Metody numeryczne

Mateusz Kwolek

7. Znajdź, z dokładnością do czterech cyfr dziesiętnych, wartości współczynników wielomianu interpolacyjnego opartego na następującej tabelce (ze względów typograficznych tabelka ma orientację pionową, a nie, jak to jest zwyczajowo, poziomą; x oznacza węzeł, $f(x)$ wartość funkcji w węźle):

x	$f(x)$
-1.00	6.00000000000000
-0.75	3.04034423828125
-0.50	1.74218750000000
-0.25	1.26361083984375
0.25	0.75982666015625
0.50	0.63281250000000
0.75	0.85809326171875
1.00	2.00000000000000

Sporządź wykres uzyskanego wielomianu w przedziale $-2 \leqslant x \leqslant 1.25$ i zaznacz na nim punkty, które posłużyły do jego konstrukcji.
Uwaga: Jeśli zadanie to zostanie wykonane prawidłowo, obliczone współczynniki wielomianu interpolacyjnego będą "ładne". Należy użyć wszystkich cyfr znaczących podanych w treści zadania.

1) Wybór algorytmu

Zaimplementowany przeze mnie algorytm to interpolacja Lagrange'a / wielomianowa. Metoda ta umożliwia przybliżenie wzoru funkcji jeżeli dysponuje się skończoną liczbą danych określających zachowanie funkcji.

2) Kod źródłowy

Do stworzenia programu użyłem języka Python ze względu na jego prostotę, bogactwo bibliotek oraz częste zastosowanie do wykonywania obliczeń matematycznych.
Do wykonania wykresu tempa zbieżności metod użyłem biblioteki matplotlib.
Po uruchomieniu programu zostanie wyświetlone okno z wykresem gdzie poruszając się możemy dokładnie przyjrzeć się wynikom. Te zostaną również zapisane w pliku png.

```python
import numpy as np
import matplotlib.pyplot as plt
np.set_printoptions(suppress=True, linewidth=1000)

# Zadane punkty określające zachowanie funkcji
xy = np.array([[-0.75, 1.1309204101562500],
               [-0.50, 2.3203125000000000],
               [-0.25, 1.9284057617187500],
               [0.00, 1.0000000000000000],
               [0.25, 0.0554809570312500],
               [0.50, -0.6015625000000000],
               [0.75, -0.7525024414062500],
               [1.00, 0.0000000000000000]])

# Algorytm interpolacji wielomianowej Lagrange'a
def lagrange_interpolation(x, y):
    n = len(x)
    coefficients = np.zeros_like(x, dtype=float)

    for i in range(n):
        numerator = np.poly1d([1])
        denominator = 1

        for j in range(n):
            if j != i:
                numerator *= np.poly1d([1, -x[j]])
                denominator *= (x[i] - x[j])

        coefficients += (numerator / denominator) * y[i]

    coefficients = np.round(coefficients, decimals=4)
    return coefficients


coefficients = lagrange_interpolation(xy[:, 0], xy[:, 1])

# Wypisanie wyniku
print("Coefficients of polynomial :", coefficients)

# Konstrukcja wykresu
x = np.linspace(-1.25, 1.25)
y = np.polyval(coefficients, x)

plt.scatter(xy[:, 0], xy[:, 1], color = "RED", label="Given points")
plt.plot(x, y, color = "BLUE", label="Interpolation polynomial")
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.grid()

plt.savefig('wykres.png')
plt.show()

```

3) Wynik

```
Coefficients of polynomial : [ 1. -1.  1. -2.  4. -0. -4.  1.]
```

4) Wykres