Obliczenia Naukowe

Lista nr 4

Eryk Krupa
244993

1 Zadanie 1

Rozważmy funkcję obliczającą ilorazy różnicowe.

1.1 Rozwiązanie

Iloraz różnicowy k-tego rzędu można obliczyć stosując następujący wzór rekurencyjny:

1. dla $k = 0$
$$f[x_i] = f(x_i),$$

2. dla $k = 1$
$$f[x_i, x_j] = \frac{f(x_j) - f(x_i)}{x_j - x_i},$$

3. dla $k > 1$
$$f[x_i, x_{i+1}, \ldots, x_{i+k}] = \frac{f[x_{i+1}, x_{i+2}, \ldots, x_{i+k}] - f[x_i, x_{i+1}, \ldots, x_{i+k-1}]}{x_k - x_i}.$$

Znając węzły i wartości funkcji, a co za tym idzie wartości ilorazów różnicowych $f[x_i] = f(x_i)$ zerowego rzędu możemy utworzyć tablice ilorazów różnicowych wyższych rzędów. W tej sytuacji można byłoby użyć dwuwymiarowej tablicy, taki algorytm nie byłby jednak optymalny. Jak się okazuje, wystarczy również jednowymiarowa tablica fx. Tablica fx inicjalizowana jest wartościami z tablicy f[i]. Dokładny opis dalszego postępowania przedstawia poniższy algorytm.

1.2 Algorytm

Dane:

x – wektor długości n + 1 zawierający węzły $x_0, ..., x_n x[1] = x_0, ..., x[n + 1] = x_n$,

f – wektor długości n + 1 zawierający wartości interpolowanej funkcji w węzłach $f(x_0), ..., f(x_n)$

Wyniki:

fx – wektor długości n + 1 zawierający obliczone ilorazy różnicowe

Algorithm 1: Obliczanie ilorazów różnicowych

```
function ilorazyRoznicowe(x,f)
    for i ← 1 to length(f) do
        fx[i] ← f[i]
    end
    for i ← 1 to length(f) do
        for j ← length(f) downto i do
            fx[j] ← (fx[j] − fx[j−1]) / (x[j] − x[j−i])
        end
    end
    return f_x
```

2 Zadanie 2

Rozważmy działającą w czasie liniowym ($O(n)$) funkcję obliczającą wartość wielomianu interpolacyjnego stopnia n w postaci Newtona $N_n(x)$ w punkcie $x = t$ za pomocą algorytmu Hornera.

2.1 Rozwiązanie

Wzór wielomianu interpolacyjnego Newtona można przedstawić używając ilorazów różnicowych:

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

Wartość tak wyrażonego wielomianu można łatwo obliczyć stosując uogólniony algorytm Hornera, który został użyty w poniższym algorytmie.

2.2 Algorytm

Dane:

x – wektor długości n + 1 zawierający węzły $x_0, ..., x_n, x[1] = x_0, ..., x[n+1] = x_n$

fx – wektor długości n + 1 zawierający ilorazy różnicowe $fx[1] = f[x_0], fx[2] = f[x_0, x_1], ..., fx[n] = f[x_0, ..., x_{n1}], fx[n + 1] = f[x_0, ..., x_n]$

t – punkt, w którym należy obliczyć wartość wielomianu

Wyniki:
nt – wartość wielomianu w punkcie t

Algorithm 2: Obliczanie wartości wielomianu interpolacyjnego w punkcie t.

```
function warNewton(x, fx, t)
    n ← length(fx)
    nt ← fx[n]
    for i ← n − 1 downto 1 do
        nt ← fx[i] + (t − x[i])
    end
    return nt
```

3 Zadanie 3

Znajdźmy w czasie $O(n^2)$ współczynniki postaci naturalnej wielomianu interpolacyjnego $a_0, ..., a_n$ znając współczynniki tegoż wielomianu w postaci Newtona $c_0 = f[x_0], c_1 = f[x_0, x_1], c_2 = f[x_0, x_1, x_2], ..., c_n = f[x_0, ..., x_n]$ oraz węzły $x_0, x_1, ..., x_n$.

3.1 Rozwiązanie

Aby znaleźć współczynniki postaci naturalnej wielomianu interpolacyjnego, posłużymy się algorytmem Hornera z zadania poprzedniego. Oznaczmy $w_n(x) = f[x_0, x_1, ..., x_n]$. Możemy zauważyć, że w wielomianie interpolacyjnym n-tego stopnia współczynnik a_n przy najwyższej potędze x jest równy c_n. Dodatkowo, $w_n(x) = a_n$. Dzięki temu, w następnych krokach algorytmu można obliczyć a_i bazując na współczynniku a_{i+1}. W tym celu algorytm przechodząc po wszystkich w_i od $i = n$ do 0, modyfikuje współczynniki postaci naturalnej, żeby doprowadzić do postaci naturalnej dla każdego w_i. Poniżej przedstawiono ten algorytm w formie pseudokodu.

3.2 Algorytm

Dane:
x – wektor długości n + 1 zawierający węzły $x_0, ..., x_n, x[1] = x_0, ..., x[n+1] = x_n$

fx – wektor długości n + 1 zawierający ilorazy różnicowe $fx[1] = f[x0], fx[2] = f[x0, x1], ..., fx[n] = f[x_0, ..., x_{n1}], fx[n + 1] = f[x_0, ..., x_n]$

Wyniki:

a – wektor długości n + 1 zawierający obliczone współczynniki postaci naturalnej $a[1] = a_0, a[2] = a_1, ..., a[n] = a_{n1}, a[n + 1] = a_n$

Algorithm 3: Współczynniki naturalne wielomianu interpolacyjnego.

```
function naturalna(x, fx)
    n ← length(fx)
    a[n] ← fx[n]
    for i ← n − 1 downto 0 do
        a[i] ← fx[i] − a[i + 1] × x[i]
        for j ← i + 1 downto n − 1 do
            a[j] ← a[j] − a[j + 1] ∗ x[i]
        end
    end
    return a
```

4 Zadanie 4

Rozważmy wielomian stopnia n w postaci Newtona interpolujący zadaną funkcję $f(x)$ w przedziale $[a, b]$ z użyciem węzłów równoodległych.

4.1 Rozwiązanie

Na początek należy obliczyć węzły interpolacji w przedziale $[a, b]$ za pomocą wzoru $x_k = a + k, h = \frac{b-a}{n}, k = 0, 1, ..., n,$ oraz wartości funkcji w obliczonych węzłach. Następnie można wykorzystać funkcje z zadania pierwszego, aby obliczyć ilorazy różnicowe. Za pomocą uzyskanych wartości można narysować wykres funkcji f oraz jego wielomian interpolacyjny.

5 Zadanie 5

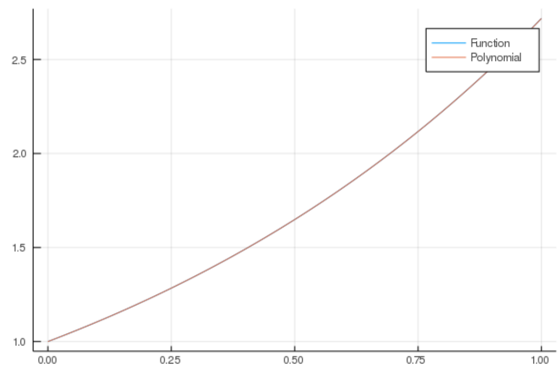
Przetestujmy funkcje z zadania poprzedniego dla następujących przypadków:

a) $f(x) = e^x$, $[a, b] = [0, 1], n \in \{5, 10, 15\}$,

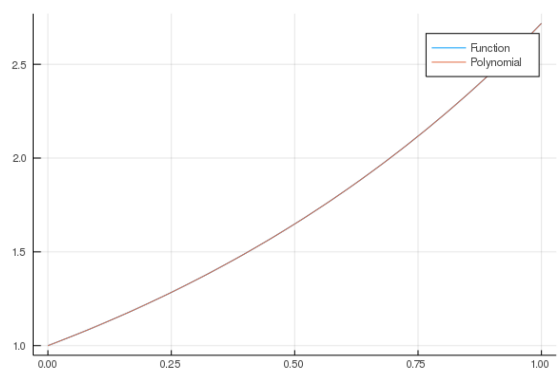
b) $f(x) = x^2 \sin x$, $[a, b] = [-1, 1]$, $n \in \{5, 10, 15\}$.

5.1 Wyniki

Poniższe wykresy prezentują otrzymane wyniki



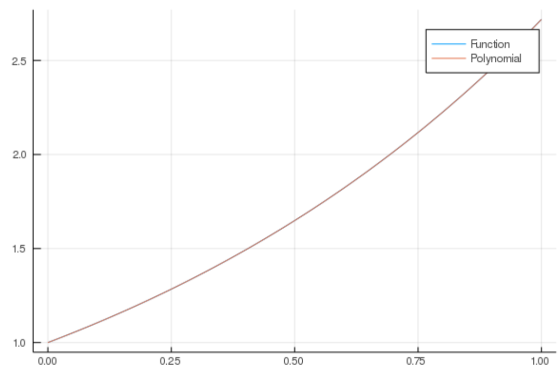
Rysunek 1: $n = 5$



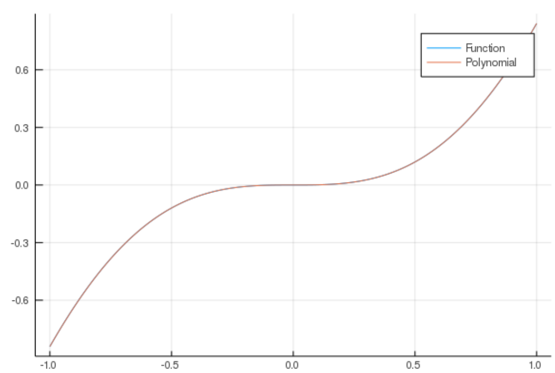
Rysunek 2: $n = 10$

5.2 Wnioski

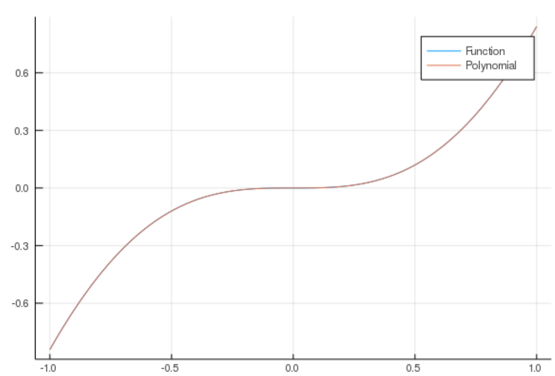
Możemy zauważyć, że dla obu rozpatrywanych funkcji, wielomiany pokrywają się z interpolowaną funkcją. Im wyższy stopień wielomianu, tym wyższa



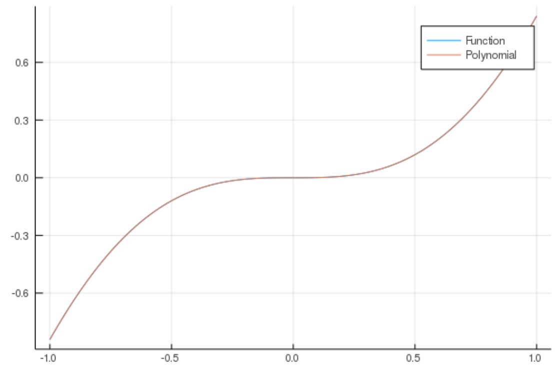
Rysunek 3: $n = 15$



Rysunek 4: $n = 5$



Rysunek 5: $n = 10$



Rysunek 6: $n = 15$

dokładność, ale już dla $n = 5$, wartości różnią się dopiero na szóstym miejscu po przecinku. Dla wyższych n są to jeszcze bardziej odległe miejsca.

6 Zadanie 6

Przetestujmy funkcje z zadania czwartego dla następujących przypadków:

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

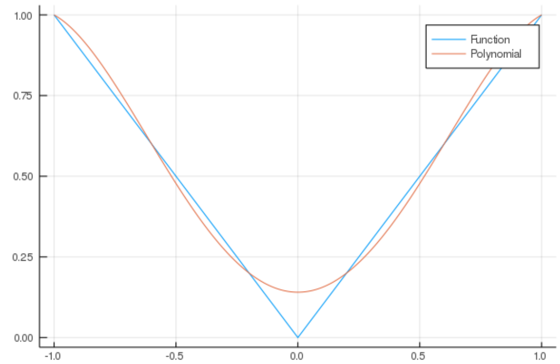
b) $\frac{1}{1+x^2}, [-5, 5], n = 5, 10, 15.$

6.1 Wyniki

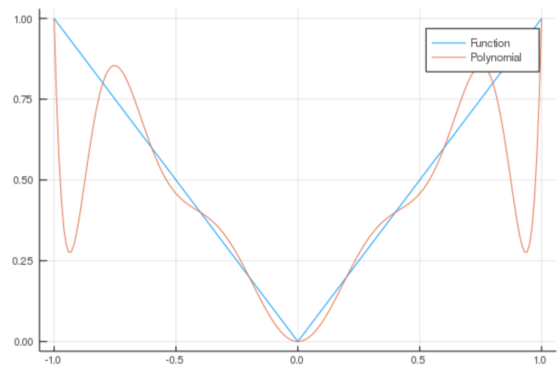
Poniższe wykresy prezentują otrzymane wyniki

6.2 Wnioski

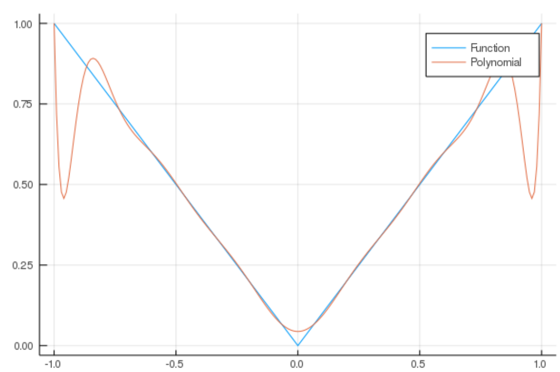
W tym przypadku widać wyraźne rozbieżności, szczególnie na krańcach przedziałów. Pierwsza funkcja nie jest różniczkowalna, z tego powodu można zaobserwować odchylenie wielomianu od funkcji. Bardziej interesująca jest jednak druga funkcja, dla której od n równego około 10, zachodzi zjawisko Runge'go, co oznacza, że pomimo zwiększania ilości węzłów, dokładność interpolacji wielomianowej spada. Zjawisko to występuje czasem, kiedy odległość między kolejnymi węzłami jest stałą. Sposobem na uniknięcie tego



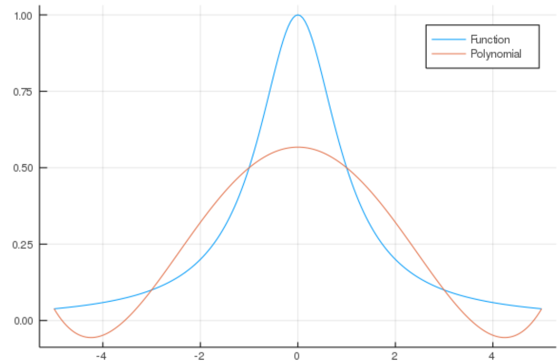
Rysunek 7: $n = 5$



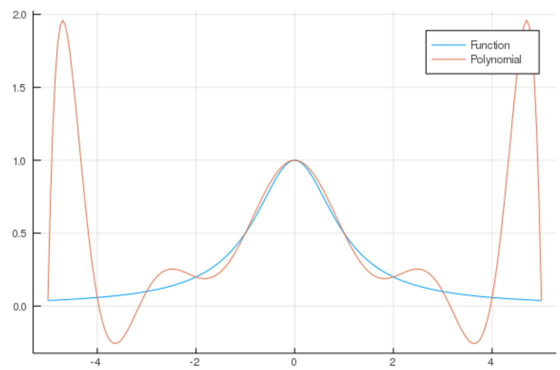
Rysunek 8: $n = 10$



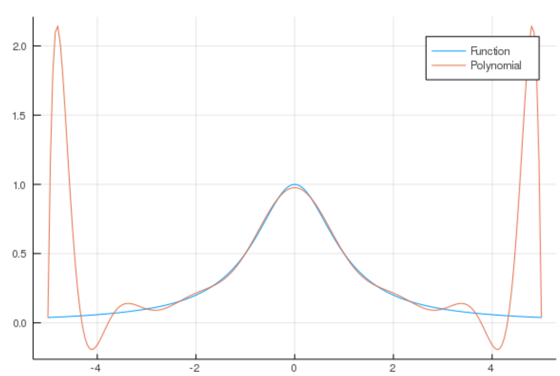
Rysunek 9: $n = 15$



Rysunek 10: $n = 5$



Rysunek 11: $n = 10$



Rysunek 12: $n = 15$

zjawiska jest zastosowanie zmiennej odległości między węzłami. Jako węzły można również wybrać zera wielomianu Czebyszewa.