Projekt nr 1
Metody Numeryczne

Politechnika Warszawska
Wydział Matematyki
i Nauk Informacyjnych
Semestr 2024Z

Imię i Nazwisko	Numer albumu
Wiktoria Kawa	333141

Warszawa, 9 Grudnia 2024

Spis treści

1 | Wprowadzenie

W ramach 1. Projektu z przedmiotu ***Metody Numeryczne*** było zaimplementowanie za pomocą środowiska programistycznego ***MATLAB*** rozwiązania zadania danego na labolatoriach.

1.1 | Temat projektu

Celem tego projektu było wyznaczenie metody trapezów obliczania przybliżonej wartości całki

$$\int_a^b w_n(x)\,dx,$$

gdzie w_n ma postać:

$$w_n = \sum_{k=0}^{n} a_k T_k(x) U_k(x)$$

I do obliczania wartości wielomianu w_n należy wykorzystać związek rekurencyjny spełniany przez wielomiany Czebyszewa.

1.2 | Zrozumienie zadania

Na samym początku warto jest pochylić się nad tym. Co w ogóle oznacza ten z początku wyglądający na skomplikowany wzór.

By zrozumieć, co mamy zrobić, należy zagłębić się w uwagę daną w zadaniu, czyli w wielomiany Czebyszewa. Od razu dowiadujemy się, że:

- $T_k(x)$ - wielomian Czebyszewa pierwszego rodzaju
- $U_k(x)$ - wielomian Czebyszewa drugiego rodzaju
- a_k - wspólczynnik k-tego wielomianu

1.3 | Co będziemy robić

Na samym początku warto dowiedzieć się czym są Wielomiany Czebyszewa i tego jak są zdefiniowane, aby móc ich sensownie użyć. Potem przejdziamy do tego, jak można je zaimplementować tak, by łączyły się one dobrze z metodą trapezów obliczania przybliżonej wartości całki. Pochylimy się też oczywiście nad samą metodą z treści zadania. Zatem projekt obejmuje:

- Wielomiany Czebyszewa
- Metoda trapezów
- Implementacja
- Sprawdzenie poprawności
- Wnioski

Serdecznie zapraszam do lektury.

2 | Wielomiany Czebyszewa

2.1 | Czym są wielomiany Czebyszewa

Wielomiany Czebyszewa są zdefiniowane (dla $k \in \mathbb{N}$) w następujący sposób:

2.1.1 | Wielomiany Czebyszewa pierwszego rodzaju

Wielomiany pierwszego rodzaju oznaczamy jako $T_n(x)$. Definiuje się je rekurencyjnie

- $T_0(x) = 1$
- $T_1(x) = x$
- $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$

Dla x zdefiniowanego na przedziale [-1, 1] możemy także napisać, że:

- $T_k(x) = cos(karccos(x))$

2.1.2 | Wielomiany Czebyszewa drugiego rodzaju

Wielomiany pierwszego rodzaju oznaczamy jako $U_n(x)$. Podobnie jak te pierwszego rodzaju, definiuje się je rekurencyjnie

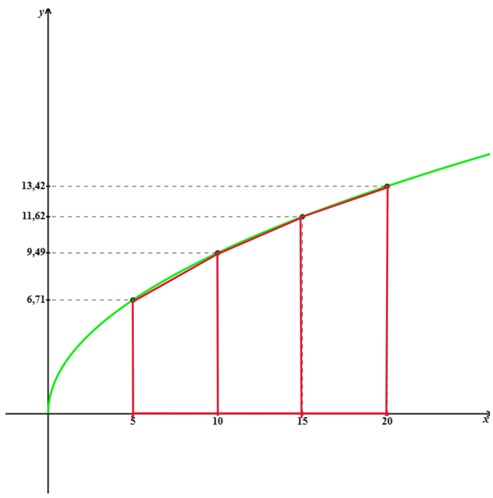
- $U_0(x) = 1$
- $U_1(x) = 2x$
- $U_k(x) = 2xU_{k-1}(x) - U_{k-2}(x)$

Dla x zdefiniowanego na przedziale (-1, 1) możemy także napisać, że:

- $U_k(x) = \frac{sin((k+1)arccos(x))}{sin(arccos(x))}$

3 | Metoda Trapezów

Metoda trapezów używana jest do obliaczania przybiżonej wartości całki. Wiemy, że całka to pole pod wykresem funkcji.



Rysunek 3.1: Przykładowa funkcja

Metoda trapezów polega na tym, że przedział całkowania $[a, b]$ dzielimy na $n + 1$ równoodległych punktów. Odległość między sąsiednimi punktami wynosi wówczas $h = \frac{b-a}{n}$.

Wyznaczając $f(x_i)$ oraz $f(x_{i+1})$ dla pewnych dwóch sąsiednich punktów z naszego podziału, jesteśmy w stanie policzyć pole pojedynczego trapezu. Będzie ono wynosić $P_i = \frac{h}{2}[f(x_i) + f(x_{i+1})]$. Przybliżona wartość całki to będzie wówczas $\sum_{i=1}^{n} P_i$.

4 | Implementacja

Po przejściu przez teorię, pora przejść do implementacji. Oczywiście nie będzie tu żadnych kodów, aby nie zanudzać czytelnika.

4.1 | Funkcje: `czebyszew_T, czebyszew_U`

Ponieważ obie z tych funkcji, tak jak widać w samej definicji Wielomianów Czebyszewa, mają podobną metodę powstawania (tak samo zdefiniowana rekurencja), to zostaną omówione wspólnie.

4.1.1 | Przyjmowane argumenty

Funkcje te przyjmują argumenty $\mathbf{X}, n$.

- $\mathbf{X}$ - wektor uwtorzony przez x należące do podziału przedziału $[a, b]$.
- n - stopień naszego w_n z treści zadania.

4.1.2 | Zwracany wynik

Zwrócona zostanie nam macież $\mathbb{T}$ (odpowiednio $\mathbb{U}$). Przyjmując, że $N + 1 =$ ilość elementów w $\mathbf{X}$ (później przyjmiemy, że N to ilość podpodziałów przedziału całkowania, zatem elementów w $\mathbf{X}$ jest $N + 1$), macierze te będą miały wymiary $N + 1$ x n. Zwrócona macierz $\mathbb{T}$ ($\mathbb{U}$ wygląda dokładnie tak samo, tylko zamiast Wielomianów Czebyszewa Pierwszego Rodzaju mamy Wielomiany Czebyszewa Drugiego Rodzaju):

$$\mathbb{T} = \begin{bmatrix} T_0(x_1) & T_1(x_1) & ... & T_n(x_1) \\ T_0(x_2) & T_1(x_2) & ... & T_n(x_2) \\ . & & & \\ . & & & \\ .T_0(x_{N+1}) & T_1(x_{N+1}) & ... & T_n(x_{N+1}) \end{bmatrix}$$

4.2 | Funkcja `eval_poly`

Funkcja ta tworzy nam wartościowanie naszego wielomianu, czyli zwraca wartości naszego wielomianu dla punktów naszego podziału - wektora $\mathbf{X}$.

4.2.1 | Przyjmowane argumenty

- $\mathbf{A}$ - wektor złożony ze współczynników funkcji
- $\mathbf{X}$ - wektor złożony z x będących punktami podziału przedziału $[a, b]$

4.2.2 | Działanie funkcji i zwracany wynik

Funkcja ta mnoży nam komórkowo macierze $\mathbb{T}$ i $\mathbb{U}$. Następnie nowopowstałą macierz mnożymy przez wektor pionowy współczynników $\mathbf{A}$. Powstała przez tę operację macierz, jest naszym wynikiem. Są to wartości funckji $w_n(x)$ w punktach naszego podziału.

$$\begin{bmatrix} T_0(x_1)U_0(x_1) & T_1(x_1)U_1(x_1) & ... & T_n(x_1)U_n(x_1) \\ T_0(x_2)U_0(x_2) & T_1(x_2)U_1(x_2) & ... & T_n(x_2)U_n(x_2) \\ . & & & \\ . & & & \\ .T_0(x_{N+1})U_0x_{N+1}) & T_1(x_{N+1})U_1(x_{N+1}) & ... & T_n(x_{N+1})U_n(x_{N+1}) \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ a_2 \\ . \\ . \\ a_n \end{bmatrix} = \begin{bmatrix} w_n(x_1) \\ w_n(x_2) \\ . \\ . \\ w_n(x_{N+1}) \end{bmatrix}$$

4.3 | Funkcja `trap_method`

Jest to nasza docelowa funkcja, która ma nam policzyć naszą całkę metodą trapezów. Zatem

4.3.1 | Przyjmowane argumenty

- **A** - współczynniki
- a - początek przedziału całkowania
- b - koniec przedziału całkowania
- N - ilość naszych podprzedziałów utworzonych przez dwa sąsiadujące ze sobą argumenty (x_i, x_{i+1})

4.3.2 | Działanie funkcji i zwracany wynik

Na początku oczywiście możemy sobie wyznaczyć "wysokość"naszych trapezów. Wiemy że długość podprzedziału możemy wyrazić wzorem $h = \frac{b-a}{N}$.

Wyznaczamy nasz wektor $\mathbf{X} = a : h : b$. Jest to zdefiniowany w języku MATLAB wektor, którego pierwszy element to a i każdy kolejny to $a + h$, $a + 2h$, ... , b

$\sum_{i=1}^{N} P_i$ - zdefiniowane wcześniej, to po rozwinięciu:

$$\sum_{i=1}^{N} P_i = \frac{h}{2}[(w_n(x_1) + w_n(x_2)) + (w_n(x_2) + w_n(x_3)) + ... + (w_n(x_N) + w_n(x_{N+1}))] \tag{4.1}$$

$$\sum_{i=1}^{N} P_i = \frac{h}{2}[w_n(x_1) + 2w_n(x_2) + 2w_n(x_3) + ... + 2w_n(x_N) + w_n(x_{N+1})] \tag{4.2}$$

$$\sum_{i=1}^{N} P_i = h(\sum_{i=1}^{N} w_n(x_i) - \frac{w_n(x_1) + w_n(x_{N+1})}{2}) \tag{4.3}$$

Wynik ten jest przybliżonym wynikiem całki zadanej w treści zadania:

$$\int_a^b w_n(x)\,dx$$

gdzie w_n ma postać:

$$w_n = \sum_{k=0}^{n} a_k T_k(x) U_k(x)$$

Funkcja `trap_method` zwraca nam przybliżoną wartość całki. Należy jeszcze oczywiście sprawdzić poprawność tej funkcji, dlatego właśnie tym się teraz zajmiemy.

5 | Sprawdzenie wyników

W tym rozdziale sprawdzimy działanie naszej funkcji `trap_method`.

5.1 | Przedstawienie przykładowych wyników

Warto sprawdzić, czy wyniki naszej funckji pokrywają się z wbudowaną funkcją do liczenia całek `integral`. Wyniki podane w tabeli zostały policzone w skrycie testującym `test_wyniki.m`. Poniżej tabeli znajduje się legenda dotyczące nazw kolumn.

$f(x)$	p	N	exact value	`integral` value	`trap_method` value	błąd `integral`	błąd `trap_method`
3	$a = 1, b = 2$	1	3	3	3	0	0
3	$a = 1, b = 2$	10	3	3	3	0	0
3	$a = 1, b = 2$	100	3	3	3	0	0
$2x^2 + 1$	$a = 1, b = 2$	1	$\frac{17}{3}$	5.6667	6	1.78×10^{-15}	0.33333
$2x^2 + 1$	$a = 1, b = 2$	10	$\frac{17}{3}$	5.6667	5.67	1.78×10^{-15}	0.0033333
$2x^2 + 1$	$a = 1, b = 2$	100	$\frac{17}{3}$	5.6667	5.6667	1.78×10^{-15}	3.33×10^{-5}
$2x^2 + 1$	$a = 1, b = 2$	9999999	$\frac{17}{3}$	5.6667	5.6667	1.78×10^{-15}	8.88×10^{-16}
x^2	$a = 1, b = 2$	100	$\frac{7}{3}$	2.3333	2.3334	4.44×10^{-16}	3.3334
x^2	$a = -1, b = 1$	100	$\frac{2}{3}$	0.66667	0.6668	1.11×10^{-16}	2.6669
$8x^4 - 6x^2 + 1$	$a = -1, b = 1$	100	1.2	1.2	1.2013	2.22×10^{-16}	0.5332

Tabela 5.1: Porównanie wyników obliczeń numerycznych

- $f(x)$ **-** Funkcja dla której liczymy całkę
- **p -** Zakres obliczeń, np. $a = 1, b = 2$ oznacza przedział od 1 do 2.
- N **-** Liczba przedziałów, na które dzielony jest przedział obliczeniowy.
- **exact value:** Dokładna wartość całki, obliczona analitycznie.
- **`integral` value -** Wartość całki obliczona za pomocą wbudowanej funkcji `MATLAB`
- **`trap_method` value -** Wartość całki obliczona za pomocą metody trapezów.
- **błąd `integral` -** różnica pomiędzy exact value a wartością `integral` value.
- **błąd `trap_method` -** różnica pomiędzy exact value a wartością `trap_method` value.

5.2 | Wnioski

Jesteśmy w stanie zauważyć, że nasza funkcja działa poprawnie. Dodatkowo z tak rozpisanymi wynikami i obliczoną różnicą między dokładnymi wartościami całek, a tymi wynikami powstałymi przez zastosowanie funkcji, jesteśmy w stanie sprawdzić, z jaką dokładnością funkcje te liczą.

Możemy zauważyć, że dla odpowiednio dużych N, czyli np. w miejscu w którym liczba naszych podpodziałow wynosi 9999999, funkcja `trap_method` jest dokładniejsza od wbudowanej w program MATLAB funkcji `integral`.

Możemy oczywiście wywnioskować też z tego wniosek taki, że dla małych N, np. dla $N = 1$, nasza funkcja podaje znacząco niedokładny wynik.

5.3 | Podsumowanie

Poprawnie zaimplementowana funkcja `trap_method` poprawnie oblicza nam przybliżoną wartość całki. Aby zwiększyć dokładność obliczeń warto jest oczywiście robić to dla dużej liczby podziałów naszego przedziału całkowania.