Федеральное государственное автономное образовательное учреждение высшего образования **«Национальный исследовательский университет ИТМО»**

Факультет Программной Инженерии и Компьютерной Техники

**Лабораторная работа по дисциплине «Вычислительная математика» №4**

Вариант: 4

Преподаватель:   
Рыбаков Степан Дмитриевич

Выполнил: Васильченко Роман

Группа: Р32081

Санкт-Петербург, 2023г

# Цель работы

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

# Листинг программы

# import numpy as np

# import matplotlib.pyplot as plt

# from scipy.optimize import curve\_fit

# from sklearn.metrics import mean\_squared\_error

# from scipy.stats import pearsonr

# import sys

# 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 exponential\_func(x, a, b):

# return a \* np.exp(b \* x)

# def logarithmic\_func(x, a, b):

# return a \* np.log(x) + b

# def power\_func(x, a, b):

# return a \* x\*\*b

# def compute\_pearson\_correlation(x, y, params):

# y\_pred = linear\_func(x, \*params)

# return pearsonr(y, y\_pred)

# def fit\_and\_plot(func, name, x, y):

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

# plt.plot(x, func(x, \*params), label=name)

# return params, mean\_squared\_error(y, func(x, \*params))

# def read\_data\_from\_file(filename):

# with open(filename, 'r') as file:

# x = []

# y = []

# for line in file:

# point = line.strip().split()

# if len(point) == 2:

# x.append(float(point[0]))

# y.append(float(point[1]))

# return np.array(x), np.array(y)

# def read\_data\_from\_input():

# str = ''

# x = []

# y = []

# while str != 'quit':

# str = input()

# point = str.strip().split()

# if len(point) == 2:

# x.append(float(point[0]))

# y.append(float(point[1]))

# return np.array(x), np.array(y)

# def main():

# while True:

# option = input("Напишите 'файл' или 'ввод' ")

# if option == 'файл':

# filename = 'data.txt'

# x, y = read\_data\_from\_file(filename)

# break

# elif option == 'ввод':

# print('Введите quit, чтобы закончить ввод')

# x,y = read\_data\_from\_input()

# break

# else:

# print("Некорректный ввод. Попробуйте еще раз")

# # Функции для исследования

# functions = [

# (linear\_func, "Линейная"),

# (poly\_2, "Полиноминальная 2-й степени"),

# (poly\_3, "Полиноминальная 3-й степени"),

# (exponential\_func, "Экспоненциальная"),

# (logarithmic\_func, "Логарифмическая"),

# (power\_func, "Степенная")

# ]

# best\_func = None

# best\_mse = float("inf")

# with open('output.txt', 'w') as output:

# option = input("Вывод в 'файл' или в 'терминал'? ")

# if option == 'файл':

# sys.stdout = output

# else:

# print("Выбран вариант вывода в терминал")

# 

# for func, name in functions:

# try:

# params, mse = fit\_and\_plot(func, name, x, y)

# print(f"{name} функция:")

# print(f" Коэффициенты: {params}")

# print(f" Среднеквадратические отклонения: {mse}\n")

# if func == linear\_func:

# correlation, \_ = compute\_pearson\_correlation(x, y, params)

# print(f"Коэффициент корреляции Пирсона: {correlation}\n")

# if mse < best\_mse:

# best\_mse = mse

# best\_func = name

# except Exception as e:

# print(f"Error fitting {name} function: {e}\n")

# 

# print(f"Лучшая функция приближения: {best\_func}")

# plt.scatter(x, y, label="Вводные точки")

# plt.legend()

# plt.xlabel("x")

# plt.ylabel("y")

# plt.title("Приближение функции различными методами")

# plt.show()

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

# main()

# Результаты выполнения программы

# Входные данные:

# -4 -0.2307

# -3.6 -0.3140

# -3.2 -0.4409

# -2.8 -0.6415

# -2.4 -0.9683

# -2 -1.5000

# -1.6 -2.2741

# -1.2 -2.9636

# -0.8 -2.7213

# -0.4 -1.4905

# 0 0.0000

# Выходные данные:

# Линейная функция:

# Коэффициенты: [-0.37144773 -1.97425 ]

# Среднеквадратические отклонения: 0.7654501708347106

# Коэффициент корреляции Пирсона: 0.47312240727810173

# Полиноминальная 2-й степени функция:

# Коэффициенты: [ 0.39836028 1.22199339 -1.01818533]

# Среднеквадратические отклонения: 0.4485761475660098

# Полиноминальная 3-й степени функция:

# Коэффициенты: [ 0.41229088 2.87210558 4.99527961 -0.06826707]

# Среднеквадратические отклонения: 0.0575601749682143

# Экспоненциальная функция:

# Коэффициенты: [-1.81877293 0.20333813]

# Среднеквадратические отклонения: 0.8339978987658473

# Error fitting Логарифмическая function: Input contains NaN.

# Error fitting Степенная function: Input contains NaN.

# Лучшая функция приближения: Полиноминальная 3-й степени

# 

# Входные данные:

# 1 2

# 2 3.5

# 3 5

# 4 7.2

# 5 8

# 6 10.3

# 7 11.5

# 8 12

# Выходные данные:

# Линейная функция:

# Коэффициенты: [1.50833333 0.65 ]

# Среднеквадратические отклонения: 0.16822916666666685

# Коэффициент корреляции Пирсона: 0.9930311837719734

# Полиноминальная 2-й степени функция:

# Коэффициенты: [-0.05297619 1.98511905 -0.14464286]

# Среднеквадратические отклонения: 0.10929315476190478

# Полиноминальная 3-й степени функция:

# Коэффициенты: [-0.0239899 0.27088745 0.74963925 1.04285714]

# Среднеквадратические отклонения: 0.06656114718614714

# Экспоненциальная функция:

# Коэффициенты: [2.93320584 0.18854186]

# Среднеквадратические отклонения: 0.9336020111182868

# Логарифмическая функция:

# Коэффициенты: [5.10793174 0.66655154]

# Среднеквадратические отклонения: 0.817076627839848

# Степенная функция:

# Коэффициенты: [2.01548111 0.87951981]

# Среднеквадратические отклонения: 0.13653019857470727

# Лучшая функция приближения: Полиноминальная 3-й степени

# Входные данные:

# 0 0

# 1 1

# 1.5 2

# 1 3

# 0 4

# Выходные данные:

# Линейная функция:

# Коэффициенты: [-2.18158824e-12 2.00000000e+00]

# Среднеквадратические отклонения: 2.0

# Коэффициент корреляции Пирсона: 0.0

# Полиноминальная 2-й степени функция:

# Коэффициенты: [-2.18114415e-12 -2.18092211e-12 2.00000000e+00]

# Среднеквадратические отклонения: 2.0

# Полиноминальная 3-й степени функция:

# Коэффициенты: [-66.50507401 166.26268502 -99.75761101 2. ]

# Среднеквадратические отклонения: 2.0

# Экспоненциальная функция:

# Коэффициенты: [ 2.00000001e+00 -2.35028173e-09]

# Среднеквадратические отклонения: 2.0

# Error fitting Логарифмическая function: Input contains infinity or a value too large for dtype('float64').

# Степенная функция:

# Коэффициенты: [2.00000000e+00 7.27663843e-09]

# Среднеквадратические отклонения: 3.6

# Лучшая функция приближения: Линейная

# 

# Вычислительная часть

|  |  |
| --- | --- |
| x | y |
| -4 | -0.2307 |
| -3.6 | -0.3140 |
| -3.2 | -0.4409 |
| -2.8 | -0.6415 |
| -2.4 | -0.9683 |
| -2 | -1.5000 |
| -1.6 | -2.2741 |
| -1.2 | -2.9636 |
| -0.8 | -2.7213 |
| -0.4 | -1.4905 |
| 0 | 0.0000 |

# *Линейная аппроксимация*

# SX = -22

# SXX = 61.6

# SY = -13.5449

# SXY = 20.55232

# Delta = SXX\*n -SX\*SX = 61.6 \* 11 - (22 \* 22) = 193.6

# Delta1 = SXY \* n - SX \* SY = 20.55232 \* 11 - (-22)\*(-13.5449) = -71.91228

# Delta2 = SXX \* SY — SX\*SXY = -382.2148

# A = Delta1 / Delta = -0.3714

# B = Delta2 / Delta = -1.9743

# Y = -0.3714x — 1.9743

# Cреднеквадратическое отклонения = sqrt(SUM(phi(x\_i) — y\_i) / n) = sqrt(0.5509 / 11) = 0.2238

# *Квадратичная аппроксимация*

# SX = -22

# SXX = 61.6

# SXXX = -193.6

# SXXXX = 648.5248

# SY = -13.5449

# SXY = 20.55232

# SXXY = -40.9516

# Решаем систему уравнений

# 11a\_0 -22a\_1 + 61.6a\_2 = -13.5449

# -22a\_0 + 61.6a\_1 -193.6a\_2 = 20.55232

# 61.6a\_0 -193.6a\_1 + 648.5248a\_2 = -40.9516

# A\_0 = 0.3984; A\_1 = 1.2220; A\_2 = -1.0182

# Y = 0.3984x^2+1.2220x -1.0182

# Среднеквадратичкое отклонение = sqrt(SUM(phi(x\_i) — y\_i) / n) = sqrt(2.2126/11) = 0.4485

# Лучшая функция приближения: Полиноминальная 2-й степени

# Выводы

# В ходе данной работы была выполнена аппроксимация функций с использованием линейного, квадратичного, кубического, экспоненциального и логарифмического приближений. Также на основе этих методов был реализован Python скрипт, который реализует метод наименьших квадратов и строит графики исходной функции и аппроксимаций. Исследование позволило определить наилучшее приближение, вычислить среднеквадратические отклонения и коэффициент корреляции Пирсона для линейной зависимости.