Laboratorium z Metod Numerycznych: Interpolacja i analiza błędów

Patryk Blacha, Radosław Szepielak

5 kwietnia 2025

Spis treści

1 Wprowadzenie

W pierwszym zadaniu analizujemy średnią geometryczną odległości punktów na osi rzędnych dla trzech różnych rozkładów punktów: punktów Czebyszewa, punktów Legendre'a oraz punktów rozmieszczonych równomiernie. W drugim zadaniu przeprowadzamy interpolację funkcji przy użyciu różnych metod, takich jak wielomiany Lagrange'a i funkcje sklejane, a także analizujemy dokładność tych metod.

2 Zadanie 1: Średnia geometryczna odległości punktów

Zadaniem jest obliczenie średniej geometrycznej odległości punktów na osi rzędnych dla trzech rozkładów punktów: Czebyszewa, Legendre'a oraz rozmieszczonych równomiernie na przedziale [-1, 1]. Obliczamy tę odległość dla różnych wartości $n = 10, 20, 50$.

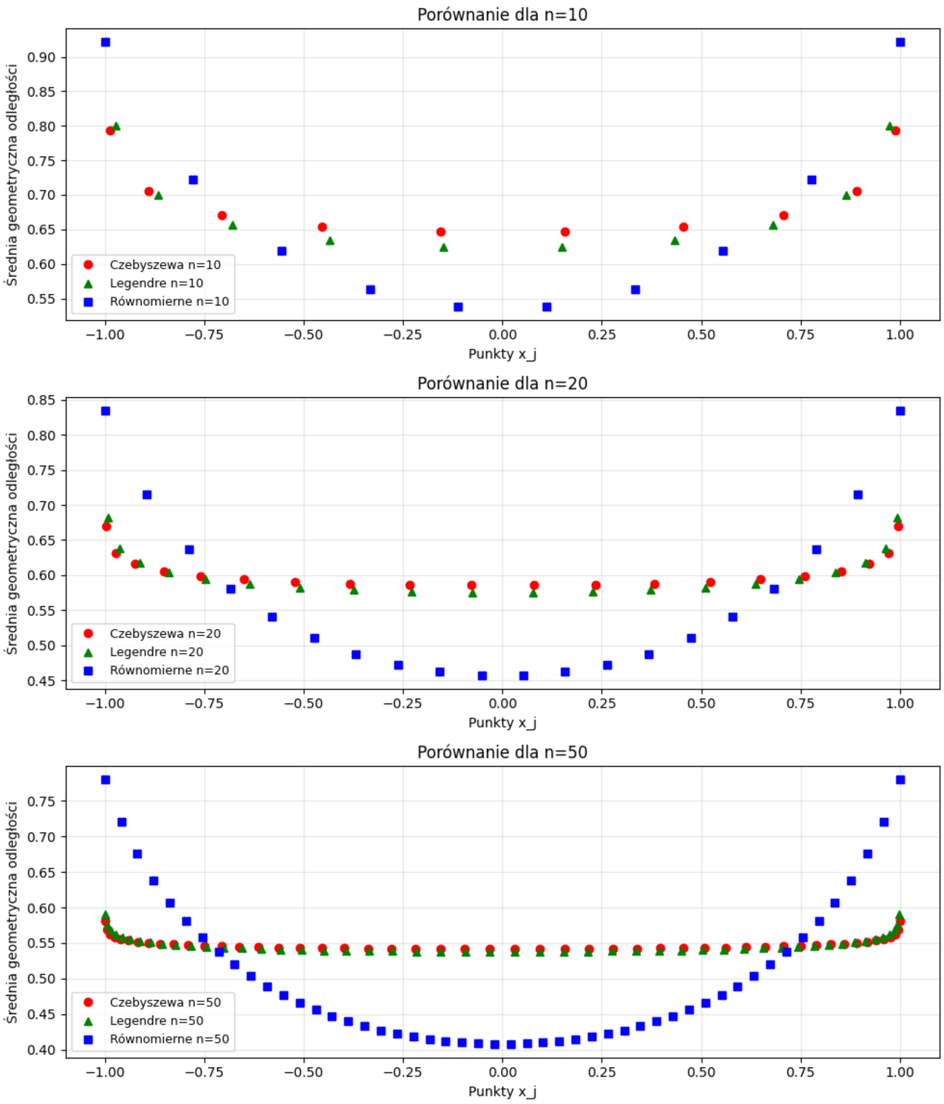
2.1 Obliczenia i wykresy

Do obliczenia średniej geometrycznej odległości każdego punktu do pozostałych użyto funkcji `geometric_mean_distance`, której kod jest poniżej:

```python
import numpy as np

def geometric_mean_distance(points):
    n = len(points)
    result = np.zeros(n)
    for i in range(n):
        distances = np.abs(points[i] - np.delete(points, i))
        result[i] = np.exp(np.mean(np.log(distances)))
    return result
```

Wyniki dla punktów Czebyszewa, Legendre'a oraz punktów równomiernych zostały przedstawione na wykresach poniżej.



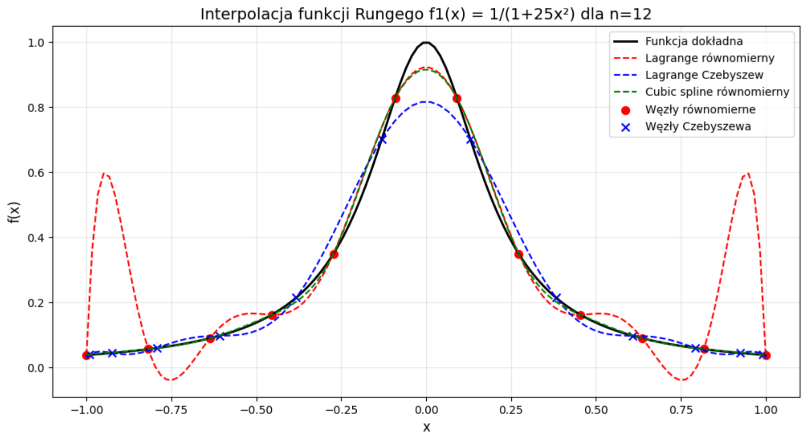
Rysunek 1: Średnia geometryczna odległości punktów na osi rzędnych dla różnych rozkładów punktów.

3 Zadanie 2: Interpolacja funkcji

W drugim zadaniu dokonaliśmy interpolacji funkcji $f_1(x)$ oraz $f_2(x)$ na różnych przedziałach. Do interpolacji użyto wielomianów Lagrange'a oraz funkcji sklejanych. Dodatkowo, przeprowadziliśmy analizę błędów interpolacji dla różnych liczby węzłów.

3.1 Interpolacja funkcji Rungego

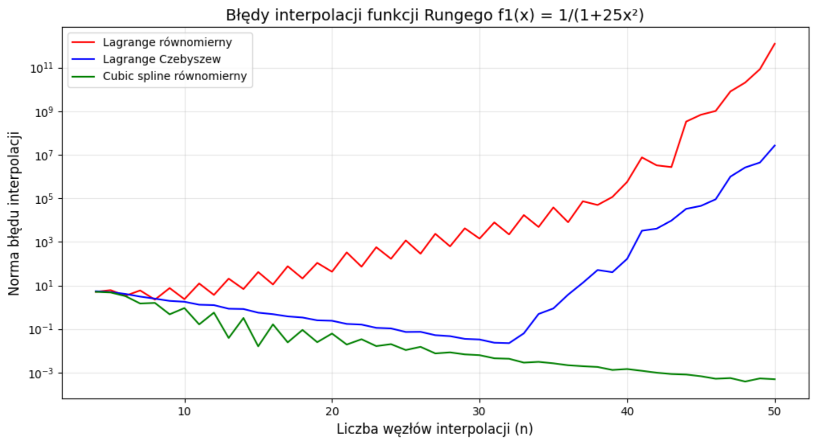
Funkcja $f_1(x) = \frac{1}{1+25x^2}$ na przedziale $[-1, 1]$ została zinterpolowana przy użyciu węzłów równomiernych oraz węzłów Czebyszewa. Wyniki interpolacji przedstawiono na poniższym wykresie.



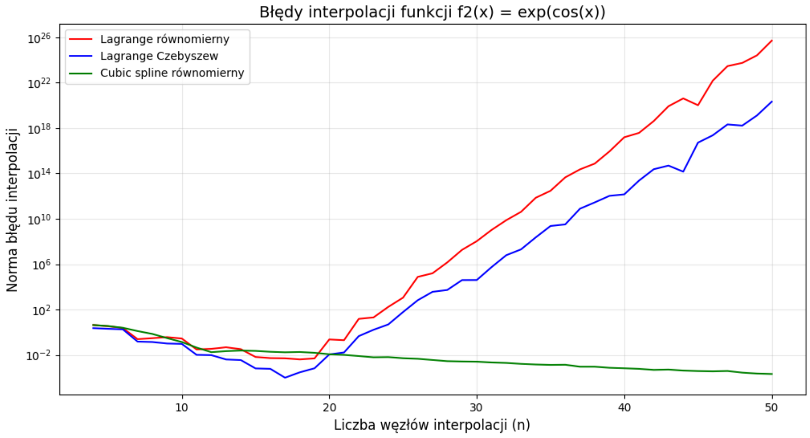
Rysunek 2: Interpolacja funkcji Rungego przy użyciu różnych metod dla $n = 12$.

3.2 Analiza błędów interpolacji

Dla różnych wartości $n = 4, 5, \ldots, 50$ obliczono błędy interpolacji dla funkcji $f_1(x)$ oraz $f_2(x)$. Błędy interpolacji przedstawiono na poniższych wykresach.



Rysunek 3: Błędy interpolacji funkcji Rungego $f_1(x)$ w zależności od liczby węzłów n.



Rysunek 4: Błędy interpolacji funkcji $f_2(x)$ w zależności od liczby węzłów n.

4 Wnioski

W analizie zadania 1 zauważono, że punkty Czebyszewa i Legendre'a zapewniają lepszą równomierność rozkładu, co wpływa na zmniejszenie średniej geometrycznej odległości. Z kolei w zadaniu 2 metoda interpolacji przy użyciu funkcji sklejanych okazała się bardziej dokładna niż metoda wielomianów Lagrange'a, szczególnie w przypadku większej liczby węzłów. Interpolacja funkcjami sklejanymi wykazała się również mniejszymi błędami w porównaniu do interpolacji wielomianami Lagrange'a, zwłaszcza dla funkcji Rungego.

- Interpolacja wielomianowa na węzłach równoodległych nie jest stabilna dla dużych n, co prowadzi do efektu Rungego.
- Węzły Czebyszewa poprawiają jakość interpolacji, zmniejszając oscylacje na końcach przedziału.
- Interpolacja splajnowa jest znacznie bardziej stabilna i skuteczna w aproksymacji funkcji, szczególnie dla większej liczby węzłów.
- Dla funkcji o dużych zmianach krzywizny wielomiany interpolacyjne mogą prowadzić do dużych błędów, co sugeruje preferowanie metod splajnowych.

4.1 Wnioski szczegółowe dla wykresu błędów interpolacji

Dla wykresu błędów interpolacji funkcji Rungego:

- **Metoda Lagrange'a na węzłach równomiernych** (wykres czerwony) generuje wykres błędów rozpoczynający się od wartości $5.6 \cdot 10^0$. Do około $n = 37$ błąd rośnie stosunkowo powoli, przy czym widoczne są charakterystyczne, nieregularne „schodki". W tym zakresie błąd osiąga wartość $8.3 \cdot 10^4$. Dla $n > 37$ następuje gwałtowny wzrost błędu, który osiąga wartość $2.03 \cdot 10^{12}$.
- **Metoda Lagrange'a na węzłach Czebyszewa** (wykres niebieski) rozpoczyna się od błędu $5.95 \cdot 10^0$, który do około $n = 33$ systematycznie maleje — bez widocznych schodków — osiągając wartość $2.3 \cdot 10^{-2}$. Dla $n > 33$ również obserwujemy gwałtowny wzrost błędu, który osiąga wartość $4.13 \cdot 10^7$.
- **Metoda funkcji sklejanych (Cubic spline)** z węzłami równomiernymi (wykres zielony) rozpoczyna się od błędu $5.58 \cdot 10^0$. Wraz ze wzrostem liczby węzłów błąd systematycznie maleje. Na początku pojawiają się delikatne schodki, które około $n = 20$ niemal całkowicie się wygładzają, aż do końcowej wartości błędu $4.45 \cdot 10^{-4}$.

Wnioskujemy zatem, że **metoda Cubic spline** charakteryzuje się największą stabilnością numeryczną. Niezależnie od liczby węzłów interpolacyjnych, uzyskiwane błędy są bardzo niskie w porównaniu do pozostałych metod.

Dla wykresu błędów interpolacji funkcji $\exp(\cos(x))$:

- **Metoda Lagrange'a na węzłach równomiernych** (wykres czerwony) zaczyna się od błędu $4.34 \cdot 10^0$. Do około $n = 19$ obserwujemy malejącą tendencję błędu, osiągając wartość $3.4 \cdot 10^{-3}$. Dla $n > 19$ następuje gwałtowny wzrost aż do ekstremalnej wartości $6.28 \cdot 10^{25}$.
- **Metoda Lagrange'a na węzłach Czebyszewa** (wykres niebieski) startuje od wartości $2.35 \cdot 10^0$. Do $n = 19$ błąd systematycznie maleje, osiągając wartość $1.0 \cdot 10^{-4}$. Dla $n > 19$ wykres również gwałtownie rośnie, osiągając wartość $2.64 \cdot 10^{20}$.
- **Metoda funkcji sklejanych (Cubic spline)** na węzłach równomiernych (wykres zielony) rozpoczyna się od błędu $4.34 \cdot 10^0$ i wykazuje silną tendencję malejącą, aż do wartości końcowej $2.8 \cdot 10^{-4}$ przy $n = 50$.

Również w tym przypadku **metoda Cubic spline** wykazuje najwyższą stabilność numeryczną — błędy interpolacji są niewielkie w całym zakresie liczby węzłów i nie ulegają nagłym skokom.

Widzimy zatem, że metoda Cubic spline cechuje się najniższymi błędami interpolacji w porównaniu do innych metod dla każdej liczby węzłów.

5 Literatura

- Dokumentacja Python: numpy, scipy, matplotlib.