Lista nr 4

1. Obliczanie ilorazów różnicowych

1.1. Zadanie interpolacji wielomianowej

Zadanie interpolacji wielomianowej polega na znalezieniu wielomianu $p(x)$ co najwyżej n-tego stopnia, takiego że dla danych $n+1$ punktów (x_i, y_i) $0 \le i \le n$ zachodzi $p(x_i) = y_i$.

Jednym ze sposobów na rozwiązanie tego zadania jest znalezienie postaci $p(x)$ w bazie $1, (x - x_0), (x - x_0)(x - x_1), ..., \Pi_{i=0}^{n-1}(x - x_i)$. Nazywamy to postacią interpolacyjną Newtona, a współczynniki wielomianu nazywamy ilorazami różnicowymi i oznaczamy $c_i = f[x_0, x_1, ..., x_i]$.

1.2. Wyznaczanie ilorazów różnicowych

Ilorazy różnicowe można wyliczyć za pomocą zależności rekurencyjnej:

$$f[x_i] = f(x_i)$$

$$f[x_0, x_1, ..., x_n] = \frac{f[x_1, x_2, ..., x_n] - f[x_0, x_1, x_{n-1}]}{x_n - x_0}$$

Z tych zależności wynika, że problem wyznaczenia ilorazów różnicowych możemy rozwiązać stosując metodę programowania dynamicznego.

$$\begin{bmatrix}
f[x_0] & & & & & & & & \\
& \searrow & & & & & & & \\
f[x_1] & \to & f[x_0, x_1] & & & & & & \\
& \searrow & & \searrow & & & & & \\
f[x_2] & \to & f[x_1, x_2] & \to & f[x_0, x_1, x_2] & & & & \\
& \searrow & & \searrow & & \searrow & & & \\
\vdots & & ... & & ... & & \ddots & & \\
& \searrow & & \searrow & & \searrow & & \searrow & \\
f[x_n] & \to & f[x_{n-1}, x_n] & \to & f[x_{n-2}, x_{n-1}, x_n] & \to & ... & \to & f[x_0, ..., x_n]
\end{bmatrix}$$

Współczynniki wielomianu rozwiązującego problem interpolacji znajdują się po przekątnej tej macierzy. Oznaczmy elementy tej macierzy jako $b_{j,i}$ dla $0 \le j, i \le n$. Wtedy znając kolumnę $i - 1$ możemy wyliczyć elementy kolumny i za pomocą wzoru:

$$b_{j,i} = \frac{b_{j,i-1} - b_{j-1,i-1}}{x_j - x_{j-i}} \text{ dla } 1 \le i \le j \le n$$

Można też zauważyć, że w końcowym algorytmie nie musimy pamiętać całej macierzy, a jedynie wektor opisujący daną kolumnę. Do obliczenia elementu w wierszu j potrzebujemy jedynie poprzednich elementów z wierszów j i $j - 1$. Zatem jeżeli zaczniemy wyznaczanie danej kolumny „od tyłu”, tj. od $j = n$ do $j = i$, to możemy nadpisywać element w wierszu j elementem z następnej kolumny, bo nie przyda się już do kolejnych obliczeń.

```
function ILORAZY-RÓŻNICOWE(x = [x_0, ..., x_n], y = [f(x_0), ..., f(x_n)])
    fx ← y
    for i from 1 to n do
        for j from n to i do
            fx_j ← (fx_j − fx_{j−1}) / (x_j − x_{j−i})
    return fx
```

2. Wyznaczanie wartości wielomianu interpolacyjnego

2.1. Uogólniony wzór Hornera

W poprzednim zadaniu udało się wyznaczyć wielomian interpolacyjny w postaci Newtona:

$$p(x) = f[x_0] + f[x_0, x_1](x - x_0) + \dots + f[x_0, \dots, x_n]\Pi_{i=0}^{n-1}(x - x_i)$$

Ten wzór można inaczej przedstawić, wyciągając wspólne czynniki:

$$p(x) = f[x_0] + (x - x_0)(f[x_0, x_1] + (x - x_1)(f[x_0, x_1, x_2] + \dots))$$

Tym samym, licząc „od środka", otrzymujemy ciąg wielomianów pośrednich $w_i(x)$ wyrażających się zależnością:

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

$$w_i(x) = f[x_0, \dots, x_i] + (x - x_i)w_{i+1}(x) \text{ dla } 0 \leq i \leq n - 1$$

Gdzie $w_0(x) = p(x)$. Tą metodę nazywamy uogólnionym algorytmem Hornera. Pozwala na wyliczenie wartości $p(x)$ w punkcie x w liniowej liczbie działań, w przeciwieństwie do metody naiwnej, która potrzebuję kwadratowej liczby działań.

```
function WARTOŚĆ-NEWTON(t, x = [x_0, ..., x_n], fx = [f[x_0], ..., f[x_0, ..., x_n]])
    w ← fx_n
    for i from n − 1 to 0 do
        w ← fx_i + w · (t − x_i)
    return w
```

3. Zamiana postaci Newtona na postać naturalną

3.1. Wykorzystanie wzoru Hornera

Wiemy, że wielomian pośredni uzyskany metodą Hornera $w_i(x)$ jest maksymalnie stopnia $n - i$. Przedstawmy dwa kolejne uzyskane wielomiany w postaciach naturalnych:

$$w_{i+1}(x) = a_0 + a_1x + a_2x^2 + \dots + a_{n-i-1}x^{n-i-1}$$

$$w_i(x) = b_0 + b_1x + b_2x^2 + \dots + b_{n-i}x^{n-i}$$

Z zależności rekurencyjnej $w_i(x) = f[x_0, \dots, x_i] + (x - x_i)w_{i+1}(x)$ wynika że:

$$b_0 + b_1x + \dots + b_{n-i}x^{n-i} =$$

$$= f[x_0, ..., x_i] + (x - x_i)(a_0 + a_1 x + ... + a_{n-i-1} x^{n-i-1}) =$$

$$= f[x_0, ..., x_i] + (a_0 x + ... + a_{n-i-1} x^{n-i}) - (a_0 x_i + ... + a_{n-i-1} x_i x^{n-i-1}) =$$

$$= f[x_0, ..., x_i] - a_0 x_i + (a_0 - a_1 x_i) x + ... + (a_{n-i-2} - a_{n-i-1} x_i) x^{n-i-1} + a_{n-i-1} x^{n-i}$$

Z równości wielomianów wynika równość ich współczynników, więc:

$$b_0 = f[x_0, ..., x_i] - a_0 x_i,\ b_{n-i} = a_{n-i-1},\ b_j = a_{j-1} - a_j x_i \text{ dla } 1 \leq j \leq n - i - 1$$

3.2. Algorytm

Znając więc współczynniki postaci naturalnej wielomianu pośredniego $w_{i+1}(x)$, możemy wyznaczyć współczynniki postaci naturalnej wielomianu $w_i(x)$. Tym samym zaczynając od $w_n(x) = f[x_0, ..., x_n]$, możemy policzyć szukane współczynniki $p(x) = w_0(x)$.

Tak samo jak w zadaniu 1, wystarczy tablica $n + 1$ elementów. Do obliczenia b_j potrzeba jedynie a_j i a_{j-1}, więc idąc „od tyłu", możemy nadpisywać niepotrzebne już do obliczeń wartości. Widać też, że algorytm wykonuje $O(n^2)$ działań.

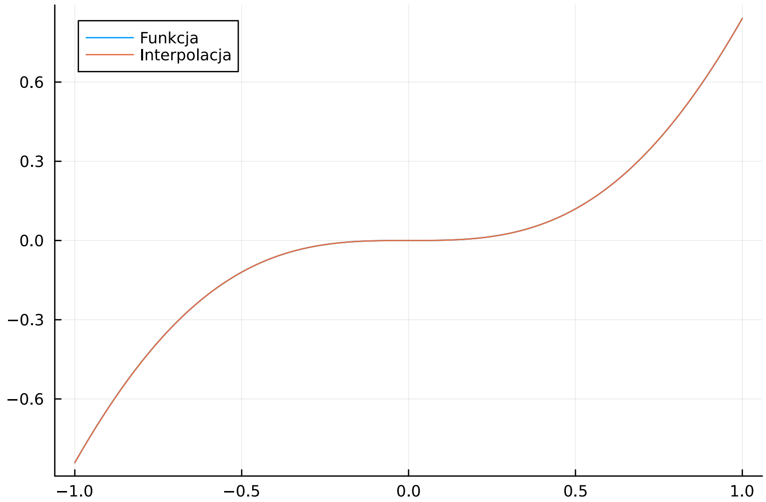
function Postać-Naturalna($x = [x_0, ..., x_n]$, $fx = [f[x_0], ..., f[x_0, ..., x_n]]$)
    $b \leftarrow [0, ..., 0]$
    $b_0 \leftarrow fx_n$
    for i from $n - 1$ to 0 **do**
        $b_{n-i} \leftarrow b_{n-i-1}$
        for j from $n - i - 1$ to 1 **do**
            $b_j \leftarrow b_{j-1} - b_j x_i$
        $b_0 \leftarrow fx_i - b_0 x_i$
    return b

4. Interpolacja na rysunkach

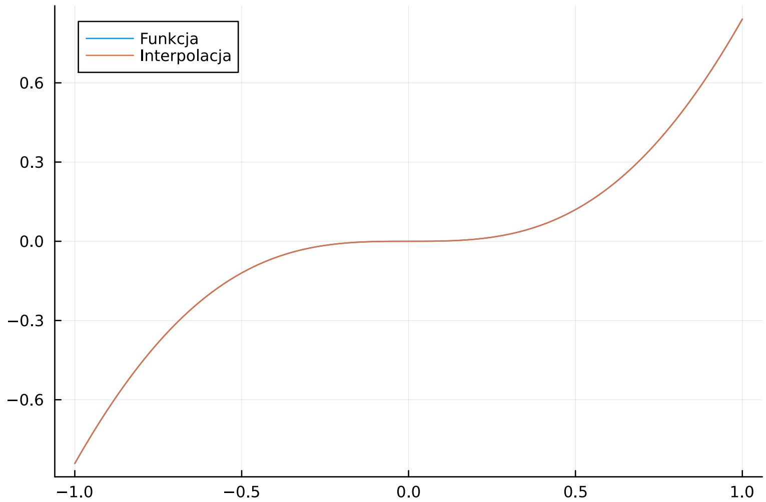
4.1. Opis

Zastosowałem pokazane algorytmy do przedstawienia interpolacji przykładowych funkcji dla równoodległych węzłów:

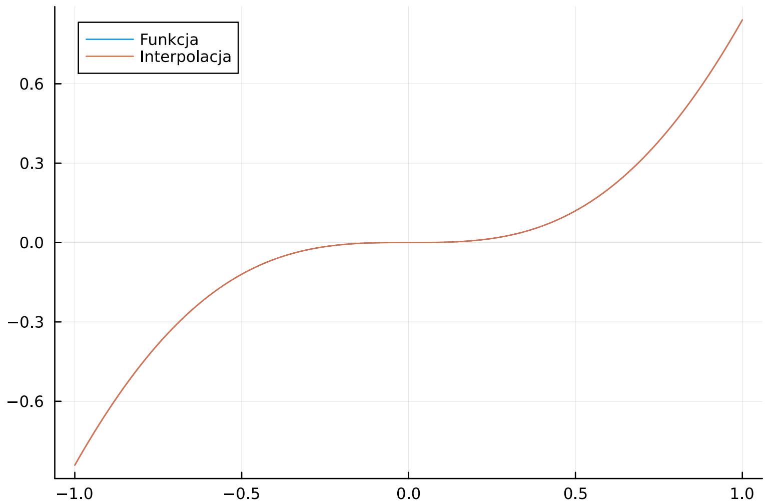
- e^x na przedziale $[0, 1]$
- $x^2 \sin x$ na przedziale $[-1, 1]$



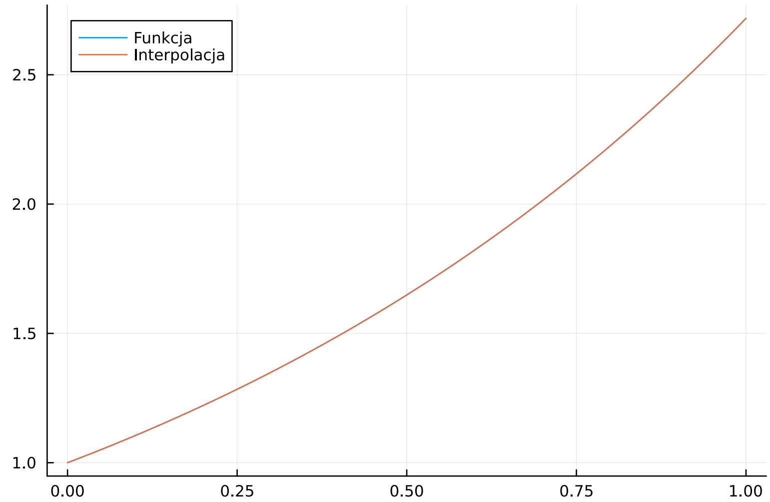
Rysunek 1: Wielomian stopnia 5 interpolujacy $x^2 \sin x$



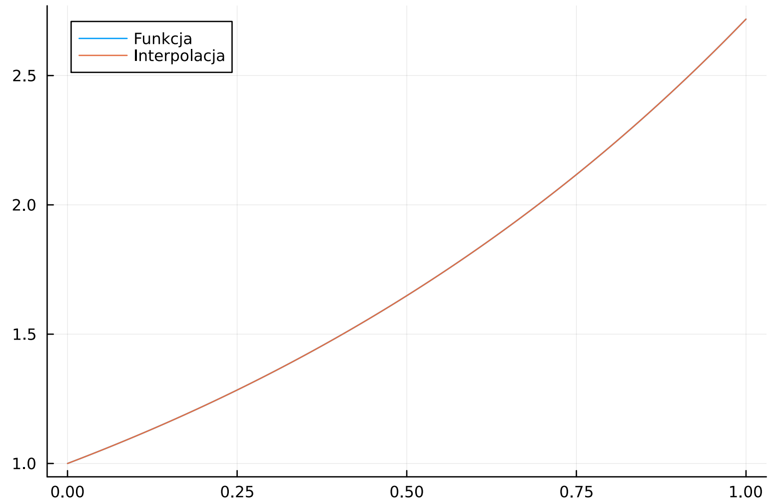
Rysunek 2: Wielomian stopnia 10 interpolujacy $x^2 \sin x$



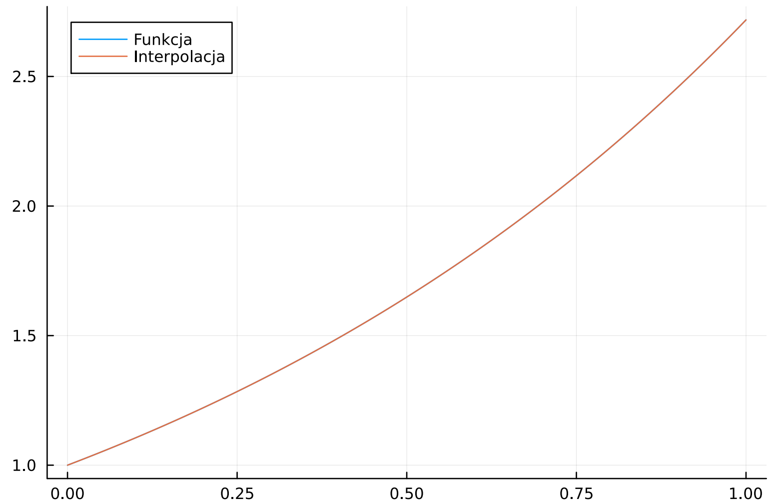
Rysunek 3: Wielomian stopnia 15 interpolujacy $x^2 \sin x$



Rysunek 4: Wielomian stopnia 5 interpolujacy e^x



Rysunek 5: Wielomian stopnia 10 interpolujacy e^x



Rysunek 6: Wielomian stopnia 15 interpolujacy e^x

4.2. Wnioski

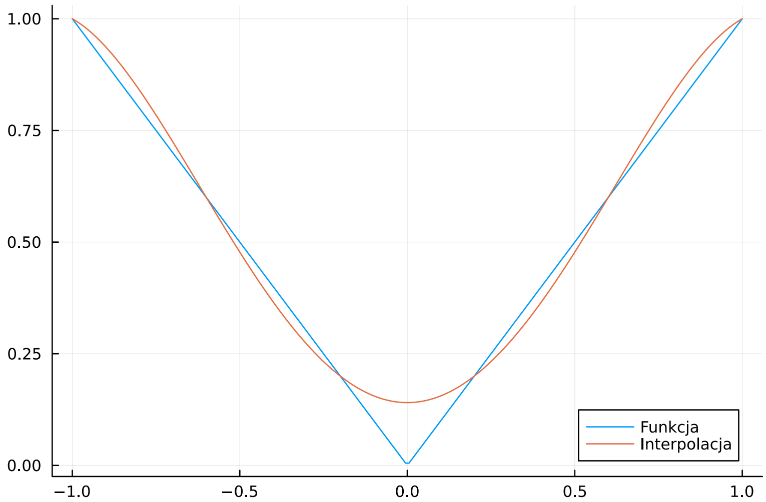
Funkcje te dają sie bardzo łatwo interpolować przy pomocy równoodległych węzłów, nie ma znaczących błędów nawet dla wielomianów niskiego stopnia.

5. Błędy interpolacji

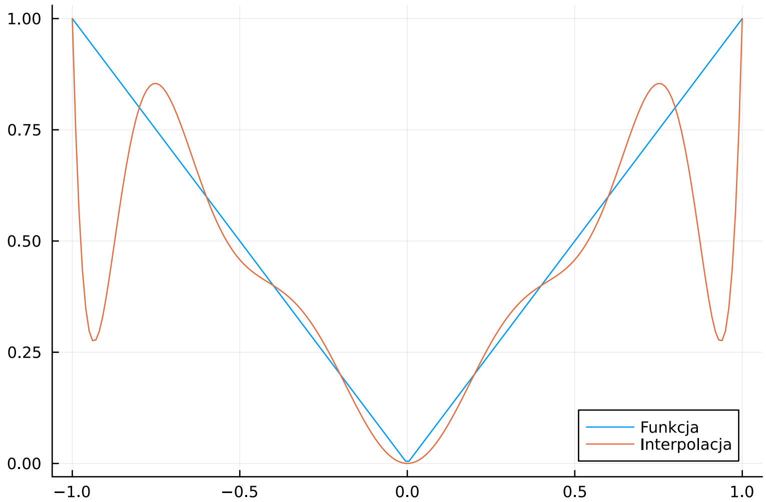
5.1. Opis

Tym razem, znowu z równoodległymi węzłami, przeprowadziłem interpolację funkcji:
- $|x|$ na przedziale $[-1, 1]$
- $\frac{1}{1+x^2}$ na przedziale $[-5, 5]$

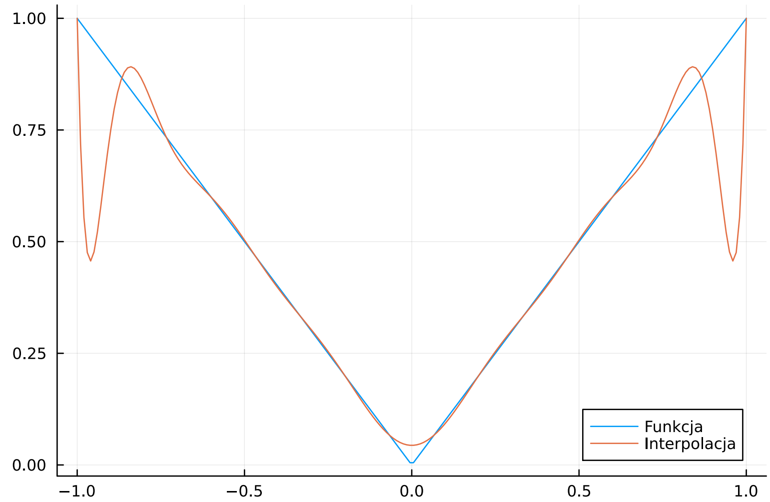
5.2. Wyniki



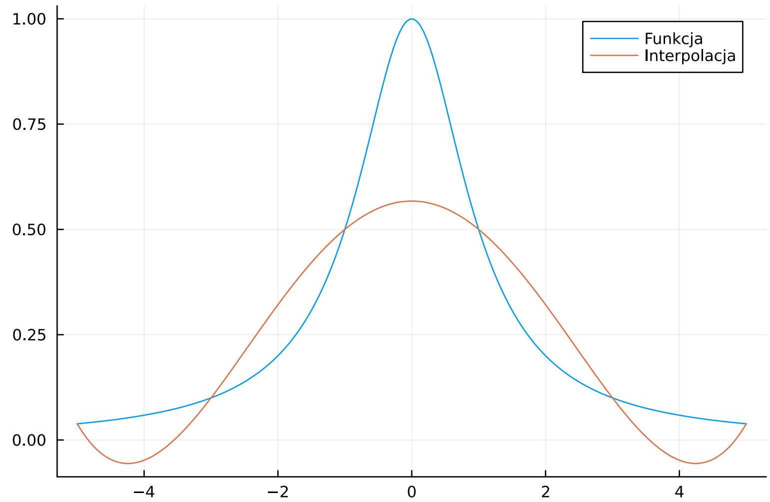
Rysunek 7: Wielomian stopnia 5 interpolujacy $|x|$



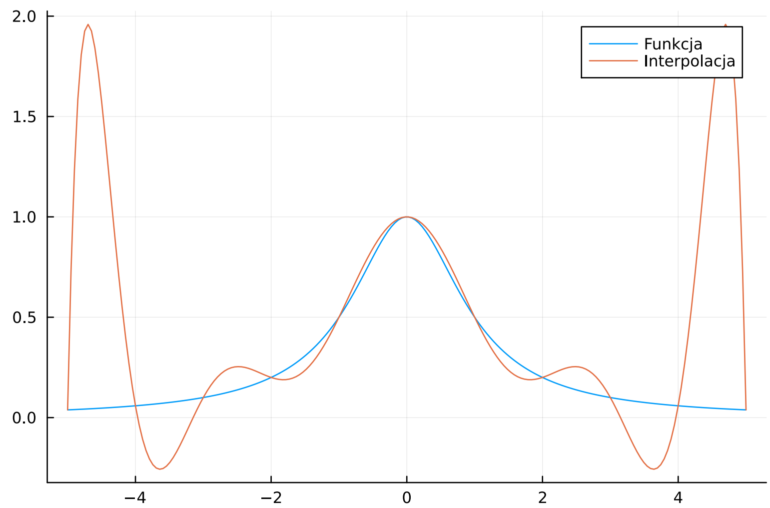
Rysunek 8: Wielomian stopnia 10 interpolujacy $|x|$



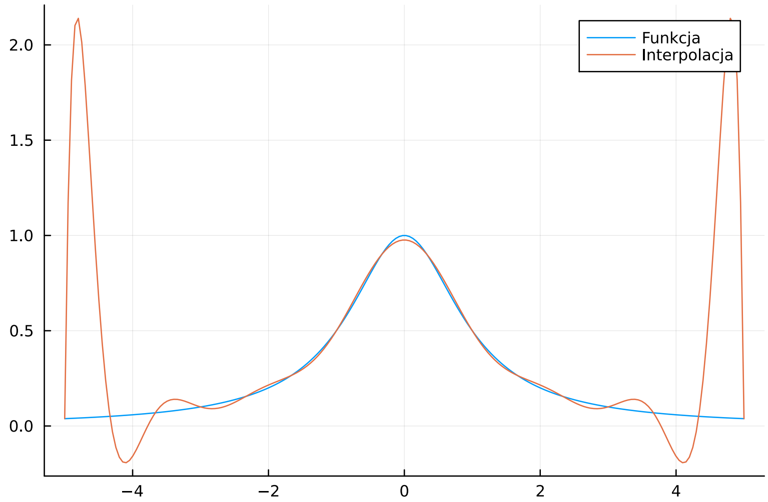
Rysunek 9: Wielomian stopnia 15 interpolujacy $|x|$



Rysunek 10: Wielomian stopnia 5 interpolujacy $\frac{1}{1+x^2}$



Rysunek 11: Wielomian stopnia 10 interpolujacy $\frac{1}{1+x^2}$



Rysunek 12: Wielomian stopnia 15 interpolujacy $\frac{1}{1+x^2}$

5.3. Wnioski

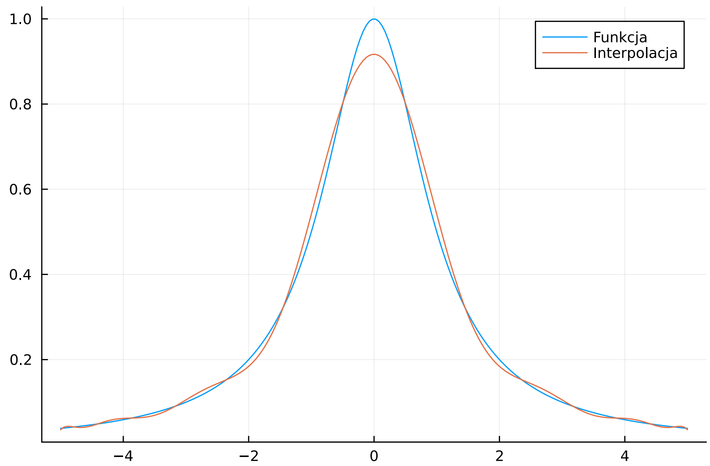
Na wykresach pojawia się widoczna różnica pomiędzy wartościami funkcji i interpolującego go wielomianu. Co istotne widoczny błąd interpolacji nie maleje wraz ze zwiększaniem stopnia wielomianu, a wręcz w niektórych miejscach rośnie.

W przypadku funkcji $f(x) = |x|$, błąd może się brać z tego, że pochodna funkcji nie jest ciągła - f nie jest różniczkowalna w punkcie $x = 0$. Natomiast algorytm używa gładkich wielomianów, co powoduje zniekształcenia.

Przy funkcji $f(x) = \frac{1}{1+x^2}$ mamy do czynienia z efektem Rungego. Błąd interpolacji możemy wyrazić wzorem:

$$f(x) - p(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) \Pi_{i=0}^{n} (x - x_i) \text{ z pewnym } \xi_x \in (a, b)$$

Okazuje się, że wraz z rosnącym n bardzo szybką rosną wartości n-tej pochodnej f i ten błąd rozbiega dla interpolacji wielomianem o równoodległych węzłach. W tym przypadku co można zrobić, to zastosować wielomiany Czebyszewa lub użyć innych metod interpolacji.



Rysunek 13: Wielomian stopnia 15 interpolujacy $\frac{1}{1+x^2}$ z wykorzystaniem wielomianu Czebyszewa