Laboratorium 5

Całkowanie numeryczne

Bartłomiej Szubiak

9.04.2024

Zadania

Zad 1

Obliczyć I = 0∫1 1/(1+x) dx wg wzoru prostokątów, trapezów i wzoru Simpsona (zwykłego i złożonego n=3, 5). Porównać wyniki i błędy..

Wartość całki obliczona analitycznie:

$$S(f) = \int \frac{1}{1+x} dx = ln(1+x) \ + \ C, \quad \Rightarrow \quad [ln(1+x)]_0^1 = ln2 \approx 0.69314718056$$

1. **Metoda prostokątów**, dla a=0 i b=1:

$$S(f) = \frac{1}{n} \sum_{i=0}^{n-1} f(x_i + \frac{1}{2n})$$

n	S(f)	Δ
1	0.6666666666666666	0.02648051389327865
3	0.6897546897546897	0.00339249080525561
5	0.6919078857159353	0.00123929484400997

Tabela 1.1: Wartość i błąd obliczonej całki z użyciem metody prostokątów

2. **Metoda trapezów**, dla a=0 i b=1:

$$S(f) = \frac{1}{2n} \sum_{i=0}^{n-1} [f(x_i) + f(x_{i+1})]$$

n	S(f)	Δ
1	0.75	0.05685281944005471
3	0.7	0.00685281944005466
5	0.6956349206349206	0.00248774007497532

Tabela 1.2: Wartość i błąd obliczonej całki z użyciem metody trapezów

3. **Metoda Simpsona**, dla a=0 i b=1:

$$S(f) = \int_a^b f(x)\,dx = \frac{h}{3} \cdot \sum_{i=1}^{n/2} [f(x_{2*i-2}) + 4f(x_{2*i-1}) + f(x_{2*i})],$$

gdzie:

$$h = \frac{b\ -\ a}{n} = \frac{1}{n}$$

dodatkowo:

$n\ \leftarrow\ n + 1$, bo n musi być liczbą parzystą, tak więc:

$$S(f) = \frac{h}{3} \cdot \sum_{i=1}^{n/2} [f(x_{2i-2}) + 4f(x_{2i-1}) + f(x_{2i})]$$

n	S(f)	Δ
3	0.6932539682539682	0.0001067876940229473
5	0.6931697931697932	0.0000226126098479273

Tabela 1.3: Wartość i błąd obliczonej całki z użyciem metody trapezów

Wnioski:

Można zauważyć, że im n większe tym wynik jest bardziej dokładny w każdym przypadku.
Metoda Simpsona daje najlepsze wyniki.

Zad 2

Obliczyć całkę I = -1∫1 1/(1+x2) dx korzystając z wielomianów ortogonalnych (np. Czebyszewa) dla n=8.

Wartość całki obliczona analitycznie:

$$S(f) = \int \frac{1}{1+x^2} dx = arctan\,x\ + C, \qquad \Rightarrow \qquad [arctan\,x]_{-1}^{1} = \frac{\pi}{2} \approx 1.570796326794$$

Możemy skorzystać z wielomianów Czebyszewa dla obliczenia tej całki. Wielomiany Czebyszewa zdefiniowane są na przedziale [-1, 1] jako $Tn(x) = cos(n \cdot arccos(x))$, tworzą układ ortogonalny z funkcją wagową $w(x) = \frac{1}{\sqrt{1-x^2}}$.

Korzystając z kwadratury Gaussa-Czebyszewa danej wzorem:

$$I(g) = \int_{-1}^{1} g(x) \frac{1}{\sqrt{1-x^2}} dx \approx \sum_{i=1}^{n} w_i \cdot g(t_i)$$

gdzie:

w_i to kolejne wagi, dane wzorem π/n

t_i to miejsca zerowe wielomianu Czebyszewa n-tego stopnia,

dane wzorem cos $\left(\frac{2k+1}{2n}\pi\right)$, gdzie k należy do [0,n-1]

Aby policzyć wartość całki $\int \frac{1}{1+x^2} dx$ tym sposobem potrzeba zmienić całkowaną funkcje aby pasowała do wzorca czyli na $\int \frac{\sqrt{1-x^2}}{1+x^2} dx$. Wtedy całkujemy:

$$\int \frac{\sqrt{1-x^2}}{1+x^2} \cdot \frac{1}{\sqrt{1-x^2}} dx$$

Otrzymujemy wtedy:

Przybliżona wartość całki: 1.5771595097128748

Rzeczywista wartość całki: 1.5707963267948966

Błąd bezwzględny: 0.006363182917978216

Użyty kod python:

```
import math


def czebyszew_weights_and_roots(n):
    weights = [math.pi / n] * n
    roots = [math.cos((2*k + 1) * math.pi / (2 * n)) for k in range(0, n)]
    return weights, roots


def f(x):
    # return 1 / (1 + x**2)
    return math.sqrt(1 - x**2) / (1 + x**2)


def approximate_integral(n):
    weights, roots = czebyszew_weights_and_roots(n)
    integral = sum(w * f(x) for w, x in zip(weights, roots))
    return integral


n = 8
```

```python
approximation = approximate_integral(n)

print("Przybliżona wartość całki:", approximation)

print("Rzeczywista wartość całki:", math.pi / 2)

print("Błąd bezwzględny:", abs(approximation - math.pi / 2))
```

Wnioski:

Do całkowania zadanej funkcji sprawdziła by się bardziej kwadratura Gaussa-Legendre'a, ponieważ musieliśmy rozszerzyć pierwotna funkcje o pierwiastek co może zwiększyć błędy numeryczne

Zadania domowe

Zad 1

Obliczyć całkę 0∫1 1/(1+x2) dx korzystając ze wzoru prostokątów dla h=0.1 oraz metody całkowania adaptacyjnego

1. **Metoda prostokątów** dla h = 0.1

Wzór prostokątów dla całkowania definiowany jest jako:

$$\int_a^b f(x)dx \approx h \sum_{i=0}^{n-1} f(x_i)$$

gdzie h to długość każdego podprzedziału, a *xi* to środek i-tego podprzedziału.

W tym przypadku, *a*=0, *b*=1, *h*=0.1, więc mamy 10 równoodległych podprzedziałów.

Otrzymujemy wartość: 0.7856064962502747

2. **Metoda całkowania adaptacyjnego**

Metoda całkowania adaptacyjnego to technika numerycznego obliczania całek, która dostosowuje ilość węzłów całkowania w zależności od kształtu funkcji. Główną ideą jest używanie więcej węzłów tam, gdzie funkcja ma bardziej zmienny charakter, a mniej węzłów w miejscach, gdzie funkcja jest bardziej regularna.

Jednym z popularnych algorytmów jest metoda Simpsona, która dzieli przedział całkowania na podprzedziały i oblicza przybliżoną wartość całki na każdym z tych podprzedziałów za pomocą wzoru Simpsona. Następnie kontroluje błąd i decyduje, czy podprzedział należy dalej dzielić na mniejsze segmenty.

Otrzymujemy wartość: 0.7853981786625575

Wykorzystany kod python:

```python
def adaptive_integration(f, a, b, tol):
  def simpson_rule(f, a, b):
    return (b - a) / 6 * (f(a) + 4 * f((a + b) / 2) + f(b))
```

```
def adaptive_integration_helper(f, a, b, tol, whole_interval):
    c = (a + b) / 2
    left_subinterval = simpson_rule(f, a, c)
    right_subinterval = simpson_rule(f, c, b)
    total_subinterval = left_subinterval + right_subinterval

    if abs(total_subinterval - whole_interval) <= 15 * tol:
        return total_subinterval + (total_subinterval - whole_interval) / 15

    return (adaptive_integration_helper(f, a, c, tol / 2, left_subinterval) +
            adaptive_integration_helper(f, c, b, tol / 2, right_subinterval))

whole_interval = simpson_rule(f, a, b)
return adaptive_integration_helper(f, a, b, tol, whole_interval)

# Funkcja, którą chcemy całkować
def function(x):
    return 1 / (1 + x**2)

# Granice całkowania
a = 0
b = 1

# Tolerancja
tolerance = 1e-6

# Obliczenie całki
integral = adaptive_integration(function, a, b, tolerance)
print("Przybliżona wartość całki:", integral)
```

Wnioski:

Całkowanie adaptacyjne jest bardziej dokładne i często zużywające mniej zasobów.

Zad 2

Metodą Gaussa obliczyć następującą całkę 0∫1 1/(x+3) dx dla n=4. Oszacować resztę kwadratury.

Wartość całki obliczona analitycznie:

$$S(f) = \int \frac{1}{x+3} dx = ln(x+3) + C$$

$$[ln(x+3)]_0^1 = \ln(4) - \ln(3) \approx 0.28768207245178$$

Musimy dokonać przekształcenia przedziału (0, 1) na przedział (−1, 1) i stąd otrzymujemy wzór na wartość aproksymowaną postaci:

$$\int_0^1 f(t)dt = \frac{1-0}{2}\int_{-1}^{1} f\left(\frac{1-0}{2}x + \frac{1+0}{2}\right) dx = 1/2 \int_{-1}^{1} \frac{1}{\frac{1}{2}x + \frac{7}{2}} dx$$

Użyty zostanie sposób kwadratury Gaussa-Legendre'a:

$$I(f) = \int_{-1}^{1} f(x)dx \approx \sum_{i=1}^{n} w_i f(t_i),$$

gdzie:

t_i to miejsca zerowe i-tego wielomianu Legendre'a.

i	w_i	t_i
1	0.6521451548625461	-0.3399810435848563
2	0.6521451548625461	0.3399810435848563
3	0.3478548451374538	-0.8611363115940526
4	0.3478548451374538	0.8611363115940526

Tabela 2: Wagi w_i *oraz miejsca zerowe* t_i *dla N = 4*

Otrzymujemy:

S(f) = 0.2876820721506314,

Δ = 0.000000000301149327697

Wnioski :

Już dla N=4 otrzymujemy bardzo dobre przybliżenie

Bibliografia:

wykład dr inż. Katarzyna Rycerz
https://www.wolframalpha.com/calculators/integral-calculator/
https://home.agh.edu.pl/~funika/mownit/lab5/calkowanie.pdf
https://home.agh.edu.pl/~funika/mownit/lab5/what_is_gauss.html
https://pl.wikipedia.org/wiki/Kwadratury_Gaussa
https://pl.wikipedia.org/wiki/Wielomiany_Czebyszewa
https://pl.wikipedia.org/wiki/W%C4%99z%C5%82y_Czebyszewa