|  |  |
| --- | --- |
| **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 №2**

**тема:** «Построение и программная реализация алгоритма многомерной интерполяции табличных функций»

Выполнил студент: \_\_*Клименко Алексей Константинович\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*фамилия, имя, отчество*

Группа: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*ИУ7-45Б*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил, к.п.н.: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*подпись, дата*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2020 г.*

**Цель работы**

Получение навыков построения алгоритма интерполяции таблично заданных функций двух переменных.

**Исходные данные**

1. Таблица функции с количеством узлов 5х5.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 - nx и ny.

3. Значение агрументов x, y, для которых выполняется интерполяция.

**Код программы**

**import sys**

**# загрузка данных из файла**

**filename = sys.argv[1]**

**X, Y, Z = [], [], {}**

**with open(filename, "rt") as file:**

**X = list(map(float, file.readline().split()[1:]))**

**for line in file:**

**y, \*vals = list(map(float, line.split()))**

**Y.append(y)**

**for x, val in zip(X, vals):**

**Z[(x, y)] = val**

**# линейный сплайн**

**def calc\_lin\_spline\_data(X, Y):**

**A = Y[:-1]**

**B = [(Y[i] - Y[i - 1]) / (X[i] - X[i - 1]) for i in range(1, len(X))]**

**return A, B**

**# квадратичный сплайн**

**def calc\_quad\_spline\_data(X, Y, slope0=0):**

**n = len(X)**

**A = Y[:-1]**

**B, C = [0] \* (n - 1), [0] \* (n - 1)**

**B[0] = slope0**

**C[0] = ((Y[1] - Y[0]) / (X[1] - X[0]) - B[0]) / (X[1] - X[0])**

**for i in range(1, n - 1):**

**B[i] = B[i - 1] + 2 \* C[i - 1] \* (X[i] - X[i - 1])**

**C[i] = ((Y[i + 1] - Y[i]) / (X[i + 1] - X[i]) - B[i]) / (X[i + 1] - X[i])**

**return A, B, C**

**# кубический сплайн**

**def calc\_qubic\_spline\_data(X, Y):**

**n = len(X)**

**A = Y[:-1]**

**ksi, eta = [0, 0], [0, 0]**

**for i in range(2, n): # прямой проход**

**hi, him1 = X[i] - X[i - 1], X[i - 1] - X[i - 2]**

**fi = 3 \* ((Y[i] - Y[i - 1]) / hi - (Y[i - 1] - Y[i - 2]) / him1)**

**ksi.append(- hi / (him1 \* ksi[i - 1] + 2 \* (him1 + hi)))**

**eta.append((fi - him1 \* eta[i - 1]) / (him1 \* ksi[i - 1] + 2 \* (him1 + hi)))**

**C = [0] \* (n - 1)**

**C[n - 2] = eta[-1]**

**for i in range(n - 2, 0, -1): # обратный проход**

**C[i - 1] = ksi[i] \* C[i] + eta[i]**

**B, D = [], []**

**for i in range(1, n - 1):**

**hi = X[i] - X[i - 1]**

**B.append((Y[i] - Y[i - 1]) / hi - hi / 3 \* (C[i] + 2 \* C[i - 1]))**

**D.append((C[i] - C[i - 1]) / 3 / hi)**

**B.append((Y[-1] - Y[-2]) / (X[-1] - X[-2]) - (X[-1] - X[-2]) / 3 \* 2 \* C[-1])**

**D.append(- C[n - 2] / 3 / (X[-1] - X[-2]))**

**return A, B, C, D**

**# использование коэф.-тов интерполяции для нахождения**

**# интерполированного значения табличной функции**

**def apply\_interp\_data(X, data, x):**

**i = max([0] + list(filter(lambda i: X[i] < x, range(len(X)))))**

**y, h = 0, x - X[i]**

**for k, row in enumerate(data):**

**y += row[i] \* h \*\* k**

**return y**

**# интерполяция по табличной функции одной**

**# переменной с заданной степенью полиномов**

**def interpolate(X, Y, x, n):**

**if   n == 1: data = calc\_lin\_spline\_data(X, Y)**

**elif n == 2: data = calc\_quad\_spline\_data(X, Y)**

**elif n == 3: data = calc\_qubic\_spline\_data(X, Y)**

**else: raise RuntimeError("invalid param 'n'. Must be 1, 2 or 3")**

**return apply\_interp\_data(X, data, x)**

**# интерполирования исходной табличной функции**

**# двух переменных с заданными степенями полиномов**

**def z\_func(x, y, n\_x, n\_y):**

**ZX = [interpolate(X, [Z[(xi, yi)] for yi in Y], x, n\_x) for xi in X]**

**return interpolate(Y, ZX, y, n\_y)**

**# начальные параметры интерполирования**

**x, y = 1.5, 1.5**

**print(" n |  x  |  y  | z real | z interp ")**

**print("===|=====|=====|========|==========")**

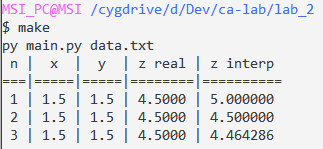
**for n in [1, 2, 3]:**

**z = z\_func(x, y, n, n)**

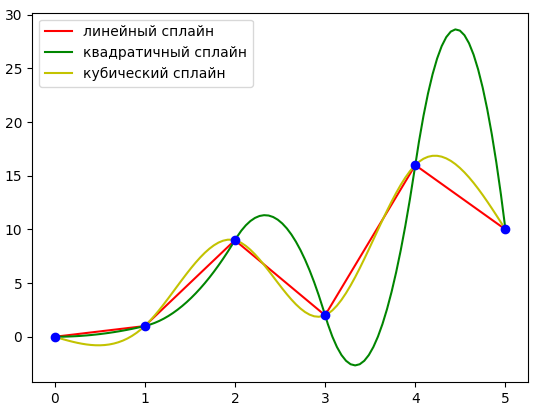
**print(f" {n} | {x} | {y} | {x \*\* 2 + y \*\* 2:.4f} | {z:.6f}")**

**Результаты работы**

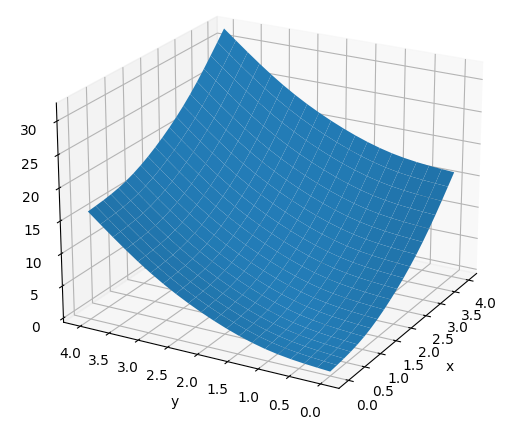
1. Вывод работы программы для x = 1.5 и y = 1.5 для n = 1, 2, 3:

 Как видим, интерполяция при степени полиномов n = 2 наиболее точно позволяет описать данную функцию. Можно ещё раз убедиться в том, что выбор степени интерполяционных полиномов зависит от задачи, и **не всегда больше значит лучше**.

2. Графический вывод интерполированной одномерной функции при разных значениях степени интерполяционных полиномов:



3. Графический вывод интерполяции кубическими сплайнами двумерной функции:



**Контрольные вопросы**

1. *Пусть производящая функция таблицы суть z(x, y) = x2 + y2. Область определения по x и y 0-5 и 0-5. Шаги по переменным равны 1. Степени: nx = 1, ny = 1, переменные: x = 1.5, y = 1.5. Приведите по шагам те значения функции, которые получаются в ходе последовательных интерполяций. По строкам и столбцу.*

Запишем таблицу функции z(x, y):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |

Проведём серию интерполяций по строкам для нахождения значений z(x, y) при x = 1.5:

Теперь интерполируем полученные значения для нахождения значения функциии при y = 1.5:

Получили итоговое значение z(x, y) = **5**

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

Граничные условия для двумерного полинома, опирающегося на 4 узла будут составлять 4 уравнения вида . Рассмотрим двумерный полином первой степени:

Как видим, в таком полиноме всего 3 неизвестных параметра, в то время как ограничивающих уравнений 4. Поэтому минимальная степень двумерного полинома, построенного на 4 узлах **не может быть меньше двух**. (Функция билинейной интерполяции)

Для случая интерполяции по шести узлам, достаточно рассмотреть полином 2 степени с 6 неизвестными коэффициентами. Их можно однозначно определить используя 6 заданных узлов, поэтому минимальная степень интерполяционного полинома на шести узлах будет 2.

3. *Предложите алгоритм двумерной интерполяции при хаотическом 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?*

Можно разбить множество узлов на группы по 3 (разбиение на треугольники). Для каждого такого треугольника определить на его вершинах углы наклона функции (например, используя соседние ближайшие вершины). Таким образом на каждом треугольнике мы будем иметь 9 параметров - значения функции и её первых производных в вершинах треугольника. По этим параметрам можно построить аппроксимирующий функцию полином 2 степени вида:

Основная сложность этого метода состоит в оптимальном разбиении множества узлов на треугольники, ведь главное ограничение данного метода состоит в том, что размерность линейного пространства на множестве узлов, используемых при интерполяции не должна быть меньше размерности исходного пространства. Другими словами, при интерполяции двумерной функции узлы не должны лежать на одной прямой.