**Национальный исследовательский университет ИТМО**

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Вычислительная математика»

**Отчет**

По лабораторной работе №4

Вариант 9

Выполнил:

*Кочнев Р. Д.*

*P32081*

Преподаватель:

*Рыбаков С. Д.*

Санкт-Петербург, 2023 г.

Цель работы

Цель работы

Найти функцию, являющуюся наилучшим приближением заданной табличной функции по методу наименьших квадратов.

Порядок выполнения

1. Изучить способы аппроксимации
2. Найти интеграл вручную
3. В Excel построить линейную и квадратичную аппроксимацию
4. Сделать красиво
5. Программно реализовать линейную функцию, полиномиальную функцию 2-й степени, полиномиальную функцию 3-й степени, экспоненциальную функцию, логарифмическую функцию, степенную функцию.
6. Проверить правильность решения

Исходная функция



Коэффициенты для линейной аппроксимации

|  |  |
| --- | --- |
| a | b |
| 0,175 | 0,119 |

Коэффициенты для квадратичной аппроксимации

|  |  |  |
| --- | --- | --- |
| a0 | a1 | a2 |
| -0,239 | 0,653 | -0,024 |

Таблица исходных данных и аппроксимирующих

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| X | 0,0000 | 0,2000 | 0,4000 | 0,6000 | 0,8000 | 1,0000 | 1,2000 | 1,4000 | 1,6000 | 1,8000 | 2,0000 |
| Y | 0,0000 | 0,0889 | 0,1773 | 0,2629 | 0,3401 | 0,4000 | 0,4335 | 0,4361 | 0,4115 | 0,3693 | 0,3200 |
| P(X) | 0,1190 | 0,1540 | 0,1890 | 0,2240 | 0,2590 | 0,2940 | 0,3290 | 0,3640 | 0,3990 | 0,4340 | 0,4690 |
| poly\_2 | -0,0240 | 0,0970 | 0,1990 | 0,2818 | 0,3454 | 0,3900 | 0,4154 | 0,4218 | 0,4090 | 0,3770 | 0,3260 |

Среднеквадратичное отклонение

|  |  |
| --- | --- |
| eps(lin) | 0,5492901113 |
| eps(poly\_2) | 0,08673190706 |

Пример выполнения программы

Linear - RMSE: 0.086 - Coefficients: [0.17568182 0.11859091]

Polynomial (2nd degree) - RMSE: 0.014 - Coefficients: [-0.23884033 0.65336247 -0.02471329]

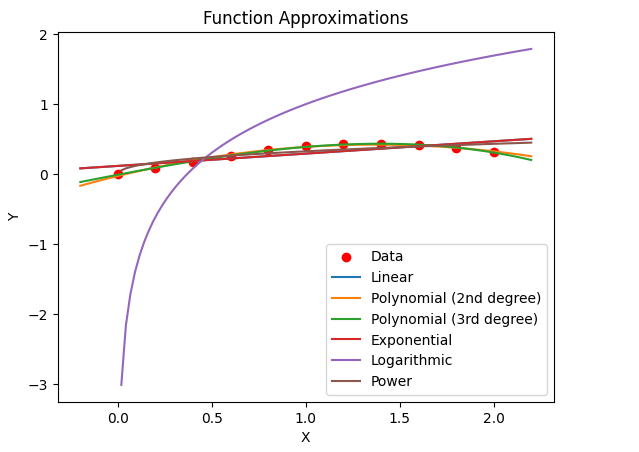
Polynomial (3rd degree) - RMSE: 0.008 - Coefficients: [-0.06106741 -0.0556381 0.51364024 -0.00712587]

Exponential - RMSE: 0.086 - Coefficients: [ 1.36077323e+03 1.29079253e-04 -1.36065462e+03]

Logarithmic - RMSE: inf - Coefficients: [1. 1. 1.]

Power - RMSE: 0.061 - Coefficients: [ 0.34495966 0.38847804 -0.01780568]

Best fit: poly\_3



Код программы

<https://github.com/poma12390/2course/tree/master/witch_math/lab4>

import numpy as np

import matplotlib.pyplot as plt

from scipy.optimize import curve\_fit

def linear\_func(x, a, b):

    return a \* x + b

def poly\_2(x, a, b, c):

    return a \* x\*\*2 + b \* x + c

def poly\_3(x, a, b, c, d):

    return a \* x\*\*3 + b \* x\*\*2 + c \* x + d

def exp\_func(x, a, b, c):

    return a \* np.exp(b \* x) + c

def log\_func(x, a, b, c):

    return a \* np.log(b \* x) + c

def power\_func(x, a, b, c):

    return a \* x\*\*b + c

functions = [linear\_func, poly\_2, poly\_3, exp\_func, log\_func, power\_func]

function\_names = ['Linear', 'Polynomial (2nd degree)', 'Polynomial (3rd degree)', 'Exponential', 'Logarithmic', 'Power']

# Ввод данных

points = np.array([

    [0, 0],

    [0.2, 0.088],

    [0.4, 0.177],

    [0.6, 0.263],

    [0.8, 0.34],

    [1, 0.4],

    [1.2, 0.433],

    [1.4, 0.436],

    [1.6, 0.411],

    [1.8, 0.369],

    [2, 0.32]

])

x = points[:, 0]

y = points[:, 1]

best\_fit\_func = None

best\_fit\_params = None

min\_rmse = float('inf')

# Подгонка функций и выбор наилучшей

for func, func\_name in zip(functions, function\_names):

    try:

        params, \_ = curve\_fit(func, x, y)

        y\_pred = func(x, \*params)

        rmse = np.sqrt(np.mean((y - y\_pred)\*\*2))

        if rmse < min\_rmse:

            min\_rmse = rmse

            best\_fit\_func = func

            best\_fit\_params = params

        print(f'{func\_name} - RMSE: {rmse:.3f} - Coefficients: {params}')

    except RuntimeError:

        print(f'{func\_name} - Failed to fit')

print(f'\nBest fit: {best\_fit\_func.\_\_name\_\_}')

# Построение графиков

x\_range = np.linspace(x.min() - 0.2, x.max() + 0.2, 100)

plt.scatter(x, y, label='Data', color='red')

for func, func\_name in zip(functions, function\_names):

    try:

        params, \_ = curve\_fit(func, x, y)

        plt.plot(x\_range, func(x\_range, \*params), label=func\_name)

    except RuntimeError:

        pass

plt.legend()

plt.xlabel('X')

plt.ylabel('Y')

plt.title('Function Approximations')

plt.show()

Вывод

Познакомился с аппроксимацией функций, узнал про способы аппроксимации, написал программу для аппроксимации 6 различными способами.