Міністерство освіти і науки України  
НТУУ «КПІ ім. Ігоря Сікорського»  
Кафедра диференціальних рівнянь

Лабораторна робота №1  
Тема: “Обчислення функціональних рядів”  
Варіант № 6

Виконав студент 2-го курсу  
ТЕФ, групи ТР-71  
Зуєв Михайло Олександрович  
Дата: 18.09.2018  
  
Перевірив: Сігайов А.О.

Київ – 2018

6**.** , де Ві – числа Бернуллі, fточ(*х*)=ctg(*x*).

Код програми :

'''

Лабораторна работа номер 1

з курсу Чисельні методи, варіант 2

Завдання: Для ряду f(x)= обчислити його наближене значення при х[0,1] з кроком h=0.05.

Обчислення припинити, якщо |ai|<eps=0.001 для двох послідовних членів ряду. Задане точне значення ряду fточ(х).

Вивести кількість ітерацій N, при яких була досягнута точність eps.

Опис програмы: програма обраховує точно значення ряду и точної функціі. Точність до епсилон.

Виводить таблицю на єкран де написані усі значення и кількість ітерацій.

Процедури: factorial, drawCoord.

Виконав студент 2 курсу: Зуєв Михайло Олександрович

'''

import math

from fractions import Fraction

import matplotlib.pyplot as plt

import numpy

def sum(x, getList = False):

''' Обчислює наближене значення функціонального ряду.

Для ряду f(x) = 1/x - E(1, oO) (2^(2i) \* B[i] \* x^(2i - 1)) / (2i)!, де B[i] - числа Бернуллі

обчислити його наближене значення. Обчислення припинити, якщо |a[i]| < eps = 0.001 для двох

послідовних членів ряду (Або коли кількість ітерацій стає більшою за 11, тому що використувуються

лише 11 перших чисел Бернуллі).

Args:

x: аргумент, для якого потрібно порахувати ряд.

getList: якщо цей параметр True, то також повертається список зі значеннями a\_i

Returns:

Функція завжди повертає два значення: обчисленний функціональний ряд і кількість ітерацій.

Якщо аргумент getList == True, то також повертається список зі значеннями елементів ряду a\_i

на кожній ітерації.

'''

# Константи

eps = 0.001

x2 = x\*x

B = [Fraction(1,6), Fraction(1,30), Fraction(1,42), Fraction(1,30), Fraction(5,66), Fraction(691,2730),

Fraction(7,6), Fraction(3617,510), Fraction(43867,798), Fraction(174611,330), Fraction(854513,138)] # Перші 11 чисел Бернуллі

flag = True # Використовується, щоб для виходу з циклу умова виконалась два рази підряд

A = [] # список зі значеннями ai

# Обчислюємо перший елемент ряду і додаємо його до суми

i = 1

a = x / 3

if getList:

A.append(a)

sum = a

while math.fabs(a) >= eps or flag:

# Таким чином вихід з циклу виконається тільки якщо два послідовних члена ряду

# будуть меньші за eps

if(math.fabs(a) >= eps):

flag = True

else:

flag = False

i += 1

# Для обчислення кожного елементу ряду використувується рекурентна формула

a \*= (2\*x2\*B[i-1]) / ((2\*i-1)\*i)

if getList:

A.append(a)

sum += a

if getList:

return 1/x - sum, i, A

return 1/x - sum, i

def ctg(x):

''' Обчислює значення cos(x).

# WARNING: Для значення х, при якому sin(x) == 0 виникне помилка.

'''

return math.cos(x) / math.sin(x)

# Списки з аргумментами та значеннями функцій для графіку

xlist = []

fxList = []

ftList = []

# Вивід шапки таблиці

print(" x | Sum | ctg | f-f | N |")

# Рахуємо значення функціонального ряду та точної функції на проміжку x є (0, 1]

# з кроком h = 0.5. Виводимо ці значення в таблицю та малюємо графіки.

for x in range(5, 101, 5):

x = x/100

xlist.append(x)

fx, n = sum(x) # Значення функціонального ряду та кількість ітерацій

ft = ctg(x) # Точне значення

fxList.append(fx)

ftList.append(ft)

print("%3.2f | %6.3f | %6.3f | %6.1f | %3d |" % (x, fx, ft, (fx - ft) / fx \* 100, n))

# Графіки y = f(x) и y = f\_точне(x)

plt.subplot(1, 2, 1)

plt.plot(xlist, fxList, xlist, ftList)

plt.xlabel(r'$x$')

plt.ylabel(r'$y$')

plt.title(r'$f\_1(x)=\ f(x),\ f\_2(x)=\ ctg(x)$')

# Графік y = a\_i, для x = 0.5

fx, n, A = sum(0.5, True)

plt.subplot(1, 2, 2)

ilist = numpy.arange(1, n + 1, 1)

plt.plot(ilist, A)

plt.xlabel(r'$i$')

plt.ylabel(r'$y$')

plt.title(r'$y=\ a(i),$')

plt.show()





