Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

**Отчет по лабораторной работе №5**

**по дисциплине «Вычислительная математика»**

**Вариант 11**

**Выполнили**:

Студенты группы P3265

Кручинина Дарья Сергеевна

Москвитина Полина

**Преподаватель:**

Машина Екатерина Алексеевна

**Цели работы**

Цель лабораторной работы: решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

**Порядок выполнения работы**

**Вычислительная реализация задачи:**

1. Выбрать из табл. 1 заданную по варианту таблицу 𝑦 = 𝑓(𝑥) (таблица 1.1 –

таблица 1.5);

2. Построить таблицу конечных разностей для заданной таблицы. Таблицу отразить в отчете;

3. Вычислить значения функции для аргумента 𝑋1 (см. табл.1), используя

первую или вторую интерполяционную формулу Ньютона. Обратить внимание какой конкретно формулой необходимо воспользоваться;

4. Вычислить значения функции для аргумента 𝑋2 (см. табл. 1), используя

первую или вторую интерполяционную формулу Гаусса. Обратить внимание

какой конкретно формулой необходимо воспользоваться;

**5. Подробные вычисления привести в отчете.**

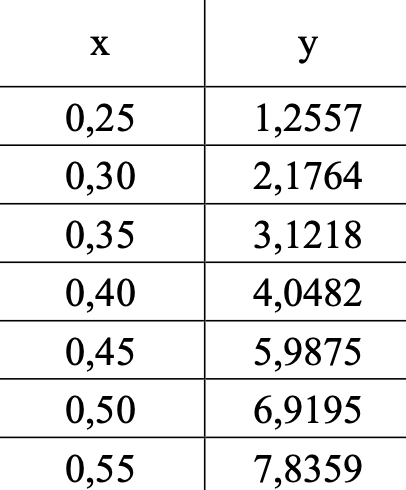


Таблица конечных разностей:

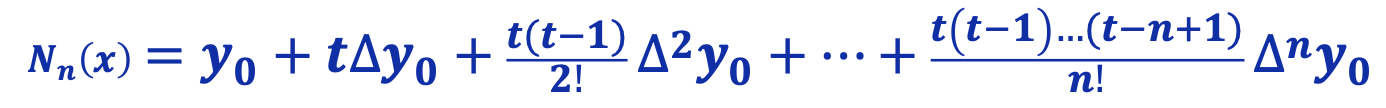
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **x** | **y** |  |  |  |  |  |  |
| 0,25 | 1,2557 | 0,9207 | 0,0247 | -0,0437 | 1,0756 | -4,1277 | 10,1915 |
| 0,30 | 2,1764 | 0,9454 | -0,0190 | 1,0319 | -3,0521 | 6,0638 |  |
| 0,35 | 3,1218 | 0,9264 | 1,0129 | -2,0202 | 3,0117 |  |  |
| 0,40 | 4,0482 | 1,9393 | -1,0073 | 0,9917 |  |  |  |
| 0,45 | 5,9875 | 0,9320 | -0,0156 |  |  |  |  |
| 0,50 | 6,9195 | 0,9164 |  |  |  |  |  |
| 0,55 | 7,8359 |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **Номер варианта** | **X1** | **X2** |
| 11 | 0,255 | 0,405 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| i = 0,…5 | i = 0,…4 | i = 0,…3 | i = 0,…2 | i = 0, 1 | i = 0 |

**Значения функции для аргумента X1:**

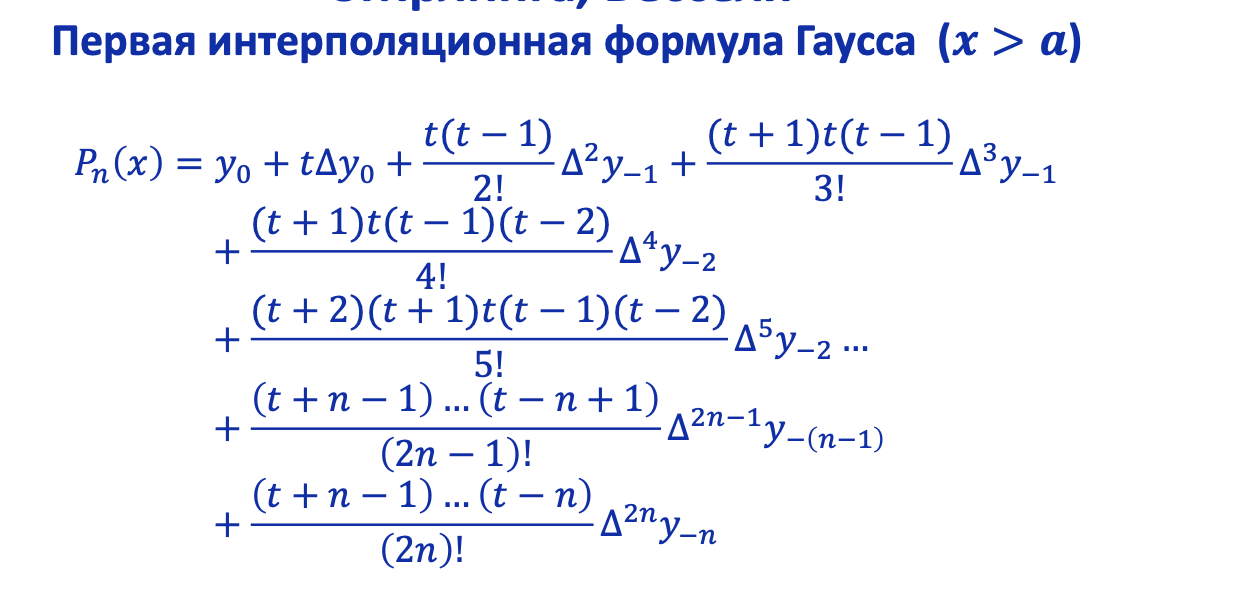
Будем использовать первую формулу Ньютона, так как точна находится в левой части отрезка.



**Значения функции для аргумента X2:**

a = 0,40

Будем использовать первую формулу Гаусса



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **x** | **y** |  |  |  |  |  |  |
| 0,25 | 1,2557 | 0,9207 | 0,0247 | -0,0437 | 1,0756 | -4,1277 | 10,1915 |
| 0,30 | 2,1764 | 0,9454 | -0,0190 | 1,0319 | -3,0521 | 6,0638 |  |
| 0,35 | 3,1218 | 0,9264 | 1,0129 | -2,0202 | 3,0117 |  |  |
| 0,40 | 4,0482 | 1,9393 | -1,0073 | 0,9917 |  |  |  |
| 0,45 | 5,9875 | 0,9320 | -0,0156 |  |  |  |  |
| 0,50 | 6,9195 | 0,9164 |  |  |  |  |  |
| 0,55 | 7,8359 |  |  |  |  |  |  |

**import** Interpolation

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

**from** math **import** sqrt, sin

FILE\_IN1 = "/Users/pm/Desktop/input1"

FILE\_IN2 = "/Users/pm/Desktop/input2b"

FILE\_IN3 = "/Users/pm/Desktop/input3"

**def** inputMode():

print("Выберите способ ввода данных:")

print("a) в виде набора данных (x,y)")

print("b) в виде сформированных в файле данных")

print("c) на основе выбранной функции")

choice = input("Способ ввода: ").strip()

**while**(choice != 'a' **and** choice != 'b' **and** choice != 'c'):

choice = input("Введите 'a', 'b' или 'c' для выбора способа ввода: ").strip()

**return** choice

**def** getDataInput():

print('Введите x, y в горизонтальных строках (не менее 5 точек):')

x = list(map(float, input('x: ').split()))

y = list(map(float, input('y: ').split()))

**return** x, y

**def** getDataFile():

print("Выберите файл: input1, input2, input3")

choice = input("Номер файла: ")

**if**(choice == '1'): FILE\_IN = FILE\_IN1

**elif**(choice == '2'): FILE\_IN = FILE\_IN2

**else**: FILE\_IN = FILE\_IN3

**with** open(FILE\_IN, 'r') **as** file:

x = list(map(float, file.readline().split()))

y = list(map(float, file.readline().split()))

**return** x, y

**def** getFunction():

FUNCTIONS = [

**lambda** x: 2 \* x\*\*2 - 8 \* x + 1,

**lambda** x: (x\*\*0.5) / 2,

**lambda** x: sin(x)

]

print("Выберите функцию")

print("1) 2x² - 8x + 1")

print("2) sqrt(x)/2")

print("3) sin(x)")

func\_id = int(input('Номер функции: '))

a, b = map(float, input('Границы отрезка через пробел: ').split())

a, b = min(a, b), max(a, b)

n = int(input('Количество узлов интерполяции: '))

h = (b - a) / (n - 1)

x = [a + i \* h **for** i **in** range(n)]

y = [FUNCTIONS[func\_id - 1](x\_val) **for** x\_val **in** x]

**return** x, y

**def** plot(x, y, plot\_x, plot\_y1, plot\_y2, plot\_y3, x0, answer1, answe2, answer3):

plt.title("Лабораторная работа 5")

OFFSET = 0.5

plt.grid(**True**)

plt.xlim((min(0, min(x)) - OFFSET, max(x) + OFFSET))

plt.ylim((min(0, min(y)) - OFFSET, max(y) + OFFSET))

ax = plt.gca()

ax.spines['left'].set\_position('zero')

ax.spines['bottom'].set\_position('zero')

ax.spines['right'].set\_color('none')

ax.spines['top'].set\_color('none')

ax.plot(1, 0, marker='>', ms=5, color='k', transform=ax.get\_yaxis\_transform(), clip\_on=**False**)

ax.plot(0, 1, marker='^', ms=5, color='k', transform=ax.get\_xaxis\_transform(), clip\_on=**False**)

plt.plot(x,y, color = 'blue')

ax.plot(x, y, label = "Заданные значения")

plt.plot(plot\_x, plot\_y1)

ax.plot(plot\_x, plot\_y1, label = "Многочлен Лагранжа")

plt.plot(plot\_x, plot\_y2)

ax.plot(plot\_x, plot\_y2, label = "Многочлен Ньютона с разделенными разностями")

plt.plot(plot\_x, plot\_y3)

ax.plot(plot\_x, plot\_y3, label = "Многочлен Ньютона с конечными разностями")

plt.plot([x0], [answer1], 'ro')

plt.plot([x0], [answer2], 'ro')

plt.plot([x0], [answer3], 'ro')

plt.legend()

plt.show()

**if** \_\_name\_\_ == '\_\_main\_\_':

print("ЛАБОРАТОРНАЯ РАБОТА")

print("--------------------------------------------")

choice = inputMode()

**if** choice == 'a': x, y = getDataInput()

**elif** choice == 'b': x, y = getDataFile()

**elif** choice == 'c': x, y = getFunction()

Interpolation.FiniteDifferenceTable(x, y)

value = float(input("Заданное значение аргумента: "))

answer1 = Interpolation.Lagrange(x, y, value)

answer2 = Interpolation.NewtonFiniteDifferences(x, y, value)

answer3 = Interpolation.NewtonSeparatedDifferences(x, y, value)

print("Приближенное значения функции по многочлену Лагранжа: " + str(answer1))

print("Приближенное значения функции по многочлену Ньютона с разделенными разностями: " + str(answer2))

print("Приближенное значения функции по многочлену Ньютона с конечными разностями: " + str(answer3))

print("Приближенное значения функции по схеме Стирлинга: " + str(Interpolation.Stirling(x, y, value)))

print("Приближенное значения функции по схеме Бесселя: " + str(Interpolation.Bessel(x, y, value)))

plot\_x = np.linspace(np.min(np.array(x)), np.max(np.array(x)), 100)

plot\_y1 = [Interpolation.Lagrange(x, y, value) **for** value **in** plot\_x]

plot\_y2 = [Interpolation.NewtonFiniteDifferences(x, y, value) **for** value **in** plot\_x]

plot\_y3 = [Interpolation.NewtonSeparatedDifferences(x, y, value) **for** value **in** plot\_x]

plot(x, y, plot\_x, plot\_y1, plot\_y2, plot\_y3, value, answer1, answer2, answer3)

**import** math

**import** numpy **as** np

**from** prettytable **import** PrettyTable

**def** FiniteDifferenceTable(x, y):

table = [[0 **for** \_ **in** range(len(y) + 1)] **for** \_ **in** range(len(y))]

**for** i **in** range(len(y)):

table[i][0] = f"{float(x[i]):.2f}"

table[i][1] = f"{float(y[i]):.4f}"

**for** j **in** range(2, len(y) + 1):

**for** i **in** range(len(y) - j + 1):

table[i][j] = float(table[i+1][j-1]) - float(table[i][j-1])

table[i][j] = f"{table[i][j]:.5f}"

**if**(len(y) < 3):\_FiniteDifferenceTable = PrettyTable(["x", "y", \*[f"∆{chr(184+j)}y" **for** j **in** range(1, len(y))]])

**elif**(len(y) < 5):\_FiniteDifferenceTable = PrettyTable(["x", "y", \*[f"∆{chr(184+j)}y" **for** j **in** range(1, 2)], \*[f"∆{chr(176+j)}y" **for** j **in** range(2, len(y))]])

**else**:\_FiniteDifferenceTable = PrettyTable(["x", "y", \*[f"∆{chr(184+j)}y" **for** j **in** range(1, 2)], \*[f"∆{chr(176+j)}y" **for** j **in** range(2, 4)], \*[f"∆{chr(8304+j)}y" **for** j **in** range(4, len(y))]])

**for** row **in** table:

\_FiniteDifferenceTable.add\_row(row)

print(\_FiniteDifferenceTable)

**def** Lagrange(x, y, value):

result = 0

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

c1 = c2 = 1

**for** j **in** range(len(x)):

**if** i != j:

c1 \*= value - x[j]

c2 \*= x[i] - x[j]

result += y[i] \* c1 / c2

**return** round(result,5)

**def** NewtonSeparatedDifferences(x, y, value):

f = subNewtonSeparatedDifferences\_createrTable(x, y)

result = y[0]

**for** j **in** range(1, len(f[0])):

temp = f[0][j]

**for** i **in** range(0, j): temp \*= (value - x[i])

result += temp

**return** round(result,5)

**def** subNewtonSeparatedDifferences\_createrTable(x, y):

f = [[0 **for** \_ **in** range(len(y) )] **for** \_ **in** range(len(y))]

**for** i **in** range(len(y)):

f[i][0] = y[i]

**for** j **in** range(1, len(y)):

**for** i **in** range(len(y) - j):

f[i][j] = (f[i+1][j-1] - f[i][j-1])/(x[i + j] - x[i])

**return** f

**def** subNewton\_createrTable(y):

table = [[0 **for** \_ **in** range(len(y))] **for** \_ **in** range(len(y))]

**for** i **in** range(len(y)):table[i][0] = y[i]

**for** j **in** range(1, len(y)):

**for** i **in** range(len(y) - j):

table[i][j] = table[i+1][j-1] - table[i][j-1]

**return** table

**def** NewtonFiniteDifferences(x, y, value):

table = subNewton\_createrTable(y)

**if** value <= x[len(x) - 1]:

x0 = 0

**for** i **in** range(len(x) - 1, -1, -1):

**if** value >= x[i]:

x0 = i

**break**

t = (value - x[x0]) / (x[1] - x[0])

result = table[x0][0]

**for** i **in** range(1, len(table[x0])):

temp = t

**for** yi **in** range(1, i): temp \*= (t - yi)

result += (temp \* table[x0][i]) / math.factorial(i)

**else**:

t = (value - x[len(x) - 1]) / (x[1] - x[0])

result = table[len(x) - 1][0]

**for** i **in** range(1, len(x)):

temp = t

**for** yi **in** range(1, i): temp \*= (temp + yi)

result += (temp \* table[len(x) - i - 1][i]) / math.factorial(i)

**return** round(result,5)

**def** createrTable\_Guass(y):

result = [y]

**for** i **in** range(len(y) - 1):

div\_dif = []

**for** j **in** range (len(result[i]) - 1):

diff = result[i][j+1] - result[i][j]

div\_dif.append(diff)

result.append(div\_dif)

**return** result

**def** Stirling(x, y, value):

**if**(len(y) % 2 == 0):

print("Четное число узло. Формула Стирлинга не применяется")

**return**

table = createrTable\_Guass(y)

mid = len(y)//2

h = x[1] - x[0]

t = (value - x[mid])/h

**if**(abs(t) > 0.25): print("Результат по формуле Стирлинга содержит большую погрешность")

result = y[mid]

**for** i **in** range(1, mid + 1):

mul = 1

**for** j **in** range(1, i):

mul \*= (t \* t - j \* j)

result += t \* mul \* (table[2\*i-1][-(i-1) + mid] + table[2 \* i - 1][-i + mid]) / (2 \* math.factorial(2\*i-1))

result += t \* t \* mul \* (table[2 \* i][-i + mid]) / math.factorial(2\*i)

**return** result

**def** Bessel(x, y, value):

**if**(len(y) % 2 != 0):

print("Нечетное число узло. Формула Бесселя не применяется")

**return**

table = createrTable\_Guass(y)

mid = len(y)//2

h = x[1] - x[0]

t = (value - x[mid])/h

**if**(abs(t) < 0.25 **or** abs(t) > 0.75): print("Результат по формуле Бесселя содержит большую погрешность")

result = (y[mid] + y[mid+1])/2 + (t - 0.5)\*table[1][mid]

**for** i **in** range(2, mid):

mul = 1

**for** j **in** range(0, i):

mul \*= (t + math.pow(-1, j)\*j)

n = i - 1

result += mul \* (table[2\*n][-n + mid] + table[2\*i - 2][-(n-1) + mid]) / (2 \* math.factorial(2\*n))

result += (t - 0.5) \* mul \* (table[2\*n + 1][-n + mid]) / math.factorial(2\*n + 1)

**return** result