|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № 1**

|  |  |
| --- | --- |
| **Тема:** Линейная интерполяция с помощью полинома Ньютона  **Студент:** Казакова Элиза  **Группа:** ИУ7-46Б  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель :** Градов В. М. |  |

Москва.

2020 г.

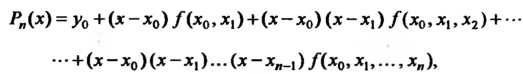
**Входные данные:** таблица с данными (x, y), степень полинома, X

**Результат работы программы:** Y(X), корень функции заданной таблицей(найденный методом половинного деления), корень функции (найденный обратным интерполированием).

**Алгоритм:**

Задача интерполяции – найти функцию http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image108.png, принимающую в точках http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image109.png те же значения http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image110.png. При этом предполагается, что среди значений http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image109.png нет одинаковых. Точки http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image109.png называют узлами интерполяции.  http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image108.png ищется только на отрезке http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image104.png , программа не рассчитывает экстраполяцию.

При создании интерполяционного полинома методом Ньютона для полинома используется такое выражение c разделенными разностями:



**Разделенные разности** нулевого порядка совпадают со значениями функции в узлах. Разделенные разности первого порядка определяются через разделенные разности нулевого порядка:

http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image154.png

Разделенные разности второго порядка определяются через разделенные разности первого порядка:

http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image155.png

Разделенные разности *k*-го порядка определяются через разделенные разности порядка http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image156.png:

http://aco.ifmo.ru/el_books/numerical_methods/lectures/images/image157.png

За точностью расчета можно следить по убыванию членов суммы.

С помощью функции read\_file считываем данные из файла в два массива x и y. Сортируем массив x и соответственно y(функция sort\_buble). Затем производим выборку узлов(функция find\_nodes).Рассчитываем для них разделенные разницицы(функция razd\_razn).И рассчитываем значение функции с помощью полинома Ньютона(функция poly).

**Код:**

from math import \*

def razd\_razn(X, Y, n):

RR = []

RR.append(X)

RR.append(Y)

for i in range(n-1):

T = []

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

T.append((RR[i+1][j]-RR[i+1][j+1])/(RR[0][j] - RR[0][i+j+1]))

RR.append(T)

return RR

def poly(RR, n, x):

p = RR[1][0];

for i in range(1, n):

tek = 1

for j in range(i):

tek = tek\*(x-RR[0][j])

p = p +tek\*RR[i+1][0];

return p;

#Сортировка пузырьком, сортирует два массива

def sort\_buble(x, y, N):

for i in range(N-1):

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

if x[j] > x[j+1]:

x[j], x[j+1] = x[j+1], x[j]

y[j], y[j+1] = y[j+1], y[j]

#Проверка экстраполяции для отсортированного массива

def extropol(xf, x, n):

if xf < x[0] or xf > x[n-1]:

return 1

else:

return 0

#Чтение значений функций в два массива

def read\_file(x, y, name):

f = open(name)

count = 0

for line in f:

line = line.split(' ')

x.append(float(line[0]))

y.append(float(line[1]))

count += 1

return count

#Функция для набора конфигураций(без учета экстраполяции)

def find\_nodes(xf, x, y, count, nodes):

xn = []

yn = []

d = left = 0

for k in range(count):

if xf < x[k]:

xn.append(x[k])

yn.append(y[k])

d += 1 #кол-во узлов уже записаных

break

left += 1

c = 1

right = count - left

while d < nodes:

if left > 0 :

xn.append(x[k - c])

yn.append(y[k - c])

left -= 1

d += 1

if d == nodes:

break

if right > 0:

xn.append(x[k + c])

yn.append(y[k + c])

d += 1

right -= 1

c += 1

return xn, yn

def bisection(a, b, x1, y, n, nodes):

xn, yn = find\_nodes(xf, x1, y, count, nodes) #находим узлы

RR = razd\_razn(x1, y, nodes)

x = (a + b) /2

while (b - a)/ 2 > 0.000001:

if (poly(RR, nodes, a) == 0.0):

return a

if (poly(RR, nodes, b) == 0.0):

return b

if (poly(RR, nodes, x) == 0.0):

return x

if (poly(RR, nodes, a) \* poly(RR, nodes, x) > 0):

a = x

else:

b = x

x = (a + b) / 2

return x

#Ввод данных

xf = float(input("Input x: "))

nodes = int(input("Input polynomial degree: ")) + 1

nodes1 = int(input("Input polynomial degree for root: ")) + 1

nodes2 = int(input("Input polynomial degree for bisection: ")) + 1

a = [0 for i in range(nodes)] #массив с коэффициентами

x = []

y = []

count = read\_file(x,y,"func1.txt")#считываем из файла данные в два массива

def P(x, y, xf, count, nodes):

sort\_buble(x, y, count)

if extropol(xf, x, count) == 1:

print("This program does not include extrapolation")

else:

xn, yn = find\_nodes(xf, x, y, count, nodes) #находим узлы

sort\_buble(xn, yn, nodes)

RR = razd\_razn(xn, yn, nodes) #матрица с разделенными разницами

print("Y(X) = ", poly(RR, nodes, xf))

return poly(RR, nodes, xf)

if (count < nodes):

print("Number of nodes more than data")

else:

P(x,y,xf,count, nodes)

x, y = y, x

print("Back interpolation")

if P(x,y,0.0,count, nodes1) != x[0]:

x, y = y, x

print("root(half division method) = ", bisection(x[0], x[count-1], x, y, count, nodes2))