Lab04

Max Stróżyk

2 grudnia 2024

Spis treści

Link do projektu na GitHubie

1 Zadanie 1

1.1 Charakterystyka metody Hornera

Metoda Hornera jest techniką stosowaną do szybkiego obliczania wartości wielomianu $P(x)$ w zadanym punkcie x. Główną zaletą tej metody jest redukcja liczby operacji wymaganych do obliczenia wartości wielomianu, co zmniejsza złożoność obliczeniową z $O(n^2)$ do $O(n)$, gdzie n oznacza stopień wielomianu.

Podstawową ideę algorytmu można przedstawić w następującej formie rekurencyjnej:

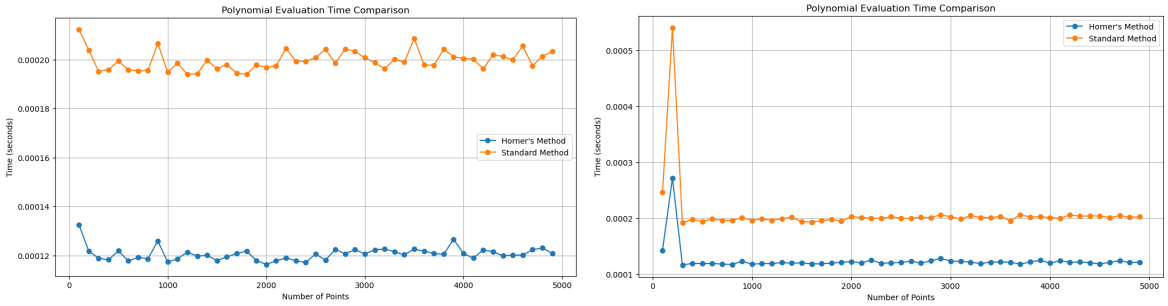
$$P(x) = a_n + x(a_{n-1} + x(a_{n-2} + \cdots + x(a_1 + xa_0)))$$

Dzięki temu każda operacja wymaga jedynie n mnożeń i n dodawań, co znacząco zmniejsza czas obliczeń dla dużych stopni wielomianów.

W przeprowadzonych testach dla wielomianów o rozmiarach od 100 do 5000, z krokiem 100, losowano 100 różnych wielomianów dla każdego stopnia. Dla każdego wielomianu mierzono czas obliczenia jego wartości metodą Hornera i standardową, a następnie obliczano średnią z uzyskanych czasów.

1.2 Wyniki i wnioski

Poniżej przedstawiono dwa typy wykresów które się pojawiały ilustrujące wyniki eksperymentów:



Wnioski:

- **Poprawa szybkości algorytmu:** Metoda Hornera pozwala na uzyskanie średniej poprawy szybkości obliczeń na poziomie 1.64 ± 0.04. Jest to zgodne z oczekiwaniami wynikającymi ze zmniejszonej złożoności obliczeniowej.
- **Nieoczekiwane skoki czasów:** W niektórych próbach zaobserwowano znaczny wzrost czasu obliczeń dla pojedynczego wielomianu. Zjawisko to nie koreluje z rzędem współczynników ani rozmiarem wielomianu, a jego przyczyna nie jest znana.
- **Skalowalność:** Metoda Hornera wykazuje dobrą skalowalność dla dużych stopni wielomianów, utrzymując stabilny czas obliczeń w porównaniu do klasycznego algorytmu.

2 Zadanie 2

2.1 Charakterystyka algorytmu Neville'a

Algorytm Neville'a to metoda numeryczna służąca do wyznaczania wartości interpolowanego wielomianu w danym punkcie. Główną zaletą tej techniki jest możliwość efektywnego obliczania wartości wielomianów na podstawie zadanego zestawu punktów, choć jej złożoność obliczeniowa wynosi $O(n^2)$, co może być kosztowne przy dużych zbiorach danych.

Interpolacja w algorytmie Neville'a bazuje na rekurencyjnym wzorze, który pozwala obliczyć wartość funkcji w punkcie x na podstawie danych punktów (x_i, y_i). Testy przeprowadzono, losując n punktów z ustalonego przedziału $D = (0, 10)$, a następnie porównując wartości interpolacji z rzeczywistymi wartościami funkcji w 100 równomiernie rozłożonych punktach.

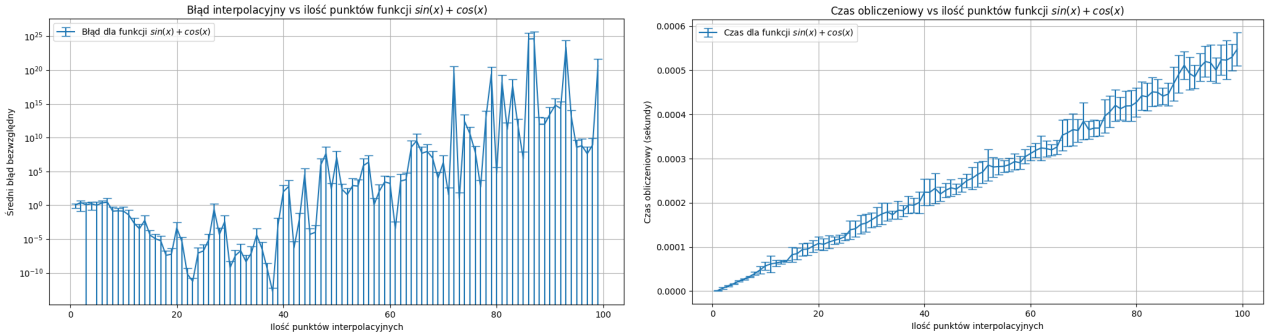
Eksperyment wykonano dla czterech funkcji:

- Funkcja trygonometryczna: $f(x) = \sin(x) + \cos(x)$,
- Wielomian 3 stopnia: $f(x) = x^3 - 2x + 1$,
- Funkcja wykładnicza: $f(x) = e^x$,
- Funkcja wymierna: $f(x) = \frac{1}{1+x^2}$.

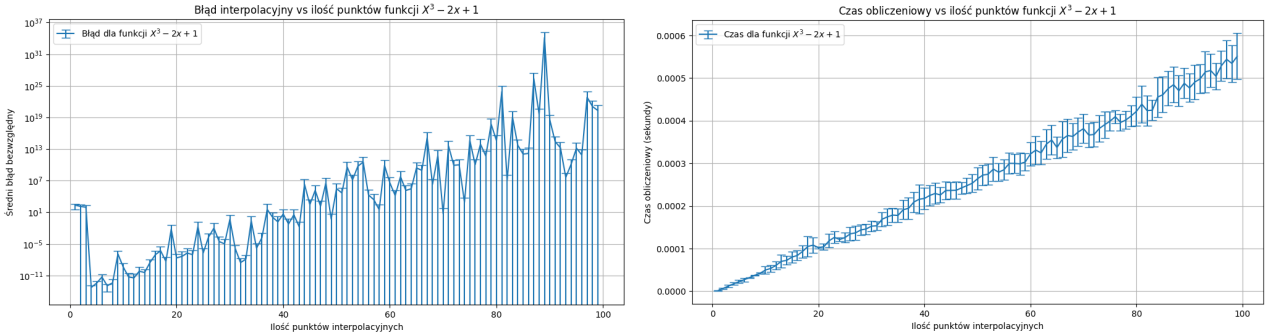
Dla każdej funkcji obliczono błąd interpolacji oraz odchylenie standardowe, a także czas wywołania algorytmu.

2.2 Wyniki i wnioski

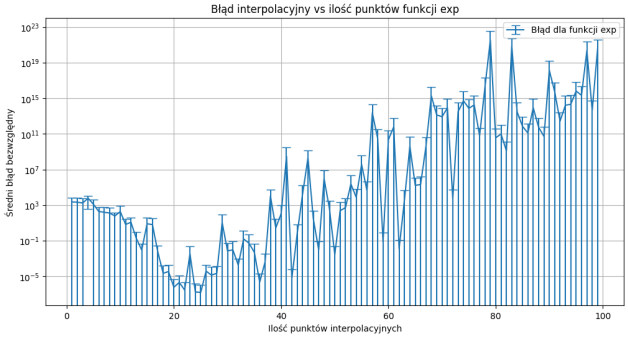
Wyniki dla funkcji trygonometrycznej: Najmnieszy błąd: $2.01 \times 10^{-13} \pm 1.03 \times 10^{-12}$ przy 23 punktach

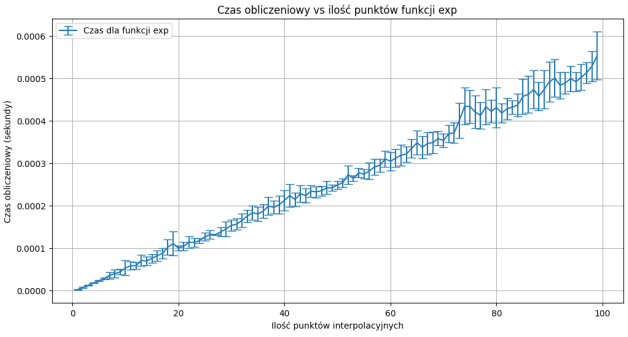


Wyniki dla wielomianu trzeciego rzędu: Najmniejszy błąd: $8.21 \times 10^{-14} \pm 1.31 \times 10^{-13}$ przy 9 punktach.

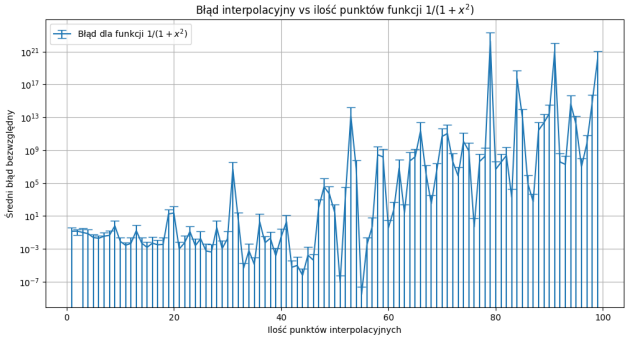


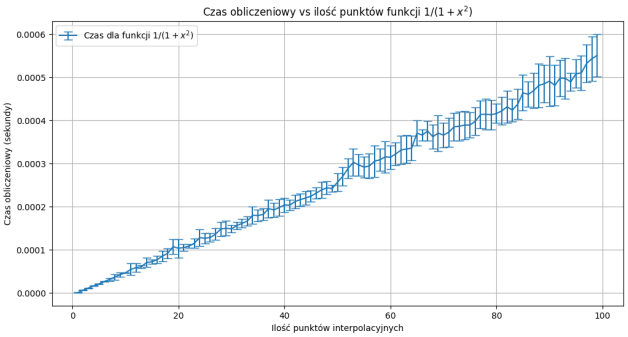
Wyniki dla funkcji exp: Najmniejszy błąd: $2.00 \times 10^{-9} \pm 1.25 \times 10^{-8}$ przy 26 punktach.





Wyniki dla funcji $\frac{1}{1+x^2}$: Najmniejszy błąd: $7.63 \times 10^{-12} \pm 3.28 \times 10^{-11}$ przy 28 punktach.





Wnioski:

- **Precyzja dla 10–20 punktów:** Przy wyborze od 10 do 20 punktów osiągano najlepsze wyniki pod względem dokładności, ponieważ przy większej liczbie punktów wzrastają błędy, głównie z powodu efektów numerycznych wynikających z dużych wartości przyrostów. Widzimy jednak duże odchylenie standardowe
- **Czas obliczeń:** Czas wykonania algorytmu Neville'a był liniowy w zależności od liczby punktów, z bardzo małym współczynnikiem przyrostu. Wynika to z implementacji.
- **Różnice między funkcjami:** Największe błędy zaobserwowano w przypadku funkcji wykładniczej, co wynika z jej szybkiego wzrostu w przedziale D. Najmniejsze błędy odnotowano dla wielomianu sześciennego, co jest zgodne z oczekiwaniami, gdyż metoda interpolacji wielomianowej jest idealna dla funkcji tego typu.
- **Odchylenie standardowe, błąd** Dziwne pokazywanie odchylenia standardowe jest spowodowane, przeskalowaniem wykresu. Głównie niesie informacje czy wyskakuje ono znacznie w górę.

3 Zadanie 3/4

3.1 Charakterystyka algorytmu Newtona

Algorytm Newtona oparty na podzielonych różnicach jest techniką służącą do interpolacji funkcji. Jego główną zaletą jest efektywne obliczanie wartości wielomianu interpolacyjnego na podstawie zadanego zbioru punktów. Złożoność algorytmu wynosi $O(n^2)$, co sprawia, że nadaje się on głównie do problemów z umiarkowaną liczbą danych wejściowych.

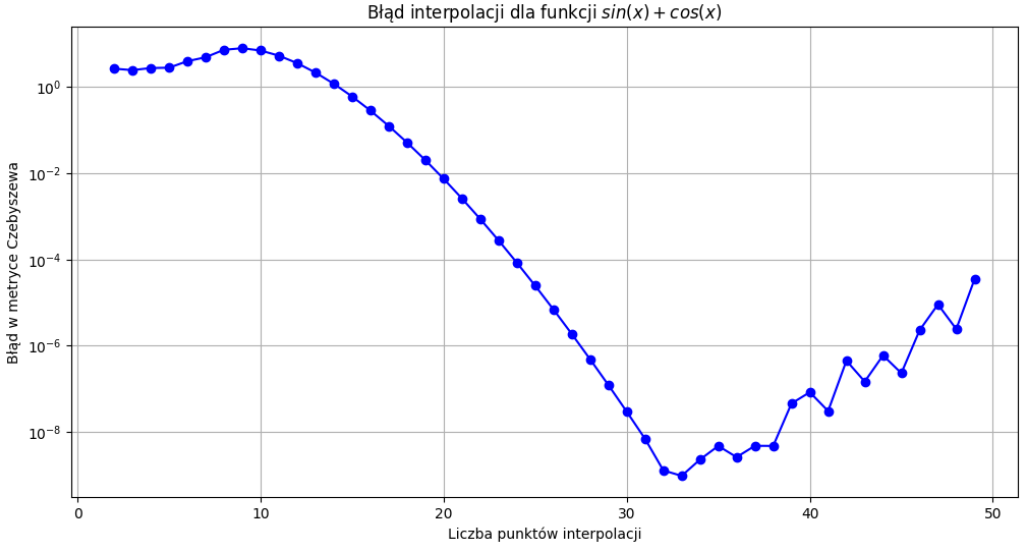
Podstawą algorytmu są podzielone różnice, które umożliwiają iteracyjne dodawanie kolejnych składników do wielomianu interpolacyjnego bez konieczności ponownego obliczania całości. W testach analizowano błędy w metryce Czebyszewa na 1000 punktach, próbie losowej n i badano, jak dobrze dopasowanie odwzorowuje funkcję.

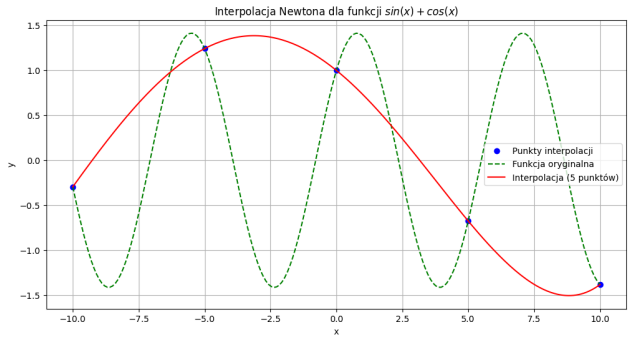
Eksperymenty przeprowadzono dla trzech funkcji:

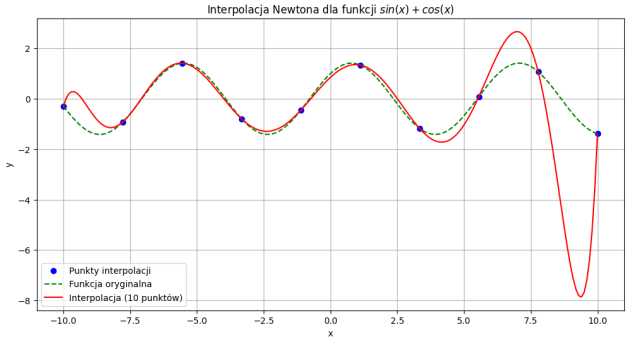
- Funkcja trygonometryczna: $f(x) = \sin(x) + \cos(x)$,
- Funkcja wymierna: $f(x) = \frac{1}{1+x^2}$,
- Funkcja nieciągła: $f(x) = \lfloor x \rfloor$.

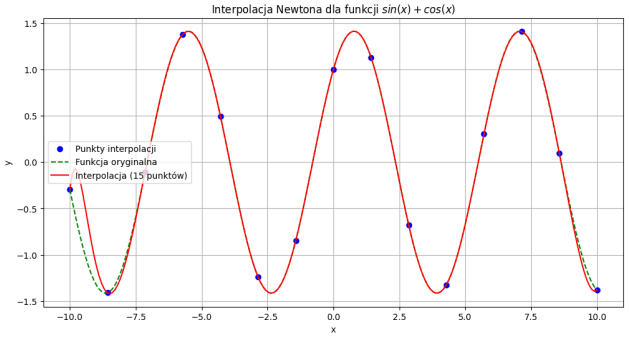
3.2 Wyniki i wnioski

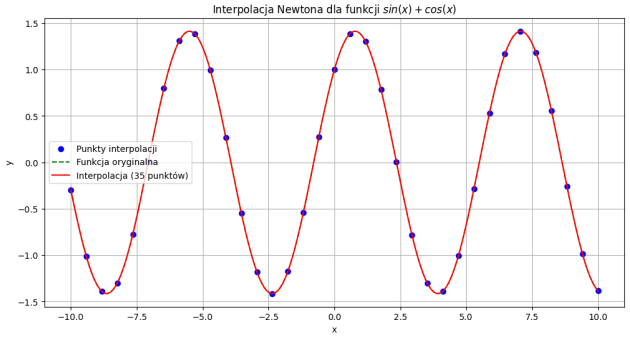
Wyniki dla funkcji $\sin(x) + \cos(x)$**:** Minimalny błąd: 9.63×10^{-10} Osiągnięty dla 33 punktów.



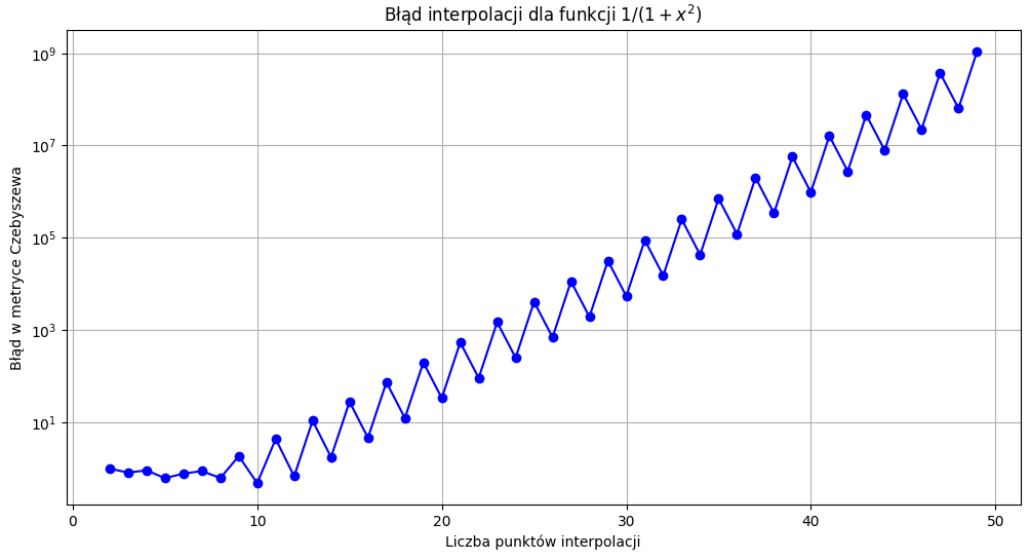


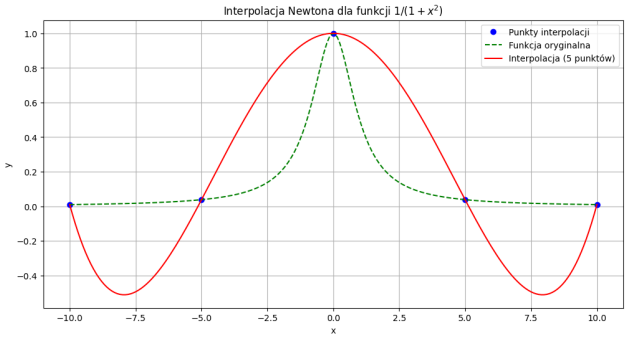


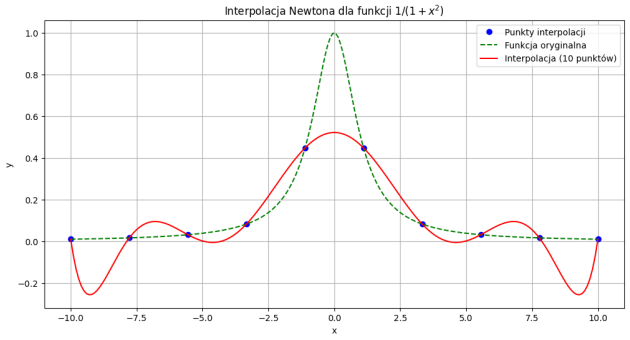


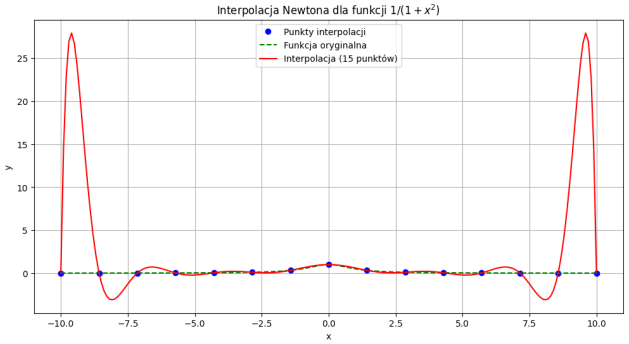


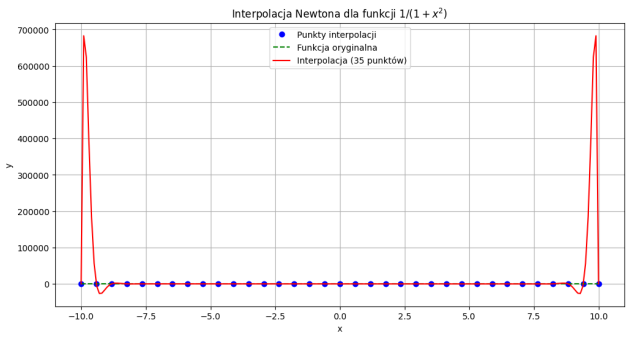
Wyniki dla funkcji $\frac{1}{1+x^2}$: Minimalny błąd: 4.78×10^{-1} Osiągnięty dla 10 punktów.



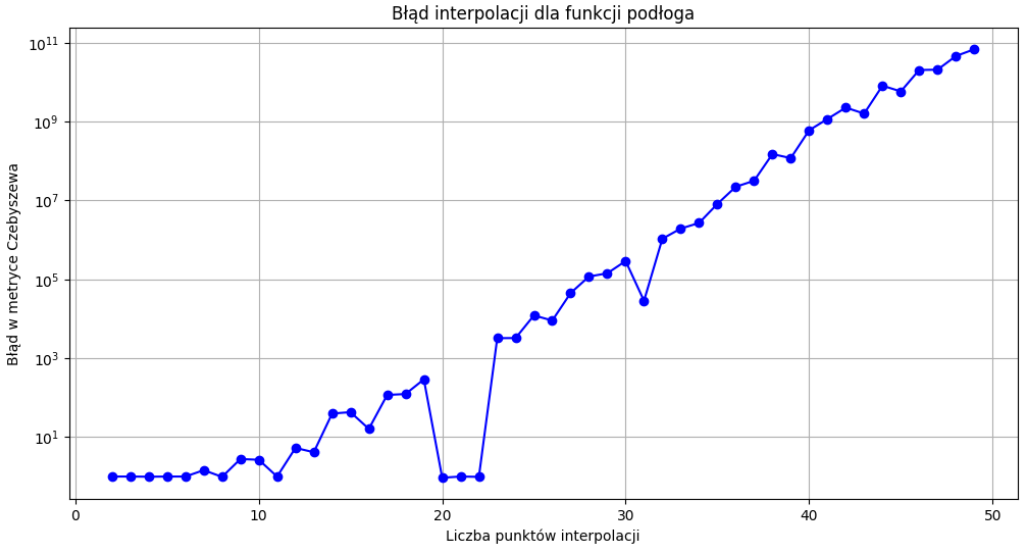


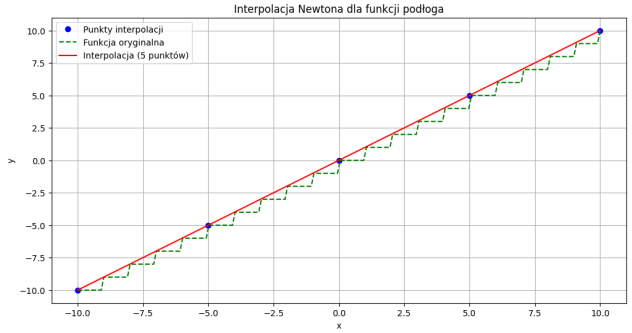


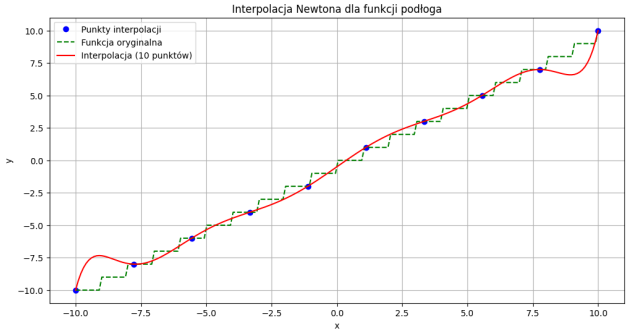


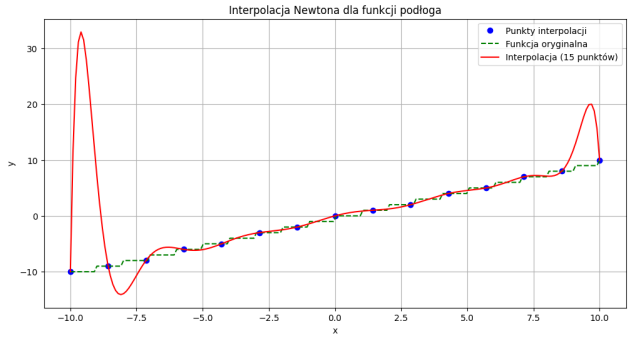


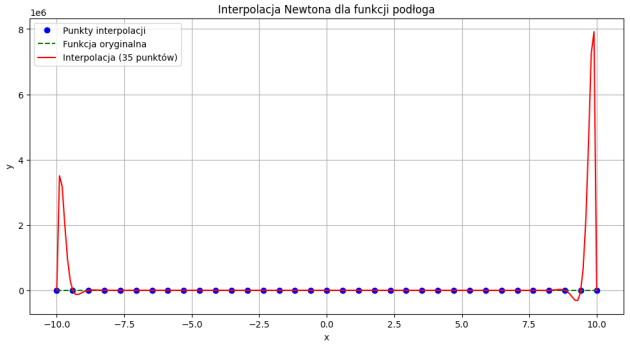
Wyniki dla funkcji $\lfloor x \rfloor$: Minimalny błąd: 9.32×10^{-1} Osiągnięty dla 20 punktów.











Wnioski:

- **Efektywność przy umiarkowanej liczbie punktów:** Dodawanie dużej liczby punktów interpolacyjnych niekoniecznie poprawia wyniki. Wręcz przeciwnie, prowadzi to do większych błędów, ponieważ algorytm próbuje dopasować wielomian, co wymaga nagłych wzrostów wartości funkcji.
- **Trudności z funkcjami nieciągłymi:** W przypadku funkcji nieciągłych, takich jak $\lfloor x \rfloor$, uzyskiwane błędy są znacznie większe, co wynika z problemów z dopasowaniem ciągłego wielomianu do nieciągłości.
- **Funkcje jednostajnie ciągłe:** Funkcje jednostajnie ciągłe, takie jak trygonometryczna i wymierna, pozwalają na uzyskanie lepszych wyników, choć nawet w tych przypadkach obserwowane są pewne ograniczenia dokładności.
- przy funkcjach mają wiele ekstermów widzimy charakterystyczny skoki, które wynikają z częstych zmian tempa przyrostu.

4 Zadanie 5

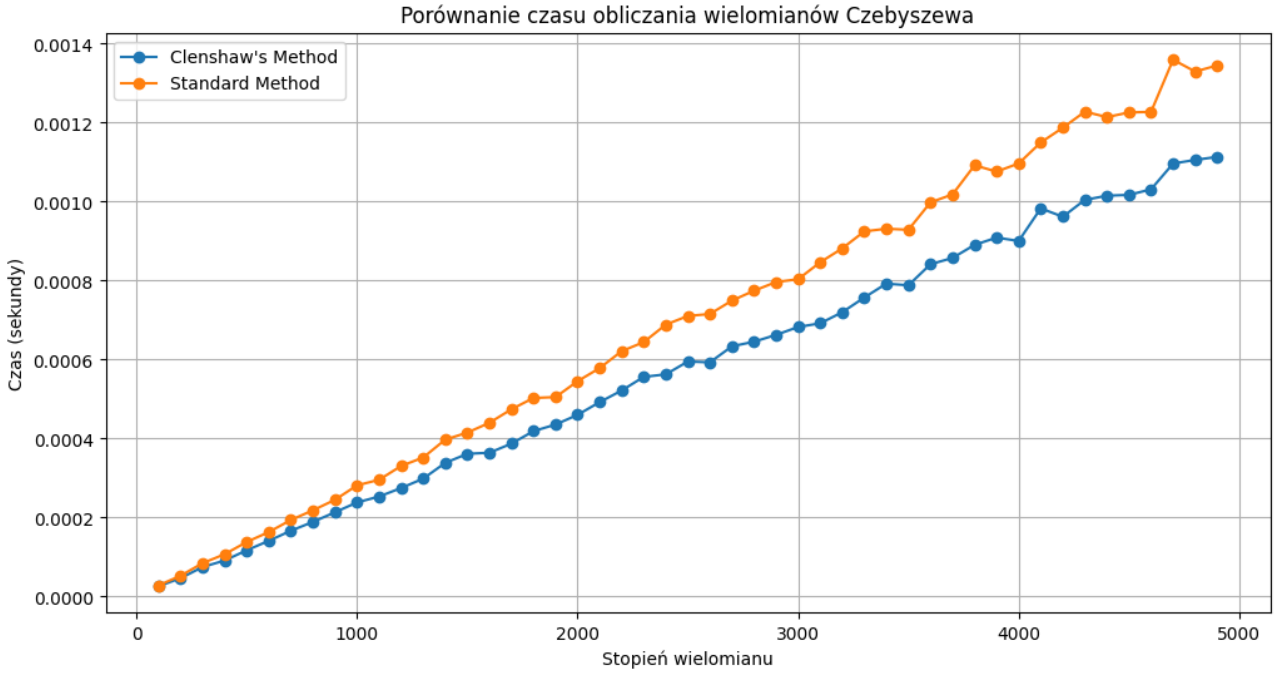
4.1 Charakterystyka algorytmu Clenshawa

Algorytm Clenshawa to efektywna metoda obliczania wartości wielomianu w postaci Czebyszewa przy złożoności $O(n)$. W porównaniu ze standardowym sposobem wyznaczania wartości wielomianów, algorytm ten jest zoptymalizowany do przetwarzania ich w postaci rekurencyjnej, co prowadzi do znacznego przyspieszenia obliczeń.

W ramach eksperymentu porównano szybkość działania algorytmu Clenshawa z klasycznym sposobem obliczania wielomianów Czebyszewa. Wyniki przedstawiono na wykresie, który uwzględnia średni czas wykonania dla obu metod. Ustawienia jak w zadaniu 1.

4.2 Wyniki i wnioski

Średni czas dla metody Clenshawa: 0.000578 ± 0.000324 **Średni czas dla metody standardowej:** 0.000692 ± 0.000393 **Przyspieszenie:** $1.20 \times \pm 0.03\times$



Wnioski:

- **Szybkość metody Clenshawa:** Algorytm Clenshawa wykazuje wyraźną przewagę nad standardową metodą liczenia wielomianów Czebyszewa, osiągając średnio 1.20x szybsze wykonanie.
- **Porównanie z Hornerem:** Metoda Clenshawa jest bliska optymalnemu czasowi obliczeń znanemu z algorytmu Hornera, który jest specjalnym przypadkiem dla wielomianów Czebyszewa.
- **Stabilność czasów:** W odróżnieniu od algorytmu Hornera, metoda Clenshawa nie wykazuje wahań w czasie wykonania, co świadczy o jej większej stabilności numerycznej.

5 Zadanie 6

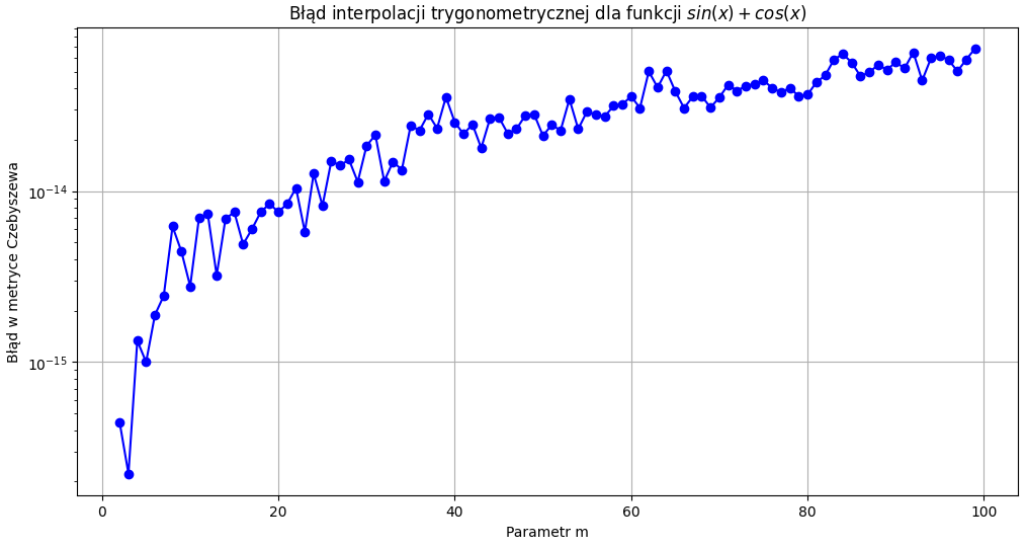
5.1 Charakterystyka algorytmu interpolacji trygonometrycznej

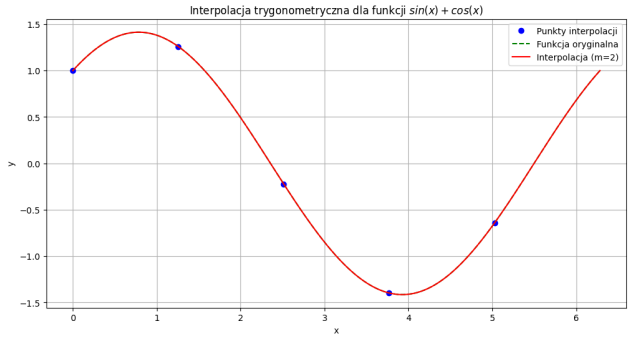
Algorytm interpolacji trygonometrycznej pozwala na aproksymację funkcji okresowych za pomocą wielomianów trygonometrycznych. Złożoność tego algorytmu wynosi $O(n \cdot m)$, gdzie n to liczba punktów interpolacyjnych ($n = 2m$), a m jest liczbą współczynników w wielomianie. W szczególności algorytm korzysta z okresowych wartości funkcji w równomiernie rozłożonych węzłach, co pozwala na wyznaczenie współczynników aproksymacji trygonometrycznej.

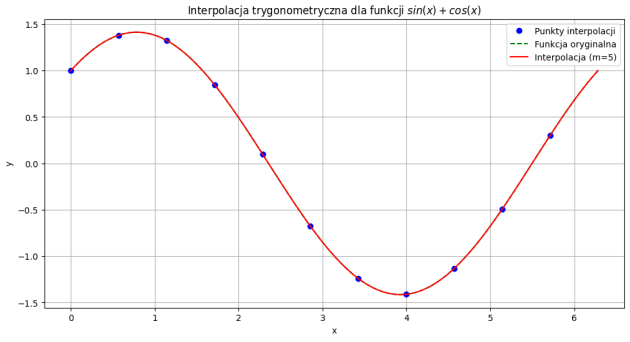
W ramach eksperymentu zbadano, jak zmienia się błąd interpolacji w zależności od liczby punktów m oraz jak dobrze algorytm odtwarza wybrane funkcje (ciągłe i nieciągłe). Wyniki zaprezentowano na wykresach. Ustawienia jak w zadaniu 3/4.

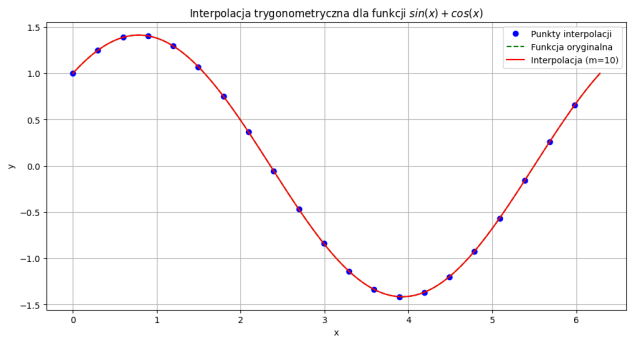
5.2 Wyniki i wnioski

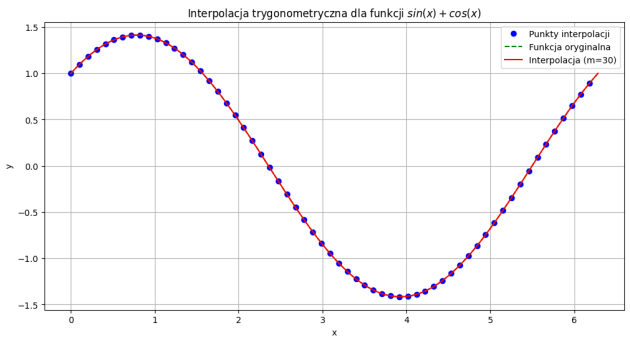
Wyniki dla funkcji $\sin(x) + \cos(x)$**:** Minimalny błąd: $2.22e-16$ Osiągnięty dla $m = 3$



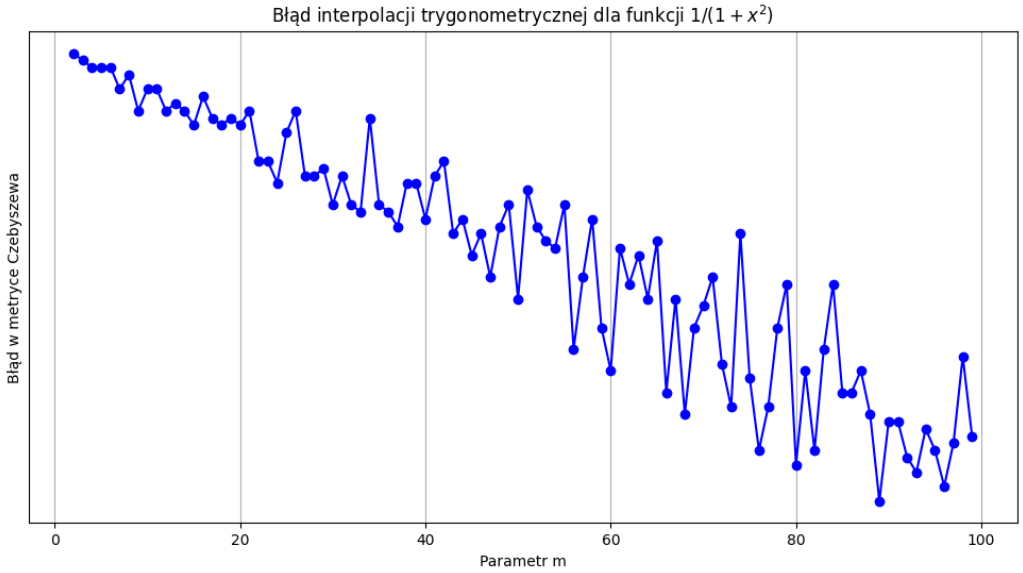


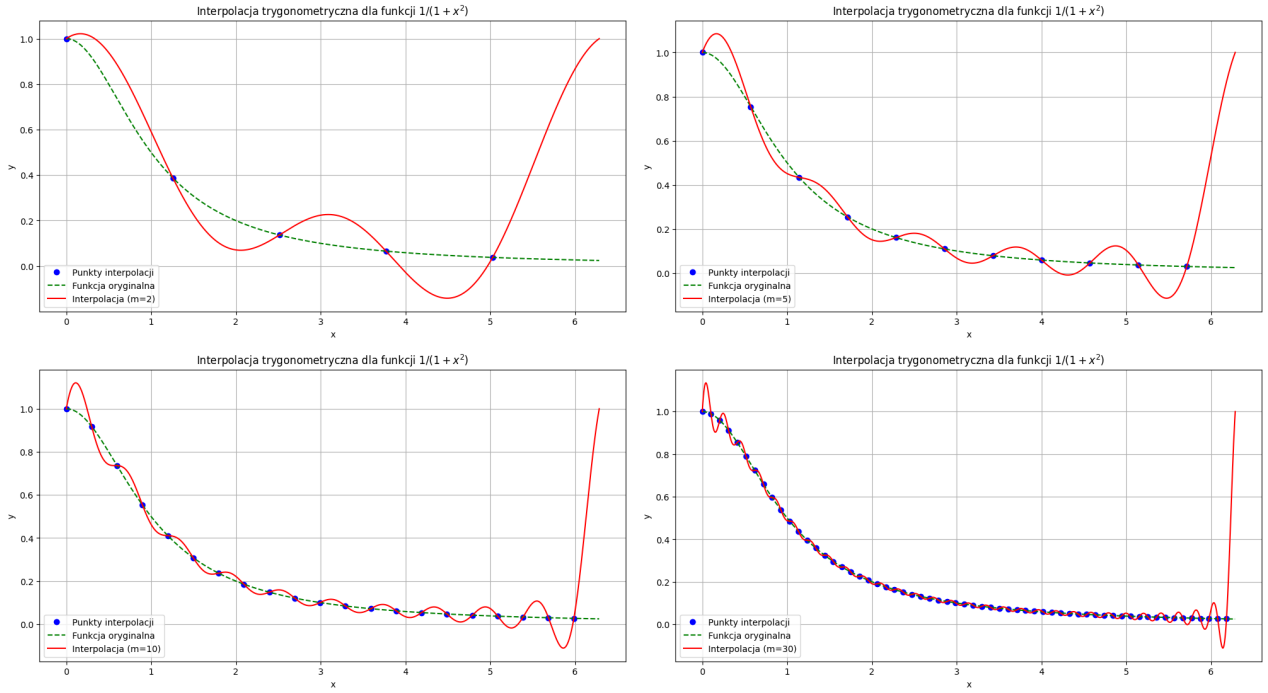




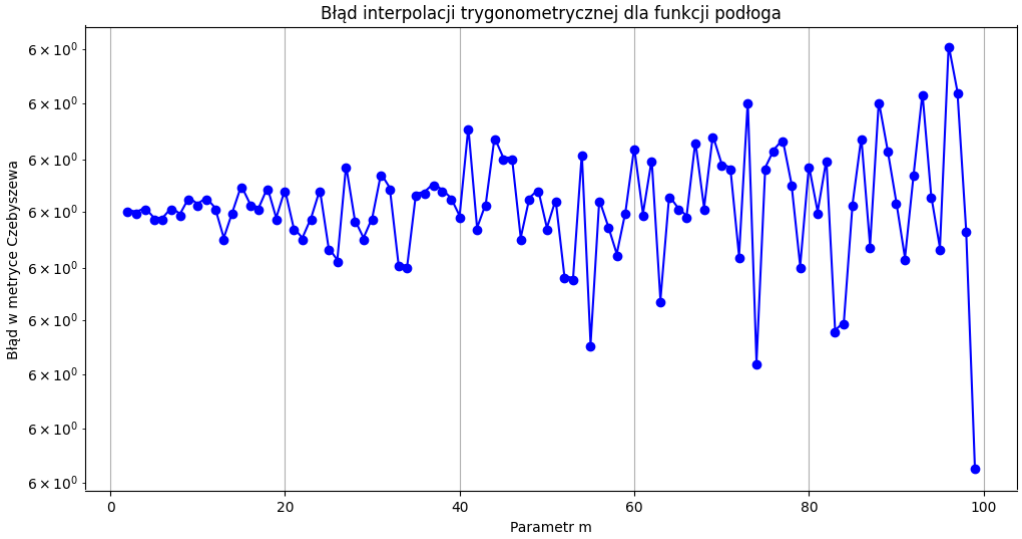


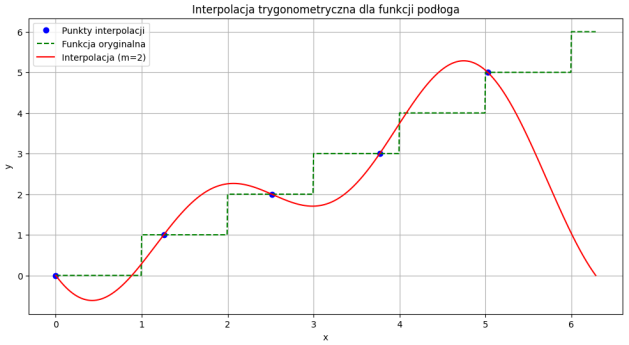
Wyniki dla funkcji $\frac{1}{1+x^2}$: Minimalny błąd: $9.75e-01$ Osiągnięty dla $m = 89$

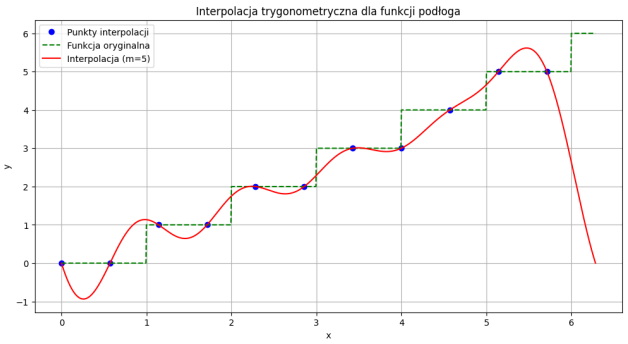


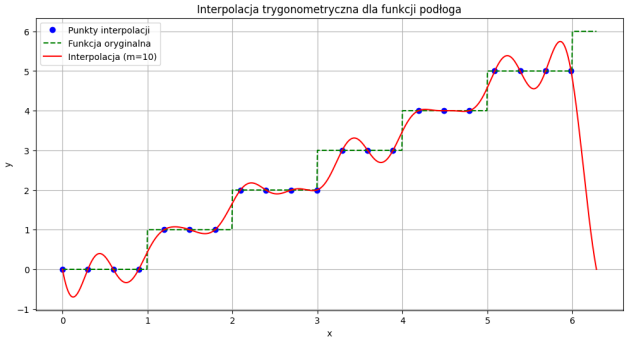


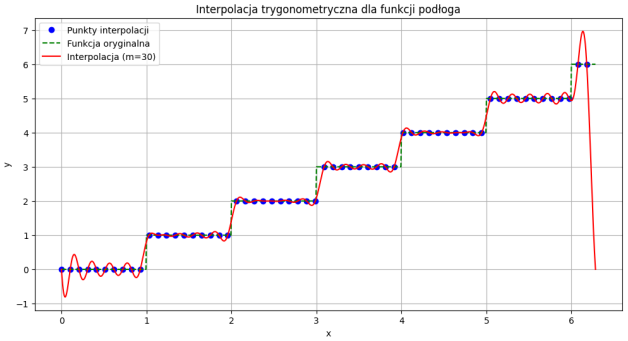
Wyniki dla funkcji podłoga: Minimalny błąd: $6.00e+00$ Osiągnięty dla $m = 99$











Wnioski:

- **Dokładność:** Algorytm interpolacji trygonometrycznej charakteryzuje się bardzo małymi błędami dla funkcji okresowych, takich jak $\sin(x) + \cos(x)$.
- **Interpolacja funkcji nieciągłych:** Pomimo swojej specyfiki, algorytm pozwala na interpolację funkcji nieciągłych (np. funkcji podłoga), choć błąd w tych przypadkach jest znacznie większy.
- **Odchylenia na końcach:** Wahania na końcu wykresów błędu wynikają z niedoskonałości implementacji, co można skorygować. W teorii, metoda powinna dawać wyniki zgodne z oczekiwaniami.
- **Uniwersalność:** Dzięki możliwości skalowania dziedziny do $[0, 2\pi]$, metoda ma szerokie zastosowania i może być wykorzystywana do analizy wielu rodzajów funkcji.