Университет ИТМО МФ КТиУ, Ф ПИиКТ

**Лабораторная работа №1**

**Дисциплина «Вычислительная математика»**

**Решение системы линейных алгебраических**

**уравнений СЛАУ**

**Выполнил:** Галлямов Камиль Рустемович

**Преподаватель:**

Машина Екатерина Алексеевна

г. Санкт-Петербург 2024 г.

# Описание метода

Основан на приведении матрицы системы к треугольному виду так, чтобы ниже ее главной диагонали находились только нулевые элементы.

Прямой ход метода Гаусса состоит в последовательном исключении неизвестных из уравнений системы. Сначала с помощью первого уравнения исключается 𝑥1 из всех последующих уравнений системы. Затем с помощью второго уравнения исключается 𝑥2 из третьего и всех последующих уравнений и т.д. Этот процесс продолжается до тех пор, пока в левой части последнего (n-го) уравнения не останется лишь один член с неизвестным 𝑥𝑛, т. е. матрица системы будет приведена к треугольному виду.

Обратный ход метода Гаусса состоит в последовательном вычислении искомых неизвестных: решая последнее уравнение, находим неизвестное 𝑥𝑛. Далее, из предыдущего уравнения вычисляем 𝑥𝑛−1 и т. д. Последним найдем 𝑥1 из первого уравнения. Метод имеет много различных вычислительных схем. Основное требование - det A ≠ 0.

# Листинг программы (python)

**from** **random** **import** randint

**def** **get\_random\_matrix**(n):

mn = -**1000**

mx = **1000**

a = [

[randint(mn, mx) **for** \_\_ **in** range(n)]

**for** \_ **in** range(n)

]

b = [randint(mn, mx) **for** \_\_ **in** range(n)]

**return** a, b

**def** **get\_data**():

random\_word = 'RANDOM'

sn = input('введите размерность матрицы (n): ').strip()

**while** **not** sn.isdigit() **or** int(sn) < **1** **or** int(sn) > **20**:

**print**('-' \* **50**)

**print**('n должно быть числом на отрезке [1;20]')

sn = input('введите размерность матрицы (n): ').strip()

n = int(sn)

rw = input(f'введите "{random\_word}" для генерации рандомной матрицы (A и B): ')

**if** rw == 'RANDOM':

a, b = get\_random\_matrix(n)

**print**('Сгенерированная матрица: ')

**for** i **in** range(n):

**print**(\*a[i], '|', b[i], sep='**\t**')

**print**('-' \* **50**)

**else**:

**print**('введите элементы матрицы A:')

a = []

**for** i **in** range(n):

fl = False

**while** **not** fl:

**try**:

**print**(f'Введите строчку ({i}) (n чисел): ')

cur = list(map(float, input().split()))

**assert** len(cur) == n

fl = True

**except** **Exception**:

**pass**

a.append(cur)

**print**('введите элементы матрицы B:')

fl = False

**while** **not** fl:

**try**:

**print**('Введите строчку (n чисел): ')

b = list(map(float, input().split()))

**assert** len(b) == n

fl = True

**except** **Exception**:

**pass**

**return** n, a, b

**def** **get\_data\_from\_file**():

**with** open('input.txt') **as** f:

n = int(f.readline())

f.readline()

a = [

list(map(float, f.readline().split()))

**for** \_ **in** range(n)

]

f.readline()

b = list(map(float, f.readline().split()))

**return** n, a, b

**def** **get\_determinant**(triangular\_matrix, k):

res = **1**

**for** i **in** range(len(triangular\_matrix)):

res \*= triangular\_matrix[i][i]

**return** (-**1**) \*\* k \* res

**def** **get\_solution\_and\_k**(a, b):

x = [None] \* n

k\_ = **0**

**for** i **in** range(**0**, n - **1**):

**while** a[i][i] == **0**:

a = a[:i] + a[i + **1**:] + [a[i]]

b = b[:i] + b[i + **1**:] + [b[i]]

k\_ += **1**

**for** k **in** range(i + **1**, n):

c = a[k][i] / a[i][i]

a[k][i] = **0**

**for** j **in** range(i + **1**, n):

a[k][j] = a[k][j] - c \* a[i][j]

b[k] = b[k] - c \* b[i]

**for** i **in** range(n - **1**, -**1**, -**1**):

s = **0**

**for** j **in** range(i + **1**, n):

s = s + a[i][j] \* x[j]

x[i] = (b[i] - s) / a[i][i]

**return** x, k\_, a, b

**def** **get\_r**(a, b, x):

n = len(a)

r = [**0**] \* n

**for** i **in** range(n):

**for** j **in** range(n):

r[i] += a[i][j] \* x[j]

r[i] -= b[i]

**return** r

n, a, b = get\_data\_from\_file()

first\_a = [el[:] **for** el **in** a]

first\_b = b[:]

x, k, a, b = get\_solution\_and\_k(a, b)

**print**('Определитель: ', get\_determinant(a, k))

**print**('Треугольная матрица: ')

**for** i **in** range(n):

**print**(\*a[i], '|', b[i], sep='**\t**')

**print**('-' \* **50**)

**print**('Вектор неизвестных: ', \*map(**lambda** xi: round(xi, **5**), x))

**print**('Вектор невязок: ', \*get\_r(first\_a, first\_b, x))

# Блок-схема численного метода



**Тестовые данные**

Тест 1:

3

1 3 5

1 3 6

9 8 5

1 2 0

Определитель: 19.0

Треугольная матрица:

1.0 3.0 5.0 | 1.0

0 -19.0 -40.0 | -9.0

0 0 1.0 | 1.0

--------------------------------------------------

Вектор неизвестных: 0.89474 -1.63158 1.0

Вектор невязок: 0.0 0.0 -1.7763568394002505e-15

# Вывод

В ходе выполнения лабораторной работы я вспомнил теоретическую базу про слау, познакомился с численными методами, реализовал метод Гаусса на языке программирования python.

Метод Гаусса: точный метод, но при маленьких значениях имеется вычислительная погрешность.

Метод Гаусса с выбором главного элемента: точный метод, избегается вычислительная погрешность для маленьких значений, модификация метода Гаусса.

Метод простой итерации: итерационный метод, универсальный, медленная скорость сходимости.

Метод Гаусса-Зейделя: итерационный метод, универсальный, модификация простой итерации, более быстрая скорость сходимости.