zmlab2_zadania

December 5, 2023

1 Interpolacja wielomianowa

2 Błedy i arytmetyka zmiennopozycyjna

Całość kodu można znaleźć w formie online pod linkiem github.com/KsawerySmoczynski/elementy-metod-numerycznych.

Jest on również dostępny pod tym linkiem w wersji interaktywnej w Google Colab

[1]:
```
# Żeby odpalić notebook na google colab będziemy musieli zainstalować potrzebne␣
↪pakiety
import subprocess
import sys
import os

REQUIRED_PACKAGES = ["numpy", "matplotlib", "scipy"]
RUN_IN_COLAB = "COLAB_JUPYTER_IP" in os.environ

def install(packages):
    subprocess.check_call([sys.executable, "-m", "pip", "install", *packages])

if RUN_IN_COLAB:
    install(REQUIRED_PACKAGES)
```

2.1 Zadania

2.1.1 Zadanie 1.

Napisz funkcję, która dla zadanego wielomianu w, wektora węzłów $(x_0, x_1, \ldots, x_n)$ oraz wektora odpowiadających im wartości $(y_0, y_1, \ldots, y_n)$ sprawdza, czy podany wielomian jest wielomianem Lagrange'a interpolującym te dane.

[2]:
```
import numpy as np
import numpy.polynomial.polynomial as P

# Example
x = np.array([6., 7., 8., 9.])
y = np.array([-3., 6., 0., 9.])
w = np.array([-2052, 836.5, -112.5, 5])
```

[3]:
```python
def is_lagrange_polynomial(w, x, y) -> bool:
    # zdefiniuj funkcję
```

[4]:
```python
is_lagrange_polynomial(w, x, y)
```

[4]: True

2.1.2 Zadanie 2.

(* 4 pkt) Napisz funkcję, która dla wektora $n + 1$ różnych punktów $(x_0, x_1, \dots, x_n)$ i wartości pewnej funkcji f w tych punktach zwraca wektor $(b_0, b_1, \dots, b_n)$ współczynników wielomianu interpolacyjnego Lagrange'a funkcji f w postaci Newtona opartego na węzłach $x_0, x_1, \dots, x_n$.

[5]:
```python
x = np.array([6., 7., 8., 9.])
y = np.array([-3., 6., 0., 9.])

def get_lagrange_coefficients(x, y):
    # Zdefiniuj funkcję
    return lagrange_coefficients
```

2.1.3 Zadanie 3.

($*$3 pkt) Napisz funkcję, która dla danych liczb rzeczywistych a, b $(a < b)$ i liczby naturalnej n oblicza wartości $n + 1$ węzłów Czebyszewa w przedziale $[a, b]$, czyli wartości:

$$x_j = \frac{b-a}{2}\cos(\frac{2j+1}{2n+2}\pi) + \frac{a+b}{2} \text{ dla } j = 0, 1, \dots, n.$$

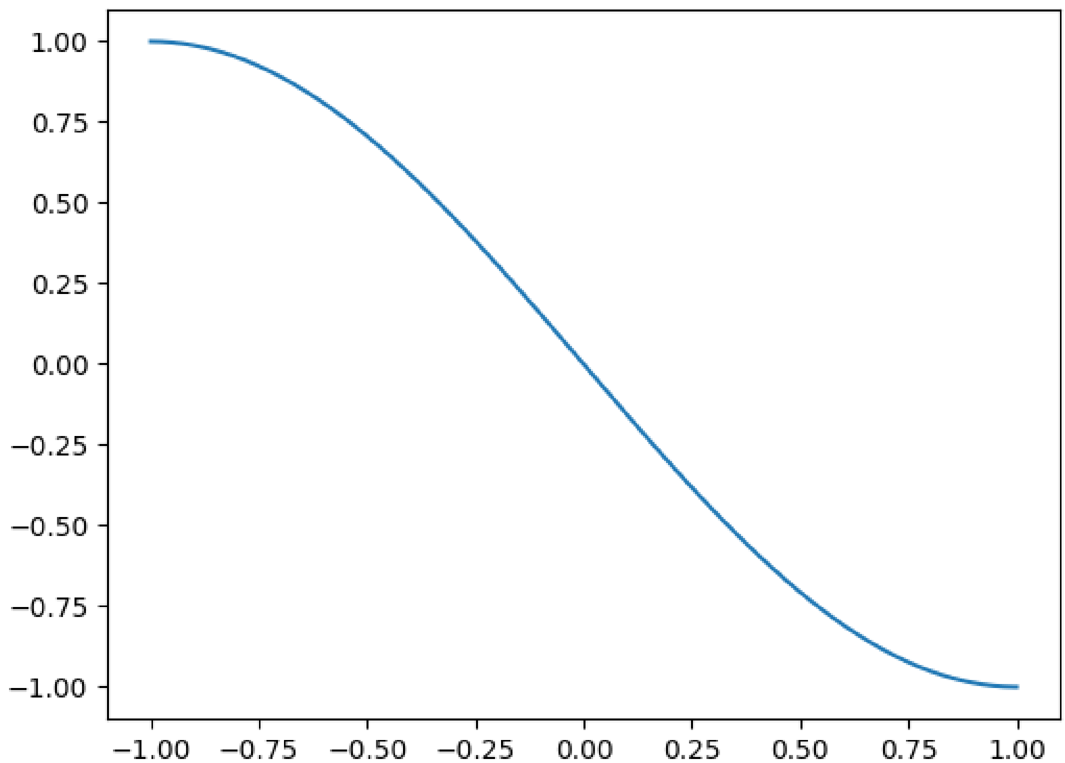
[6]:
```python
[a, b] = [-1, 1]
n = 1000
[a, b], n
```

[6]: ([-1, 1], 1000)

[7]:
```python
def get_chebyshev_nodes(a, b, n):
    # zdefiniuj funkcję
    return chebyshev_nodes
```

[8]:
```python
chebyshev_nodes = get_chebyshev_nodes(a, b, n)
```

[9]:
```python
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
# Zdefiniuj wykres
plt.show()
```



2.1.4 Zadanie 4.

(* 2 pkt) Rozważmy funkcję $f(x) = \frac{1}{1+x^2}$ w przedziale $I = [-5, 5]$.

1. Znajdź współczynniki b_i wielomianu interpolacyjnego Lagrange'a tej funkcji opartego na 6 równoodległych węzłach w przedziale I.
2. Znajdź współczynniki b_i wielomianu interpolacyjnego Lagrange'a tej funkcji opartego na 11 równoodległych węzłach w przedziale I.
3. Narysuj w jednym oknie wykresy funkcji f i dwóch obliczonych w poprzednich podpunktach wielomianów interpolacyjnych tej funkcji w przedziale I.

[10]:
```
def evaluate_lagrange_base_polynomials(x, x_nodes):
    # zdefiniuj funkcję
    return p0_n

def evaluate_lagrange(b, p):
    return # zdefiniuj funkcję
```

[11]:
```
f = lambda x: # zdefiniuj funkcję
```

[12]:
```
# 1. 6 równoodległych węzłów
x1_nodes = # zdefiniuj wartości
y1_nodes = # zdefiniuj wartości
```

```
b1 = get_lagrange_coefficients(x1_nodes, y1_nodes)
```

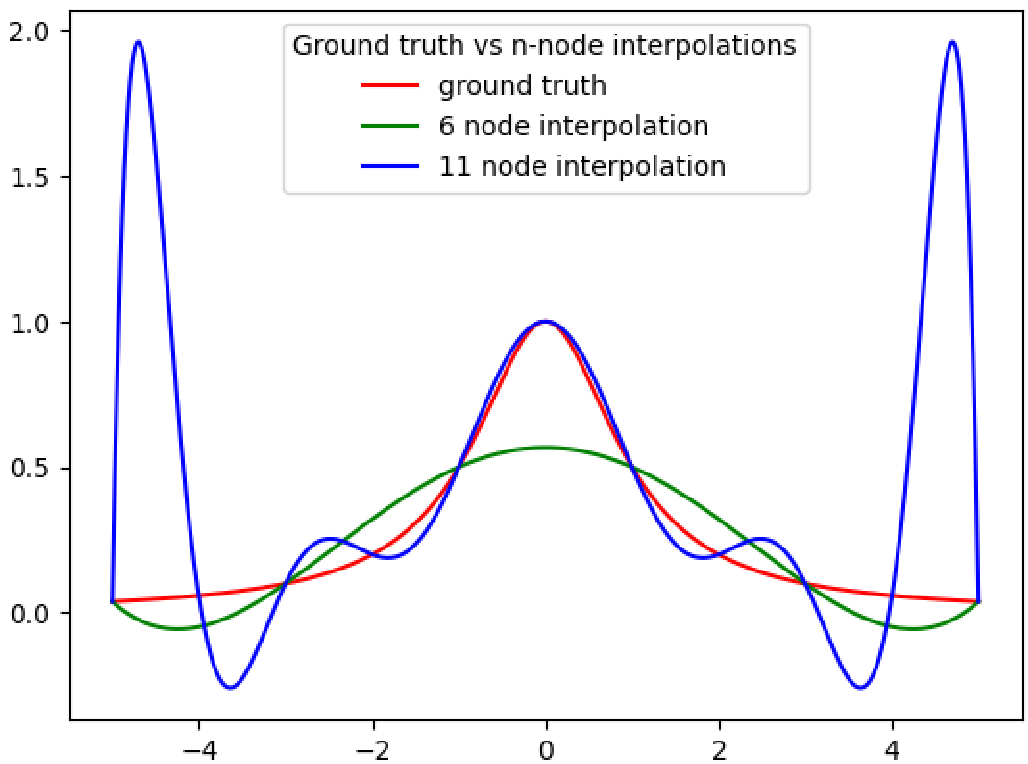
[13]:
```
# 2. 11 równoodległych węzłów
x2_nodes = # zdefiniuj wartości
y2_nodes = # zdefiniuj wartości
b2 = get_lagrange_coefficients(x2_nodes, y2_nodes)
```

[14]:
```
# 3. Wykresiki
x = np.linspace(-5, 5, 10_000)
# f(x) - czerwony
y_ground_truth = f(x)

# 6 węzłów - zielony
p1 = evaluate_lagrange_base_polynomials(x, x1_nodes)
y_6_nodes_interpolation = evaluate_lagrange(b1, p1)

# 11 węzłów - niebieski
p2 = evaluate_lagrange_base_polynomials(x, x2_nodes)
y_11_nodes_interpolation = evaluate_lagrange(b2, p2)
```

[15]:
```
fig, ax = plt.subplots()
# Zdefiniuj wykres
plt.legend(title='Ground truth vs n-node interpolations')
plt.show()
```



2.1.5 Zadanie 5.

($*$ 2 pkt) Dla funkcji f z poprzedniego zadania wyznacz współczynniki wielomianów interpolacyjnych Lagrange'a w postaci Newtona opartych na 6 i 11 węzłach Czebyszewa w przedziale $[-5, 5]$.

Następnie narysuj w jednym oknie wykresy tych wielomianów i wyjściowej funkcji w tym przedziale.

[ ]:
```
f = lambda x: # Zdefiniuj funkcję
```

[ ]:
```
# 1. 6 węzłów
[a, b] = # Zdefiniuj przedział
n1 = # Zdefiniuj ilość węzłów
x1_nodes = # Zdefiniuj węzły Czybyszewa
y1_nodes = # Zdefiniuj wartości dla tych węzłów
b1 = # Policz wartości współczynników wielomianów interpolacyjnych Lagrange'a
```

[ ]:
```
# 2. 11 węzłów
[a, b] = # Zdefiniuj przedział
n2 = # Zdefiniuj ilość węzłów
x2_nodes = # Zdefiniuj węzły Czybyszewa
y2_nodes = # Zdefiniuj wartości dla tych węzłów
b2 = # Policz wartości współczynników wielomianów interpolacyjnych Lagrange'a
```

[ ]:
```
# 3. Wykresiki
x = np.linspace(-5, 5, 10_000)
# f(x) - czerwony
y_ground_truth = f(x)

# 6 węzłów - zielony
p1 = evaluate_lagrange_base_polynomials(x, x1_nodes)
y_6_nodes_interpolation = evaluate_lagrange(b1, p1)

# 11 węzłów - niebieski
p2 = evaluate_lagrange_base_polynomials(x, x2_nodes)
y_11_nodes_interpolation = evaluate_lagrange(b2, p2)
```

[ ]:
```
fig, ax = plt.subplots(figsize=(11.5,5))
# Zdefiniuj wykres
plt.legend(title='Ground truth vs chebyshev n-node interpolation')
plt.show()
```