Eliza Zawisza, 244967

SPRAWOZDANIE 4

ZADANIE 1

Opis problemu:

W zadaniu należało zaimplementować funkcję obliczającą ilorazy różnicowe.

Rozwiązanie:

Dane wejściowe:

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

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

Dane wyjściowe:

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

Poniżej pseudokod algorytmu. Kompletny kod źródłowy znajduje się w pliku *interpolation.jl*

```
function ilorazyRoznicowe(x, f)
  for i in 1:n
    fx[i] = f[i]
  end
  for i = 2:n
    for j = n:-1:i
      fx[j] = (fx[j] - fx[j - 1]) / (x[j] - x[j - i + 1])
    end
  end
```

Podsumowanie algorytmu:

W celu obliczenia ilorazu różnicowego stosuje się następujący wzór rekurencyjny:

$$* \quad f[x_0, x_1, \dots, x_k] = \frac{f[x_1, x_2, \dots, x_k] - f[x_0, x_1, \dots, x_{k-1}]}{x_k - x_0}$$

Analizując powyższy wzór łatwo zauważyć, że iloraz różnicowy nie jest zależny od kolejności węzłów.

Ponadto ilorazy różnicowe można wyznaczyć za pomocą tablicy trójkątnej, do której stworzenia potrzebujemy znać węzły oraz wartości funkcji w tych węzłach, czyli ilorazy rzędu zerowego.

$$\begin{array}{cccccc} x_0 & f[x_0] & f[x_0, x_1] & \cdots & f[x_0, x_1, \dots, x_{n-1}] & f[x_0, x_1, \dots, x_n] \\ x_1 & f[x_0] & f[x_1, x_2] & \dots & f[x_1, x_2, \dots, x_n] & \\ \vdots & \vdots & & & & \\ x_{n-1} & f[x_{n-1}] & f[x_{n-1}, x_n] & & & \\ x_n & f[x_n] & & & & \end{array}$$

Warto zauważyć, że celem zadania było obliczenie jedynie pierwszego rzędu powyższej macierzy, dzięki czemu w algorytmie można było zamiast macierzy wykorzystać zwykłą tablicę długości n + 1. Zastosowanie wzoru rekurencyjnego(*) pozwala na obliczenie kolejnych kolumn pierwszego rzędu macierzy podczas przechodzenia pętli w algorytmie (kod powyżej).

ZADANIE 2

Opis problemu:

Celem zadania było napisać funkcję obliczająca wartość wielomianu interpolacyjnego stopnia n w postaci Newtona $N_n(x)$ w punkcie $x = t$ za pomocą uogólnionego algorytmu Hornera, w czasie $O(n)$.

Rozwiązanie:

Dane wejściowe:

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

f_x – wektor długości n + 1, zawierający obliczone ilorazy różnicowe fx[1] = f[x_0], fx[2] = f[x_0, x_1],..., 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

Dane wyjściowe:

nt – wartość wielomianu w punkcie t

Poniżej pseudokod algorytmu. Kompletny kod źródłowy znajduje się w pliku *interpolation.jl*

```
function warNewton(x, fx, t)
  for i = n-1:-1:1
    nt = fx[i] + (t-x[i]) * nt
  end
end
```

Podsumowanie algorytmu:

Wzór interpolacyjny Newtona można przedstawić przy pomocy ilorazów różnicowych jako:

$$\boldsymbol{N_n(x)} = \sum_{k=0}^{n} \boldsymbol{f[x_{0,} x_1, \dots, x_k]} \prod_{j=0}^{k-1} (\boldsymbol{x - x_j}) = f[x_0] + f[x_0, x_1](x - x_0) + \cdots + f[x_0, x_1, \dots, x_n](x - x_0)(x - x_1)(x - x_{n-1})$$

W celu obliczenia wartości wielomianu interpolacyjnego należało skorzystać z uogólnionego algorytmu Hornera, który można wyrazić przy pomocy poniższych wzorów:

$$w_n(x) := f[x_{0,} x_1, \dots, x_n]$$

$$w_k(x) := f[x_{0,} x_1, \dots, x_k] + (x - x_k) w_{k+1}(x) \quad (k = n - 1, \dots, 0)$$

$$N_n(x) := w_0(x)$$

Począwszy od $w_n(t)$, iterując w pętli do $w_0(t)$, po n krokach zostanie otrzymana wartość wielomianu interpolacyjnego w postaci Newtona w zadanym punkcie t.

Jak widać na podstawie pseudokodu algorytm posiada jedną pętlę, którą musi przejść n -1 razy, co spełnia warunek zadania o złożoności $O(n)$.

ZADANIE 3

Opis problemu:

Celem zadania było napisać funkcję obliczająca w czasie $O(n^2)$ współczynniki postaci naturalnej $a_0, \dots, a_n$wielomianu interpolacyjnego w postacie Newtona, tzn. $a_n x^n + \ a_{n-1}x^{n-1} + \cdots + \ a_1 x + \ a_0$, znając jego współczynniki: $c_0 = f[x_0], \ c_1 = f[x_0, x_1], \ c_2 = f[x_0, x_1, x_2], \dots , \ c_n = f[x_0, \dots, x_n]$

Rozwiązanie:

Dane wejściowe:

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

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

Dane wyjściowe:

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

Poniżej pseudokod algorytmu. Kompletny kod źródłowy znajduje się w pliku *interpolation.jl*

```
function naturalna(x, fx)
   for i = n-1:-1:1
      a[i] = fx[i] - a[i + 1] * x[i]
      for j = i+1:n-1
         a[j] = a[j] - a[j + 1] * x[i]
      end
   end
end
```

Podsumowanie algorytmu:

W celu obliczenia współczynników postaci naturalnej wielomianu interpolacyjnego Newtona należy wykorzystać uogólnione wzory Hornera z poprzedniego zadania z drobną modyfikacją. Analizując podane w zadaniu współczynniki wielomianu w postaci Newtona łatwo dostrzec, iż współczynnik a_n przy największej potędze x jest równy c_n, korzystając ponadto z uogólnionego algorytmu Hornera dochodzimy do wniosku, że $a_n = w_n$. Wykorzystując tę zależność podobnie jak w zadaniu poprzednim w pętli wyliczamy kolejne wartości wielomianu interpolacyjnego, pamiętając, by podczas każdej iteracji sprowadzić wielomian składowy do postaci naturalnej (pętla wewnętrzna w powyższym pseudokodzie). W ten sposób w wyniku otrzymany zostaje wektor zawierający współczynniki naturalne interpolacyjnego wielomianu Newtona. Analizując powyższy pseudokod można zauważyć, że wykorzystane zostały dwie pętle, pętla zewnętrzna wykona się n razy, podobnie jak pętla wewnętrzna w najgorszym przypadku zostanie wykonana n razy, zatem zostało spełnione założenie zadania dotyczące złożoności $O(n^2)$.

ZADANIE 4

Opis problemu:

Celem zadania było napisać funkcję, która zinterpoluje zadaną funkcję $f(x)$ w przedziale $[a, b]$ za pomocą wielomianu interpolacyjnego stopnia n w postaci Newtona. Następnie narysuje wielomian interpolacyjny i

interpolowaną funkcję za pomocą jednego z dostępnych w Julii pakietów do interpretacji graficznych funkcji. W interpolacji należało użyć węzłów równoodległych, tzn. $x_k = a + kh, h = \frac{b-a}{n}, k = 0,1, \dots, n$.

Rozwiązanie:

Dane wejściowe:

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

Dane wyjściowe:

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

Poniżej pseudokod algorytmu. Kompletny kod źródłowy znajduje się w pliku *interpolation.jl*

```
function rysujNnfx(f, a, b, n)
   kh = 0.0
   max_nodes = n + 1
   h = (b-a)/n
   for i = 1: max_nodes
      x[i] = a + kh
      y[i] = f(x[i])
      kh += h
   end
   fx = ilorazyRoznicowe(x, y);
   kh = 0.0
   max_nodes *= density
   h = (b - a)/(max_nodes-1)
   for i = 1: max_nodes
      plot_x[i] = a + kh
      plot_inp[i] = warNewton(x, fx, plot_x[i])
      plot_y[i] = f(plot_x[i])
      kh += h
   end    rysowanie_wykresu()
end
```

Podsumowanie algorytmu:

W celu narysowania wielomianu interpolacyjnego oraz interpolowanej funkcji na początku należy wyznaczyć węzły interpolacji $(x_1, \dots, x_{n+1})$, które są od siebie równoodległe o $\frac{b-a}{n}$ na przedziale $[a, b]$ oraz wartości funkcji interpolowanej w stworzonych wcześniej węzłach: $(f(x_1), \dots, f(x_{n+1}))$. Następnie korzystając z funkcji zaimplementowanej w zadaniu 1 (ilorazyRozniocowe) obliczane są wartości ilorazu różnicowego dla stworzonych wcześniej węzłów. W celu uzyskania większej dokładności wykresu zostaje wprowadzona zmienna *density* (jej wartość zostaje ustawiona na 20), czyli zarówno wielomian interpolacyjny, jak i funkcja interpolowana będą próbkowane w 20(n + 1) równoodległych punktach, dla których wartość wielomianu zostaje obliczona za pomocą funkcji z zadania 2 (warNewton). Na samym końcu zostaje wywołana funkcja rysująca wykres wielomianu inrterpolacyjnego oraz funkcji interpolowanej za pomocą pakietu PyPlot.

ZADANIE 5

Opis problemu:

Celem zadania było przetestowanie działania funkcji rysujNnfx(f, a, b, n) na następujących przykładach:

a) $e^x, [0,1], n = 5, 10, 15$
b) $x^2 sinx, [-1,1], n = 5, 10, 15$

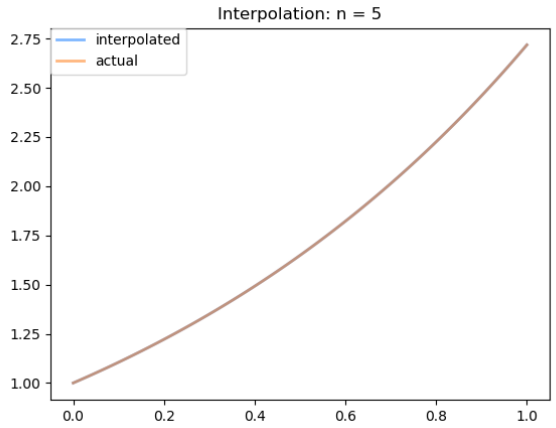
Rozwiązanie:

W ceku rozwiązania zadania została wywołana funkcja rysujNnfx(f, a, b, n) z zadania 4 z odpowiednimi danymi.

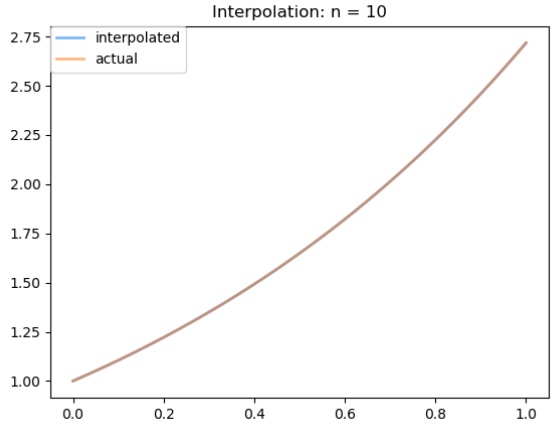
Wyniki oraz ich interpretacja:

Wykresy funkcji e^x i jej wielomianu interpolacyjnego na przedziale $[0,1]$:

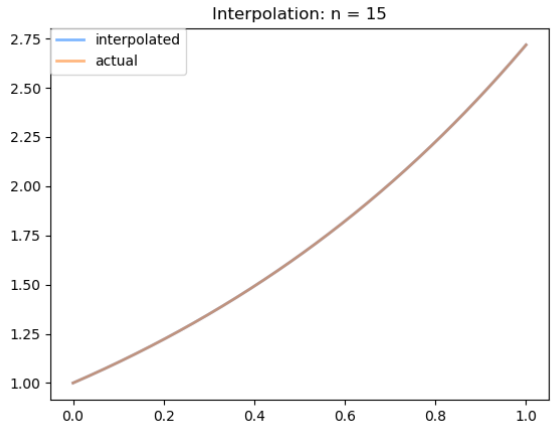
- $n = 5$



- $n = 10$



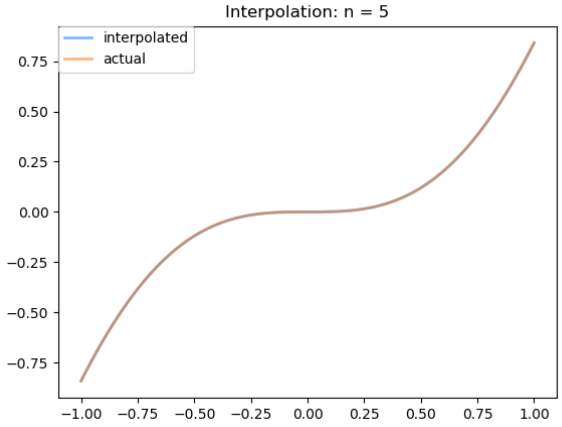
- $n = 15$



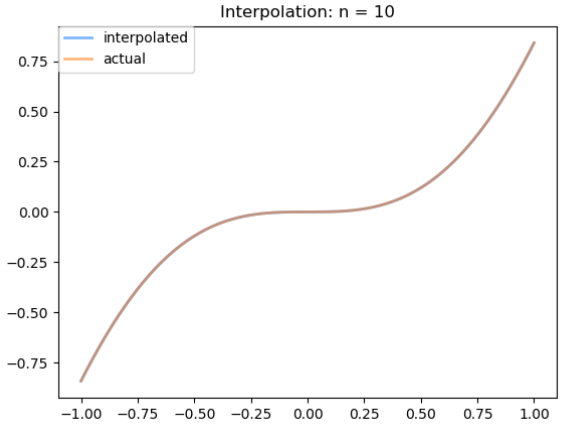
Jak widać na powyższych wykresach wartości funkcji e^x oraz jej wielomianu interpolacyjnego nachodzą na siebie, co oznacza, że wielomian bardzo dokładnie przybliżył interpolowaną funkcję.

Wykresy funkcji $x^2 sinx$ i jej wielomianu interpolacyjnego na przedziale $[-1, 1]$:

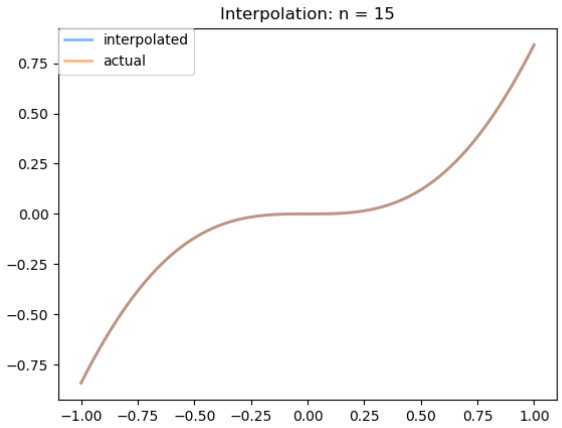
- $n = 5$



- $n = 10$



- $n = 15$



Jak widać na powyższych wykresach wartości funkcji $x^2 sinx$ oraz jej wielomianu interpolacyjnego nachodzą na siebie, co oznacza, że wielomian bardzo dokładnie przybliżył interpolowaną funkcję.

Wnioski:

Na podstawie analizy powyższych wykresów można wywnioskować, że wartość funkcji i jej wielomianu interpolacyjnego są do siebie bardzo zbliżone. Powodem, dla którego tak się dzieje jest fakt, iż w celu obliczenia wartości użyte zostały węzły równoodległe od siebie.

ZADANIE 6

Opis problemu:

Celem zadania było przetestowanie działania funkcji rysujNnfx(f, a, b, n) na następujących przykładach (zjawisko rozbieżności)

a) $|x|, [-1, 1], n = 5, 10, 15$

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

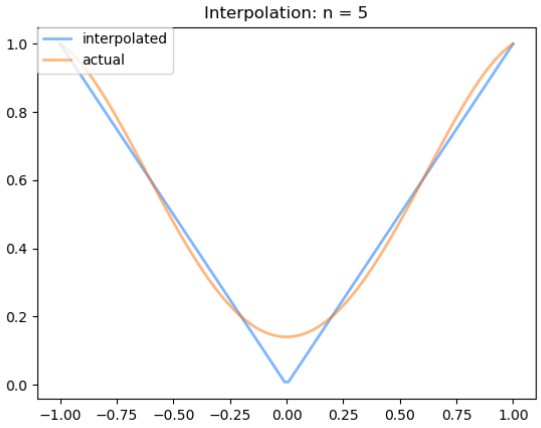
Rozwiązanie:

W ceku rozwiązania zadania została wywołana funkcja rysujNnfx(f, a, b, n) z zadania 4 z odpowiednimi danymi.

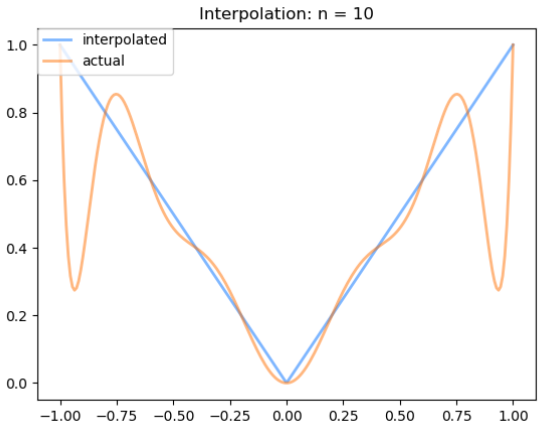
Wyniki oraz ich interpretacja:

Wykresy funkcji $|x|$ i jej wielomianu interpolacyjnego na przedziale $[-1, 1]$:

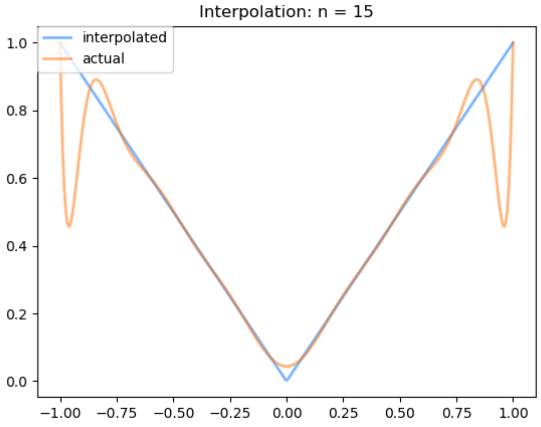
- $n = 5$



- $n = 10$



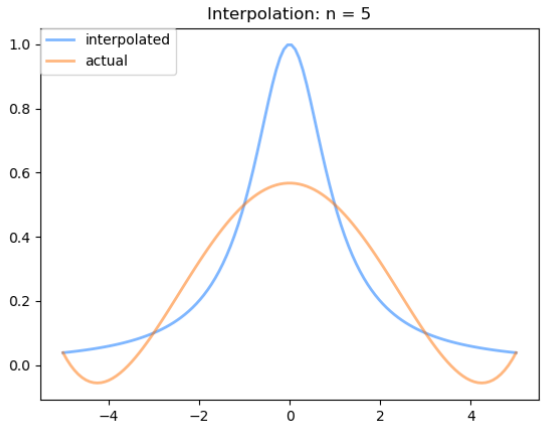
- $n = 15$



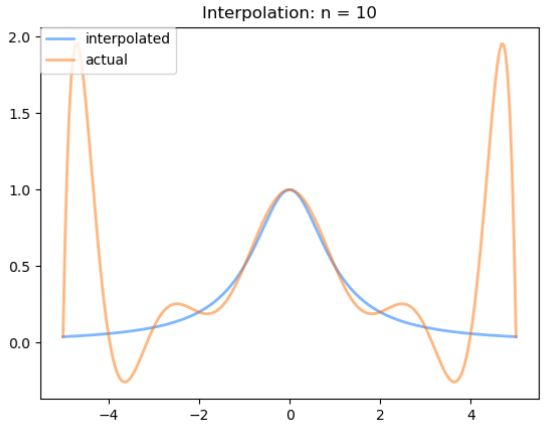
Jak widać na powyższych wykresach wartości funkcji $|x|$ oraz jej wielomianu interpolacyjnego znacznie różnią się od siebie.

Wykresy funkcji $\frac{1}{1+x^2}$ i jej wielomianu interpolacyjnego na przedziale $[-5, 5]$:

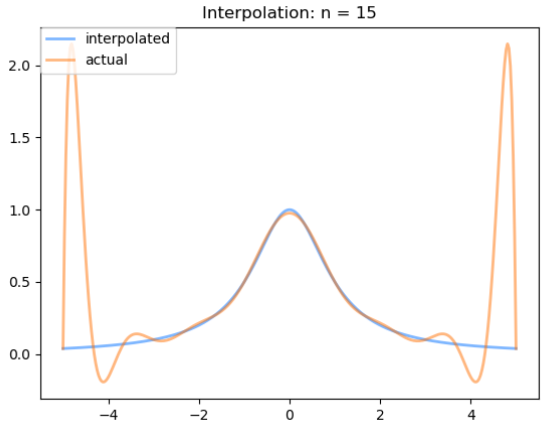
- $n = 5$



- $n = 10$



- $n = 15$



Jak widać na powyższych wykresach wartości funkcji $\frac{1}{1+x^2}$ oraz jej wielomianu interpolacyjnego znacznie różnią się od siebie.

Wnioski:

Przeciwnie do zadania 5 tutaj wraz ze wzrostem stopnia wielomianu interpolacyjnego mamy do czynienia ze zwiększeniem odchyleń wartości wielomianu od rzeczywistych wartości funkcji interpolowanych.

W przypadku funkcji $f(x) = |x|$ zjawisko rozbieżności funkcji i jej wielomianu interpolacyjnego wynika z faktu, że funkcja ta nie jest różniczkowalna. Nie istnieje pochodna w punkcie $x_0 = 0$.

Natomiast w przypadku funkcji $f(x) = \frac{1}{1+x^2}$ mamy do czynienia ze zjawiskiem Runge`go typowym dla wielomianów dużego stopnia i równoodległych węzłach. Interpolacyjny wielomian funkcji przy równoodległych węzłach sprawia, że na krańcach przedziału mamy duże odchylenia od interpolowanej funkcji. W celu zapobiegania temu zjawisku stosuje się wielomian oparty na węzłach Czybyszewa (miejsca zerowe wielomianu Czybyszewa), węzły są rozłożone dużo gęściej na przedziale, dzięki czemu możemy interpolować funkcję z większą dokładnością.