НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

про виконання лабораторної роботи №1

по дисципліні «Алгоритми і системи комп’ютерної математики - 1 »

на тему

«Розв’язання систем лінійних рівнянь»

Виконав: Перевірив:

студент групи КМ-73 асистент кафедри ПМА

Звягін М.С. Дрозденко О.М.

Київ — 2020

Зміст

1 Вступ 3

2 Вимоги до ПЗ 4

3. Опис методів 5

4. Висновки 7

5. Демонстрація роботи програми 8

6. Додаток А. Графи потоку керування 10

7. Додаток Б. Лістинг роботи програми 11

1 Вступ

Мета виконання цієї лабораторної роботи полягає у написанні та тестуванні програмного забезпечення на мовах програмування Python та Octave, що здатне розв’язувати системи лінійних алгебраїчних рівнянь, розмірності від «4 на 4» (згідно з варіантом).

На мові Python та в Octave були реалізовані: схема Гауса та метод Крамера.Графи потоку керування наведено в додатку А.

Код кожної програми наведено у додатку Б.

2 Вимоги до ПЗ.

Програмне забезпечення повинно виконувати наступні функції:

1. Розв’язувати лінійне рівняння відповідно до варіанту методом Крамера та методом Гаусса за допомогою мови програмування Python.

2.Розв’язувати лінійне рівняння відповідно до варіанту методом Крамера та методом Гаусса за допомогою Octave.

3.Програмне забезпечення повинно валідувати вхідні дані, тобто не допускати уведення наприклад пустого значення, строк, та інших даних, які можуть не допустити якісної роботи програми.

3 Опис методів

Усі описані далі методи є прямим, тобто, за відсутності похибок округлень, вони дають точний розв’язок

**Схема Гауса**

У класичному варіанті складається з двох кроків: прямого і оберненого ходу.

Прямий хід: Шляхом елементарних перетворень рядків (додавань до рядка іншого рядка, помноженого на число, і переставлянь рядків) матрицю приводять до верхньотрикутного вигляду (сходинчастого вигляду).

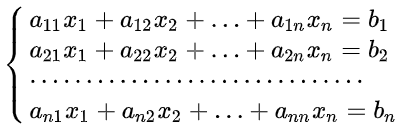
З цього моменту починається зворотний хід. З останнього ненульового рівняння виражають кожну з базисних змінних через небазисні й підставляють до попередніх рівнянь. Повторюючи цю процедуру для всіх базисних змінних, отримують фундаментальний розв'язок.

У класичному варіанті складається з двох кроків: прямого і оберненого ходу.

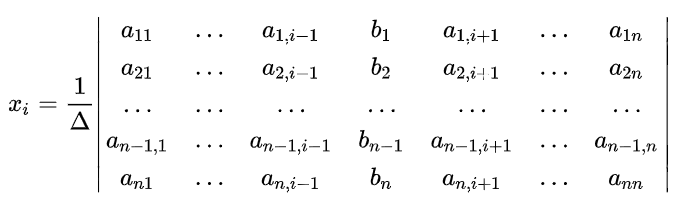
**Метод Крамера**

Суть методу полягає у наступному:

Для системи n лінійних рівнянь з n невідомими

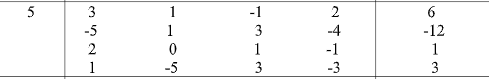


з визначником матриці системи , що не рівний нулю, розв'язок записується у такому вигляді:



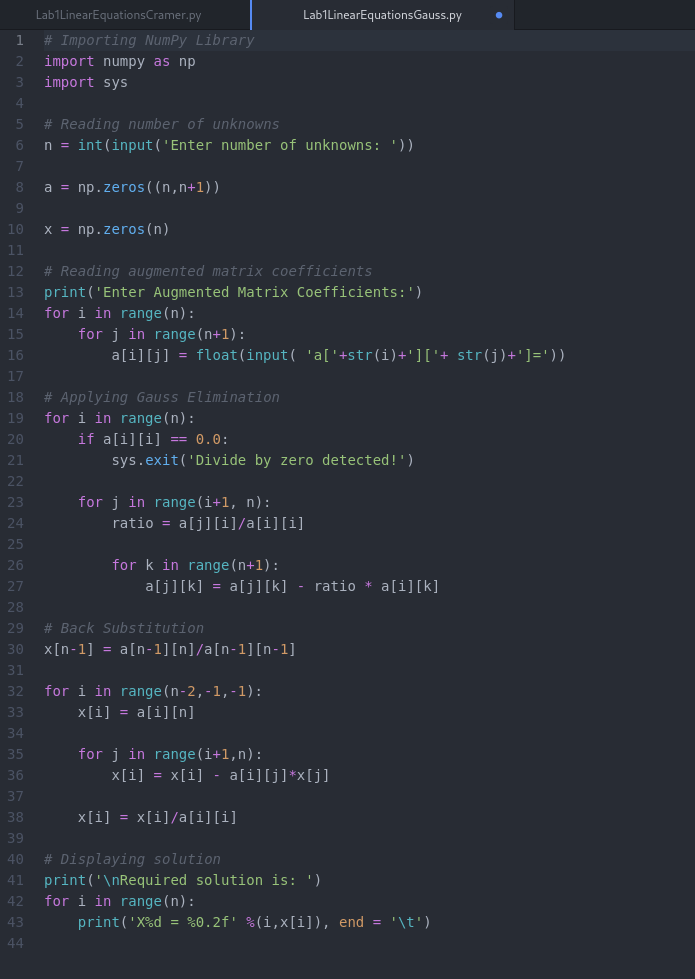
Висновки

Насамкінець, продемонструємо роботу усіх чотирьох підпрограм на індивідуальному завданні для варіанту №5, а саме системи:



Перевіримо, чи розв’язують розроблені програми цю систему правильно.

Зазначимо ще раз, що усі ці методи є прямим, тобто похибка методу у результуючій похібці – відсутня. Таким чином, похибка, яку ми отримаємо, після виконання програм повинна бути досить малою (співрозмірна до похибки округлень мови програмування).

Схема Гауcса 

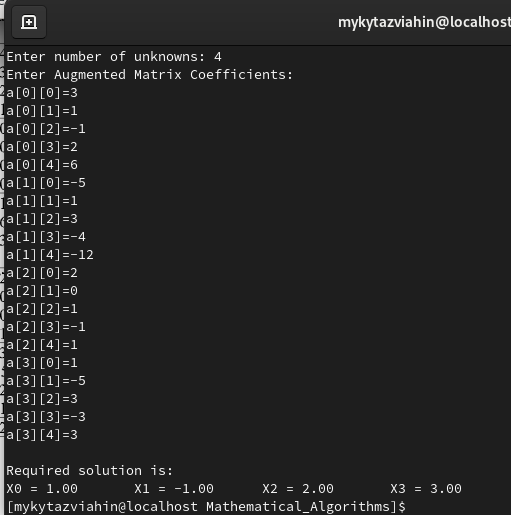
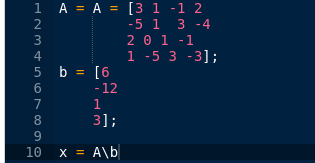
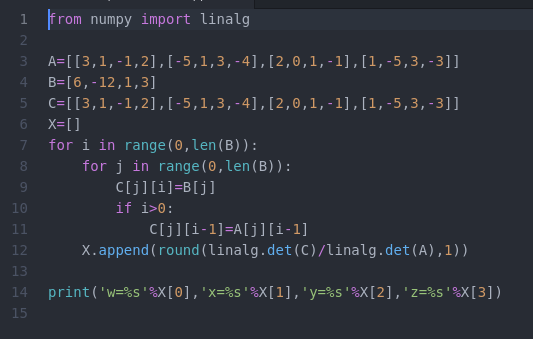


Схема Гаусса в Octave



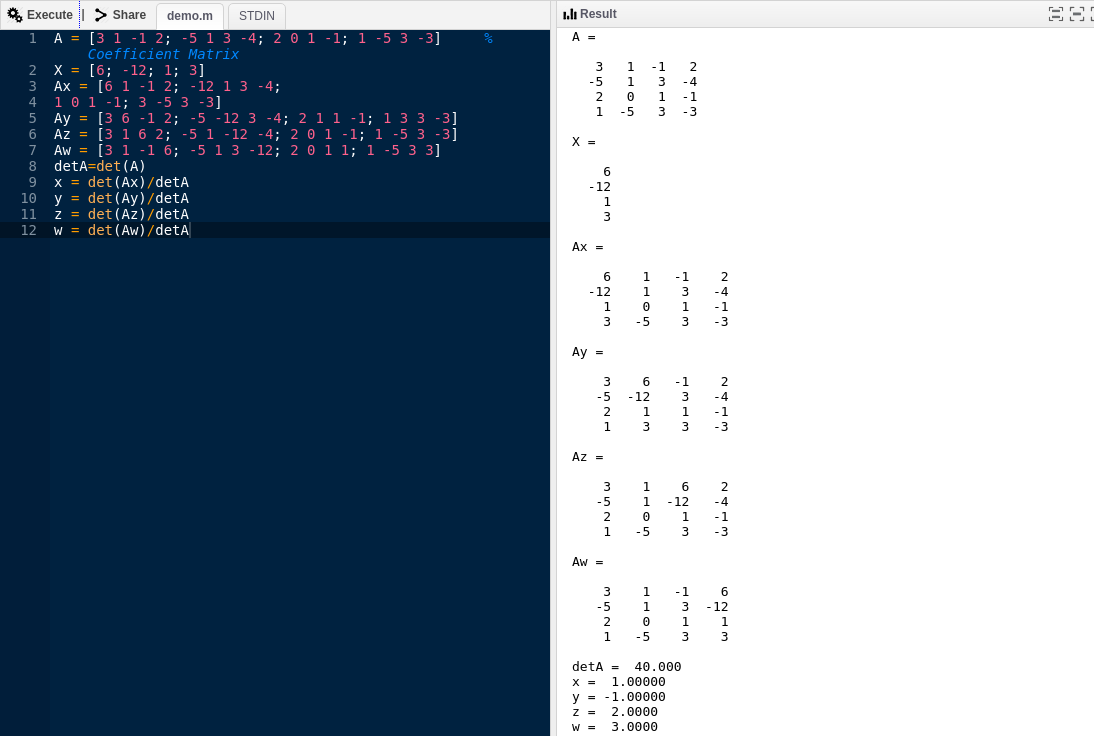


Метод Крамера



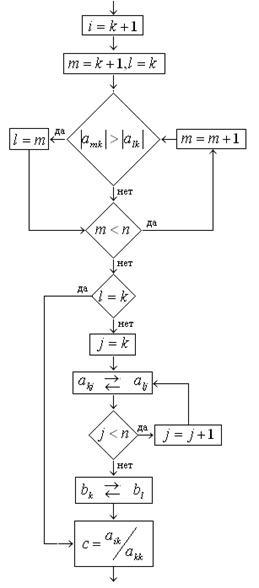


Метод Крамера в Octave

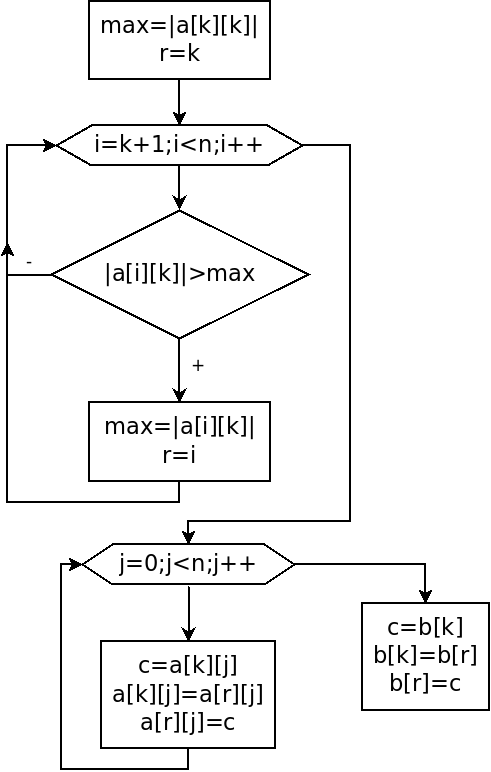


Додаток А. Графи потоку керування.

Метод Крамера:



Метод Гаусса:



Додаток Б. Лістинг програми.

Гаусс:

# Importing NumPy Library

import numpy as np

import sys

# Reading number of unknowns

n = int(input('Enter number of unknowns: '))

a = np.zeros((n,n+1))

x = np.zeros(n)

# Reading augmented matrix coefficients

print('Enter Augmented Matrix Coefficients:')

for i in range(n):

for j in range(n+1):

a[i][j] = float(input( 'a['+str(i)+']['+ str(j)+']='))

# Applying Gauss Elimination

for i in range(n):

if a[i][i] == 0.0:

sys.exit('Divide by zero detected!')

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

ratio = a[j][i]/a[i][i]

for k in range(n+1):

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

# Back Substitution

x[n-1] = a[n-1][n]/a[n-1][n-1]

for i in range(n-2,-1,-1):

x[i] = a[i][n]

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

x[i] = x[i] - a[i][j]\*x[j]

x[i] = x[i]/a[i][i]

# Displaying solution

print('\nRequired solution is: ')

for i in range(n):

print('X%d = %0.2f' %(i,x[i]), end = '\t')

Крамер:

from numpy import linalg

A=[[3,1,-1,2],[-5,1,3,-4],[2,0,1,-1],[1,-5,3,-3]]

B=[6,-12,1,3]

C=[[3,1,-1,2],[-5,1,3,-4],[2,0,1,-1],[1,-5,3,-3]]

X=[]

for i in range(0,len(B)):

for j in range(0,len(B)):

C[j][i]=B[j]

if i>0:

C[j][i-1]=A[j][i-1]

X.append(round(linalg.det(C)/linalg.det(A),1))

print('w=%s'%X[0],'x=%s'%X[1],'y=%s'%X[2],'z=%s'%X[3])