**Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»**

**Факультет программной инженерии и компьютерной техники**

**Вычислительная математика**

**Лабораторная работа №5**

**Вариант 2**

Группа: P3267

Выполнила:

Бореева В. Ю.  
Тихонов И. В.

Проверил:

Машина Е. А.

Г. Санкт-Петербург

2024

# Цель работы

Разобраться с понятием интерполяции, научиться решать задачи по этой теме и реализовать это в программе

# 1 Вычислительная часть задачи:

1. Выбранная таблица

Изображение выглядит как текст, число, снимок экрана, Шрифт

Автоматически созданное описание

1. Таблица конечных разностей

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, число, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, число, снимок экрана

Автоматически созданное описание

1. Вычислить значения функции для аргумента 𝑋1, используя

первую или вторую интерполяционную формулу Ньютона.

Изображение выглядит как текст, Шрифт, число, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, число, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, число, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, линия, чек, График

Автоматически созданное описание

Изображение выглядит как текст, линия, График, чек

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, число, снимок экрана

Автоматически созданное описание

1. Вычислить значения функции для аргумента 𝑋2, используя

первую или вторую интерполяционную формулу Гаусса.

Изображение выглядит как текст, Шрифт, число, снимок экрана

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, чек, линия, График

Автоматически созданное описание

Изображение выглядит как текст, чек, линия, График

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

# 2 Программная часть задачи:

**from** Equations **import** calculate\_y

**from** Lagrange **import** lagrange

**from** Newton **import** newton

**from** Gauss **import** gauss

**from** Stirling **import** stirling

**from** Bessel **import** bessel

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

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

datamode = int(input("Выберите способ ввода данных:\n1. С клавиатуры.\n2. Из файла.\n3. На основе функции.\n> "))

**def** diff\_table(arr\_y, k):

res = [arr\_y]

c = 0

**while** **True**:

tmp = []

**for** q **in** range(k):

tmp.append(res[c][q + 1] - res[c][q])

**if** sum(tmp) == 0:

**return** res

**else**:

c += 1

k -= 1

res.append(tmp)

**if** datamode == 1:

n = int(input("Введите n: "))

xarr = list(map(float, input("Введите строку X: ").split()))

yarr = list(map(float, input("Введите строку Y: ").split()))

**if** len(xarr) != n + 1 **or** len(yarr) != n + 1:

**raise** Exception("Неверная длина массива.")

**elif** datamode == 2:

table\_number = int(input(

"Выберите таблицу:\n1. X: 0.1, 0.2, 0.3, 0.4, 0.5; Y: 1.25, 2.38, 3.79, 5.44, 7.14\n2. X: 1, 2, 3, 4; Y: 0, "

"3, 5, 7\n3. X: 100, 121, 144; Y: 10, 11, 12\n4. X: 0.15, 0.2, 0.33, 0.47; Y: 1.25, 2.38, 3.79, 5.44\n> "))

**if** table\_number == 1:

xarr, yarr = [0.1, 0.2, 0.3, 0.4, 0.5], [1.25, 2.38, 3.79, 5.44, 7.14]

n = 4

**elif** table\_number == 2:

xarr, yarr = [1, 2, 3, 4], [0, 3, 5, 7]

n = 3

**elif** table\_number == 3:

xarr, yarr = [100, 121, 144], [10, 11, 12]

n = 2

**elif** table\_number == 4:

xarr, yarr = [0.15, 0.2, 0.33, 0.47], [1.25, 2.38, 3.79, 5.44]

n = 3

**else**:

**raise** Exception("Таблица не найдена.")

**elif** datamode == 3:

function\_number = int(input("Выберите функцию:\n1. sin(x)\n2. cos(x)\n> "))

a, b = map(int, input("Введите интервал [A, B]: ").split())

n = int(input("Введите количество точек на интервале: "))

step = (b - a) / n # шаг

n -= 1

xarr, yarr = [], []

**for** i **in** np.arange(a, b, step):

xarr.append(i)

yarr.append(calculate\_y(i, function\_number))

**else**:

**raise** Exception("Неизвестный режим.")

diff\_table = diff\_table(yarr, n)

print("Таблица разностей: ", end="")

**for** i **in** range(len(diff\_table)):

print(f"\ny{i}: ", end="")

**for** j **in** diff\_table[i]:

print(round(j, 4), end=" ")

print()

arg = float(input("Введите значение аргумента: "))

print(

f"Многочлен Лагранжа: {lagrange(arg, xarr, yarr, n)}\nМногочлен Ньютона с разделенными разностями:"

f" {newton(arg, xarr, yarr, n)}\nМногочлен Гаусса: {gauss(arg, xarr, n, diff\_table)}")

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

print(f"Многочлен Бесселя: {bessel(arg, xarr, n, diff\_table)}")

**else**:

print(f"Многочлен Стирлинга: {stirling(arg, xarr, n, diff\_table)}")

plt.plot(xarr, yarr)

**if** datamode == 3:

x = np.linspace(-np.pi, np.pi, 200) # Диапазон значений x

**if** function\_number == 1:

y = np.sin(x)

**elif** function\_number == 2:

y = np.cos(x)

**else**:

**raise** Exception("Неизвестная функция")

plt.plot(x, y)

plt.scatter(xarr, yarr, marker='o', s=50, color='red')

plt.show()

**import** math

**def** bessel(x, xarr, n, diff\_table):

a\_index = len(xarr) // 2-1 # y0

h = (xarr[-1] - xarr[0]) / n

t = (x - xarr[a\_index]) / h

res = 0

**for** i **in** range(len(diff\_table)):

**if** i == 0:

res += (diff\_table[i][a\_index] + diff\_table[i][a\_index + 1]) / 2

**elif** i == 1:

res += diff\_table[i][a\_index] \* (t - 0.5)

**else**:

**if** i % 2 == 0:

tempn = i // 2

tmpres = (diff\_table[i][a\_index - tempn] + diff\_table[i][a\_index]) / (2 \* math.factorial(i))

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

tmpres \*= (t - j) \* (t + j - 1)

res += tmpres

**else**:

tempn = (i - 1) // 2

tmpres = (t - 0.5) \* diff\_table[i][a\_index - tempn]/math.factorial(i)

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

tmpres \*= (t - j) \* (t + j - 1)

res += tmpres

**return** res

**def** newton(x, xarr, yarr, n):

res = 0

**for** i **in** range(n + 1):

**if** i == 0:

res += yarr[i]

**else**:

narr = []

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

narr.append(j)

tmp = \_\_recurrent(xarr, yarr, i + 1, narr)

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

tmp \*= (x - xarr[j])

res += tmp

**return** res

**def** \_\_recurrent(xarr, yarr, k, narr):

**if** k == 2:

**return** (yarr[narr[-1]] - yarr[narr[0]]) / (xarr[narr[-1]] - xarr[narr[0]])

**else**:

**return** (\_\_recurrent(xarr, yarr, k - 1, \_\_remfirst(narr)) - \_\_recurrent(xarr, yarr, k - 1, \_\_remlast(narr))) / (

xarr[narr[-1]] - xarr[narr[0]])

**def** \_\_remfirst(arr):

new\_array = arr.copy()

new\_array.pop(0)

**return** new\_array

**def** \_\_remlast(arr):

new\_array = arr.copy()

new\_array.pop()

**return** new\_array

**import** math

**def** calculate\_y(x, equ):

**if** equ == 1:

**return** math.sin(x)

**elif** equ == 2:

**return** math.cos(x)

**def** lagrange(x, xarr, yarr, n):

res = 0

**for** i **in** range(n + 1):

numerator = 1

denominator = 1

**for** j **in** range(n + 1):

**if** j != i:

denominator \*= (xarr[i] - xarr[j])

numerator \*= (x - xarr[j])

res += (numerator / denominator) \* yarr[i]

**return** res

ВЫВОД

В ходе выполнения лабораторной работы мы разобрались с понятием интерполяции, научились решать задачи по этой теме и реализовали это в программе