Analiza błędów
Sprawozdanie z laboratorium 1

Jakub Grześ

08.03.2024

1 Treść zadań

1.1 Zadanie 1

Oblicz przybliżoną wartość pochodnej funkcji, używając wzoru:

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

Sprawdź działanie programu dla funkcji $tan(x)$ oraz $x = 1$. Wyznacz błąd porównując otrzymaną wartość numerycznej pochodnej z prawdziwą wartością. Pomocna będzie tożsamość $tan(x) = 1 + tan^2(x)$. Na wspólnym rysunku przedstaw wykresy wartości bezwzględnej błędu metody numerycznego oraz błędu obliczeniowego w zależności od h dla $h = 10^{-k}$, $k = 0, \ldots, 16$. Użyj skali logarytmicznej na obu osiach. Czy wykres wartości bezwzględnej błędu obliczeniowego posiada minimum? Porównaj wyznaczoną wartość h_{min} z wartością otrzymaną ze wzoru:

$$h_{\min} \approx \sqrt{\frac{2 \cdot \epsilon_{\text{mach}}}{M}}, \text{ gdzie } M \approx |f''(x)|$$

Powtórz ćwiczenie używając wzoru różnic centralnych:

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

Porównaj wyznaczoną wartość h_{min} z wartością otrzymaną ze wzoru

$$h_{\min} \approx \sqrt[3]{\frac{3 \cdot \epsilon_{\text{mach}}}{M}}, \text{ gdzie } M \approx |f'''(x)|$$

1.2 Zadanie 2

Napisz program generujący pierwsze wyrazów ciągu zdefiniowanego równaniem różnicowym:

$$x_{k+1} = 2.25x_k - 0.5x_{k-1}$$

z wyrazami początkowymi:

$$x_0 = \frac{1}{3}, \quad x_1 = \frac{1}{12}$$

Wykonaj obliczenia używając pojedynczej precyzji oraz przyjmując $n = 225$, używając podwójnej precyzji oraz przyjmując $n = 60$, oraz używając reprezentacji z biblioteki fractions oraz przyjmując $n = 225$. Narysuj wykres wartości ciągu w zależności od k. Użyj skali logarytmicznej na osi y. Następnie narysuj wykres przedstawiający wartość bezwzględną błędu względnego w zależności od k. Dokładne rozwiązanie równania różnicowego

$$x_k = \frac{4^{-k}}{3}$$

maleje wraz ze wzrostem. Czy otrzymany wykres zachowuje się w ten sposób? Wyjaśnij otrzymane wyniki.

2 Metoda rozwiązania

2.1 Zadanie 1

2.1.1 Wzór 1

W tej części zadania szacowano wartość pochodnej funkcji tangens dla argumentu 1, używając wzoru (1). Korzystając z rozwinięcia funkcji w szereg Taylora:

$$f(x+h) = f(x) + f'(x) \cdot h + \frac{f''(\theta)}{2} \cdot h^2, \quad \theta \in [x, x+h]$$

$$\frac{f(x+h) - f(x)}{h} = f'(x) + \frac{f''(\theta)}{2} \cdot h, \quad \theta \in [x, x+h]$$

Błąd metody jest ograniczony przez $\frac{Mh}{2}$, gdzie M jest górnym ograniczeniem $f''(x)$ na tym przedziale. Natomiast błąd numeryczny wynika z ograniczonej precyzji reprezentacji liczb w komputerze i jest ograniczony przez $\frac{2\epsilon}{h}$, gdzie ϵ jest precyzją maszynową. Jest tak poniważ zarówno $f(x)$ jak i $f(x+h)$ mogą być obarczone tym błędem. Błąd obliczeniowy $E(h)$ można więc oszacować od góry:

$$E(h) \leq \frac{Mh}{2} + \frac{2\varepsilon}{h} \quad (3)$$

Powyższe ograniczenie przyjmuje minimum dla $h = 2\sqrt{\frac{\epsilon}{M}}$.

2.1.2 Metoda różnic centralnych

W tej części zadania oszacowano wartość pochodnej funkcji tangens dla argumentu 1, wykorzystując metodę różnic centralnych. Metoda ta wykorzystuje

rozwinięcia funkcji w szereg Taylora dla punktów $x+h$ oraz $x-h$:

$$f(x+h) = f(x) + f'(x)\cdot h + \frac{f''(x)}{2}\cdot h^2 + \frac{f'''(\theta_1)}{6}\cdot h^3, \quad \theta_1 \in [x, x+h]$$

$$f(x-h) = f(x) - f'(x)\cdot h + \frac{f''(x)}{2}\cdot h^2 - \frac{f'''(\theta_2)}{6}\cdot h^3, \quad \theta_2 \in [x-h, x]$$

Przybliżenie wartości pochodnej funkcji f w punkcie x metodą różnic centralnych jest następujące:

$$\frac{f(x+h) - f(x-h)}{2h} = f'(x) + \frac{f'''(\theta_1) + f'''(\theta_2)}{2}\cdot\frac{h^2}{6}$$

Błąd metody jest ograniczony przez $\frac{Mh^2}{6}$, gdzie M jest maksymalną wartością trzeciej pochodnej funkcji f w rozważanym przedziale. Natomiast błąd numeryczny przez $\frac{\varepsilon}{h}$. Całkowity błąd obliczeniowy $E(h)$ dla metody różnic centralnych można ograniczyć:

$$E(h) \leq \frac{Mh^2}{6} + \frac{\varepsilon}{h}$$

Powyższe ograniczenie przyjmuje minimum dla $h = \sqrt[3]{\frac{3\epsilon}{M}}$.

2.2 Zadanie 2

W drugiej części wystarczyło wykonać potrzebne obliczenia, zwizualizować je oraz przeanalizować.

3 Reprezentacja wyników, fragmenty algorytmu

3.1 Zadanie 1

Do obliczeń wykorzystano tożsamość $tan'(x) = 1 + tan^2(x)$ oraz środowisko Python i jego bibliotekę Numpy. Obliczenia wykonano dla $h = 10^{-k}, k = 0, ..., 16$.

```
def f(x):
    return np.tan(x)

def f_real_derivative(x):
    return 1 + np.tan(x)**2

def second_tan_derivative(x):
    return 2 * (np.tan(x)**2 + 1) * np.tan(x)

x = 1
epsilon = np.finfo(float).eps
```

```
M = second_tan_derivative(x)
h_min_theoretical = 2 * np.sqrt(epsilon / M)
h_values = np.logspace(0, -16, 17, base=10)

method_errors = []
numerical_errors = []
computational_errors = []

for h in h_values:
    numerical_derivative = (f(x + h) - f(x)) / h
    real_derivative = f_real_derivative(x)

    method_error = M * h / 2
    method_errors.append(method_error)

    numerical_error = 2 * epsilon / h
    numerical_errors.append(numerical_error)

    computational_error = abs(numerical_derivative - real_derivative)
```

3.1.1 Otrzymane wyniki

Wykresy przedstawiają zależność błędów metody numerycznej od wielkości kroku h. Zastosowano skalę logarytmiczną na obu osiach.

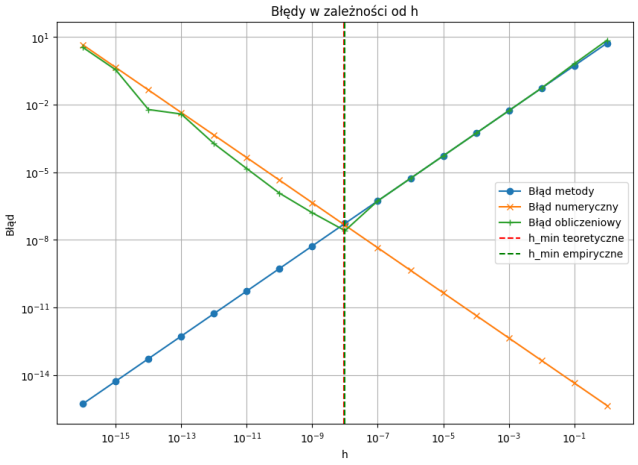

Figure 1: Metoda różnic progresywnych

Różnica między teoretycznym h_{min} a wyznaczonym empirycznie wynosi: 8.8e-10, błąd względny: 9.6%. Błąd dla h_{min} wyniósł 2.6e-08.

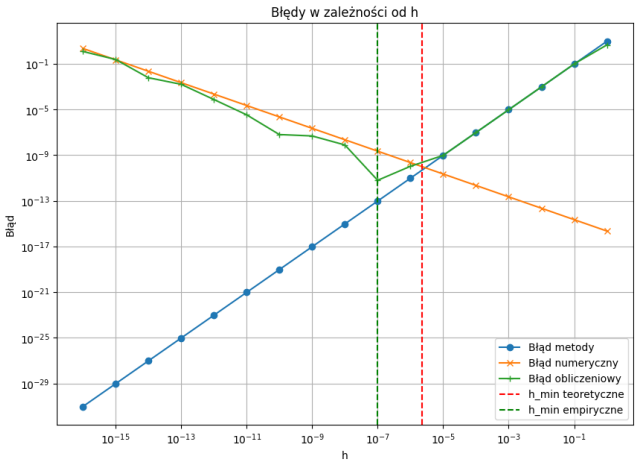

Figure 2: Metoda różnic centralnych

Różnica między teoretycznym h_{min} a wyznaczonym empirycznie wynosi: 2.2e-06, błąd względny: 95.6%. Błąd dla h_{min} wyniósł 6.2e-12.

Dla obu metod zostało wyznaczone empirycznie h, dla którego otrzymany błąd jest najmniejszy. Empiryczne oszacowanie jest znacznie bliższe teoretycznemu dla metody wykorzystującej wzór z zadania 1 niż dla metody różnic centralnych. Błąd otrzymany dla pierwszej metody jest jednak ponad 4100 razy większy. Oznacza to, że mimo poprawności obu metod ta dla różnic centralnych daje znacznie lepszy rezultat.

3.2 Zadanie 2

3.2.1 Generowanie ciągu

Aby wykonać zadanie 2 wykonano poniższy kod.

```
def generate_sequence(n, d_type):
    if n < 2:
        raise ValueError('n must be greater or equal 2')
    seq = np.zeros(n, dtype=d_type)
    seq[0] = d_type(1/3)
    seq[1] = d_type(1/12)

    for i in range(2, n):
        seq[i] = d_type(2.25) * seq[i - 1] - d_type(0.5) * seq[i - 2]
    return seq
```

Wartości poszczególnych wyrazów ciągu były reprezentowane za pomocą trzech różnych typów danych o zróżnicowanej precyzji: float32, float64 oraz Fraction z biblioteki fractions. W trakcie obliczeń z użyciem typu float32 wystąpiły ostrzeżenia 'overflow', sygnalizujące przekroczenie maksymalnej wartości, którą może przechowywać typ zmiennoprzecinkowy. Pojawił się również komunikat 'invalid value', który jest rezultatem próby przeprowadzenia operacji na wartościach, które nie są prawidłowymi liczbami. Te ostrzeżenia wskazują na ograniczenia typów zmiennoprzecinkowych w zakresie reprezentacji dużych liczb oraz na akumulację błędów numerycznych. Na dołączonym wykresie prezentowane są wartości kolejnych wyrazów ciągu, co ilustruje zachowanie się reprezentacji numerycznej w zależności od wybranego typu danych.

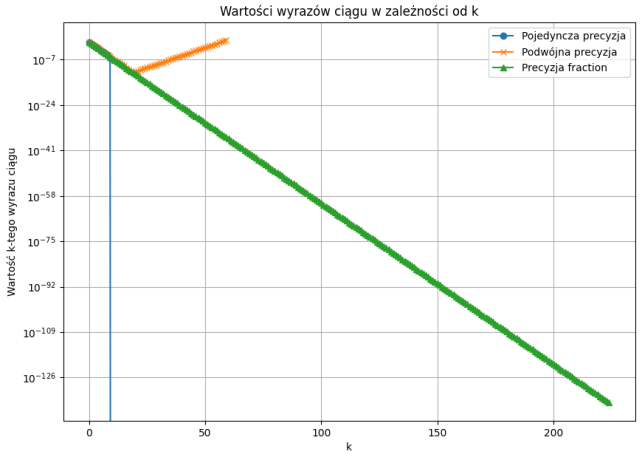

Figure 3: Wspólny wykres wartości wyrazów wygenerowanych ciągów

Jak obserwujemy, pomimo wykorzystania identycznego wzoru rekurencyjnego dla generowania wyrazów ciągu, tylko reprezentacja typu fraction zachowuje oczekiwany schemat zmienności. Typy zmiennoprzecinkowe float64 i float32 po przekroczeniu pewnego momentu zaczynają przejawiać znaczną niestabilność obliczeniową. Jest to efekt mniejszej dokładności reprezentacji numerycznej w porównaniu do typu fraction, co sprawia, że są one bardziej podatne na akumulację błędów zaokrąglenia. Reprezentacja typu fraction zapewnia wyższą precyzję niż system liczbowy zmiennoprzecinkowy wykorzystywany w typach float. Dzięki temu, nawet przy długich ciągach obliczeniowych, zachowuje stabilność i precyzję.

3.2.2 Błąd względny w zależności od k

Aby dokładniej zbadać podatność na błędy skorzystano z dokłanego rozwiązania różnicowego:

$$x_k = \frac{4^{-k}}{3}$$

a następnie obliczono błąd względny dla kolejnych wyrazów utworzonych ciągów ze wzoru:

$$\frac{f(x) - f(x_0)}{f(x)}$$

Wizualizację przedstawiono poniżej.

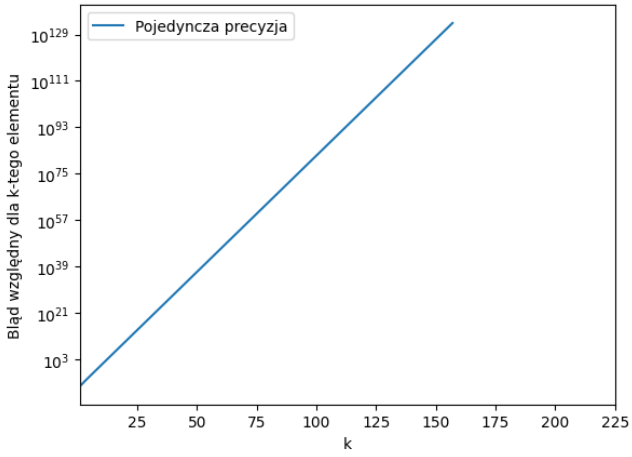

Figure 4: Błędy względne dla pojedynczej precyzji

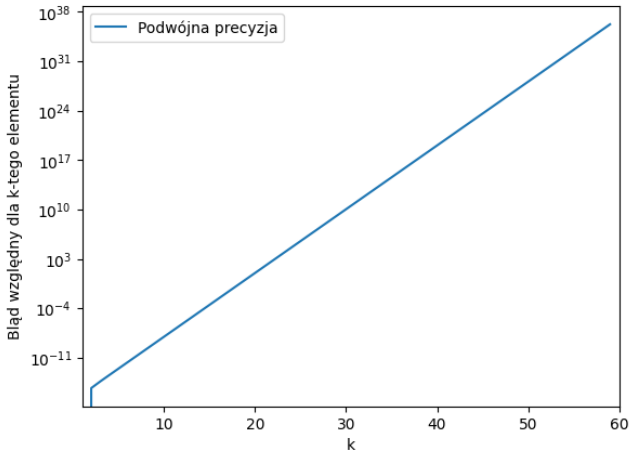

Figure 5: Błędy względne dla podwójnej precyzji

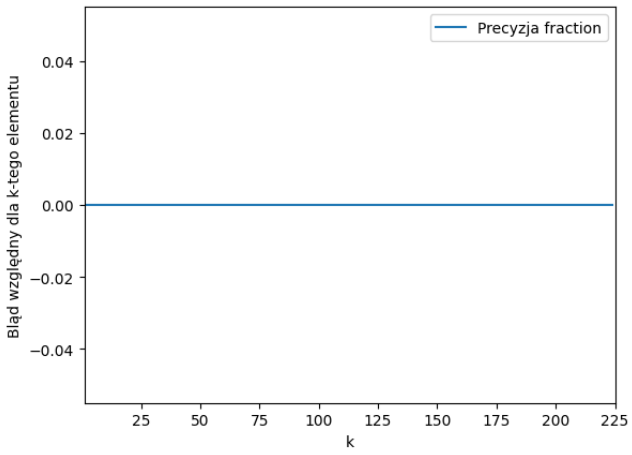

Figure 6: Błędy względne dla precyzji fraction

4 Wnioski

Podczas wykonywania zadań laboratoryjnych zauważono znaczący wpływ błędów wynikających z korzystania z metod numerycznych. Mimo że rozwiązania analityczne były poprawne, niedoskonałości arytmetyki komputerowej mogą znacząco wpłynąć na wyniki uzyskane w drodze obliczeń numerycznych. Ulepszenie dokładności reprezentacji liczbowej lub zastosowanie alternatywnej metody może przyczynić się do lepszego przybliżenia oczekiwanego rozwiązania. Jest zatem kluczowe, aby starannie dobierać środowisko obliczeniowe i poziom dokładności w zależności od specyfiki rozwiązywanego problemu.

Bibliografia

[1] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, Travis E. Oliphant. *NumPy: A guide to NumPy*, 2020, `http://www.numpy.org/`, Version 1.19.

[2] Marcin Kuta, *Error analysis*.