# Практическая работа №8 Обработка экспериментальных данных

# Цели практической работы:

1. Построение Компьютерной математической модели.

2. Закрепление умений строить аналитичсекие модели на основе экспериментальных данных.

Рассматриваемые вопросы:

1. Отработка навыка построения компьютерной модели на основе экспериментальных данных

## Методические указания

## Практикум

### Задание 4

Процесс y=f(x) задан таблицей

|  |  |  |
| --- | --- | --- |
| **№** | **xi** | **yi** |
|  | 15 | 150 |
|  | 14 | 145 |
|  | 12 | 120 |
|  | 10 | 100 |
|  | 9 | 95 |
|  | 8 | 75 |
|  | 7 | 70 |
|  | 5 | 55 |

Разработайте **две** программы – компьютерные математические модели на основе экспериментальных данных.

1. Найдите коэффициенты модели по МЕТОДУ КРАМЕРА.
2. Найдите коэффициенты модели по МЕТОДУ НАИМЕНЬШИХ КВАДРАТОВ (МНК).
3. Запрограммируйте на знакомом для вас языке программирования алгоритм решения задачи
4. Разместите программный код в отчете
5. РАЗМЕСТИТЕ ОТЧЕТ В ЛИЧНОМ КАБИНЕТЕ ПРЕПОДАВАТЕЛЯ.

## Формат отчета

Второй лист

Задание № 4.

### I этап: Вербальное описание задачи

### II этап: Построение математической модели

### III этап: ПРОГРАММНЫЙ КОД –

1. Метод КРАМЕРА
2. МЕТОД НАИМЕНЬШИХ КВАДРАТОВ (МНК)

### IV этап: Выводы

**public class CramerMethod {**

**public static void main(String[] args) {**

**double[][] coefficients = {{2, -1, 3}, {1, 2, 1}, {3, 1, 4}};**

**double[] constants = {1, 3, 7};**

**double[] solutions = solveCramer(coefficients, constants);**

**if (solutions != null) {**

**for (int i = 0; i < solutions.length; i++) {**

**System.out.println("x" + (i + 1) + " = " + solutions[i]);**

**}**

**} else {**

**System.out.println("Метод Крамера не может быть применен.");**

**}**

**}**

**public static double[] solveCramer(double[][] coefficients, double[] constants) {**

**int n = coefficients.length;**

**double[] solutions = new double[n];**

**double determinantA = determinant(coefficients);**

**if (determinantA == 0) {**

**return null; // Метод Крамера не может быть применен, если определитель A равен нулю.**

**}**

**for (int i = 0; i < n; i++) {**

**double[][] tempMatrix = new double[n][n];**

**for (int j = 0; j < n; j++) {**

**System.arraycopy(coefficients[j], 0, tempMatrix[j], 0, n);**

**}**

**for (int j = 0; j < n; j++) {**

**tempMatrix[j][i] = constants[j];**

**}**

**solutions[i] = determinant(tempMatrix) / determinantA;**

**}**

**return solutions;**

**}**

**public static double determinant(double[][] matrix) {**

**int n = matrix.length;**

**if (n == 1) {**

**return matrix[0][0];**

**}**

**double det = 0;**

**for (int i = 0; i < n; i++) {**

**double[][] subMatrix = new double[n - 1][n - 1];**

**for (int j = 1; j < n; j++) {**

**int k = 0;**

**for (int l = 0; l < n; l++) {**

**if (l != i) {**

**subMatrix[j - 1][k] = matrix[j][l];**

**k++;**

**}**

**}**

**}**

**det += (i % 2 == 0 ? 1 : -1) \* matrix[0][i] \* determinant(subMatrix);**

**}**

**return det;**

**}**

**}  
  
  
  
import org.apache.commons.math3.linear.Array2DRowRealMatrix;**

**import org.apache.commons.math3.linear.ArrayRealVector;**

**import org.apache.commons.math3.linear.DecompositionSolver;**

**import org.apache.commons.math3.linear.LUDecomposition;**

**public class LeastSquaresMethod {**

**public static void main(String[] args) {**

**double[] x = {1, 2, 3, 4, 5};**

**double[] y = {2, 3, 5, 7, 10};**

**double[] coefficients = solveLeastSquares(x, y);**

**for (int i = 0; i < coefficients.length; i++) {**

**System.out.println("Coefficient " + i + ": " + coefficients[i]);**

**}**

**}**

**public static double[] solveLeastSquares(double[] x, double[] y) {**

**int n = x.length;**

**int degree = 2; // Порядок полинома**

**// Создаем матрицу A и вектор B для метода МНК**

**double[][] A = new double[n][degree + 1];**

**double[] B = new double[n];**

**for (int i = 0; i < n; i++) {**

**for (int j = 0; j <= degree; j++) {**

**A[i][j] = Math.pow(x[i], j);**

**}**

**B[i] = y[i];**

**}**

**Array2DRowRealMatrix matrixA = new Array2DRowRealMatrix(A);**

**ArrayRealVector vectorB = new ArrayRealVector(B);**

**// Решаем систему уравнений методом LU-разложения**

**DecompositionSolver solver = new LUDecomposition(matrixA).getSolver();**

**ArrayRealVector solution = solver.solve(vectorB);**

**return solution.toArray();**

**}**

**}  
  
Обратите внимание, что в коде для метода наименьших квадратов используется библиотека Apache Commons Math (org.apache.commons.math3). Убедитесь, что вы добавили эту библиотеку в свой проект, чтобы код компилировался и работал правильно**