МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное

учреждение высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

**Институт компьютерных технологий и информационной безопасности**

**Кафедра Математического обеспечения и применения ЭВМ**

**ОТЧЁТ**

по лабораторной работе №1

по курсу «МКиОДП»

Выполнили:

студенты группы КТмо1-3

Куприянова А.А.

Шепель И.О.

Проверил:

ассистент кафедры МОП ЭВМ

Жиглатый А.А.

Оценка

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2017 г.

Таганрог 2017

# Задание и цель работы

Цель: ознакомление с новым алгоритмом построения сплайновой кривой, базирующимся на дельта-преобразованиях второго порядка.

Задание: разработать и реализовать на компьютере программу построения и графического отображения интерполирующих одномерных сплайновых кривых на основе дельта-преобразований второго порядка в соответствии с заданным вариантом.

Вариант №1: дельта-сплайны с различным изменением шага дискретизации дискретизации (в функции от t) между узлами интерполяции.

# Математические основы построения сплайнов на основе дельта-преобразований второго порядка

В качестве сплайна на основе дельта-преобразований второго порядка (дельта-сплайна) будем понимать локально независимую одномерную функцию, проходящую через узлы интерполяции, которые задают общую форму кривой, обладающую гладкостью первого порядка. Локальная независимость означает, что значения отсчетов фрагмента сплайна между базовыми отсчетами можно определять независимо от других фрагментов сплайна. Эта функция характеризуется следующим краевым условиям и особенностями:

* фрагменты соседних сплайнов пересекаются (конечная точка одного фрагмента совпадает с начальной точкой следующего фрагмента);
* первые производные двух последовательно идущих фрагментов сплайна равны в точке (узле) их соединения, что гарантирует гладкость перехода от одного фрагмента к другому.
* модуль второй производной интерполирующей функции (кванта преобразования) между узлами интерполяции являются постоянной величиной.
* на интервале между узлами имеется два участка знакопостоянства (с противоположными знаками) кванта преобразования.

В общем случае участок сплайна между соседними отсчетами включает две траектории (рис. 1.1), которые реализуют участки «разгона» (траектория *B*) и «торможения» (траектория *A*).

Здесь   интерполирующая функция;   значение второй производной (кванта преобразования); – точка переключения знака кванта преобразования;  – краевые условия в узлах интерполяции; *T* – интервал между узлами интерполяции; *D* – расположение точки  переключения знака кванта преобразования.





























Рисунок 1. Интерполяционная кривая, включающая траектории *А* и *В*

В качестве исходных данных для расчёта сплайна используются значения отчётов в узлах интерполяции и интервалы между узлами. Возможно также априорное задание значений производных в узлах, а так же значения шага дискретизации Δt формируемой интерполируемой кривой на интервале между узлами.

Алгоритмически решение данной задачи можно представить в виде следующей последовательности действий:

1. Определение (при необходимости)  и , шага Δt;

2. Нахождение параметров *P* (модуль кванта преобразования) и *D*;

3. Построение собственно сплайна на основе найденных параметров.

# Алгоритм построения сплайновой кривой на основе дельта-преобразований второго порядка

В ходе выполнения лабораторной работы был разработан следующий алгоритм построения сплайновой кривой:

1. Задаются (считываются из файла) следующие данные:

* tn и yn, – базовые отсчёты;
* Kn – количество шагов дискретизации для каждого из участка сплайна.

1. Для каждого базового отсчёта рассчитываются значения первых производных в базовых отсчётах. Производные в крайних точках принимаются равными нулю. Во всех остальных точках производные рассчитываются согласно следующему алгоритму:

В случае, если для промежуточного узла выполняются соотношения

или

то принимать

В противном случае значения производных в узлах определять как среднее арифметическое двух производных для узлов слева и справа от узла в соответствии с выражением

1. Для каждого сплайна рассчитывается длина участка сплайна:

,

а также шаг дискретизации ∇tn:

.

1. Считая шаг дискретизации ∇tn постоянным в пределах участка сплайна, программа рассчитывает массив t аргументов функции Y(t) интерполяционной кривой:

ti, , .

1. Рассчитывается сама сплайновая кривая, то есть массив Y значений функции Y(t):

Yi, ,

а также массив D – расстояние от правой точки сплайна до точки переключения кванта преобразования.

Формулы расчёта параметров для каждого сплайна:









Построение участка *n* сплайна:

Из левого узла сплайна (траектория *В*) строится кривая с шагом  до точки  переключения кванта преобразования. Расчет выполняется по формуле:

Далее кривая строится от точки  до правого узла сплайна (траектория *А*) с тем же шагом . Расчет выполняется по формуле:

; 

1. Производится запись в файл данных массивов Y и t.
2. По данным массивов Y и t строится график интерполяционной кривой. Также на графике отображаются базовые отсчёты и точки переключения кванта преобразования.

# Выводы о проделанной работе

В ходе выполнения лабораторной работы был разработан и реализован на языке Python алгоритм построения сплайновой кривой на основе дельта-преобразований второго порядка.

# Результаты работы программы

Для тестовых входных данных

tn = [0, 20, 35, 47, 56, 63, 69, 80, 90, 120],

yn = [100, 90, 110, 60, 45, 10, -30, 15, 45, 200],

Kn = [3, 7, 8, 5, 3, 5, 7, 5, 9]

программа выдаёт следующий результат в выходной файл:

|  |  |
| --- | --- |
| t | y(t) |
| 0.0 | 100.0 |
| 6.666666666666667 | 97.77777777777777 |
| 13.333333333333334 | 92.22222222222223 |
| 20.0 | 90.0 |
| 22.142857142857142 | 90.81632653061224 |
| 24.285714285714285 | 93.26530612244898 |
| 26.428571428571427 | 97.3469387755102 |
| 28.57142857142857 | 102.65306122448979 |
| 30.71428571428571 | 106.73469387755101 |
| 32.857142857142854 | 109.18367346938776 |
| 35.0 | 110.0 |
| 36.5 | 108.91543659920963 |
| 38.0 | 105.66174639683854 |
| 39.5 | 100.23892939288669 |
| 41.0 | 92.64698558735412 |
| 42.5 | 82.88591498024081 |
| 44.0 | 73.08825360316146 |
| 45.5 | 65.45956340079037 |
| 47.0 | 60.0 |
| 48.8 | 55.80533008588991 |
| 50.599999999999994 | 53.72132034355964 |
| 52.39999999999999 | 52.77867965644037 |
| 54.19999999999999 | 49.94466991411011 |
| 56.0 | 45.0 |
| 58.333333333333336 | 35.87333594346394 |
| 60.66666666666667 | 24.048899329411334 |
| 63.0 | 10.0 |
| 64.2 | 0.9598245749008356 |
| 65.4 | -11.638421174107815 |
| 66.60000000000001 | -21.839298299603506 |
| 67.80000000000001 | -27.9598245749009 |
| 69.0 | -30.0 |
| 70.57142857142857 | -28.82446118696839 |
| 72.14285714285714 | -25.297844747873558 |
| 73.71428571428571 | -19.42015068271551 |
| 75.28571428571428 | -11.191378991494236 |
| 76.85714285714285 | -0.8450123949836605 |
| 78.42857142857142 | 8.253032615539754 |
| 80.0 | 15.0 |
| 82.0 | 21.422090963963463 |
| 84.0 | 26.506545674035664 |
| 86.0 | 31.34193917444918 |
| 88.0 | 37.50215146027896 |
| 90.0 | 45.0 |
| 93.33333333333333 | 61.151048543483 |
| 96.66666666666666 | 82.38197195170982 |
| 99.99999999999999 | 108.69277022468046 |
| 103.33333333333331 | 136.50156419070197 |
| 106.66666666666664 | 159.3610010820492 |
| 109.99999999999997 | 177.14056310865263 |
| 113.3333333333333 | 189.84025027051223 |
| 116.66666666666663 | 197.46006256762803 |
| 120.0 | 200.0 |

и рисует график (рисунок 2). На графике голубым цветом обозначена построенная сплайновая кривая, сплошными вертикальными линиями обозначены границы участков сплайна, пунктирными – точки переключения кванта преобразования.

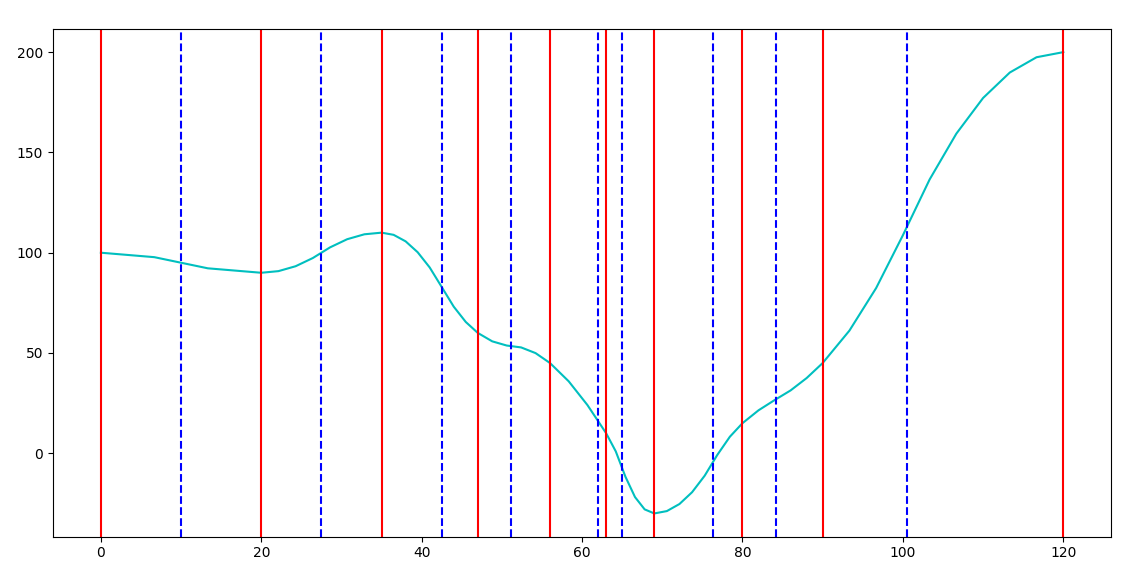


Рисунок 2. – Результат работы программы

# Листинг программы

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

def read\_data(fname): #чтение данных из файла

df = pd.read\_csv(fname, sep=" ", escapechar="#")

in\_val = df.values

tn = in\_val[:, 0]

yn = in\_val[:, 1]

Kn = in\_val[:, 2]

Kn = Kn[~np.isnan(Kn)]

Kn = np.array(Kn, dtype=np.int16)

return tn, yn, Kn

def write\_data(fname, t, Y): #запись данных в файл

out\_df = pd.DataFrame(

{"t" : t,

"y(t)" : Y}

)

out\_df.to\_csv(fname, sep="\t", index=False)

def steps(tn): # вычисление длин участков сплайна и величины шага дискретизации для каждого сплайна

N = yn.shape[0]

for i in range(N - 1):

if i == (N - 1):

T[i] = tn[N] - tn[N - 1]

else:

T[i] = tn[i+1] - tn[i]

dt[i] = (T[i]\*1.0)/((Kn[i])\*1.0)

return T, dt

def get\_t(dt, Kn): #вычисление аргументов сплайна

ind = 0

N = dt.shape[0] + 1

PointCnt = Kn.sum()

t = np.zeros(PointCnt + 1)

t[PointCnt] = tn[N - 1]

for i in range(N - 1):

for j in range((Kn[i])):

if ind >= PointCnt:

break

if j == 0:

t[ind] = tn[i]

ind = ind + 1

continue

else:

t[ind] = t[ind - 1] + dt[i]

ind = ind+1

return t

def calc\_dy(yn, tn): #вычисление производных

N = yn.shape[0]

dy = np.zeros(N)

for i in range(1, N-1):

if (yn[i-1]< yn[i] > yn[i+1] or yn[i-1]> yn[i] < yn[i+1]):

dy[i] = 0

else:

dy[i] = (yn[i] - yn[i-1]) / (tn[i] - tn[i-1]) + (yn[i+1] - yn[i])/(tn[i+1]-tn[i])

dy /= 2

return dy

def calc\_spline2D(yn, tn, dy, T, Kn, t, dt): #построение сплайна

ind = 0

N = yn.shape[0]

P = np.zeros(N - 1)

D = np.zeros(N - 1)

PointCnt = Kn.sum()

Y = np.zeros(PointCnt + 1)

for i in range(N-1):

L = yn[i+1] - yn[i] - 0.5\*T[i]\*(dy[i+1]+dy[i])

P[i] = (-L - np.sign(L)\*np.sqrt(L\*L + 0.25\*T[i]\*T[i]\*pow(dy[i+1]-dy[i], 2))) / (0.5\*T[i]\*T[i])

if P[i] == 0 or L == 0:

D[i] = 0

else:

D[i] = (dy[i+1]-dy[i]+T[i]\*P[i]) / (2\*P[i])

tp = tn[i + 1] - D[i]

for j in range((Kn[i])):

if ind >= PointCnt:

break

if j == 0:

Y[ind] = yn[i]

ind = ind + 1

continue

tL = 1.0 \* (t[ind] - tn[i])

if t[ind] < tp:

Y[ind] = yn[i] + dy[i] \* tL - (tL \* 1.0 \* tL \* P[i]) / 2.0

else:

tR = T[i] - tL

Y[ind] = yn[i + 1] - dy[i + 1] \* tR + pow(tR, 2) \* P[i] / 2.0

ind = ind + 1

Y[PointCnt] = yn[N - 1]

return Y, D

def print\_spline2D(t, Y, tn, D): #построение графика

plt.plot(t,Y, marker="", markersize="4", c="C")

for i, v in enumerate(tn):

plt.axvline(v, ls="-", c="R")

if i == 0:

continue

plt.axvline(tn[i] - D[i-1], ls="--", c="B")

plt.show()

tn, yn, Kn = read\_data("in.txt")

N = yn.shape[0]

dy = np.zeros(N)

dy = calc\_dy(yn, tn)

#параметры сплайнов

T = np.zeros(N - 1)

dt = np.zeros(N-1, float)

P = np.zeros(N - 1)

D = np.zeros(N - 1)

T, dt = steps(tn)

t = get\_t(dt, Kn)

Y, D = calc\_spline2D(yn, tn, dy, T, Kn, t, dt)

write\_data("out.txt", t, Y)

print\_spline2D(t, Y, tn, D)