|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ДИСЦИПЛИНА «Вычислительные алгоритмы»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № 6**

|  |  |
| --- | --- |
| **Тема «**Построение и программная реализация алгоритмов численного дифференцирования.»  **Студент** Кузин А.А.  **Группа** ИУ7 – 42 Б  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель** Градов В.М. |  |

Москва.

2020 г.

**Цель работы**: Получение навыков построения алгоритма вычисления производных от сеточных функций.

**Задание:** Задана табличная (сеточная) функция. Имеется информация, что закономерность, представленная этой таблицей, может быть описана формулой

,

параметры функции неизвестны **и определять их не нужно**..

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| x | y | 1 | 2 | 3 | 4 | 5 |
| 1 | 0.571 |  |  |  |  |  |
| 2 | 0.889 |  |  |  |  |  |
| 3 | 1.091 |  |  |  |  |  |
| 4 | 1.231 |  |  |  |  |  |
| 5 | 1.333 |  |  |  |  |  |
| 6 | 1.412 |  |  |  |  |  |

Вычислить первые разностные производные от функции и занести их в столбцы (1)-(4) таблицы:

1 - односторонняя разностная производная,

2 - центральная разностная производная,

3- 2-я формула Рунге с использованием односторонней производной,

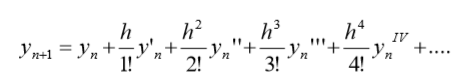
4 - введены выравнивающие переменные.

В столбец 5 занести вторую разностную производную.

**Описание алгоритма:**

**Разложение в ряды Тейлора**

Это наиболее универсальный метод построения формул численного дифференцирования заданных порядков точности относительно шага сетки (таблицы). При этом достаточно просто получается оценка погрешности формул. Таблица задана на множестве значений аргумента, которые при постоянном шаге h образуют сетку  , точки xn называются узлами сетки. Выберем тройку узлов таблицы (xn-1, xn, xn+1). Выполним разложение функции в ряд Тейлора, приняв за центр разложения точку xn





Получим разностные формулы для вычисления первых производных.

Из первого – правую разностную производную 

Из второго – левую разностную производную 

Вычитая (2) из (1), получим центральную формулу для первой производной



Разностный аналог второй производной:

 (6)

**Формулы Рунге**

Погрешность вышеприведенных формул имеет вид  - некоторая функция. Если некоторая приближенная формула Φ для вычисления величины Ω имеет структуру



то записав (6) для сетки с шагом mh, получим



Теперь комбинируя две вышеприведённые формулы, получим вторую формулу Рунге, позволяющую за счет расчета на двух сетках с отличающимися шагами получить решение с более высокой точностью, чем заявленная теоретическая точность используемой формулы



**Выравнивающие переменные**

Идея введения выравнивающих переменных, рассмотренная при изучении аппроксимации функций, очень хорошо работает при проведении операций дифференцирования. Действительно, при удачном выборе этих переменных исходная кривая может быть преобразована в прямую линию, производная от которой вычисляется точно по самым простым формулам. Итак, пусть задана функция *y(x)* и введены выравнивающие переменные .

После вычисления производной в новых переменных возврат к заданным переменным осуществляется следующим образом 

В новых переменных производная будет вычислена точно по

любой односторонней формуле.

**Kод:**

def left\_side\_diff(y, h):

return [None if i == 0

else (y[i] - y[i - 1]) / h for i in range(len(y))]

def right\_side\_diff(y, h):

return [None if i == (len(y) - 1)

else (y[i + 1] - y[i]) / h for i in range(len(y))]

def center\_diff(y, h):

return [None if i == (len(y) - 1) or i == 0

else (y[i + 1] - y[i - 1]) / (h \* 2) for i in range(len(y))]

def runge\_left(y, h):

n = len(y)

p = 1

yh = left\_side\_diff(y, h)

y2h = [0 if i < 2 else (y[i] - y[i - 2]) / (2\*h) for i in range(n)]

return [None if i < 2 else (yh[i] + (yh[i] - y2h[i]) / (2\*\*p - 1)) for i in range(n)]

def eta(y):

return 1 / y

def ksi(x):

return 1 / x

def align(x, y):

n = len(x)

eta\_a = [eta(x[i]) for i in range(n)]

ksi\_a = [ksi(y[i]) for i in range(n)]

return [None if i == len(x) - 1

else ((eta\_a[i + 1] - eta\_a[i]) / (ksi\_a[i + 1] - ksi\_a[i])) \* y[i] / x[i]

for i in range(len(x))]

def second\_diff(y, h):

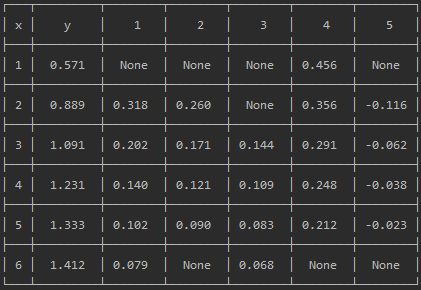
return [None if i == (len(y) - 1) or i == 0

else (y[i - 1] - 2 \* y[i] + y[i + 1]) / h\*\*2

for i in range(len(y))]

**Полученные результаты:**

Заполненная таблица с краткими комментариями по поводу использованных формул и их точности



**1** - односторонняя разностная производная(левая, , она, как и правая разностная производная , имеют самый низкий, первый порядок точности, относительно шага.

**2** - центральная разностная производная, , которая обладает более высоким, вторым порядком точности.

**3** - 2-я формула Рунге с использованием односторонней производной,

, при использовании односторонней производной (левой), она имеет второй порядок точности, так как p=1, для односторонней разностной производной.

**4** - введены выравнивающие переменные. введем выравнивающие переменные η =1/y, ξ=1/x, после чего производную вычисляю по правосторонней разностной производной.

В столбец 5 занести вторую разностную производную. Рассчитываю ее по формуле 

**Вопросы при защите лабораторной работы:**

1. Получить формулу порядка точности  для первой разностной производной  в крайнем правом узле .
2. Используя 2-ую формулу Рунге, дать вывод выражения (9) из Лекции №7 для первой производной  в левом крайнем узле

.

Для левого узла имеем n=0, n+1=1, n+2=2