

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** Зайцева А. А.

**Группа** ИУ7 – 42Б

**Оценка (баллы) \_\_\_\_\_\_\_**

**Преподаватель** Градов В. М.

Москва.

2021 г

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

1. Исходные данные
2. Таблица функции c количеством узлов 5x5

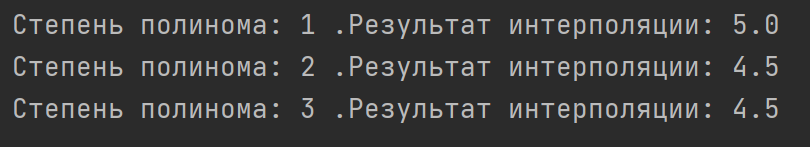
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X, Y | 0 | 1 | 2 | 3 | 4 |
| 0 | 0 | 1 | 4 | 9 | 16 |
| 1 | 1 | 2 | 5 | 10 | 17 |
| 2 | 4 | 5 | 8 | 13 | 20 |
| 3 | 9 | 10 | 13 | 18 | 25 |
| 4 | 16 | 17 | 20 | 25 | 32 |

1. Степень аппроксимирующих полиномов – nx и ny.
2. Значение аргументов x, y, для которого выполняется интерполяция.
3. Код программы

from copy import deepcopy  
  
  
# Выбор узлов из данной таблицы nodes значений функций при значениях  
# x в интервале x\_values и y в интервале y\_values для построения  
# аппроксимирующего полинома степени n в точке (x0, y0)  
# Возвращает таблицу выбранных узлов и соотвтествующие значения x и y  
def choose\_nodes(nodes, x\_values, y\_values, n, x0, y0):  
   # необходимо выбрать n + 1 узлов  
   need\_to\_take = n + 1  
   if need\_to\_take > len(x\_values):  
       print('ОШИБКА: не хватает узлов')  
   # находим узел, ближайший к значениям x0, y0  
   x\_closest\_i = (sorted(range(len(x\_values) - 1, -1, -1), key=lambda i: abs(x\_values[i] - x0)))[0]  
   y\_closest\_j = (sorted(range(len(y\_values) - 1, -1, -1),  
                         key=lambda j: abs(y\_values[j] - y0)))[0]  
   # определяем индексы необходимых узлов в исходной таблице  
   # если не удается равномерно распределить узлы вокруг точки, выбираем из  
   # того, что есть  
   from\_i = x\_closest\_i - need\_to\_take // 2  
   if from\_i < 0:  
       from\_i = 0  
   to\_i = from\_i + need\_to\_take  
   if to\_i > len(x\_values):  
       to\_i = len(x\_values)  
       from\_i = to\_i - need\_to\_take  
   from\_j = y\_closest\_j - need\_to\_take // 2  
   if from\_j < 0:  
       from\_j = 0  
   to\_j = from\_j + need\_to\_take  
   if to\_j > len(y\_values):  
       to\_j = len(y\_values)  
       from\_j = to\_j - need\_to\_take  
  
   # формируем таблицу из выбранных узлов  
   x\_chosen = x\_values[from\_i:to\_i]  
   y\_chosen = y\_values[from\_j:to\_j]  
   chosen\_nodes = deepcopy(nodes)  
   chosen\_nodes = [row[from\_j:to\_j] for row in chosen\_nodes[from\_i:to\_i]]  
  
   return chosen\_nodes, x\_chosen, y\_chosen  
  
  
# Нахождение коэффициентов для одномерной аппроксимирующего полинома степени n  
# по значениям функции func\_values от значений переменной x\_values  
# Возвращает найденные коэффициенты  
def find\_coeffs(func\_values, x\_values, n):  
   func\_values = deepcopy(func\_values)  
   coeffs = [func\_values[0]]  
   # step - шаг (номер столбца в таблице после y)  
   for step in range(n):  
       # i - строка столбца  
       for i in range(n - step):  
           # вычисление разделенной разности для y от (step + 2) переменных  
           func\_values[i] = ((func\_values[i + 1] - func\_values[i]) /  
                             (x\_values[i + step + 1] - x\_values[i]))  
       coeffs.append(func\_values[0])  # собираем значения из верхней строчки таблицы  
   return coeffs  
  
  
# Вычисление полинома степени n с коэффициентами coeffs  
# по массиву переменной x в точке x0  
# Возвращает значение полинома в точке x0  
def count\_polynom(x\_values, coeffs, n, x0):  
   summ = 0  
   # вычисление очередного слагаемого  
   for stage in range(n + 1):  
       summand = coeffs[stage]  
       for i in range(stage):  
           summand \*= (x0 - x\_values[i])  
       # Формирование ответа  
       summ += summand  
   return summ  
  
# Интерполяция по x\_values: находит f(x, yj), при  j = 0,1,..n  
# Возвращает массив вычисленных f(x, yj), при  j = 0,1,..n  
def interpolate\_by\_x(func\_table, x\_values, n, x0):  
   func\_arr = []  
   # выполняем одномерную интерполяцию при выбранном значении yj  
   for yj in range(n + 1):  
       coeffs = find\_coeffs([row[yj] for row in func\_table], x\_values, n)  
       func\_cur = count\_polynom(x\_values, coeffs, n, x0)  
       func\_arr.append(func\_cur)  
   return func\_arr  
  
# Интерполяция по y\_values: по полученным значениям функции, привязанным к yj,  
# интерполирует по y в точке y0  
# Возвращает приближенное значение функции в точке (x0, y0)  
def interpolate\_by\_y(func\_array, y\_values, n, y0):  
   coeffs = find\_coeffs(func\_array, y\_values, n)  
   result = count\_polynom(y\_values, coeffs, n, y0)  
   return result  
  
def main():  
   # исходные данные  
   x\_values = [0, 1, 2, 3, 4]  
   y\_values = [0, 1, 2, 3, 4]  
   nodes = [[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]]  
   x0 = 1.5  
   y0 = 1.5  
   n\_range = [1, 2, 3]  
   # выполнение задания  
   answer\_table = []  
   # для степени полинома n  
   for n in n\_range:  
       # выбираем необходимые узлы для интерполяции в точке (x0, y0)  
       chosen\_nodes, x\_chosen, y\_chosen = choose\_nodes(nodes, x\_values, y\_values, n, x0, y0)  
       # интерполируем по x  
       func\_array = interpolate\_by\_x(chosen\_nodes, x\_chosen, n, x0)  
       # интерполируем по y  
       answer = interpolate\_by\_y(func\_array, y\_chosen, n, y0)  
       # заносим ответ в таблицу  
       answer\_table.append([n, answer])  
   # выводим результаты  
   for answer in answer\_table:  
       print('Степень полинома:', answer[0], '.Результат интерполяции:', answer[1])  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
   main()

1. Результаты работы

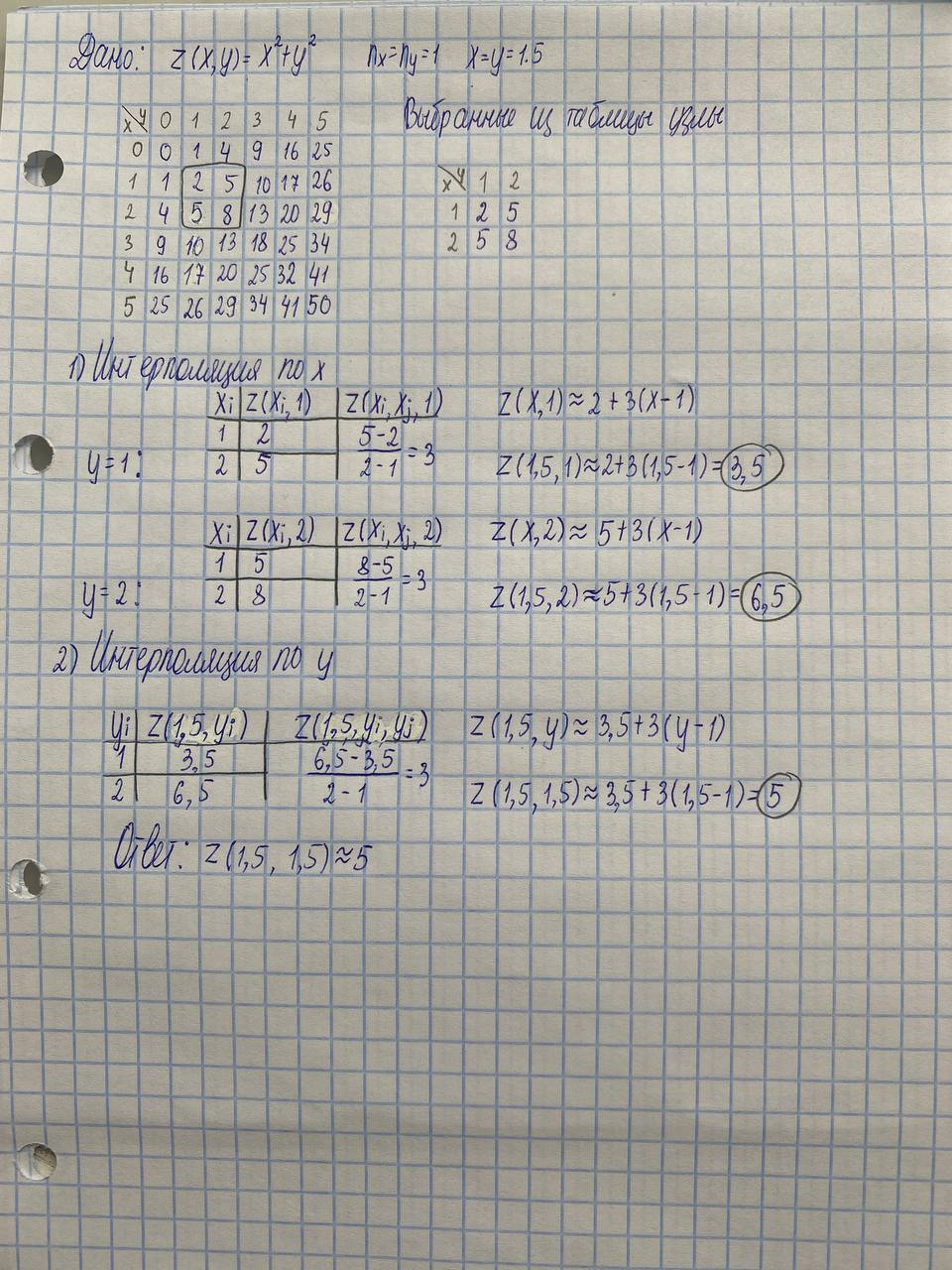
Результат интерполяции z(x,y) при степенях полиномов 1,2,3 для x=1.5, y=1.5 .



1. Вопросы при защите лабораторной работы

Ответы на вопросы дать письменно в Отчете о лабораторной работе.

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 nx = ny =1, x=y=1.5. Приведите по шагам те значения функции, которые получаются в ходе последовательных интерполяций по строкам и столбцу.



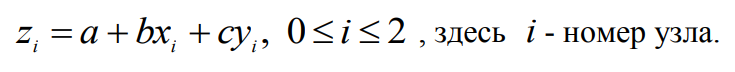
1. Какова минимальная степень двумерного полинома, построенного на четырех узлах? На шести узлах?

На 4 узлах можно построить полиномы с nx=ny принадлежат [0, 3]

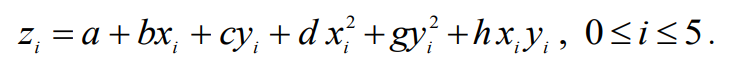
На 6 узлах можно построить полиномы с nx=ny принадлежат [0, 5]

1. 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 хаотичном расположении узлов, ограничиваясь интерполяционным полиномом первой степени, имеем z = a + bx + cy , и его коэффициенты находят по трем узлам, выбираемым в окрестности точки интерполяции:



Точно так же можно использовать полином второй степени



Ограничения на расположение узлов: Например, при интерполяции полиномом первой степени P1(x, y) узлы не должны лежать на одной прямой в плоскости, при интерполяции полиномом второй степени P2(x, y) узлы не должны лежать на одной плоскости в пространстве и т.д.

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

Используя алгоритм двумерной интерполяции. Вначале провести интерполяцию, например, по x. При этом выполнить nyz + 1 одномерных интерполяций при выбранных значениях yz[j], j = 0,1,...n , и вычислить значения функции

f(x, yz[j]), j = 0,1,...n . А затем по полученным значениям функции, привязанным теперь к yz[j], провести одну интерполяция по yz.

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

Да, так как интерполяции по разным направлениям проводятся фактически независимо.

1. Опишите алгоритм двумерной интерполяции на треугольной конфигурации узлов.

При треугольной конфигурации расположения узлов степень многочлена будет минимальной. Многочлен n-й степени в форме Ньютона для двумерной интерполяции в этом случае можно представить, как обобщение одномерного варианта записи:

