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

Мегафакультет компьютерных технологий и управления
Факультет программной инженерии и компьютерной техники

Отчёт
по лабораторной работе №4
по дисциплине «Вычислительная математика»

Вариант №4

Группа: P3218

Студент: Горло Евгений Николаевич

Преподаватель: Бострикова Дарья Константиновна

Санкт-Петербург
2024

Содержание

1 Цель работы

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

2 Задание

Программная реализация

Для исследования использовать:

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

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 $x_i, y_i, \phi(x_i), \epsilon_i$;
4. Для линейной зависимости вычислить коэффициент корреляции Пирсона;
5. Вычислить коэффициент детерминации, программа должна выводить соответствующее сообщение в зависимости от полученного значения R^2;
6. Программа должна отображать наилучшую аппроксимирующую функцию;
7. Организовать вывод графиков функций, графики должны полностью отображать весь исследуемый интервал (с запасом);
8. Программа должна быть протестирована при различных наборах данных, в том числе и некорректных;

Вычислительная реализация

1. Сформировать таблицу табулирования заданной функции на указанном интервале.
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.

Функция из варианта:

$y = \frac{15x}{x^4+4}$, $x \in [-4, 0]$, $h = 0,4$

3 Рабочие формулы методов

Метод наименьших квадратов

$$S = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (\phi(x_i) - y_i)^2 \to min$$

Линейная аппроксимация

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

$$\begin{cases} aSXX + bSX = SXY \\ aSX + bn = SY \end{cases} \tag{1}$$

Квадратичная аппроксимация

$SX = \sum_{i=1}^{n} x_i \quad SXX = \sum_{i=1}^{n} x_i^2 \quad SXXX = \sum_{i=1}^{n} x_i^3 \quad SXXXX = \sum_{i=1}^{n} x_i^4$
$SY = \sum_{i=1}^{n} y_i \quad SXY = \sum_{i=1}^{n} x_i \cdot y_i, \quad SXXY = \sum_{i=1}^{n} x_i^2 \cdot y_i,$

$$\begin{cases} an + SXb + SXXc = SY \\ SXa + SXXb + SXXXc = SXY \\ SXXa + SXXXb + SXXXXc = SXXY \end{cases} \tag{2}$$

Среднекватратичное отклонение

$$\delta = \sqrt{\frac{\sum_{i=1}^{n} (\phi(x_i) - y_i)^2}{n}}$$

Коэффициент детерминации

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (\phi(x_i) - y_i)^2}{\sum_{i=1}^{n} (\hat{\phi}(x_i) - y_i)^2} \quad \hat{\phi}(x_i) = \frac{\sum_{i=1}^{n} \phi(x_i)}{n}$$

Критерий корреляции Пирсона

$$r = \frac{\sum_{i=1}^{n} (x_i - \hat{x_i})(y_i - \hat{y_i})}{\sqrt{\sum_{i=1}^{n} (x_i - \hat{x_i})^2 \cdot \sum_{i=1}^{n} (y_i - \hat{y_i})^2}} \quad \hat{x_i} = \frac{\sum_{i=1}^{n} x_i}{n} \quad \hat{y_i} = \frac{\sum_{i=1}^{n} y_i}{n}$$

4 Вычислительная часть

$y = \frac{15x}{x^4+4}$, $x \in [-4, 0]$, $h = 0,4$

x	y
-4	-0.230769
-3,6	-0.3140236
-3,2	-0.44094303
-2,8	-0.641558
-2,4	-0.968325
-2	-1.5
-1,6	-2.274106
-1,2	-2.963646
-0,8	-2.721335
-0,4	-1.490461
0	0

Линейная аппроксимация:

$$\begin{cases} a\sum\limits_{i=1}^{n} x_i^2 + b\sum\limits_{i=1}^{n} x_i = \sum\limits_{i=1}^{n} x_i y_i \\ a\sum\limits_{i=1}^{n} x_i + bn = \sum\limits_{i=1}^{n} y_i \end{cases}$$

$$\begin{cases} a \cdot 61.6 + b \cdot -22 = 20.553 \\ a \cdot -22 + b \cdot 11 = -13.545 \end{cases}$$

$$a = -0.37142; b = -1.9742$$

$$f(x) = y = -0.37142x + -1.9742$$

$$CKO = \delta = 0.875$$

Квадратичная аппроксимация:

$$\begin{cases} a_0 n + a_1\sum\limits_{i=1}^{n} x_i + a_2\sum\limits_{i=1}^{n} x_i^2 = \sum\limits_{i=1}^{n} y_i \\ a_0\sum\limits_{i=1}^{n} x_i + a_1\sum\limits_{i=1}^{n} x_i^2 + a_2\sum\limits_{i=1}^{n} x_i^3 = \sum\limits_{i=1}^{n} x_i y_i \\ a_0\sum\limits_{i=1}^{n} x_i^2 + a_1\sum\limits_{i=1}^{n} x_i^3 + a_2\sum\limits_{i=1}^{n} x_i^4 = \sum\limits_{i=1}^{n} x_i^2 y_i \end{cases}$$

$$\begin{cases} a_0 \cdot 11 + a_1 \cdot -22 + a_2 \cdot 61.6 = -13.545 \\ a_0 \cdot -22 + a_1 \cdot 61.6 + a_2 \cdot -193.6 = 20.553 \\ a_0 \cdot 61.6 + a_1 \cdot -193.6 + a_2 \cdot 648.5248 = -40.954 \end{cases}$$

$$a_0 = -1.15069; a_1 = 1.0011; a_2 = 0.343129$$
$$f(x) = y = 0.343129x^2 + 1.0011x - 1.15069$$
$$CKO = \delta = 0.669$$

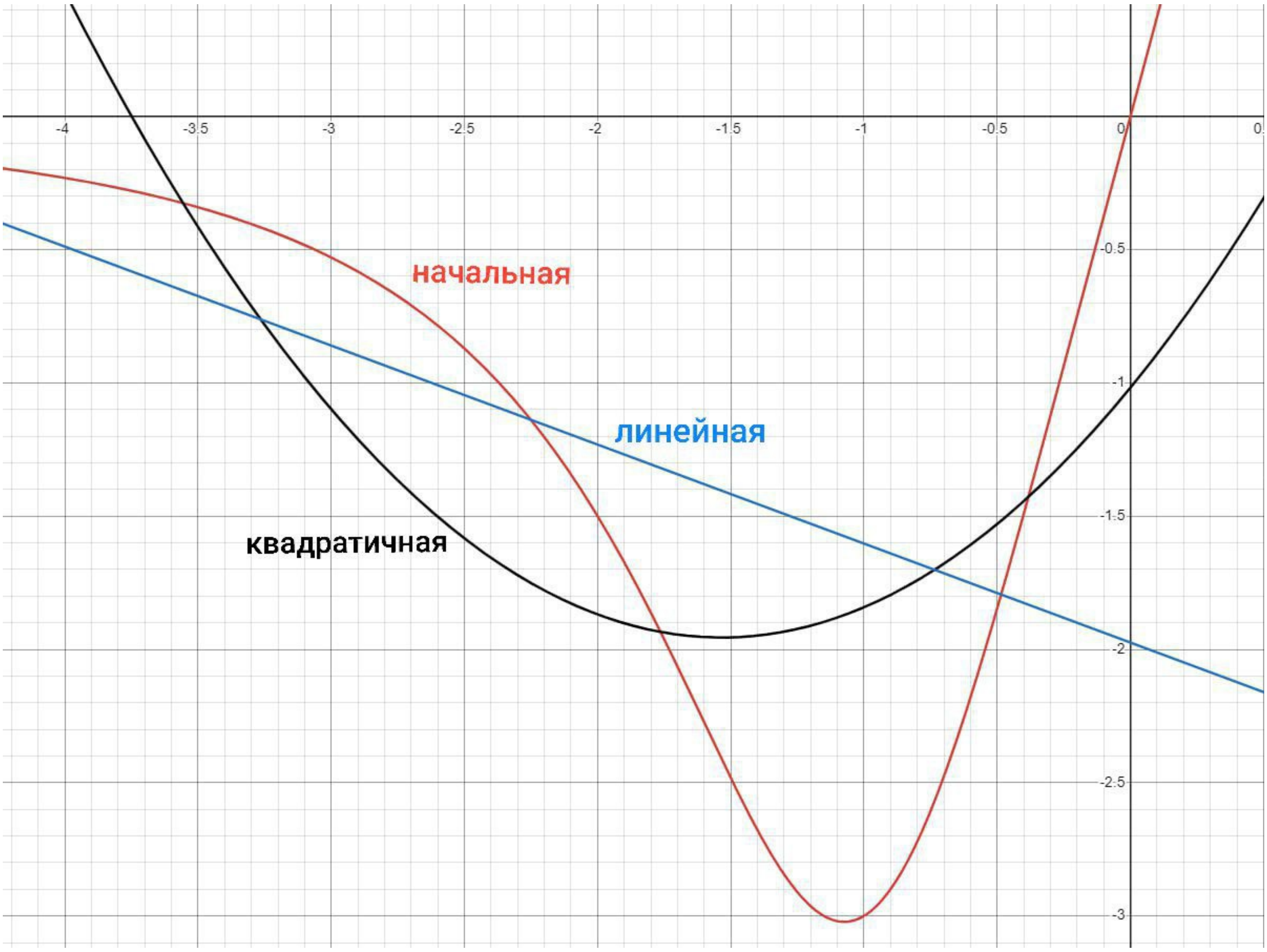

Рис. 1: График

Лучашая аппроксимация: квадратичная

5 Листинг программы

Ссылка на Github репозиторий

Метод для линейной функции:

```
def linear(coordinates):
    try:
        x_coord = coordinates[0]
        y_coord = coordinates[1]
        a, b = lin_approx(x_coord, y_coord)
        f = [a * x + b for x in x_coord]
        eps, delta, S, SS_total, R_squared = aprox_staff_computing(x_coord,
            y_coord, f)
        return a, b, cof_cor(x_coord, y_coord), S, delta, R_squared, x_coord,
            y_coord, f, eps
    except BaseException:
        return ['']
```

Метод для полиномиальной функции 2 степени:

```
def squared(coordinates):
    try:
        x_coord = coordinates[0]
        y_coord = coordinates[1]
        matrix = [[sum([x ** 4 for x in x_coord]), sum([x ** 3 for x in x_coord])
            , sum([x ** 2 for x in x_coord]),
                   sum([x ** 2 * y for x, y in zip(x_coord, y_coord)])],
                  [sum([x ** 3 for x in x_coord]), sum([x ** 2 for x in x_coord])
                      , sum(x_coord),
                   sum([x * y for x, y in zip(x_coord, y_coord)])],
                  [sum([x ** 2 for x in x_coord]), sum(x_coord), len(x_coord),
                      sum(y_coord)]]
        a, b, c = solve_matrix(matrix)
        f = [a*x ** 2 + b * x + c for x in x_coord]
        if a == 0:
            return '1'
        eps, delta, S, SS_total, R_squared = aprox_staff_computing(x_coord,
            y_coord, f)
        return a, b, c, cof_cor(x_coord, y_coord), S, delta, R_squared, x_coord,
            y_coord, f, eps
    except BaseException:
        return ['']
```

Метод для полиномиальной функции 3 степени:

```
def triple(coordinates):
    try:
        x_coord = coordinates[0]
        y_coord = coordinates[1]
        matrix = [[sum([x ** 6 for x in x_coord]), sum([x ** 5 for x in x_coord])
            , sum([x ** 4 for x in x_coord]),
                   sum([x ** 3 for x in x_coord]), sum([x ** 3 * y for x, y in
                       zip(x_coord, y_coord)])],
                  [sum([x ** 5 for x in x_coord]), sum([x ** 4 for x in x_coord])
                      , sum([x ** 3 for x in x_coord]),
                   sum([x ** 2 for x in x_coord]), sum([x ** 2 * y for x, y in
                       zip(x_coord, y_coord)])],
                  [sum([x ** 4 for x in x_coord]), sum([x ** 3 for x in x_coord])
                      , sum([x ** 2 for x in x_coord]),
                   sum(x_coord), sum([x * y for x, y in zip(x_coord, y_coord)])],
```

```
                    [sum([x ** 3 for x in x_coord]), sum([x ** 2 for x in x_coord]), sum(x_coord), len(x_coord),
                     sum(y_coord)]]
        a, b, c, d = solve_matrix(matrix)
        if a == 0:
            return ['1',b]
        f = [a*x**3+b*x**2+c*x+d for x in x_coord]
        eps, delta, S, SS_total, R_squared = aprox_staff_computing(x_coord, y_coord, f)
        return a, b, c, d, cof_cor(x_coord, y_coord), S, delta, R_squared, x_coord, y_coord, f, eps
    except BaseException:
        return ''
```

Метод для экспоненциальной функции:

```
def exponential(coordinates):
    try:
        x_coord = coordinates[0]
        y_coord = coordinates[1]
        if check_less_zero(y_coord):
            y_coord_log = [math.log(y) for y in y_coord]
        else:
            return ['']
        b, a = lin_approx(x_coord, y_coord_log)
        a = math.exp(a)
        f = [a * (math.exp(x*b)) for x in x_coord]
        eps, delta, S, SS_total, R_squared = aprox_staff_computing(x_coord, y_coord, f)
        return a, b, cof_cor(x_coord, y_coord), S, delta, R_squared, x_coord, y_coord, f, eps
    except BaseException:
        return ['']
```

Метод для логарифмической функции:

```
def logarithm(coordinates):
    try:
        x_coord = coordinates[0]
        y_coord = coordinates[1]
        if check_less_zero(x_coord):
            x_coord_log = [math.log(x) for x in x_coord]
        else:
            return ['']
        a, b = lin_approx(x_coord_log, y_coord)
        f = [a * x + b for x in x_coord_log]
        eps, delta, S, SS_total, R_squared = aprox_staff_computing(x_coord, y_coord, f)
        return a, b, cof_cor(x_coord, y_coord), S, delta, R_squared, x_coord, y_coord, f, eps
    except BaseException:
        return ['']
```

Метод для степенной функции:

```
def power(coordinates):
    try:
        x_coord = coordinates[0]
        y_coord = coordinates[1]
        if check_less_zero(x_coord) and (check_less_zero(y_coord)):
            x_coord_log = [math.log(x) for x in x_coord]
```

```
            y_coord_log = [math.log(y) for y in y_coord]
        else:
            return ['']
        b, a = lin_approx(x_coord_log, y_coord_log)
        a = math.exp(a)
        f = [a*(x**b) for x in x_coord]
        eps, delta, S, SS_total, R_squared = aprox_staff_computing(x_coord,
            y_coord, f)
        return a, b, cof_cor(x_coord, y_coord), S, delta, R_squared, x_coord,
            y_coord, f, eps
    except BaseException:
        return ['']
```

6 Результаты выполнения программы

6.1 Пример 1

Входные данные:
2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 9
Результат:

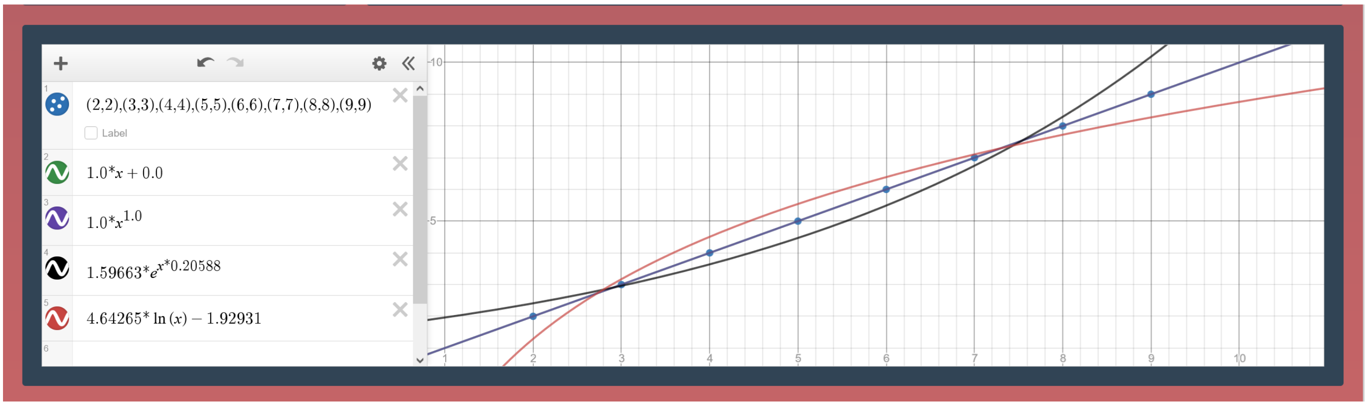

Рис. 2: Пример 1

Линейная аппроксимация:
$f = 1.0 * x + 0.0$
Коэффициент корреляции Пирсона r = 1.0
Мера отклонения = 0.0
СКО = 0.0
Коэффициент детерминации $R^2 = 1.0$
Высокая точность аппроксимации (модель хорошо описывает явление)
$fi(x_i)$:
2.03.04.0
5.06.07.0
8.09.0
eps$_i$:

0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0

Квадратичная аппроксимация:
Квадратичная аппроксимация сводится к линейной

Кубическая аппроксимация:
Кубическая аппроксимация сводится к линейной

Степенная аппроксимация:
$f = 1.0 * x^1.0$
Мера отклонения $= 0.0$
СКО $= 0.0$
Коэффициент детерминации $R^2 = 1.0$
Высокая точность аппроксимации (модель хорошо описывает явление)
$fi(x_i)$:
2.0 3.0 4.0
5.0 6.0 7.0
8.0 9.0
eps_i:
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0

Экспоненциальная аппроксимация:
$f = 1.59663 * e^(x * 0.20588)$
Мера отклонения $= 2.390162793685118$
СКО $= 0.5465988924345162$
Коэффициент детерминации $R^2 = 0.9430974347729265$
Удовлетворительная аппроксимация (модель в целом адекватно описывает явление)
$f_i(x_i)$:
2.41006 2.96101 3.63791
4.46955 5.49132 6.74666
8.28898 10.18388
eps_i:
-0.41006 0.03899 0.36209
0.53045 0.50868 0.25334
-0.28898 -1.18388

Логарифмическая аппроксимация:
$f = 4.64265 * ln(x) - 1.92931$

Мера отклонения = 1.8553133789257865
СКО = 0.48157467994665515
Коэффициент детерминации R^2 = 0.9558258719303384
Высокая точность аппроксимации (модель хорошо описывает явление)
$fi(x_i)$:
1.28874 3.17117 4.50678
5.54276 6.38921 7.10488
7.72482 8.27165
eps_i:
0.71126 -0.17117 -0.50678
-0.54276 -0.38921 -0.10488
0.27518 0.72835

6.2 Пример 2

Входные данные:
0.5 1 1.5 2 2.5 3 3.5 4 4.5 5
0.25 1 2.25 4 6.25 9 12.25 16 20.25 25
Результат:

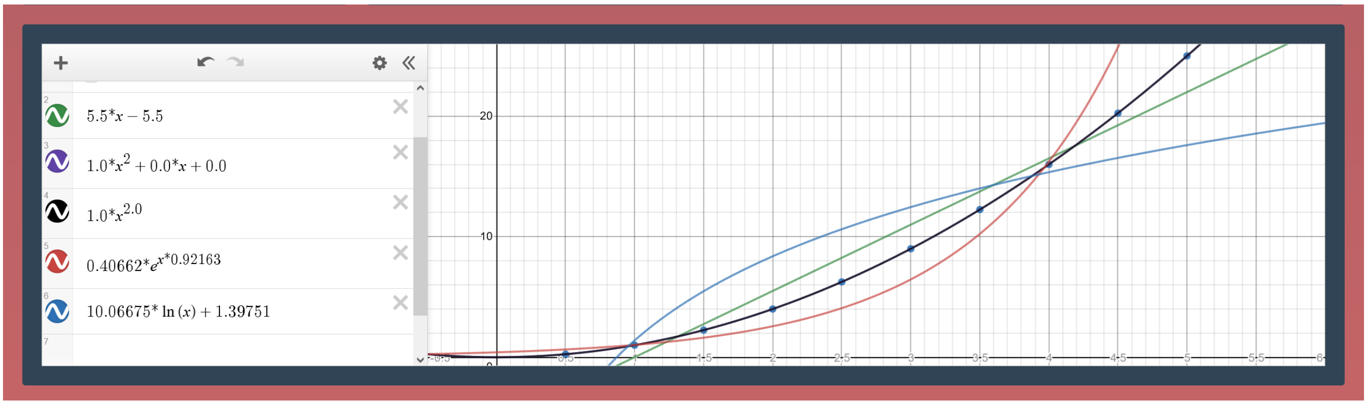

Рис. 3: Пример 2

Линейная аппроксимация:
$f = 5.5 * x - 5.5$
Коэффициент корреляции Пирсона r = 0.97456
Мера отклонения = 32.99999999999999
СКО = 1.8165902124584947
Коэффициент детерминации R^2 = 0.9497645211930926
Удовлетворительная аппроксимация (модель в целом адекватно описывает явление)
$fi(x_i)$:
-2.75 0.0 2.75
5.5 8.25 11.0

13.75 16.5 19.25
22.0
eps_i :
3.0 1.0 -0.5
-1.5 -2.0 -2.0
-1.5 -0.5 1.0
3.0

Квадратичная аппроксимация:
$f = 1.0 * x^2 + 0.0 * x + 0.0$
Мера отклонения = 0.0
СКО = 0.0
Коэффициент детерминации $R^2 = 1.0$
Высокая точность аппроксимации (модель хорошо описывает явление)
$fi(x_i)$:
0.25 1.0 2.25
4.0 6.25 9.0
12.25 16.0 20.25
25.0
eps_i :
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0

Кубическая аппроксимация:
Кубическая аппроксимация сводится к квадратической

Степенная аппроксимация:
$f = 1.0 * x^2.0$
Мера отклонения = 0.0
СКО = 0.0
Коэффициент детерминации $R^2 = 1.0$
Высокая точность аппроксимации (модель хорошо описывает явление)
$fi(x_i)$:
0.25 1.0 2.25
4.0 6.25 9.0
12.25 16.0 20.25
25.0
eps_i :
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0

0.0

Экспоненциальная аппроксимация:
$f = 0.40662 * e^(x * 0.92163)$
Мера отклонения = 296.9789990072679
СКО = 5.4495779562023685
Коэффициент детерминации R^2 = 0.5594244323990236
Слабая аппроксимация (модель слабо описывает явление)
$fi(x_i)$:
0.64463 1.02198 1.62021
2.56862 4.07219 6.4559
10.23495 16.22612 25.72429
40.78235
eps_i :
-0.39463 -0.02198 0.62979
1.43138 2.17781 2.5441
2.01505 -0.22612 -5.47429
-15.78235

Логарифмическая аппроксимация:
$f = 10.06675 * \ln(x) + 1.39751$
Мера отклонения = 166.8374451954847
СКО = 4.084573970385219
Коэффициент детерминации R^2 = 0.7460254865964745
Слабая аппроксимация (модель слабо описывает явление)
$fi(x_i)$:
-5.58023 1.39751 5.47922
8.37524 10.62157 12.45696
14.00876 15.35298 16.53867
17.59931
eps_i : 5.83023 -0.39751 -3.22922
-4.37524 -4.37157 -3.45696
-1.75876 0.64702 3.71133
7.40069

7 Вывод

В рамках данной лабораторнои работы были исследованы различные методы аппроксимации функции, такие как линеиная, квадратичная, кубическая, экспоненциальная, логарифмическая и степенная аппроксимации. Каждыи из этих методов позволяет наити функцию, которая наилучшим образом приближает набор данных.

1. Линейная аппроксимация: приближает функцию прямой линией $y = ax + b$, где a и b коэффициенты. Этот метод подходит для данных, которые имеют линейную зависимость.

2. Квадратичная аппроксимация: использует параболу $y = ax^2 + bx + c$ для приближения функции. Этот метод подходит для данных, которые имеют выпуклую или вогнутую форму.

3. Кубическая аппроксимация: использует кубическую функцию $y = ax^3 + bx^2 + cx + d$ для приближения функции.

4. Экспоненциальная аппроксимация: использует функцию вида $y = ae^{bx}$ для приближения функции. Этот метод подходит для данных, которые растут или убывают экспоненциально.

5. Логарифмическая аппроксимация: использует функцию вида $y = a\ln(x) + b$ для приближения функции. Этот метод подходит для данных, которые имеют логарифмическую зависимость.

6. Степенная аппроксимация: использует функцию вида $y = ax^b$ для приближения функции. Этот метод подходит для данных, которые имеют степенную зависимость.