Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Методы численного анализа»

ОТЧЕТ

к лабораторной работе №8

на тему:

«ЧИСЛЕННОЕ ДИФФЕРЕНЦИРОВАНИЕ И ИНТЕГРИРОВАНИЕ ФУНКЦИЙ»

БГУИР 1-40 04 01

Выполнил студент группы 253505
БЕКАРЕВ Станислав Сергеевич

(дата, подпись студента)

Проверил доцент кафедры информатики
АНИСИМОВ Владимир Яковлевич

(дата, подпись преподавателя)

Минск 2023

Содержание

Цель работы

- изучить методы численного вычисления производных и методы численного интегрирования;
- сравнить методы по трудоемкости, точности;
- составить программу методов численного вычисления;
- выполнить тестовые примеры и проверить правильность работы программы

Задание. В каждом варианте найти численное значение первой и второй производной в точке, являющейся серединой заданного интервала, с точностью до 0,01. Вычислить с заданной точностью интегралы по формулам прямоугольников, трапеций, Симпсона. Сравнить методы по точности.

№ вари-анта	**Функция** $f(x)$	Интервал	**Метод**	**Точность**	**Значение интеграла**
3.	$ch\,x$	[0 ; 2]	Средних	0,000001	**0,4995949**

Вариант 3

Исходные данные

```
Изначальная функция f(x): cosh(x)
```

Программная реализация

Код формул производной:

```
def diff_method_1(func, point, accuracy):
    return (func.subs(x, point + accuracy) - func.subs(x, point)) /
accuracy

def diff_method_2(func, point, accuracy):
    return (func.subs(x, point + accuracy) - func.subs(x, point -
accuracy)) / (2 * accuracy)

def diff2_method(func, point, accuracy):
    return (func.subs(x, point + accuracy) - 2 * func.subs(x, point) +
func.subs(x, point - accuracy)) / (accuracy ** 2)
```

Код средних прямоугольников:

```
def integral_middle_rectangle(func, left, right, accuracy):
    i = left
    Integ = 0.
    while(i < right - 0.5 * accuracy):
        Integ += accuracy * func.subs(x, i + 0.5 * accuracy)
        i += accuracy
    return Integ
```

Код формулы Симпсона:

```
def simpthon_method(func, left, right, accuracy):
    X, Y = [], []
    check = False
    i = left
    while i < right + 0.5 * accuracy:
        X.append(i)
        i += accuracy
    if(len(X) % 2 == 0):
        X.append(right + accuracy)
        check = True
    for i in range(0, len(X)):
        Y.append(func.subs(x, X[i]))
    Integ = 0.
    j = 0
    while j  < len(Y) - 2:
        Integ += Y[j] + 4 * Y[j + 1] + Y[j + 2]
        j += 2
    if(check):
        Integ -= 0.5 * (Y[len(Y)-3] + 4 * Y[len(Y)-2] + Y[len(Y)-1])
    return Integ * accuracy / 3
```

Полученные результаты

```
Изначальная функция f(x): cosh(x)
Производная в точке 1.0:
1.17520119364380
1-ая формула производной:
Y'[k] = (Y[k]-Y[k-1])/(X[k]-X[k-1]
1.18293624789763
Погрешность:
-0.00773505425382437
2-ая формула производной:
Y'[k] = (Y[k+1]-Y[k-1])/(X[k+1]-X[k-1]
1.17522078042830
Погрешность:
-1.95867844974273e-5
Вторая производная в точке 1.0:
1.54308063481524
Формула второй производной:
Y''[k] = (Y'[k+1]-Y'[k-1])/(X[k+1]-X[k-1])
1.54309349386539
Погрешность:
-1.28590501446979e-5
Интеграл ф-ии cosh(x) на интервале(0.0, 2.0):
3.62686040784702
Метод средних прямоугольников (0.001)
3.62686025672769
Погрешность:
1.51119326829985e-7
Метод Симпсона (0.01)
3.62686040804851
Погрешность:
-2.01492600382380e-10
Метод трапеций (0.001)
3.62686071008524
Погрешность:
-3.02238217120276e-7
```

Тестовый пример 1

```
Изначальная функция f(x): sin(x)
Производная в точке 0.5:
0.877582561890373
1-ая формула производной:
Y'[k] = (Y[k]-Y[k-1])/(X[k]-X[k-1]
0.875170827870447
Погрешность:
0.00241173401992545
2-ая формула производной:
Y'[k] = (Y[k+1]-Y[k-1])/(X[k+1]-X[k-1]
0.877567935587473
Погрешность:
1.46263029000560e-5
Вторая производная в точке 0.5:
-0.479425538604203
Формула второй производной:
Y''[k] = (Y'[k+1]-Y'[k-1])/(X[k+1]-X[k-1])
-0.479421543405079
Погрешность:
-3.99519912441804e-6
Интеграл ф-ии sin(x) на интервале(0.0, 1.0):
0.459697694131860
Метод средних прямоугольников (0.001)
0.459697713285932
Погрешность:
-1.91540713845306e-8
Метод Симпсона (0.01)
0.459697694157400
Погрешность:
-2.55393484138722e-11
Метод трапеций (0.001)
0.459697655823719
Погрешность:
3.83081417143494e-8
```

Тестовый пример 2

```
Изначальная функция f(x): log(x)
Производная в точке 2.0:
0.500000000000000
1-ая формула производной:
Y'[к] = (Y[к]-Y[к-1])/(X[к]-X[к-1])
0.498754151103897
Погрешность:
0.00124584889610269
2-ая формула производной:
Y'[к] = (Y[к+1]-Y[к-1])/(X[к+1]-X[к-1])
0.500004166729162
Погрешность:
-4.16672916214722e-6
Вторая производная в точке 2.0:
-0.250000000000000
Формула второй производной:
Y''[к] = (Y'[к+1]-Y'[к-1])/(X[к+1]-X[к-1])
-0.250003125052967
Погрешность:
3.12505296662380e-6
Интеграл ф-ии log(x) на интервале(1.0, 3.0):
1.29583686600433
Метод средних прямоугольников (0.001)
1.29583689378201
Погрешность:
-2.77776759372017e-8
Метод Симпсона (0.01)
1.29583686589735
Погрешность:
1.06983533143534e-10
Метод трапеций (0.001)
1.29583681044868
Погрешность:
5.55556516346201e-8
```

Вывод

В ходе выполнения лабораторной работы я изучил методы численного вычисления производных и методы численного интегрирования, написал программу их реализации на языке Python, правильность работы программы проверил на тестовых примерах.

На основании тестов можно сделать следующие выводы:

- Метод численного дифференцирования по формуле(2) более точный чем метод численного дифференцирования по формуле(1);
- Численное интегрирование по формуле Симпсона является более точным по сравнению с численным интегрированием методом средних прямоугольников и трапеций.
- Программа позволяет получить производную и интеграл с заданной точностью;