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

Факультет Программной Инженерии и Компьютерной Техники

**Лабораторная работа №1**

**по дисциплине «Вычислительная математика»**

Вариант: **2**

Преподаватель:   
Малышева Татьяна Алексеевна

Выполнил: Барсуков Максим

Группа: Р3215

Санкт-Петербург, 2024 г

# Цель работы

Изучить численные методы решения систем линейных алгебраических уравнений и реализовать один из них средствами программирования.

# Описание метода

Итерационные методы дают возможность для системы (1) построить последовательность векторов 𝑥 (0) , 𝑥 (1) , … , 𝑥 (𝑘) , пределом которой должно быть точное решение 𝑥 (∗) : 𝑥 (∗) = lim 𝑘→∞ 𝑥 (𝑘) Построение последовательности заканчивается, как только достигается желаемая точность.

Приведем систему уравнений, выразив неизвестные 𝑥1, 𝑥2, … , 𝑥𝑛 соответственно из первого, второго и т.д. уравнений системы.

# Листинг программы

[https://github.com/maxbarsukov/itmo/tree/master/4%20%D0%B2%D1%8B%D1%87%D0%BC%D0%B0%D1%82/%D0%BB%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D1%8B%D0%B5/lab1](https://github.com/maxbarsukov/itmo/tree/master/4%20вычмат/лабораторные/lab1)

## SimpleIteration.java

package ru.itmo.computionalmath.lab1.algo;

import ru.itmo.computionalmath.lab1.models.Matrix;

import ru.itmo.computionalmath.lab1.models.Result;

import ru.itmo.computionalmath.lab1.utils.Printer;

import java.util.ArrayList;

import static java.lang.Math.abs;

import static ru.itmo.computionalmath.lab1.utils.Color.\*;

public class SimpleIteration {

  private static final Printer printer = new Printer();

  private static double[][] data;

  private static int permuts;

  public static void compute(Matrix matrix, double eps) {

    printer.printMatrix(matrix);

    printer.println("");

    if (checkDiagonal(matrix.getMatrix(), matrix.getSize())) {

      Result rs = methodOfSimpleIterations(matrix, eps);

      printer.println(rs.getTable());

      printer.printVector(CYAN\_BOLD + "Решение системы: " + RESET, rs.getResult());

      printer.printVector(CYAN\_BOLD + "Вектор невязки: " + RESET, rs.getResiduals());

      return;

    }

    permuteMatrix(matrix, 0);

    printer.println(CYAN\_BOLD + "Перестановок строк: " + RESET + permuts);

    if (data != null) {

      Matrix matrix1 = new Matrix(data);

      printer.println(BLUE + "Матрица после перестановки строк" + RESET);

      printer.printMatrix(matrix1);

      Result rs = methodOfSimpleIterations(matrix1, eps);

      printer.println(rs.getTable());

      printer.printVector(CYAN\_BOLD + "Решение системы: " + RESET, rs.getResult());

      printer.printVector(CYAN\_BOLD + "Вектор невязки: " + RESET, rs.getResiduals());

    } else {

      printer.println(RED + "Отсутствие диагонального преобладания" + RESET);

    }

  }

  public static boolean checkDiagonal(double[][] matrix, int size) {

    int i, j, k = 1;

    double sum;

    for (i = 0; i < size; i++) {

      sum = 0;

      for (j = 0; j < size; j++) {

        sum += abs(matrix[i][j]);

      }

      sum -= abs(matrix[i][i]);

      if (sum >= abs(matrix[i][i])) {

        k = 0;

      }

    }

    return (k == 1);

  }

  private static void permuteMatrix(Matrix matrix, int index) {

    permuts = 0;

    if (index >= matrix.getMatrix().length - 1) {

      if (checkDiagonal(matrix.getMatrix(), matrix.getSize())) {

        data = new double[matrix.getSize()][matrix.getSize() + 1];

        for (int i = 0; i < matrix.getSize(); i++) {

          for (int j = 0; j < matrix.getSize() + 1; j++) {

            data[i][j] = matrix.getMatrix()[i][j];

          }

        }

      }

    } else {

      for (int i = index; i < matrix.getMatrix().length; i++) {

        var t = matrix.getMatrix()[index];

        matrix.getMatrix()[index] = matrix.getMatrix()[i];

        matrix.getMatrix()[i] = t;

        permuteMatrix(matrix, index + 1);

        t = matrix.getMatrix()[index];

        matrix.getMatrix()[index] = matrix.getMatrix()[i];

        matrix.getMatrix()[i] = t;

      }

      permuts++;

    }

  }

  private static Result methodOfSimpleIterations(Matrix matrix, double eps) {

    var rs = new Result();

    var x = new double[matrix.getSize()];

    double norma, sum, t;

    do {

      ArrayList<Double> esps = new ArrayList<>();

      norma = 0;

      for (int i = 0; i < matrix.getSize(); i++) {

        t = x[i];

        sum = 0;

        for (int j = 0; j < matrix.getSize(); j++) {

          if (j != i)

            sum += matrix.getMatrix()[i][j] \* x[j];

        }

        x[i] = (matrix.getVector()[i] - sum) / matrix.getMatrix()[i][i];

        esps.add(abs(x[i] - t));

        if (abs(x[i] - t) > norma) {

          norma = abs(x[i] - t);

        }

      }

      rs.addIter(x);

      rs.addE(esps);

    } while (norma > eps);

    rs.setResult(x);

    ArrayList<Double> residuals = new ArrayList<>();

    for (int i = 0; i < matrix.getSize(); i++) {

      double S = 0;

      for (int j = 0; j < matrix.getSize(); j++) {

        S += matrix.getMatrix()[i][j] \* x[j];

      }

      residuals.add(S - matrix.getVector()[i]);

    }

    rs.setResiduals(residuals);

    return rs;

  }

}

# Примеры работы программы

Вычислительная математика. Барсуков Максим, P3215

Введите:

1 - Ввести матрицу с консоли

2 - Ввести матрицу с файла

3 - Генерация случайной матрицы

1

Вводим матрицу с консоли

Введите размерность матрицы:

3

Введите строки матрицы:

10 2 -1 5

-2 -6 -1 24.4

1 -3 12 36

Введите точность:

0.01

Матрица:

10 2 -1 5

-2 -6 -1 24.4

1 -3 12 36

№ |x1 |x2 |x3 | eps1 | eps2 | eps3 |

0 |0 |0 |0 |

1 |0,500000 |-4,236667 |1,899167 | 0,500000 |4,236667 |1,899167 |

2 |1,537250 |-4,898944 |1,647160 | 1,037250 |0,662278 |0,252007 |

3 |1,644505 |-4,892695 |1,639784 | 0,107255 |0,006250 |0,007376 |

4 |1,642517 |-4,890803 |1,640423 | 0,001987 |0,001892 |0,000639 |

Решение системы:

[1] = 1,642517401620371

[2] = -4,890803167438272

[3] = 1,640422758005401

Вектор невязки:

[1] = 0,003144923321760

[2] = -0,000638556616508

[3] = 0,000000000000000

# Вывод:

В результате выполнения данной лабораторной работой я познакомился с численными методами решения математических задач на примере систем алгебраических уравнений, реализовав на языке программирования Java метод простых итераций.