РГПУ им. А.И. Герцена

**Отчет по лабораторной работе №3**

“Численные методы решения систем линейных алгебраических уравнений. Метод треугольной факторизации”

Работу выполнили: Иванова Мария Алексеевна

Буряков Иван Олегович

Чернышева Виктория Викторовна

Егор Собинин Яковлевич

Факультет 2об-ИВТ

Группа 1 гр. 2 п.гр

# **Цель лабораторной работы:**

Научиться реализовывать решение систем линейных алгебраических уравнений методом треугольной факторизации посредством языка программирования.

# **Используемое оборудование:**

ПК, язык программирования Python.

# **Постановка задачи:**

Разработать программу для решения систем линейных уравнений методом треугольной факторизации.

# **Результат выполненной работы:**

## Отчет Ивановой Марии:

Для начала работы необходимо реализовать ввод и вывод матрицы.

Далее разрабатываем программу используя материалы лекции Метод треугольной факторизации.

def matrixInput(n):

a = []

print('Вводите элементы ряда через пробел ')

for i in range(n):

ai = list(map(float, input('Элементы ряда: ').split()))

a.append(ai)

print()

return a

def matrixOutput(n, a):

for i in range(n):

print(a[i])

print()

def metod(n, a):

print('Метод треугольной факторизации')

print('Мatrix A')

matrixOutput(n, a)

L = [([0] \* n) for i in range(n + 1)]

R = [([0] \* n) for i in range(n + 1)]

for i in range(n):

L[i][0] = a[i][0]

R[i][i] = 1

for j in range(1, n):

R[0][j] = round(a[0][j]/L[0][0], 2)

for i in range(1, n):

for t in range(i, n):

sum = 0

sumL = 0

sumR = 0

for j in range(i):

sum += L[i][j]\*R[j][i]

sumL += L[t][j]\*R[j][i]

sumR += L[i][j]\*R[j][t]

L[i][i] = round(a[i][i] - sum, 2)

L[t][i] = round(a[t][i] - sumL, 2)

R[i][t] = round((a[i][t] - sumR) / L[i][i], 2)

print('Мatrix L')

matrixOutput(n, L)

print('Мatrix R')

matrixOutput(n, R)

Z = [0] \* n

X = [0] \* n

Z[0] = round(a[0][-1] / L[0][0], 2)

for i in range(1, n):

sum = 0

for k in range(i):

sum += L[i][k] \* Z[k]

Z[i] = round((a[i][-1] - sum) / L[i][i], 2)

X[n - 1] = Z[n - 1]

for i in reversed(range(n - 1)):

sum = 0

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

sum += R[i][j] \* X[j]

X[i] = Z[i] - sum

print('Мatrix Z')

print(Z)

print('\nПолучим результат:')

print(X)

def main():

n = int(input('Введите кол-во строк: '))

myA = matrixInput(n)

metod(n, myA)

main()

## Отчет Чернышевой Виктории:

Рассмотрим код программы:  
1) Введенные исходные данные - строки 1-14 (функции ввода и вывода матрицы)  
2) Организация этапа прямого хода - строки 17-63  
3) Организация этапа обратного хода и его вывод - строки 65-76

def matrixInput(n):

a = []

print('Вводите элементы ряда через пробел ')

for i in range(n):

ai = list(map(float, input('Элементы ряда: ').split()))

a.append(ai)

print()

return a

def matrixOutput(n, a):

for i in range(n):

print(a[i])

print()

def metod(n, a):

print('Метод треугольной факторизации')

print('Мatrix A')

matrixOutput(n, a)

L = [([0] \* n) for i in range(n + 1)]

R = [([0] \* n) for i in range(n + 1)]

for i in range(n):

L[i][0] = a[i][0]

R[i][i] = 1

for j in range(1, n):

R[0][j] = round(a[0][j]/L[0][0], 2)

for i in range(1, n):

for t in range(i, n):

sum = 0

sumL = 0

sumR = 0

for j in range(i):

sum += L[i][j]\*R[j][i]

sumL += L[t][j]\*R[j][i]

sumR += L[i][j]\*R[j][t]

L[i][i] = round(a[i][i] - sum, 2)

L[t][i] = round(a[t][i] - sumL, 2)

R[i][t] = round((a[i][t] - sumR) / L[i][i], 2)

print('Мatrix L')

matrixOutput(n, L)

print('Мatrix R')

matrixOutput(n, R)

Z = [0] \* n

X = [0] \* n

Z[0] = round(a[0][-1] / L[0][0], 2)

for i in range(1, n):

sum = 0

for k in range(i):

sum += L[i][k] \* Z[k]

Z[i] = round((a[i][-1] - sum) / L[i][i], 2)

X[n - 1] = Z[n - 1]

for i in reversed(range(n - 1)):

sum = 0

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

sum += R[i][j] \* X[j]

X[i] = Z[i] - sum

print('Мatrix Z')

print(Z)

print('\nПолучим результат:')

print(X)

def main():

n = int(input('Введите кол-во строк: '))

myA = matrixInput(n)

metod(n, myA)

main()

Алгоритмы метода треугольной факторизации основаны на методе исключений Гаусса.  
  
Важные черты данного метода:

* Легко вычисляется определитель матрицы **А**.
* Если требуется найти решение для другого вектора **В** в правой части, то не нужно повторно проводить LU-разложение, а достаточно выполнить прямую и обратную подстановки.
* Как и метод Гаусса, метод треугольной факторизации чувствителен к появлению нулей на главной диагонали.
  1. Отчет Собинина Егора

Как и в предыдущих работах сперва организуем ввод и вывод матриц. После на основе лекций и предоставленного кода реализуем решение с помощью метода треугольной факторизации.

def matrixInput(n):

a = []

print('Вводите элементы ряда через пробел ')

for i in range(n):

ai = list(map(float, input('Элементы ряда: ').split()))

a.append(ai)

print()

return a

def matrixOutput(n, a):

for i in range(n):

print(a[i])

print()

def metod(n, a):

print('Метод треугольной факторизации')

print('Мatrix A')

matrixOutput(n, a)

L = [([0] \* n) for i in range(n + 1)]

R = [([0] \* n) for i in range(n + 1)]

for i in range(n):

L[i][0] = a[i][0]

R[i][i] = 1

for j in range(1, n):

R[0][j] = round(a[0][j]/L[0][0], 2)

for i in range(1, n):

for t in range(i, n):

sum = 0

sumL = 0

sumR = 0

for j in range(i):

sum += L[i][j]\*R[j][i]

sumL += L[t][j]\*R[j][i]

sumR += L[i][j]\*R[j][t]

L[i][i] = round(a[i][i] - sum, 2)

L[t][i] = round(a[t][i] - sumL, 2)

R[i][t] = round((a[i][t] - sumR) / L[i][i], 2)

print('Мatrix L')

matrixOutput(n, L)

print('Мatrix R')

matrixOutput(n, R)

Z = [0] \* n

X = [0] \* n

Z[0] = round(a[0][-1] / L[0][0], 2)

for i in range(1, n):

sum = 0

for k in range(i):

sum += L[i][k] \* Z[k]

Z[i] = round((a[i][-1] - sum) / L[i][i], 2)

X[n - 1] = Z[n - 1]

for i in reversed(range(n - 1)):

sum = 0

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

sum += R[i][j] \* X[j]

X[i] = Z[i] - sum

print('Мatrix Z')

print(Z)

print('\nПолучим результат:')

print(X)

def main():

n = int(input('Введите кол-во строк: '))

myA = matrixInput(n)

metod(n, myA)

main()

# Преимуществами являются: простота нахождения определителя А, простое решение для другого вектора В.

Недостатком является чувствительность к появлению значений “0” на главной диагонали матрицы.

# **Заключение:** Мы проверили правильность работы программ из задач на конкретных примерах вводимых данных.

| Вводимые данные | Метод треугольной факторизации |
| --- | --- |
| [[5, 7, 6, 5, 23],  [7, 10, 8, 7, 32],  [6, 8, 10, 9, 33],  [5, 7, 9, 10, 31]] | [1.0, 1.0, 1.0, 1.0] |

Программа выводит ожидаемый нами результат, соответствующий предыдущим методам.