Министерство образования и науки РФ Санкт-Петербургский политехнический университет Петра Великого Институт компьютерных наук и технологий Высшая школа программной инженерии

**ЛАБОРАТОРНАЯ РАБОТА №1**

по дисциплине «Вычислительная математика»

Выполнил

студент гр. 3530904/80004 Петров А.И.

Руководитель,

к.т.н, доцент Леонтьева Т.В.

Санкт-Петербург

2020

Формулировка задания

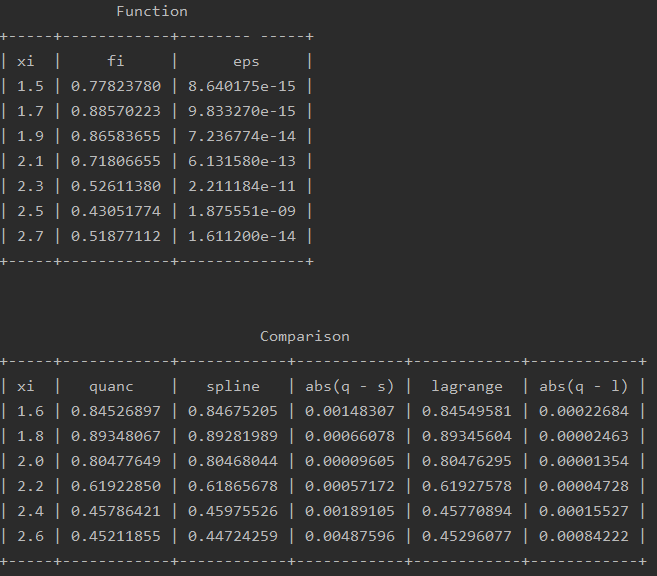
Для с шагом h=0.2 вычислить значения функции f(x) с использованием программы QUANC8, где . По полученным точкам построить сплайн-функцию и полином Лагранжа 6-й степени. В точках для k=0,1,…,5 сравнить значения сплайн-функции и полинома с точным значением f(x) (вычислить интеграл по QUANC8 с высокой точностью)

Выполнение работы

Используя библиотеку scipy была написана функция, являющаяся представлением анализируемой функции f(x). Затем было вычислено её значение в узлах таблицы и построена сама таблица. По полученным значениям построены сплайн и полином Лагранжа (также с помощью пакета scipy) и вычислены их значения в разных точках. Проведен анализ погрешности.

Результат работы

## Вывод программы:



Вывод

Сплайн-интерполяция и полиномы Лагранжа достаточно точно приближают исходную функцию. При сравнении погрешностей оказалось, что погрешность полиномов Лагранжа меньше, чем при использовании сплайна. Это объясняется тем, что полином Лагранжа, построенный по данным точкам, получился не слишком высокой степени, а значит его вид меньше зависит от исходных данных и приближает лучше. Если бы точек было больше, лучше было бы выбрать сплайн.

Исходный код программы:

import numpy

from scipy.integrate import quad

from scipy.interpolate import splrep, splev, lagrange

from numpy import sin, abs

# For comparing floats

DELTA = 0.000001

def f(x):

return quad(

lambda t: sin(t\*t),

0, x)

start = 1.5

end = 2.7

h = 0.2

# From 1.5 to 2.7 step 0.2

x = numpy.arange(start, end + DELTA, h)

# Map x to f(x)

y\_with\_eps = numpy.array([f(xi) for xi in x])

print(" Function ")

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

print("| xi | fi | eps |")

for i in range(x.size):

(fi, eps\_i) = y\_with\_eps[i]

print("| {0:2.1f} | {1:2.8f} | {2:e} |".format(x[i], fi, eps\_i))

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

print()

print()

new\_start = 1.6

new\_end = 2.6

# From 1.6 to 2.6 step 0.2

x\_new = numpy.arange(new\_start, new\_end + DELTA, h)

# Map x\_new to f(x\_new)

y\_quanc = numpy.array([f(xi) for xi in x\_new])

# Map x\_new to spline(x\_new)

y = numpy.array([yi for (yi, \_) in y\_with\_eps])

spline\_coefficients = splrep(x, y, s=0) # spline

y\_spline = splev(x\_new, spline\_coefficients, der=0) # seval

y\_lagrange = lagrange(x, y)

print(" Comparison ")

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

print("| xi | quanc | spline | abs(q - s) | lagrange | abs(q - l) |")

for i in range(x\_new.size):

(quanc, \_) = y\_quanc[i]

spline = y\_spline[i]

lagrange = y\_lagrange(x\_new[i])

dqs = abs(quanc - spline)

dql = abs(quanc - lagrange)

print("| {0:2.1f} | {1:2.8f} | {2:2.8f} | {3:2.8f} | {4:2.8f} | {5:2.8f} |"

.format(x\_new[i], quanc, spline, dqs, lagrange, dql))

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