|  |  |
| --- | --- |
| 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.

**Студент:** Солнцева Т. В.

**Группа:** ИУ7-45Б

**Оценка (баллы):** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Преподаватель:** Градов В.М.

*Москва*

*2020 г*

**Цель работы:** Получение навыков построения алгоритма интерполяции

таблично заданных функций двух переменных.

**Исходные данные**

1. 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 | 2 | 5 | 8 | 13 | 20 |
| 3 | 4 | 10 | 13 | 18 | 25 |
| 4 | 16 | 17 | 20 | 25 | 32 |

2. Степень аппроксимирующих полиномов - nx и ny .

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

**Идея реализации:**

Для выполнения задания применен алгоритм линейной интерполяции полинома Ньютона из предыдущей лабораторной работы.

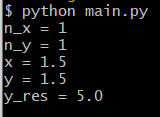
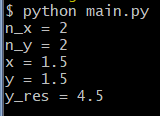
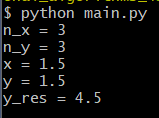
1. Интерполяция для z(x) и х, результат сохраняется в массив.
2. Интерполяция для у и полученного массива. Полученный результат является искомым приближением.

**Код программы**

|  |
| --- |
| def read\_table():      """          Чтение таблицы из файла "test.txt", также оттуда считываются n и х      """      x\_tbl = []      y\_tbl = []      z\_tbl = []      f = open('test.txt')      txt\_table = f.read()      txt\_table = txt\_table.replace(",", ".")      txt\_table = txt\_table.split("\n")      tbl\_len = len(txt\_table)      if txt\_table[tbl\_len - 1] == '':          txt\_table.pop(tbl\_len - 1)          tbl\_len -= 1      x\_tbl.extend(txt\_table[0].split())      for i in range(len(x\_tbl)):          x\_tbl[i] = float(x\_tbl[i])      for i in range(1, tbl\_len - 2):          tbl\_str = txt\_table[i].split()          y\_tbl.append(float(tbl\_str[0]))          z\_tbl.append([])          for j in range(1, len(tbl\_str)):              z\_tbl[i - 1].append(float(tbl\_str[j]))      tbl\_str = txt\_table[tbl\_len - 2].split()      n\_x, n\_y = float(tbl\_str[0]), float(tbl\_str[1])      tbl\_str = txt\_table[tbl\_len - 1].split()      x, y  = float(tbl\_str[0]), float(tbl\_str[1])      return x\_tbl, y\_tbl, z\_tbl, n\_x, n\_y, x, y  def find\_nearest(x\_tbl, n, x):      """          Возвращение таблиц для n узлов      """      curr\_ind = 0      for i in range(1, len(x\_tbl)):          if (abs(x - x\_tbl[i]) <= abs(x - x\_tbl[i - 1])):              curr\_ind = i      left = n // 2      right = left - (n % 2 + 1) % 2      while (curr\_ind - left < 0):          left -= 1          right += 1      while(curr\_ind + right > len(x\_tbl) - 1):          right -= 1          left += 1      x\_table = x\_tbl[int(curr\_ind - left):int(curr\_ind + right) + 1]      return x\_table  def create\_dif\_tbl(x\_table, y\_table, n):      """          Создание таблицы разделенных разностей      """      dif\_tbl = []      dif\_tbl.append(y\_table)      for i in range(int(n)):          tmp = []          for j in range(int(n) - i):              if (abs(x\_table[j] - x\_table[j + i + 1]) > 1e-6):                  tmp.append((dif\_tbl[i][j] - dif\_tbl[i][j + 1]) / (x\_table[j] - x\_table[j + i + 1]))          dif\_tbl.append(tmp)      return dif\_tbl  def count\_polynomial(n, x, x\_table, dif\_tbl):      """          Вычисление полинома      """      result = dif\_tbl[0][0]      for i in range(1, int(n) + 1):          multiple = 1          for j in range(i):              multiple \*= (x - x\_table[j])          result += (dif\_tbl[i][0] \* multiple)      return result  def interpolate(table\_x, table\_y, n, x):      """          Построение интерполяционного полинома Ньютона      """      dif\_tbl = create\_dif\_tbl(table\_x, table\_y, n)      return count\_polynomial(n, x, table\_x, dif\_tbl)  def interpolate\_multi(x\_tbl, y\_tbl, z\_tbl, n\_x, n\_y, x, y):      """          Многомерная интерполяция      """      x\_table = find\_nearest(x\_tbl, n\_x + 1, x)      y\_table = find\_nearest(y\_tbl, n\_y + 1, y)      start\_ind = y\_tbl.index(y\_table[0])      z\_table = z\_tbl[start\_ind:start\_ind+len(y\_table)]      start\_ind = x\_tbl.index(x\_table[0])      for i in range(int(n\_y) +  1):          z\_table[i] = z\_table[i][start\_ind:start\_ind+len(x\_table)]      x\_first = []      for i in range(int(n\_y) + 1):          x\_first.append(interpolate(x\_table, z\_table[i], n\_x, x))      y\_res = interpolate(y\_table, x\_first, len(x\_first) - 1, y)      return y\_res  def main():      try:          x\_tbl, y\_tbl, z\_tbl, n\_x, n\_y, x, y = read\_table()      except IndexError:          print("Something wrong with the file test.txt")          return      except FileNotFoundError:          print("File not found")          return      y\_res = interpolate\_multi(x\_tbl, y\_tbl, z\_tbl, n\_x, n\_y, x, y)      print(y\_res)  if \_\_name\_\_ == "\_\_main\_\_":      main() |

**Результаты работы**

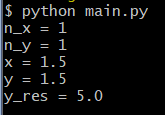
Результат интерполяции z(x,y) при степенях полиномов 1,2,3 для x=1.5, y=1.5 .

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| nx | ny | x | y | Искомое приближение | F(x, y) |
| 1 | 1 | 1.5 | 1.5 | 5.0 | 4.5 |
| 2 | 2 | 1.5 | 1.5 | 4.5 | 4.5 |
| 3 | 3 | 1.5 | 1.5 | 4.5 | 4.5 |

**Контрольные вопросы**

***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.***

**

Массив х: [1, 2]

Массив y: [1, 2]

Массив z: [[2, 5], [5, 8]]

Далее производим ny интерполяций и получаем массив с значениями: [3.5, 6.5]

Далее проводим еще раз операцию интерполяции (теперь для у и полученного массива) и уже получаем значение 5.0, (при подстановке в функцию z = x^2 + y^2 получим значение 4.5 => погрешность = 0.5)

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

Минимальная степень двумерного полинома, построенного на 4 узлах - 0, всего степени могут быть от 0 до 3. На 6 узлах тоже 0, всего степени могут быть от 0 до 5.

***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 хаотичном расположении узлов, ограничиваясь интерполяционным полиномом первой степени, получим . Все коэффициенты находят по трем узлам, которые выбираются в окрестности точки интерполяция 

Ограничения: узлы не могут лежать на одной прямой в плоскости при интерполяции полиномом первой степени P(x, y), а при интерполяции второй степени не должны лежать на одной плоскости в пространстве.

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

1. Выбираем три переменные и их отрезки
2. Пусть есть функция F(x, y, z). Делаем двумерную интерполяцию для переменных (x, y) nz раз, где nz - это степень переменной z. Записываем в массив.
3. Проводим еще одну двумерную интерполяцию для (массив, z). Таким образом находим искомое значение.

***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. Результат данной интерполяции не зависит от порядка шагов, поэтому можно сначала интерполировать вдоль оси ординат, а затем абсцисс, или наоборот, результат не изменится. Также не зависит от метода интерполяции.

***6. Опишите алгоритм двумерной интерполяции на треугольной конфигурации узлов.***

Алгоритм двумерной интерполяции на треугольной конфигурации узлов будет отличаться от выше использованного алгоритма (многомерной интерполяции) дополнительной проверкой на нужное количество узлов при одномерной интерполяции по оси ординат/оси абсцисс. Если интерполяция не одномерная, то будут использоваться сразу все узлы.