Obliczenia naukowe

Prowadzący: dr hab. Paweł Zieliński

wtorek TN 7^{30}

Sprawozdanie 4

Weronika Jasiak

236733

1. Zadanie 1

1.1. Opis problemu

Implementacja funkcji obliczającej ilorazy różnicowe bez użycia tablicy dwuwymiarowej (macierzy).

1.2. Rozwiązanie

Algorytm 1: Ilorazy różnicowe

Input: (x, f), gdzie:

x - wektor długości $n+1$ zawierający węzły $x_0, \ldots, x_n$
$x[1] = x_0, \ldots, x[n+1] = x_n$

f - wektor długości $n+1$ zawierający wartości interpolowanej funkcji w węzłach $f(x_0), \ldots, f(x_n)$

Output: fx, gdzie:

fx - wektor długości $n+1$ zawierający obliczone ilorazy różnicowe
$fx[1] = f[x_0], fx[2] = f[x_0, x_1], \ldots,$
$fx[n] = f[x_0, \ldots, x_{n-1}], fx[n+1] = f[x_0, \ldots, x_n]$

```
ILORAZYROZNICOWE (x, f)
n ← length(f)
for i ← 0 to n
   do fx[i] ← f[i]
end do
for j ← 1 to n
   do for i ← n to j step −1
         do fx[i] ← (fx[i] − fx[i − 1])/(x[i] − x[i − j])
      end do
end do
return fx
```

Posiadając węzły x_n i wartości interpolowanej funkcji w węzłach $f(x_n)$ można utworzyć jednowymiarową tablicę d zmiennych z jednym wskaźnikiem. Jako wartości początkowe zmiennej d_i przyjmowane są odpowiednio wartości funkcji w węzłach $f[x_i]$. Następnie każde d_i wyznaczane jest ze wzoru:

$$d_i = \frac{d_i - d_{i-1}}{x_i - x_{i-j}}$$

gdzie j oznacza numer kolumny, a i wiersz. Sposób tworzenia tablicy jest zgodny z następującymi algorytmem:

1. tworzenie kolumny,
2. wypełnianie kolumny z dołu do góry,

kroki 1-2 są powtarzane, dopóki nie zostanie osiągnięta wcześniej zadana ilość kolumn. Dzięki wykonywaniu obliczeń w takiej kolejności tablica d zawiera w danym momencie ilorazy, które będą później potrzebne.

2. Zadanie 2

2.1. Opis problemu

Implementacja funkcji obliczającej wartość wielomianu interpolacyjnego stopnia n w postaci Newtona $N_n(x)$ w punkcie $x = t$ za pomocą uogólnionego schematu Hornera, w czasie $O(n)$.

2.2. Rozwiązanie

Algorytm 2: Wielomian interpolacyjny

Input: (x, fx, t), gdzie:

x - wektor długości $n + 1$ zawierający węzły $x_0, \; \dots, x_n$
$x[1] = x_0, \dots, x[n+1] = x_n$

fx - wektor długości $n + 1$ zawierający ilorazy różnicowe
$fx[1] = f[x_0], fx[2] = f[x_0, x_1], \dots,$
$fx[n] = f[x_0, \dots, x_{n-1}], fx[n+1] = f[x_0, \dots, x_n]$

t - punkt, w którym należy obliczyć wartość wielomianu

Output: nt, gdzie:

nt - wartość wielomianu w punkcie t

WARNEWTON (x, fx, t)
1. $n \leftarrow length(fx)$
2. $nt \leftarrow fx[n]$
3. **for** $k \leftarrow n - 1$ **to** 1 **step** -1
4. **do** $nt \leftarrow \; fx[k] + (t - x[k]) * nt$
5. **end do**
6. **return** nt

Wielomian interpolacyjny Newtona można zapisać w postaci sumy:

$$\sum_{i=0}^{n} f[x_0, x_1, \dots, x_i] \prod_{j=0}^{i-1} (x - x_j)$$

gdzie $f[x_0, x_1, \dots, x_i]$ to kolejne iloczyny różnicowe, a $x_j$to węzły interpolacji.

Uogólniony algorytmu Hornera można zapisać za pomocą następujących wzorów:

$w_n(x) \coloneqq f[x_0, x_1, \dots, x_n]$

$w_k(x) \coloneqq f[x_0, x_1, \dots, x_k] + (x - x_k) w_{k+1}(x)$ dla $k = n - 1, \dots, 0$

$N_n(x) \coloneqq w_0(x)$

Korzystając z wyżej przedstawionego algorytmu została zaimplementowana funkcja wyznaczająca wartość wielomianu interpolacyjnego stopnia n w punkcie $x = t$.

3. Zadanie 3

3.1. Opis problemu

Implementacja funkcji przyjmującej jako dane wejściowe współczynniki wielomianu interpolacyjnego w postaci Newtona $c_0 = f[x_0]$, $c_1 = f[x_0, x_1]$, $c_2 = f[x_0, x_1, x_2]$, …, $c_n = f[x_0, \dots, x_n]$ oraz węzły x_0, $x_2, \dots, x_n$, następnie obliczającej w czasie $O(n^2)$ współczynniki jego postaci naturalnej $a_0, \dots, a_n$ tzn. $a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$.

3.2. Rozwiązanie

Algorytm 3: Postać naturalna

Input: (x, fx), gdzie:

x - wektor długości $n + 1$ zawierający węzły $x_0, \dots, x_n$
$x[1] = x_0, \dots, x[n+1] = x_n$

fx - wektor długości $n + 1$ zawierający ilorazy różnicowe
$fx[1] = f[x_0], fx[2] = f[x_0, x_1], \dots,$
$fx[n] = f[x_0, \dots, x_{n-1}], fx[n+1] = f[x_0, \dots, x_n]$

Output: a, gdzie:

a - wektor długości $n + 1$ zawierający obliczone współczynniki postaci naturalnej
$a[1] = a_0, a[2] = a_1, \dots,$
$a[n] = a_{n-1}, a[n+1] = a_n$

```
NATURALNA (x, fx)
n ← length(fx)
a[n] ← fx[n]
for i ← n − 1 to 1 step −1
  do z ← a[i + 1] * x[i]
     a[i] ← fx[i] − z
     for k ← i + 1 to n − 1
       do z ← a[k + 1] * x[i]
          a[k] ← a[k] − z
     end do
end do
return a
```

Współczynnik wielomianu interpolacyjnego w postaci Newtona można w sposób ogólny zapisać jako c_n, jest on równy współczynnikowi a_n stojącemu przy najwyższej potędze wielomianu w postaci naturalnej. Korzystając z zaobserwowanej zależności można posłużyć się uogólnionym algorytmem Hornera, w który występuje podobna zależność pomiędzy w_n oraz a_n. Algorytm korzystając z powyższych zależności będzie wykonywał kolejne kroki tworząc a_i w oparciu o uprzednio policzone współczynniki stojące przy najwyższych potęgach. Szukanie zależności pomiędzy a_i oraz w_i polega na przejściu po wszystkich w_i w dół i modyfikacji współczynników tak, aby dla każdego w_i przyjmować w danej chwili postać naturalną.

4. Zadanie 4

4.1. Opis problemu

Implementacja funkcji, która zinterpoluje zadaną funkcję $f(x)$ w przedziale $[a, b]$ za pomocą wielomianu interpolacyjnego stopnia n w postaci Newtona. Następnie narysuje wielomian interpolujący i interpolowaną funkcję przy użyciu węzłów równoległych $x_k = a + kh$, $h = \frac{b-a}{n}, k = 0, 1, \dots, n$.

4.2. Rozwiązanie

Algorytm 4: Wykres

Input: (f, a, b, n), gdzie:

- f - funkcja $f(x)$ zadana jako anonimowa funkcja
- a, b - przedział interpolacji
- n - stopień wielomianu interpolacyjnego

Output: funkcja rysuje wielomian interpolacyjny i interpolowaną funkcję w przedziale $[a, b]$

```
RYSUJNNFX (f, a, b, n)
h ← (b − a)/n
k ← 0
for i ← 1 to n + 1
   do x[i] ← a + (k ∗ h)
      y[i] ← f(x[i])
      k ← k + 1
end do
fx ← ILORAZYROZNICOWE(x, y)
n ← (n + 1) ∗ 33
h ← (b − a)/(n − 1)
k ← 0
for i ← 1 to n
   plotX[i] ← a + (k ∗ h)
   plotY[i] ← f(plotX[i])
   plotH[i] ← WARNEWTON(x, fx, plotX[i])
   k ← k + 1
end do
draw
```

Na samym początku funkcja wyznacza węzły oraz wartości interpolowanej funkcji w węzłach, na podstawie, których zostają wyliczone ilorazy różnicowe (przy pomocy funkcji `ilorazyRoznicowe` zaimplementowanej na potrzeby Zadania 1). Następnie stopień wielomianu interpolacyjnego zostaje poddany próbkowaniu, czyli przemnożony o wcześniej zadaną wartość, aby nie utracić precyzji wykresów. W kolejnym kroku zostają również wyznaczone węzły i wartości funkcji w węzłach oraz wartości wielomianu (przy pomocy funkcji `warNewton` zaimplementowanej na potrzeby Zadania 2). Uzyskane w ten sposób dane służą do wygenerowania wykresu z wykorzystaniem pakietu PyPlot.

5. Zadanie 5

5.1. Opis problemu

Wykonanie testów funkcji `rysujNnfx(f, a, b, n)` na zadanych danych:

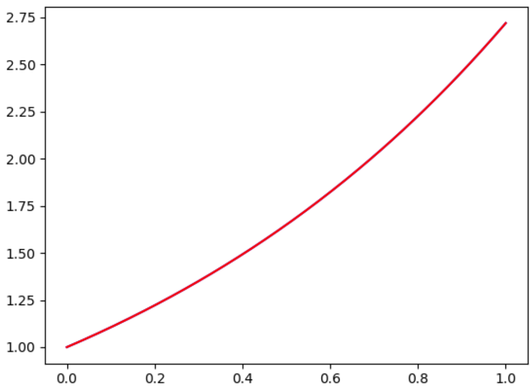
1. e^x, $[0,1]$, $n = 5, 10, 15$,
2. $x^2 \sin x$, $[-1,1]$, $n = 5, 10, 15$.

5.2. Rozwiązanie

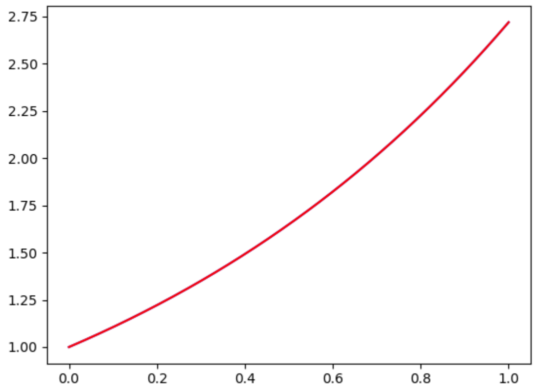
Wywołanie funkcji `rysujNnfx(f, a, b, n)` dla wyżej zadanych danych.

5.3. Wyniki

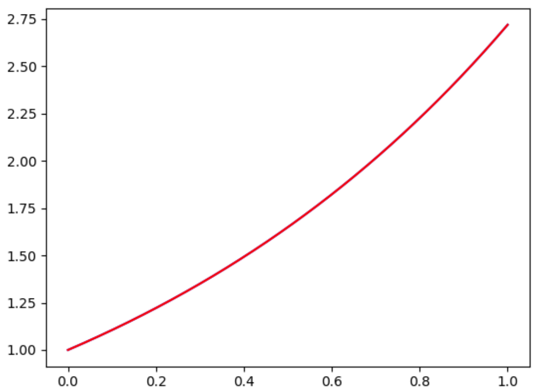
Dla wcześniej zaimplementowanej funkcji otrzymano następujące wyniki (Wykres 1., Wykres2.)



a) $n = 5$



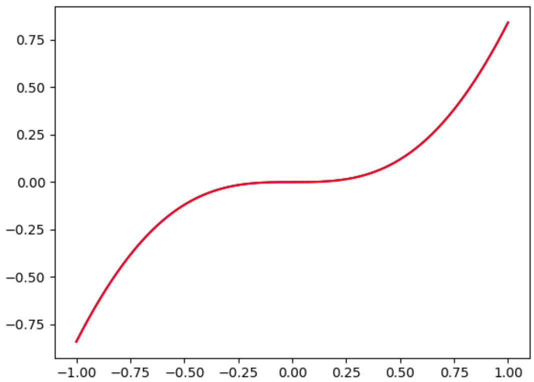
b) $n = 10$



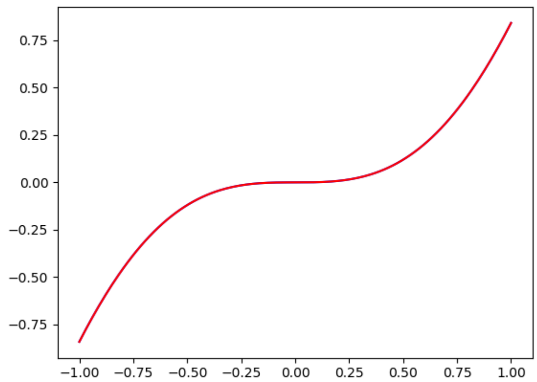
c) $n = 15$

• interpolowana funkcja • wielomian interpolacyjny

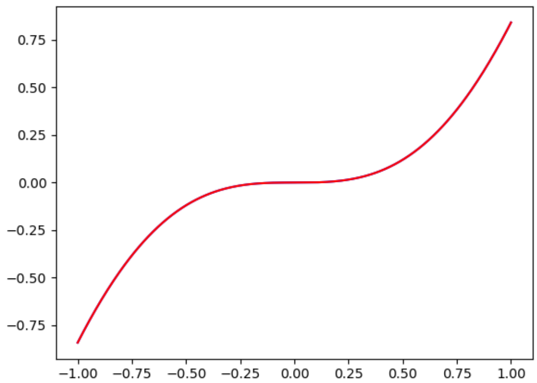
Wykres 1. Wykres funkcji e^x w przedziale [0,1] dla zmiennego parametru n wykonany przy pomocy funkcji `rysujNnfx`



a) $n = 5$



b) $n = 10$



c) $n = 15$

• interpolowana funkcja • wielomian interpolacyjny

Wykres 2. Wykres funkcji $x^2 \sin x$ w przedziale $[-1, 1]$ dla zmiennego parametru n wykonany przy pomocy funkcji `rysujNnfx`

5.4. Wnioski

Uzyskane powyżej wykresy nie wykazują rozbieżności dla zmiennego parametru n w przypadku obu funkcji, co więcej wykres wielomianu interpolacyjnego pokrywa się z wykresem interpolowanej funkcji dla zadanego przedziału. Powodem tego są małe różnice pomiędzy obiema funkcjami, które dodatkowo maleją wraz ze zwiększaniem stopnia wielomianu. Funkcja `rysujNnfx` została zaimplementowana z wykorzystaniem węzłów równoległych przy interpolacji, które pozwoliły uzyskać niezwykle dobrą precyzję obliczeń.

6. Zadanie 6

6.1. Opis problemu

Wykonanie testów funkcji `rysujNnfx(f, a, b, n)` na zadanych danych:

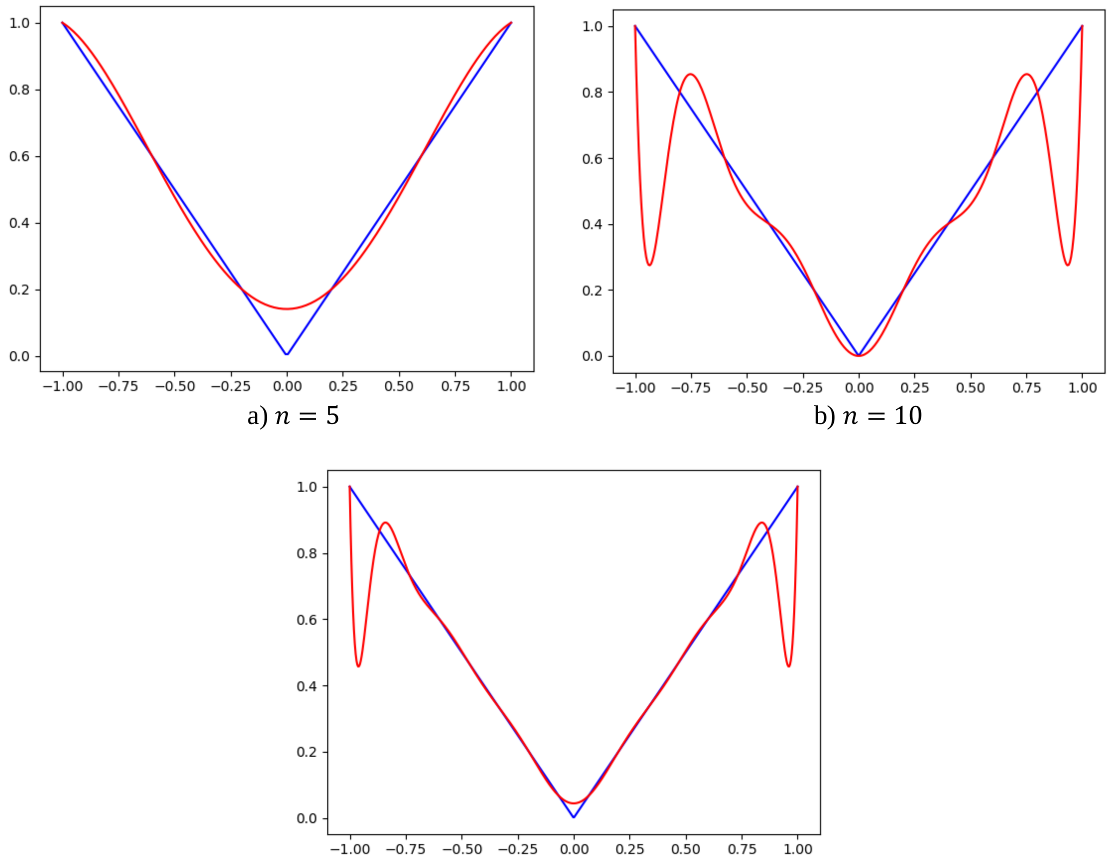
1. $|x|, [-1, 1], n = 5, 10, 15$,
2. $\frac{1}{1+x^2}, [-5, 5], n = 5, 10, 15$ (zjawisko Runge'go).

6.2. Rozwiązanie

Wywołanie funkcji `rysujNnfx(f, a, b, n)` dla wyżej zadanych danych.

6.3. Wyniki

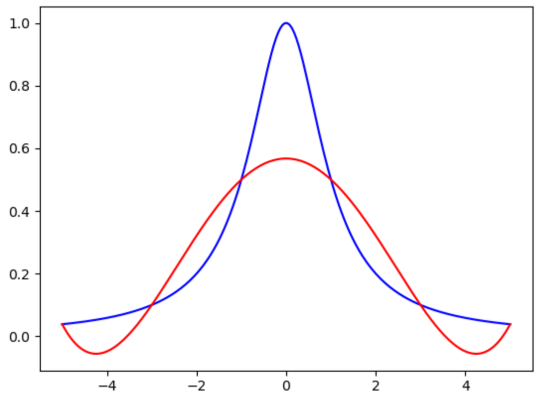
Dla wcześniej zaimplementowanej funkcji otrzymano następujące wyniki (Wykres 3., Wykres4.)



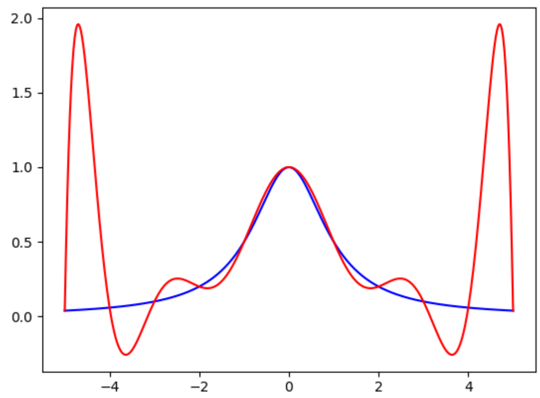
a) $n = 5$ b) $n = 10$ c) $n = 15$

• interpolowana funkcja • wielomian interpolacyjny

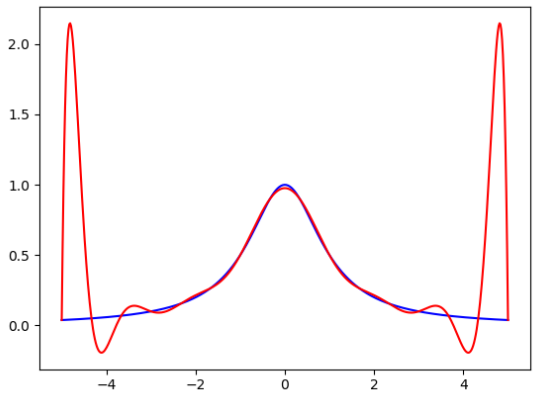
Wykres 3. Wykres funkcji $|x|$ w przedziale $[-1, 1]$ dla zmiennego parametru n wykonany przy pomocy funkcji `rysujNnfx`



a) $n = 5$



b) $n = 10$



c) $n = 15$

• interpolowana funkcja • wielomian interpolacyjny

Wykres 4. Wykres funkcji $\frac{1}{1+x^2}$ w przedziale $[-5, 5]$ dla zmiennego parametru n wykonany przy pomocy funkcji `rysujNnfx`

6.4. Wnioski

W przeciwieństwie do Zadania 6 uzyskane powyżej wykresy wykazują jednoznacznie rozbieżności pomiędzy wykresem wielomianu interpolacyjnego, a wykresem interpolowanej funkcji dla zmiennego parametru n. Odchylenia widoczne na wykresach funkcji $|x|$ (Wykres 3.) są spowodowane faktem, iż funkcja ta nie jest różniczkowalna. Wykres funkcji $\frac{1}{1+x^2}$ wraz ze wzrostem stopnia wielomianu n poprawia się, jednak widoczne są znaczne odchylenia skupione głównie na końcach przedziału. Efekt ten nosi nazwę zjawiska Rungego, które polega na pogorszeniu jakości interpolacji wielomianowej, pomimo zwiększenia liczby węzłów. Powodem tego jest stała odległość pomiędzy kolejnymi węzłami wielomianu interpolacyjnego. Zatem używając zmiennej odległości pomiędzy węzłami można ograniczyć oscylację wykresu na końcach przedziału lub stosując zera wielomianu Czybyszewa jako węzły.