Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное 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 «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 6

по курсу «Численные методы»

Студент группы ИУ9-62Б Нащёкин Н.Д.

Преподаватель: Домрачева А.Б.

Москва, 2025

1 Задача

1. Построить графики таблично заданной функции и функции $z(x)$.

2. Найти значения x_a, x_g, x_h, y_a, y_g, y_h, $z(x_a)$, $z(x_g)$, $z(x_h)$, δ_1, ..., δ_9, $\delta_k = min(\delta_i)$.

3. Составить систему уравнений для определения a и b и решить её.

4. Найти среднеквадратичное отклонение Δ.

2 Индивидуальный вариант

Вариант 22.

Функция задана таблично:

x	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
y	1.24	1.74	1.61	2.16	3.06	2.88	4.53	5.40	7.07

Таблица 1 — Значения x и y

3 Основная теория

Аппроксимация методом наименьших квадратов для двупараметрических моделей

Существует формальный метод выбора вида аппроксимирующей функции, зависящей от двух параметров.

Введём обозначения:
$x_a = \frac{x_0+x_n}{2}$ – среднее арифметическое двух чисел,
$x_g = \sqrt{x_0 x_n}$ – среднее геометрическое двух чисел,
$x_h = \frac{2}{\frac{1}{x_0}+\frac{1}{x_n}}$ – среднее гармоническое двух чисел.

Рассмотрим девять видов эмпирических зависимостей, а также их свойства:
$z_1(x) = ax + b \iff z(x_a) = z_a$
$z_2(x) = ax^b \iff z(x_g) = z_g$
$z_3(x) = ae^{bx} \iff z(x_a) = z_g$
$z_4(x) = aln(x) + b \iff z(x_g) = z_a$
$z_5(x) = \frac{a}{x} + b \iff z(x_h) = z_a$
$z_6(x) = \frac{1}{ax+b} \iff z(x_a) = z_h$
$z_7(x) = \frac{x}{ax+b} \iff z(x_h) = z_h$
$z_8(x) = ae^{\frac{b}{x}} \iff z(x_h) = z_g$
$z_9(x) = \frac{1}{aln(x)+b} \iff z(x_g) = z_h.$
Здесь z_a, z_g, z_h – среднее арифметическое, среднее геометрическое и среднее гармоническое значения функции $z(x)$ соответственно в точках x_0 и x_n.

Таким образом, для выбора нужной функции из перечисленного набора нужно выполнить следующие шаги:
1. Нанести на график заданные точки (x_i, y_i) и провести гладкую монотонную кривую $z(x)$, аппроксимирующую эту зависимость.
2. Вычислить значения x_a, x_g, x_h, y_a, y_g, y_h относительно x_0, x_n и y_0, y_n из заданной таблицы, а также по построенному графику $z(x)$ определить значения $z(x_a)$, $z(x_g)$ и $z(x_h)$.
3. Найти минимальное из нижеперечисленных величин:

$\delta_1 = |z(x_a) - y_a|$, $\delta_2 = |z(x_g) - y_g|$, $\delta_3 = |z(x_a) - y_g|$,
$\delta_4 = |z(x_g) - y_a|$, $\delta_5 = |z(x_h) - y_a|$, $\delta_6 = |z(x_a) - y_h|$,
$\delta_7 = |z(x_h) - y_h|$, $\delta_8 = |z(x_h) - y_g|$, $\delta_9 = |z(x_g) - y_h|$.
Номер минимальной δ_i определяет номер аппроксимирующей функции. Пусть такой номер – k.

Затем для выбранной функции $z_k(x)$ нужно методом наименьших квадратов определить коэффициенты a и b. Если, например, мы выбрали функцию $z_1(x) = ax + b$, то среднеквадратичное уклонение равно $\sum_{i=0}^{n}(ax_i + b - y_i)^2$. Коэффициенты a и b находим методом наименьших квадратов:

$$a\sum_{i=0}^{n} x_i^2 + b\sum_{i=0}^{n} x_i = \sum_{i=0}^{n} x_i y_i$$

$$a\sum_{i=0}^{n} x_i + b(n+1) = \sum_{i=0}^{n} y_i$$

Если же выбранная функция нелинейна, необходимо провести предварительную линеаризацию, введя соответствующие замены, а затем, после вычисления коэффициентов, вернуться к ним. Например, для функции z_9 нужно предварительно перейти к обратной величине: $\frac{1}{z_9} = aln(x) + b$. Элементы системы уравнений метода наименьших квадратов в таком случае будут состоять из сумм величин $ln(x_i)$ и $\frac{1}{y_i}$.

4 Практическая реализация

Листинг 1 — реализация программы

```python
from matplotlib import pyplot as plt
import numpy as np


def plot_fun(x_vals, y_vals, z, z_9):
    x_vals_z = np.linspace(1, 5, 1000)
    z_vals = z(x_vals_z)
    z_9_vals = z_9(x_vals_z)

    plt.plot(x_vals, y_vals, 'o', label='Точки таблично заданной функции')
    plt.plot(x_vals_z, z_vals, label='Функция, подобранная вручную')
    plt.plot(x_vals_z, z_9_vals, label='Функция, найденная программно')
    plt.axhline(0, color='gray', linestyle='--')  # ось X
    plt.axvline(0, color='gray', linestyle='--')  # ось Y
    plt.title("Точки из таблицы и z(x)")
    plt.xlabel("x")
    plt.ylabel("z(x)")
    plt.grid(True)
    plt.legend(loc='best')
    plt.show()


def calculate_vals(x_vals, y_vals, z):
    x_a = (x_vals[0] + x_vals[-1]) / 2
    y_a = (y_vals[0] + y_vals[-1]) / 2
    x_g = np.sqrt(x_vals[0] * x_vals[-1])
    y_g = np.sqrt(y_vals[0] * y_vals[-1])
    x_h = 2 / (1 / x_vals[0] + 1 / x_vals[-1])
    y_h = 2 / (1 / y_vals[0] + 1 / y_vals[-1])
    z_a = z(x_a)
    z_g = z(x_g)
    z_h = z(x_h)

    print(f'x_a = {x_a:.1f}, y_a = {y_a:.1f}, \n'
        f'x_g = {x_g:.1f}, y_g = {y_g:.1f}, \n'
        f'x_h = {x_h:.1f}, y_h = {y_h:.1f}, \n'
        f'z_a = {z_a:.1f}, z_g = {z_g:.1f}, z_h = {z_h:.1f} \n')

    delta_1 = abs(z_a - y_a)
    delta_2 = abs(z_g - y_g)
    delta_3 = abs(z_a - y_g)
    delta_4 = abs(z_g - y_a)
    delta_5 = abs(z_h - y_a)
    delta_6 = abs(z_a - y_h)
    delta_7 = abs(z_h - y_h)
    delta_8 = abs(z_h - y_g)
```

```python
    delta_9 = abs(z_g - y_h)

    print(f'delta 1 ... 9: \n'
        f'{delta_1:.1f}, {delta_2:.1f}, {delta_3:.1f}, \n'
        f'{delta_4:.1f}, {delta_5:.1f}, {delta_6:.1f}, \n'
        f'{delta_7:.1f}, {delta_8:.1f}, {delta_9:.1f}')
    print(f'delta min: {min([delta_1, delta_2, delta_3,
        delta_4, delta_5, delta_6,
        delta_7, delta_8, delta_9]):.1f} \n')
    # delta_9 -- минимальное


def min_quad(x_vals, y_vals):
    u = np.log(x_vals)          # ln x_i
    w = 1 / np.array(y_vals)    # 1 / y_i

    n  = len(x_vals)
    S1 = np.sum(u * u)          # sum u_i^2
    S2 = np.sum(u)              # sum u_i
    S3 = np.sum(u * w)          # sum u_i * w_i
    S4 = np.sum(w)              # sum w_i

    det = S1 * n - S2 ** 2
    if abs(det) < 1e-12:
        raise ValueError("Система вырождается. Где-то ошибка")

    a = (S3 * n - S2 * S4) / det
    b = (S1 * S4 - S2 * S3) / det
    return a, b


def sko(x_vals, y_vals, z):
    n = len(x_vals)
    summ = 0
    for i in range(n):
        summ += (z(x_vals[i]) - y_vals[i]) ** 2
    return np.sqrt(summ) / np.sqrt(n - 1)


def main():
    x_vals = np.linspace(1, 5, 9)
    y_vals = [1.24, 1.74, 1.61, 2.16, 3.06, 2.88, 4.53, 5.40, 7.07]

    z = lambda x: 0.7 * np.exp(x * 0.44)

    calculate_vals(x_vals, y_vals, z) # delta_9 оказалось минимальным

    # z_9 = 1/(alnx + b)
    # 1 / z_9 = alnx + b

```

```
    a, b = min_quad(x_vals, y_vals)
    z_9 = lambda x: 1 / (a * np.log(x) + b)

    plot_fun(x_vals, y_vals, z, z_9)

    print(f'СКО: {sko(x_vals, y_vals, z_9):.3}')


if __name__ == '__main__':
    main()
```

Перед выполнением работы была вручную подобрана функция $z(x) = 0.7e^{0.44x}$. Сначала в программе проводится вычисление значений x_a, x_g, x_h, y_a, y_g, y_h, $z(x_a)$, $z(x_g)$, $z(x_h)$, а также δ_1, ..., δ_9, $\delta_k = min(\delta_i)$ и вывод их на экран. Оказалось, что δ_9 является минимальной среди всех δ_i, поэтому в качестве аппроксимационной выбрана функция z_9.

После того, как был определён нужный вид функции, была реализована функция $min\_quad()$ вычисления её коэффициентов методом наименьших квадратов. В ней предварительно проводится линеаризация функции $z_9(x)$: $ln(x_i)$ заменяется на u_i, $\frac{1}{y_i}$ – на w_i. Система уравнений решается методом Крамера, после чего по найденным коэффициентам определяется функция $z_9(x)$. На экран также выводится среднеквадратичное отклонение, вычисляемое в функции $sko()$ и формулы для подобранной вручную и найденной аппроркcимационных функций.

Результат, который вывод программа, приведён на рисунке 1. Графики исходной и найденной программно аппроксимационных функций представлены на рисунке 2. Жёлтым цветом показана функция, подобранная вручную, а зелёным – функция, найденная программно.

```
z(x) = 0.7e^(0.44x)

x_a = 3.0, y_a = 4.2,
x_g = 2.2, y_g = 3.0,
x_h = 1.7, y_h = 2.1,
z_a = 2.6, z_g = 1.9, z_h = 1.5

delta 1 ... 9:
1.5, 1.1, 0.3,
2.3, 2.7, 0.5,
0.7, 1.5, 0.2
delta min: 0.2

z_9(x) = 1 / (-0.408 ln(x) + 0.811)

СКО: 0.381
```

Рис. 1 — Результат выполнения программы

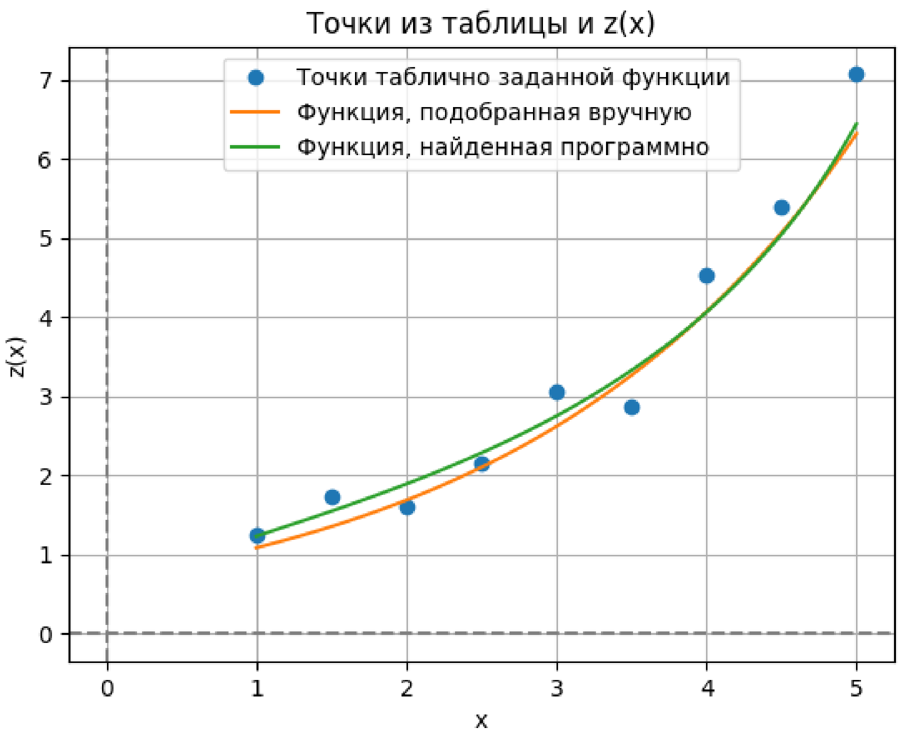

Рис. 2 — Построенный график функций

5 Вывод

В данной лабораторной работе была реализована аппроксимация методом наименьших квадратов для двупараметрических моделей. Заданная таблично функция, исходная аппроксимационная функция, найденная вручную, а также найденная программно аппроксимационная функция выводятся на график. Также на экран выводится итоговое среднеквадратичное отклонение для найденной функции и формулы, описывающие исходную и найденную функции.