**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 № 4**

**Тема** Построение и программная реализация алгоритма наилучшего среднеквадратичного приближения.

**Студент** Алахов А.Г.

**Группа** ИУ7-42Б

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

**Преподаватель** Градов В.М.

Москва.

2021 г

**Цель работы**. Получение навыков построения алгоритма метода наименьших квадратов с использованием полинома заданной степени при аппроксимации табличных функций с весами.

# Исходные данные

1. Таблица функции с весами ρ i с количеством узлов N. Сформировать таблицу самостоятельно со случайным разбросом точек.

Сформированная таблица:

|  |  |  |
| --- | --- | --- |
| X | Y | Вес |
| 0 | 0 | 1 |
| 1 | -1,5 | 1 |
| 2 | -1 | 1 |
| 3 | 2,5 | 1 |
| 4 | 6 | 1 |
| 5 | 8 | 1 |
| 6 | 7 | 1 |
| 7 | 5 | 1 |
| 8 | 5 | 1 |
| 9 | 7,5 | 1 |
| 10 | 11,5 | 1 |

Предусмотреть в интерфейсе удобную возможность изменения пользователем весов в таблице.

1. Степень аппроксимирующего полинома - n.

# Код программы

Код программы представлен на листингах 1-2.

Листинг 1. functions.py

from math import fabs

import matplotlib.pyplot as plt

def least\_squares\_method(table, n):

n += 1

matr = [[0] \* (n + 1) for i in range(n)]

coefs = []

for i in range(n):

for j in range(i, n):

summ = 0

for k in range(len(table)):

summ += table[k][2] \* table[k][0] \*\* (i + j)

matr[i][j] = matr[j][i] = summ

summ = 0

for k in range(len(table)):

summ += table[k][2] \* table[k][1] \* table[k][0] \*\* i

matr[i][n] = summ

for i in range(n - 1, 0, -1):

tmp = matr[i][i]

for j in range(n + 1):

matr[i][j] /= tmp

for j in range(i):

tmp = matr[j][i]

for k in range(n + 1):

matr[j][k] -= matr[i][k] \* tmp

matr[0][n] /= matr[0][0]

for i in range(n):

summ = matr[i][n]

for j in range(len(coefs)):

summ -= coefs[j] \* matr[i][j]

coefs.append(summ)

dots = []

x = table[0][0]

while x <= table[len(table) - 1][0]:

y = 0

for j in range(len(coefs)):

y += coefs[j] \* x \*\* j

dots.append(y)

x += 0.1

return dots

Листинг 2. main.py

from functions import \*

def main():

func\_table = [[0, 0, 1],

[1, -1.5, 1],

[2, -1, 1],

[3, 2.5, 1],

[4, 6, 1],

[5, 8, 1],

[6, 7, 1],

[7, 5, 1],

[8, 5, 1],

[9, 7.5, 1],

[10, 11.5, 1]]

'''

func\_table = [[0, 0, 4],

[1, -1.5, 3],

[2, -1, 7],

[3, 2.5, 11],

[4, 6, 1],

[5, 8, 5],

[6, 7, 23],

[7, 5, 1],

[8, 5, 2],

[9, 7.5, 1],

[10, 11.5, 99]]

'''

print('Заданная таблица:\n\

| №| X |Y(x)|Вес|')

for i in range(len(func\_table)):

print('|{:2d}|{:3d}|{:4.1f}|{:3d}|'.format(i + 1,

func\_table[i][0],

func\_table[i][1],

func\_table[i][2]))

print('\n1 - Изменить вес точки\n\

2 - Вывести таблицу\n\

3 - Вывести графики для разных степерей полинома\n\

4 - Вывести графики для текущей таблицы и аналогичной, с весом 1 у всех узлов\n\

0 - Выйти из меню\n\

Выбор: ', end = '')

choice = int(input())

while choice:

if choice == 1:

num = int(input('\nВведите номер точки: '))

weight = int(input('\nВведите новый вес точки: '))

func\_table[num - 1][2] = weight

elif choice == 2:

print('Заданная таблица:\n\

| №| X |Y(x)|Вес|')

for i in range(len(func\_table)):

print('|{:2d}|{:3d}|{:4.1f}|{:3d}|'.format(i + 1,

func\_table[i][0],

func\_table[i][1],

func\_table[i][2]))

elif choice == 3:

graph(func\_table)

else:

graph\_compare(func\_table)

print('\n1 - Изменить вес точки\n\

2 - Вывести таблицу\n\

3 - Вывести графики для разных степерей полинома\n\

4 - Вывести графики для текущей таблицы и аналогичной, с весом p=1 для всех узлов\n\

0 - Выйти из меню\n\

Выбор: ', end = '')

choice = int(input())

def graph(func\_table):

dots = [[]]

x = func\_table[0][0]

while x <= func\_table[len(func\_table) - 1][0]:

dots[0].append(x)

x += 0.1

dots.append(least\_squares\_method(func\_table, 1))

dots.append(least\_squares\_method(func\_table, 2))

dots.append(least\_squares\_method(func\_table, 4))

dots.append(least\_squares\_method(func\_table, 6))

funcs = ['p = 1',

'p = 2',

'p = 4',

'p = 6',

'Данные']

fig, ax = plt.subplots()

ax.plot(dots[0], dots[1],

dots[0], dots[2],

dots[0], dots[3],

dots[0], dots[4])

ax.scatter([i[0] for i in func\_table], [i[1] for i in func\_table],

c = 'black')

plt.legend(funcs, loc=2)

plt.grid()

ax.set\_ylabel('Y')

ax.set\_xlabel('X')

plt.show()

def graph\_compare(func\_table):

dots = [[]]

x = func\_table[0][0]

while x <= func\_table[len(func\_table) - 1][0]:

dots[0].append(x)

x += 0.1

dots.append(least\_squares\_method(func\_table, 1))

dots.append(least\_squares\_method(func\_table, 2))

for i in range(len(func\_table)):

func\_table[i][2] = 1

dots.append(least\_squares\_method(func\_table, 1))

dots.append(least\_squares\_method(func\_table, 2))

funcs = ['p = 1 (вес точек различный)',

'p = 2 (вес точек различный)',

'p = 1 (вес точек равен 1)',

'p = 2 (вес точек равен 1)',

'Данные']

fig, ax = plt.subplots()

ax.plot(dots[0], dots[1],

dots[0], dots[2],

dots[0], dots[3],

dots[0], dots[4])

ax.scatter([i[0] for i in func\_table], [i[1] for i in func\_table],

c = 'black')

plt.legend(funcs, loc=2)

plt.grid()

ax.set\_ylabel('Y')

ax.set\_xlabel('X')

plt.show()

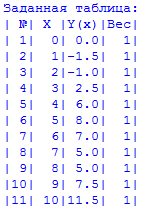
if \_\_name\_\_ == "\_\_main\_\_":

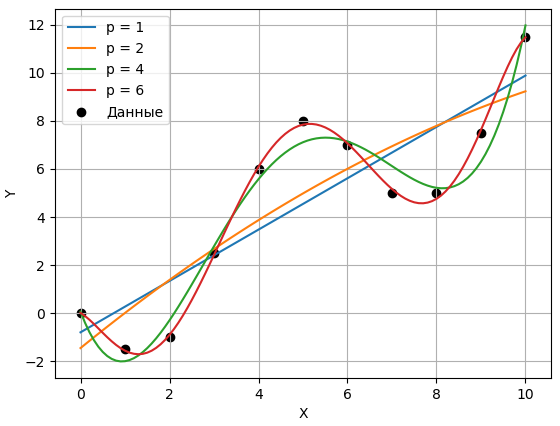
main()

# Результаты работы

Графики, построенные по аналогии с рис.1 в тексте Лекции №4: точки - заданная табличная функция, кривые - найденные полиномы. Обязательно приводить таблицы, по которым работала программа.

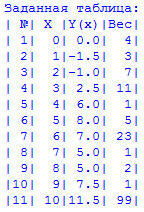
1. Веса всех точек одинаковы и равны, например, единице. Обязательно построить полиномы степеней n=1 и 2. Можно привести результаты и при других степенях полинома, однако, не загромождая сильно при этом рисунок.

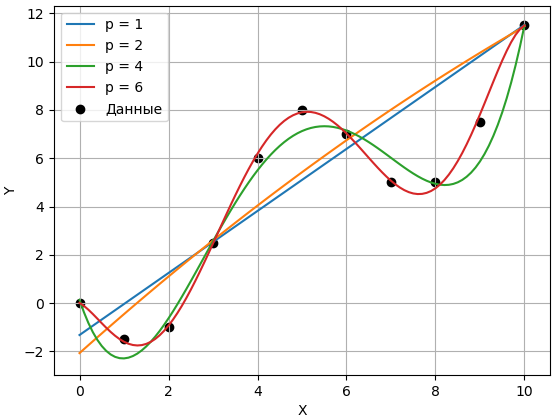




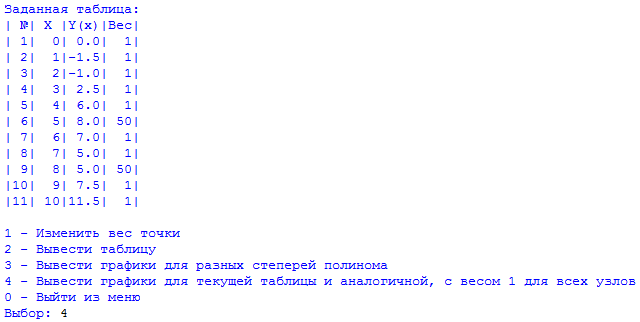
2. Веса точек разные. Продемонстрировать, как за счет назначения весов точкам можно изменить положение на плоскости прямой линии (полином первой степени), аппроксимирующей один и тот же набор точек (одну таблицу y(x)). Например, назначая веса узлам в таблице изменить знак углового коэффициента прямой. На графике в итоге должны быть представлены точки исходной функции и две аппроксимирующие их прямые линии. Одна отвечает значениям ρ i =1 для всех узлов, а другая- назначенным разным весам точек. Информацию о том, какие именно веса были использованы в расчете обязательно указать, чтобы можно было проконтролировать работу программы (лучше это сделать в виде таблицы).

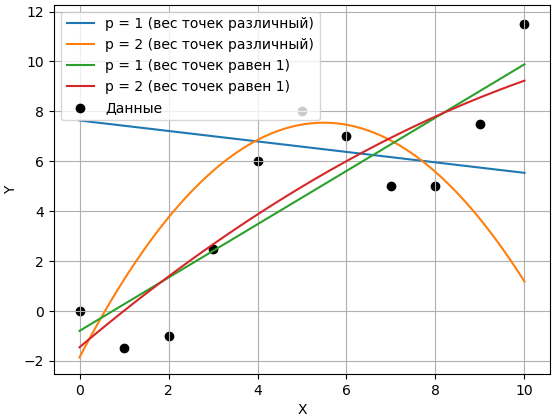
Графики различных степеней полинома при разных весах точек:





Изменение знака углового коэффициента прямой с помощью изменения веса:





# Вопросы при защите лабораторной работы

1. Что произойдет при задании степени полинома n=N-1 (числу узлов таблицы минус 1)?

График полинома будет проходить через все точки, независимо от их веса.

2. Будет ли работать Ваша программа при n ≥ N ? Что именно в алгоритме требует отдельного анализа данного случая и может привести к аварийной остановке?

Программа будет работать, но результат работы программы будет некорректным. Аварийная ситуация может возникнуть из-за деления на ноль в процессе решения системы уравнений, т.к. система будет линейно зависимой. В данной программе аварийная ситуация не возникает из-за погрешности при работе с действительными числами.

3. Получить формулу для коэффициента полинома a0 при степени полинома n=0. Какой смысл имеет величина, которую представляет данный коэффициент?

Данный коэффициент будет являться взвешенным средним арифметическим ординат функции.

4. Записать и вычислить определитель матрицы СЛАУ для нахождения коэффициентов полинома для случая, когда n=N=2. Принять все ρ i =1.

Определитель матрицы:

Так как определитель матрицы равен нулю, система не имеет решений.

5. Построить СЛАУ при выборочном задании степеней аргумента полинома , причем степени n и m в этой формуле известны.

6. Предложить схему алгоритма решения задачи из вопроса 5, если степени n и m подлежат определению наравне с коэффициентами ak , т.е. количество неизвестных равно.

Подобную систему можно решить методом перебора всех возможных значений n и m. Для каждой пары значений находят коэффициенты, а также ошибку. В качестве конечного результата выбирается пара с минимальной ошибкой.