Отчет по Лабораторной работе №4
по курсу “вычислительная математика”

Вариант №3

Выполнил:
Студент группы р32082
Дробыш Дмитрий Александрович

Преподаватель:
Машина Екатерина Алексеевна

Санкт-Петербург, 2023

ЛАБОРАТОРНАЯ РАБОТА №4.
«АППРОКСИМАЦИЯ ФУНКЦИИ МЕТОДОМ НАИМЕНЬШИХ КВАДРАТОВ»

Цель лабораторной работы: найти функцию, являющуюся наилучшим приближением заданной табличной функции по методу наименьших квадратов.

Лабораторная работа состоит из двух частей: вычислительной и программной.

№ варианта задания лабораторной работы определяется как номер в списке группы согласно ИСУ.

Порядок выполнения работы

Вычислительная реализация задачи

Вычислительная часть лабораторной работы должна быть представлена только в отчете.

Задание:

1. Сформировать таблицу табулирования заданной функции на указанном интервале (см. табл. 1)
2. Построить линейное и квадратичное приближения по 11 точкам заданного интервала;
3. Найти среднеквадратические отклонения для каждой аппроксимирующей функции. Ответы дать с тремя знаками после запятой;
4. Выбрать наилучшее приближение;
5. Построить графики заданной функции, а также полученные линейное и квадратичное приближения;
6. Привести в отчете **подробные вычисления**.

Программная реализация задачи

Для исследования использовать:

- линейную функцию,
- полиномиальную функцию 2-й степени,
- полиномиальную функцию 3-й степени,
- экспоненциальную функцию,
- логарифмическую функцию,
- степенную функцию.

Методика проведения исследования:

1. Вычислить меру отклонения: $S = \sum_{i=1}^{n}[\varphi(x_i) - y_i]^2$ для всех исследуемых функций;
2. Уточнить значения коэффициентов эмпирических функций, минимизируя функцию S;
3. Сформировать массивы предполагаемых эмпирических зависимостей $(\varphi(x_i), \varepsilon_i)$;
4. Определить среднеквадратичное отклонение для каждой аппроксимирующей функции. Выбрать наименьшее значение и, следовательно, наилучшее приближение;
5. Построить графики полученных эмпирических функций.

Задание:

1. Предусмотреть ввод исходных данных из файла/консоли (таблица $y = f(x)$ должна содержать от 8 до 12 точек);
2. Реализовать метод наименьших квадратов, исследуя все указанные функции;
3. Предусмотреть вывод результатов в файл/консоль: коэффициенты аппроксимирующих функций, среднеквадратичное отклонение, массивы значений $\boldsymbol{x_i}, \boldsymbol{y_i}, \boldsymbol{\varphi(x_i)}, \boldsymbol{\varepsilon_i}$;
4. Для линейной зависимости вычислить коэффициент корреляции Пирсона;
5. Программа должна отображать наилучшую аппроксимирующую функцию;
6. Организовать вывод графиков функций, графики должны полностью отображать весь исследуемый интервал (с запасом);
7. Программа должна быть протестирована при различных наборах данных, в том числе и некорректных;

Требования и содержание отчета

Отчет должен содержать следующие разделы:

- Цель работы,
- Рабочие формулы метода,
- Вычислительная часть лабораторной работы,
- Листинг программы (по крайней мере, коды используемого метода),
- Графики аппроксимирующих функций,
- Результаты выполнения программы при различных исходных данных (не менее трех),
- Выводы.

Варианты задания

Таблица 1. Варианты задания для вычислительной реализации задачи

№ варианта	Функция	Исследуемый интервал	№ варианта	Функция	Исследуемый интервал
1	$y = \dfrac{12x}{x^4 + 1}$	$x \in [0, 2]$ $h = 0{,}2$	21	$y = \dfrac{14x}{x^4 + 21}$	$x \in [-4, 0]$ $h = 0{,}4$
2	$y = \dfrac{15x}{x^4 + 2}$	$x \in [0, 4]$ $h = 0{,}4$	22	$y = \dfrac{5x}{x^4 + 22}$	$x \in [-2, 0]$ $h = 0{,}2$
3	$y = \dfrac{4x}{x^4 + 3}$	$x \in [-2, 0]$ $h = 0{,}2$	23	$y = \dfrac{16x}{x^4 + 23}$	$x \in [0, 4]$ $h = 0{,}4$
4	$y = \dfrac{15x}{x^4 + 4}$	$x \in [-4, 0]$ $h = 0{,}4$	24	$y = \dfrac{7x}{x^4 + 24}$	$x \in [-4, 0]$ $h = 0{,}4$
5	$y = \dfrac{6x}{x^4 + 5}$	$x \in [0, 2]$ $h = 0{,}2$	25	$y = \dfrac{28x}{x^4 + 25}$	$x \in [0, 4]$ $h = 0{,}4$
6	$y = \dfrac{12x}{x^4 + 6}$	$x \in [0, 2]$ $h = 0{,}2$	26	$y = \dfrac{7x}{x^4 + 26}$	$x \in [0, 4]$ $h = 0{,}4$
7	$y = \dfrac{23x}{x^4 + 7}$	$x \in [-2, 0]$ $h = 0{,}2$	27	$y = \dfrac{18x}{x^4 + 27}$	$x \in [0, 2]$ $h = 0{,}2$
8	$y = \dfrac{3x}{x^4 + 8}$	$x \in [-2, 0]$ $h = 0{,}2$	28	$y = \dfrac{21x}{x^4 + 28}$	$x \in [-4{,}0]$ $h = 0{,}4$
9	$y = \dfrac{4x}{x^4 + 9}$	$x \in [0, 2]$ $h = 0{,}2$	29	$y = \dfrac{15x}{x^4 + 29}$	$x \in [0, 4]$ $h = 0{,}4$
10	$y = \dfrac{18x}{x^4 + 10}$	$x \in [0, 4]$ $h = 0{,}4$	30	$y = \dfrac{16x}{x^4 + 30}$	$x \in [-4{,}0]$ $h = 0{,}4$
11	$y = \dfrac{5x}{x^4 + 11}$	$x \in [-2, 0]$ $h = 0{,}2$	31	$y = \dfrac{15x}{x^4 + 31}$	$x \in [-4{,}0]$ $h = 0{,}4$
12	$y = \dfrac{4x}{x^4 + 12}$	$x \in [-2, 0]$ $h = 0{,}2$	32	$y = \dfrac{25x}{x^4 + 32}$	$x \in [0, 4]$ $h = 0{,}4$
13	$y = \dfrac{31x}{x^4 + 13}$	$x \in [0, 4]$ $h = 0{,}4$	33	$y = \dfrac{26x}{x^4 + 33}$	$x \in [-2{,}0]$ $h = 0{,}2$
14	$y = \dfrac{25x}{x^4 + 14}$	$x \in [0, 4]$ $h = 0{,}4$	34	$y = \dfrac{22x}{x^4 + 34}$	$x \in [-4{,}0]$ $h = 0{,}4$

15	$y = \dfrac{4x}{x^4 + 15}$	$x \in [-2, 0]$ $h = 0{,}2$	35	$y = \dfrac{19x}{x^4 + 35}$	$x \in [0, 4]$ $h = 0{,}4$
16	$y = \dfrac{17x}{x^4 + 16}$	$x \in [-4, 0]$ $h = 0{,}4$	36	$y = \dfrac{47x}{x^4 + 36}$	$x \in [-2{,}2]$ $h = 0{,}4$
17	$y = \dfrac{2x}{x^4 + 17}$	$x \in [0, 2]$ $h = 0{,}2$	37	$y = \dfrac{23x}{x^4 + 37}$	$x \in [-2, 0]$ $h = 0{,}2$
18	$y = \dfrac{30x}{x^4 + 18}$	$x \in [0, 4]$ $h = 0{,}4$	38	$y = \dfrac{14x}{x^4 + 38}$	$x \in [0, 2]$ $h = 0{,}2$
19	$y = \dfrac{5x}{x^4 + 19}$	$x \in [0, 2]$ $h = 0{,}2$	39	$y = \dfrac{20x}{x^4 + 39}$	$x \in [0, 4]$ $h = 0{,}4$
20	$y = \dfrac{11x}{x^4 + 20}$	$x \in [0, 4]$ $h = 0{,}4$	40	$y = \dfrac{5x}{x^4 + 40}$	$x \in [-2, 0]$ $h = 0{,}2$

Контрольные вопросы

1. Чем вызвана необходимость аппроксимирования табличных функций?
2. Чем отличается аппроксимации от интерполяции?
3. Сформулируйте задачу аппроксимации.
4. Как выбирается вид аппроксимирующего уравнения?
5. Объясните суть метода наименьших квадратов (МНК).
6. Что такое мера отклонения и как ее вычислить?
7. К решению какой задачи сводится МНК?
8. Сформулируйте задачу полиномиальной аппроксимации МНК.
9. Что такое линейная и квадратичная аппроксимация?
10. Приведите графическую интерпретацию линейной и квадратичной аппроксимаций?
11. Что такое среднеквадратическое отклонение?
12. Как выполняется аппроксимации данных неполиномиальными функциями?
13. Как оценить качество полученной аппроксимации?
14. Как выбирается наилучшая аппроксимирующая функция?
15. Корректно ли применять аппроксимирующие уравнения за пределами исследуемого диапазона?

Исходная функция

X	Y
0	0
0,2	0,266524520255864
0,4	0,528820729772607
0,6	0,766871165644172
0,8	0,938526513374003
1	1
1,2	0,946073793755913
1,4	0,818521983161833
1,6	0,669904538603249
1,8	0,533428165007112
2	0,421052631578947

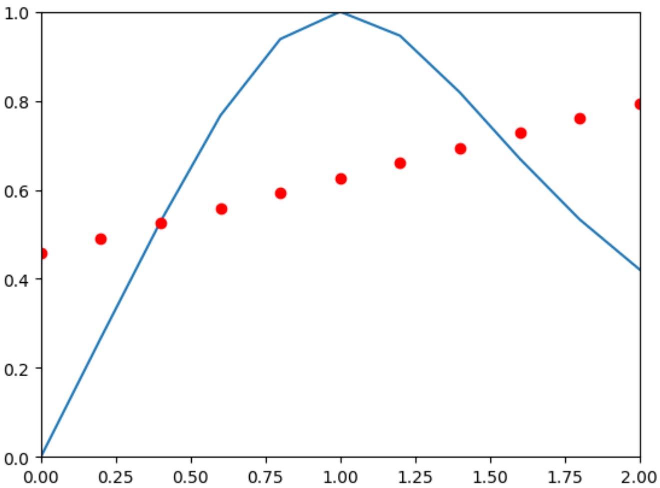
Stat	Value
N	11
sum_xi	11
sum_xi2	15,4
sum_xi3	24,2
sum_xi4	40,5328
sum_yi	6,8897240411537
sum_xi_yi	7,63111965691548
sum_xi2_yi	10,066125635534

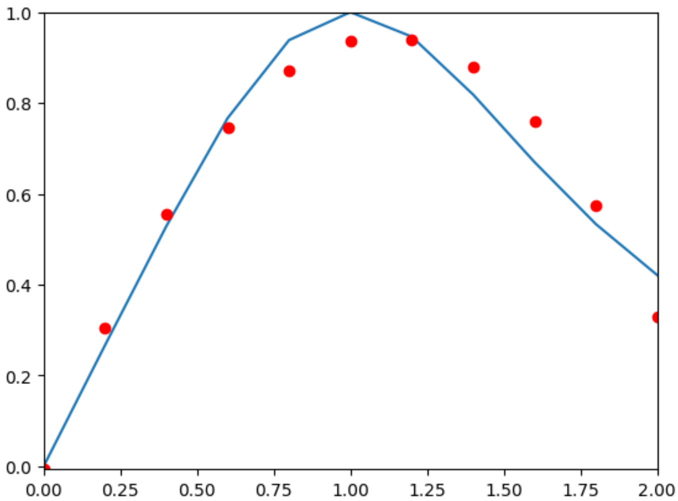
matrix_with_equations_linear = [[sum_xi2, sum_xi, sum_xi_yi], [sum_xi, n, sum_yi]]

matrix_of_equations_square =
[[sum_xi3, sum_xi2, sum_xi, sum_xi_yi], [sum_xi4, sum_xi3, sum_xi2, sum_xi2_yi],[sum_xi2, sum_xi, n, sum_yi]]

При помощи собственного метода Гаусса решаем систему линейных уравнений:
Линейная: 0.16849900358222247x + 0.4578395456135685
Квадратичная: -0.7738048176025312x*x + 1.7161086387872844x + -0.006443349479498256

Построим графики:





Найдем корень из диспресии = СКО
На итерации 0 сумма квадратов отклонений равна : 0
На итерации 1 сумма квадратов отклонений равна : 0.20961704952763885
На итерации 2 сумма квадратов отклонений равна : 0.26024872148081857
На итерации 3 сумма квадратов отклонений равна : 0.2602615492156428
На итерации 4 сумма квадратов отклонений равна : 0.30349735644866666
На итерации 5 сумма квадратов отклонений равна : 0.4231357023524877
На итерации 6 сумма квадратов отклонений равна : 0.5627585821695941
На итерации 7 сумма квадратов отклонений равна : 0.6445748573044436
На итерации 8 сумма квадратов отклонений равна : 0.6601458621661057
На итерации 9 сумма квадратов отклонений равна : 0.6634559557478327
На итерации 10 сумма квадратов отклонений равна : 0.7153076117843438
Для нахождения Дисперсии разделим на 11 и возьмем корень
0.2787998270346626

Найдем корень из диспресии = СКО
На итерации 0 сумма квадратов отклонений равна : 0
На итерации 1 сумма квадратов отклонений равна : 4.151669411827054e-05
На итерации 2 сумма квадратов отклонений равна : 0.0015861379470806848
На итерации 3 сумма квадратов отклонений равна : 0.0023352882376458405
На итерации 4 сумма квадратов отклонений равна : 0.0028289749385483924
На итерации 5 сумма квадратов отклонений равна : 0.007360692176721895
На итерации 6 сумма квадратов отклонений равна : 0.011474570685291563
На итерации 7 сумма квадратов отклонений равна : 0.011530307503730982
На итерации 8 сумма квадратов отклонений равна : 0.015242689989228946
На итерации 9 сумма квадратов отклонений равна : 0.023072392360369932
На итерации 10 сумма квадратов отклонений равна : 0.024836092562816218
Для нахождения Дисперсии разделим на 11 и возьмем корень
0.05479380773038943

Линейная аппроксимация

Введем обозначения:

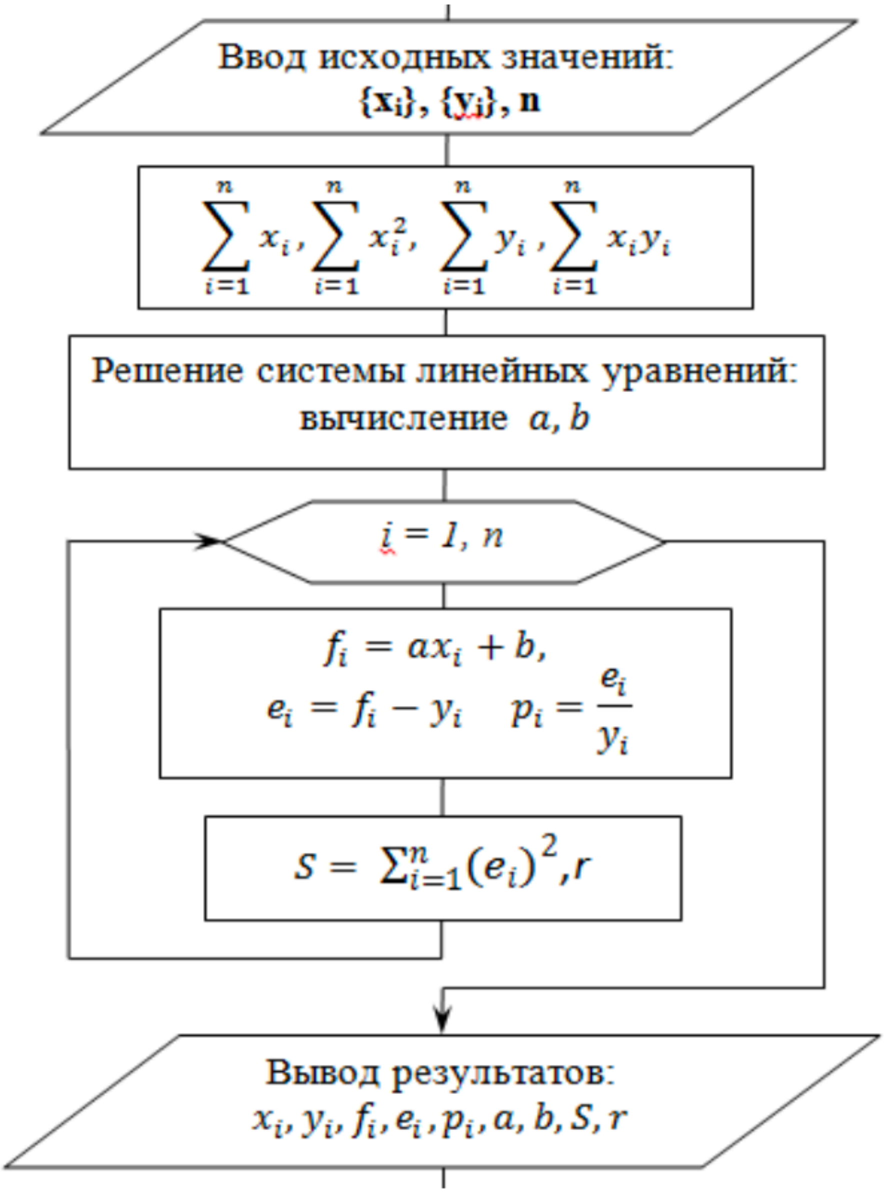
$SX = \sum_{i=1}^{n} x_i$, $SXX = \sum_{i=1}^{n} x_i^2$, $SY = \sum_{i=1}^{n} y_i$, $SXY = \sum_{i=1}^{n} x_i y_i$

Получим систему уравнений для нахождения параметров *a* и *b:*

$$\begin{cases} aSXX + bSX = SXY \\ aSX + bn = SY \end{cases},$$

из которой находим:

$$\Delta = SXX \cdot n - SX \cdot SX$$
$$\Delta_1 = SXY \cdot n - SX \cdot SY$$
$$\Delta_2 = SXX \cdot SY - SX \cdot SXY$$
$$a = \frac{\Delta_1}{\Delta}, \quad b = \frac{\Delta_2}{\Delta}$$



```python
def linear_approximation_from_array(X, Y):
    X, Y = list(X).copy(), list(Y).copy()
    n = len(X)

    sum_xi = 0
    sum_xi2 = 0

    sum_yi = 0
    sum_xi_yi = 0

    for i in range(n):
        sum_xi += X[i]
        sum_xi2 += X[i] ** 2

        sum_yi += Y[i]
        sum_xi_yi += Y[i] * X[i]

    matrix_with_equations = [[sum_xi2, sum_xi, sum_xi_yi], [sum_xi, n, sum_yi]]
    gauss_answer = gauss(matrix_with_equations)
    a, b = gauss_answer

    def new_function(x):
        return a * x + b

    return True, new_function, a, b
```

КВАДРАТИЧНАЯ АППРОКСИМАЦИЯ

Рассмотрим в качестве эмпирической формулы квадратичную функцию:

$$\varphi(x, a_0, a_1, a_2) = a_0 + a_1 x + a_2 x^2$$

Сумма квадратов отклонений запишется следующим образом:

$$S = S(a_0, a_1, a_2) = \sum_{i=1}^{n} (a_0 + a_1 x_i + a_2 x_i^2 - y_i)^2 \to min$$

```python
def square_approximation_from_array(X, Y):
    X, Y = list(X).copy(), list(Y).copy()
    n = len(X)

    sum_xi = 0
    sum_xi2 = 0
    sum_xi3 = 0
    sum_xi4 = 0

    sum_yi = 0
    sum_xi_yi = 0
    sum_xi2_yi = 0

    for i in range(n):
        sum_xi += X[i]
        sum_xi2 += X[i] ** 2
        sum_xi3 += X[i] ** 3
        sum_xi4 += X[i] ** 4

        sum_yi += Y[i]
        sum_xi_yi += X[i] * Y[i]
        sum_xi2_yi += X[i] ** 2 * Y[i]

    matrix_of_equations = [[sum_xi3, sum_xi2, sum_xi, sum_xi_yi], [sum_xi4, sum_xi3, sum_xi2, sum_xi2_yi],
                           [sum_xi2, sum_xi, n, sum_yi]]
    gauss_answer = gauss(matrix_of_equations)

    a, b, c = gauss_answer

    def new_function(x):
        return a * x ** 2 + b * x + c

    return True, new_function, a, b, c

```

Аппроксимация с помощью других функций

Вид функции	Табличный X	Табличный Y
Степенная	Ln X	Ln Y
Экспоненциальная	X	Ln Y
Логарифмическая	Ln X	Y

Аппроксимирующая функция задана степенной функцией вида:

$$\boldsymbol{\varphi(x) = ax^b}$$

Для применения метода наименьших квадратов степенная функция ***линеаризуется***:

$$\ln(\varphi(x)) = \ln(ax^b) = \ln(a) + b\ln(x)$$

Введем обозначения: Y=$\ln(\varphi(x))$; A=$\ln(a)$; B=b; X=$\ln(x)$

Получаем линейную зависимость: Y=A+BX.

После определения коэффициентов A и B вернемся к принятым ранее обозначениям:

$$\boldsymbol{a = e^{\mathrm{A}}} \qquad \boldsymbol{b = B}$$

```python
def power_approximation_from_array(X, Y):
    X, Y = list(X).copy(), list(Y).copy()
    n = len(X)

    sum_xi = 0
    sum_xi2 = 0

    sum_yi = 0
    sum_xi_yi = 0

    for i in range(n):
        if X[i] <= 0 or Y[i] <= 0:
            return False, False
        sum_xi += ln(X[i])
        sum_xi2 += ln(X[i]) ** 2

        sum_yi += ln(Y[i])
        sum_xi_yi += ln(X[i]) * ln(Y[i])
    matrix_with_equations = [[sum_xi2, sum_xi, sum_xi_yi], [sum_xi, n, sum_yi]]
    gauss_answer = gauss(matrix_with_equations)
    a, b = math.e ** gauss_answer[1], gauss_answer[0]

    def new_function(x):
        return a * x ** b

    return True, new_function, a, b
```

Аппроксимация с помощью других функций

Аппроксимирующая функция задана экспоненциальной функцией вида:

$$\varphi(x) = ae^{bx}$$

Для применения метода наименьших квадратов экспоненциальная функция линеаризуется:

$$\ln\big(\varphi(x)\big) = \ln\big(ae^{bx}\big) = \ln a + bx$$

Введем обозначения: Y=$\ln\big(\varphi(x)\big)$; A=$\ln(a)$; B=b

Получаем линейную зависимость: Y=A+Bx.

После определения коэффициентов A и B вернемся к принятым ранее обозначениям:

$$a = e^{A} \quad b = B$$

Аппроксимирующая функция задана логарифмической функцией вида:

$$\varphi(x) = aln(x) + b$$

```python
def logarithm_approximation_from_array(X, Y):
    X, Y = list(X).copy(), list(Y).copy()
    n = len(X)

    sum_xi = 0
    sum_xi2 = 0

    sum_yi = 0
    sum_xi_yi = 0

    for i in range(n):
        if X[i] <= 0:
            return False, False
        sum_xi += ln(X[i])
        sum_xi2 += ln(X[i]) ** 2

        sum_yi += Y[i]
        sum_xi_yi += ln(X[i]) * Y[i]
    matrix_with_equations = [[sum_xi2, sum_xi, sum_xi_yi], [sum_xi, n, sum_yi]]
    gauss_answer = gauss(matrix_with_equations)
    a, b = gauss_answer

    def new_function(x):
        return a * ln(x) + b

    return True, new_function, a, b
```

```python
def exponential_approximation_from_array(X, Y):
    X, Y = list(X).copy(), list(Y).copy()
    n = len(X)

    sum_xi = 0
    sum_xi2 = 0

    sum_yi = 0
    sum_xi_yi = 0

    for i in range(n):
        if Y[i] <= 0:
            return False, False
        sum_xi += X[i]
        sum_xi2 += X[i] ** 2

        sum_yi += ln(Y[i])
        sum_xi_yi += X[i] * ln(Y[i])
    matrix_with_equations = [[sum_xi2, sum_xi, sum_xi_yi], [sum_xi, n, sum_yi]]
    gauss_answer = gauss(matrix_with_equations)
    a, b = math.e ** gauss_answer[1], gauss_answer[0]

    def new_function(x):
        return a * math.e**(b * x)

    return True, new_function, a, b
```

```python
def cube_approximation_from_array(X, Y):
    X, Y = list(X).copy(), list(Y).copy()
    n = len(X)

    sum_xi = 0
    sum_xi2 = 0
    sum_xi3 = 0
    sum_xi4 = 0
    sum_xi5 = 0
    sum_xi6 = 0

    sum_yi = 0
    sum_xi_yi = 0
    sum_xi2_yi = 0
    sum_xi3_yi = 0

    for i in range(n):
        sum_xi += X[i]
        sum_xi2 += X[i] ** 2
        sum_xi3 += X[i] ** 3
        sum_xi4 += X[i] ** 4
        sum_xi5 += X[i] ** 5
        sum_xi6 += X[i] ** 6

        sum_yi += Y[i]
        sum_xi_yi += X[i] * Y[i]
        sum_xi2_yi += X[i] ** 2 * Y[i]
        sum_xi3_yi += X[i] ** 3 * Y[i]
    matrix_with_equations = [[sum_xi6, sum_xi5, sum_xi4, sum_xi3, sum_xi3_yi],
                             [sum_xi5, sum_xi4, sum_xi3, sum_xi2, sum_xi2_yi],
                             [sum_xi4, sum_xi3, sum_xi2, sum_xi, sum_xi_yi], [sum_xi3, sum_xi2, sum_xi, n, sum_yi]]
    gauss_answer = gauss(matrix_with_equations)
    a, b, c, d = gauss_answer

    def new_function(x):
        return a * x ** 3 + b * x ** 2 + c * x + d

    return True, new_function, a, b, c, d
```

Для оценки R2 нужно:

Выбор аппроксимирующей функции

Достоверность аппроксимации:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \varphi_i)^2}{\sum_{i=1}^{n}\varphi_i{}^2 - \frac{1}{n}\left(\sum_{i=1}^{n}\varphi_i\right)^2}$$

Чем ближе значение R^2 к единице ($R^2 \rightarrow 1$), тем надежнее функция φ аппроксимирует исследуемый процесс.

Вывод:
В ходе лабораторной работы я ознакомился с методами аппроксимации: линейная, квадратичная, кубическая, логарифмическая, экспоненциальная, степенная. К сожалению, даже такой большой список не станет панацеей, тк существует множество функций, для которых эти методы не смогут подобрать функцию схожего вида… Хотя для большинства функций этого будет достаточно. Кроме того, лабораторная работа вынудила освежить свои знания в области статистики). Довольно неплохое тз в этот раз и интересное задание! Respect.