\*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 12

Выполнил:

Студент группы P3212

Медведев Ярослав Александрович

Преподаватель:

Машина Екатерина Сергеевна

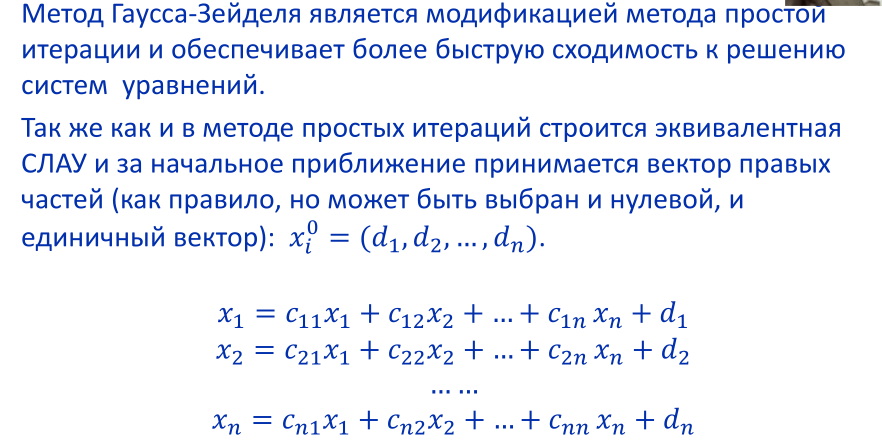
г. Санкт-Петербург

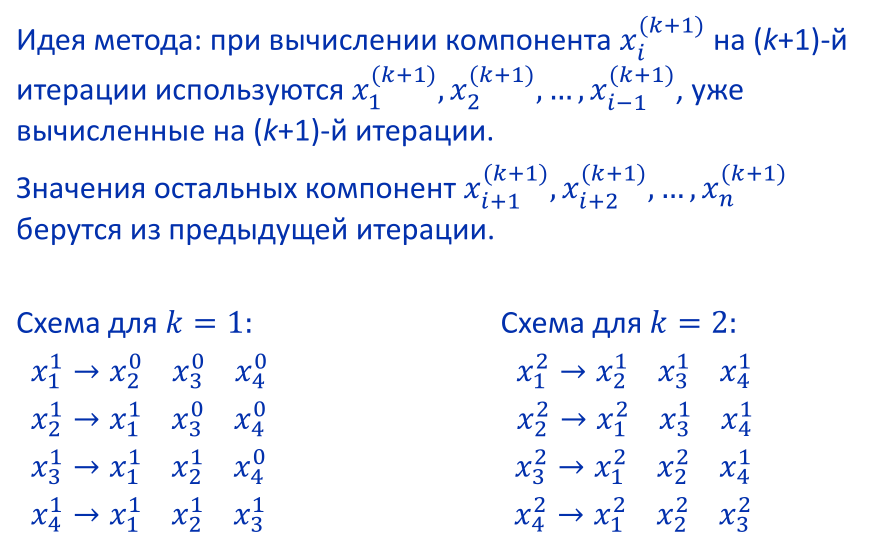
2025

# Цель работы

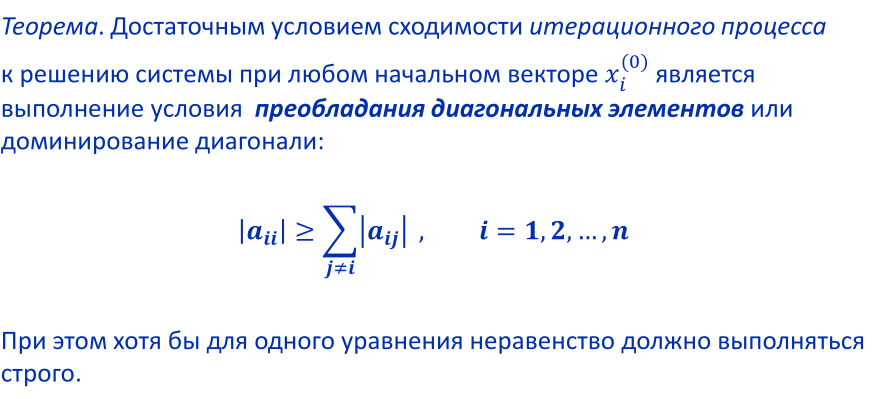
Реализация итерационного метода Гаусса - Зейделя для решения СЛАУ на языке программирования Golang

# Описание метода









# Листинг программы

Полный код программы: <https://github.com/gr1ckly/Computional_Mathematic/tree/master/Lab_1>

### Реализация преобразования для диагонального преобладания и метода Гаусса-Зейделя

type Matrix struct {

size, maxIter int

data [][]float64

vecB []float64

accuracy float64

}

func (matrix \*Matrix) IsDiagonallyDominant() bool {

for i := 0; i < matrix.size; i++ {

sum := 0.0

for j := 0; j < matrix.size; j++ {

if i != j {

sum += math.Abs(matrix.data[i][j])

}

}

if math.Abs(matrix.data[i][i]) <= sum {

return false

}

}

return true

}

func (matrix \*Matrix) MakeDiagonallyDominant() bool {

for i := 0; i < matrix.size; i++ {

maxRow := i

maxVal := math.Abs(matrix.data[i][i])

for k := i + 1; k < matrix.size; k++ {

if math.Abs(matrix.data[k][i]) > maxVal {

maxVal = math.Abs(matrix.data[k][i])

maxRow = k

}

}

if maxRow != i {

matrix.data[i], matrix.data[maxRow] = matrix.data[maxRow], matrix.data[i]

matrix.vecB[i], matrix.vecB[maxRow] = matrix.vecB[maxRow], matrix.vecB[i]

}

}

if matrix.IsDiagonallyDominant() {

return true

} else {

return false

}

}

func (matrix \*Matrix) GaussSeidel() ([]float64, int, []float64, error) {

x := make([]float64, matrix.size)

for idx, \_ := range x {

x[idx] = matrix.vecB[idx] / matrix.data[idx][idx]

}

errors := make([]float64, matrix.size)

for iter := 0; iter < matrix.maxIter; iter++ {

maxDiff := 0.0

for i := 0; i < matrix.size; i++ {

sum := 0.0

for j := 0; j < matrix.size; j++ {

if j != i {

sum += matrix.data[i][j] \* x[j]

}

}

newX := float64(matrix.vecB[i]-sum) / float64(matrix.data[i][i])

errors[i] = math.Abs(newX - x[i])

if errors[i] > maxDiff {

maxDiff = errors[i]

}

x[i] = newX

}

if maxDiff <= matrix.accuracy {

return x, iter + 1, errors, nil

}

}

return nil, matrix.maxIter, errors, fmt.Errorf("Решения за %d итераций при указанной точности не найдено\n", matrix.maxIter)

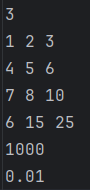
}

# Примеры работы программы

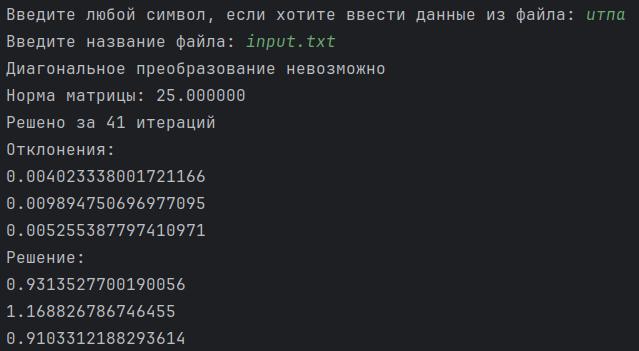
### Работа программы при вводе данных через консоль для матрицы с диагональным преобладанием:

### Работа программы при вводе из файла input.txt матрицы без диагонального преобладания:

Содержимое файла:



Работа программы:



# Вывод

В процессе выполнения данной лабораторной работы я смог реализовать итерационный метод Гаусса - Зейделя для решения СЛАУ на языке программирования Golang.