Министерство образования и науки 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 (ИУ)

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

Лабораторная работа №2

Тема: Построение и программная реализация алгоритма многомерной интерполяции табличных функций

Студент: Хамзина Р.Р.

Группа: ИУ7-43Б

Оценка (баллы): _____

Преподаватель: Градов В.М.

Москва.
2021 г.

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

1 Исходные данные

1. Таблица функции.

y\x	x_0	x_1	...
y_0	$f(x_0,y_0)$	$f(x_1,y_0)$	...
y_1	$f(x_0,y_1)$	$f(x_1,y_1)$	...
...	...	...	...

Для отладки:

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. Степень аппроксимирующих полиномов.

Степень полинома по координате x — n_x, степень полинома по координате y - n_y.

Для отладки: $n_i = 1, 2, 3$, где $i \in \{x, y\}$.

3. Значение аргументов x, y, для которого выполняется интерполяция.

Для отладки: x = 1.5, y = 1.5.

2 Код программы

Код программы представлен на листингах.

Листинг 1 — NewtonInterpolation.py

```
COUPLE = 2
```

```
def SortTable(Table, SizeTable):
    """
        Сортировка таблицы по возрастанию.
    """

    for i in range(SizeTable - 1):
        MinIndex = i
        for j in range(i + 1, SizeTable):
            if Table[j][0] < Table[MinIndex][0]:
                MinIndex = j

        Table[MinIndex], Table[i] = Table[i], Table[MinIndex]
    return Table

def CreateConfig(Table, SizeTable, power, argument):
    """
        Построение конфигурации узлов из таблицы Table
        размера SizeTable для построение полинома степени
        power при аргументе argument.
    """
    center = 0

    while center < SizeTable:
        if Table[center][0] >= argument:
            break
        center += 1

    if center == 0:
        return Table[:power + 1]

    if center == SizeTable:
        return Table[SizeTable - power - 1:]

    if abs(Table[center][0] - argument) > abs(argument - Table[center - 1][0]):
        center -= 1

    low = center - power // 2 - 1
    top = center + power // 2 + 1

    if power % 2 == 0:
```

```
            return Table[center - power // 2:top]

        if abs(Table[top][0] - argument) > abs(argument - Table[low][0]):
            return Table[low:top]

        return Table[low: top + 1]

def CreateSplitDiff(Table, power):
    """
        Построение таблицы разделенных разностей.
        Параметры выбираются из таблицы Table,
        степень полинома power.
    """

    SplitDiff = []
    diffs = []

    for i in range(power + 1):
        diffs.append(Table[i][1])

    SplitDiff.append(diffs)

    for i in range(power):
        size = len(SplitDiff)
        diffs = []
        DiffX = Table[0][0] - Table[i + 1][0]

        for j in range(1, len(SplitDiff[size - 1])):
            DiffY = SplitDiff[size - 1][j - 1] - SplitDiff[size - 1][j]
            diffs.append(DiffY/DiffX)

        SplitDiff.append(diffs)

    return SplitDiff

def NewtonPolynomial(Config, power, argument, SplitDiff):
    """
        Получение значения интерполяционного полинома
        Ньютона степени power при аргументе argument.
        Начальная конфигурация Config, таблица разде-
```

```
        ленных разностей SplitDiff.
    """

    result = SplitDiff[0][0]
    factor = 1

    for i in range(power):
        factor *= argument - Config[i][0]
        result += SplitDiff[i + 1][0] * factor

    return result

def NewtonInterpolation(Table, SizeTable, power, argument):
    """
        Значение интерполяцинного полинома Ньютона
        при заданной степени power и аргументе argument.
        Параметры выбираются из таблицы Table размером
        SizeTable.
    """

    Table = SortTable(Table, SizeTable)
    Config = CreateConfig(Table, SizeTable, power, argument)
    SplitDiff = CreateSplitDiff(Config, power)
    result = NewtonPolynomial(Config, power, argument, SplitDiff)

    return result
```

Листинг 2 — TwoNewtonInterpolation.py

```
from NewtonInterpolation import NewtonInterpolation

def TwoNewtonIntrepolation(StartTable, X, Y, PowerX, PowerY,
ArgumentX, ArgumentY):
    """
        Интерполяция функции двух переменных.
        X, Y - значения координат x и y соответственно.
        StartTable - таблица со значениями z(x,y).
        PowerX, PowerY - степени по координатам x и y
        соответственно.
        ArgumentX, ArgumentY - значения аргументов x и y
        соотвественно.
    """
```

```
    TableY = []

    for i in range(len(Y)):
        column = []
        column.append(Y[i])
        TableY.append(column)

    for i in range(len(StartTable)):
        TableX = []

        for j in range(len(StartTable[i])):
            row = []
            row.append(X[j])
            row.append(StartTable[i][j])
            TableX.append(row)

        ResultX = NewtonInterpolation(TableX, len(TableX), PowerX, ArgumentX)
        TableY[i].append(ResultX)

    result = NewtonInterpolation(TableY, len(TableY), PowerY, ArgumentY)

    return result
```

3 Результаты работы

1) Интерполяция по координате x.

1. Строим таблицу из значений x и значений $f(x, y_j)$, j = 0, 1, …, n_y.

2. Сортируем таблицу в порядке возрастания/убывания.

3. Из заданной таблицы TableX строим конфигурацию узлов Config, которые примыкают к заданному значению аргумента x:

x	$f(x, y_j)$
x_0	$f(x_0, y_j)$
x_1	$f(x_1, y_j)$
…	…

4. Из конфигурации узлов Config строим таблицу разделенных разностей ($z(x_{i,}, x_t) = (f(x_i,y_j) - f(x_t,y_j)/ (x_i — x_t))$ SplitDiff:

$f(x_0, y_j)$	$(f(x_0, y_j) — f(x_1, y_j)) / (x_0 — x_1)$	$((f(x_0, y_j) — f(x_1, y_j)) / (x_0 — x_1) — (f(x_1, y_j) — f(x_2, y_j)) / (x_1 — x_2)) / (x_0 — x_2)$
$f(x_1, y_j)$		
$f(x_2, y_j)$	$(f(x_1, y_j) — f(x_2, y_j)) / (x_1 — x_2)$	

5. Выбираем значения из первой строки каждого столбца таблицы разделенных разностей и для значения аргумента x находим полином Ньютона степени n_x:

$$P_n(x) = y_0 + \sum_{k=0}^{n} (x - x_n) \dots (x - x_{k-1}) y(x_0, x_1, \dots, x_k)$$

6. Пункты 1-5 повторяем j раз, строя таблицу TableY:

y	$z(x, y_j)$
y_0	$z(x, y_0)$
y_1	$z(x, y_1)$
...	...

2) Интерполяция по координате y.

1. Повторяем пункты 2-5 к таблице TableY, для значения y и степени n_y.

Для x = 1.5 и y = 1.5 получается следующая таблица значений:

n_x	n_y	Значение полинома
1	1	5.0
1	2	4.75
1	3	4.75
2	1	4.75
2	2	4.5
2	3	4.5

3	1	4.75
3	2	4.5
3	3	4.5

4 Вопросы при защите лабораторной работы

1. Пусть производящая функция таблицы : $z(x, y) = x^2 + y^2$. 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.

Из условия задания строим таблицу:

y\x	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

Так как $x = y = 1.5$ и степени $n_x = n_y = 1$, то рабочая таблица :

y\x	1	2
1	2	5
2	5	8

1. Проведем интерполяцию по x:

1) y = 1, x = 1, 2

$p(x_0, x_1, y_0) = (f(x_0, y_0) — f(x_1, y_0)) / (x_0 — x_1) = (2 - 5) / (1 - 2) = 3$

$z(x_0, x_1, y_0) = f(x_0, y_0) + p(x_0, x_1, y_0)(x — x_0) = 2 + 3(1.5 — 1) = 3.5$

2) y = 2, x = 1, 2

$p(x_0, x_1, y_1) = (f(x_0, y_1) — f(x_1, y_1)) / (x_0 — x_1) = (5 — 8) / (1 — 2) = 3$

$z(x_0, x_1, y_1) = f(x_0,y_1) + p(x_0, x_1, y_1)(x — x_0) = 5 + 3(1.5 — 1) = 6.5$

Получилась таблица:

y	$z(x, y_j)$
1	3.5
2	6.5

2. Проведем интерполяцию по y:

$z(x_0, x_1, y_0, y_1) = (z(x_0, x_1, y_0) - z(x_0, x_1, y_0)) / (y_0 — y_1) = (3.5 - 6.5) / (1 - 2) = 3$

$P_1(x, y) = z(x_0, x_1, y_0) + z(x_0, x_1, y_0, y_1)(y — y_0) = 3.5 + 3(1.5 — 1) = 5.0$

Результат : $P_1(x, y) = 5.0$

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

Исходя из условия можно построить полином 0-ой степени $P_0(x, y) = f(x_0, y_0)$, поэтому минимальная степень при заданных условиях — 0 (и для четырех узлов, и для шести узлов).

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?

Алгоритм:

1. Представляем интерполяционный полином по формуле, например для первой степени : $z = a + bx + cy$:
2. Выбираем шесть узлов, ближайших к точке интерполяции, с учетом ограничений для степени полинома.
3. Составляем систему уравнений выбранных шести узлов: $z_i = a + bx_i + cy_i$, i = 0, 1, 2.
4. Находим коэффициенты a, b, c из системы уравнений пункта 3.
5. Подставляем значения x и y и коэффициенты a, b, c в формулу из пункта 1.

Ограничения: при интерполяции полиномом первой степени узлы не должны лежать на одной прямой в плоскости.

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

Пусть нужно провести интерполяцию по переменным x, y и z, и написана функция, выполняющая интерполяцию по переменным x и y. Степени по координатам — n_x, n_y, n_z.

1) $n_z + 1$ раз проводим интерполяцию по переменным x и y, используя написанную функцию: фиксируя значение z_k, найдем значение $p(x, y, z_k)$, где $k = 0, 1, \ldots, n_z$.
2) 1 раз интерполируем по переменной z по полученным значениям $p(x, y, z_k)$, привязанным к z_k, где $k = 0, 1, \ldots, n_z$.

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

При последовательной интерполяции по разным направлениям можно использовать полиномы несовпадающих степеней, алгоритм интерполяции при этом не изменяется. Отличается только начальная таблица: у совпадающих степеней число столбцов и строк совпадает, а у несовпадающих — нет.

При последовательной интерполяции по разным направлениям нельзя использовать разные методы одномерной интерполяции, так как алгоритмы интерполяции различны.

6. Опишите алгоритм двумерной интерполяции на треугольной конфигурации узлов.

1. Определяем число узлов. При двумерной интерполяции на треугольной конфигурации степень полинома n — минимальная из степеней по каждой координате. Число узлов, определяющих многочлен n-ой степени, равно $(n + 1)(n + 2) / 2$.
2. Вводим разделенные разности функции двух переменных:
$z(x_0, x_1, y) = (z(x_0, y) — z(x_1, y)) / (x_0 — x_1)$ — по x,
$z(x, y_0, y_1) = (z(x, y_0) — z(x, y_1)) / (y_0 — y_1)$ — по y

3. Строим полином n-ой степени по формуле

$$P_n(x, y) = \sum_{i=0}^{n} \sum_{j=0}^{n-1} z(x_0, \ldots, x_i, y_0, \ldots, y_j) \prod_{p=0}^{i-1} (x - x_p) \prod_{q=0}^{j-1} (y - y_q)$$