Obliczenia naukowe
Lista 4

Stanisław Woźniak

1 Zadanie 1.

1.1 Ilorazy różnicowe

1.2 Opis

Iloraz różnicowy jest to wartość pokazująca stosunek różnic wartości do argumentów. Jest on używany do badania przyrostu funkcji w danych punktach i najczesciej jest używany wzorem: $\frac{f(x+h)-f(x)}{h}$ gdzie h jest różnicą między wybranymi punktami co przy wersji z indeksami można zastapić wzorem $\frac{f(x_i)-f(x_j)}{x_j-xi}$ gdzie $i \neq j$. Natomiast uogólniony wzór:

$$f[x_0, ..., x_n] = \sum_{i=0}^{n} \frac{f(x_i}{\prod_{j=0j\neq i}^{n}(x_i - x_j)}$$

Każde kolejne ilorazy różnicowe można generować ze wzoru rekurencyjnego:

$$f[x_k] = f(x_k)$$

$$f[x_i, ..., x_j] = \frac{f[x_{i+1}, .., x_j] - f[x_i, .., x_{j-1}]}{x_j - x_i} \text{ gdzie } (i \neq j)$$

Zadanie polegało na zaimplementowaniu funkcji liczącej ilorazy różnicowe mając dane węzły oraz wartości funkcji w tych węzłach. Zadanie miało zostać zrealizowane bez użycia macierzy. W implementacji zostało użyte liczenie kolejnych ilorazów zgodnie ze wzorem powyżej podanym. Dzięki temu na bierząco są generowane kolejne ilorazy różnicowe, pozwalając rekurencyjnie obliczyć kolejne (w implementacji rekurencje zastąpiono odpowiednią iteracją).

1.3 Pseudokod

Algorithm 1: Funkcja licząca kolejne ilorazy różnicowe

ilorazyRoznicowe (x, f);

Input : x - wektor węzłów, f - wektor wartości funkcji w odpowiednich węzłach z x

Output: fx - wektor ilorazów różnicowych w postaci $\{f[x_0, ..., x_k] : k \in \{0, ..., n\}\}$

$n \leftarrow length(x)$;
for $i \leftarrow 1$ **to** n **do**
    $tmp[i] \leftarrow f[i]$;
    for $k \leftarrow 1$ **to** $i - 1$ **do**
        $tmp[i-k] \leftarrow \frac{tmp[i-k+1] - tmp[i-k]}{x[i] - x[i-k]}$;
    end
    $fx[i] = tmp[1]$;
end
return fx;

2 Zadanie 2.

2.1 Wielomian interpolacyjny Newtona

2.2 Opis

Wielomian interpolacyjny Newtona jest to wielomian, który interpoluje podaną funkcję f oraz jest postaci podanej poniżej.

Postać Newtona wielomianu

$$N_n(x) = f[x_0] + f[x_0, x_1](x - x_0) + ... + f[x_0, x_1, x_2, ..., x_n](x - x_0)(x - x_1)...(x - x_{n-1})$$

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

Zadanie polegało na zaimplementowaniu funkcji, która z pomocą algorytmu Hornera wylicza wartość wielomianu postaci Newtona.

Uogólniony algorytm Hornera

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

$$w_i(x) = f[x_0, ..., x_i] + (x - x_i)w_{i+1}(x)$$

$$i = n - 1, ..., 0$$

Związek pomiędzy wartością wielomianu Newtona a algorytmem Hornera

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

Aby obliczyć wartość $N_n(x_0)$ dla podanego x_0 powyższym algorytmem w czasie $O(n)$ rekurencja została zastąpiona odpowiednią iteracją. Użyta iteracja do obliczenia wartości został przedstawiony w poniższym pseudokodzie.

2.3 Pseudokod

Algorithm 2: Wyliczenie wartości wielomianu Newtona używając uogólnionego algorytmu Hornera w czasie $O(n)$

```
warNewton (x, fx, t);
Input  : x - wektor węzłów, fx - wektor ilorazów różnicowych względem wektora x (odpowiednio
         fx[i] = f[x_0, ..., x_{i-1}]), t - argument wielomianu
Output: nt - wyliczona wartość
n ← length(x); nt ← fx[n];
for i ← n − 1 downto 1 do
    nt ← fx[i] + (t − x[i]) ∗ nt;
end
return nt;
```

3 Zadanie 3.

3.1 Postać naturalna wielomianu interpolacyjnego Newtona

3.2 Opis

Postać naturalna wielomianu jest to wielomian postaci: $W(x) = a_n x^n + ... + a_1 x + a_0$. Zwijając otrzymujemy:

$$W(x) = \sum_{k=0}^{n} a_k x^k$$

Poniższy pseudokod przedstawia zaimplementowaną funkcją obliczającą współczynniki a_i w czasie $O(n^2)$. Aby otrzymać czas kwadratowy, został użyty uogólniony algorytm Hornera do wyliczenia odpowiednich iloczynów. W tej metodzie są użyte pewne własności:

$$a_n^{(i)} = f[x_0, ..., x_n]$$

$$a_i^{(i)} = f[x_0, ..., x_n] - a_{i+1}^{(i+1)} x_i$$

$$a_k^{(i)} = a_k^{(i+1)} - a_{k+1}^{(i+1)} x_i$$

gdzie $a_k^{(i)}$ oznacza współczynnik k w wielomianie w_i z algorytmu Hornera

Używając powyższego wzoru także można użyć zależności wartości wielomianu Newtona do uogólnionego algorytmu Hornera. Z poprzedniego zadania wiadomo, że $N_n(x) = w_0(x)$, dlatego też szukane a_k są jednoznaczne z $a_k^{(0)}$. W implementacji rekurencja została zastąpiona odpowiednimi dwoma iteracjami względem indeksów k oraz i, dzięki czemu czas działania wynosi $O(n^2)$;

3.3 Pseudokod

Algorithm 3: Obliczenie współczynników wielomianu interpolacyjnego postaci naturalnej w czasie $O(n^2)$

```
naturalna (x, fx);
Input  : x - wektor węzłów, fx - wektor ilorazów różnicowych względem wektora x (odpowiednio
         fx[i] = f[x_0, ..., x_{i-1}])
Output: a - wektor zawierający współczynniki postaci naturalnej gdzie a[i] = a_{i-1}
n ← length(x); a[n] = fx[n];
for i ← n − 1 downto 1 do
    a[i] ← fx[i];
    for k ← i to n − 1 do
        a[k] ← a[k] − x[i]a[k + 1];
    end
end
return a;
```

4 Zadanie 4.

4.1 Rysowanie wykresów

4.2 Opis

Zadanie polegało na zaimplementowaniu funkcji, która rysowałaby wykres funkcji oraz wykres wielomianu interpolacyjnego używając implementacji z poprzednich zadań. Funkcja rysująca polegała na stworzeniu dwóch wektorów. Do tego zostały użyte dane wejściowe, tzn: funkcji f, przedziału rysowania oraz liczby całkowitej n, która odpowiada stopniu wielomianu interpolacyjnego. Na początku funkcja generuje $n + 1$ węzłów, które są w równej odległości od siebie, następnie wylicza dla podanych węzłów odpowiadające im wartości. Następnie wygenerowane wektory węzłow oraz ich wartości zostały przekazane do zaimplementowanej wcześniej funkcji *ilorazyRoznicowe* dzięki której wylicza tablice ilorazów różnicowych postaci $\{f[x_0, ..., x_i] : i \in \{0, ..., n\}\}$. Następnie generuje w podobny sposób co zostały generowane węzły, punkty, które zostaną nałożone na wykres z tą różnicą, że jest ich znacznie więcej, aby rysunek był dokładniejszy. Dla wykresu wielomianu interpolacyjnego, do obliczenia wartości danych węzłów została użyta wcześniej zaimplementowana funkcja $warNewton$. Ostatnim krokiem jest ostatecznie nałożenie wyliczonych danych na wykres.

4.3 Pseudokod

Algorithm 4: Funkcja rysująca wykres funkcji oraz wielomianu interpolacyjnego stopnia n

```
1  rysujNnfx (f, a, b, n);
   Input  : f - dana funkcja, a, b - przedział rysowania, n - stopień wielomianu interpolacyjnego
   Output: wykres
2  h ← (b−a)/n;
3  for k ← 0 to n do
4  |  xk ← a + kh;
5  |  append(x, xk); append(values, f(xk));
6  end
7  fx ← ilorazyRoznicowe(x, values);
8  temporaryX ← x;
9  n ← wybrana dokładność;
10 Powtórzenie pętli z linii 3. z tą różnicą, że zamiast f(xk) używa się
   warNewton(temporaryX, fx, xk)
11 Nałożenie wygenerowanych wartości na wykresy
```

5 Zadanie 5.

5.1 Problem

W zadaniu został przedstawiony problem polegający na porównaniu dwóch funkcji z ich wielomianami interpolacyjnymi stopnia n używając powyższych funkcji oraz podanych przedziałów rysowania.

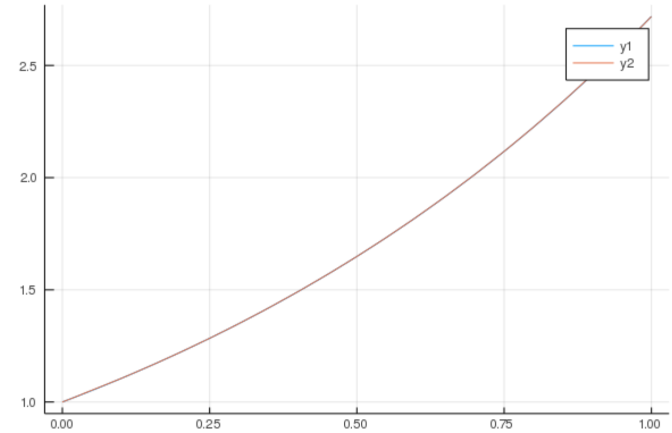
$n \in \{5, 10, 15\}$
$a(x) = e^x$, przedział rysowania: $[0, 1]$
$b(x) = x^2 \sin x$, przedział rysowania: $[-1, 1]$

5.2 Wyniki

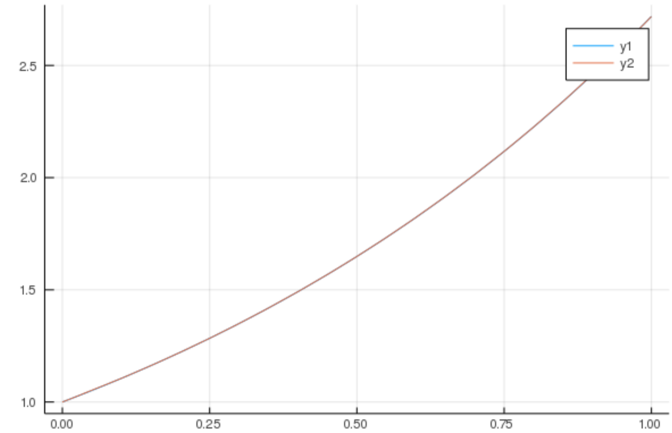
Objaśnienie wykresów:
y1 (kolor niebieski) - graficzne przedstawienie wyliczonyego wielomianu interpolacyjnego.
y2 (kolor czerwony) - graficzne przedstawienie podanej funkcji.

Wykresy funkcji a:

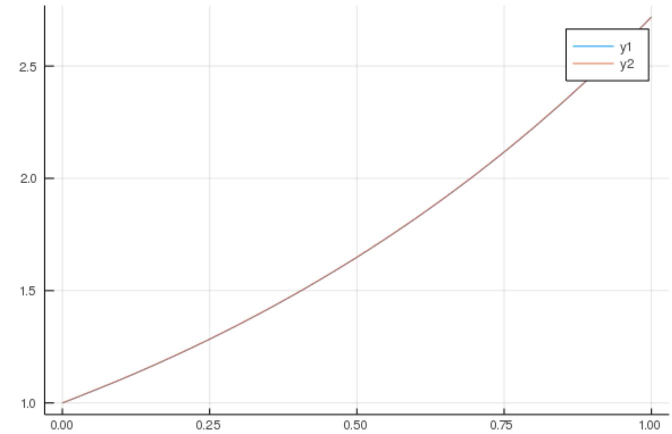
$n = 5$



$n = 10$

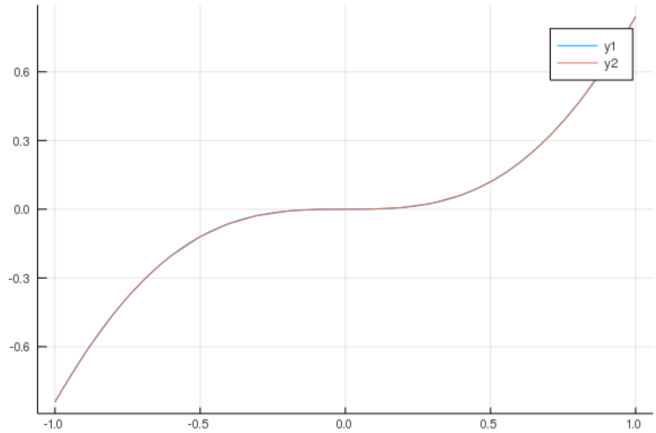


$n = 15$

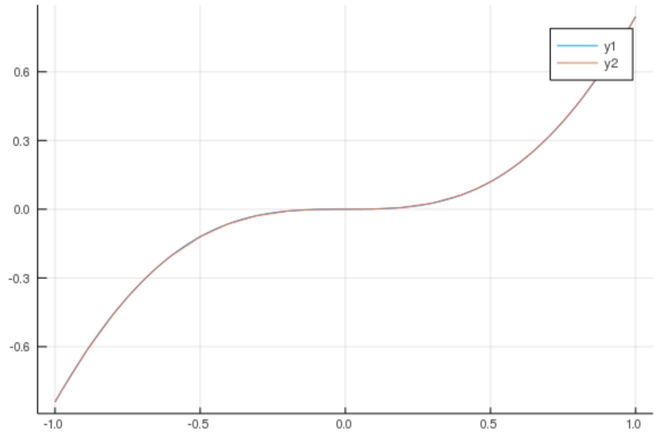


Wykresy funkcji b:

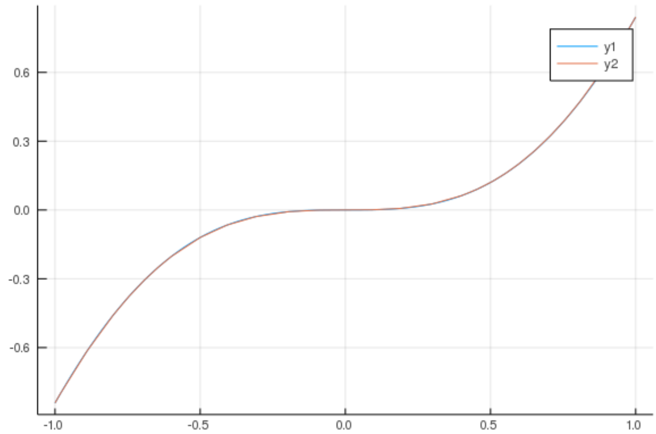
$n = 5$



$n = 10$



$n = 15$



5.3 Wnioski

Obserwując powyższe wykresy można zauważyć zgodność wielomianu interpolacyjnego z pierwotną funkcją. Oznacza to, że dla podanych funckji dobranie węzłów równoodległych od siebie jest odpowiednim rozwiązaniem do uzyskania szukanego przybliżenia wielomianem interpolacyjnym. Dzieje się to dlatego, że obie z funkcji są funkcjami gładkimi, co oznacza, że istnieją ich pochodne wszystkich rzędów oraz są one ciągłe.

6 Zadanie 6.

6.1 Problem

Problemem w zadaniu było pokazanie na wykresach podane funkcjie oraz ich wielomiany interpolacyjne stopnia n.

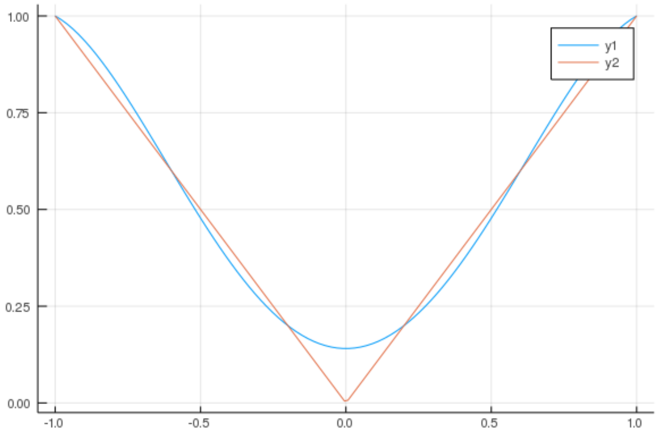
$n \in \{5, 10, 15\}$
$a(x) = |x|$, przedział rysowania: $[-1, 1]$
$b(x) = \frac{1}{1+x^2}$, przedział rysowania: $[-5, 5]$

6.2 Wyniki

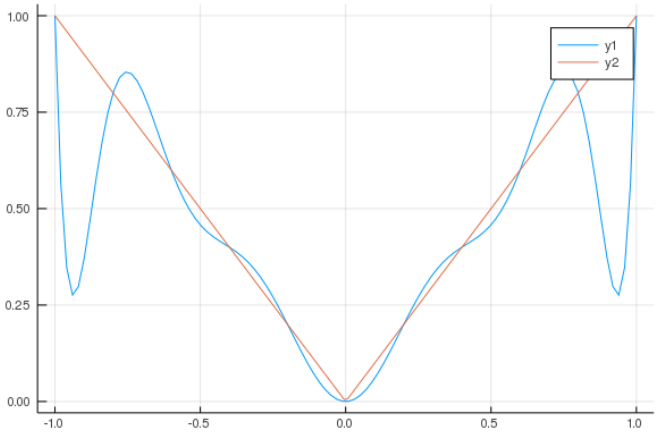
Objaśnienie wykresów:
y1 (kolor niebieski) - graficzne przedstawienie wyliczonyego wielomianu interpolacyjnego.
y2 (kolor czerwony) - graficzne przedstawienie podanej funkcji.

Wykresy funkcji a:

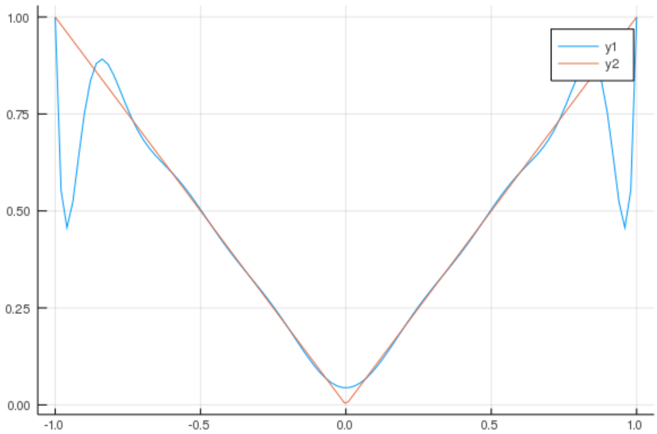
$n = 5$



$n = 10$

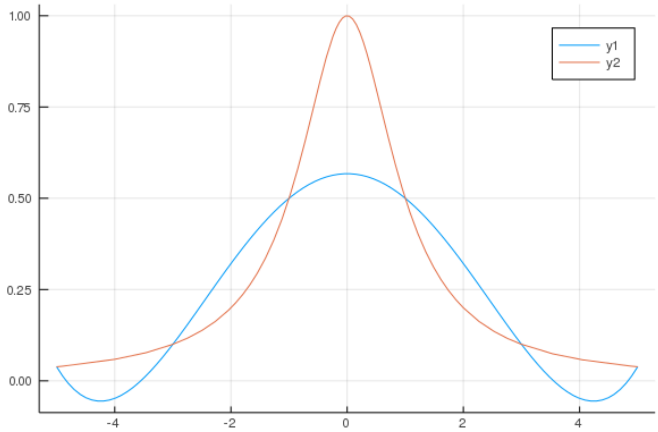


$n = 15$

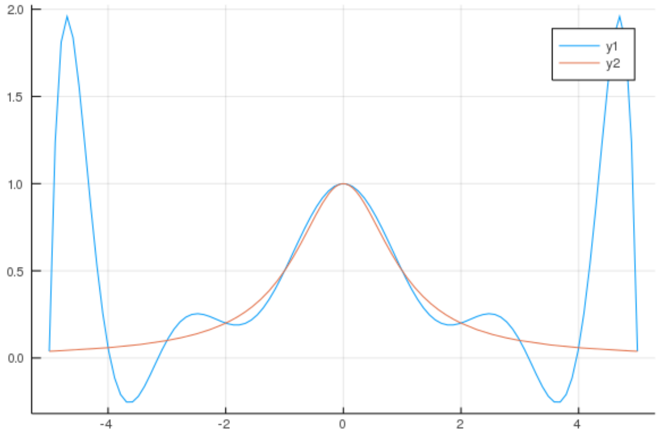


Wykresy funkcji b:

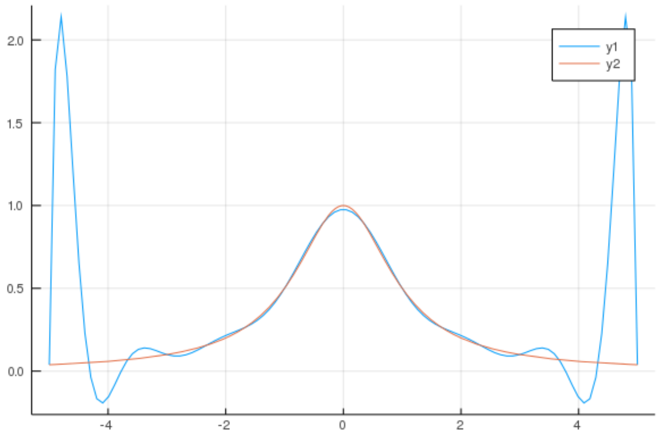
$n = 5$



$n = 10$



$n = 15$



6.3 Wnioski

Analizując powyższe wykresy od razu można zauważyć, że wielomian interpolujący jest rozbiezny względem funkcji. Rozbiezność najwidoczniejsza jest przy końcach przedziału. Także dla większej ilości węzłów rozbieżność jest większa. Taki efekt nazywa się efeketem Rungego. Zachowanie to jest częste dla wielomianów interpolacyjnych wysokiego stopnia kiedy węzły są równoodległe od siebie. W przypadku funkcji a dzieje się to dlatego, że nie jest ona różniczkowalna i nie istnieje jej pochodna. Oznacza to, że funkcja a odbiega znaczaco od funkcji gładkiej, co jest jednym z przypadków kiedy występuj efekt Rungego. Natomiast funkcja b jest gładka oraz ciągła, przez co można zakładać, że wraz z wzrostem liczby węzłów wielomian bedzie przybliżał się do interpolowanej funkcji lecz tak sie nie dzieje. Jest to związane tym, że węzły są w równej odległości od siebie, więc na krawędziach przedziału, gdzie daną funkcje trudniej przybliżyć jest tyle samo węzłów co przy środku przedziału. Można pomyśleż, że aby temi zapobiec nalezy dołożyć odpowiednią liczbe węzłów przy końcach przedziału. Takie zachowanie jest typowe dla wielomianów Czebyszewa, które posiadają rozmieszczenie węzłów zagęszczone przy końcach przedziału. Rozwiązanie to zapobiega rozbieżności wielomianu interpolacyjnego względem interpolującej funkcji.