OBLICZENIA NAUKOWE: Lista nr 4

Niedziela, 10 Grudnia 2017

Tymoteusz Surynt

Numer indeksu: 229794

Podsumowanie

Zadanie 1

Opis problemu

Celem zadania było stworzenie funkcji, która będzie w stanie wyliczać ilorazy różnicowe dla zadanej funkcji.

Opis rozwiązania

Dane:
x - tablica zawierająca wartości węzłów $x_0, x_1, x_2, \ldots$
f - tablica zawierająca wartości funkcji $f(x_0), f(x_1), \ldots$
Wynik: fx – wektor długości $n+1$ zawierający obliczone ilorazy różnicowe
Funkcja: ilorazyRoznicowe(x::VectorFloat64, f::VectorFloat64)

```
size ← length(x);
output[size];
for i in 1 : size do
    output[i] ← f[i];
end
for i in 1 : size do
    j ← size;
    while j > i do
        output[j] ← (output[j]−output[j−1]) / (x[j]−x[j−i]);
        j ← j − 1;
    end
end
return output
```

Algorithm 1: Ilorazy Różnicowe

Powyższy algorytm opiera się na własności iloczynu różnicowego: $f[x_0, x_1, x_2, \ldots, x_n] = \frac{f[x_1,x_2,\ldots,x_n]-f[x_0,x_1,\ldots,x_{n-1}]}{x_n-x_0}$ z której łatwo da się zauważyć, że kolejny iloczyn można szybko wyliczyć mając poprzedni. Jako, że wszystkie iloczyny są zapisywane w tablicy, dostęp do poprzednich nie jest problemem, jednak każda zmiana przy wyliczeniu iloczynu powoduje szereg zmian w następnych iloczynch. Żeby to poprawić występuje druga pętla, while, które umożliwia poprawianie wyników. Zmiany istnieją, ponieważ $f[x_1, x_2, \ldots, x_n]$ jest wyliczane na bieżąco.

Testy

Testy dla powyższej funkcji były przeprowadzane w następujący sposób: wartości uzyskane przez funkcje były porównywane do prawdziwych, pamiętając o dokładności obliczeń, która wynosiła: 10.0^{-5}. Aby zdać test, funkcja musiała być w odległości dokładności od prawdziwego wyniku.

1. Test 1:

$$ilorazyRoznicowe([1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0], [-1.0, -5.0, 2.0, 3.0, 0.3, 0.2, 0.6])$$

   Wynik właściwy:

$$[-1.0, -4.0, 5.5, -2.83333, 0.804167, -0.1275, 0.00402778]$$

   Wynik otrzymany:

$$[-1.0, -4.0, 5.5, -2.83333, 0.804167, -0.1275, 0.00402778]$$

   Test zdany

2. Test 2:

   $ilorazyRoznicowe([1.0, 10.0, 20.0, 30.0, 40.0, 50.0, 60.0], [-1.202, 111.0, -4.44122, 12.0, -0.3222113, 0.15, 10.3])$

   Wynik właściwy:

   $$[-1.202, 12.4668888889, -1.26374, 0.0663155, -0.00238692, 0.0000659068, 0]$$

   Wynik otrzymany:

   $$[-1.202, 12.4669, -1.26374, 0.0663155, -0.00238692, 6.59068e-5, -1.46576e-6]$$

   Test zdany

3. Test 3:

   $$ilorazyRoznicowe([3.0, 1.0, 5.0, 6.0], [1.0, -3.0, 2.0, 4.0])$$

   Wynik właściwy:

   $$[1.0, 2.0, -0.375, 0.175]$$

   Wynik otrzymany:

   $$[1.0, 2.0, -0.375, 0.175]$$

   Test zdany

Zadanie 2

Opis problemu

Celem zadania było stworzenie funkcji, która będzie w stanie wyliczać wartości wielomianu interpolacyjnego n-tego stopnia w postaci Newtona, dla podanego punktu.

Opis rozwiązania

Dane:
x - tablica zawierająca wartości węzłów $x_0, x_1, x_2, \ldots$
fx - tablica zawierająca wartości ilorazów różnicowych $f(x_0), f(x_0, x_1), \ldots$
t - punkt dla którego liczymy wartość wielomianu
Wynik: nt – wartość wielomianu w punkcie t
Funkcja: warNewton (x::VectorFloat64, fx::VectorFloat64, t::Float64)
$size \leftarrow length(x)$;
$reszta \leftarrow fx[size]$;
$i \leftarrow size - 1$;
while $i > 0$ **do**
    $reszta \leftarrow reszta * (t - x[i]) + fx[i]$;
    $i \leftarrow i - 1$;
end
return $reszta$

Algorithm 2: Algorytm Hornera dla wielomianu w postaci Newton

Powyższy algorytm działa w myśl uogólnionego algorytmu Hornera, jednak uwzględnia postać Newtona. Pierwsza wartość (w tym wypadku znajduje się na końcu tablicy, ponieważ ta postać zakłada, że największa potęga znajduje się na końcu) zostaje bez zmian, kolejne uzyskuje się przez dodanie do współczynika z oryginalnej funckji, poprzedniej wartości pomnożonej przez różnicę miejsca w którym szukamy wartości i wartości x_i (dla normlanego wielomianu byłoby po prostu t). Ostatnia wartość jak i w algorytmie Hornera jest poszukiawną wartością.

Testy

Testy dla powyższej funkcji były przeprowadzane w następujący sposób: wartości uzyskane przez funkcje były porównywane do prawdziwych, pamiętając o dokładności obliczeń, która wynosiła: 10.0^{-5}. Aby zdać test, funkcja musiała być w odległości dokładności od prawdziwego wyniku.

1. Test 1:
$$warNewton([1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0], [-1.0, -5.0, 2.0, 3.0, 0.3, 0.2, 0.6], 3.0)$$
   Wynik właściwy:
$$-7.0$$
   Wynik otrzymany:
$$-7.0$$
   Test zdany
2. Test 2:
$$warNewton([1.0, 10.0, 20.0, 30.0, 40.0, 50.0, 60.0], [-1.202, 111.0, -4.44122, 12.0, -0.3222113, 0.15, 10.3], 5.0)$$
   Wynik właściwy:
$$-121622826.79285003$$
   Wynik otrzymany:
$$-1.2162282679285003e8$$
   Test zdany
3. Test 3:
$$warNewton([3.0, 1.0, 5.0, 6.0], [1.0, -3.0, 2.0, 4.0], 3.0)$$
   Wynik właściwy:
$$1.0$$
   Wynik otrzymany:
$$1.0$$
   Test zdany

Zadanie 3

Opis problemu

Celem zadania było stworzenie funkcji, która będzie w stanie znajdować współczyniki wielomianu interpolacyjnego w postaci Newtona.

Opis rozwiązania

Dane:
x - tablica zawierająca wartości węzłów $x_0, x_1, x_2, \ldots$
fx - tablica zawierająca wartości ilorazów różnicowych $f(x_0), f(x_0, x_1), \ldots$
Wynik: a - wektor zawierający obliczone współczynniki postaci naturalnej
Funkcja: naturalna (x::VectorFloat64, fx::VectorFloat64)
$size \leftarrow length(x)$;
$a[size] \leftarrow fx[size]$;
$i \leftarrow size - 1$;
while $i > 0$ **do**
  $a[i] \leftarrow fx[i]$;
  for j *in* $i : size - 1$ **do**
    $a[j] \leftarrow a[j] - x[i] * a[j+1]$;
  end
  $i \leftarrow i - 1$;
end
return a

Algorithm 3: Współczynniki postaci naturalnej

Testy

Testy dla powyższej funkcji były przeprowadzane w następujący sposób: wartości uzyskane przez funkcje były porównywane do prawdziwych, pamiętając o dokładności obliczeń, która wynosiła: 10.0^{-5}. Aby zdać test, funkcja musiała być w odległości dokładności od prawdziwego wyniku.

1. Test 1:
$$naturalna([3.0, 1.0, 5.0, 6.0], [1.0, 2.0, -0.375, 0.175])$$
Wynik właściwy:
$$[-8.75, 7.525, -1.95, 0.175]$$
Wynik otrzymany:
$$[-8.75, 7.525, -1.95, 0.175]$$
Test zdany

2. Test 2:
$$naturalna([1.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0], [-1.2, 1.0, -4.2, 1.0, -0.33, 0.5, 10.3])$$
Wynik właściwy:
$$[1213.62, -4001.19, 5083.39, -3167.19, 1024.17, -164.3, 10.3]$$
Wynik otrzymany:
$$[1213.62, -4001.19, 5083.39, -3167.19, 1024.17, -164.3, 10.3]$$
Test zdany

3. Test 3:
$$naturalna([1.0, 7.0, 2.0, 0.20, 1.0, 0.022, 0.02], [-1.2, 0.20, -3.2, 11.0, -2.33, 0.9, 1.7])$$

Wynik właściwy:

$$[-186.73928, 335.83764, -172.67936, -8.99048, 47.84888, -18.1774, 1.7]$$

Wynik otrzymany:

$$[-186.73928, 335.83764, -172.67936, -8.99048, 47.84888, -18.1774, 1.7]$$

Test zdany

Zadanie 4

Opis problemu

Celem zadania było stworzenie funkcji, która będzie w stanie rysować wykresy dla wielomianu interpolacyjnego i podanej funkcji.

Opis rozwiązania

Dane:
f - funkcja dla, której rysujemy wykres
a, b - przedział interpolacji
n - stopień wielomianu interpolacyjnego
[GlobalPrec- precyzja wykresu (zmienna globalna, standardowo ustawiona na 100)]
Wynik: Wykres z zaznaczonymi funkcjiami
Funkcja: rysujNnfx(f,a::Float64,b::Float64,n::Int)
$prec \leftarrow \frac{b-a}{n}$;
$x[n+1]$;
$fx[n+1]$;
for i *in* $0 : n$ **do**
  $x[i+1] \leftarrow a + i * prec$;
  $fx[i+1] \leftarrow f(x[i+1])$;
end
$fn \leftarrow ilorazyRoznicowe(x, fx)$;
$outputFun[GlobalPrec]$;
$outputInt[GlobalPrec]$;
$array[GlobalPrec]$;
$prec \leftarrow \frac{b-a}{GlobalPrec}$;
for i *in* $0 : GlobalPrec$ **do**
  $t \leftarrow a + i * prec$;
  $outputInt[i+1] \leftarrow warNewton(x, fn, t)$;
  $outputFun[i+1] \leftarrow f(t)$;
  $array[i+1] \leftarrow t$;
end
plot(array,[outputFun , outputInt], label=["Wynik dla funkcji" "Wynik dla interpolacji"])

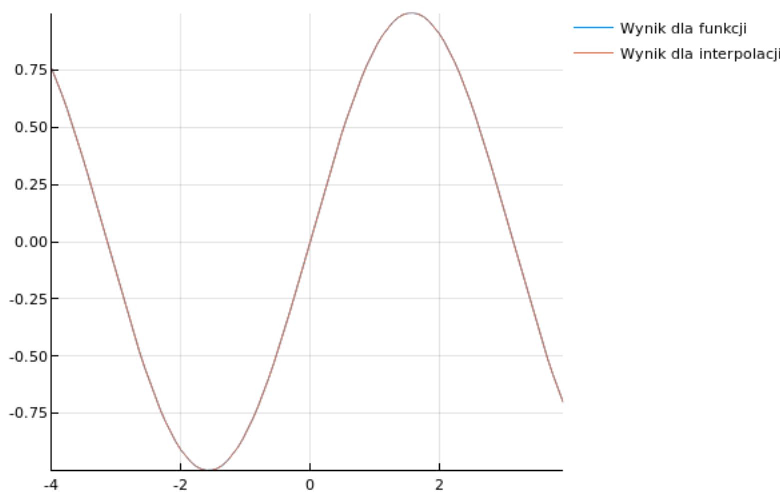
Algorithm 4: Rysowanie wykresów funkcji

Powyższa funkcja w pierwszej kolejności dzieli przedział na ilość węzłów interpolacji aby punkty były równomiernie rozrzucone i wylicza $x_0, x_1, \ldots$ oraz wartości funkcji w tych punktach. Dla wcześniej obliczonych danych wykonuje fukcję opisaną w Zadaniu 1, która oblicza ilorazy różnicowe. Następnie przygotowuje dwie tablice w których będą zapisywane wyniki pod wyświetlanie ich na wykresie. Po wyliczeniu kroku co który będą wyliczane wartości, wykonuje się pętla w której owe wartości są wyliczane. Dla wielomianu interplacyjnego są one wyliczane funcją z Zadania 2. Na sam koniec uzyskane wyniki są prezentowane na wykresie używając pakietu Plots i Plotly.

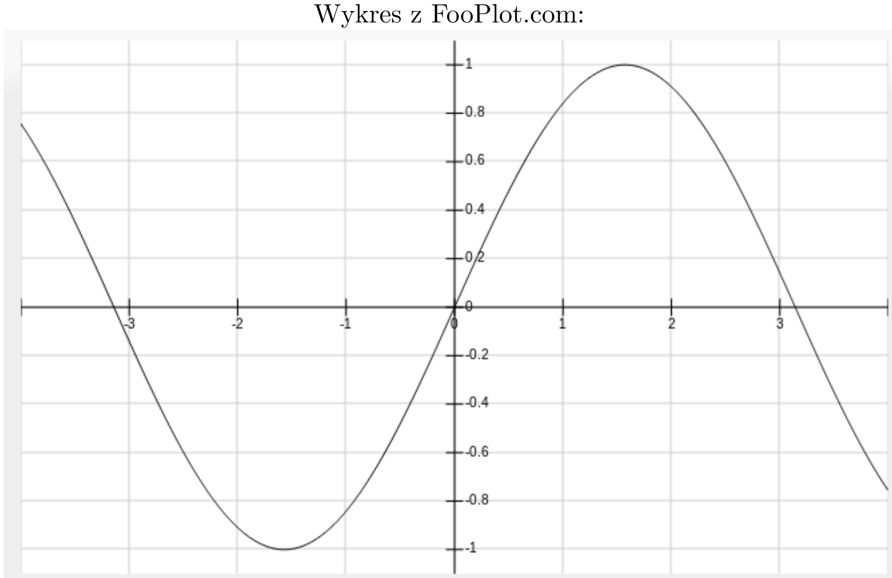
Testy

W tym przypadku przeprowadzenie testów było bardzo trudne, a odpowiednie testy na bardziej zaawanoswoanych przykładach będą przeprowadzane w zadaniu 5 i 6, więc w tej części porównam tylko czy wykres funkcji (nie wielomianu) podanej przez naszą funkcje będzie podobny do takiego narysowanego przez niezależny program do generownania wykresów.

Wykres wykonany przez naszą funkcję:



Wykres z FooPlot.com:



Widzimy, że wykresy są podobne, więc nasza funkcja poprawnie rysuje wykresy funkcji.

Zadanie 5

Opis problemu

Zadanie polegało na narysowaniu wykresów poprzez skorzystanie z funkcji napisanej w Zadaniu 4 dla podanych przypadków:

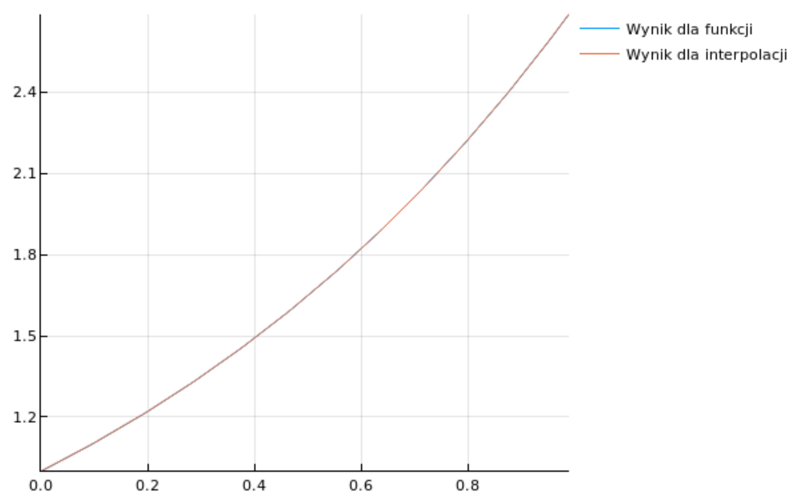
1. e^x, [0,1], n= 5, 10, 15
2. $x^2 sin(x)$, [-1,1], n= 5, 10, 15

Opis rozwiązania

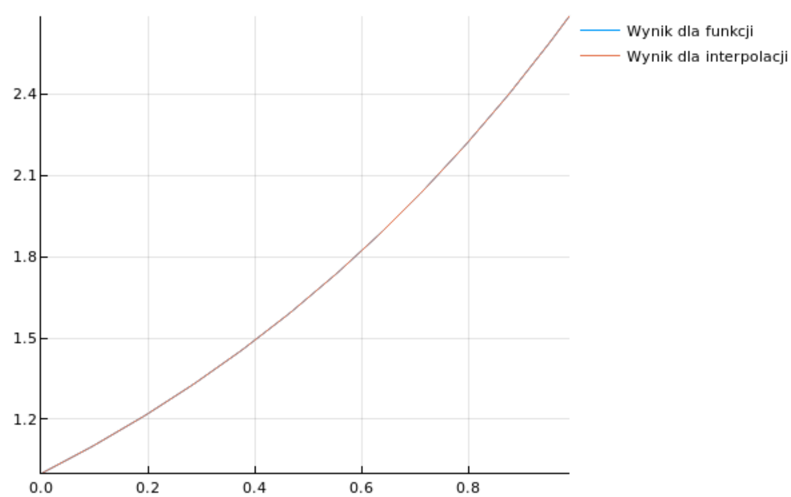
Korzystam z uprzednio napisanej funckji poprzez uruchomienie jej z podanymi parametrami.

Wyniki

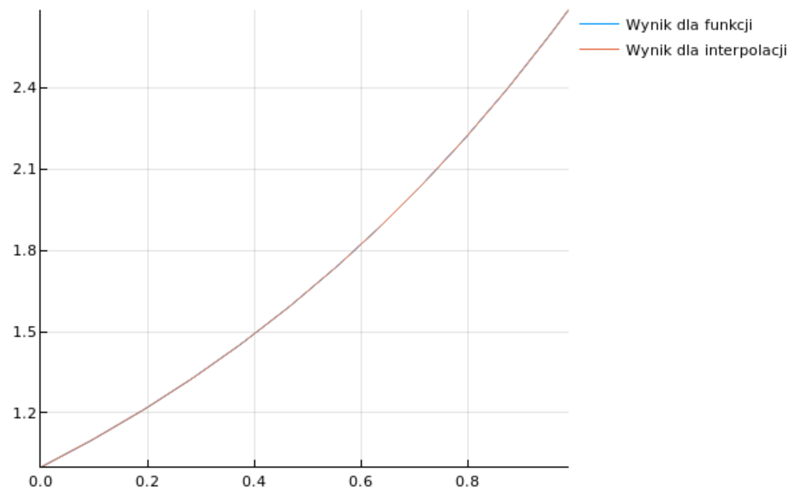
Wynik dla podpunktu 1. z n=5



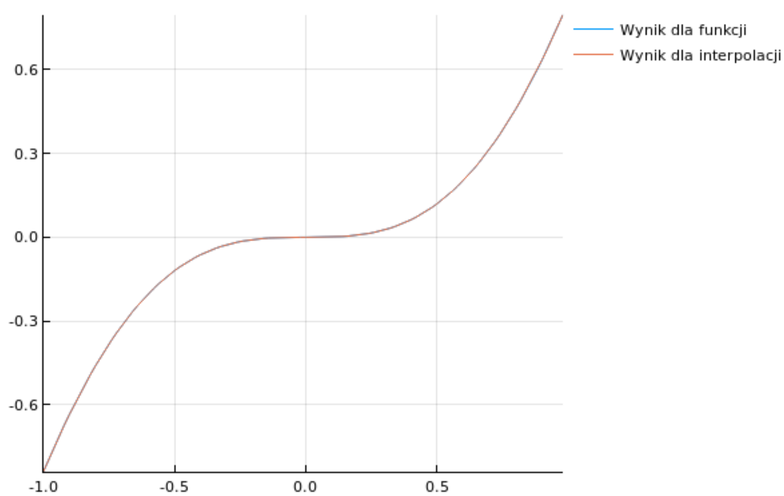
Wynik dla podpunktu 1. z n=10



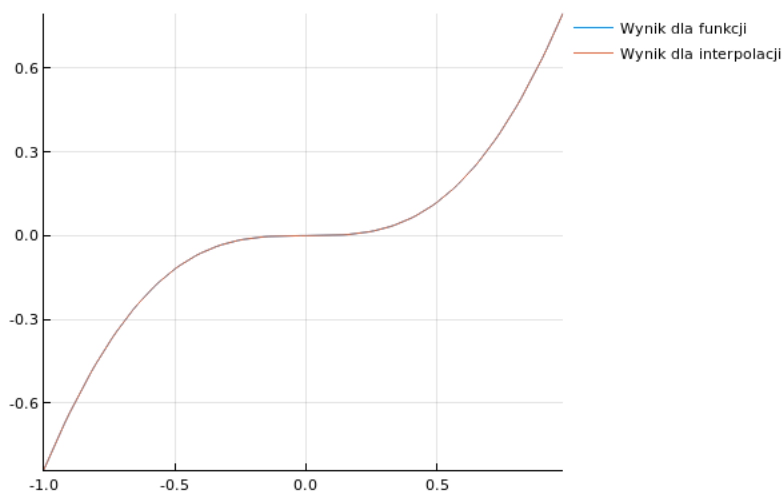
Wynik dla podpunktu 1. z n=15



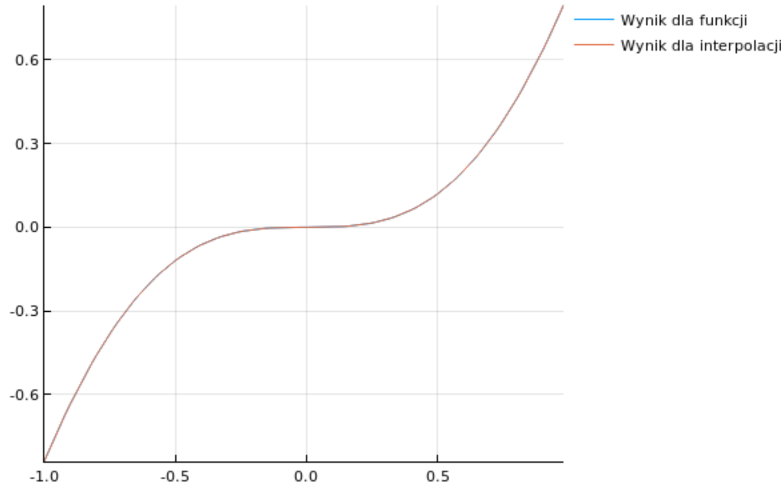
Wynik dla podpunktu 2. z n=5



Wynik dla podpunktu 2. z n=10

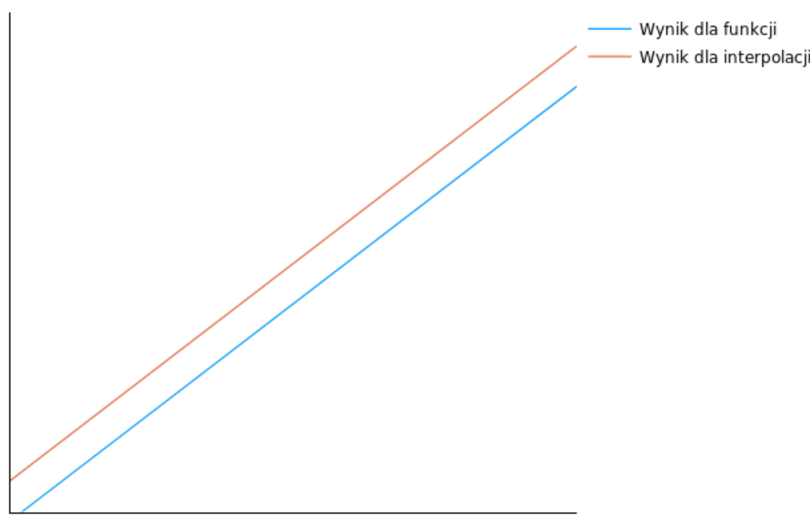


Wynik dla podpunktu 2. z n=15



Wnioski

Łatwo da się zauważyć, że oba wykresy funkcji są do siebie bardzo zbliżone i nawet jeśli różnice istnieją to są ona pomijalnie miałe.



Jeśli bardzo przybliżymy to zauważymy pewne różnice i wraz ze wzrostem n są one coraz mniejsze.

Zadanie 6

Opis problemu

Zadanie polegało na narysowaniu wykresów poprzez skorzystanie z funkcji napisanej w Zadaniu 4 dla podanych przypadków:

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

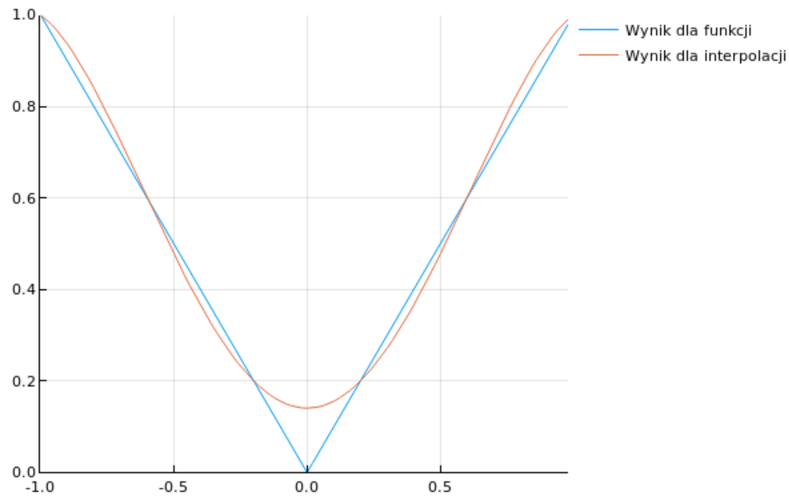
2. $\frac{1}{1+x^2}$, [-1,1], n= 5, 10, 15

Opis rozwiązania

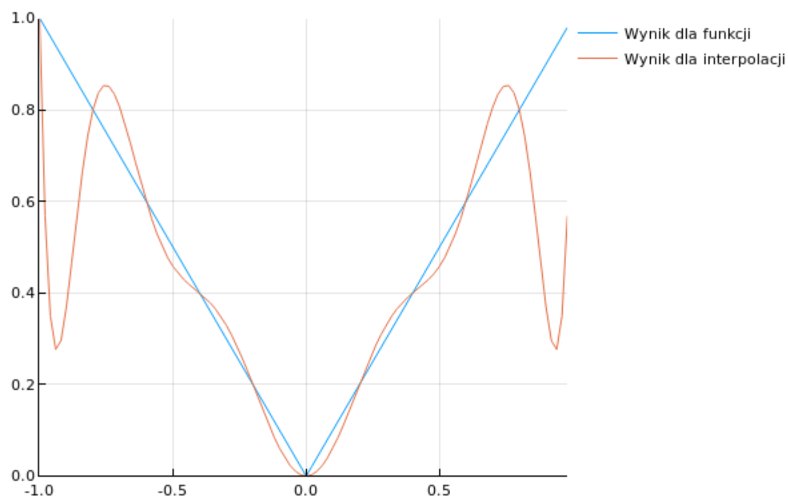
Korzystam z uprzednio napisanej funckji poprzez uruchomienie jej z podanymi parametrami.

Wyniki

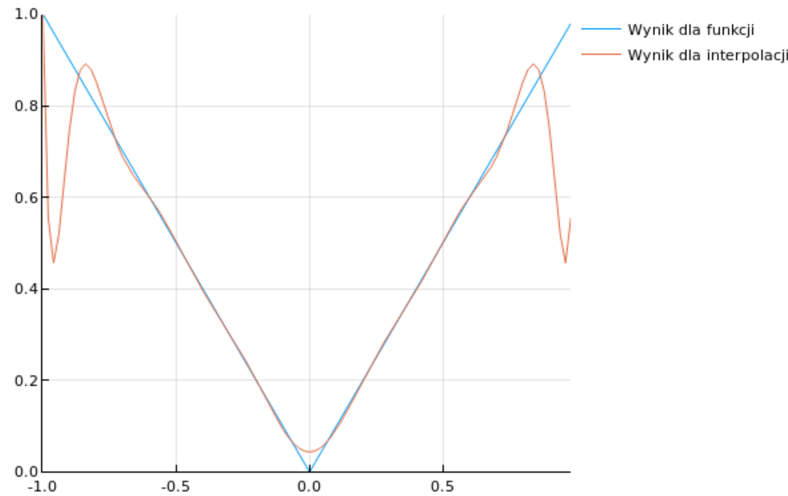
Wynik dla podpunktu 1. z n=5



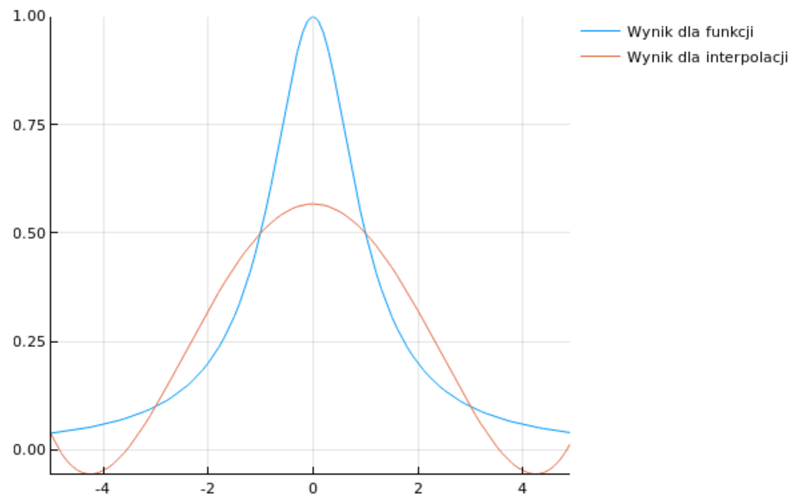
Wynik dla podpunktu 1. z n=10



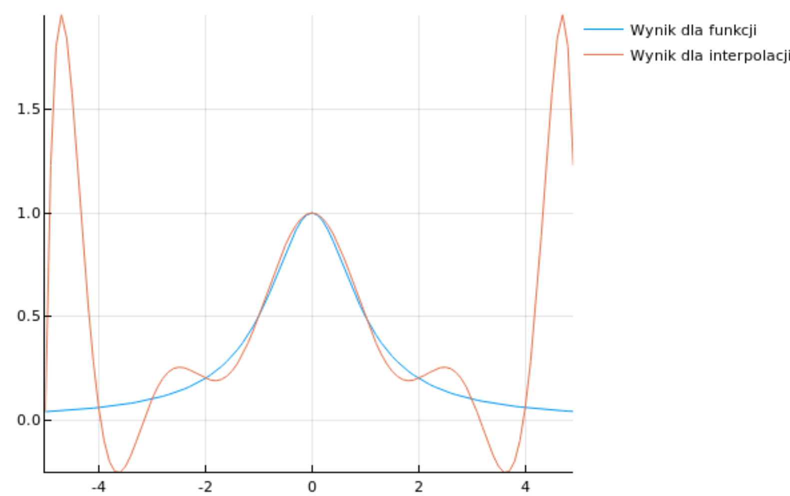
Wynik dla podpunktu 1. z n=15



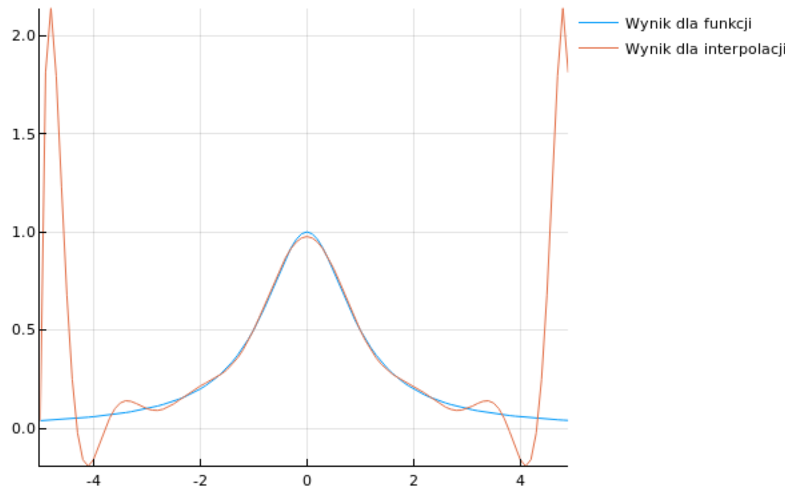
Wynik dla podpunktu 2. z n=5



Wynik dla podpunktu 2. z n=10



Wynik dla podpunktu 2. z n=15



Wnioski

W pierwszym jak i w drugim przypadku, łatwo da się zauważyć, że niedokładność wielomianu interpolacyjnego są dość znaczące i rosną wraz z n. Wynika to z faktu, że węzły są rozłożone równomiernie i na końcach przedziału zachodzi efekt Rungr'go. Przez to da się zaobserwować niechcianą "falę". Aby uniknąć tego problemu należałoby zwiększyć ilość węzłów na krańcach przedziału, sprawiając, że węzły nie byłyby już równoodległe od siebie. Do wyznaczania takich węzłów mogłby nam posłużyć zera wielomianu Czebyszewa.