Министерство образования и науки Российской Федерации

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра прикладной математики

Лабораторная работа №6

по дисциплине «Численные методы»

Группа ПМ-63

Студент Майер В. А.

Преподаватель Задорожный А.Г.

Новосибирск, 2018

1. **Цель работы**

Построить интерполяционный сплайн с базисными функциями Лагранджа третьего порядка.

1. **Анализ**

Задача интерполяции состоит в построении достаточно гладкой функции по набору значений некоторой функции в точках .

Построим интерполяционный кубический сплайн с базисными функциями Лагранджа может быть представлен в виде:

1. **Тесты**

|  |  |
| --- | --- |
| 1. Тест показывающий разницу добавления остатка в предыдущее звено или следующее | |
| Следующее | Предыдущее |
|  |  |
| Нет остатка | |
|  |  |

|  |  |  |
| --- | --- | --- |
| 1. Прямая | | |
|  | |  |
|  |  | |
| 1. Синус     Остаток добавляется к предыддущему блоку. блоки[1-4][4-9] | | |
|  | | |
| Остаток образует новый блок с меньшей степенью. Блоки[1-4][4-7][7-9] | | |
|  | | |
| Без остатка число элементов удовлетворяет 3k + 1 блоки [1-4][4-7][7-10] | | |
|  | | |
| 1. Зависимость результат от количества точек для функции, с явным вырожденим точки ln|X| | | |
|  | | |
|  | | |
| Как мы видем, каждая точка влияет в интеполяционном сплайне и его нельзя использовать с неточными данными | | |
| 1. Проверка на кубической функции y = x^3 | | |
|  | | |

1. **Код программы**

import matplotlib.pyplot as plt

import math

import numpy as np

n = 4

m = 1000

#i базисная функция

def fi(ep, x, i):

mult = 1

for j in range(n):

if(i != j):

mult \*= ep - x[j]

mult /= x[i] - x[j]

return mult

#Значение функции в точке ep

def f(ep, x, y):

sum = 0

for i in range(n):

sum += y[i] \* fi(ep, x, i)

return sum

b = 0 #При разбиении с остатком остаток добавлять в последний или рисовать свой

xrange = []

yrange = []

xp = [1, 1.5, 6, 7, 7.5, 8, 9, 10]

yp = [3, 2, 4, 1, 6, 2, 4, 3]

x = []

y = []

k = int((len(xp) - 1) / (n-1))

#Проверка на разбиение без остатка

p = len(xp) - 1 - (n-1) \* k

lastn = n

for j in range(k):

s = (n-1)\*j

if j == k-1 and b == 1 and p != 0:

n += p

x += [xp[s: s + n]]

y += [yp[s: s + n]]

for i in np.linspace(x[j][0], x[j][-1], m):

xrange += [i]

yrange += [f(i, x[j], y[j])]

n = lastn

#если количество элементов не кратно (n-1)\*k - 1, k ∈ N

if b == 0 and p != 0:

x += [xp[(n-1)\*k: (n-1)\*k + p + 1]]

y += [yp[(n-1)\*k: (n-1)\*k + p + 1]]

for i in range(n - p - 1):

x[k] += [2 \* x[k][-1] - x[k][-2]]

y[k] += [2 \* y[k][-1] - y[k][-2]]

for i in np.linspace(x[k][0], x[k][p], m):

xrange += [i]

yrange += [f(i, x[k], y[k])]

plt.plot(xrange, yrange)

plt.plot(xp, yp, "\*")

#Поиск значений в точках

def Find(X):

for j in range(k):

if x[j][0] <= X <= x[j][-1]:

return f(X, x[j], y[j])

if p != 0

if x[k][0] <= x <= x[k][p]

return f(X, x[k], y[k])

return 'Ошибка'