|  |  |
| --- | --- |
|  | **Министерство образования и науки 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 (ИУ)

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

# Лабораторная работа №6

**Тема:** Численное дифференцирование

**Студент:** Артемьев И.О.

**Группа:** ИУ7-43Б

**Оценка (баллы): \_\_\_\_\_\_\_\_\_**

**Преподаватель:** Градов В.М.

Москва.

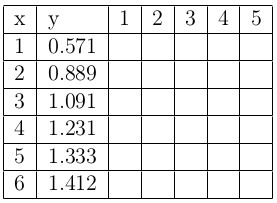
2021 г*.*

## **Задание**

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



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



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

**Входные данные**

Таблица из задания.

**Выходные данные**

Заполненная таблица.

## **Описание алгоритма**

Используя разложение в ряд Тейлора можно получить левую



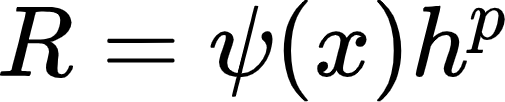
и правую разностную формулы



Данные формулы имеют самый низкий - первый - порядок точности. Из данных формул можно получить центральную разностную формулу



Центральная разностная формула имеет второй порядок точности.

Приведенные выше формулы имеют погрешность вида . С помощью

преобразований в рядах Тейлора можно получить первую формулу Рунге



Отсюда можно получить вторую формулу Рунге



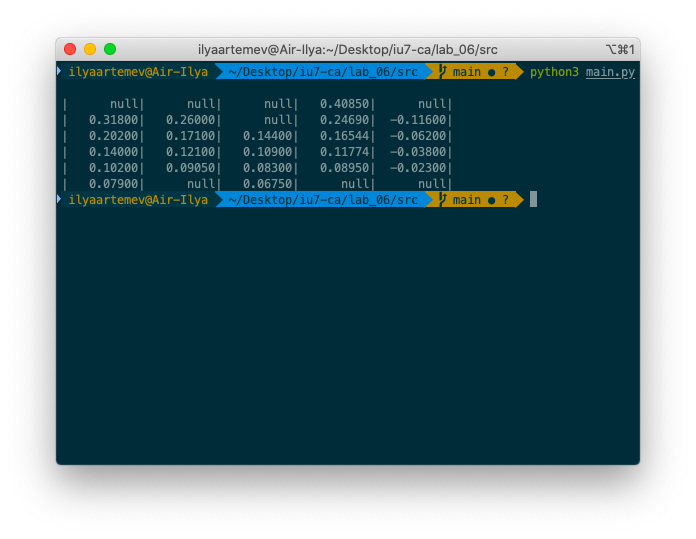
Формулы Рунге справедливы не только для операций дифференцирования, но и для других приближенных вычислений (при условии, что погрешность формул имеет вышеприведенный вид).

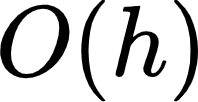
Помимо приведенных выше методов стоит отметить метод, заключающийся в применении выравнивающих переменных. При правильном подборе исходная кривая может быть преобразована в прямую, производная от которой вычисляется точно даже по простым формулам.

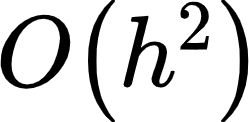
Пусть задана функция 𝑦(𝑥) с введенными переменными 𝜉 = 𝜉(𝑥) и 𝜂 = 𝜂(𝑦). Тогда возврат к заданным переменным будет осуществлен по формуле



## **Результаты работы программы**



Первый столбец - левосторонняя формула (точность ).

Второй столбец - центральная формула (точность ).

Третий столбец - вторая формула Рунге (с использованием левосторонней формулы). Четвертый столбец - применение выравнивающих переменных (оценка точность сложна,

так как неизвестны параметры). Использовано соотношение



Пятый столбец - вторая разностная производная.

## **Код программы**

"""

main.py

"""

from class\_numerical\_differentiator import NumericalDifferentiator

*def* main():

x = [1, 2, 3, 4, 5, 6]

y = [0.571, 0.889, 1.091, 1.231, 1.333, 1.412]

exe = NumericalDifferentiator(x, y)

exe.printCompletedTable()

if \_\_name\_\_ == "\_\_main\_\_":

main()

"""

class\_numerical\_differentiator.py

"""

*class* NumericalDifferentiator:

*def* \_\_init\_\_(*self*, *x*, *y*):

*self*.x = x

*self*.y = y

@staticmethod

*def* \_\_left(*y*, *step*, *ind*):

return (y[ind] - y[ind - 1]) / step if ind > 0 else "%10s" % "null"

@staticmethod

*def* \_\_center(*y*, *step*, *ind*):

return (

(y[ind + 1] - y[ind - 1]) / 2 / step

if ind > 0 and ind < len(y) - 1

else "%10s" % "null"

)

@staticmethod

*def* \_\_secondDiff(*y*, *step*, *ind*):

return (

(y[ind - 1] - 2 \* y[ind] + y[ind + 1]) / step \*\* 2

if ind > 0 and ind < len(y) - 1

else "%10s" % "null"

)

@staticmethod

*def* \_\_alignCoeff(*y*, *x*, *step*, *ind*):

if ind > len(y) - 2:

return "%10s" % "null"

res = (

(1 / y[ind + 1] - 1 / y[ind])

/ (1 / x[ind + 1] - 1 / x[ind])

\* y[ind]

\* y[ind]

/ x[ind]

/ x[ind]

)

return res

*def* \_\_rungeLeft(*self*, *y*, *step*, *ind*):

if ind < 2:

return "%10s" % "null"

f1 = *self*.\_\_left(y, step, ind)

f2 = (y[ind] - y[ind - 2]) / 2 / step

return 2 \* f1 - f2

*def* \_\_getMethodsLinks(*self*):

return [

*self*.\_\_left,

*self*.\_\_center,

*self*.\_\_rungeLeft,

*self*.\_\_alignCoeff,

*self*.\_\_secondDiff,

]

*def* printCompletedTable(*self*):

methods\_links = *self*.\_\_getMethodsLinks()

for i in range(len(*self*.x)):

print("|", end="")

for j in range(len(methods\_links) - 2):

res = methods\_links[j](*self*.y, *self*.x[1] - *self*.x[0], i)

print(

f"{res:10.5f}" if res != "%10s" % "null" else res,

"|",

sep="",

end="",

)

res = *self*.\_\_alignCoeff(*self*.y, *self*.x, *self*.x[1] - *self*.x[0], i)

print(

f"{res:10.5f}" if res != "%10s" % "null" else res, "|", sep="", end=""

)

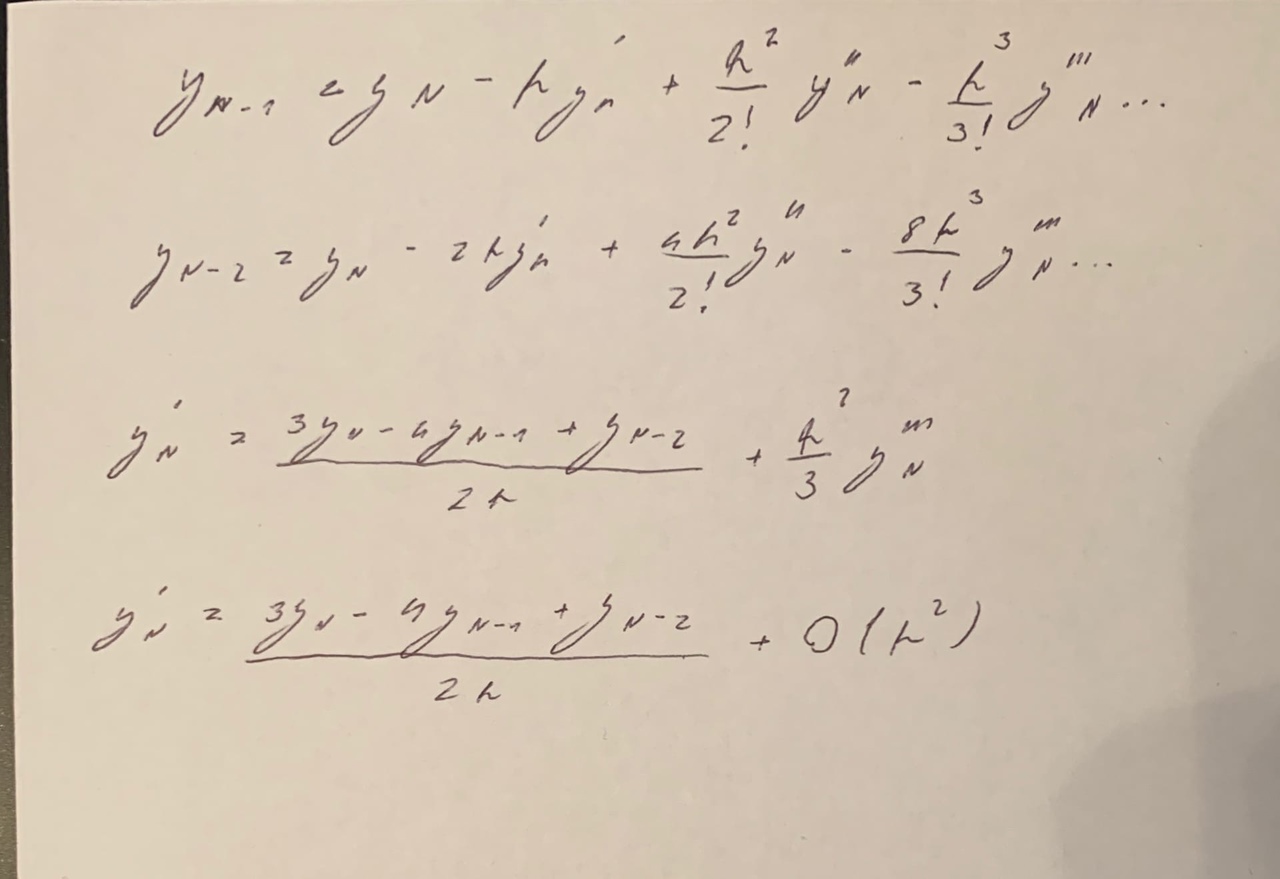
res = *self*.\_\_secondDiff(*self*.y, *self*.x[1] - *self*.x[0], i)

print(f"{res:10.5f}" if res != "%10s" % "null" else res, "|", sep="")

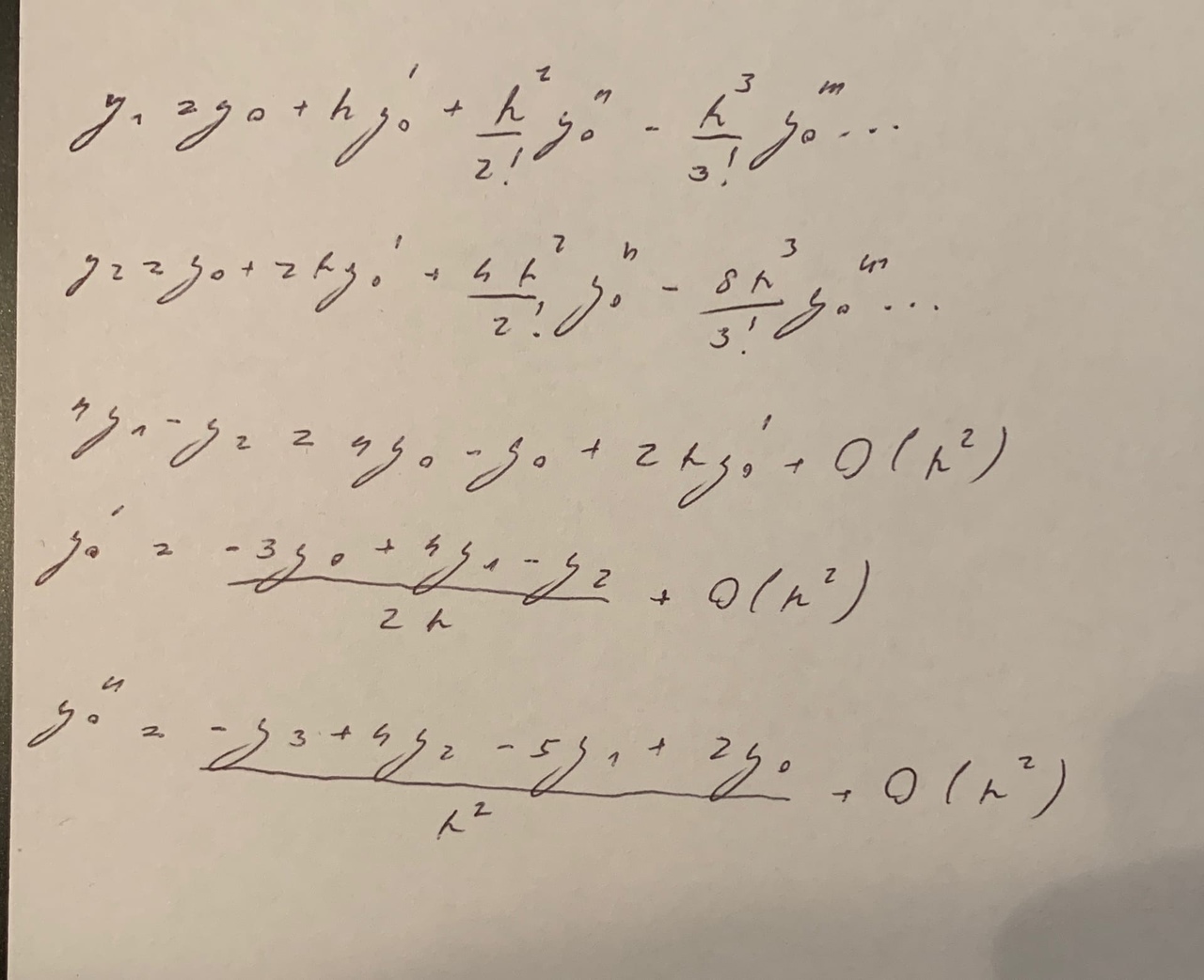
## **Контрольные вопросы**

1. Получить формулу порядка точности O(h2 ) для первой разностной производной y'N в

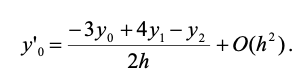
крайнем правом узле xN .

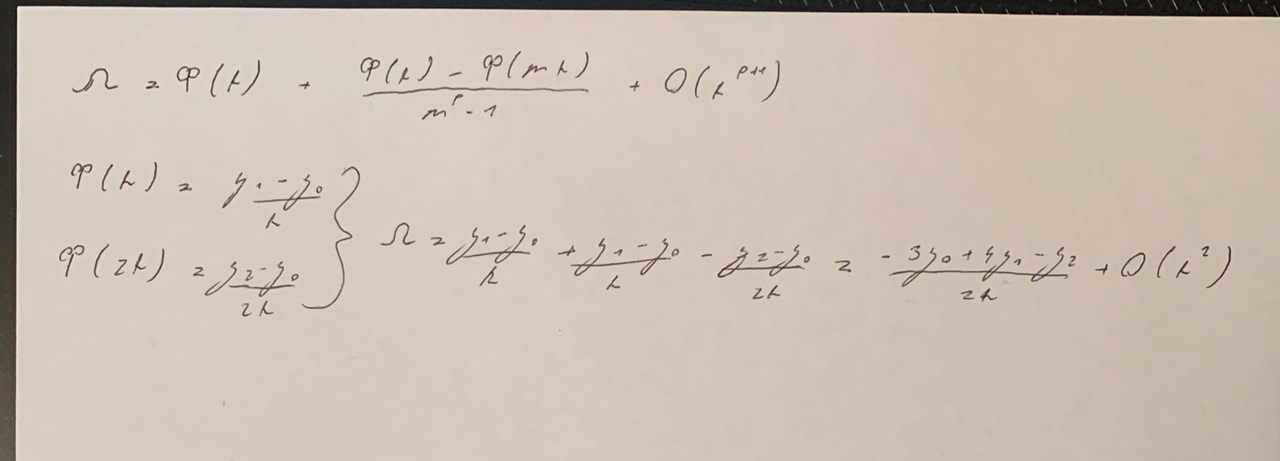


2. Получить формулу порядка точности O(h2 ) для второй разностной производной y''0 в крайнем левом узле x0 .



3. Используя 2-ую формулу Рунге, дать вывод выражения (9) из Лекции No7 для первой производной y'0 в левом крайнем узле





4. Любым способом из Лекций No7, 8 получить формулу порядка точности O(h3) для

первой разностной производной y'0 в крайнем левом узле x0 .

