***Санкт-Петербургский национальный исследовательский университет ИТМО***

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Вычислительная математика»

**Отчет**

По лабораторной работе №5

Вариант 9

Выполнила:

Миличенко А.Е.

P3210

Преподаватель:

Малышева Т.А.

Санкт-Петербург

2024 г.

**Цель работы**

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

**Листинг программы**

Файл main.py:

from methods import \*

import numpy as np

#func = lambda x: x\*\*2

func = lambda x: np.sin(x)

if ask\_input\_data() == 1:

x, y = get\_points\_file()

temp\_func = None

else:

print("Введите границы интервала для набора точек")

a = ask\_num()

b = ask\_num()

if a > b:

a, b = b, a

st = 0

print("Введите шаг: ")

while st <= 0:

st = ask\_num()

x, y = get\_points\_from\_func(func, a, b, st)

temp\_func = func

x\_plot = np.linspace(np.min(x) - plot\_area, np.max(x) + plot\_area, 1000)

plot\_lag\_newton(x\_plot, temp\_func,

[lagrange\_method(x, y, x\_now) for x\_now in x\_plot],

[newton\_method(x, y, x\_now) for x\_now in x\_plot],

[stirling\_method(x, y, x\_now) for x\_now in x\_plot],

[bessel\_method(x, y, x\_now) for x\_now in x\_plot], x, y)

while True:

print("Введите X: ")

n = ask\_num()

print("Методом Лагранжа: ", lagrange\_method(x, y, n))

print("Методом Ньютона: ", newton\_method(x, y, n))

print("Методом Гаусса: ", gauss\_polynomial(x, y, n))

print("Методом Стирлинга: ", stirling\_method(x, y, n))

print("Методом Бесселя: ", bessel\_method(x, y, n))

Файл methods.py:

import matplotlib.pyplot as plt

from functools import reduce

from math import factorial

def newton\_method(x\_arr, y\_arr, x):

diff = [y\_arr]

for i in range(len(x\_arr)):

tmp\_dif = []

for j in range(len(x\_arr) - i - 1):

tmp\_dif.append((diff[-1][j + 1] - diff[-1][j])/(x\_arr[j + i + 1] - x\_arr[j]))

diff.append(tmp\_dif)

mul = 1

answer = y\_arr[0]

for i in range(len(x\_arr)-1):

mul \*= (x - x\_arr[i])

answer += diff[i + 1][0] \* mul

return answer

#конечные разности

def final\_differences(y\_values):

final\_diff = [y\_values]

for i in range(len(y\_values)-1):

temp\_fin\_dif = []

for j in range(len(final\_diff[i]) - 1):

diff = final\_diff[i][j + 1] - final\_diff[i][j]

temp\_fin\_dif.append(diff)

final\_diff.append(temp\_fin\_dif)

return final\_diff

def lagrange\_method(x\_arr, y\_arr, x):

l = 0

for j in range(len(y\_arr)):

numerator = 1

denominator = 1

for i in range(len(x\_arr)):

if i != j:

numerator = numerator \* (x - x\_arr[i])

denominator = denominator \* (x\_arr[j] - x\_arr[i])

l = l + y\_arr[j] \* numerator / denominator

return l

def get\_points\_from\_func(f, a, b, step):

x = []

y = []

x\_now = a

while x\_now <= b:

x.append(x\_now)

y.append(f(x\_now))

x\_now += step

return x, y

def create\_factorial(n):

result = []

for i in range(n+1):

fact = 1

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

fact \*= j

result.append(fact)

return result

def stirling\_method(x\_arr, y\_arr, x):

if len(y\_arr) % 2 == 0:

return

h = x\_arr[1] - x\_arr[0]

#проверка, что все остальные разности одинаковые

for i in range(len(y\_arr) - 1):

if round(x\_arr[i + 1] - x\_arr[i], 8) != round(h, 8):

return

fin\_diff = final\_differences(y\_arr)

fact = create\_factorial(len(y\_arr))

mid = len(y\_arr)//2

t = (x - x\_arr[mid]) / h

result = y\_arr[mid]

for n in range(1, mid + 1):

mul = 1

for j in range(1, n):

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

result += 1 / fact[2 \* n - 1] \* t \* mul \* (fin\_diff[2 \* n - 1][-(n - 1) + mid] + fin\_diff[2 \* n - 1][-n + mid]) / 2

result += 1 / fact[2 \* n] \* (t \*\* 2) \* mul \* (fin\_diff[2 \* n][-n + mid])

return result

def bessel\_method(x\_arr, y\_arr, x):

if len(y\_arr) % 2 == 1:

return

h = x\_arr[1] - x\_arr[0]

for i in range(len(y\_arr)-1):

if round(x\_arr[i+1] - x\_arr[i], 8) != round(h, 8):

return

fin\_diff = final\_differences(y\_arr)

fact = create\_factorial(len(y\_arr))

mid = (len(y\_arr) - 2) // 2

t = (x - x\_arr[mid]) / h

result = 0

for n in range(0, mid + 1):

mul = 1

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

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

result += (1 / fact[2 \* n]) \* mul \* (fin\_diff[2 \* n][-n + mid] + fin\_diff[2 \* n][-(n - 1) + mid]) / 2

result += (1 / fact[2 \* n + 1]) \* (t - (1 / 2)) \* mul \* (fin\_diff[2 \* n + 1][-n + mid])

return result

def gauss\_polynomial(xs, ys, x):

n = len(xs) - 1

alpha\_ind = n // 2

fin\_difs = [ys[:]]

# Вычисление конечных разностей

for k in range(1, n + 1):

last = fin\_difs[-1]

fin\_difs.append([last[i + 1] - last[i] for i in range(n - k + 1)])

h = xs[1] - xs[0]

dts1 = [0, -1, 1, -2, 2, -3, 3, -4, 4]

# Функция для вычисления суммы

def calc\_sum(x, dts, alpha\_ind, h, fin\_difs):

return sum([

reduce(lambda a, b: a \* b,

[(x - xs[alpha\_ind]) / h + dts[j] for j in range(k)])

\* fin\_difs[k][len(fin\_difs[k]) // 2] / factorial(k)

for k in range(1, n + 1)

])

# Выбор корректной формулы в зависимости от значения x

if x > xs[alpha\_ind]:

return ys[alpha\_ind] + calc\_sum(x, dts1, alpha\_ind, h, fin\_difs)

else:

return ys[alpha\_ind] + calc\_sum(x, [-dt for dt in dts1], alpha\_ind, h, fin\_difs)

plot\_area = 2

def plot\_lag\_newton(x, f\_y, lag\_y, newton\_y, stirling\_y, bessel\_y, point\_x, point\_y):

if f\_y != None:

plt.plot(x, f\_y(x), linewidth=2.0, label="function")

plt.plot(x, lag\_y, linewidth=2.0, label="lagrange")

plt.plot(x, newton\_y, linewidth=2.0, label="newton")

plt.plot(x, stirling\_y, linewidth=2.0, label="stirling")

plt.plot(x, bessel\_y, linewidth=2.0, label="bessel")

plt.plot(point\_x, point\_y, '\*', linewidth=0, label="points")

plt.legend()

plt.grid(True)

minimum\_x = min(point\_x)

minimum\_y = min(point\_y)

maximum\_x = max(point\_x)

maximum\_y = max(point\_y)

plt.xlim(minimum\_x - plot\_area, maximum\_x + plot\_area)

plt.ylim(minimum\_y - plot\_area, maximum\_y + plot\_area)

plt.show()

INPUT = "./input.txt"

def get\_points\_file():

# Получение точек из файла

with open(INPUT, 'rt') as file:

try:

x = []

y = []

for line in file:

new\_row = list(map(float, line.strip().split()))

if len(new\_row) != 2:

raise ValueError

x.append(new\_row[0])

y.append(new\_row[1])

except ValueError:

print("Неверный формат файла")

exit()

return x, y

def ask\_input\_data():

mode = 0

while mode != 1 and mode != 2:

try:

mode = int(input("Ведите источник точек. Для файла - 1, из функции - 2: "))

except Exception:

print("Введите число")

return mode

def ask\_num():

a = 0

while True:

try:

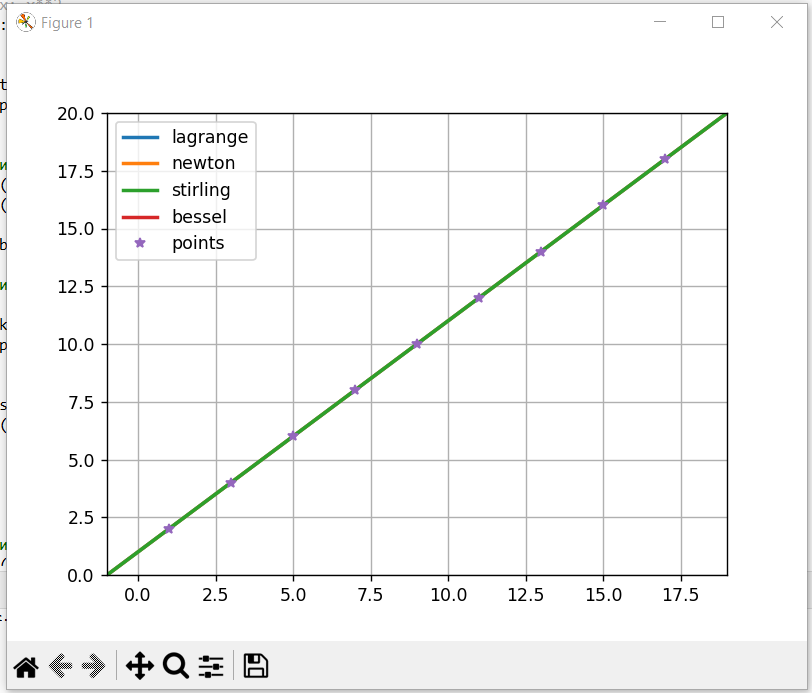
a = float(input("Введите число: "))

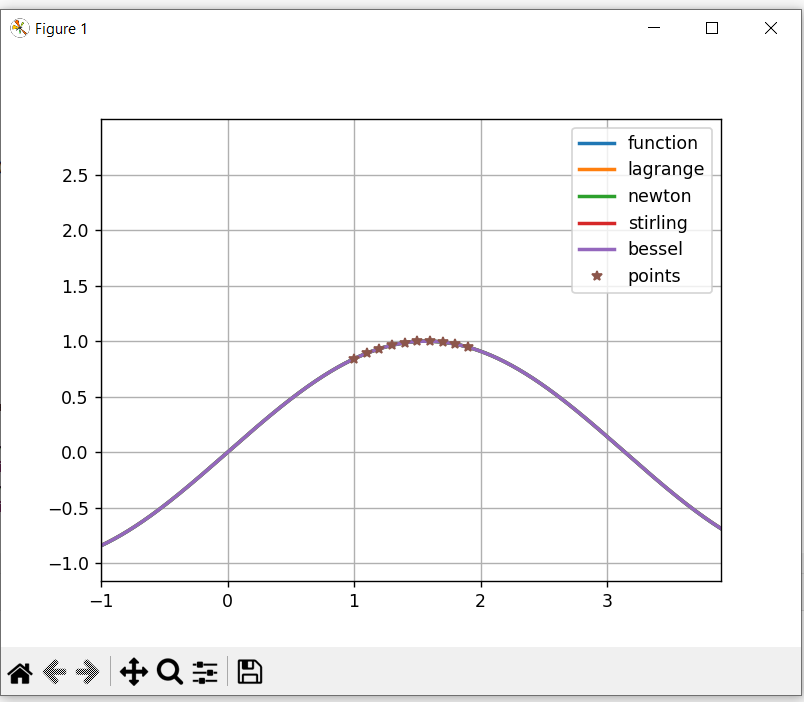
return a

except Exception:

print("Нужно вводить число!")

**Результат работы программы**





Ведите источник точек. Для файла - 1, из функции - 2: 2

Введите границы интервала для набора точек

Введите число: 1

Введите число: 2

Введите шаг:

Введите число: 0.1

Введите X:

Введите число: 7

Методом Лагранжа: 7.014577865600586

Методом Ньютона: 7.015037564626239

Методом Гаусса: 7.084605487909197

Методом Стирлинга: None

Методом Бесселя: 7.015038652774936

Введите X:

Введите число: 8

Методом Лагранжа: 32.212249755859375

Методом Ньютона: 32.21361547183464

Методом Гаусса: 32.56591284424968

Методом Стирлинга: None

Методом Бесселя: 32.21362029418154

Введите X:

Введите число:

**Вывод**

Целью данной лабораторной работы было решить задачу интерполяции и найти значения функции при заданных значениях аргумента, отличных от узловых точек. В ходе работы мы рассмотрели различные методы интерполяции, включая интерполяцию Лагранжа, интерполяцию Ньютона и сплайновую интерполяцию.