Sprawozdanie Lista 4

Paweł Krzyszczak

December 2024

Problem interpolacji

Problem interpolacji można sformułować następująco: mamy $n+1$ par $(x_i, y_i = f(x_i))$, gdzie $\forall_{i,j}\ x_i \neq x_j$ oraz $i, j \in \{0, \ldots, n\}$. Chcemy znaleźć wielomian $p_n(x)$ stopnia co najwyżej n, spełniający warunek:

$$\forall_{i \in \{0,\ldots,n\}}\ \ p_n(x_i) = f(x_i) = y_i.$$

Wiemy, że taki wielomian istnieje i jest jednoznaczny.
Wielomian interpolacyjny można zapisać w postaci:

$$p_n(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n,$$

gdzie współczynniki $a_0, a_1, \ldots, a_n$ można wyznaczyć, rozwiązując układ równań z macierzą Vandermonde'a. Jednakże takie podejście jest niekorzystne z powodu słabej uwarunkowalności macierzy Vandermonde'a przy obliczeniach zmiennoprzecinkowych.
Przedstawmy więc p_n w innej bazie. Zdefiniujmy wielomiany bazowe:

$$q_0(x) = 1,$$

$$q_1(x) = (x - x_0),$$

$$q_2(x) = (x - x_0)(x - x_1),$$

$$\vdots$$

$$q_n(x) = (x - x_0)(x - x_1)\cdots(x - x_{n-1}).$$

Wówczas istnieją takie współczynniki c_j, że:

$$p_n(x) = \sum_{j=0}^{n} c_j q_j(x).$$

Wykorzystując warunek interpolacji, otrzymujemy układ równań pozwalający wyznaczyć współczynniki $c_0, c_1, \ldots, c_n$:

$$\forall_{i \in \{0,\ldots,n\}}\ \ \sum_{j=0}^{n} c_j q_j(x_i) = f(x_i).$$

Zauważmy, że każdy współczynnik c_k zależy od wartości $f(x_0), f(x_1), \ldots, f(x_k)$. Wprowadźmy więc oznaczenie $c_k = f[x_0, x_1, \ldots, x_k]$, nazywane ilorazem różnicowym. Po podstawieniu do wzoru wielomianu interpolacyjnego otrzymujemy postać Newtona:

$$p_n(x) = \sum_{k=0}^{n} c_k q_k(x) = \sum_{k=0}^{n} f[x_0, x_1, \ldots, x_k] \prod_{j=0}^{k-1} (x - x_j).$$

1 Zadanie 1

1.1 Opis problemu

Celem zadania jest napisanie funkcji obliczającej ilorazy różnicowe bez użycia tablicy dwuwymiarowej.

Funkcja przyjmuje następujące parametry:

`x` – wektor długości $n+1$, zawierający węzły $x_0, \ldots, x_n$:

$$\begin{aligned} &\texttt{x[1]} = x_0, \\ &\vdots \\ &\texttt{x[n+1]} = x_n. \end{aligned}$$

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

Funkcja zwraca:

`fx` – wektor długości $n+1$, zawierający obliczone ilorazy różnicowe:

$$\begin{aligned} &\texttt{fx[1]} = f[x_0], \\ &\texttt{fx[2]} = f[x_0, x_1], \\ &\vdots \\ &\texttt{fx[n]} = f[x_0, \ldots, x_{n-1}], \\ &\texttt{fx[n+1]} = f[x_0, \ldots, x_n]. \end{aligned}$$

Implementacja ma być wykonana bez użycia tablicy dwuwymiarowej.

1.2 Opis metody

Algorytm ma na celu obliczenie ilorazów różnicowych dla danych par $(x_i, y_i = f(x_i))$.

Wiemy, że:

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

a kolejne ilorazy różnicowe spełniają wzór:

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

Naturalnym podejściem byłoby utworzenie macierzy trójkątnej i wykorzystanie jej elementów do dalszych obliczeń, jak poniżej:

$$\begin{bmatrix} f[x_0] & f[x_0, x_1] & \ldots & f[x_0, \ldots, x_n] \\ f[x_1] & f[x_1, x_2] & \ldots & f[x_1, \ldots, x_n] \\ \vdots & \vdots & \ddots & \vdots \\ f[x_n] & & & \end{bmatrix}.$$

Chcemy jednak uniknąć użycia tablicy dwuwymiarowej. Zauważmy, że po obliczeniu wszystkich ilorazów różnicowych zależnych od k węzłów, pozostałe stają się zbędne. Dzięki temu algorytm można zrealizować na jednej tablicy o długości $n+1$.

Na początku w tablicy zapisujemy wartości funkcji w węzłach (bo $f[x_i] = f(x_i)$). Elementy tablicy są stopniowo nadpisywane kolejnymi ilorazami różnicowymi. Przykład ilustruje poniższa tabela:

$\mathbf{f[x_0]}$	$\mathbf{f[x_0]}$	$\mathbf{f[x_0]}$	$\dots$	$\mathbf{f[x_0]}$	$\mathbf{f[x_0]}$
$f[x_1]$	$\mathbf{f[x_0, x_1]}$	$\mathbf{f[x_0, x_1]}$	$\dots$	$\mathbf{f[x_0, x_1]}$	$\mathbf{f[x_0, x_1]}$
$f[x_2]$	$f[x_1, x_2]$	$\mathbf{f[x_0, x_1, x_2]}$	$\dots$	$\mathbf{f[x_0, x_1, x_2]}$	$\mathbf{f[x_0, x_1, x_2]}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$f[x_{n-1}]$	$f[x_{n-2}, x_{n-1}]$	$f[x_{n-3}, x_{n-2}, x_{n-1}]$	$\dots$	$\mathbf{f[x_0, \dots, x_{n-1}]}$	$\mathbf{f[x_0, \dots, x_{n-1}]}$
$f[x_n]$	$f[x_{n-1}, x_n]$	$f[x_{n-2}, x_{n-1}, x_n]$	$\dots$		$\mathbf{f[x_0, \dots, x_n]}$

Tabela 1: Stany tablicy po kolejnych iteracjach.

Metoda ma złożoność obliczeniową $O(n^2)$ i pamięciową $O(n)$.

1.3 Pseudokod

Algorithm 1: Funkcja obliczająca ilorazy różnicowe

Input: $\overline{x}$ – wektor punktów, $\overline{f}$ – wektor wartości funkcji w punktach z $\overline{x}$

Output: $\overline{c}$ – wektor ilorazów różnicowych

$\overline{c} \leftarrow \overline{f}$;
for j ***od*** 1 ***do*** n **do**
  for k ***od*** n ***w dół do*** j **do**
    $c_k \leftarrow \frac{c_k - c_{k-1}}{x_k - x_{k-j}}$;
  end
end
return $\overline{c}$

“‘latex

2 Zadanie 2

2.1 Opis problemu

Napisać funkcję obliczającą wartość wielomianu interpolacyjnego stopnia n w postaci Newtona $N_n(x)$ w punkcie $x = t$ za pomocą uogólnionego algorytmu Hornera, w czasie $O(n)$.
Funkcja przyjmuje następujące parametry:

`x` – wektor długości $n+1$, zawierający węzły $x_0, \dots, x_n$:

`x[1]` $= x_0$,

$\vdots$

`x[n+1]` $= x_n$,

`fx` – wektor długości $n+1$, zawierający ilorazy różnicowe:

`fx[1]` $= f[x_0]$,

`fx[2]` $= f[x_0, x_1]$,

$\vdots$

`fx[n]` $= f[x_0, \dots, x_{n-1}]$,

`fx[n+1]` $= f[x_0, \dots, x_n]$,

`t` – punkt, w którym należy obliczyć wartość wielomianu.

Funkcja zwraca:

`nt` – wartość wielomianu w punkcie t.

2.2 Opis metody

Zadanie polega na obliczeniu wartości wielomianu interpolacyjnego stopnia n w postaci Newtona w punkcie t, przy złożoności obliczeniowej $O(n)$. Tradycyjne podejście oparte na rozwinięciu wzoru ma złożoność $O(n^2)$, której chcemy uniknąć.
Przekształćmy wielomian:

$$p_n(x) = \sum_{k=0}^{n} c_k \cdot q_k(x) = f[x_0] + \sum_{k=1}^{n} f[x_0, \dots, x_k](x - x_0)\dots(x - x_k),$$

co możemy zapisać w postaci iteracyjnej:

$$p_n(x) = f[x_0] + (x - x_0)\left(f[x_0, x_1] + (x - x_1)(\dots(f[x_0, \dots, x_{n-1}] + (x - x_{n-1})f[x_0, \dots, x_n])\dots)\right).$$

Rekurencyjnie:

$$w_n(x) = f[x_0, \dots, x_n],$$
$$w_k(x) = f[x_0, \dots, x_k] + (x - x_k)w_{k+1}, \quad \text{dla } k \in \{0, \dots, n-1\}.$$

Finalnie:

$$p_n(x) = w_0(x).$$

Dzięki takiemu przekształceniu algorytm oblicza wartość wielomianu w czasie $O(n)$.

2.3 Pseudokod

Algorithm 2: Funkcja obliczająca wartość wielomianu interpolacyjnego w punkcie t

Input: $\overline{x}$ – wektor węzłów, $\overline{c}$ – wektor ilorazów różnicowych, t – punkt, dla którego szukamy wartości wielomianu
Output: v – wartość wielomianu w punkcie t

$v \leftarrow c_n$;
for i ***od*** $n - 1$ ***do*** 0 **do**
  $v \leftarrow c_i + (t - x_i) \cdot v$;
end
return v;

3 Zadanie 3

3.1 Opis problemu

Znając współczynniki wielomianu interpolacyjnego w postaci Newtona: $c_0 = f[x_0],\ c_1 = f[x_0, x_1],\ \dots,\ c_n = f[x_0, x_1, \dots, x_n]$ (ilorazy różnicowe), oraz węzły $x_0, x_1, \dots, x_n$, należy napisać funkcję obliczającą współczynniki postaci naturalnej wielomianu $a_0, a_1, \dots, a_n$:

$$P(x) = a_n x^n + a_{n-1}x^{n-1} + \dots + a_1 x + a_0$$

w czasie $O(n^2)$.
Funkcja przyjmuje parametry:

`x` – wektor długości $n + 1$ zawierający węzły $x_0, \dots, x_n$:

`x[1]` $= x_0$,

$\vdots$

`x[n+1]` $= x_n$,

`f` – wektor długości $n + 1$ zawierający ilorazy różnicowe:

`fx[1]` $= f[x_0]$,

`fx[2]` $= f[x_0, x_1]$,

$\vdots$

`fx[n]` $= f[x_0, \ldots, x_{n-1}]$,

`fx[n+1]` $= f[x_0, \ldots, x_n]$,

Funkcja zwraca:

`a` – wektor długości $n + 1$ zawierający współczynniki postaci naturalnej:

`a[1]` $= a_0$,

`a[2]` $= a_1$,

$\vdots$

`a[n]` $= a_{n-1}$,

`a[n+1]` $= a_n$.

3.2 Opis metody

Zadaniem algorytmu jest przekształcenie wielomianu w postaci Newtona:

$$P(x) = \sum_{k=0}^{n} c_k \prod_{j=0}^{k-1} (x - x_j),$$

do postaci naturalnej:

$$P(x) = \sum_{i=0}^{n} a_i x^i.$$

Wykorzystajmy rekurencję z poprzedniego zadania:

$$w_n = c_n, \text{ skąd wiemy, że przy } x^n \text{ współczynnik będzie wynosił } c_n$$

$$w_{n-1} = c_{n-1} + (x - x_{n-1})w_n = c_{n-1} + (x - x_{n-1})c_n = c_{n-1} + c_n x - c_n x_{n-1}$$

$$\text{skąd wiemy, że przy } x^{n-1} \text{ współczynnik będzie wynosił } c_{n-1} - x_{n-1}c_n$$

$$w_{n-2} = c_{n-2} + (x - x_{n-2})w_{n-1} = c_{n-2} + (x - x_{n-2})(c_{n-1} + (x - x_{n-1})c_n) =$$

$$= c_{n-2} + (xc_{n-1} + x(x - x_{n-1})c_n - x_{n-2}c_{n-1} - x_{n-2}(x - x_{n-1})c_n) =$$

$$= c_{n-2} + c_{n-1}x + c_n x^2 - c_n x_{n-1} x - c_{n-1} x_{n-2} - c_n x_{n-2} x + c_n x_{n-1} x_{n-2} =$$

$$= c_{n-2} + c_n x^2 + (c_{n-1} - c_n x_{n-1} - c_n x_{n-2})x + x_{n-2}(c_n x_{n-1} - c_{n-1})$$

Obserwacja:

$w_k(x)$ jest zdefiniowane rekurencyjnie jako kombinacja współczynników $c_k, \ldots, c_n$.

Możemy zatem iteracyjnie przekształcić wielomian do postaci naturalnej, zapisując kolejne obliczone współczynniki. Każdy krok wymaga przekształcenia wcześniejszych wyrazów, co prowadzi do złożoności $O(n^2)$.

3.3 Pseudokod

Algorithm 3: Obliczanie współczynników postaci naturalnej wielomianu interpolacyjnego

Input: $\overline{x}$ – wektor węzłów, $\overline{c}$ – wektor ilorazów różnicowych

Output: $\overline{a}$ – wektor współczynników postaci naturalnej wielomianu interpolacyjnego

$a \leftarrow [0, \ldots, 0]$; `// Zainicjuj wektor współczynników na 0`

$a_n \leftarrow c_n$; `// Ustaw współczynnik przy najwyższej potędze`

for k ***od*** $n - 1$ ***do*** 0 **do**

  $a[k] \leftarrow c_k$; `// Ustaw współczynnik bieżącej potęgi`

  for j ***od*** $k + 1$ ***do*** n **do**

    $a[j-1] \leftarrow a[j-1$**end**

    - x[k] $\cdot a[j]$; `// Przesuń współczynniki w dół`

  end

return $\overline{a}$;

4 Zadanie 4

4.1 Opis problemu

Napisać funkcję, która zinterpoluje zadaną funkcję $f(x)$ w przedziale $[a, b]$ za pomocą wielomianu interpolacyjnego stopnia n w postaci Newtona. Następnie narysuje wielomian interpolacyjny i interpolowaną funkcję. Do rysowania zainstaluj np. pakiet Plots, PyPlot lub Gadfly. W interpolacji użyć węzłów równoodległych, tj. $x_k = a + kh, h = (b - a)/n, k = 0, 1, \dots, n$.

Funkcja przyjmuje parametry:

`f` - funkcja $f(x)$ zadana jako anonimowa funkcja

`a, b` - przedział interpolacji

`n` - stopień wielomianu interpolacyjnego

Funkcja zwraca:

- funkcja rysuje wielomian interpolacyjny i interpolowaną funkcję w przedziale $[a, b]$

4.2 Opis metody

Celem zadania jest połączenie zaimplementowanych wcześniej metod w jedną, umożliwiającą graficzne porównanie otrzymanego wielomianu interpolacyjnego z dokładną funkcją.

Z racji tego, że chcemy interpolować za pomocą wielomianu stopnia n, to musimy w przedziale $[a, b]$ wyznaczyć $n + 1$ równoodległych węzłów, takich, że: $a = x_0 < x_2 < \dots < x_{n-1} < x_n = b$, gdzie $x_i = a + i \cdot h, h = \frac{b-a}{n}$. Dla obliczonych x_i obliczamy wartości $f(x_i)$. Za pomocą metody `ilorazyRoznicowe (x::VectorFloat64, f::VectorFloat64)` obliczmy ilorazy różnicowe c_i dla tych punktów. Mając te dane możemy skorzystać z funkcji `warNewton (x::VectorFloat64, fx::VectorFloat64, t::Float64)` i obliczać wartości wielomianu interpolacyjnego dla dowolnego punktu.

Aby móc narysować sensowny wykres, potrzebujemy znacznie więcej punktów niż te wyznaczone węzły interpolacji. Dlatego w przedziale $[a, b]$ wyznaczmy $P \cdot n + 1, P \in \mathrm{N}$ punktów. Wówczas w każdym przedziale $[x_i, x_{i+1})$ znajduje się dokładnie P punktów, a ponadto punktem jest także wartość prawego końca przedziału.

Dla tak obliczonych punktów obliczamy wartości funkcji interpolowanej i wielomianu interpolującego i umieszczamy te dane na wykresie.

4.3 Pseudokod

Algorithm 4: Funkcja przedstawiająca graficznie wykresy funkcji interpolowanej i wielomianu interpolującego

Input: f – funkcja, którą chcemy interpolować, a, b – końce przedziału interpolacji, n – stopień wielomianu interpolacyjnego

Output: wykres wielomianu interpolującego i funkcji interpolowanej

$P \leftarrow 100$;
$h \leftarrow \frac{b-a}{n}$;
for i ***from*** 0 ***to*** n **do**
  $x_i \leftarrow a + i \cdot h$;
  $y_i \leftarrow f(x_i)$;
end
$\overline{c} \leftarrow$ `ilorazyRoznicowe`$(\overline{x}, \overline{y})$;
$z \leftarrow P \cdot n + 1$ // liczba punktów ;
$v \leftarrow \frac{b-a}{z-1}$ // odstęp między punktami ;
for i ***from*** 0 ***to*** z **do**
  $X_i \leftarrow a + i \cdot v$ // wartość x na wykresie;
  $W_i \leftarrow$ `warNewton`$(\overline{x}, \overline{c}, X_i)$ // wartość wielomianu w punkcie x;
  $Y_i \leftarrow f(X_i)$ // wartość funkcji f w punkcie x;
end
$wykres = (x = \overline{X}, y = \overline{W}, \overline{Y})$
return $wykres$

5 Zadanie 5

5.1 Opis problemu

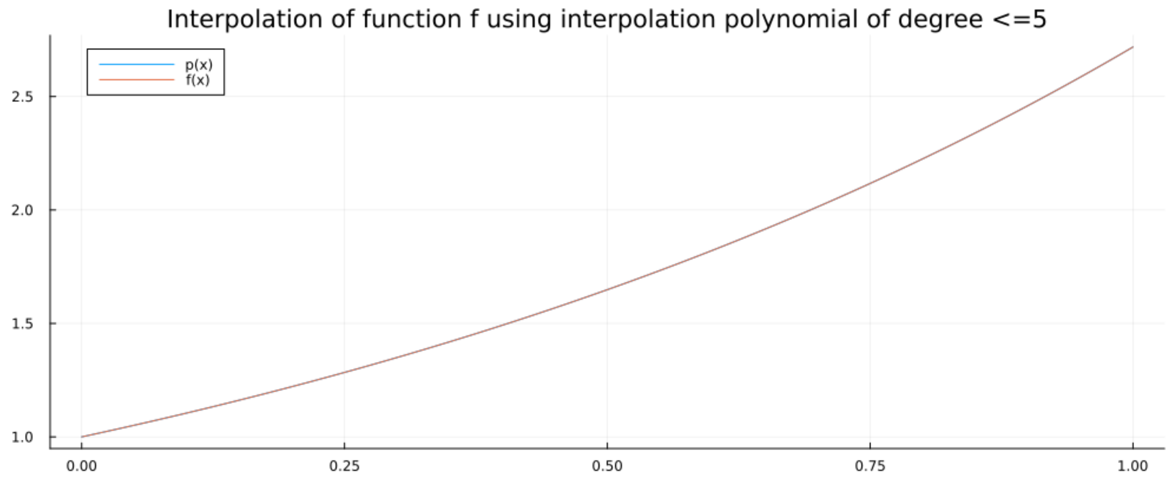
Zadaniem jest przetestowanie funkcji `rysujNnfx(f, a, b, n)` na następujących przykładach:

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

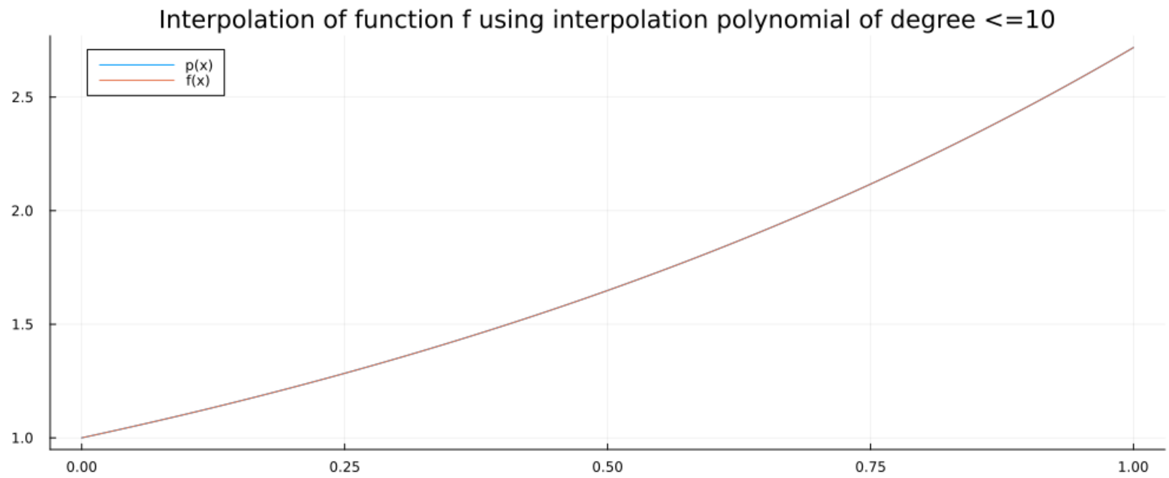
5.2 Rozwiązanie

Funkcja `rysujNnfx(f, a, b, n)` została wykorzystana do generowania wykresów dla podanych danych wejściowych. Wygenerowane wykresy zapisano do plików i wykorzystano do analizy wyników.

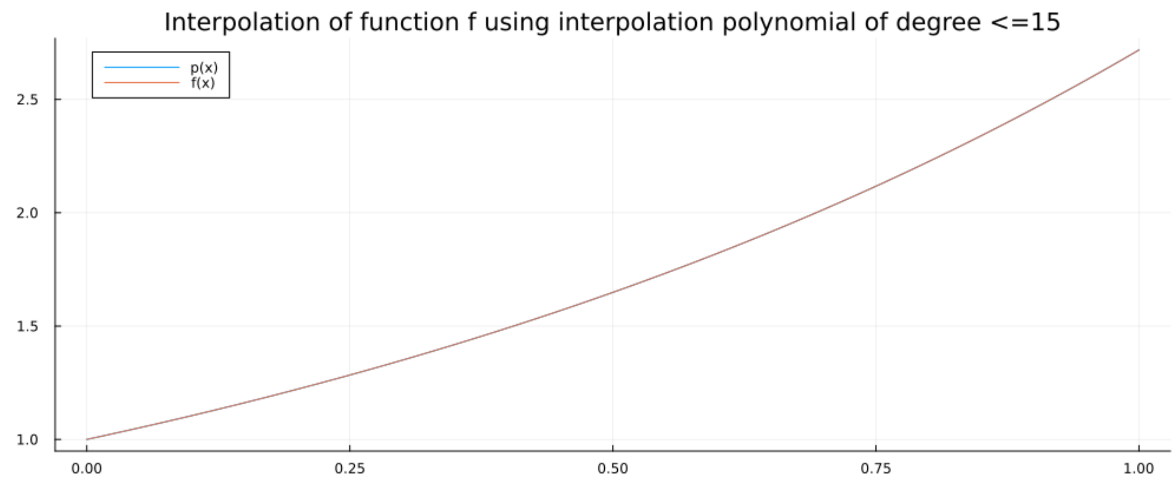
5.3 Wyniki



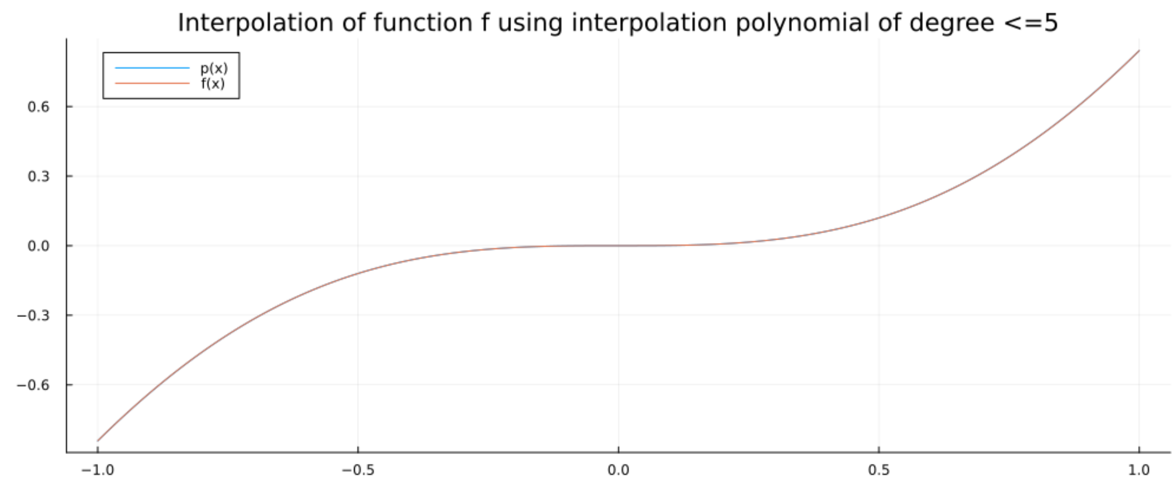
Rysunek 1: Wykres wielomianu interpolacyjnego stopnia ≤ 5 oraz funkcji $f(x) = e^x$ na przedziale $[0, 1]$.



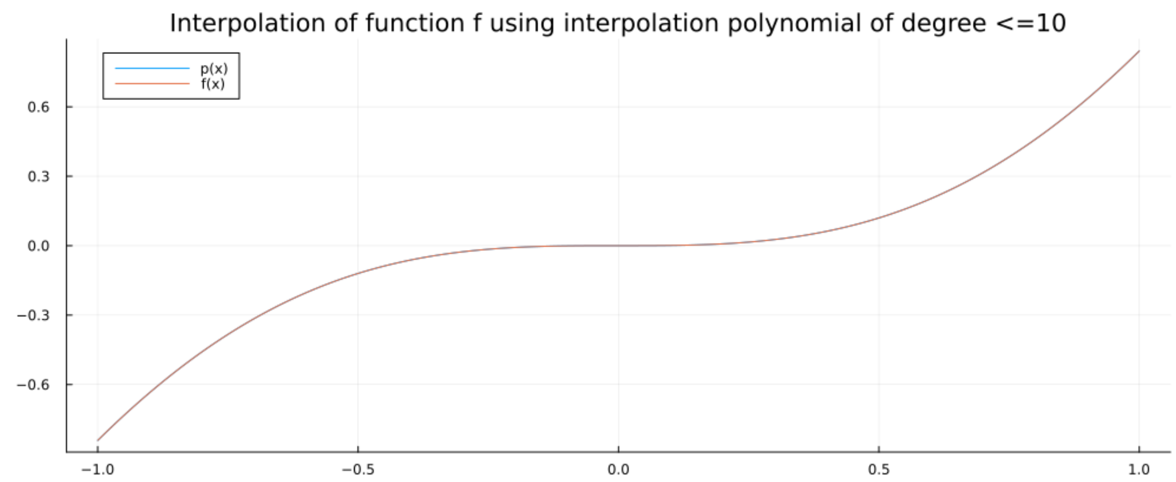
Rysunek 2: Wykres wielomianu interpolacyjnego stopnia ≤ 10 oraz funkcji $f(x) = e^x$ na przedziale $[0, 1]$.



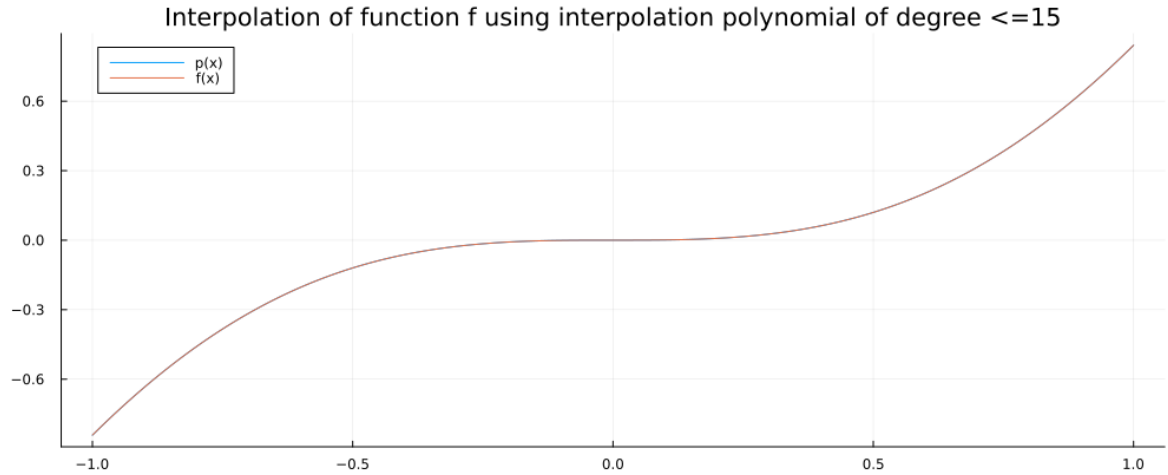
Rysunek 3: Wykres wielomianu interpolacyjnego stopnia ≤ 15 oraz funkcji $f(x) = e^x$ na przedziale $[0, 1]$.



Rysunek 4: Wykres wielomianu interpolacyjnego stopnia ≤ 5 oraz funkcji $f(x) = x^2 \sin x$ na przedziale $[-1, 1]$.



Rysunek 5: Wykres wielomianu interpolacyjnego stopnia ≤ 10 oraz funkcji $f(x) = x^2 \sin x$ na przedziale $[-1, 1]$.



Rysunek 6: Wykres wielomianu interpolacyjnego stopnia ≤ 15 oraz funkcji $f(x) = x^2 \sin x$ na przedziale $[-1, 1]$.

5.4 Interpretacja wyników i wnioski

Analizując wykresy, można zauważyć, że dla funkcji $f(x) = e^x$ oraz $f(x) = x^2 \sin x$ interpolacja wielomianowa działa bardzo dobrze. Nawet przy użyciu wielomianów niskiego stopnia różnice między wielomianem interpolacyjnym a interpolowaną funkcją są znikome.
Funkcje te są dobrze przystosowane do interpolacji, co potwierdza poprawność działania zaimplementowanej metody.

“‘latex

6 Zadanie 6

6.1 Opis problemu

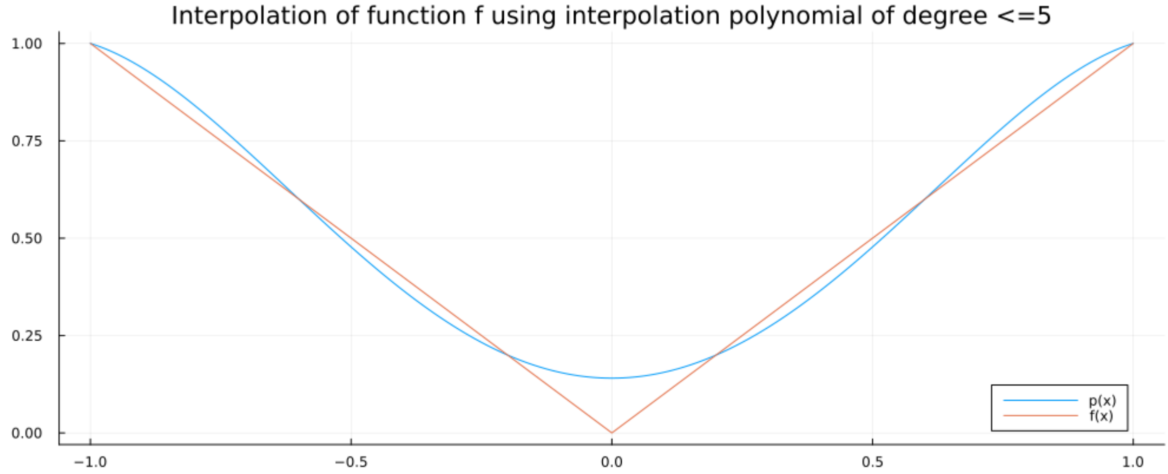
Celem zadania jest przetestowanie funkcji `rysujNnfx(f, a, b, n)` na następujących przykładach:

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

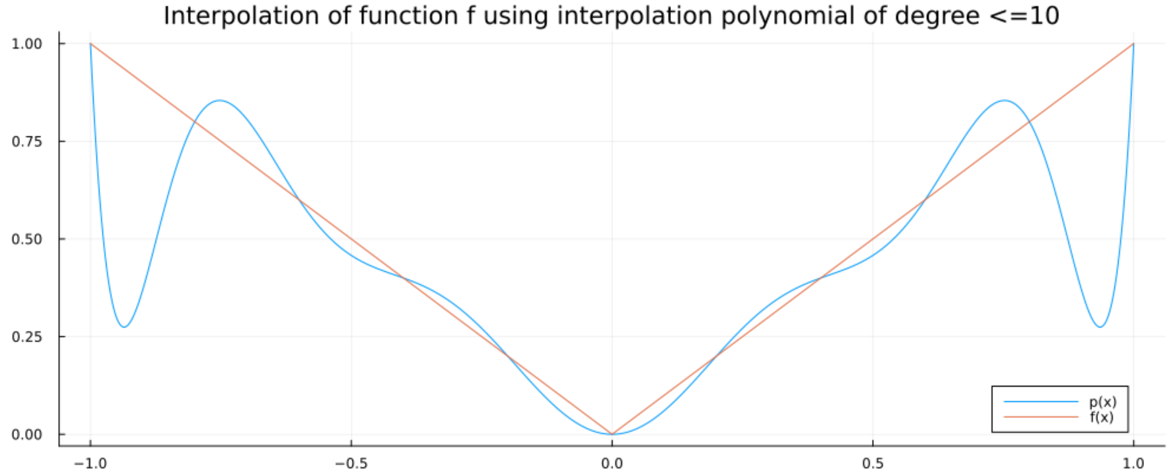
6.2 Rozwiązanie

Do analizy wykorzystano funkcję `rysujNnfx(f, a, b, n)`, która generowała wykresy na podstawie podanych danych wejściowych. Wygenerowane wyniki zapisano w postaci plików graficznych, które posłużyły do dalszej interpretacji.

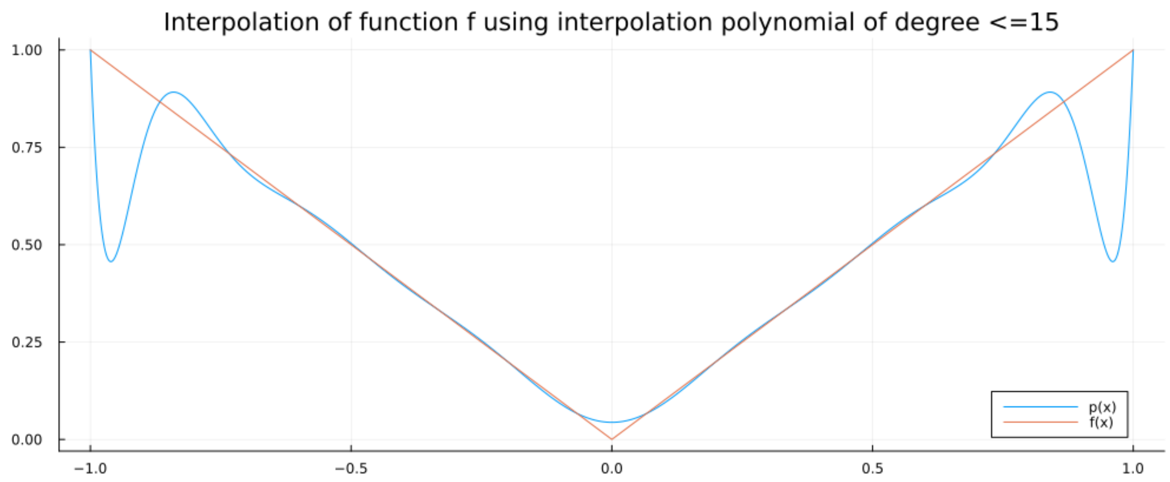
6.3 Wyniki



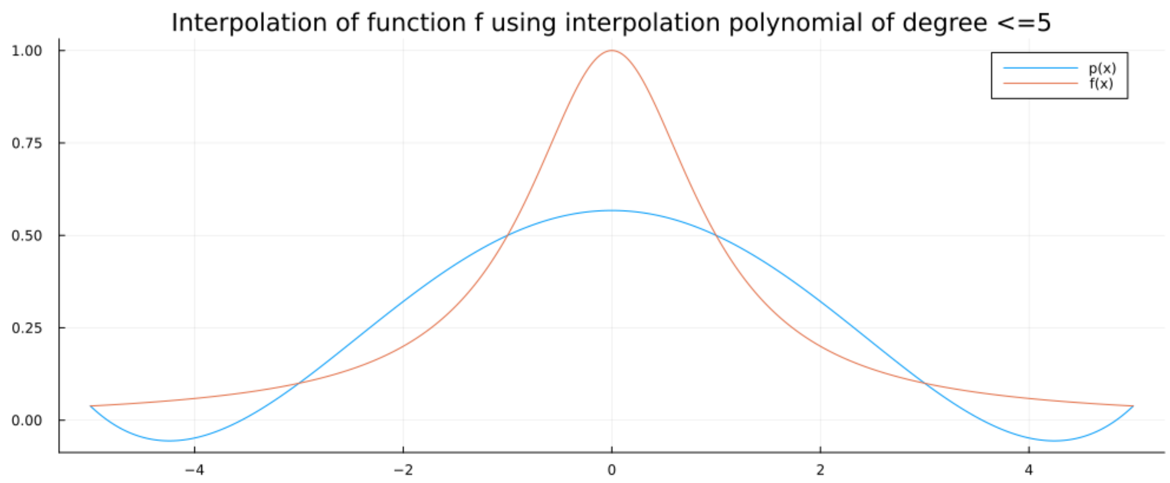
Rysunek 7: Wykres wielomianu interpolacyjnego stopnia ≤ 5 oraz funkcji $f(x) = |x|$ na przedziale $[-1, 1]$.



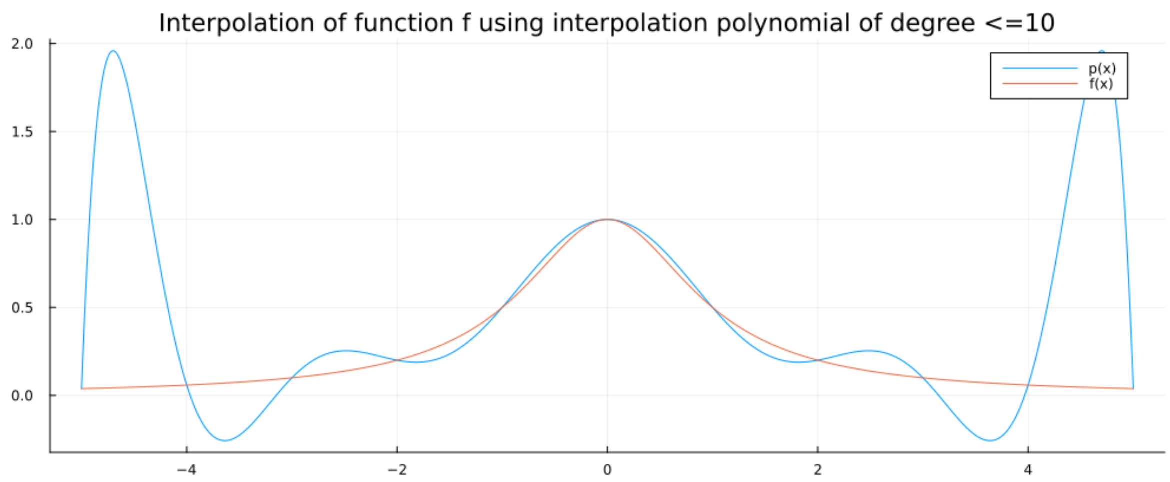
Rysunek 8: Wykres wielomianu interpolacyjnego stopnia ≤ 10 oraz funkcji $f(x) = |x|$ na przedziale $[-1, 1]$.



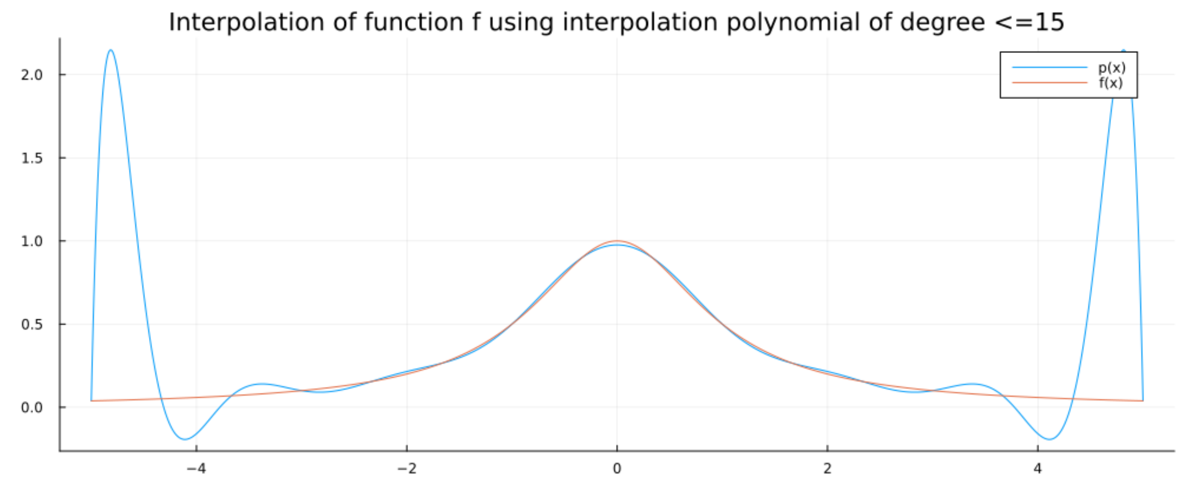
Rysunek 9: Wykres wielomianu interpolacyjnego stopnia ≤ 15 oraz funkcji $f(x) = |x|$ na przedziale $[-1, 1]$.



Rysunek 10: Wykres wielomianu interpolacyjnego stopnia ≤ 5 oraz funkcji $f(x) = \frac{1}{1+x^2}$ na przedziale $[-5, 5]$.



Rysunek 11: Wykres wielomianu interpolacyjnego stopnia ≤ 10 oraz funkcji $f(x) = \frac{1}{1+x^2}$ na przedziale $[-5, 5]$.



Rysunek 12: Wykres wielomianu interpolacyjnego stopnia ≤ 15 oraz funkcji $f(x) = \frac{1}{1+x^2}$ na przedziale $[-5, 5]$.

6.4 Interpretacja wyników i wnioski

W odróżnieniu od poprzedniego zadania, w tym przypadku interpolacja wielomianowa okazała się mniej efektywna. Zwiększenie stopnia wielomianu nie przyniosło oczekiwanych rezultatów.
W przypadku funkcji $f(x) = |x|$ głównym problemem jest brak różniczkowalności w punkcie $x = 0$. Wielomiany są gładkimi funkcjami, co utrudnia wierne odwzorowanie funkcji z "ostrym"wierzchołkiem, takim jak $|x|$.
Dla funkcji $f(x) = \frac{1}{1+x^2}$ zauważono, że wielomian interpolacyjny zaczyna znacznie odbiegać od rzeczywistej funkcji na końcach przedziału. To zjawisko, znane jako **zjawisko Rungego**, występuje szczególnie przy użyciu równoodległych węzłów interpolacyjnych, jak w tym zadaniu.
Możliwym rozwiązaniem tego problemu jest:

- zastosowanie węzłów Czebyszewa, które lepiej radzą sobie z niestabilnością w interpolacji,
- wykorzystanie nierówno rozmieszczonych węzłów, szczególnie zagęszczonych w miejscach krytycznych,
- zmiana metody interpolacji na inną, bardziej odporną na zjawisko Rungego, np. interpolację trygonometryczną.

Wnioski

Interpolacja wielomianowa jest skuteczną metodą przybliżania funkcji, szczególnie gdy dysponujemy ograniczoną liczbą jej wartości. Niemniej jednak, nie jest to metoda pozbawiona ograniczeń. Dla funkcji nieróżniczkowalnych w niektórych punktach, interpolacja może znacząco odbiegać od rzeczywistych wartości. Nawet w przypadku funkcji ciągłych i gładkich, które teoretycznie powinny być dobrze przybliżane, mogą pojawić się nieoczekiwane błędy. Zwiększanie stopnia wielomianu nie zawsze prowadzi do poprawy dokładności – w niektórych przypadkach może wręcz pogorszyć wynik.
Aby zwiększyć wiarygodność wyników, warto stosować różne podziały przedziału na węzły, szczególnie w obszarach, które wydają się problematyczne. Dobrą praktyką jest dokładna analiza przebiegu funkcji, aby zidentyfikować potencjalnie trudne obszary i lepiej zrozumieć naturę problemu. W miarę możliwości, warto porównać uzyskane wyniki z dokładnym wykresem funkcji, co pozwoli na lepszą ocenę jakości przybliżenia i identyfikację ewentualnych błędów.