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 №2

Тема Построение и программная реализация алгоритма многомерной интерполяции табличных функций.

Студент Бугаенко А.П.

Группа ИУ7-45Б

Оценка (баллы) _______________

Преподаватель Градов В.М.

Москва.
2021 г

Цель работы. Получение навыков построения алгоритма интерполяции таблично заданных функций двух переменных.

1 Исходные данные

Таблица функции с количеством узлов 5x5.

y \ x	0	1	2	3	4
0	0	1	4	9	16
1	1	2	5	10	17
2	4	5	8	13	20
3	9	10	13	18	25
4	16	17	20	25	32

2. степени аппроксимируемых полиномов n_x и n_y.
3. значения x, y для которых выполняется интерполяция.

2 Код программы

Код программы представлен на листингах ниже.

Листинг newton.py

```
def Divided_diff(x, y):
    new_y = []
    n = len(x) - len(y)
    for i in range(0, len(y) - 1):
        new_y.append((y[i] - y[i + 1]) / (x[i] - x[i + n + 1]))
    return new_y

ddef SortTableNewton(table, x, n):
    table = sorted(table, key=lambda d: abs(d["x"] - x))
    table = sorted(table[:n+1], key=lambda t: t["x"])
    return table

def SortTableNewton(table, x, n):
    table = sorted(table, key=lambda d: abs(d["x"] - x))
    table = sorted(table[:n+1], key=lambda t: t["x"])
    return table

def FormXYNewton(table):
    X = []
    Y = []
    for row in table:
        X.append(row["x"])
        Y.append(row["y"])
    return X, Y
```

```
def CountDiffDivNewton(X, Y):
    Y_arr = [Y]
    while len(Y_arr[-1]) != 1:
        Y = Divided_diff(X, Y)
        Y_arr.append(Y)
    return Y_arr

def CountPolynomNewton(Y_arr):
    polym = []
    for Y in Y_arr:
        polym.append(Y[0])
    return polym

def GetValApproxNewton(polym, X, x):
    x_mult = 1
    result = 0
    for i in range(0, len(polym)):
        result += x_mult * polym[i]
        x_mult = x_mult * (x - X[i])
    return result

def NewtonApprox(x, n, table):

    table = SortTableNewton(table, x, n)
    X, Y = FormXYNewton(table)

    Y_arr = CountDiffDivNewton(X, Y)
    polym = CountPolynomNewton(Y_arr)

    return GetValApproxNewton(polym, X, x)
```

Листинг functions.py

```
import numpy as np
import newton as nwt

def ConvertVecXToTable(x, y):
    table = []
    for i in range(0, len(x)):
        table.append({"x": x[i], "y": y[i]})
    return table

def InterpolXY(nx, ny, x, y, table):
    vector_x = []
    for i in range(0, len(table["x"])):
        x_table = ConvertVecXToTable(table["x"], table["z"][i])
        vector_x.append(nwt.NewtonApprox(x, nx, x_table))
    y_table = ConvertVecXToTable(table["y"], vector_x)
    return nwt.NewtonApprox(y, ny, y_table)
```

Листинг main.py

```
import pandas as pd import functions as fc

import pandas as pd
import functions as fc

table = {
    "x": [0, 1, 2, 3, 4],
    "y": [0, 1, 2, 3, 4],
    "z":
    [[0, 1, 4, 9, 16],
    [1, 2, 5, 10, 17],
    [4, 5, 8, 13, 20],
    [9, 10, 13, 18, 25],
    [16, 17, 20, 25, 32]]
}

def FormResultApproxTable(x, y, table, out_name):
    results = {"nx": [], "ny": [], "z": []}

    for n in range(1, 4):
        results["nx"].append(n)
        results["ny"].append(n)
        results["z"].append(round(fc.InterpolXY(n, n, x, y, table), 5))
    df = pd.DataFrame(results)
    df.to_csv(out_name + ".csv", index=False)

FormResultApproxTable(1.5, 1.5, table, "out")
```

3 Результаты работы

1. Результат интерполяции z(x,y) при степенях полиномов 1,2,3 для x=1.5, y=1.5.

nx	ny	z
1	1	5.0
2	2	4.5
3	3	4.5

4 Вопросы при защите лабораторной работы

1. Пусть производящая функция таблицы суть $z(x,y)=x^2+y^2$. Область определения по x и y 0-5 и 0-5. Шаги по переменным равны 1. Степени $n_x = n_y = 1$, x=y=1.5. Приведите по шагам те. значения функции, которые получаются в ходе последовательных интерполяций по строкам и столбцу.

Для начала нам нужно рассчитать таблицу значений z для заданных векторов x, y с помощью функции z(x, y). В нашем случае она выглядит так:

x\y	0	1	2	3	4	5
0	0	1	4	9	16	25
1	1	2	5	10	17	26
2	4	5	8	13	20	29
3	9	10	13	18	25	34
4	16	17	20	25	32	41
5	25	26	29	34	41	50

Теперь мы можем начать расчёт значений, которые мы получаем при последовательной интерполяции. Разницы, интерполировать сначала значения x или y, нет, поскольку алгоритмически это не имеет значения, брать x или y, к тому же значения x и y одинаковы, а таблица симметрична относительно диагонали. Для наших расчётов мы возьмём сначала y, потом x.

Степени вычисляемых полиномов $n_x = n_y = 1$, то есть мы строим и для x и для y полиномы первой степени. При расчёте интерполяций в плоскости yz, мы берём значение y = 1.5 и интерполируем столбцы с помощью интерполяции Ньютона.

Так как степень полинома $n_y = 1$, то нам нужно взять $n_y + 1$ узлов,

находящихся ближе всего к нашему значению y. Тогда у нас получится

следующая таблица:

y	z_0	z_1	z_2	z_3	z_4	z_5
1	1	2	5	10	17	26
2	4	5	8	13	20	29

И тогда для каждой "подтаблицы" для y мы вычисляем разделённые разности и подставляем значения в полиномы первой степени:

yz_0:
Разделённая разность: (1 - 4) / (1 - 2) = -3 / -1 = 3
Полином: yz(y) = 1 + 3(y - 1) = 1 + 3(1.5 - 1) = 2.5
yz_1:
Разделённая разность: (2 - 5) / (1 - 2) = -3 / -1 = 3
Полином: yz(y) = 1 + 3(y - 1) = 2 + 3(1.5 - 1) = 3.5
yz_2:
Разделённая разность: (5 - 8) / (1 - 2) = -3 / -1 = 3
Полином: yz(y) = 1 + 3(y - 1) = 5 + 3(1.5 - 1) = 6.5
yz_3:
Разделённая разность: (10 - 13) / (1 - 2) = -3 / -1 = 3
Полином: yz(y) = 1 + 3(y - 1) = 10 + 3(1.5 - 1) = 11.5
yz_4:
Разделённая разность: (17 - 20) / (1 - 2) = -3 / -1 = 3
Полином: yz(y) = 1 + 3(y - 1) = 17 + 3(1.5 - 1) = 18.5
yz_5:
Разделённая разность: (26 - 29) / (1 - 2) = -3 / -1 = 3
Полином: yz(y) = 1 + 3(y - 1) = 26 + 3(1.5 - 1) = 27.5

В итоге мы получаем вектор значений y, который теперь нам нужно интерполировать по значению x. $n_x = 1$, x = 1.5, тогда ближайшие значения это x = 1 и x = 2. Соответствующие y: y_1 = 3.5 и y_2 = 6,5. Вычисляем разделённую разность:
(3.5 - 6.5) / (1 - 2) = 3
z(x, y) = 3.5 + 3(1.5 - 1) = 5 - итоговое значение.

2. Какова минимальная степень двумерного полинома, построенного на четырех узлах? На шести узлах?

Пусть полином, построенный на 4-х узлах - P(x, y). Тогда мы можем сказать, что для каждого из четырёх узлов, задаваемых координатами x, y и z будет верно, что для каждого i = 1..4 выполняется $P(x_i, y_i) = z_i$. Предположим, что

полином, который мы построили, первой степени. Тогда мы можем представить его в следующем виде:
$P(x, y) = k_1 + k_2(x - x_1) + k_3(y - y_1)$
В данном полиноме есть три неизвестных коэффициента - k_1, k_2 и k_3. Так ранее мы показали, что для полинома на 4-х узлах существует четыре ограничивающих уравнения $P(x_i, y_i)$, то мы можем сказать, что полином первой степени не может быть построен на 4-х узлах (в общем случае) и поэтому минимальная степень такого полинома должна быть больше или равна двум.
В случае же интерполяции по шести узлам мы можем рассмотреть полином второй степени. Количество неизвестных коэффициентов данного полинома будет равняться 6, что значит, что их можно будет однозначно определить с помощью 6 уравнений, которые в свою очередь могут быть составлены для каждого из узлов. Отсюда мы можем сказать, что минимальная степень интерполяционного полинома на 6 узлах будет равняться двум.

3. Предложите алгоритм двумерной интерполяции при хаотичном расположении узлов, т.е. когда таблицы функции на регулярной сетке нет, и метод последовательной интерполяции не работает. Какие имеются ограничения на расположение узлов при разных степенях полинома?

При отсутствии таблицы на регулярной стеке можно провести интерполяцию по трем точкам: если f_A, f_B, f_C — значение функции $f(x, y)$ в вершинах A, B, C некоторого треугольника, то вычислить приближенное значение функции внутри этого треугольника можно с помощью билинейной функции $f(x, y) \approx F(x, y) = ax + by + c$, находя коэффициенты a, b, c из условий
$ax_A + by_A + c = f_A$
$ax_B + by_B + c = f_B$
$ax_C + by_C + c = f_C$,
где $\{x_A, y_A\}$, $\{x_B, y_B\}$, $\{x_C, y_C\}$ - координаты вершин A, B , C . Погрешность такой интерполяции для функции $f(x, y)$ с непрерывными вторыми производными будет O(h2) , где h — длина наибольшей стороны треугольника ABC. Основное требование данного метода к положению узлов состоит в том, что они не должны лежать на одной прямой.

4. Пусть на каком-либо языке программирования написана функция, выполняющая интерполяцию по двум переменным. Опишите алгоритм использования этой функции для интерполяции по трем переменным.

По сути для того, чтобы реализовать интерполяцию по трём переменным нам необходимо попросту дописать ещё один "интерполяционный слой" для уже имеющегося кода. И тогда алгоритм будет выглядеть следующим образом: Положим, что в качестве входных данных у нас есть многомерная таблица, содержащая в себе значения для параметров x, y, z и t, являющегося тем параметром, который нам надо получить. Также на входе мы имеем значения x_0, y_0 и z_0, для которых нам нужной найти значение t. И при этом мы имеем степени nx, ny и nz для полиномов. Сначала мы проведём аппроксимацию значений для z с помощью ранее написанной функции. После этого мы собираем все значения z в вектор, который аппроксимируем с помощью интерполяции Ньютона и на выходе получаем нужное нам значение t.

5. Можно ли при последовательной интерполяции по разным направлениям использовать полиномы несовпадающих степеней или даже разные методы одномерной интерполяции, например, полином Ньютона и сплайн?

Можно, поскольку нет разницы, как мы получили значения при предыдущей интерполяции. Алгоритм принимает на вход только вектор (или матрицу) уже интерполированных значений, и не важен способ, которым мы их получили. Однако стоит учесть, что использование различных степеней и алгоритмов при расчёте значения может сильно повлиять на итоговое значение как в лучшую так и в худшую сторону, поскольку из-за того, что значения разных уровней интерполяции зависят друг от друга если мы интерполировали с большой погрешностью в начале, то эта погрешность останется с нами до последнего значения.

6. Опишите алгоритм двумерной интерполяции на треугольной конфигурации узлов.

Распишем алгоритм двумерной интерполяции для треугольной конфигурации узлов на нерегулярной сетке. Пусть у нас есть три узла, для которых необходимо найти интерполированное значение, и которые являются ближайшими тремя узлами из всех узлов нерегулярной сетки. Полином первой степени будет выглядеть следующим образом:
$z = a + bx + cy$
Составим уравнения для коэффициентов из условий для каждого из узлов:
$z_1 = a + bx_1 + cy_1$
$z_2 = a + bx_2 + cy_2$
$z_3 = a + bx_3 + cy_3$
Коэффициенты легко находятся с помощью решения матричного уравнения, так как количество уравнений равно количеству неизвестных. В итоге мы получаем полином первой степени и вычисляем значение.