Федеральное государственное автономное образовательное учреждение высшего образования

***Университет ИТМО***

Факультет программной инженерии и компьютерной техники

Дисциплина: **Вычислительная математика**

Лабораторная работа №5

*“* **«Интерполяция функции*”***

# Вариант: 9

Выполнил: Кузнецов Максим Александрович

Группа: Р3211

Преподаватель: Малышева Татьяна Алексеевна

Санкт-Петербург 2022 г

**Цель работы:**

Решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

Для исследования использовать:

* многочлен Лагранжа;
* многочлен Ньютона;
* многочлен Гаусса.

**Условия и задание:**

1. Вычислительная реализация задачи:
   1. Используя первую или вторую интерполяционную формулу Ньютона, первую или вторую интерполяционную формулу Гаусса вычислить значения функции при данных значениях аргумента (для значения Х1 и Х2, см. табл. 1 - 4).
   2. Построить таблицу конечных разностей.
   3. **Подробные вычисления привести в отчете.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | y |  | № варианта | X1 | X2 |
| 0,25 | 1,2557 |  | 1 | 0,251 | 0,402 |
| 0,30 | 2,1764 |  | 5 | 0,512 | 0,372 |
| 0,35 | 3,1218 |  | 9 | 0,255 | 0,405 |
| 0,40 | 4,0482 |  | 13 | 0,534 | 0,384 |
| 0,45 | 5,9875 |  | 17 | 0,272 | 0,445 |
| 0,50 | 6,9195 |  | 21 | 0,551 | 0,351 |
| 0,55 | 7,8359 |  | 25 | 0,294 | 0,437 |

1. Программная реализация задачи:

3.1. Исходные данные задаются в виде: а) набора данных (таблицы x,y), б) на основе выбранной функции (например, ).

3.2. Вычислить приближенное значение функции для заданного значения аргумента, введенного с клавиатуры, указанными методами (см. табл.5).

3.3. Построить графики заданной функции с отмеченными узлами интерполяции и интерполяционного многочлена Ньютона/Гаусса (разными цветами).

|  |  |
| --- | --- |
| 9 | 1, 2 |

**Вычислительная часть:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| i/k | k0 | k1 | k2 | k3 | k4 | k5 | k6 |
| i0 | 1,2557 | 0,9207 | 0,0247 | -0,0437 | 1,0756 | -4,1277 | 10,1917 |
| i1 | 2,1764 | 0,9454 | -0,019 | 1,0319 | -3,0521 | 6,064 |  |
| i2 | 3,1218 | 0,9264 | 1,0129 | -2,0202 | 3,0119 |  |  |
| i3 | 4,0482 | 1,9393 | -1,0073 | 0,9917 |  |  |  |
| i4 | 5,9875 | 0,932 | -0,0156 |  |  |  |  |
| i5 | 6,9195 | 0,9164 |  |  |  |  |  |
| i6 | 7,8359 |  |  |  |  |  |  |

Для X = 0.255:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| i/k | k0 | k1 | k2 | k3 | k4 | k5 | k6 |
| i0 | 1,2557 | 0,9207 | 0,0247 | -0,0437 | 1,0756 | -4,1277 | 10,1917 |
| i1 | 2,1764 | 0,9454 | -0,019 | 1,0319 | -3,0521 | 6,064 |  |
| i2 | 3,1218 | 0,9264 | 1,0129 | -2,0202 | 3,0119 |  |  |
| i3 | 4,0482 | 1,9393 | -1,0073 | 0,9917 |  |  |  |
| i4 | 5,9875 | 0,932 | -0,0156 |  |  |  |  |
| i5 | 6,9195 | 0,9164 |  |  |  |  |  |
| i6 | 7,8359 |  |  |  |  |  |  |

t = (x-xn) / h = (0.255 – 0.3) / 0.05 = -0.9

Для X = 0.405:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| i/k | k0 | k1 | k2 | k3 | k4 | k5 | k6 |
| i0 | 1,2557 | 0,9207 | 0,0247 | -0,0437 | 1,0756 | -4,1277 | 10,1917 |
| i1 | 2,1764 | 0,9454 | -0,019 | 1,0319 | -3,0521 | 6,064 |  |
| i2 | 3,1218 | 0,9264 | 1,0129 | -2,0202 | 3,0119 |  |  |
| i3 | 4,0482 | 1,9393 | -1,0073 | 0,9917 |  |  |  |
| i4 | 5,9875 | 0,932 | -0,0156 |  |  |  |  |
| i5 | 6,9195 | 0,9164 |  |  |  |  |  |
| i6 | 7,8359 |  |  |  |  |  |  |

t = (x-xn) / h = (0.405 – 0.45) / 0.05 = -0.9

**Листинг программы:**

import numpy as np

from numpy import genfromtxt

import matplotlib

import matplotlib.pyplot as plt

import matplotlib.lines as mlines

from matplotlib.pyplot import figure

import sys

import math

import pandas as pd

from math import factorial as fc

def in\_float(s = 'Введите число', integer = False, check = [False, 0, 0]):

flag = True

while flag:

flag = False

try:

if integer:

val = int(input(s +': '))

else:

val = float(input(s +': '))

if check[0] and (val < check[1] or val > check[2]):

raise ValueError

except ValueError:

flag = True

if check[0]:

print(f'Попробуйте снова! Введенное число должно принадлежать интервалу [{check[1]}; {check[2]}]\n')

else:

print(f'Попробуйте снова!\n')

return val

def parse():

flag = True

while flag:

path = input('Путь:\n').strip()

try:

a = genfromtxt(path, delimiter=',')

if True in np.isnan(a) or a.shape[0] != 2:

raise ValueError

return a

except ValueError:

print('В файле должно быть 2 строчки, в каждой одинаковое количество чисел\n')

except OSError:

print('Такого файла нет.\n')

print('Попробуйте снова!\n')

def input\_vals():

n = in\_float(s = 'Введите количество точек', integer = True)

print()

a = []

for i in range(int(n)):

a.append([in\_float('x'), in\_float('y')])

print()

return np.array(a).transpose()

def to\_df(array, f, eps, f\_type = 'f'):

np\_arr = np.concatenate((array, [f], [eps]), axis=0)

return pd.DataFrame(data=np\_arr, index=["X", "Y", f\_type, 'eps'])

def newline(p1, p2, color = 'black'):

ax = plt.gca()

xmin, xmax = ax.get\_xbound()

if(p2[0] == p1[0]):

xmin = xmax = p1[0]

ymin, ymax = ax.get\_ybound()

else:

ymax = p1[1]+(p2[1]-p1[1])/(p2[0]-p1[0])\*(xmax-p1[0])

ymin = p1[1]+(p2[1]-p1[1])/(p2[0]-p1[0])\*(xmin-p1[0])

l = mlines.Line2D([xmin,xmax], [ymin,ymax], color = color)

ax.add\_line(l)

return l

def check\_and\_draw(x, y, approximate\_function, point):

fig, ax = plt.subplots()

xnew=np.linspace(np.min(x),np.max(x),100)

ynew=[approximate\_function(x,y,i) for i in xnew]

plt.plot(x,y,'o', label = 'Входные точки')

plt.plot(xnew,ynew, label = 'Функция аппр.')

plt.plot(point[0], point[1], '.', markersize=12, label = 'Аппроксимация')

plt.title(approximate\_function)

ax.legend()

plt.grid(True)

plt.show()

def interpolate\_lagrange(x, y, x\_cur) :

res = 0.0

for i in range (0, len(x)):

p = 1.0

for j in range (0, len(x)):

if(i != j):

p \*= (x\_cur - x[j])/(x[i]-x[j])

res += p\*y[i]

return res

def coef(y, n, i):

if n == 0:

return (y[i + 1] - y[i])

return (coef(y, n - 1, i + 1) - coef(y, n - 1, i))

def newton\_forward\_interpolation(x, y, x\_cur):

i = 0

n = len(x) - 1

t = (x\_cur - x[i]) / (x[1] - x[0])

return y[i] + sum(np.prod([t - j for j in range(k)]) / math.factorial(k) \* coef(y, k - 1, i) for k in range(1, n - i + 1))

def newton\_backward\_interpolation(x, y, x\_cur):

n = len(x) - 1

t = (x\_cur - x[n]) / (x[1] - x[0])

return y[n] + sum(np.prod([t + j for j in range(k)]) / math.factorial(k) \* coef(y, k - 1, n - k) for k in range(1, n + 1))

def interpolate\_newton(x, y, x\_cur):

if x\_cur > x[int((x[-1]+x[0])/2)]:

return newton\_backward\_interpolation(x, y, x\_cur)

else:

return newton\_forward\_interpolation(x, y, x\_cur)

def run():

again = True

while again:

again = False

in\_type = input('Введите:\n\t\* k - если вводить с клавиатуры\n\t\* f - если вводить из файла\n')

if in\_type.strip() == 'k':

data = input\_vals()

elif in\_type.strip() == 'f':

data = parse()

else:

print('Введено неверно, попробуйте снова.')

again = True

print(f'Итоговый сет данных:\n{data[0]}\n{data[1]}')

cur\_x = in\_float('Число, для которого интерполировать значение: ', check = [True, min(data[0]), max(data[0])])

lagrange\_result = interpolate\_lagrange(data[0],data[1],cur\_x)

print(f'Lagrange\nОтвет методом Лагранжа: {lagrange\_result}')

check\_and\_draw(data[0], data[1], interpolate\_lagrange, [cur\_x, lagrange\_result])

print()

newtone\_result = interpolate\_newton(data[0],data[1],cur\_x)

print(f'Newtone\nОтвет методом Ньютона: {newtone\_result}')

check\_and\_draw(data[0], data[1], interpolate\_newton, [cur\_x, newtone\_result])

run()

**Пример работы:**

Введите:

\* k - если вводить с клавиатуры

\* f - если вводить из файла

f

Путь:

data.txt

Итоговый сет данных:

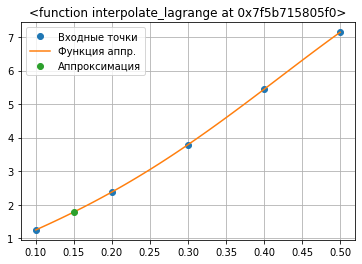
[0.1 0.2 0.3 0.4 0.5]

[1.25 2.38 3.79 5.44 7.14]

Число, для которого интерполировать значение: : 0.15

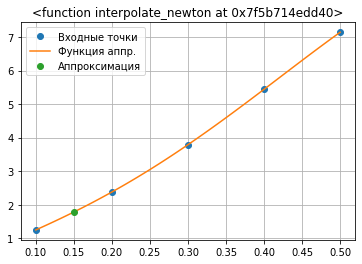
Lagrange

Ответ методом Лагранжа: 1.783359375

****

Newtone

Ответ методом Ньютона: 1.7833593750000007

****

**Вывод:**

В результате выполнения лабораторной работы я:

* Познакомился с различными способами решения интерполирования функции, нахождении с помощью численных методов аппр. точек.
* Попрактиковался как в решении задачи на бумаге, так и написании программной реализации.
* Поработал с Python, в особенности с библиотекой отрисовки графиков matplotlib и numpy.