższych wykresach przedstawiających wyniki.



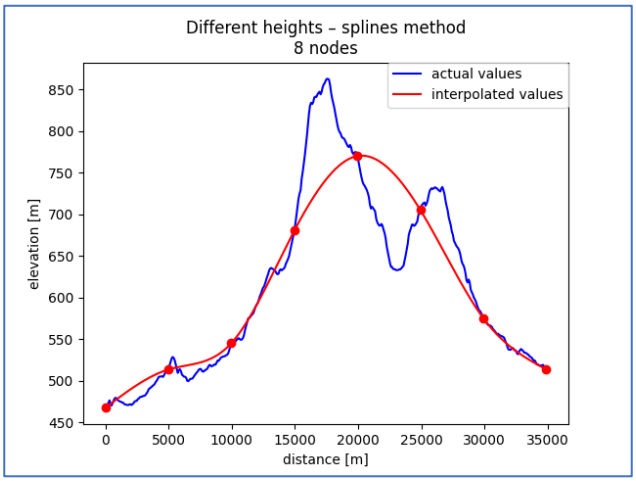
Zrzut ekranu nr 5 – Czas wykonania interpolacji funkcjami sklejanymi

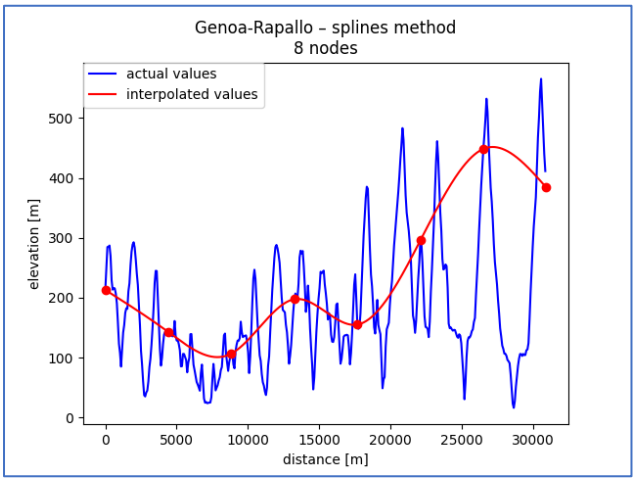
```
for (int i = 1; i < nodes_number - 1; i++) {
    h = samples[i].x - samples[i - 1].x;

    M->setAAt(4 * i, 4 * i, 1);
    b[4 * i] = samples[i].y;

    M->setAAt(4 * i + 1, 4 * i, 1);
    M->setAAt(4 * i + 1, 4 * i + 1, h);
    M->setAAt(4 * i + 1, 4 * i + 2, pow(h, 2));
    M->setAAt(4 * i + 1, 4 * i + 3, pow(h, 3));
    b[4 * i + 1] = samples[i + 1].y;

    M->setAAt(4 * i + 2, 4 * (i - 1) + 1, 1);
    M->setAAt(4 * i + 2, 4 * (i - 1) + 2, 2 * h);
    M->setAAt(4 * i + 2, 4 * (i - 1) + 3, 3 * pow(h, 2));
    M->setAAt(4 * i + 2, 4 * i + 1, -1);
    b[4 * i + 2] = 0;

    M->setAAt(4 * i + 3, 4 * (i - 1) + 2, 2);
    M->setAAt(4 * i + 3, 4 * (i - 1) + 3, 6 * h);
    M->setAAt(4 * i + 3, 4 * i + 2, -2);
    b[4 * i + 3] = 0;
}
```

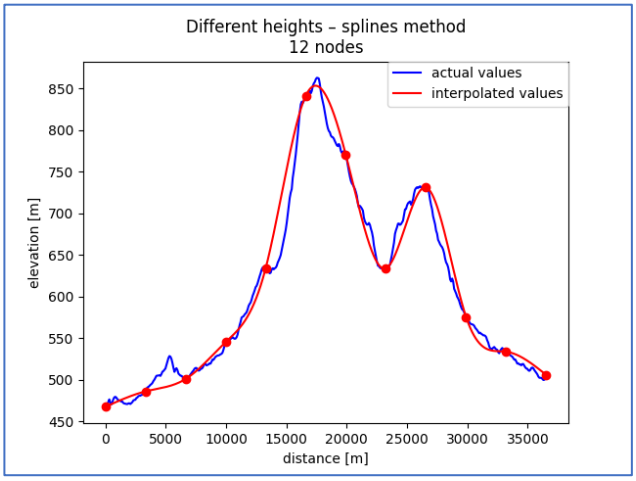
Zrzut ekranu nr 6 – Jeden z etapów budowania systemu równań

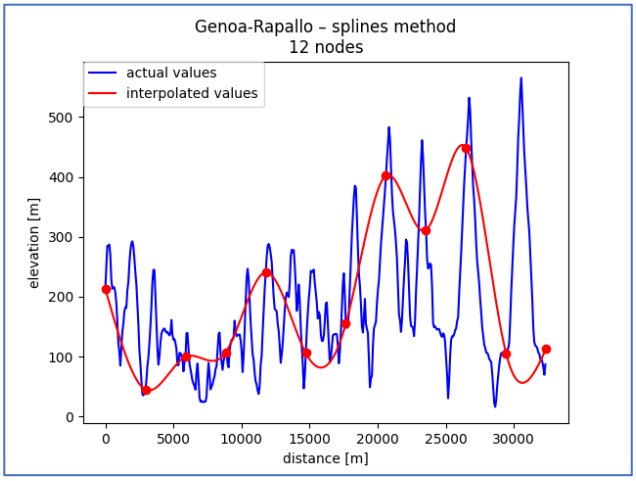
```
lowerUpperDecomposition(M, b, x, N);

double elevation{};
for (int i = 0; i < nodes_number - 1; i++) {
    elevation = 0;

    if (distance >= samples[i].x && distance <= samples[i + 1].x) {
        for (int j = 0; j < 4; j++) {
            double h = distance - samples[i].x;

            elevation += x[4 * i + j] * pow(h, j);
        }
        break;
    }
}
```

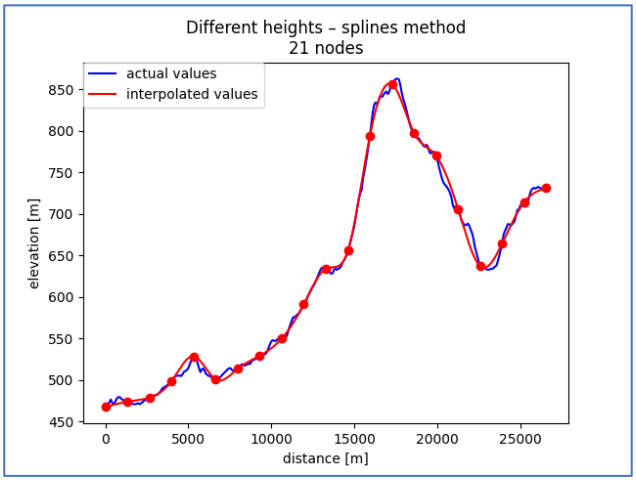
Zrzut ekranu nr 7 – Rozwiązanie systemu równań

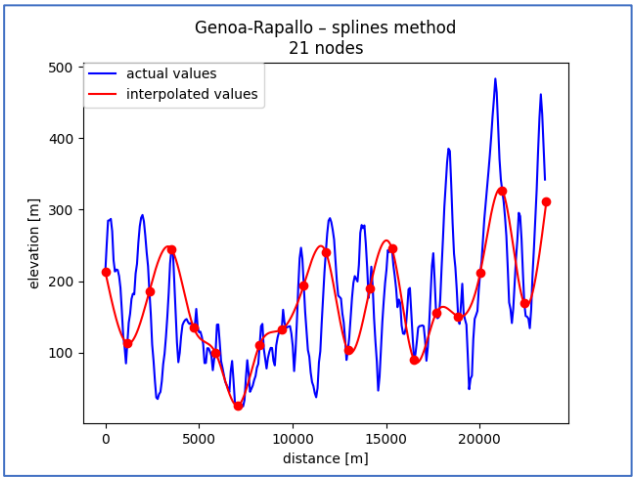

Different heights – splines method
8 nodes
actual values
interpolated values
elevation [m]
distance [m]

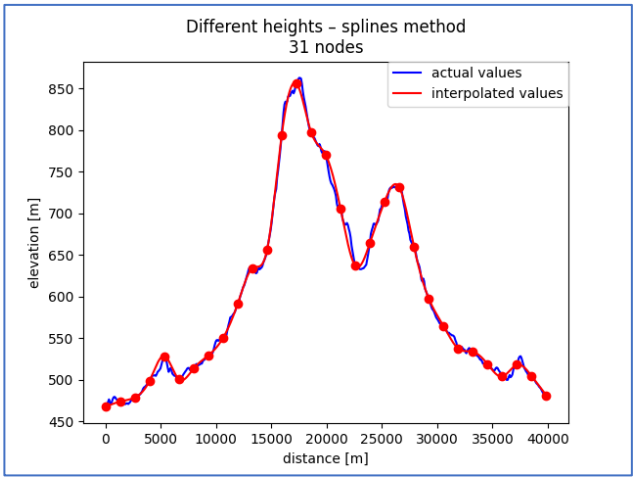

Genoa-Rapallo – splines method
8 nodes
actual values
interpolated values
elevation [m]
distance [m]

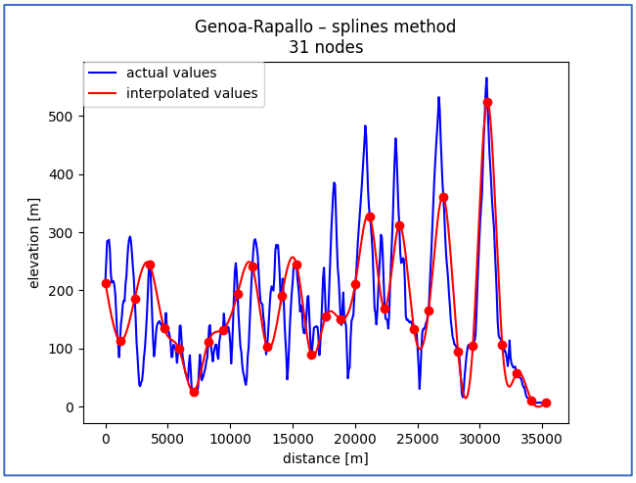

Different heights – splines method
12 nodes
actual values
interpolated values
elevation [m]
distance [m]

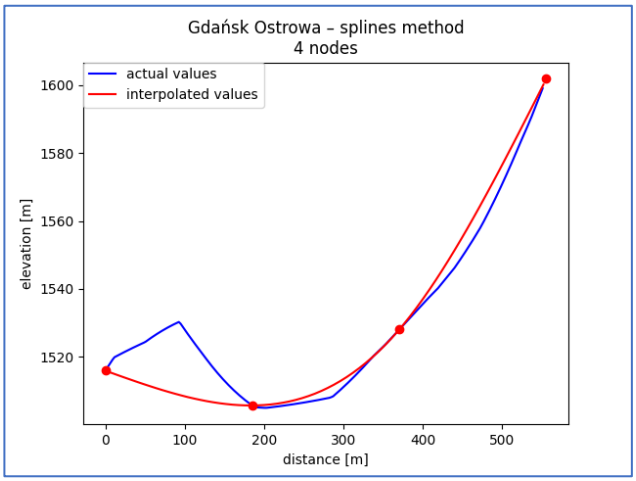

Genoa-Rapallo – splines method
12 nodes
actual values
interpolated values
elevation [m]
distance [m]

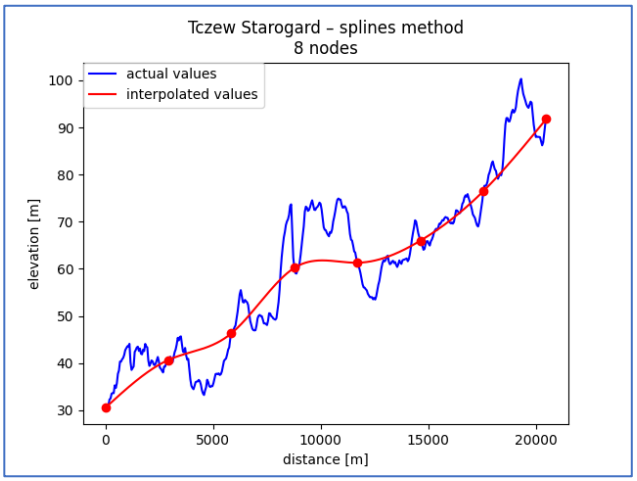

Different heights – splines method
21 nodes
actual values
interpolated values
elevation [m]
distance [m]

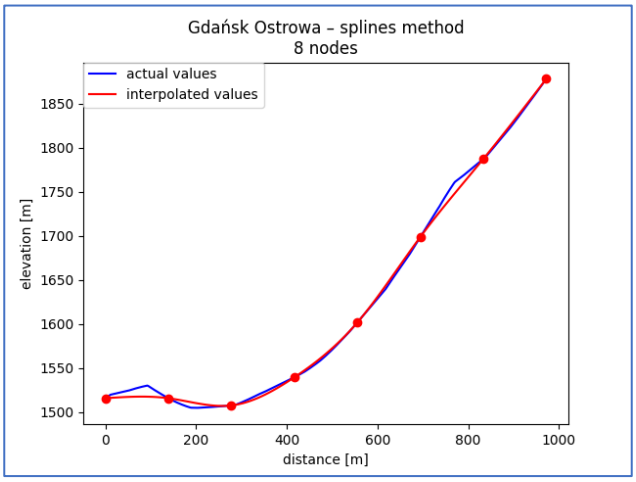

Genoa-Rapallo – splines method
21 nodes
actual values
interpolated values
elevation [m]
distance [m]

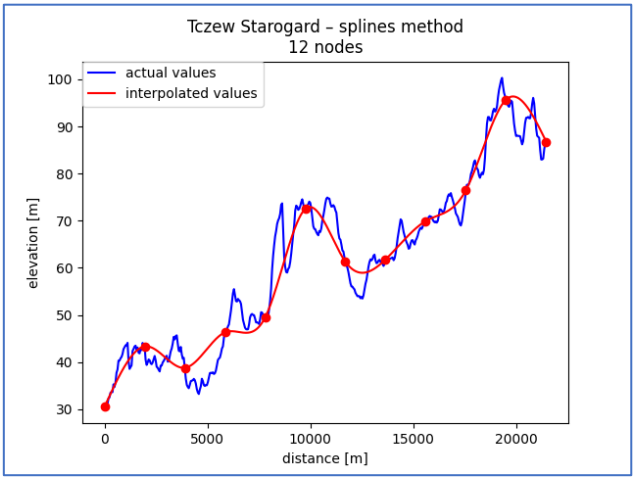

Different heights – splines method
31 nodes
actual values
interpolated values
elevation [m]
distance [m]

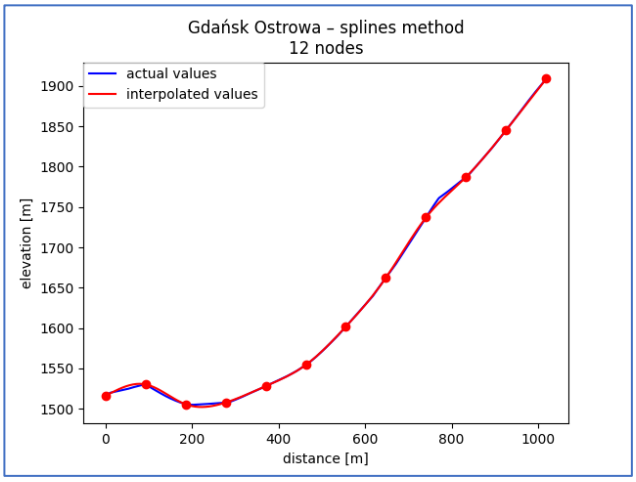

Genoa-Rapallo – splines method
31 nodes
actual values
interpolated values
elevation [m]
distance [m]

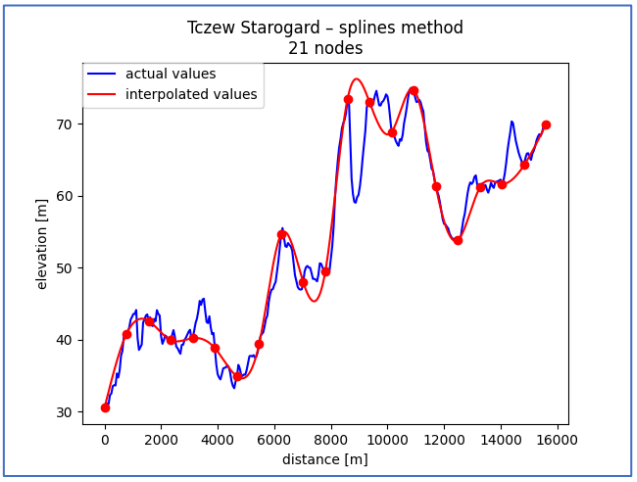

Gdańsk Ostrowa – splines method
4 nodes
actual values
interpolated values
elevation [m]
distance [m]

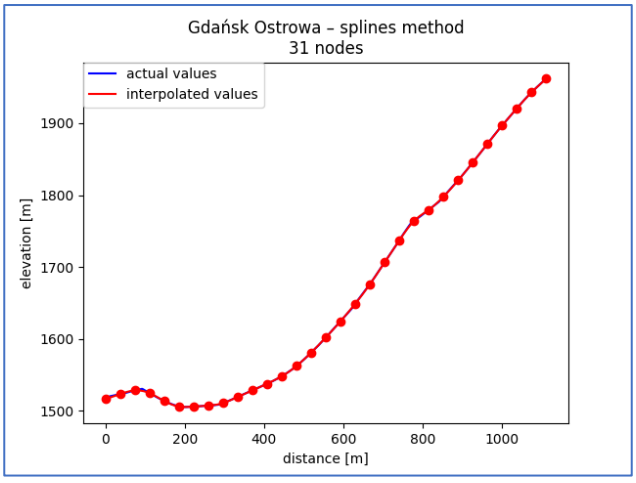

Tczew Starogard – splines method
8 nodes
actual values
interpolated values
elevation [m]
distance [m]

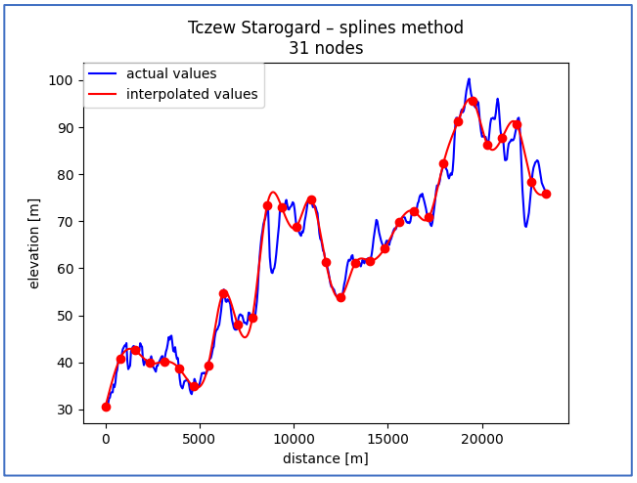

Gdańsk Ostrowa – splines method
8 nodes
actual values
interpolated values
elevation [m]
distance [m]


Tczew Starogard – splines method
12 nodes
actual values
interpolated values
elevation [m]
distance [m]


Gdańsk Ostrowa – splines method
12 nodes
actual values
interpolated values
elevation [m]
distance [m]


Tczew Starogard – splines method
21 nodes
actual values
interpolated values
elevation [m]
distance [m]


Gdańsk Ostrowa – splines method
31 nodes
actual values
interpolated values
elevation [m]
distance [m]


Tczew Starogard – splines method
31 nodes
actual values
interpolated values
elevation [m]
distance [m]

Podobnie jak w przypadku wcześniej przedstawionej techniki interpolacji wielomianowej, wykorzystanie niewielkiej liczby węzłów interpolacji sprawdza się wyłącznie wtedy, gdy aproksymowana funkcja jest „gładka" – nie posiada licznych, gęsto ulokowanych ekstremów lokalnych. W przeciwnej sytuacji, ze względu na rzadkie rozmieszczenie węzłów, wiele z tych ekstremów jest „pomijanych", bowiem zawierają się wewnątrz przedziałów wyznaczonych przez kolejne węzły interpolacji. Jeśli zatem występują liczne gwałtowne zmiany wartości badanej funkcji (uskoki, stromizny, silne pofałdowanie terenu – Genoa-Rapallo, Tczew Starogard) niezbędne okazuje się wykorzystanie większej liczby węzłów, tak aby możliwie jak najwięcej tych zmian zostało zarejestrowanych. Jednakże, w przeciwieństwie do interpolacji Lagrange'a, dzięki niewystępowaniu efektu Rungego ten sposób interpolowania funkcji umożliwia bezpieczne zwiększanie liczby węzłów interpolacji, bez narażania się na niekorzystne oscylacje i wynikającą z nich utratę czytelności wyniku. Jest to zatem znacznie bardziej przydatna metoda, gdyż pozwala na dowolne manipulowanie liczbą węzłów interpolacyjnych, a co za tym idzie – dokładnością rezultatu.

Podsumowanie

Po przeanalizowaniu otrzymanych wyników wydawać by się mogło, że wniosek jest oczywisty – interpolacja splajnami jest znacznie lepszą metodą aproksymacji, dającą więcej możliwości w kwestii doboru stopnia dokładności oraz zapewniającą bardziej wiarygodne i czytelne rezultaty. Analizując problem wyłącznie pod kątem skuteczności algorytmu rzeczywiście tak jest, jednak w praktyce często istotnym czynnikiem, niekiedy wręcz krytycznym, jest wydajność. Biorąc to pod uwagę trzeba uwzględnić znaczącą różnicę w złożoności zaprezentowanych metod, która przekłada się na czas przetwarzania danych. Interpolacja funkcjami sklejanymi pozwala uzyskać bardzo dobre wyniki, ale wiąże się to z ogromną liczbą wykonywanych operacji, która znacząco wzrasta z każdym dodanym węzłem interpolacji. Po przeanalizowaniu problemu może okazać się, że koszt obliczeniowy nie jest możliwy do zaakceptowania. Możliwe także, że w danej sytuacji znajomość dokładnej postaci funkcji nie jest tak znacząca, jak poznanie trendu, jaki ona wyznacza – w takim przypadku warto rozważyć skorzystanie ze znacznie szybszej metody interpolacji wielomianowej w połączeniu z niewielką liczbą węzłów interpolacji, co może wystarczyć do oszacowania ogólnej tendencji. Najczęściej ostateczny wybór metody aproksymacji stanowić musi kompromis między oczekiwaniami/wymaganiami dotyczącymi dokładności wyniku oraz wydajności algorytmu.