Obliczenia Naukowe – Sprawozdanie Laboratoria 4.

Jakub Kogut

7 grudnia 2025

1 Wstęp

Na liście pojawia się problem interpolacji funkcji; polega ona na znalezieniu funkcji wielomianowej, która przechodzi przez zadane punkty i w "dobry sposób" przybliża funkcję oryginalną.
Dokładniej dla zadanych $(x_i, f(x_i))$ *węzłów interpolacji*, gdzie $i = 0, 1, \ldots, n$ chcemy znaleźć wielomian $p_n(x)$ stopnia co najwyżej n, taki że:

$$p_n(x_i) = f(x_i), \quad i = 0, 1, \ldots, n. \tag{1}$$

Również celem jest zapisanie go w postacji Newtona:

$$p_n(x) = \sum_{i=0}^{n} f[x_0, x_1, \ldots, x_i] \prod_{j=0}^{i-1} (x - x_j), \tag{2}$$

gdzie $f[x_0, x_1, \ldots, x_i]$ są *ilorazami różnicowymi* funkcji f w punktach $x_0, x_1, \ldots, x_i$.

1.1 Motywacja

W tej sekcji wyjaśnie dlaczego stosujemy postać Newtona do zapisu wielomianu interpolacyjnego.

1.1.1 Dlaczego nie postać naturalna?

Postać naturalna wielomianu to:

$$p_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n. \tag{3}$$

Chociaż jest to najprostsza postać, to ma kilka wad:

- Trudność w obliczaniu współczynników a_i z danych punktów (rozwiązanie równania z macierzą Vandermonde'a, która jest źle uwarunkowana).
- Niestabilność numeryczna dla dużych n

1.1.2 Dlaczego postać Newtona?

Postać Newtona jest zbudowana na innej bazie:

$$\begin{aligned} B &= \left\{ \prod_{j=0}^{k-1} (x - x_j) : k = 0, 1, \ldots, n \right\} \\ &= \{1, (x - x_0), (x - x_0)(x - x_1), \ldots, (x - x_0)(x - x_1) \cdots (x - x_{n-1})\}. \end{aligned} \tag{4}$$

Wtedy $p_n(x)$ można zapisać jako:

$$p_n(x) = \sum_{i=0}^{n} c_i \prod_{j=0}^{i-1} (x - x_j), \tag{5}$$

gdzie współczynniki c_i są równe ilorazom różnicowym:

$$c_i = f[x_0, x_1, \ldots, x_i]. \tag{6}$$

Postać Newtona ma kilka zalet:

- Łatwość obliczania współczynników za pomocą ilorazu różnicowego (zadanie 1.)
- Łatwe obliczanie wartości wielomianu w danym punkcie za pomocą schematu Hornera (zadanie 2.)
- Stabilność numeryczna (mniejsze błędy przy dużych n)

2 Implementacja

Implementacja algorytmów znajduje się w pliku `interpolacja.jl`, natomiast implentacja testów w `zadanie56.jl`. Poniżej znajduje się opis zadań.

2.1 Zadanie 1.

W zadaniu 1. należy zaimplementować funkcję obliczającą ilorazy różnicowe dla zadanych węzłów interpolacji i wartości funkcji w tych punktach. Kozystając ze wzoru:

$$f[x_0, x_1, \ldots, x_k] = \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0} \tag{7}$$

Poniżej znajduje się pseudokod algorytmu:

Algorithm 1 ilorazyRoznicowe

```
1:  function ilorazyRoznicowe(x, f)
2:      n ← length(x)
3:      F ← array of size n
4:      for i ← 0 to n − 1 do
5:          F[i] ← f[i]
6:      end for
7:      for j ← 1 to n − 1 do
8:          for i ← n − 1 downto j do
9:              F[i] ← (F[i] − F[i − 1]) / (x[i] − x[i − j])
10:         end for
11:     end for
12:     return F
13: end function
```

2.2 Zadanie 2.

W zadaniu 2. należy zaimplementować funkcję obliczającą wartość wielomianu interpolacyjnego w danym punkcie x korzystając z ilorazów różnicowych i schematu Hornera. Poniżej znajduje się pseudokod algorytmu:

Algorithm 2 warNewton

```
1: function WARNEWTON(x, F, x_0)
2:     n ← length(F)
3:     p ← F[n − 1]
4:     for i ← n − 2 downto 0 do
5:         p ← F[i] + (x − x_0) · p
6:     end for
7:     return p
8: end function
```

2.3 Zadanie 3.

W zadaniu 3. należy zaimplementować funkcję zwracającą współczynniki wielomianu interpolacyjnego w postaci naturalnej.

$$p_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n.$$

W tym celu należy skorzystać z ilorazów różnicowych i wzoru:

$$a_k = f[x_0, x_1, \ldots, x_k] + \sum_{j=0}^{k-1} a_j \prod_{i=0}^{j-1} (x_i - x_j).$$

Poniżej znajduje się pseudokod algorytmu:

Algorithm 3 naturalna

```
1: function NATURALNA(x, fx)
2:     n ← length(x)
3:     for i ← 0 to n − 1 do                          ▷ init tablicy
4:         fx[i] ← f[x_0, x_1, ..., x_i]
5:     end for
6:     a[n] ← fx[n]
7:     for i ← n − 1 downto 1 do
8:         a[i] ← fx[i]
9:         for j ← i + 1 to n − 1 do
10:            a[j] ← a[j] − a[j + 1] · x[i]
11:        end for
12:    end for
13:    return a
14: end function
```

2.4 Zadanie 4.

W zadaniu 4. należy zaimplementować funkcję wykożystującą wszystkie poprzednie do narysowania wykresu funkcji interpolowanej i oryginalnej. W tym celu należy skorzystać z biblioteki `Plots.jl` i funkcji `plot`.
Funkcja ma również mieć możliwość wyboru węzłów interpolacji: równoodległych lub węzłów Czebyszewa:

- Węzły równoodległe:

$$x_i = a + i \cdot \frac{b-a}{n}, \quad i = 0, 1, \ldots, n$$

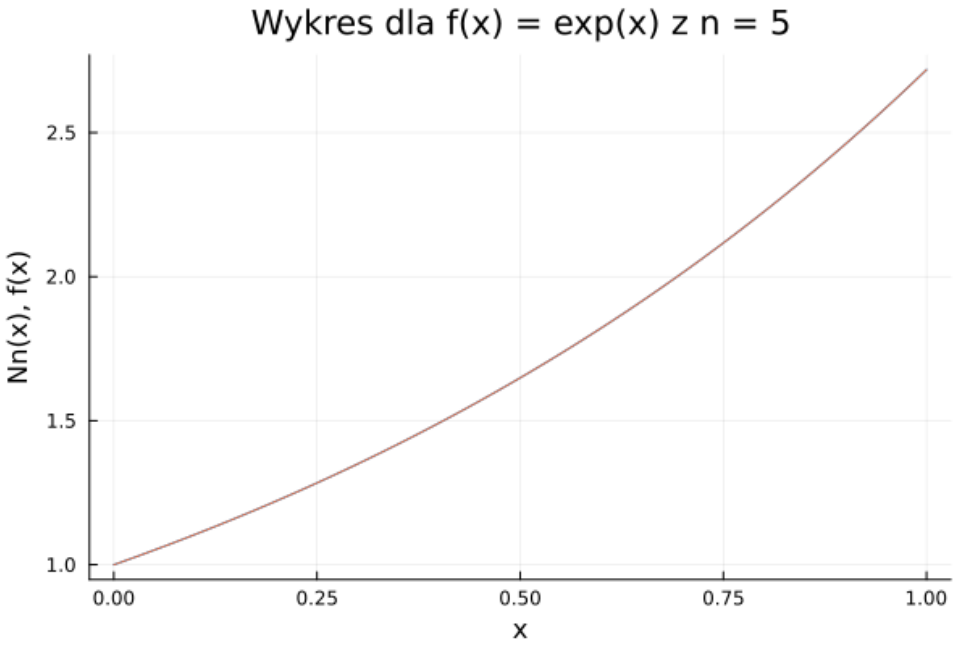
- Węzły Czebyszewa:

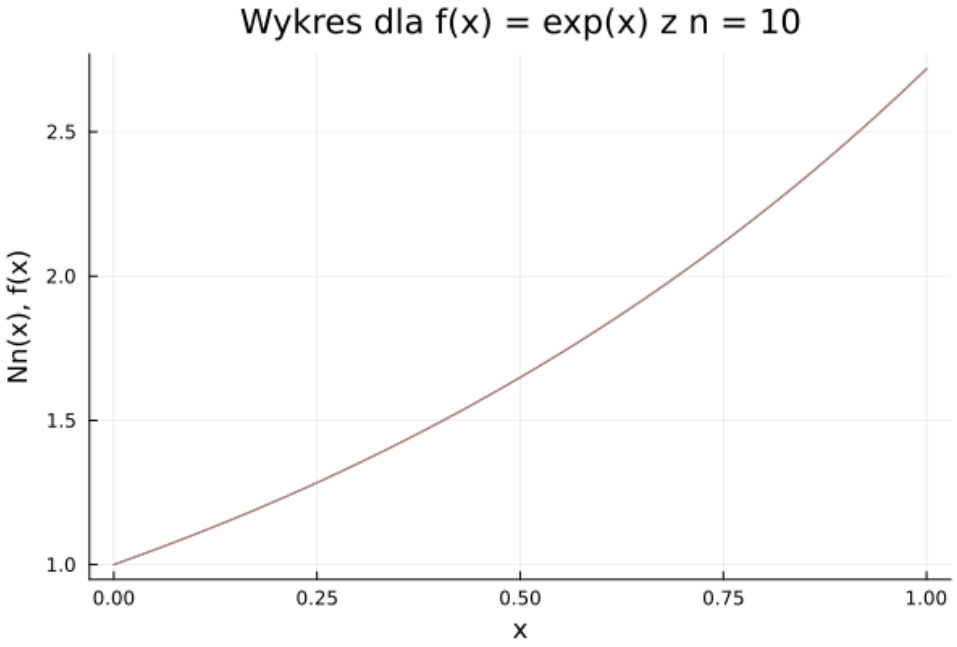
$$x_i = \frac{a+b}{2} + \frac{b-a}{2} \cos\left(\frac{2\pi i + \pi}{2n+2}\right), \quad i = 0, 1, \ldots, n$$

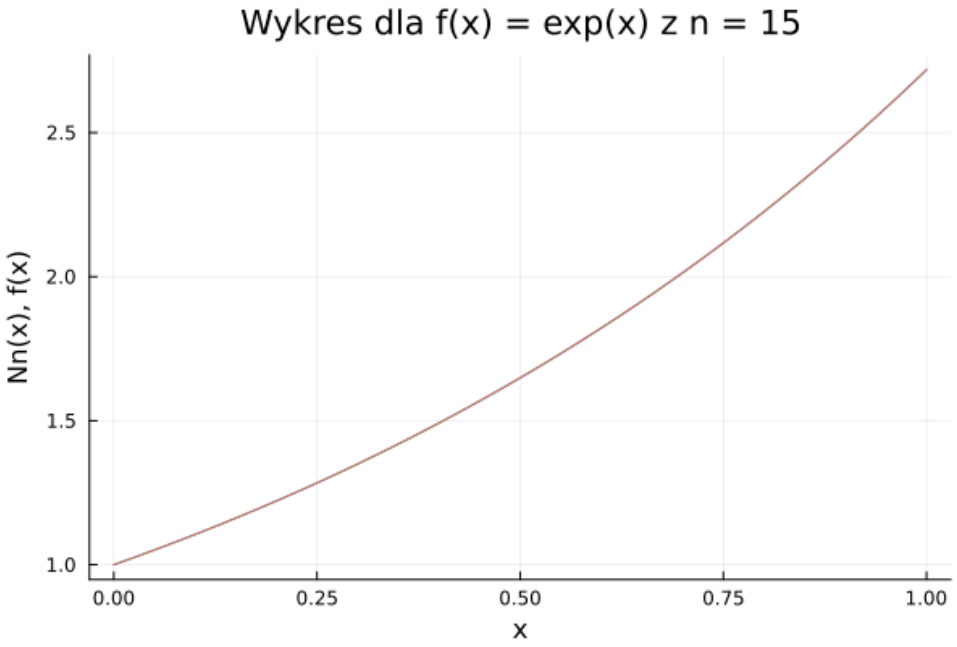
2.5 Zadanie 5.

W zadaniu 5. należy przetestować działanie zaimplementowanych funkcji na przykładach:

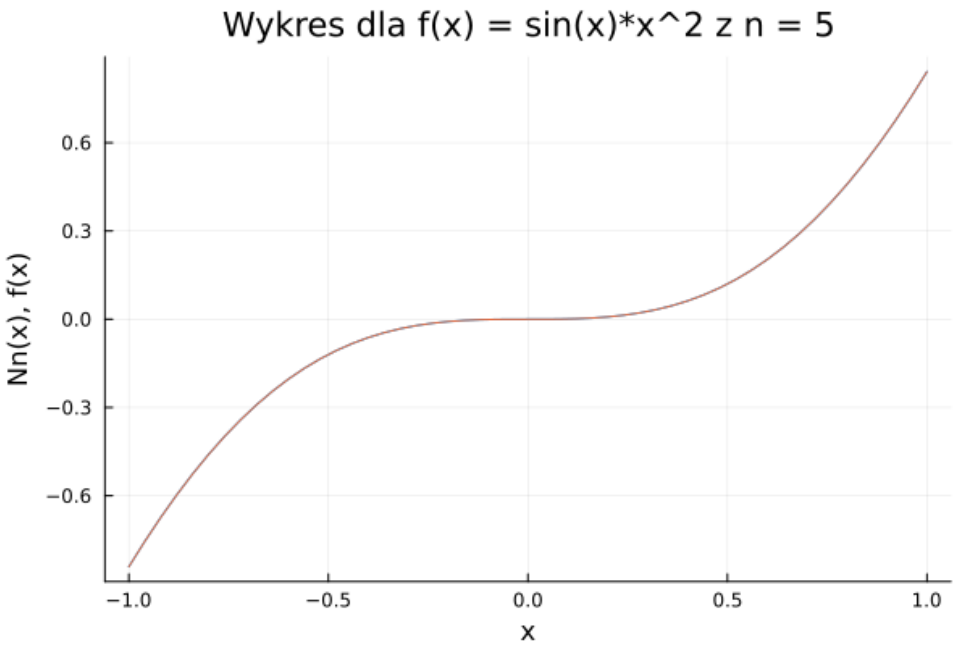
- $f(x) = e^x$ na przedziale $[0, 1]$ z stopniami $n = 5, 10, 15$.
- $f(x) = x^2 \sin(x)$ na przedziale $[-1, 1]$ z stopniami $n = 5, 10, 15$.

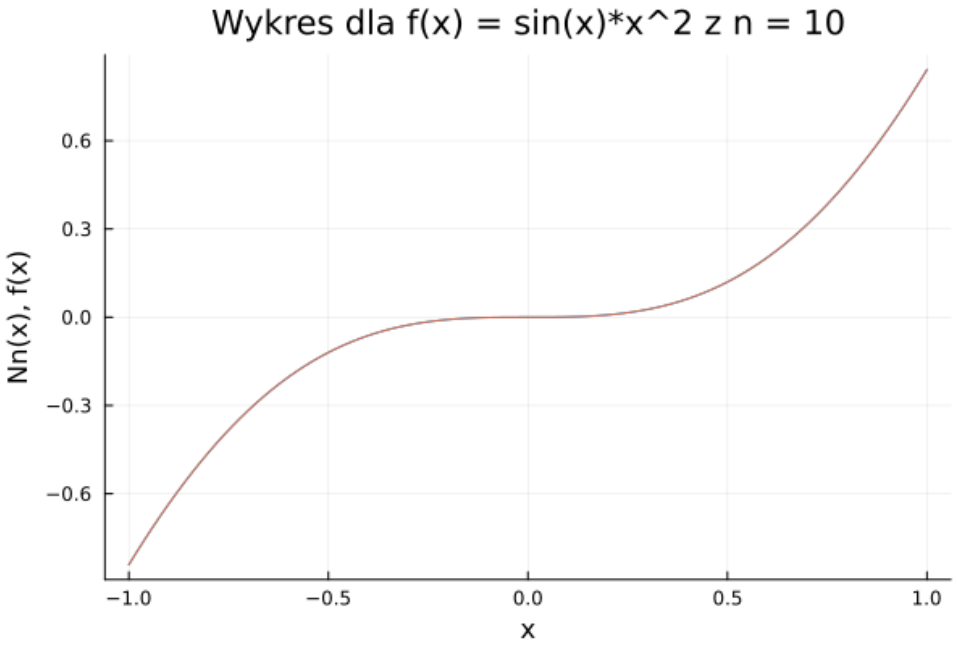


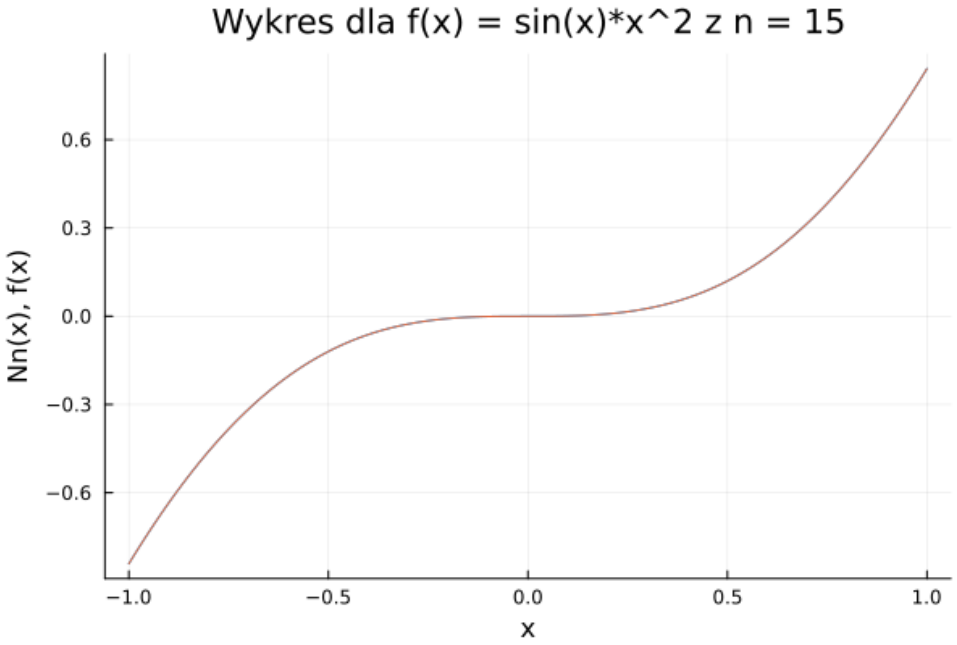




Jak widać na powyższych wykresach, wraz ze wzrostem liczby węzłów interpolacji nie zauważalna jest różnica między funkcją oryginalną a interpolowaną. Zatem interpolacja działa poprawnie dla funkcji $f(x) = e^x$. Na powyższych wykresach widać, że dla funkcji $f(x) = x^2 \sin(x)$ zachowuje





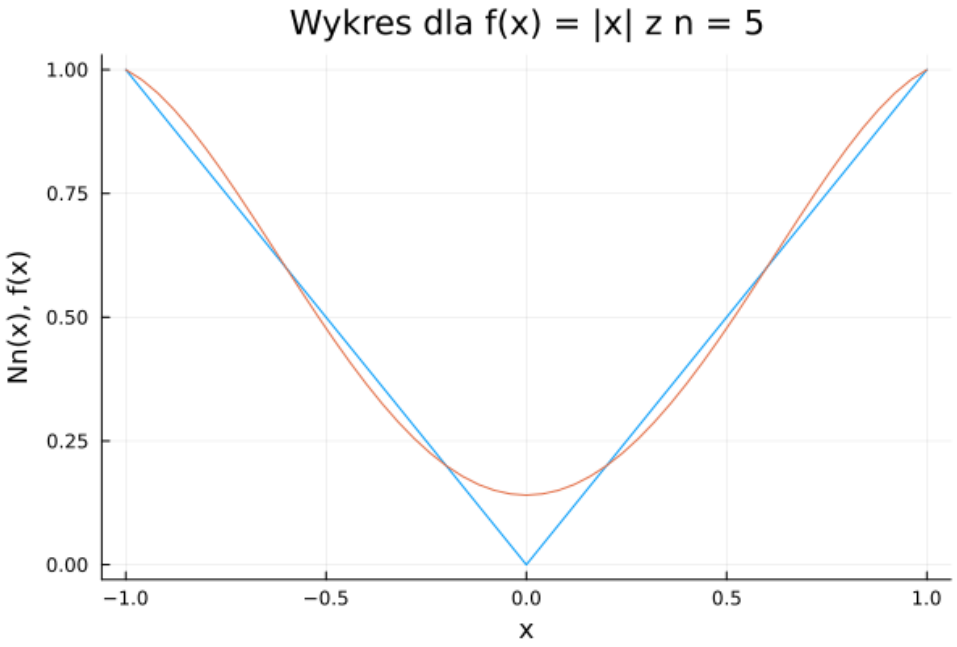


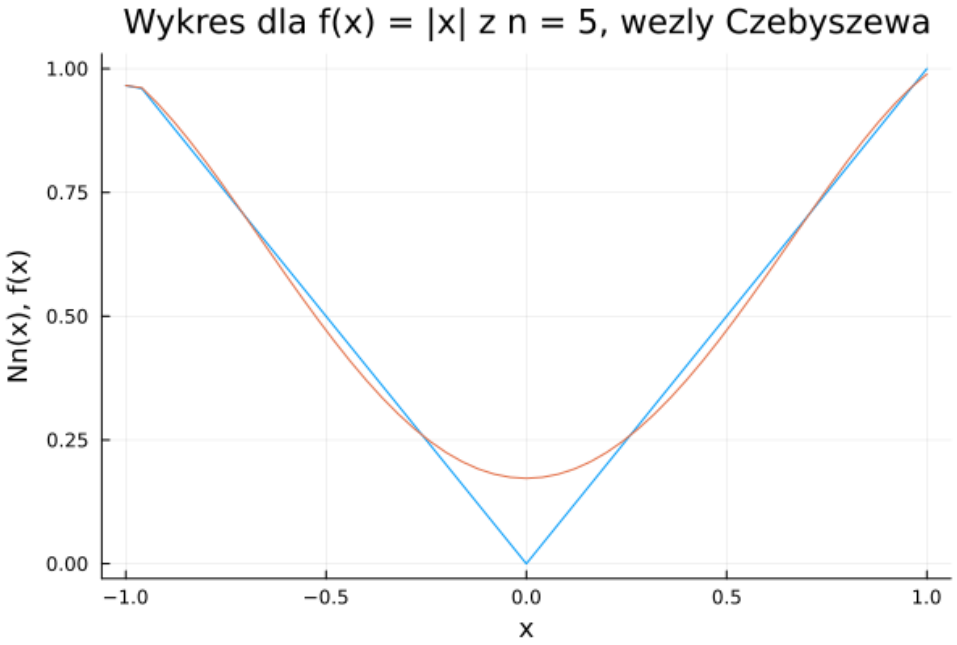
się w identyczny sposób jak dla $f(x) = e^x$. Zatem interpolacja działa poprawnie również dla tej funkcji.

2.6 Zadanie 6.

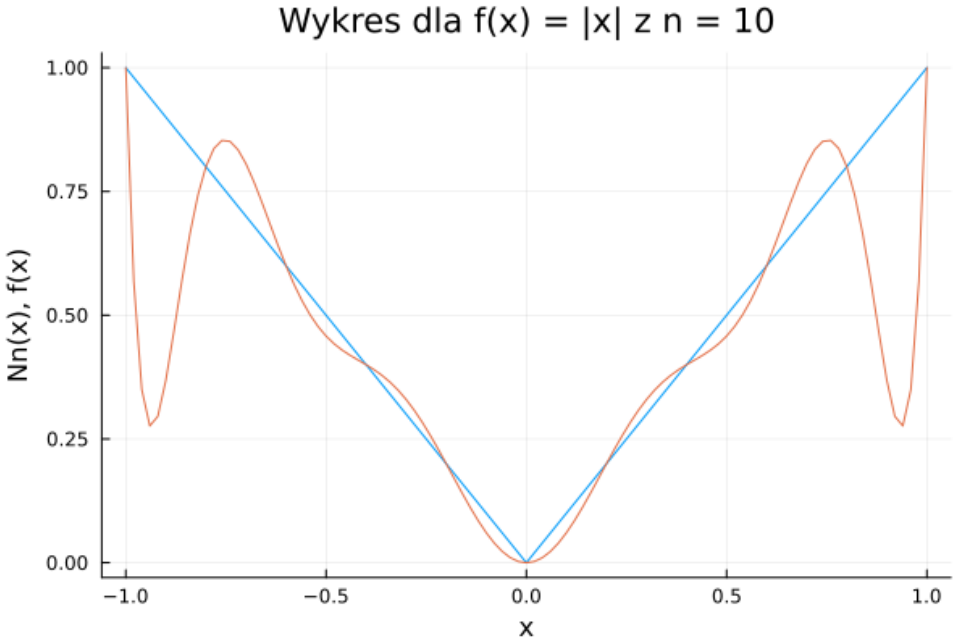
W zadaniu 6. należy porównać efekty interpolacji dla węzłów równoodległych i węzłów Czebyszewa na przykładzie funkcji:

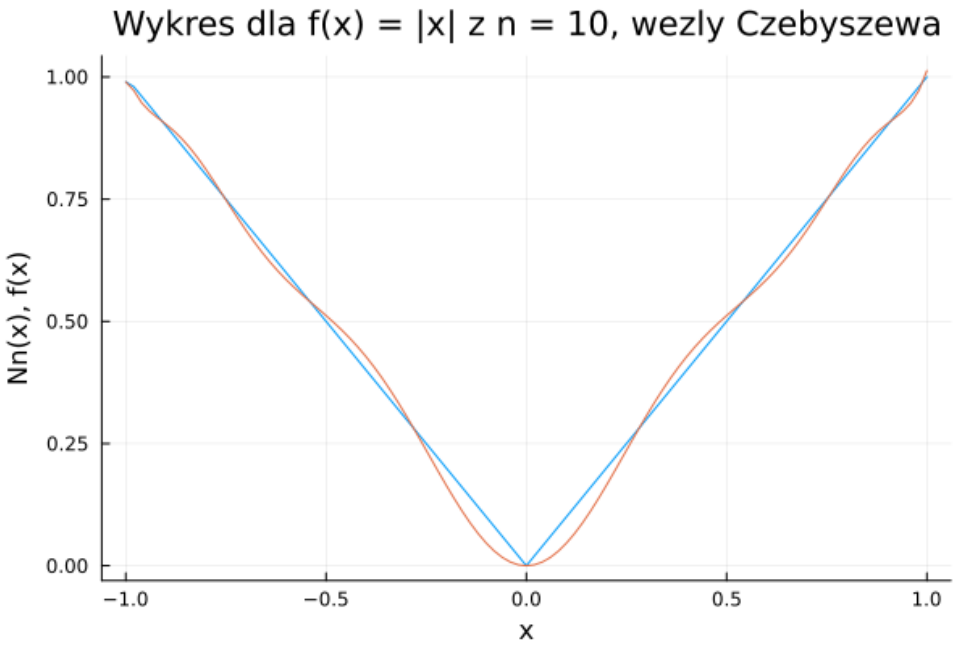
- $f(x) = |x|$ na przedziale $[-1, 1]$ z stopniami $n = 5, 10, 15$.
- $f(x) = \frac{1}{1+x^2}$ na przedziale $[-5, 5]$ z stopniami $n = 5, 10, 15$.

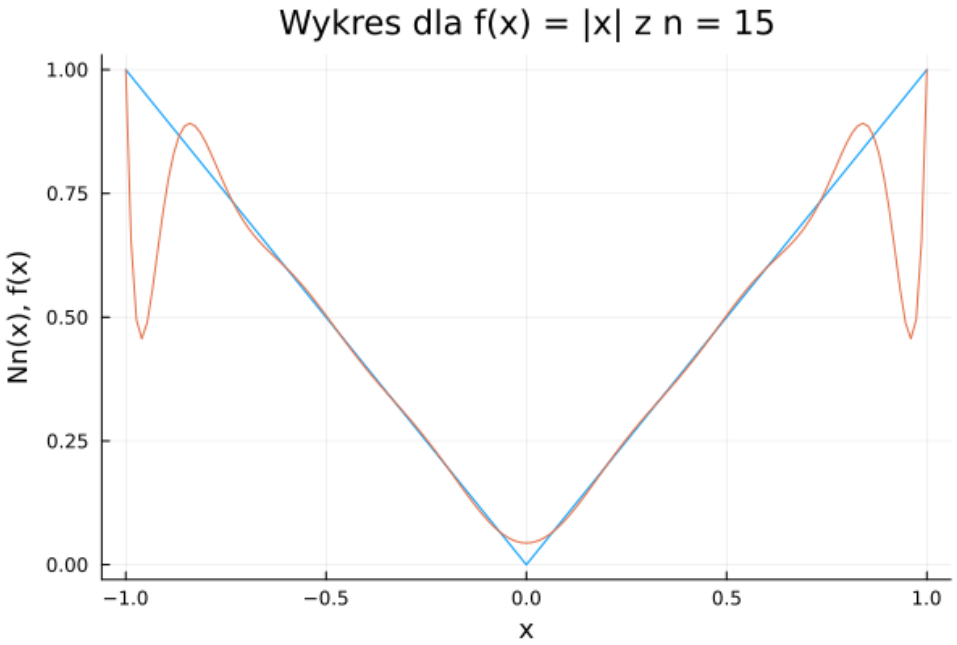


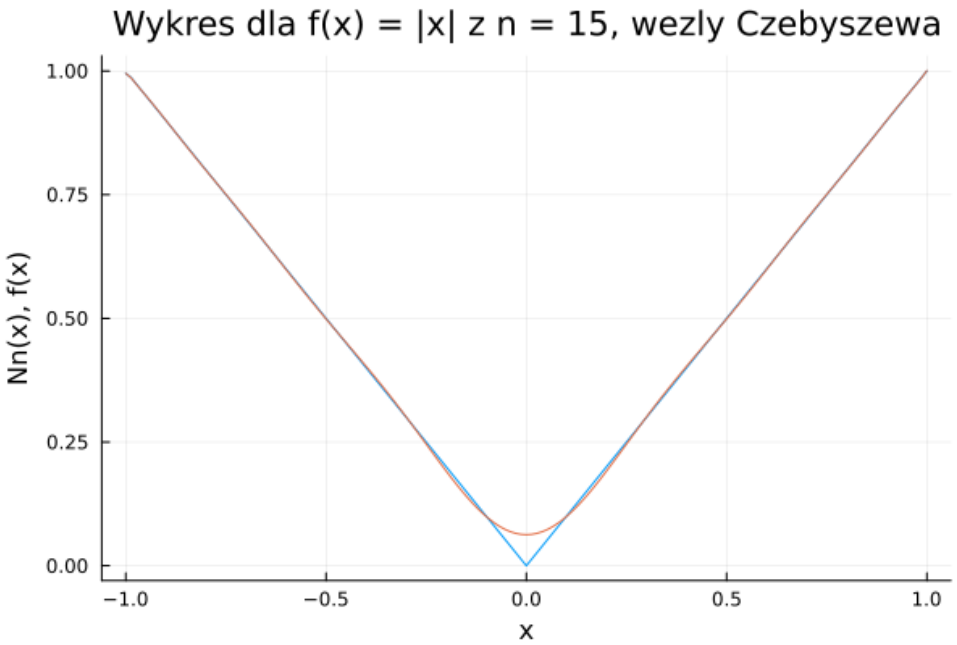


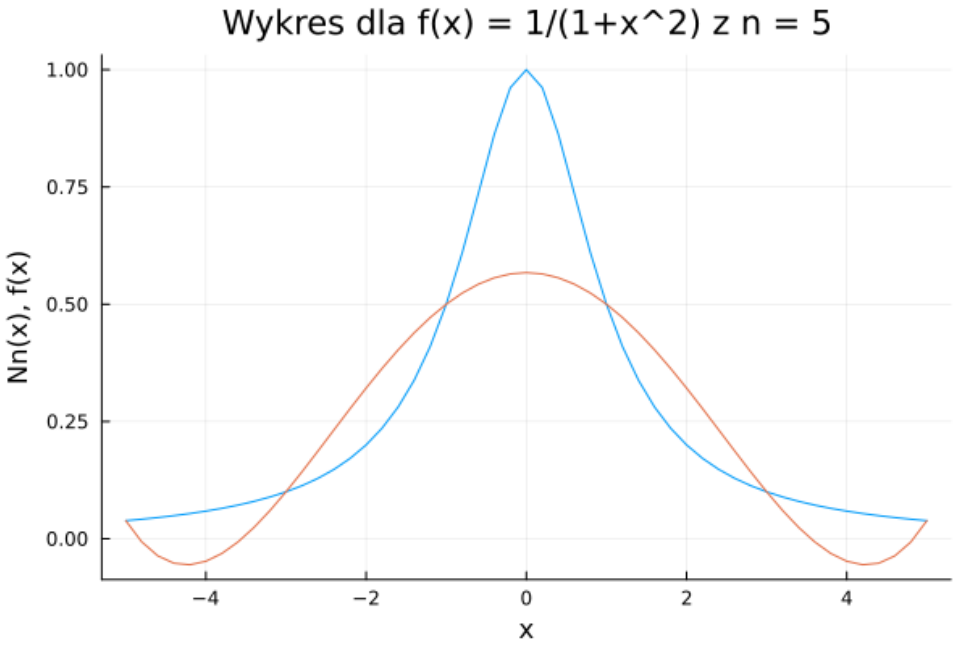
Zdaje się, że interpolacja funkcji $f(x) = |x|$ prowadzi do znacznie większych błędów niżeli dla poprzednich funkcji.
Na powyższych wykresach widać, że dla interpolacja za pomocą węzłów równoodległych prowadzi do dużych błędów przy krańcu przedziału. Wynika to z faktu, że funckja nie ma ciągłej pochodnej w punkcie $x = 0$ (ma ostry "czubek"). Natomiast interpolacja za pomocą węzłów Czebyszewa radzi sobie lepiej, zmniejszając błąd przy krańcach przedziału, i generalnie lepiej dopasowując się do funkcji oryginalnej. Widać również, że wraz ze wzrostem liczby węzłów interpolacji błąd maleje, inaczej jak w przypadku węzłów równoodległych, gdzie błąd pozostaje na podobnym rozmiarze przy krańcach przedziału.

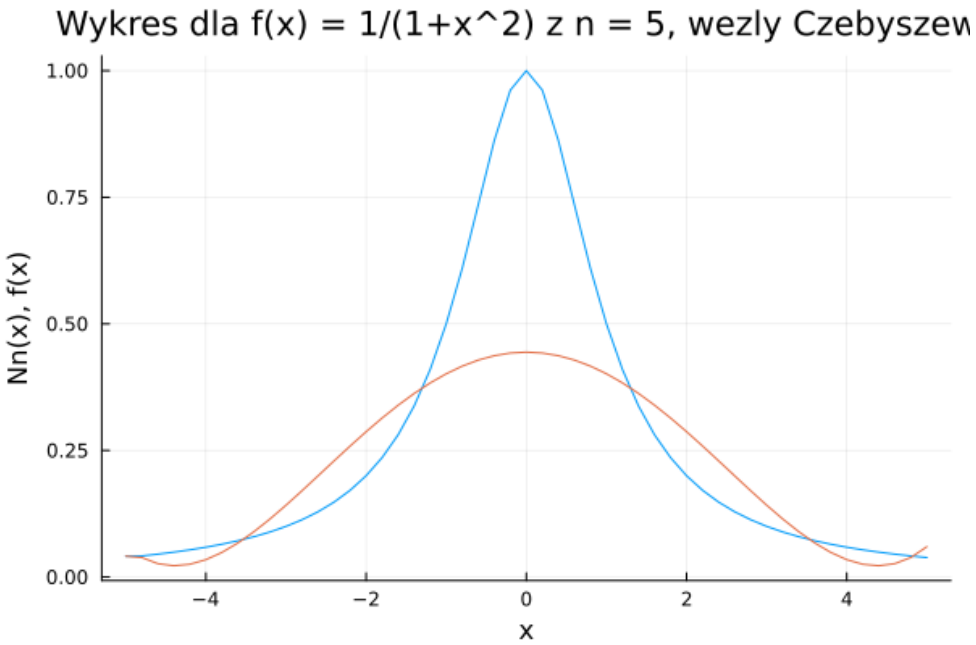

Wykres dla f(x) = |x| z n = 10
1.00
0.75
0.50
0.25
0.00
Nn(x), f(x)
−1.0
−0.5
0.0
0.5
1.0
x

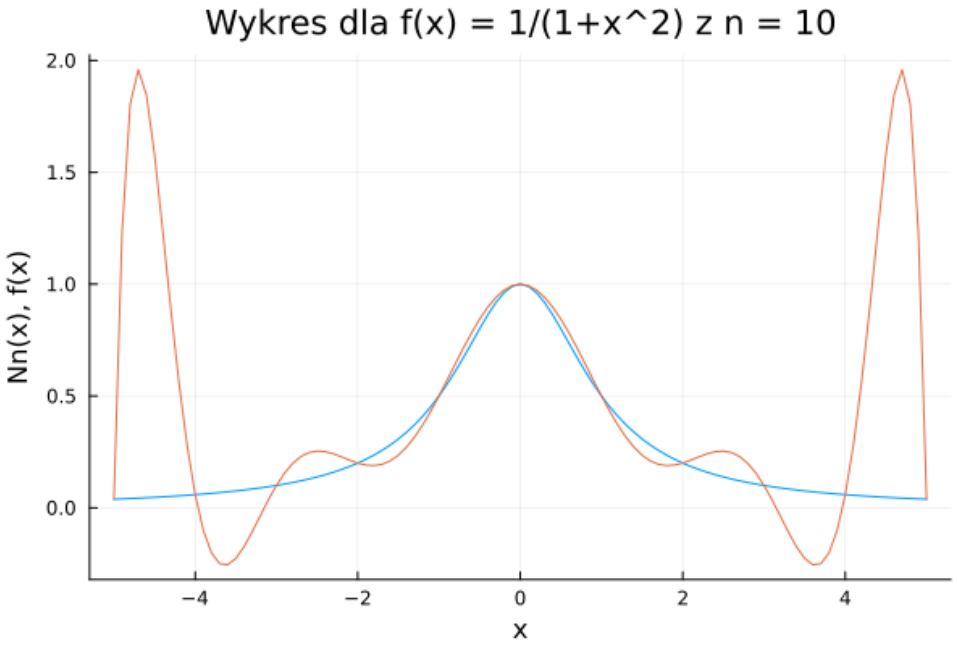

Wykres dla f(x) = |x| z n = 10, wezly Czebyszewa
1.00
0.75
0.50
0.25
0.00
Nn(x), f(x)
−1.0
−0.5
0.0
0.5
1.0
x

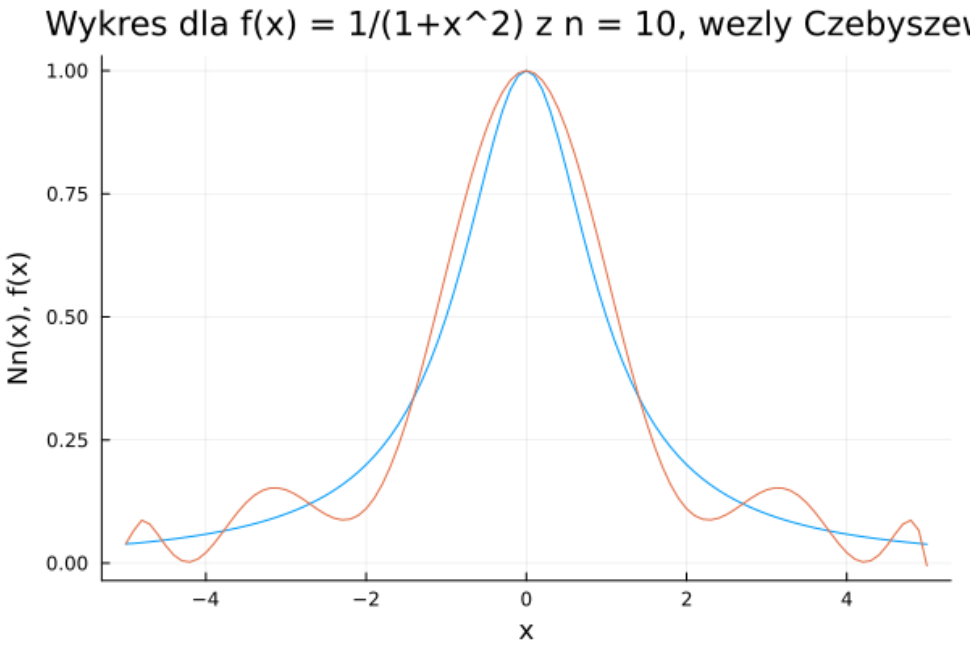

Wykres dla f(x) = |x| z n = 15
Nn(x), f(x)
x
1.00
0.75
0.50
0.25
0.00
−1.0
−0.5
0.0
0.5
1.0

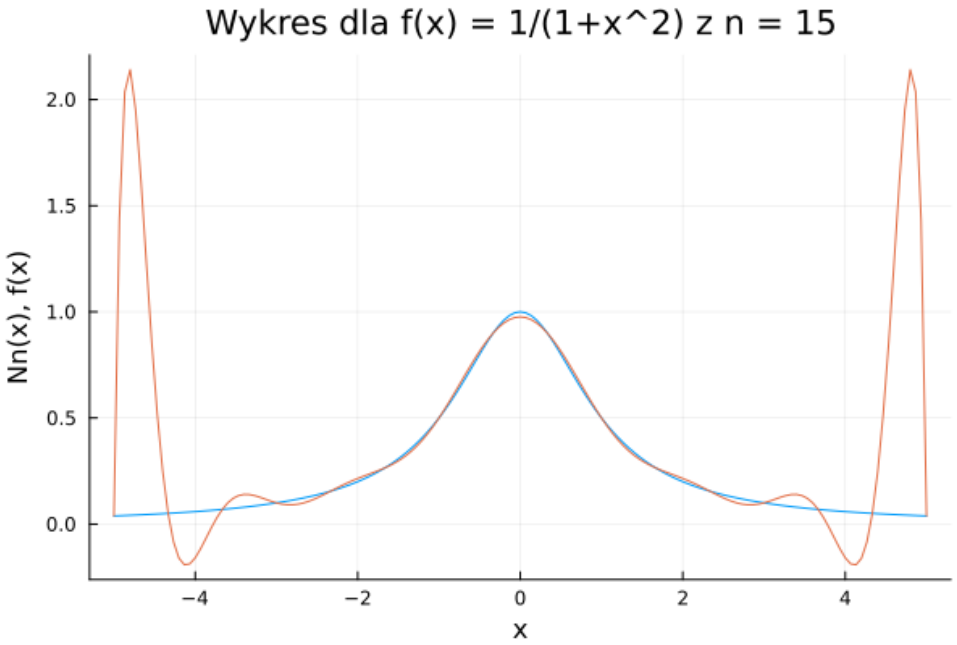

Wykres dla f(x) = |x| z n = 15, wezly Czebyszewa
Nn(x), f(x)
x
1.00
0.75
0.50
0.25
0.00
−1.0
−0.5
0.0
0.5
1.0

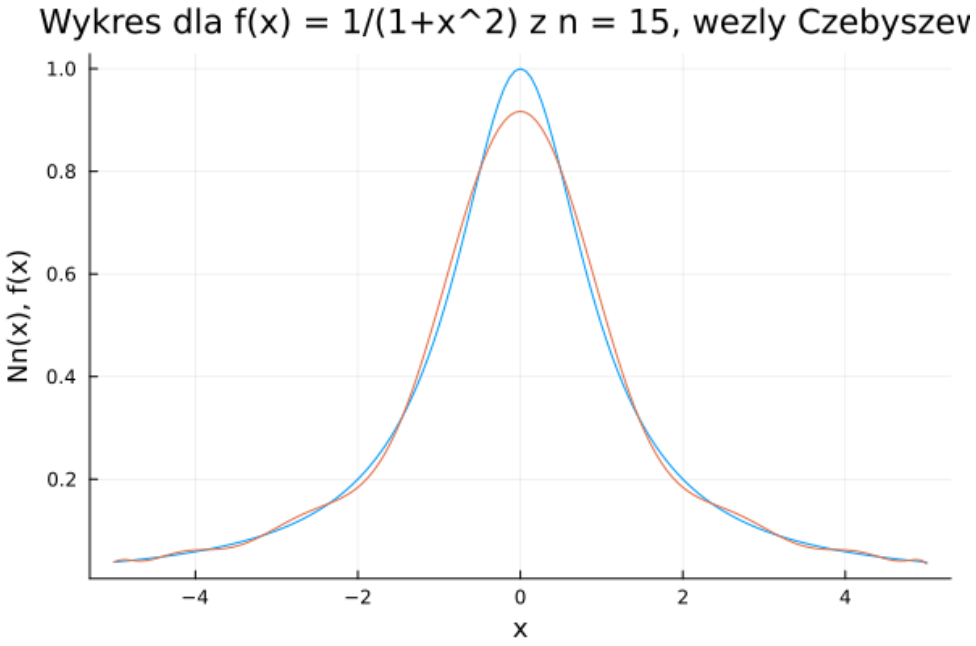

Wykres dla f(x) = 1/(1+x^2) z n = 5
1.00
0.75
0.50
0.25
0.00
Nn(x), f(x)
−4
−2
0
2
4
x


Wykres dla f(x) = 1/(1+x^2) z n = 5, wezly Czebyszew
1.00
0.75
0.50
0.25
0.00
Nn(x), f(x)
−4
−2
0
2
4
x


Wykres dla f(x) = 1/(1+x^2) z n = 10
Nn(x), f(x)
x
2.0
1.5
1.0
0.5
0.0
−4
−2
0
2
4


Wykres dla f(x) = 1/(1+x^2) z n = 10, wezly Czebysze
Nn(x), f(x)
x
1.00
0.75
0.50
0.25
0.00
−4
−2
0
2
4


Wykres dla f(x) = 1/(1+x^2) z n = 15
Nn(x), f(x)
x
2.0
1.5
1.0
0.5
0.0
−4
−2
0
2
4


Wykres dla f(x) = 1/(1+x^2) z n = 15, wezly Czebyszev
Nn(x), f(x)
x
1.0
0.8
0.6
0.4
0.2
−4
−2
0
2
4

Podobnie jak w przypadku funkcji $f(x) = |x|$, interpolacja funkcji $f(x) = \frac{1}{1+x^2}$ prowadzi do znacznych błędów przy użyciu węzłów równoodległych, szczególnie przy krańcach przedziału. Wraz ze wzrostem liczby węzłów interpolacji, błąd ten się zwiększa, co jest przykładem efektu Rungego. Natomiast użycie węzłów Czebyszewa znacząco redukuje te błędy, prowadząc do lepszego dopasowania do funkcji oryginalnej, zwłaszcza przy większej liczbie węzłów.

3 Wnioski

W przeprowadzonych eksperymentach zaobserwowano, że interpolacja wielomianowa jest skuteczną metodą przybliżania funkcji, jednak jej efektywność zależy od wyboru węzłów interpolacji. Węzły równoodległe mogą prowadzić do znacznych błędów, zwłaszcza przy większej liczbie węzłów, co jest ilustrowane przez efekt Rungego. Z kolei węzły Czebyszewa okazały się bardziej stabilne i skuteczne, redukując błędy interpolacji, szczególnie przy funkcjach z ostrymi zmianami.