Obliczenia Naukowe Laboratorium 4

Adam Jamroziński

November 2024

Na tej liście implementujemy różne funkcje z zakresu interpolacji. Następnie przeprowadzamy na nich testy.

Zadania implementacyjne 1-4

Obliczanie ilorazów różnicowych

dla zadanych $\mathbf{x} = [x_0, x_1, ..., x_n]$ gdzie $x_i, 0 \leqslant i \leqslant n$ kolejne n więzłów oraz $\mathbf{f} = [f(x_k) : x_k \in \mathbf{x}]$ dla jakiejś funkcji f. Oczywiście korzystamy z faktu, że

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

Stąd prosta procedura łącząca poprzednie w coraz dłuższcze ilorazy dzięki której możemy obliczyć iloraz modyfikując jedną tabele quotients zawierającą coraz dłuższe pod względem węzłów iloczyny różnicowe. Pierwsza pętla przejdzie po $i : 1 \leqslant i \leqslant n$, gdzie i reprezentuje odległość pierwszego i ostatniego węzła potrzebnego do obliczenia $\frac{1}{x_j - x_{j+i}}$ oraz to o ile mniej istnieje ilorazow roznicowych opartych na $i + 1$ różnych węzłach niż długości 0 (czyli $f[x_0], f[x_1], ...$ itd. których jest $n + 1$). Następnie $j : 0 \leqslant j \leqslant n - i$ gdzie obliczamy kolejne ilorazy $\mathbf{f}[j] = f[x_j, x_{j+1}, ..., x_{j+i}]$

Algorithm 1: differenceQuotient

Output: quotients $= [f[x_0], f[x_0, x_1], ..., f[x_0, ..., x_n]]$

Function `differenceQuotients(x,f)`:
- $n \leftarrow size(x)$
- quotients $\leftarrow$ []
- quotients.append($\mathbf{f}[0]$)
- **for** i *from* 1 *to* n **do**
  - **for** j *from* 0 *to* $n - i$ **do**
    - $\mathbf{f}[j] \leftarrow \frac{f[j]-f[j+1]}{x[j]-x[j+i]}$
  - **end**
  - quotients.append($f[0]$)
- **end**
- **return** (quotients)

Obliczanie wartości funkcji w postaci Newtona

Dla przypomnienia, weźmy postać wielomianu w postaci Newtona, wygląda on tak:

$$N_n(x) = \sum_{k=0}^{n} f[x_0, ..., x_k] \prod_{i=0}^{k-1} (x - x_i)$$

Podobnie jak w Algorytmie Hornera gdzie zauważyliśmy, że wielomian

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_0 x^0$$

Można zapisać w postaci

$$p(x) = (...(((a_n x + a_{n-1})x + a_{n-2})x + ...)...)x + a_0$$

Ponieważ po wymnożeniu odpowiednich wyrazów otrzymujemy spowrotem wielomian w postaci naturalnej stwierdzamy, że oba zapisy reprezentują ten sam wielomian.
W uogólnionym algorytmie Hornera rozważamy wielomian, w którym mamy niektóre x zamienione na $x \to (x - x_k)$ dla różnych k. Stąd wniosek aby rozważyć nastepujący wielomian

$$N(x) = \Bigg(... \Bigg(\bigg(\Big(c_n(x - x_{n-1}) + c_{n-1} \Big)(x - x_{n-2}) + c_{n-2} \bigg)(x - x_{n-3}) + c_{n-3} \Bigg) ... \Bigg)(x - x_0) + c_0$$

Po wymnożeniu odpowiednich wyrazów (przy każdym c_k mamy $\prod_{t=0}^{k-1}(x - x_t)$) okazuje się że $N(x)$ to dokładnie wielomian

$$N_n(x) = \sum_{k=0}^{n} f[x_0, ..., x_k] \prod_{i=0}^{k-1}(x - x_i)$$

stąd wynika procedura

Algorithm 2: warNewton

Output: $N_n(t)$

Function `ilorazyRoznicowe(`$nodes = \mathbf{x}$, $quotients = \mathbf{f}$, $argument = t$`):`

$n \leftarrow size(x) - 1$

result $\leftarrow$ f[n+1]

for k *from* n *to* 1 *step* -1 **do**

  result = result $\cdot (t - \mathbf{x}[k]) + \mathbf{f}[k]$

end

return result

Mapa z współczynników z bazy $\{q_n : q_n(x) = \prod_{k=0}^{n-1}(x - x_k)\}$ na współczynikki w bazie $\{q_n : q_n(x) = x^n\}$

Zauważmy pewien fakt wynikający wprost z poprzednich rozważań. Mianowicie możemy przedstawić $N_n(x)$ za pomocą wielomianow w_n którye zdefiniujemy rekursyjnie

$$w_{n+1} = 0,$$

$$w_k(x) = f[x_0, x_1, ..., x_k] + (x - x_k) \cdot (w_{k+1}(x))$$

wtedy

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

Z tej postaci widać że obliczając w_{k+1} i zapamiętując jego współczynniki możemy obliczyć w_k przechodząc przez wszystkie zapamiętane współczynniki. Jest ich co najwyżej n oraz aby obliczyć współczynniki w_0 potrzebujemy obliczyć wszystkie poprzednie wielomiany których jest n stąd możemy dokonać tego w czasie $O(n^2)$.
stąd wynika procedura o złożoności $O(n^2)$:

Algorithm 3: naturalna

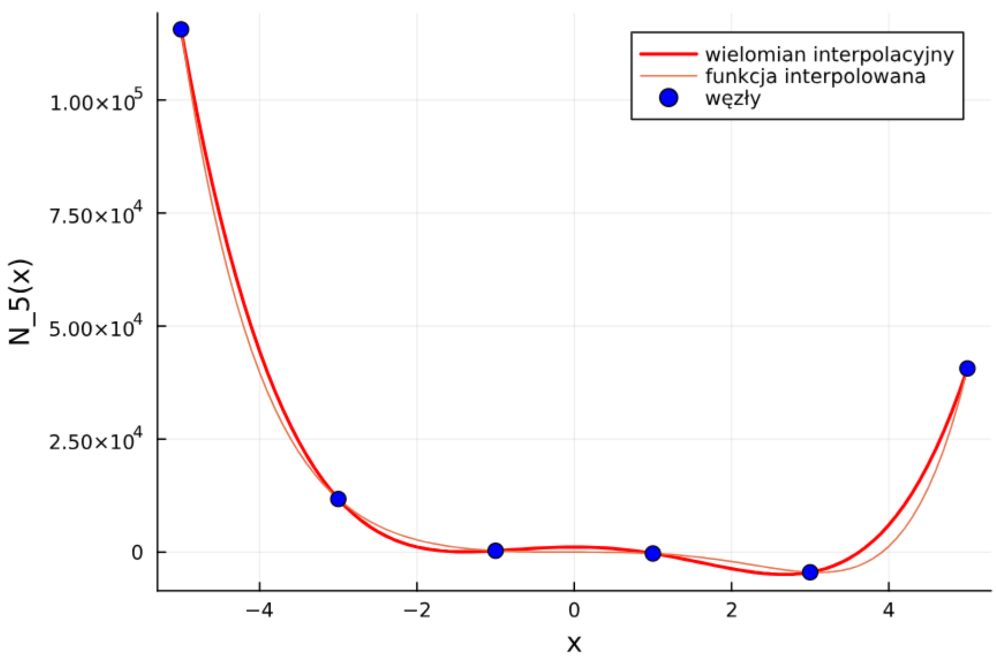
Output: $[a_0, a_1, ..., a_n]$

Function `naturalna`($nodes = \mathbf{x}$, $quotients = \mathbf{f}$)**:**

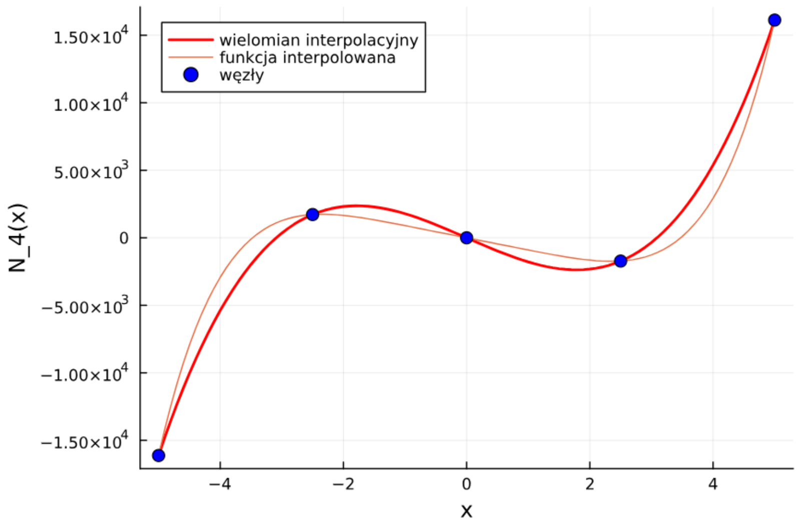
```
n ← size(x) − 1
coeffs ← [0,0,0,0...,0]
for i from n to 0 step −1 do
    for j from i to n − 1 do
        coeffs[j] = coeffs[j+1] - x[i]
    end
    coeffs[n] = f[i] - coeffs[n] * x[i]
end
return coeffs
```

funkcja rysująca

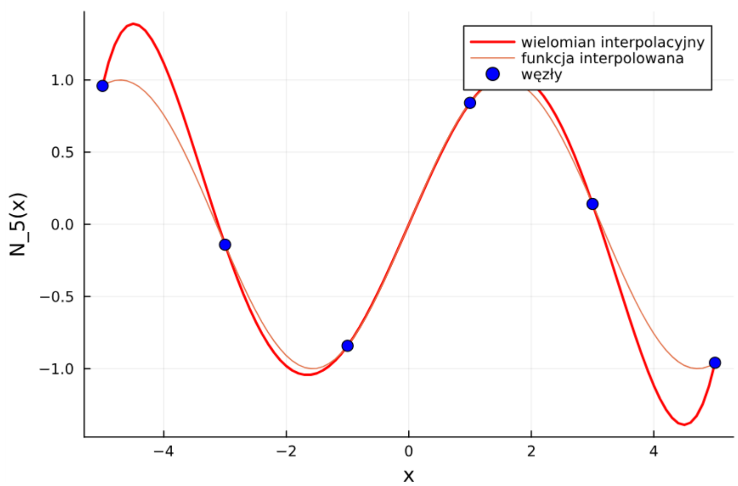
W kolejnym zadaniu tworzymy funkcje intrepolującą f rysującą wykres otrzymanego wielomianu interpolacyjnego. Funkcja korzysta z poprzednich i produkuje przykładowe wyniki:



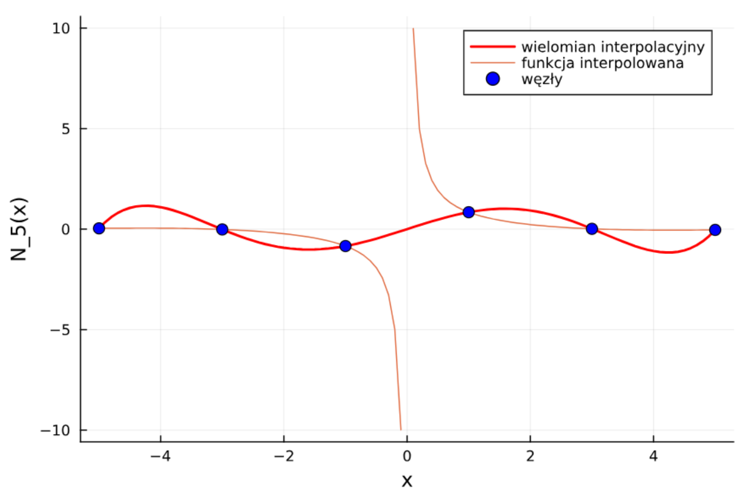
Rysunek 1: $5x^6 - 300x^3$



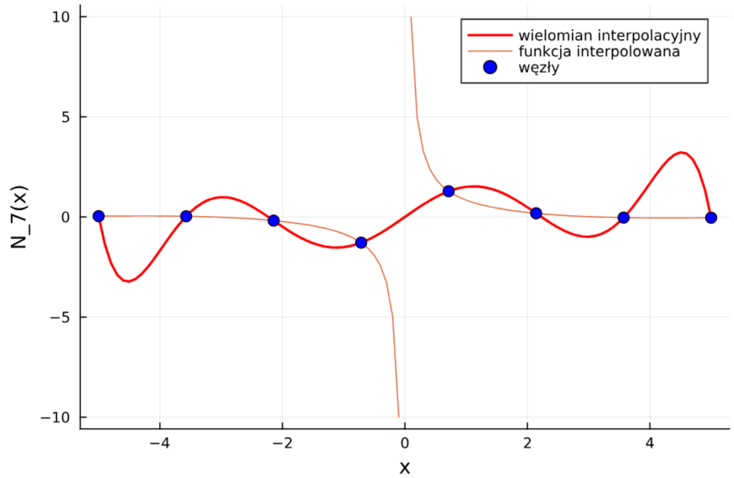
Rysunek 2: $7x^5 - 10x^3 - 900x$



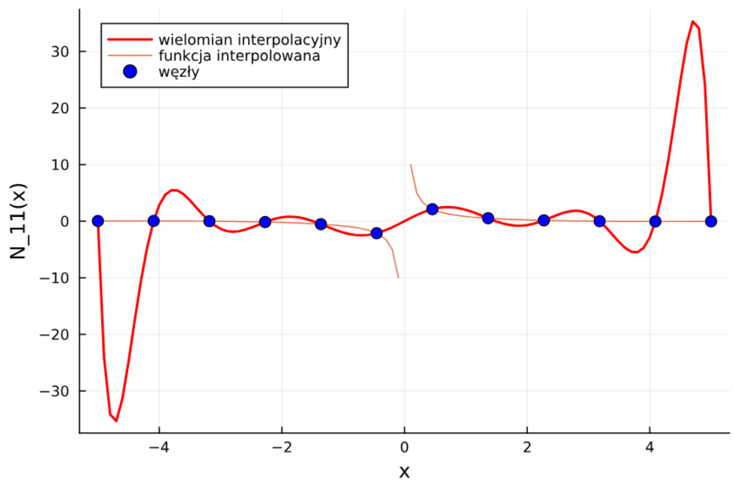
Rysunek 3: $sin(x)$



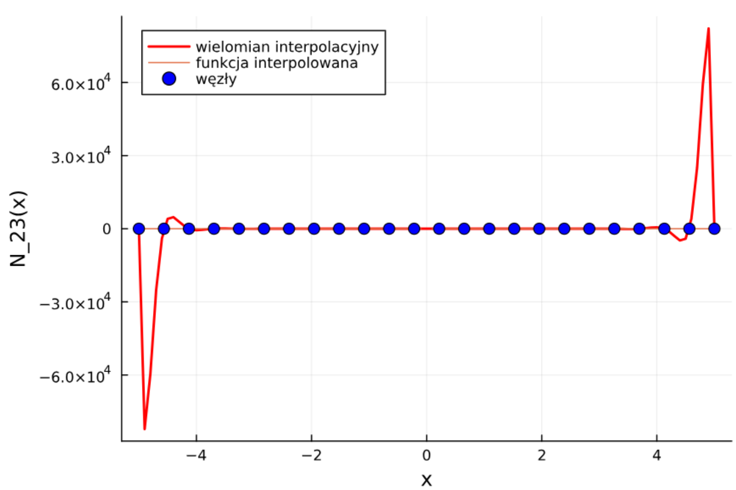
Rysunek 4: $\frac{sin(x)}{x^2}$



Rysunek 5: $\frac{sin(x)}{x^2}$



Rysunek 6: $\frac{sin(x)}{x^2}$

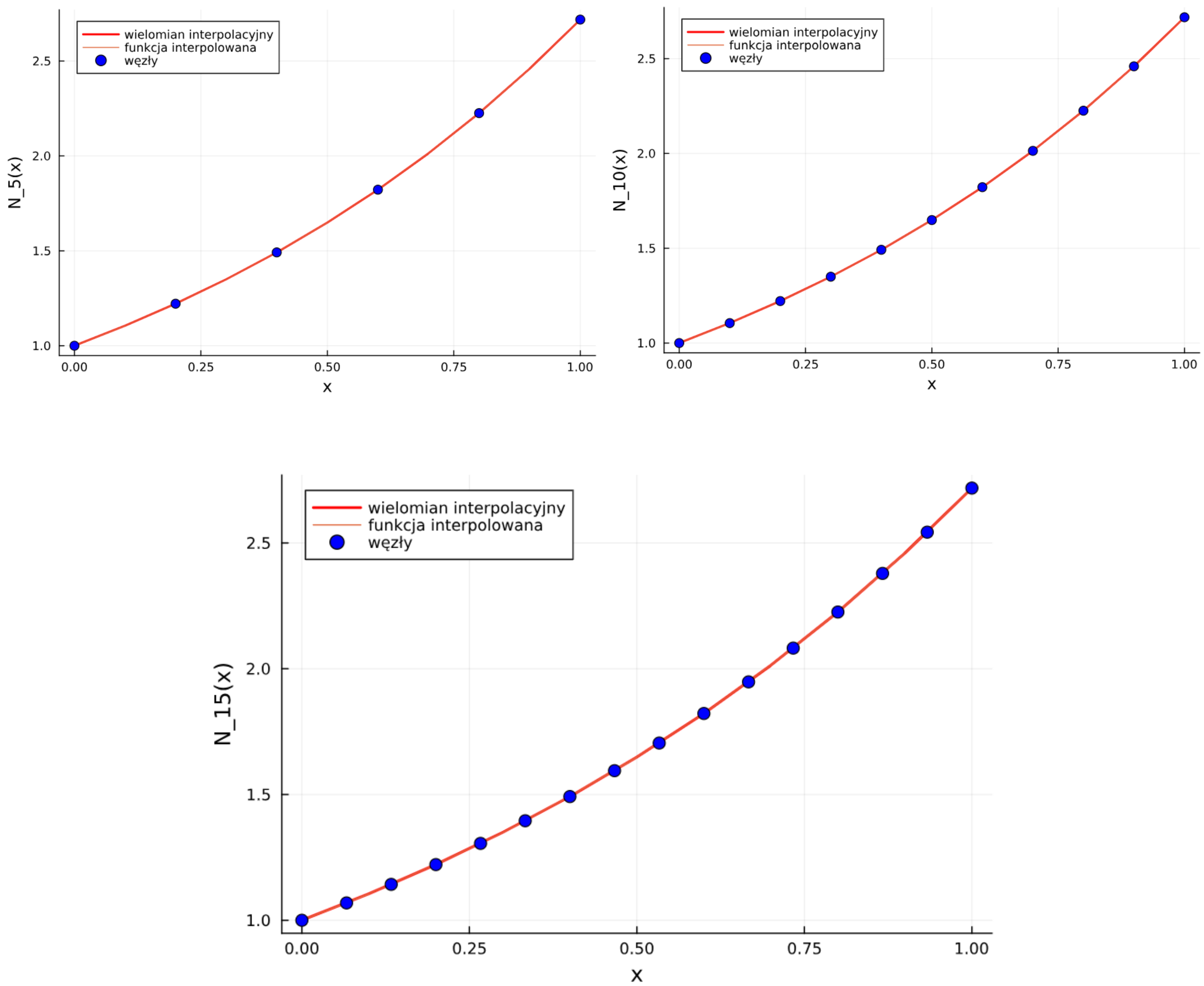


Rysunek 7: $\frac{sin(x)}{x^2}$

zadanie 5

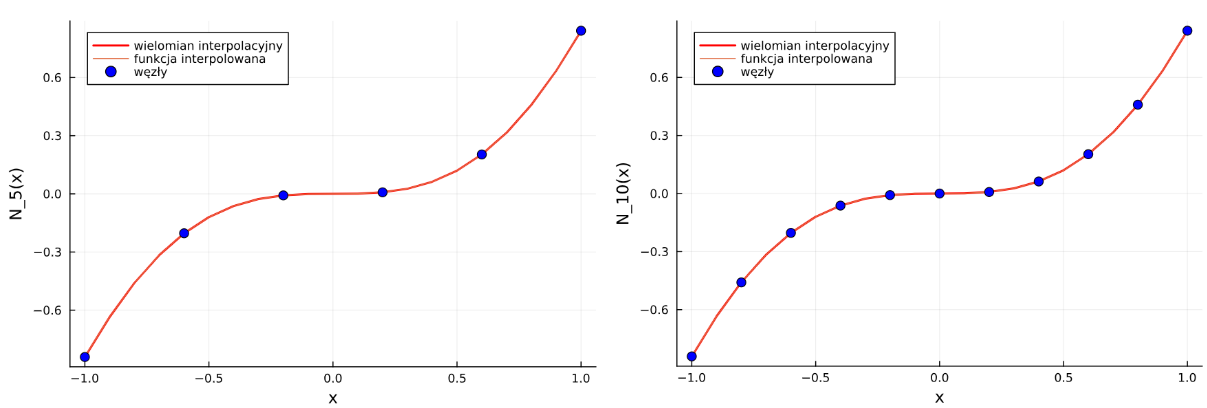
Rysujemy wykresy dla kolejno:

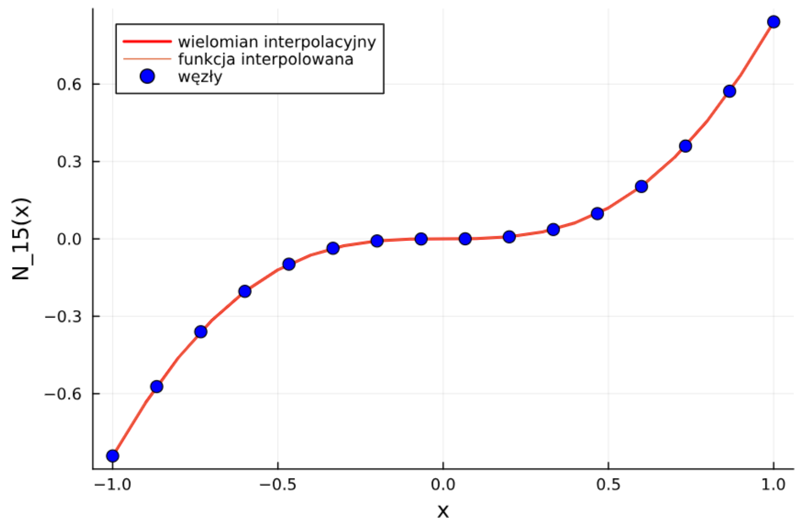
- e^x



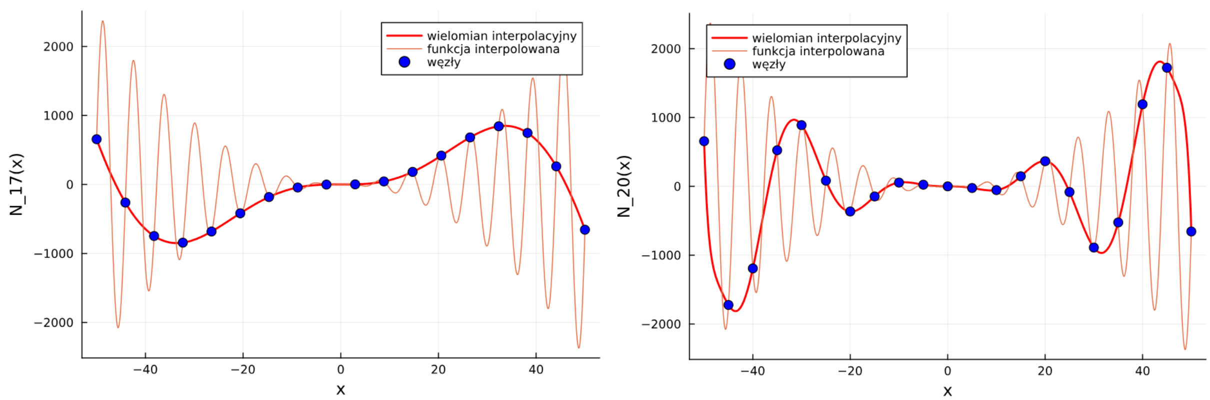
Rysunek 8: e^x

- $x^2 sin(x)$





Rysunek 9: $x^2 sin(x)$



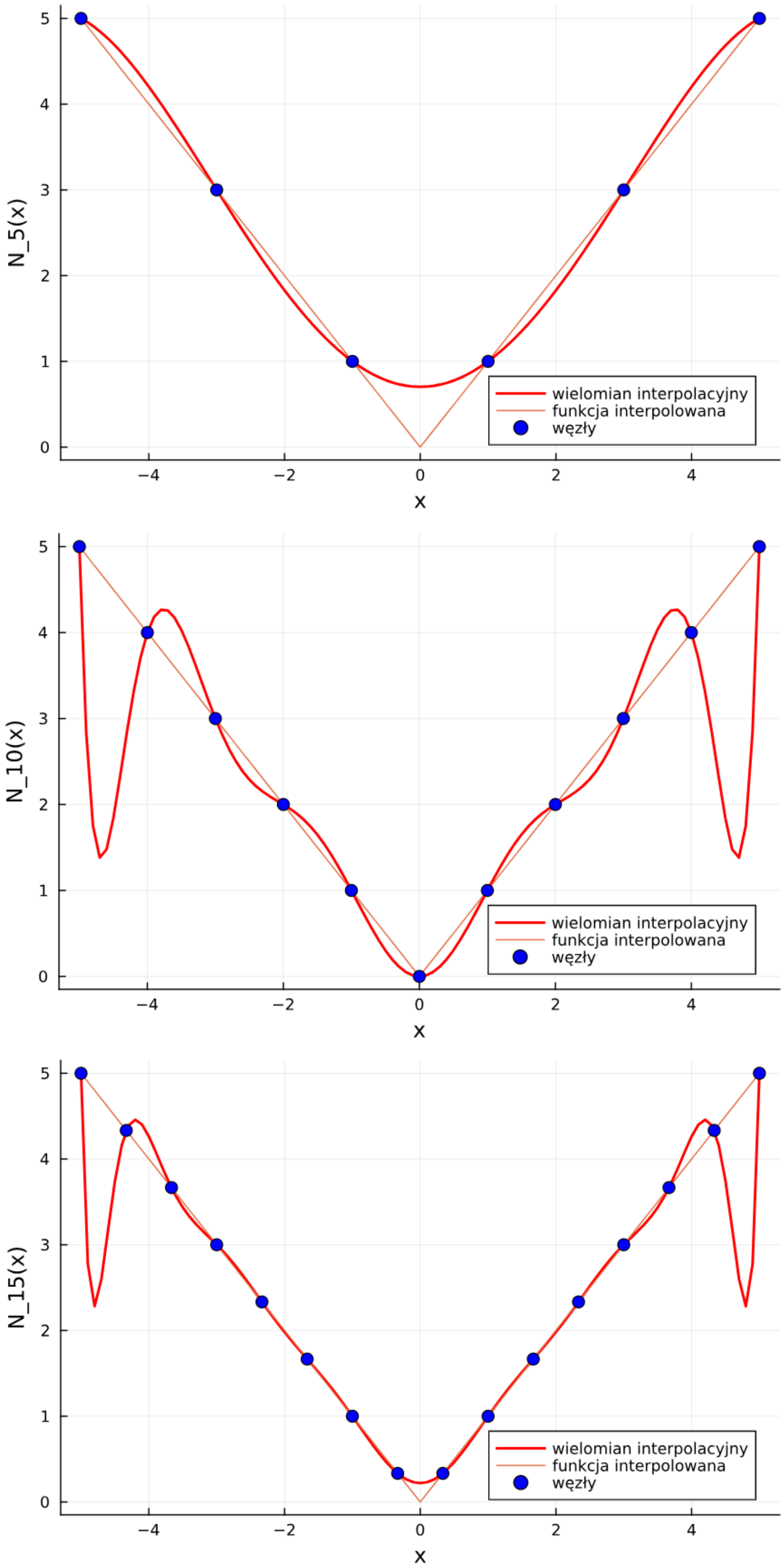
Rysunek 10: $x^2 sin(x)$ dla szerszego przedziału

Zadanie 6

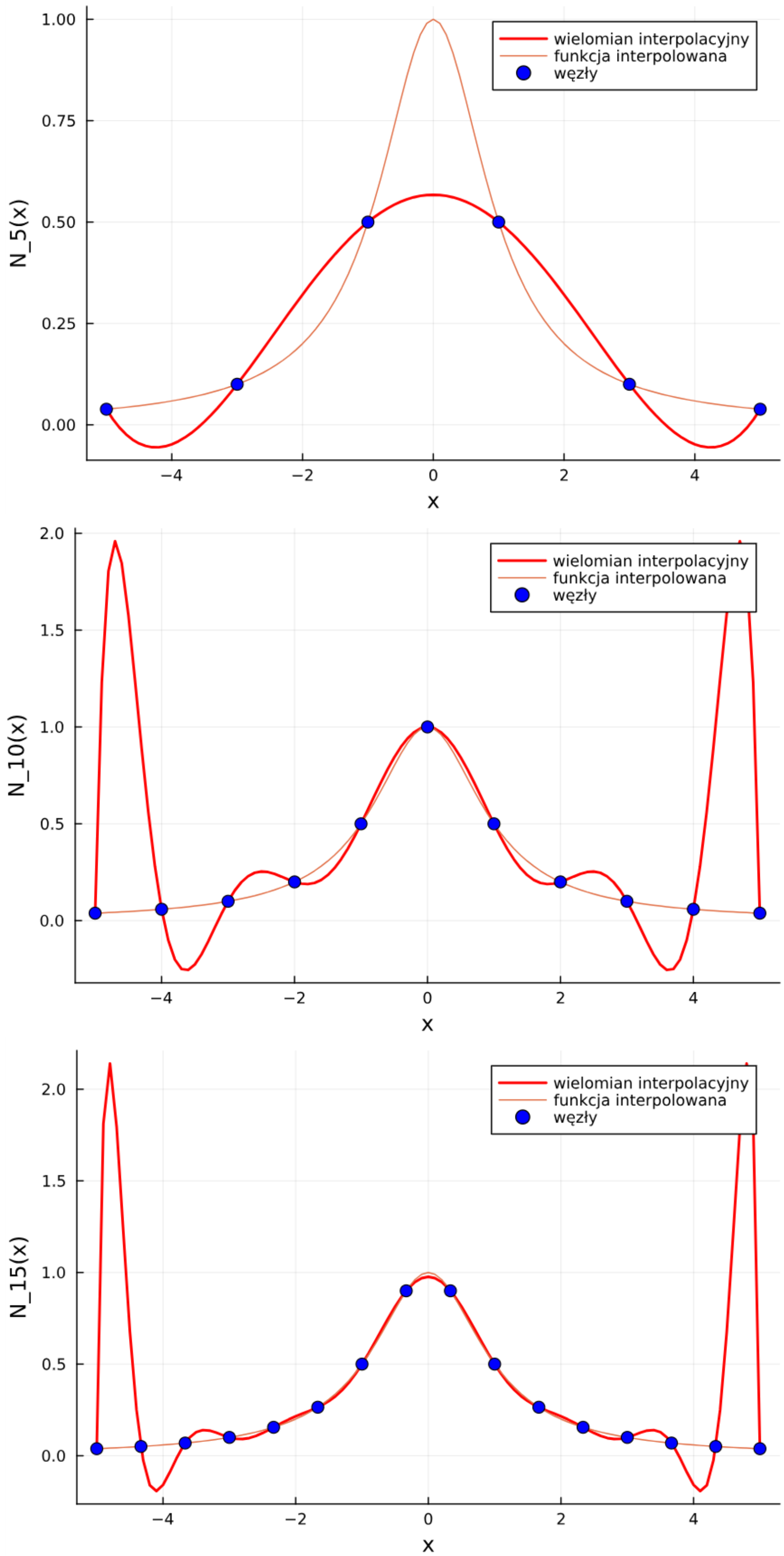
Rysujemy wykresy dla kolejno:

- $|x|$
- $\frac{1}{1+x^2}$

Zauważmy, że pierwsza funkcja nie jest nawet ciągła, stąd przed eksperymentem mogliśmy spodziewać się że nie uda nam się przybliżyć ją dowolnie dobrze wielomianem. Mimo niejednoznacznego trędu (błedy rosną i maleją) na wykresach, to próbując dokładać n np. $n = 25$ zauwazymy że wartości na brzegach przedziałów bardzo szybko rosną. Jedynym sposobem na poprawę tego zjawiska jest lepszy dobór węzłów w których interpolujemy funkcje.


N_5(x)
N_10(x)
N_15(x)
x
wielomian interpolacyjny
funkcja interpolowana
węzły

Rysunek 11: $|x|$



Rysunek 12: $\frac{1}{1+x^2}$