Zadanie Numeryczne 12

Mateusz Wojtyna

1. Wstęp

Należało skonstruować interpolację według algorytmu Floatera I Hormanna z parametrem $d = 3$ dla funkcji

$$f(x) = \frac{1}{1 + 5x^2}, \quad x \in [-1.25, 1.25]$$

w równoodległych węzłach $-\frac{7}{8}$, $-\frac{5}{8}$, $-\frac{3}{8}$, $-\frac{1}{8}$, $\frac{1}{8}$, $\frac{3}{8}$, $\frac{5}{8}$, $\frac{7}{8}$.

2. Opis

Według algorytmu Floatera i Hormanna funkcja interpolująca w przypadku równoodległych węzłów $x_0, x_1, ..., x_n$ dana jest przez

$$r(x) = \frac{\sum_{k=0}^{n} \frac{w_k}{x - x_k} f(x_k)}{\sum_{k=0}^{n} \frac{w_k}{x - x_k}}$$

$$w_k = (-1)^{k-d} \sum_{i \in J_k} \binom{d}{k-i}$$

gdzie $J_k = \{i \in I : k - d \leq i \leq k\}$, $I = \{0, 1, ..., n - d\}$, $d \in \mathbb{R}$.

3. Kod

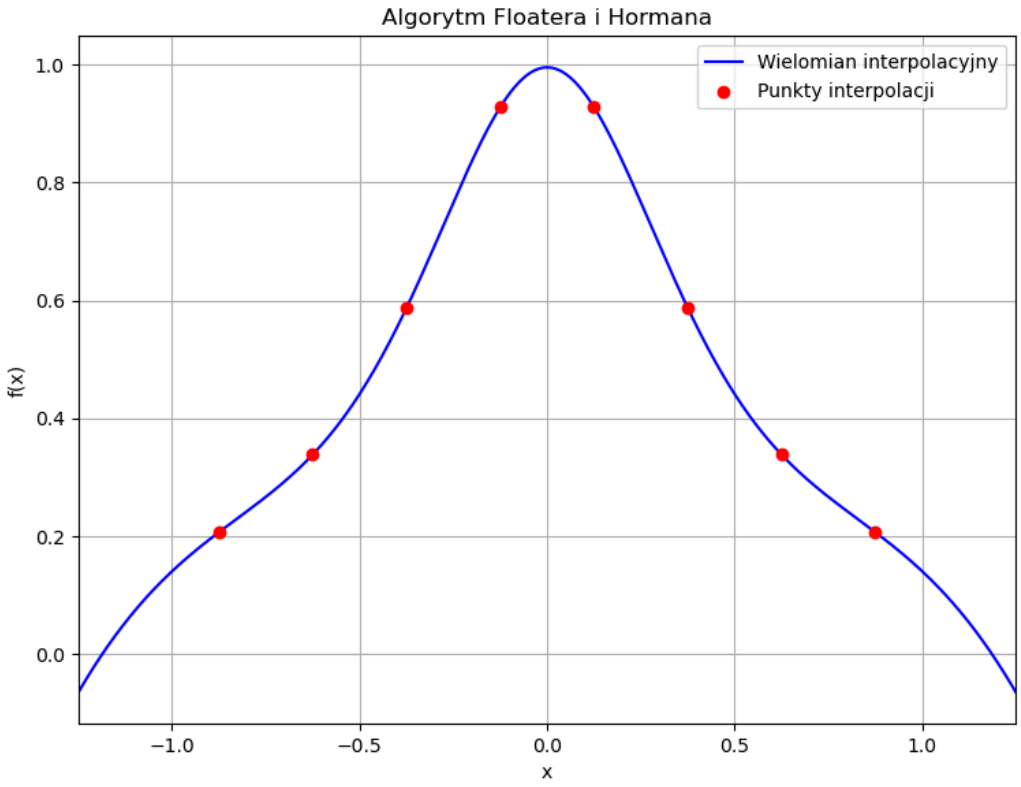
Program napisano w Pythonie z użyciem pakietów *NumPy* oraz *Matplotlib*.

```
import numpy as np
import matplotlib.pyplot as plt
import math
from numpy.typing import NDArray

array = NDArray[np.float64]
```

```
d = 3
eps = 1e-12


def f(x: float):
    return 1 / (1 + 5 * (x**2))


def w(k: int, n: int):
    res = 0
    for i in range(max(k - d, 0), min(k, n - d) + 1):
        res += math.comb(d, k - i)

    if (k - d) % 2 == 1:
        res *= -1

    return res


def r(x: int, n: int, nodes: array, values: array, weights):
    nominator = 0

    for k in range(0, n + 1):
        if abs(x - nodes[k]) > eps:
            nominator += (weights[k] / (x - nodes[k])) * values[k]
        else:
            return nodes[k]

    denominator = 0
    for k in range(0, n + 1):
        denominator += weights[k] / (x - nodes[k])

    return nominator / denominator


def main():
    np.set_printoptions(suppress=True)

    nodes = np.array([-7 / 8, -5 / 8, -3 / 8, -1 / 8, 1 / 8, 3 / 8, 5 / 8, 7 / 8])
    values = np.array([f(x) for x in nodes])
    n = len(nodes) - 1
    weights = [w(k, n) for k in range(0, n + 1)]

    x_plot = np.linspace(-1.25, 1.25, 1000)
    y_plot = np.array([r(x, n, nodes, values, weights) for x in x_plot])
    plt.plot(x_plot, y_plot, label="Wielomian interpolacyjny", color="blue")
```

```
    plt.scatter(nodes, values, color="red", label="Punkty interpolacji", zorder=5)
    plt.xlabel("x")
    plt.ylabel("f(x)")
    plt.title("Algorytm Floatera i Hormana")
    plt.grid(True)
    plt.legend()
    plt.xlim(-1.25, 1.25)
    plt.show()


if __name__ == "__main__":
    main()
```

4. Wyniki



Rysunek 1: Interpolacja funkcji f według algorytmu Floatera i Hormanna w przedziale $[-1.25, 1.25]$.