**Задача 1 по курсу Введение в Численные Методы**

**Насибов Данил, 217**

**30 декабря 2022 г.**

*Постановка задачи:*

Функция f(x) задана таблично на отрезке [0, a] в точках xi, xi = ih, i = 0, 1, ..., N, h = a/N

1. Построить интерполяционный многочлен по точкам xi

2. Интерполировать функцию кубическим сплайном

3. Результаты сравнить. Оценить разность

В качестве входных данных была представлена таблица точек функции:

|  |  |  |
| --- | --- | --- |
| i | x | f(x) |
| 0 | 0 | 0 |
| 1 | 0.1 | 0.529847 |
| 2 | 0.2 | 1.027775 |
| 3 | 0.3 | 1.346477 |
| 4 | 0.4 | 1.356512 |
| 5 | 0.5 | 0.986714 |
| 6 | 0.6 | 0.257137 |
| 7 | 0.7 | 0.706391 |
| 8 | 0.8 | 1.684295 |
| 9 | 0.9 | 2.404336 |
| 10 | 1 | 2.606626 |
| 11 | 1.1 | 2.11956 |
| 12 | 1.2 | 0.927692 |
| 13 | 1.3 | 0.789339 |
| 14 | 1.4 | 2.664212 |
| 15 | 1.5 | 4.203824 |
| 16 | 1.6 | 4.900323 |
| 17 | 1.7 | 4.370876 |
| 18 | 1.8 | 2.493172 |
| 19 | 1.9 | 0.502452 |
| 20 | 2 | 4.019803 |

*Пункт 1: построение полинома Лагранжа*

Представим искомый полином P(x) в виде:

где (x) – полиномы степени n, «ориентированные» на точки , в том смысле, что

Полиномы выражаются следующим образом:

Комментарии к реализации:

Основной функцией является mk\_poly, получающая на вход х (точку, в которой считается значение полинома), rangelist (набор начальных точек), data (значения функции в этих точках). Эта функция вычисляет два массива: 20 множителей числителя и 20 множителей знаменателя для каждого i. Далее применяется формула

*Пункт 2: построение с помощью кубических сплайнов*

Рассмотрим сетку узлов: а = < <… < = b

В нашем случае h = , I = 1, …, n постоянно и равно 0.1

Кубический сплайн функции y = f(x), x [a, b] на рассматриваемой сетке – функция S(x), удовлетворяющая следующим условиям:

1.На каждом отрезке [] функция S(x) является полиномом третьей степени.

2.Фнкция S(x), ее первая и вторая производные непрерывны на сегменте [a, b].

3.S() = f() = , i = 0, …, n

4. (а) = (b) = 0.

Каждому сегменту [] сопоставим полином (x) вида:

(x) = + (x - ) + , x [], I = 1, …, n

Из указанных выше условий можно редуцировать систему к следующим уравнениям:

.

= 6 - ) I = 1, …, n – 1

В нашем случае и получим:

*(1)*

*, i = 1, …, n,*

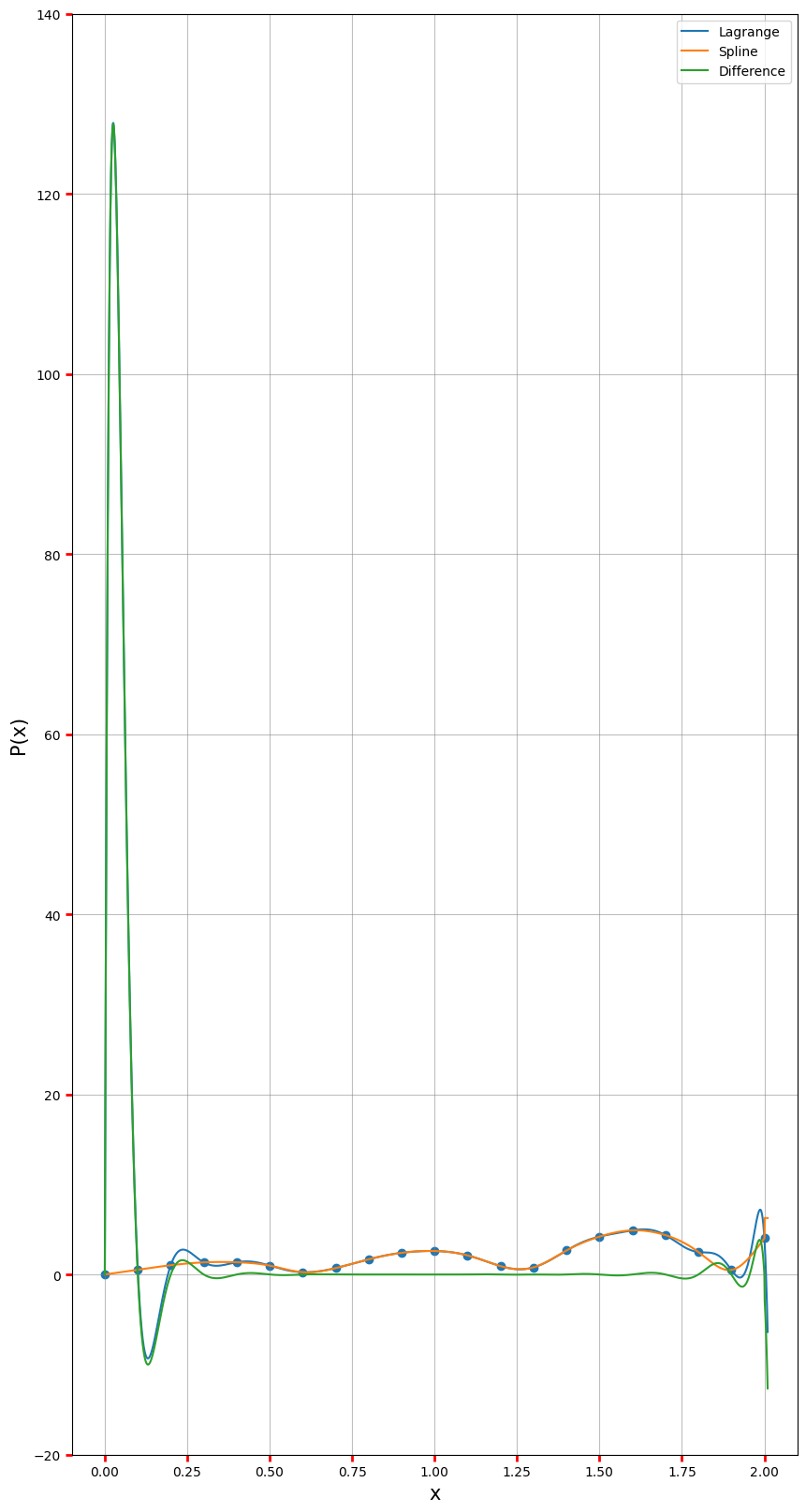
= , i = 1, …, n.

Для нахождения коэффициентов в (1) применим метод прогонки.

*Графическая реализация*

С помощью библиотеки mathplotlib в Jupiter notebook были построены графики полинома Лагранжа, кубического сплайна и их разности на равномерной сетке.

Точками отмечены начальные данные.



*Оценка методов*

На примере данного набора точек видно, что оба метода совпадают на большем множестве точек, хотя ближе к концам рассматриваемого промежутка сильно отличаются. Причем в нашем случае полином Лагранжа значительно отличается от значений в заданных точках на отрезках между узлами сетки, в то время как график, построенный методом кубических сплайнов, не имеет таких скачков.

**Код программы**

from matplotlib import pyplot as plt

import numpy as np

%matplotlib inline

plt.grid()

import matplotlib.pyplot as plt

plt.rcParams["figure.figsize"] = (10,20)

#полином Лагранжа

class razn(object): #разность двух элементов

def \_\_init\_\_(self, x, xi, xj):

self.raz = xi

self.vich = xj

def op(self, x, xi):

if(xi == '.'):

self.raz = x

return self.raz - self.vich

class proizved(object): #произведение 20-ти элементов(числетель и знаменатель Q)

def \_\_init\_\_(self,x,xi,arr):

self.tmp\_arr = [i for i in range(20)] #20

for i in range(20):

tmp = razn(x,xi,arr[i]);

self.tmp\_arr[i] = tmp.op(x,xi)

def mul(self):

tmp = 1

for i in range(20): #20

tmp \*= self.tmp\_arr[i]

return tmp

def mk\_poly(x, rangelist,data): #задание полинома

chisl = [i for i in range(21)]

znam = [i for i in range(21)]

Q = [i for i in range(21)]

for i in range(21): #числитель

k = 0

arr = [i for i in range(20)]

for j in range(21):

if(i != j):

arr[k] = rangelist[j]

k+=1

tmp = proizved(x,'.',arr) #x

chisl[i] = tmp.mul()

for i in range(21): #знаменатель

k = 0

arr = [i for i in range(20)]

for j in range(21):

if(i != j):

arr[k] = rangelist[j]

k+=1

tmp = proizved(0,rangelist[i],arr)

znam[i] = tmp.mul()

for i in range(21):

Q[i] = chisl[i]/znam[i]

ans = 0

for i in range(21):

ans += Q[i]\*data[i]

return ans

rangelist = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2]

data = [0, 0.529847, 1.027775, 1.346477, 1.356512, 0.986714, 0.257137, 0.706391, 1.684295, 2.404336, 2.606626, 2.11956, 0.927692, 0.789339, 2.664212, 4.203824, 4.900323, 4.370876, 2.493172, 0.502452, 4.019803]

#интерполирование сплайнами

h = 0.1 #шаг сетки

#метод прогонки для поиска C

A\_prog = [0.1 for i in range(0,19)]

B\_prog = [0.1 for i in range(0,19)]

C\_prog = [0.4 for i in range(0,19)]

F\_prog = [0.0 for i in range(0,19)]

alpha = [0.0 for i in range(0,20)]

beta = [0.0 for i in range(0, 20)]

C = [0.0 for i in range(0, 21)]

for i in range(1,20):

F\_prog[i-1] = 6\*(data[i-1] - 2\*data[i] + data[i+1])/h

alpha[0] = 0

beta[0] = 0

alpha[19] = 0.0

beta[19] = 0.0

# Прямой проход

for i in range(1, 20):

alpha[i] = -B\_prog[i-1] / (A\_prog[i-1] \* alpha[i-1] + C\_prog[i-1]);

beta[i] = (F\_prog[i-1] - A\_prog[i-1]\*beta[i-1]) / (A\_prog[i-1] \* alpha[i-1] + C\_prog[i-1]);

#Обратный проход

C[20] = 0

C[0] = 0

for i in range(19, 0, -1):

C[i] = (beta[i] + alpha[i]\*C[i+1])

#коэффициенты сплайна

#C.insert(0,0.0)

A = [0.0 for i in range(21)]

B = [0.0 for i in range(21)]

D = [0.0 for i in range(21)]

for i in range(1,21):

A[i] = data[i]

for i in range(1,21):

D[i] = (C[i] - C[i-1])/h

for i in range(1,21):

B[i] = 0.5\*h\*C[i] - (h\*\*2)\*D[i]/6 + (data[i] - data[i-1])/h

#графическое представление

x = [0]

y = [0]

x2 = [0]

y2 = [0]

x3 = [0]

y3 = [0]

step = 0.001

i = 0.001

k = 1

while i <= 2.01:

x.append(i)

x2.append(i)

x3.append(i)

y.append(mk\_poly(x[k],rangelist,data))

tmp = 1

while(i > rangelist[tmp] and i <= 2.0):

tmp += 1

y2.append(A[tmp] + B[tmp]\*(i-rangelist[tmp]) + C[tmp]\*((i-rangelist[tmp])\*\*2)/2 + D[tmp]\*((i-rangelist[tmp])\*\*3)/6)

y3.append(y[k]-y2[k])

i+=step

k+=1

plt.xlabel('x', fontsize=15)

plt.ylabel('P(x)', fontsize=15)

plt.xlim(-0.1, 2.1)

plt.ylim(-20, 140)

plt.tick\_params(axis='both', direction='out', length=5, width=2, labelcolor='black', colors='r', grid\_color='gray', grid\_alpha=0.5)

plt.scatter(rangelist,data)

plt.plot(x,y, label='Lagrange')

plt.plot(x2,y2, label='Spline')

plt.plot(x3,y3, label='Difference')

plt.legend()