Obliczenia Naukowe

Lista 4

Jakub Gogola
236412

5 grudnia 2018

1 Zadanie I

1.1 Problem

W zadaniu 1. należało, używając języka `Julia`, zaimplementować funkcję obliczającą ilorazy różnicowe dla zadanej funkcji f. Problem należało rozwiązać bez używania tablicy dwuwymiarowej do obliczania tychże.

1.2 Algorytm

Do obliczania ilorazów różnicowych można skorzystać z następującego uogólnionego wzoru:

$$f[x_i, x_{i+1}, \ldots, x_{i+j}] = \frac{f[x_{i+1}, x_{i+2}, \ldots, x_{i+j}] - f[x_i, x_{i+1}, \ldots, x_{i+j-1}]}{x_{i+j} - x_i}$$

Zakładamy warunki początkowe: $f[x_i] = f(x_i)$, $f[x_i, x_j] = \frac{f(x_j) - f(x_i)}{x_j - x_i}$ oraz $i, j \in 0, \ldots, k$, gdzie k jest rzędem ilorazu różnicowego. Ilorazy wyliczone za pomocą powyższego wzoru możemy reprezentować w tablicy trójkątnej przyjmując następujące oznaczenie $c_k = f[x_i, x_{i+1}, \ldots, x_{i+j}]$:

$$\begin{array}{cc|cccccc} x_0 & c_{00} & c_{01} & c_{02} & \ldots & c_{0,n-1} & c_{0,n} \\ x_1 & c_{10} & c_{11} & c_{12} & \ldots & c_{1,n-1} & \\ \ldots & \ldots & \ldots & \ldots & & & \\ x_{n-1} & c_{n-1,0} & c_{n-1,1} & & & & \\ x_n & c_{n,0} & & & & & \end{array}$$

Można zatem podać wzór na obliczenie czynników c_{ij} o następującej postaci: $c_{ij} = \frac{c_{i+1,j-1} - c_{i,j-1}}{x_{i+j} - x_i}$. Analizując algorytm obliczania kolejnych ilorazów różnicowych można zauważyć, że do ich wyliczenia nie potrzeba wcale tablic dwuwymiarowej, a wystarczy wykorzystać tablicę jednowymiarową z jednym tylko wskaźnikiem (oznaczmy tę tablicę jako f_x). Początkowa wartość zmiennej f_{x_i} to $c_{i0} = f(x_i)$ (z drugiej kolumny prezentowanej powyżej tablicy trójkątnej). Następne wartości to odpowiednio: $c_{i-1,1}, \ldots, c_{1,i-1}, c_{0i}$. W każdej iteracji przedstawionego poniżej algorytmu otrzymywane są w ten sposób kolejne kolumny tablicy trójkątnej niezbędne do wyliczenia kolejnych ilorazów różnicowych. Kolumny są aktualizowane "od dołu" (linie 6-8), co zapewnia nam, że tablica zawiera w każdej kolejnej iteracji ilorazy potrzebne do wyliczenia nowych wartości.

```
1:  function ILORAZY-RÓŻNICOWE(x, f)
2:      len ← LENGTH(f)
3:      for i ← 1 to len do
4:          f_x[i] ← f[i]   ▷ f_x jest tablicą długości len, są w niej przechowywane ilorazy różnicowe
5:      end for
6:      for i ← 2 downto len do
7:          f_x[j] ← (f_x[j] − f_x[j−1]) / (x[j] − x[j−i+1])
8:      end for
9:      return f_x
10: end function
```

Opis parametrów:
`x` - wektor długości $n+1$ zawierający węzły $x_0, \ldots, x_n$,
`f` - wektor długości $n+1$ zawierający wartości interpolowanej funkcji w podanych węzłach.

Dane wynikowe:
`fx` - wektor długości $n+1$ zawierający obliczone ilorazy różnicowe

Wszystkie obliczenia są wykonywane przy użyciu arytmetyki `Float64`.

2 Zadanie II

2.1 Problem

W zadaniu 2., używając języka `Julia`, należało napisać funkcję obliczającą wartość wielomianu interpolacyjnego Newtona $N_n(x)$ w punkcie $x = t$ za pomocą uogólnionego algorytmu Hornera, który działa w czacie $\mathcal{O}(n)$.

2.2 Algorytm

Wzór interpolacyjny Newtona można przedstawić używając, ilorazów różnicowych, jako następującą sumę:

$$N_n(x) = \sum_{k=0}^{n} c_k q_k(x) = \sum_{k=0}^{n} f[x_0, x_1, \ldots, x_k] \prod_{j=0}^{k-1} (x - x_j)$$

Przedstawiona powyżej metoda jest dobrze uwarunkowana pod względem numerycznym. Powyższą zależność można również łatwo zapisać używając uogólnionych wzorów Hornera:

$w_n(x) = f[x_0, x_1, \ldots, x_n]$
$w_k(x) = f[x_0, x_1, \ldots, x_k] + (x - x_k) w_{k+1}(x)$, gdzie $(k = n-1, \ldots, 0)$
$N_n(x) = w_0(x)$.

Poniżej przedstawiony jest pseudokod algorytmu. Algorytm jest dokładną implementacją przedstawionych wzorów i w pętli (linie 4-6) wykonuje kolejne kroki i na końcu zwraca wyliczoną wartość n_t.

```
1: function WAR-NEWTON(x, f_x, t)
2:     len ← LENGTH(x)
3:     n_t = f_x
4:     for i ← len − 1 downto 1 do
5:         n_t ← f_x[i] + (t − x[i]) · n_t
6:     end for
7:     return n_t
8: end function
```

Opis parametrów:
`x` - wektor długości $n+1$ zawierający węzły $x_0, \ldots, x_n$,
`fx` - wektor długości $n+1$ zawierający obliczone ilorazy różnicowe
`t` - punkt, w którym należy obliczyć wartość wielomianu

Dane wynikowe:
`nt` - wartość wielomianu w punkcie t.

Wszystkie obliczenia są wykonywane przy użyciu arytmetyki `Float64`.

3 Zadanie III

3.1 Problem

W zadaniu 3. należało, używając języka `Julia`, znając współczynniki wielomianu interpolacyjnego Newtona $c_0 = f[x_0]$, $c_1 = f[x_0, x_1]$, $c_2 = f[x_0, x_1, x_2]$, $\ldots$, $c_n = f[x_0, \ldots, x_n]$ (ilorazy różnicowe) oraz węzły $x_0, x_1, \ldots, x_n$ napisać funkcję obliczającą, w czasie $\mathcal{O}(n^2)$, współczynniki jego postaci naturalnej $a_0, \ldots, a_n$, to znaczy: $a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$.

3.2 Algorytm

W celu znalezienia współczynników dla naturalnej postaci wielomianu interpolacyjnego Newtona wykorzystano uogólnione wzory Hornera przedstawione w poprzednim zadaniu. Na początku korzystamy faktu, że $a_n = c_n$ (linia 3). Następnie, bardzo podobnie jak w poprzednim zadaniu, wyliczamy kolejne wartości częściowe dla wielomianu interpolacyjnego (linia 5), ale z tą różnicą, że podczas każdej iteracji każdy "składowy" wielomian doprowadzamy do postaci naturalnej (linie 6-8) i w ten sposób uzyskujemy kolejne współczynniki dla postaci naturalnej wielomianu interpolacyjnego.

Na podstawie obserwacji pętli użytych w algorytmie oraz tego, ile razy i w jakiej kolejności każda z nich się wykona (pętla zewnętrzna wykona się n razy, podobnie pętla wewnętrzna, w najgorszym przypadku, wykona n iteracji). Daje to złożoność przedstawionego algorytmu rzędu $\mathcal{O}(n^2)$.

```
1:  function NATURALNA(x, f_x)
2:      len ← LENGTH(x)
3:      a[len] ← f_x[len]                ▷ a jest tablicą długości len z wyliczanymi współczynnikami
4:      for i ← len − 1 downto 1 do
5:          a[i] = f_x[i] − a[i + 1] · x[i]
6:          for j ← i + 1 to len − 1 do
7:              a[j] ← a[j] − a[j + 1] · x[i]
8:          end for
9:      end for
10:     return a
11: end function
```

Opis parametrów:
`x` - wektor długości $n + 1$ zawierający węzły $x_0, \ldots, x_n$,
`fx` - wektor długości $n + 1$ zawierający obliczone ilorazy różnicowe

Dane wynikowe:
`a` - wektor długości $n + 1$ zawierający obliczone współczynniki postaci naturalnej.

Wszystkie obliczenia są wykonywane przy użyciu arytmetyki `Float64`.

4 Zadanie IV

4.1 Problem

W zadaniu 4., używając języka `Julia`, należało zaimplementować metodę, która zinterpoluje funkcję $f(x)$ na zadanym przedziale $[a, b]$ za pomocą wielomianu interpolacyjnego Newtona. Wynikiem funkcji jest wykres, na którym są zobrazowane funkcja $f(x)$ oraz interpolujący ją wielomian. W iterpolacji należało użyć węzłów równoodległych, czyli $x_k = a + kh$, $h = (b - a)/n$, $k = 0, 1, \ldots, n$.

4.2 Algorytm

Algorytm użyty w zadaniu 4. jest w swoich założeniach dosyć prosty i oczywisty, bowiem, na podstawie otrzymanych danych i zaimplementowanych we wcześniejszych zadaniach funkcji, wyznacza on ilorazy różnicowe i wartość wielomianu interpolacyjnego. W linia 3-4 wyznaczamy warunki początkowe (zmienne h i kh) dla późniejszego obliczania kolejnych, równoodległych od siebie węzłów wielomianu. Definiujemy również zmienną *maxnodes* mówiącą o maksymalnej liczbie węzłów użytych do wyznaczenia wielomianu. Następnie (pętla w liniach 6-10) wyznaczamy kolejne węzły i liczymy w nich wartości dla zadanej funkcji $f(x)$ (węzły są wyznaczane z krokiem h). W następnej kolejności liczymy ilorazy różnicowe na podstawie wyliczonych przed momentem wartości (linia 11). Dla uzyskania większej dokładności wykresu zwiększone zostaje jego próbkowanie (wartości wielomianu zostaną wyznaczone w $maxnodes \cdot mult$ węzłach - linia 13; mnożnikiem ($mult$) jest w tym przypadku wartość 20, czyli będziemy próbkowali w $20 \cdot (n + 1)$ węzłach. Następnie (linie 15-19) wyliczane są wartości wielomianu interpolacyjnego dla zadanych węzłów za pomocą funkcji `warNewton` i na końcu rysowany jest odpowiedni wykres (w tym zadaniu użyto pakietu `PyPlot` współpracującego z językiem `Julia`.

```
1: function RYSUJ-NNFX(f, a, b, n)
2:     maxnodes ← n + 1
3:     h ← (b−a)/n
4:     kh ← 0
5:     mult ← 20
6:     for I ← 1 to maxnodes do
7:         x[i] ← a + kh
8:         y[i] ← f(x[i])
9:         kh ← kh + h
10:    end for
11:    f_x ← ILORAZY-RÓŻNICOWE(x, y)
12:    kh ← 0
13:    maxnodes ← maxnodes · mult
14:    h ← (b−a)/(maxnodes−1)
15:    for i ← 1 to maxnodes do
16:        plotargs[i] ← a + kh
17:        plotip[i] ← WAR-NEWTON(x, f_x, plotargs[i])
18:        plotval[i] ← f(plotargs[i])
19:        kh ← kh + h
20:    end for
21:    DRAW-PLOT(plotargs, plotval, plotip)        ▷ wywołanie funkcji rysującej wykres
22: end function
```

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

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

Wszystkie obliczenia są wykonywane przy użyciu arytmetyki `Float64`.

5 Zadanie V

5.1 Problem

W zadaniu 5. należało przetestować funkcję `rysujNnfx(f, a, b, n)` zaimplementowaną w zadaniu 4. na następujących przykładach:

(a) e^x, $[0, 1]$, $n = 5, 10, 15$,

(b) $x^2 \sin x$, $[-1, 1]$, $n = 5, 10, 15$.

5.2 Algorytm

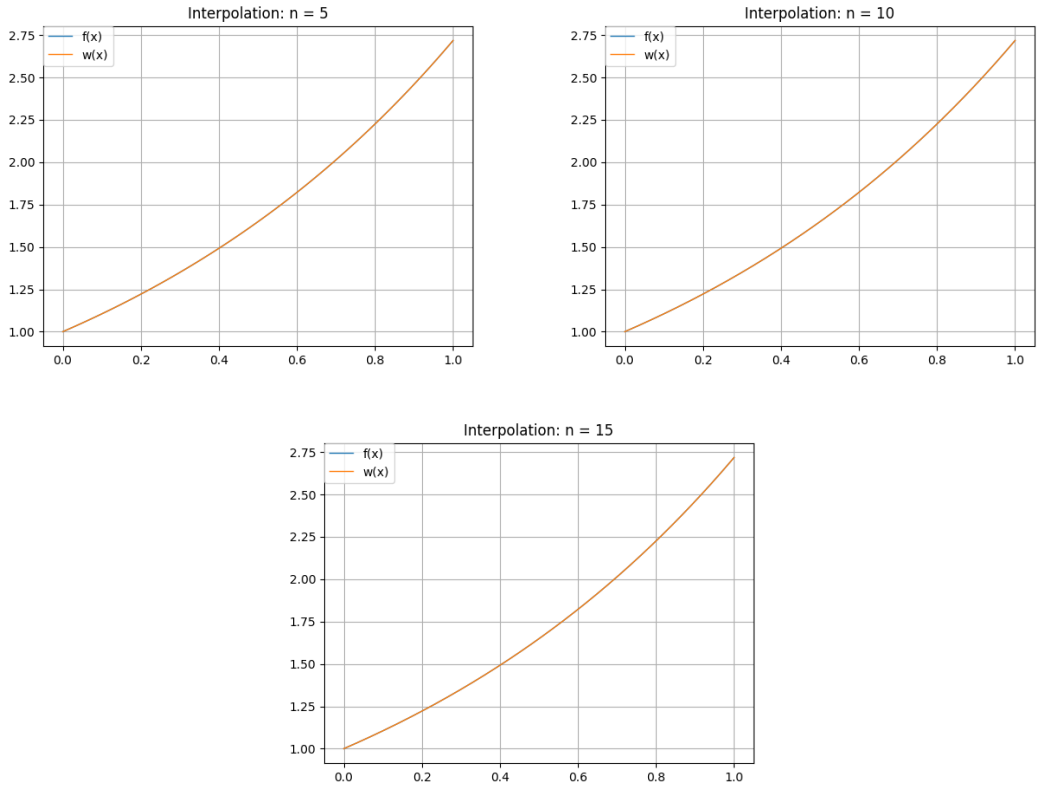
W zadaniu użyto algorytmu z zadania 4.

5.3 Wyniki

Wyniku działania programu otrzymano wykresy przedstawiające interpolowaną funkcję oraz wielomian interpolacyjny. Dla podpunktu (a) otrzymano wykresy przedstawione na rysunku 1., a dla podpunktu (b) wyniki przedstawione są na rysunku 2.

5.4 Obserwacje

Wykresy wielomianów interpolacyjnego i interpolowanych funkcji praktycznie się ze sobą pokrywają.



Rysunek 1: Wykresy dla funkcji e^x

5.5 Wnioski

Na podstawie obserwacji wykresów możemy wnioskować, że wartości dla zadanych funkcji i ich wielomianów interpolacyjnych są do siebie bardzo zbliżone. Spowodowane jest to wyliczaniem wartości dla równo odległych od siebie węzłów co daje w konsekwencji bardzo dobre przybliżenie rzeczywistych danych.

6 Zadanie VI

6.1 Problem

W zadaniu 6. należało zbadać zjawisko rozbieżności dla zadanych danych stosując funkcję `rysujNnfx(f, a, b, n)` z zadania 4.:

(a) $|x|$, $[0, 1]$, $n = 5, 10, 15$,

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

6.2 Algorytm

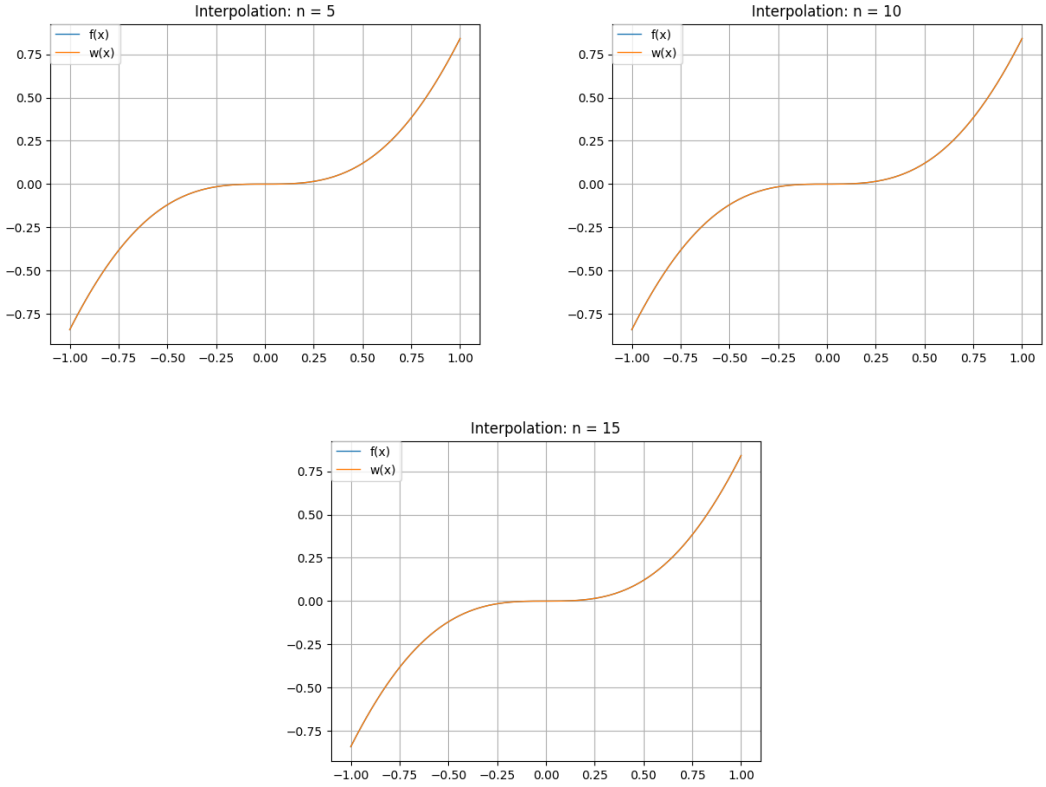
W zadaniu użyto algorytmu z zadania 4.

6.3 Wyniki

Wyniku działania programu otrzymano wykresy przedstawiające interpolowaną funkcję oraz wielomian interpolacyjny. Dla podpunktu (a) otrzymano wykresy przedstawione na rysunku 3., a dla podpunktu (b) wyniki przedstawione są na rysunku 4.

6.4 Obserwacje

Dla obu funkcji badanych w tym zadaniu widać znaczne rozbieżności pomiędzy wartościami funkcji a wartościami jej wielomianu interpolacyjnego. Dla funkcji z podpunktu (b) obserwujemy, że dla



Rysunek 2: Wykresy dla funkcji $x^2 \sin x$

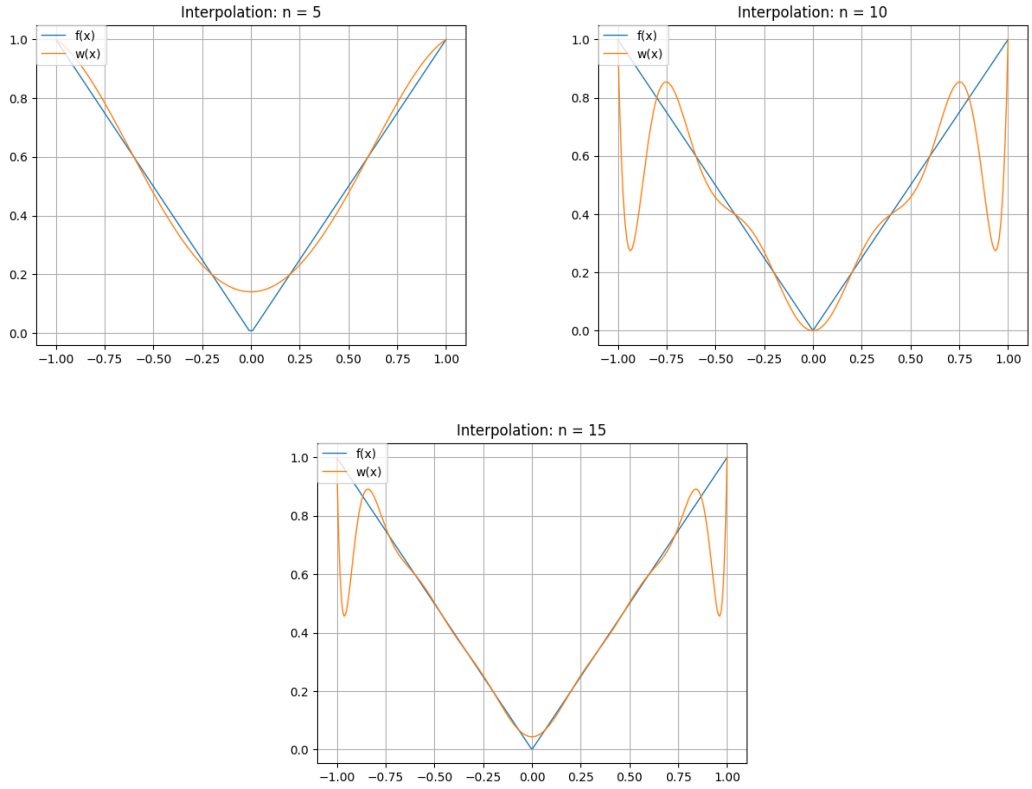
małej liczby węzłów (wartość n) przybliżenie jest w miarę dokładne, ale wraz ze wzrostem wartości n przybliżenie to znacznie się pogarsza, co szczególnie łatwo zaobserwować na końcach zadanego przedziału.

6.5 Wnioski

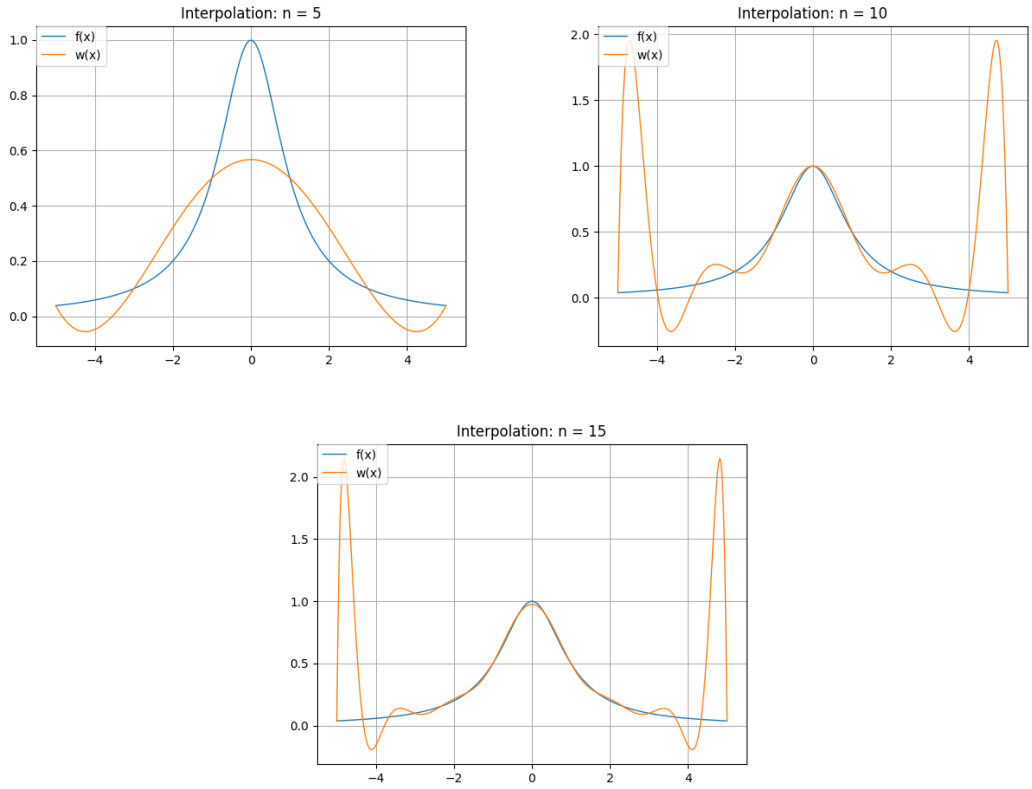
W przypadku funkcji $f(x) = |x|$ zjawisko rozbieżności pomiędzy samą funkcją f a jej wielomianem interpolacyjnym wynika z faktu, że funkcja $|x|$ nie jest różniczkowalna.

Dla funkcji $f(x) = \frac{1}{1+x^2}$ mamy do czynienia ze zjawiskiem (zwanym również efektem) Runge'go (obserwacje dla tego zjawiska zapisano w punkcie **6.4**). Jest to typowe zjawisko dla przypadku, gdy węzły dla wielomianu interpolacyjnego są równo od siebie odległe, a sam wielomian interpolacyjny jest wysokiego stopnia, co powoduje znaczne odchylenia wartości wielomianu od wartości funkcji f na końcach przedziału $[a, b]$. W celu zapobieżenia temu efektowi stosuje się interpolację, w której węzły są gęściej rozmieszczone na końcach przedziału, na którym funkcja jest interpolowana. Można zastosować wtedy wielomiany Czebyszewa n-tego stopnia, których miejsca zerowe stają się węzłami. Wiemy bowiem, że dla takich wielomianów ich miejsca zerowe są gęściej rozmieszczone na zadanym przedziale.

Warto również wspomnieć, że zjawisko Runge'go występuje również, gdy interpolowana funkcja jest nieciągła albo odbiega znacząco od funkcji gładkiej.



Rysunek 3: Wykresy dla funkcji $|x|$



Rysunek 4: Wykresy dla funkcji $\frac{1}{1+x^2}$