LINEAR ALGEBRA

LAB REPORT

(23MAT117)

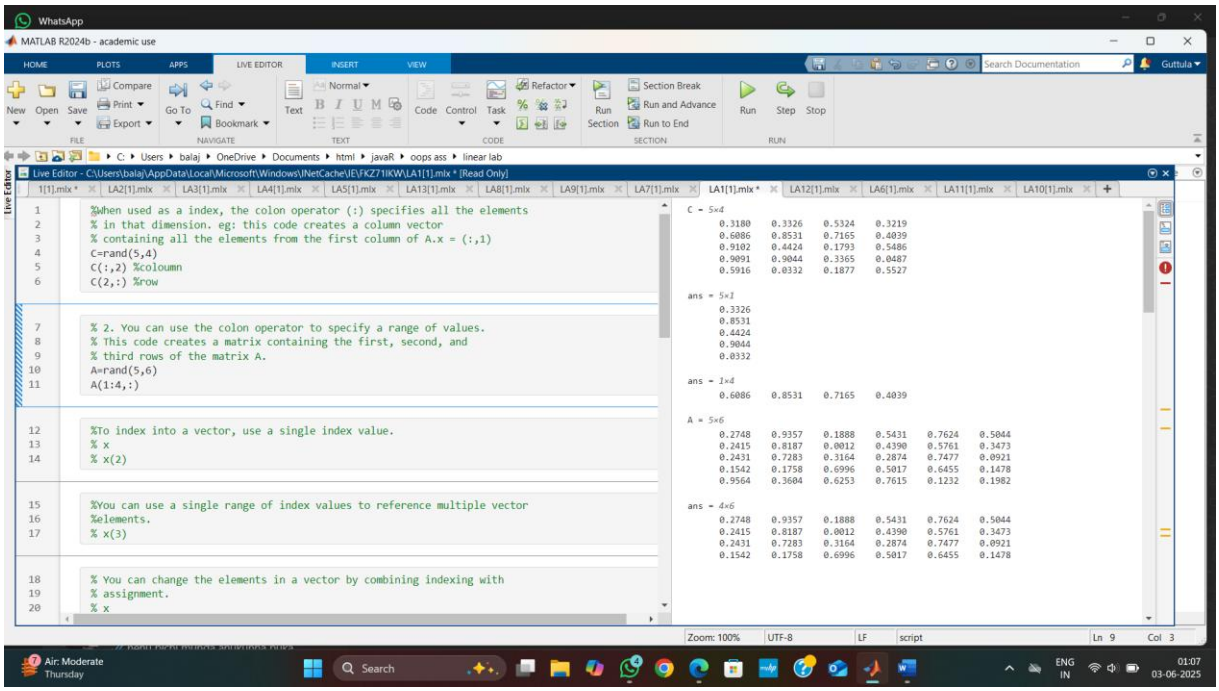
Amrita School of Engineering

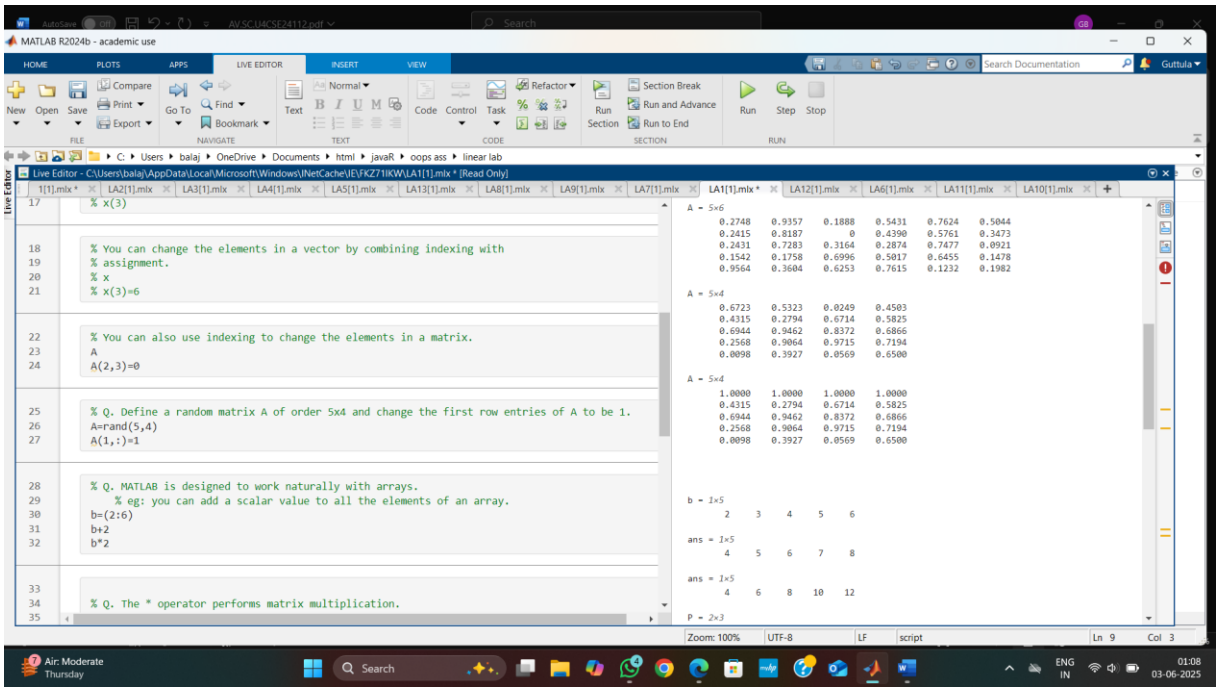
Amrita Vishwa Vidyapeetham, Amaravati Campus

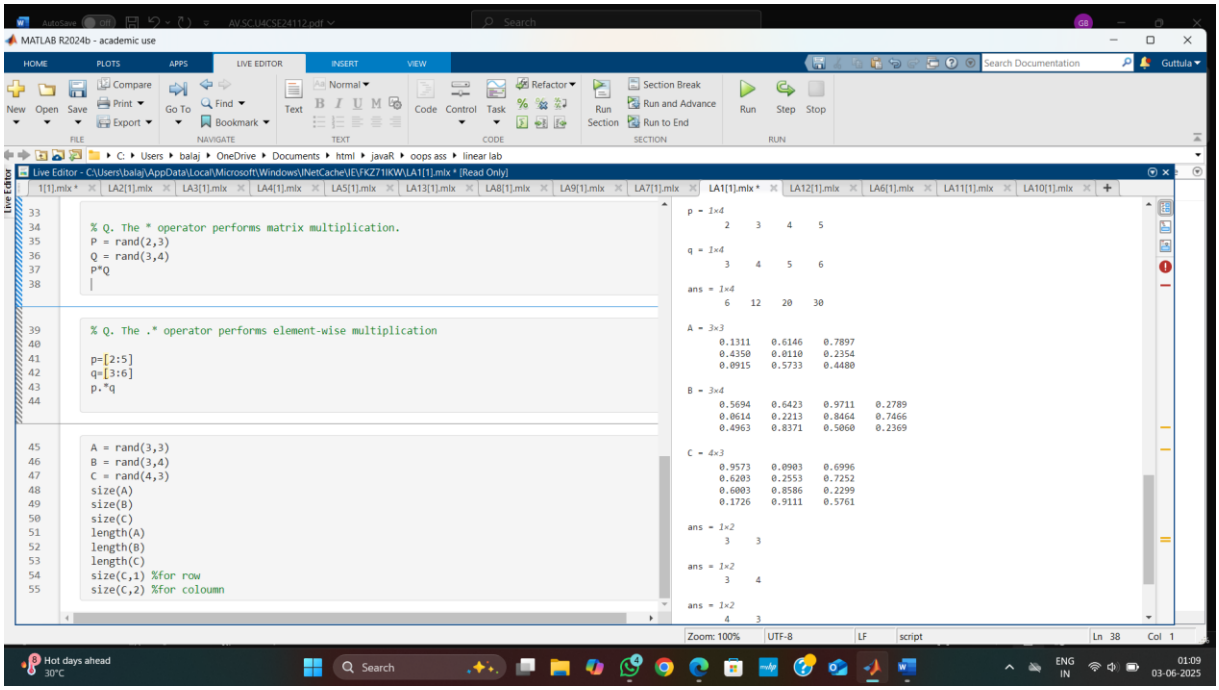
Name: G. Balaji Ajay
Section: CSE-B
Roll No: AV.SC.U4CSE24124
Semester: 2nd semester

MATRICS FUNCTIONS:

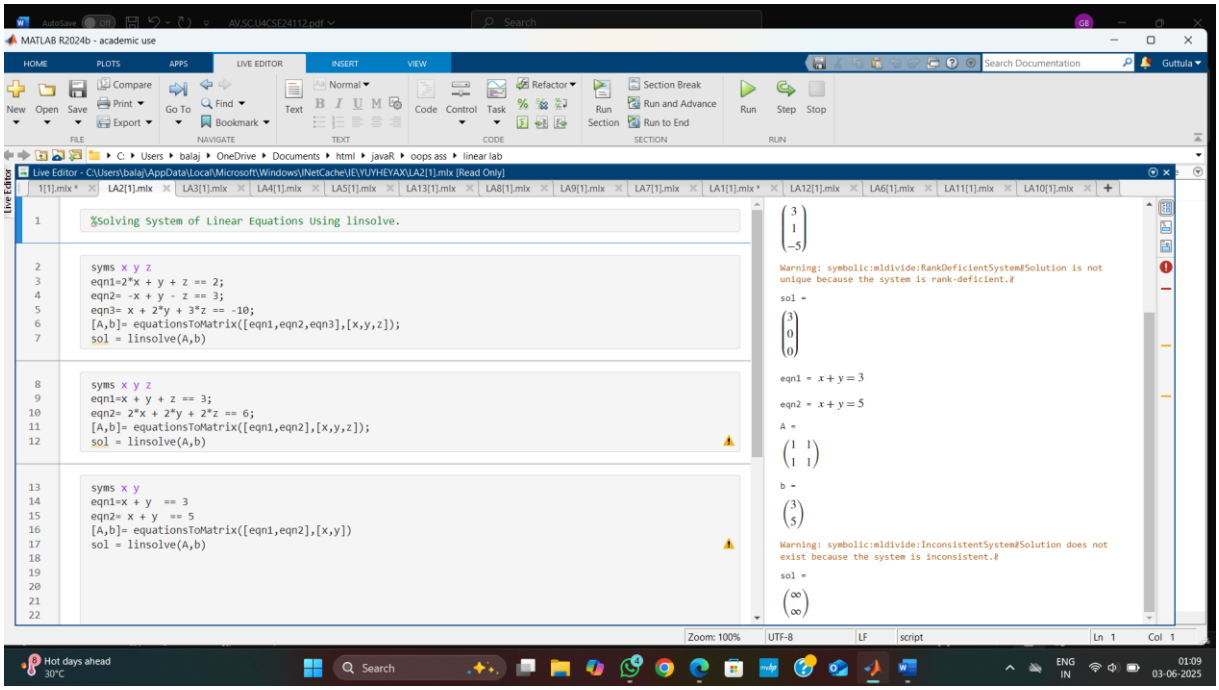
1.



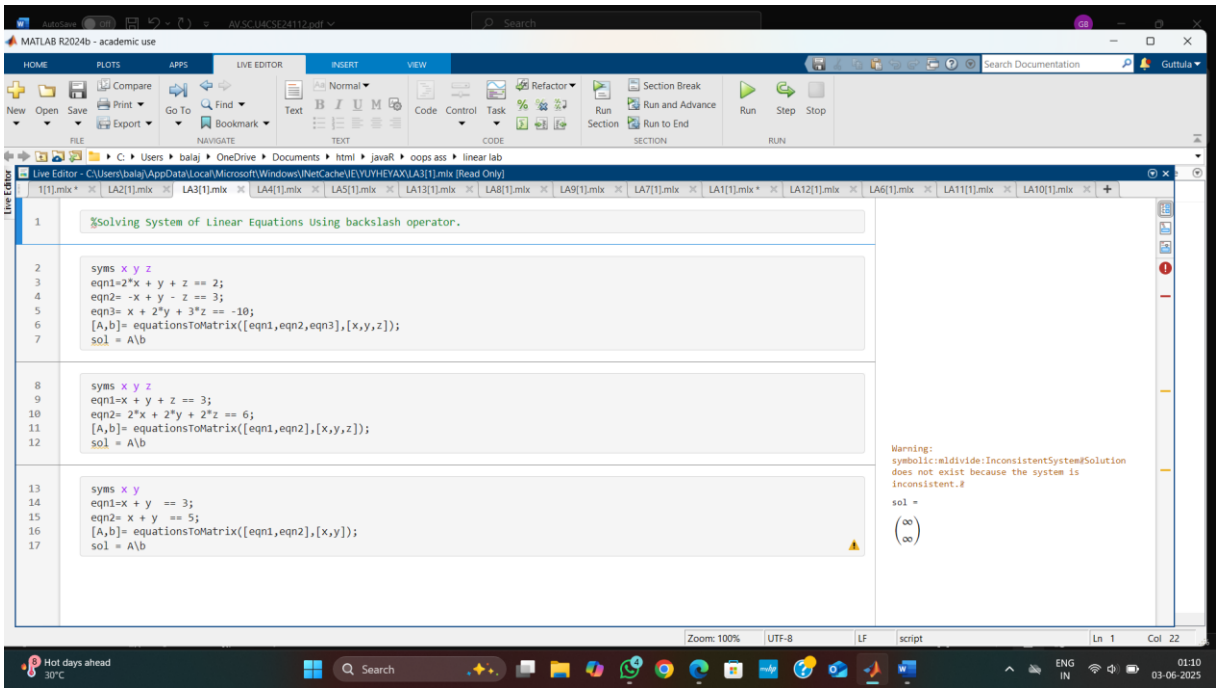



MATLAB R2024b - academic use
% Q. The * operator performs matrix multiplication.
P = rand(2,3)
Q = rand(3,4)
P*Q
% Q. The .* operator performs element-wise multiplication
p=[2:5]
q=[3:6]
p.*q
A = rand(3,3)
B = rand(3,4)
C = rand(4,3)
size(A)
size(B)
size(C)
length(A)
length(B)
length(C)
size(C,1) %for row
size(C,2) %for coloumn
p = 1×4
2 3 4 5
q = 1×4
3 4 5 6
ans = 1×4
6 12 20 30
A = 3×3
0.1311 0.6146 0.7897
0.4350 0.0110 0.2354
0.0915 0.5733 0.4480
B = 3×4
0.5694 0.6423 0.9711 0.2789
0.0614 0.2213 0.8464 0.7466
0.4963 0.8371 0.5060 0.2369
C = 4×3
0.9573 0.0903 0.6996
0.6203 0.2553 0.7252
0.6003 0.8586 0.2299
0.1726 0.9111 0.5761
ans = 1×2
3 3
ans = 1×2
3 4
ans = 1×2
4 3

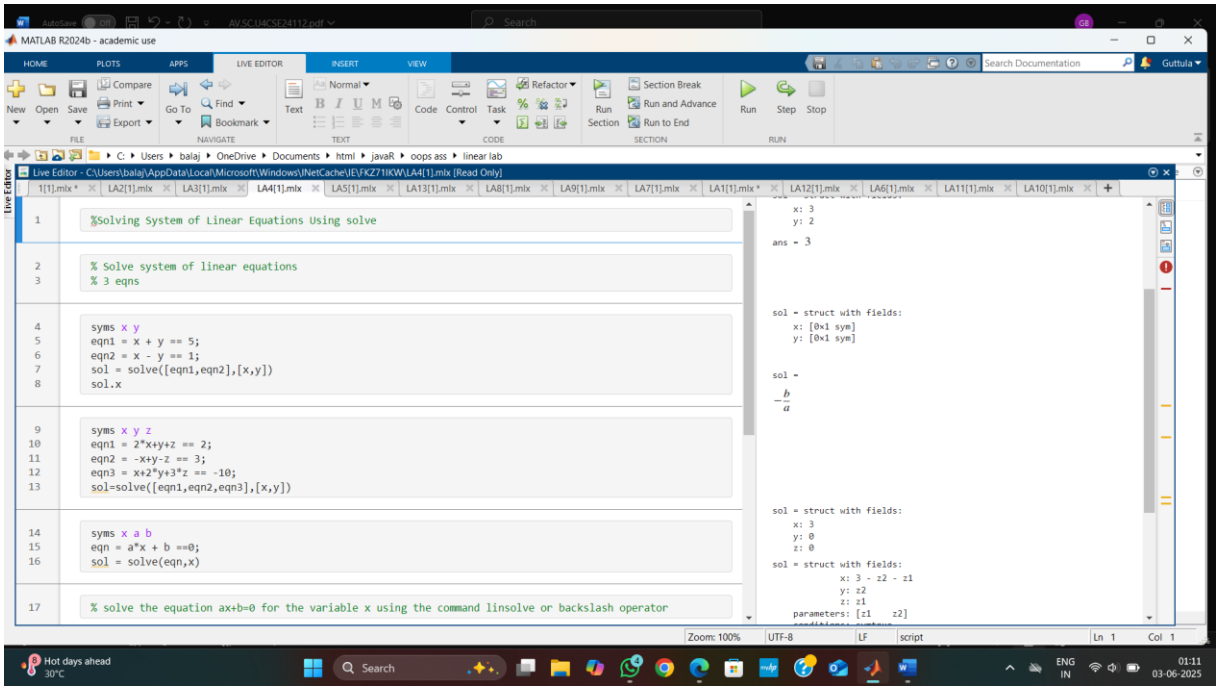
2.

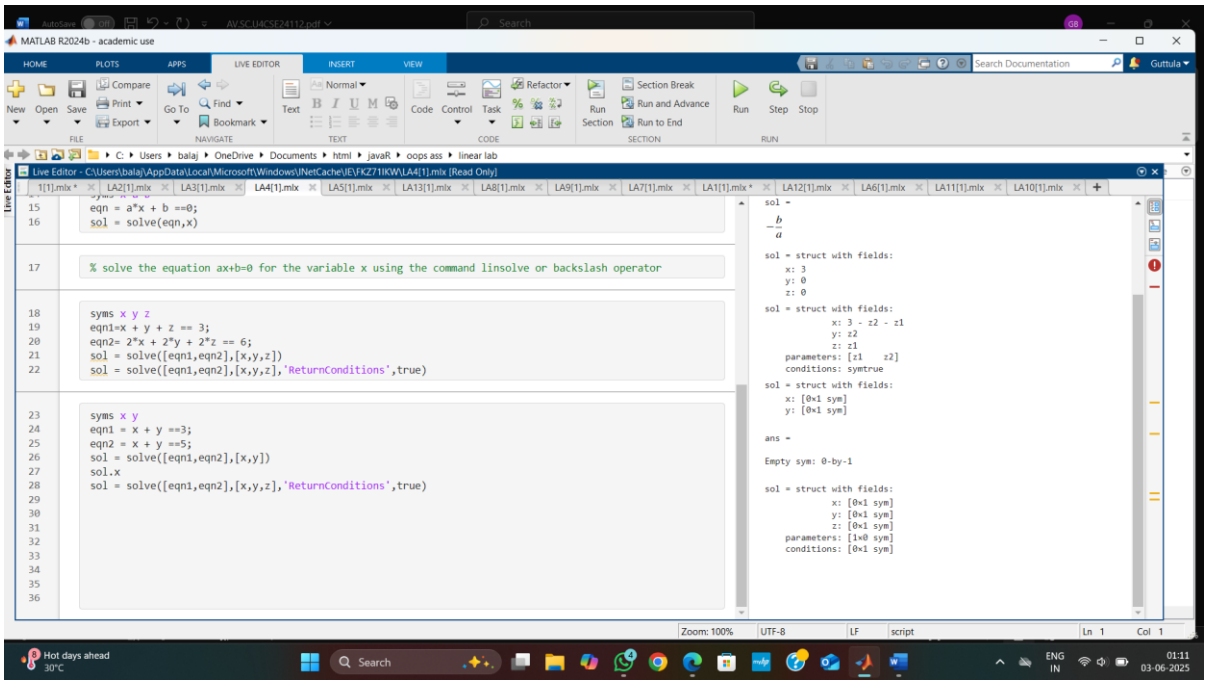

MATLAB R2024b - academic use
%Solving System of Linear Equations Using linsolve.
syms x y z
eqn1=2*x + y + z == 2;
eqn2= -x + y - z == 3;
eqn3= x + 2*y + 3*z == -10;
[A,b]= equationsToMatrix([eqn1,eqn2,eqn3],[x,y,z]);
sol = linsolve(A,b)
syms x y z
eqn1=x + y + z == 3;
eqn2= 2*x + 2*y + 2*z == 6;
[A,b]= equationsToMatrix([eqn1,eqn2],[x,y,z]);
sol = linsolve(A,b)
syms x y
eqn1=x + y == 3
eqn2= x + y == 5
[A,b]= equationsToMatrix([eqn1,eqn2],[x,y])
sol = linsolve(A,b)
Warning: symbolic:mldivide:RankDeficientSystem#Solution is not unique because the system is rank-deficient.#
sol =
eqn1 = x + y = 3
eqn2 = x + y = 5
A =
b =
Warning: symbolic:mldivide:InconsistentSystem#Solution does not exist because the system is inconsistent.#
sol =

3.

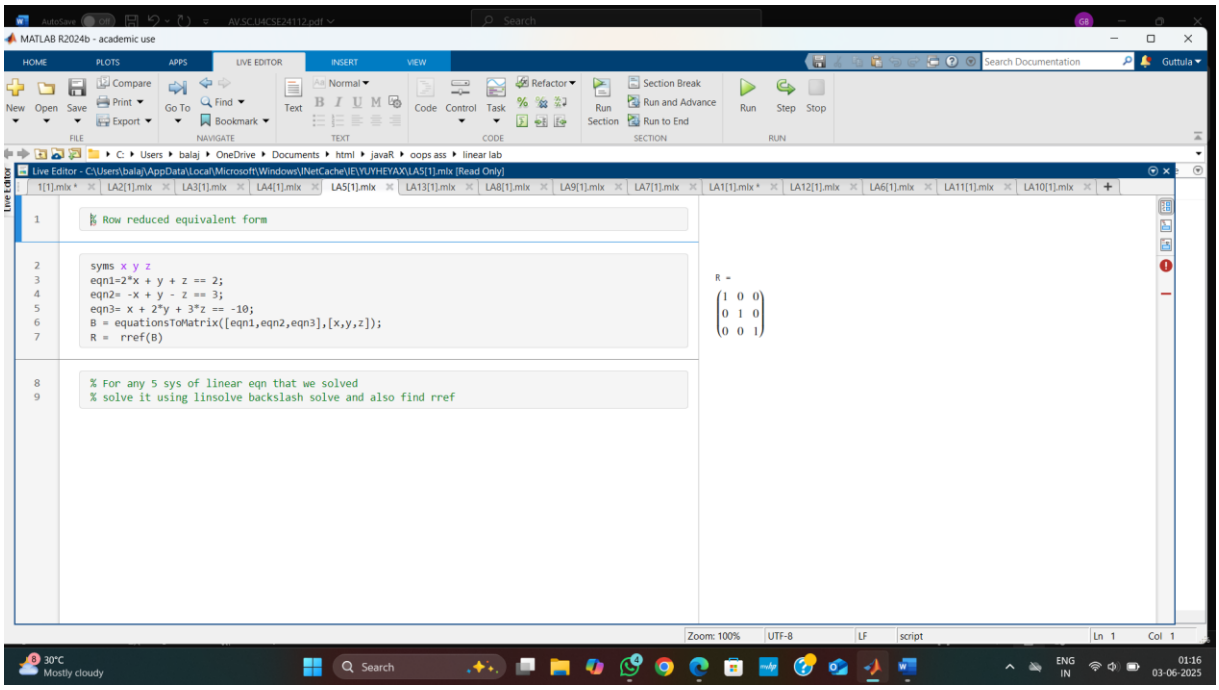

%Solving System of Linear Equations Using backslash operator.
syms x y z
eqn1=2*x + y + z == 2;
eqn2= -x + y - z == 3;
eqn3= x + 2*y + 3*z == -10;
[A,b]= equationsToMatrix([eqn1,eqn2,eqn3],[x,y,z]);
sol = A\b
syms x y z
eqn1=x + y + z == 3;
eqn2= 2*x + 2*y + 2*z == 6;
[A,b]= equationsToMatrix([eqn1,eqn2],[x,y,z]);
sol = A\b
syms x y
eqn1=x + y == 3;
eqn2= x + y == 5;
[A,b]= equationsToMatrix([eqn1,eqn2],[x,y]);
sol = A\b
Warning:
symbolic:mldivide:InconsistentSystem#Solution
does not exist because the system is
inconsistent.#
sol =
∞
∞

4.


%Solving System of Linear Equations Using solve
% Solve system of linear equations
% 3 eqns
syms x y
eqn1 = x + y == 5;
eqn2 = x - y == 1;
sol = solve([eqn1,eqn2],[x,y])
sol.x
syms x y z
eqn1 = 2*x+y+z == 2;
eqn2 = -x+y-z == 3;
eqn3 = x+2*y+3*z == -10;
sol=solve([eqn1,eqn2,eqn3],[x,y])
syms x a b
eqn = a*x + b ==0;
sol = solve(eqn,x)
% solve the equation ax+b=0 for the variable x using the command linsolve or backslash operator
x: 3
y: 2
ans = 3
sol = struct with fields:
x: [0x1 sym]
y: [0x1 sym]
sol =
−b/a
sol = struct with fields:
x: 3
y: 0
z: 0
sol = struct with fields:
x: 3 - z2 - z1
y: z2
z: z1
parameters: [z1 z2]

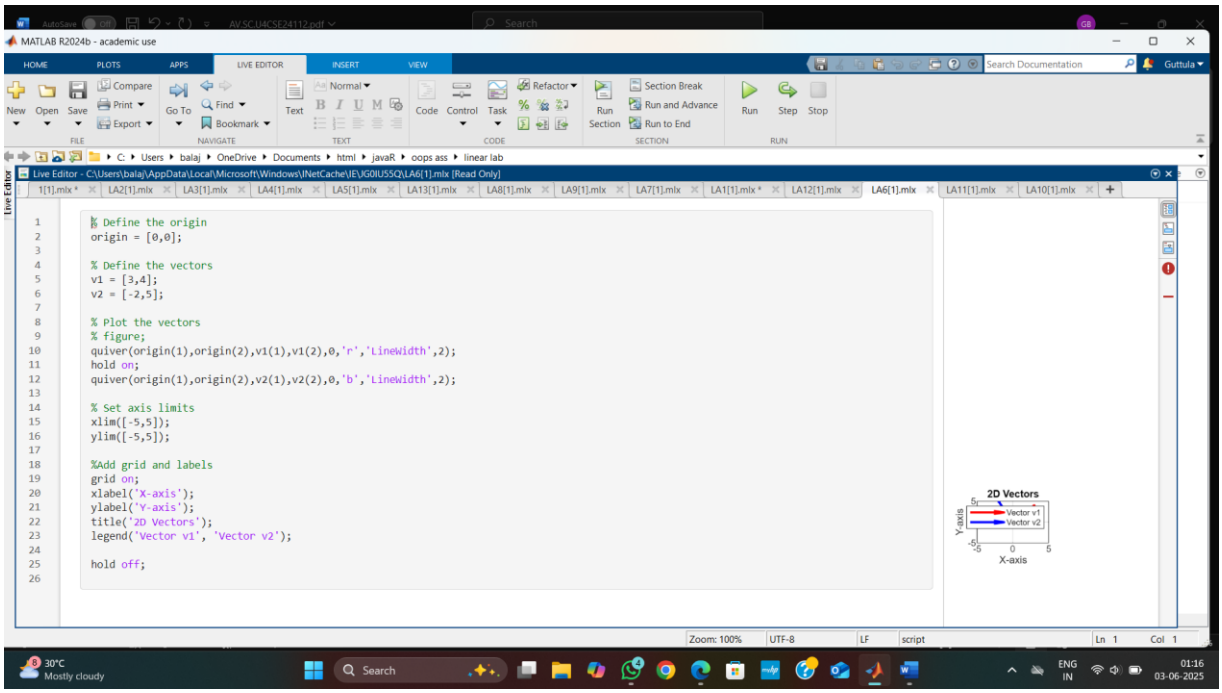

MATLAB R2024b - academic use
eqn = a*x + b ==0;
sol = solve(eqn,x)
% solve the equation ax+b=0 for the variable x using the command linsolve or backslash operator
syms x y z
eqn1=x + y + z == 3;
eqn2= 2*x + 2*y + 2*z == 6;
sol = solve([eqn1,eqn2],[x,y,z])
sol = solve([eqn1,eqn2],[x,y,z],'ReturnConditions',true)
syms x y
eqn1 = x + y ==3;
eqn2 = x + y ==5;
sol = solve([eqn1,eqn2],[x,y])
sol.x
sol = solve([eqn1,eqn2],[x,y,z],'ReturnConditions',true)
sol =
sol = struct with fields:
x: 3
y: 0
z: 0
sol = struct with fields:
x: 3 - z2 - z1
y: z2
z: z1
parameters: [z1 z2]
conditions: symtrue
sol = struct with fields:
x: [0×1 sym]
y: [0×1 sym]
ans =
Empty sym: 0-by-1
sol = struct with fields:
x: [0×1 sym]
y: [0×1 sym]
z: [0×1 sym]
parameters: [1×0 sym]
conditions: [0×1 sym]

5.

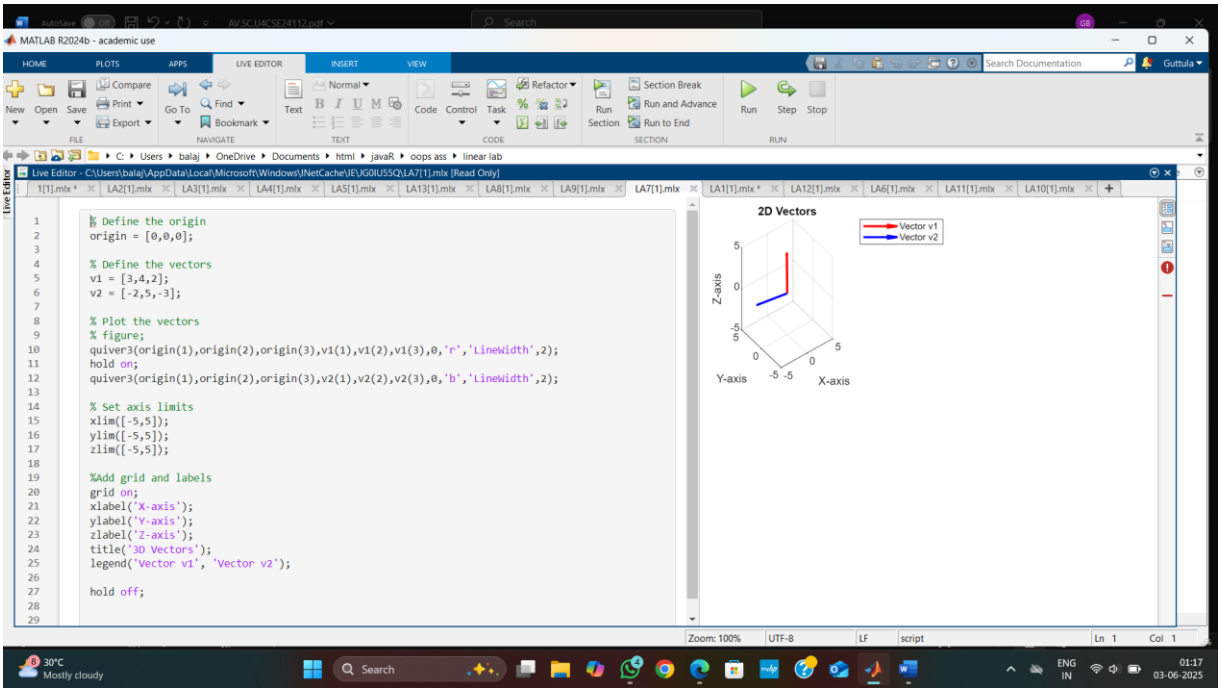

MATLAB R2024b - academic use
% Row reduced equivalent form
syms x y z
eqn1=2*x + y + z == 2;
eqn2= -x + y - z == 3;
eqn3= x + 2*y + 3*z == -10;
B = equationsToMatrix([eqn1,eqn2,eqn3],[x,y,z]);
R = rref(B)
% For any 5 sys of linear eqn that we solved
% solve it using linsolve backslash solve and also find rref
R =

VECTORS GRAPHS

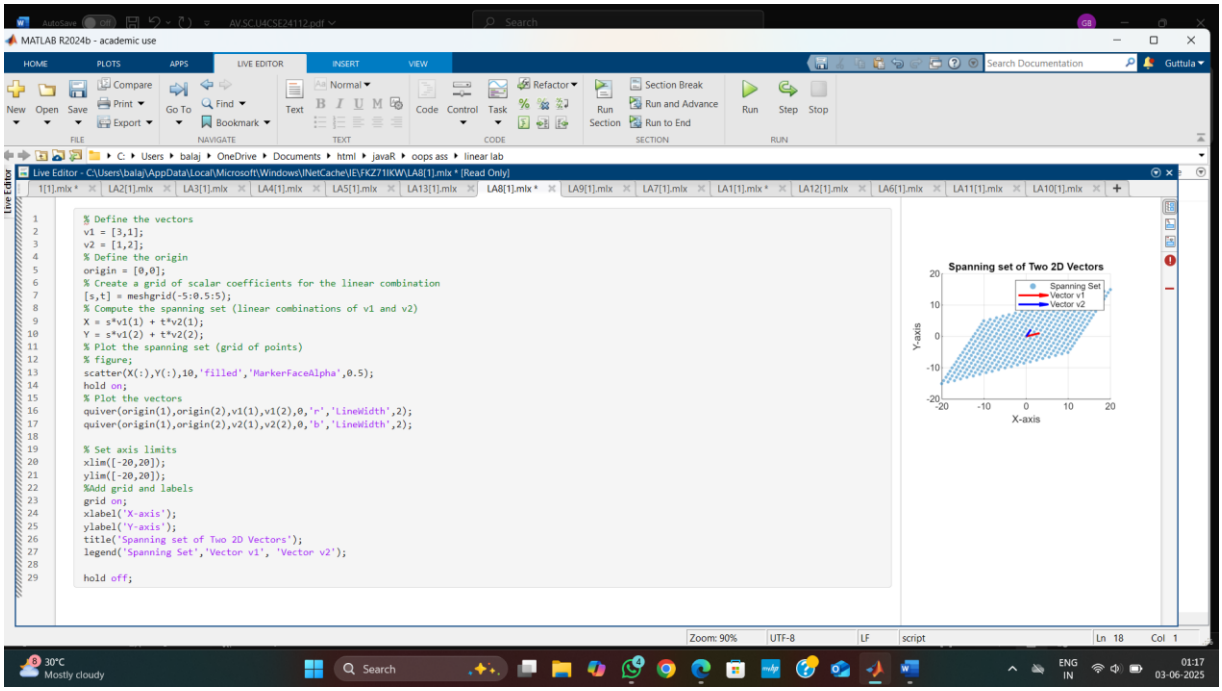
6.



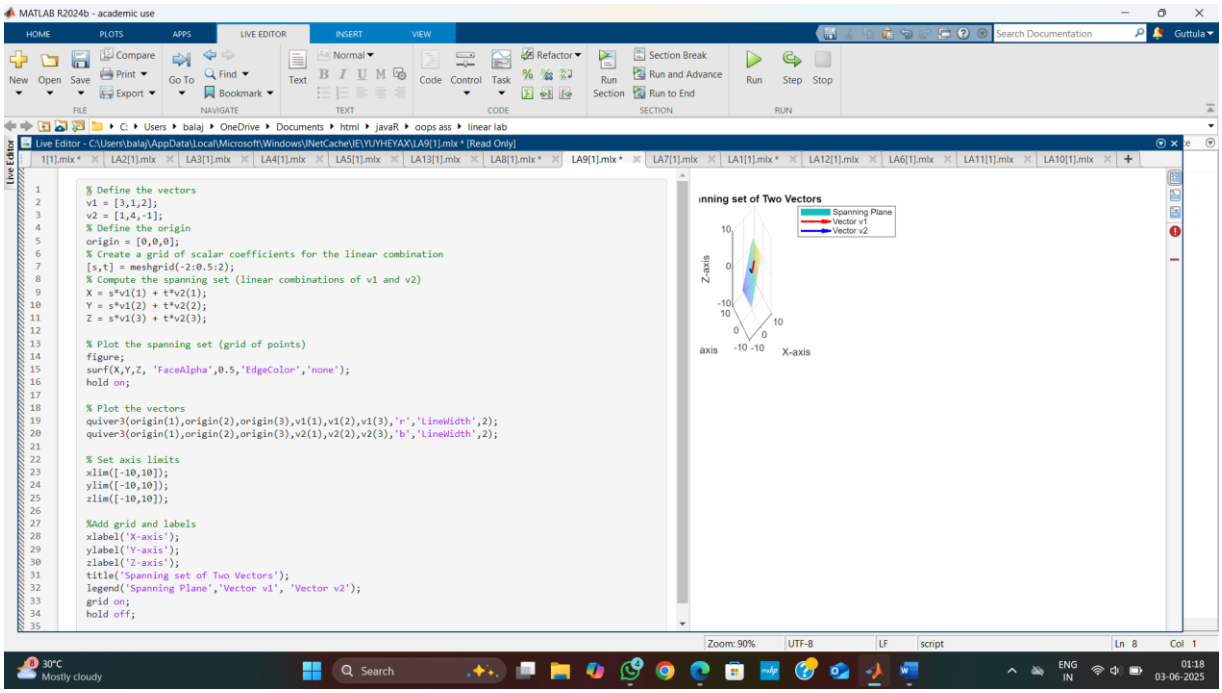
7.



8.

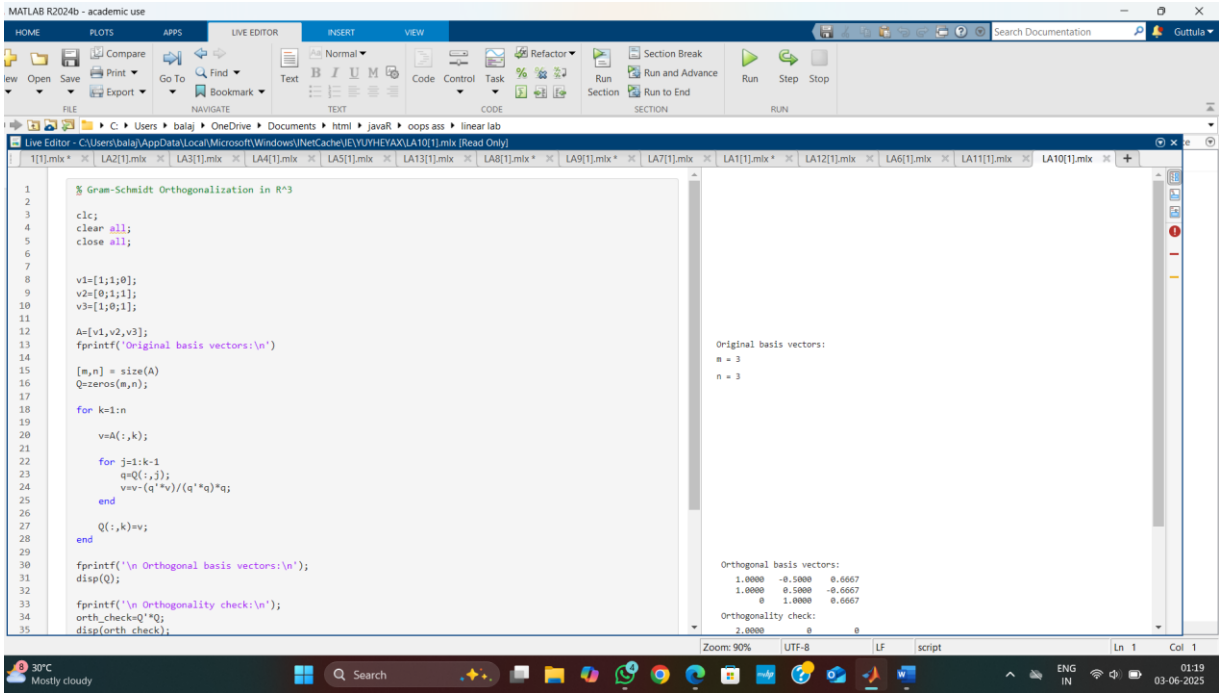


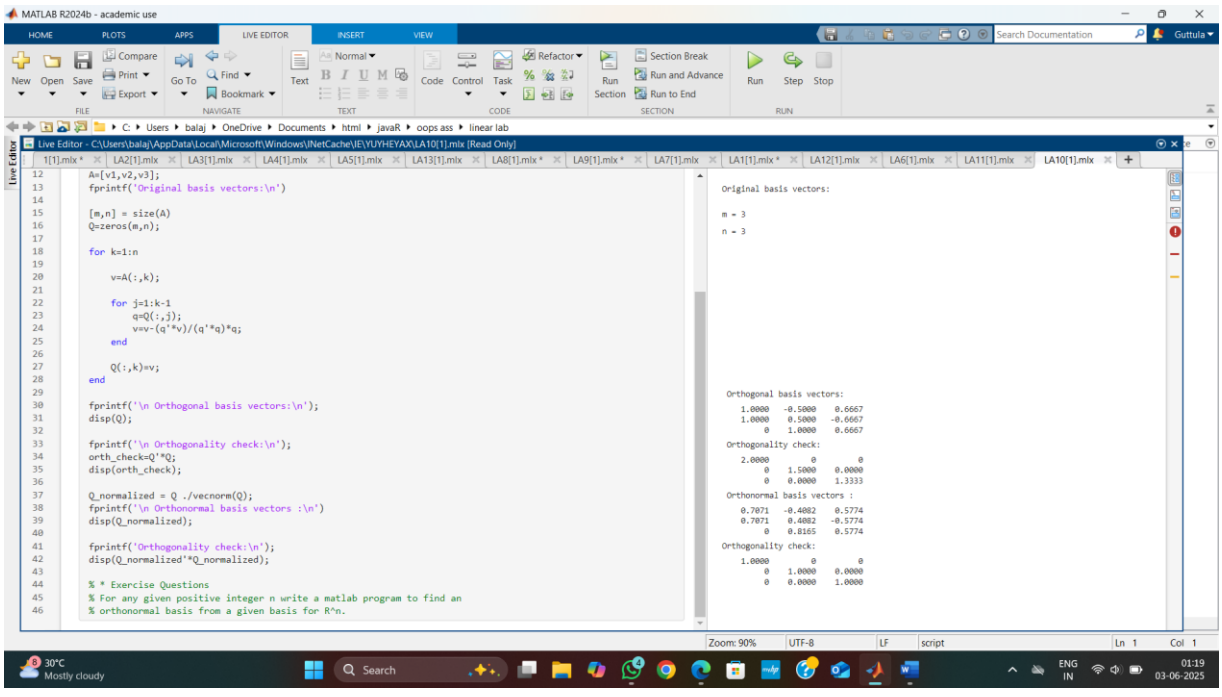
9.



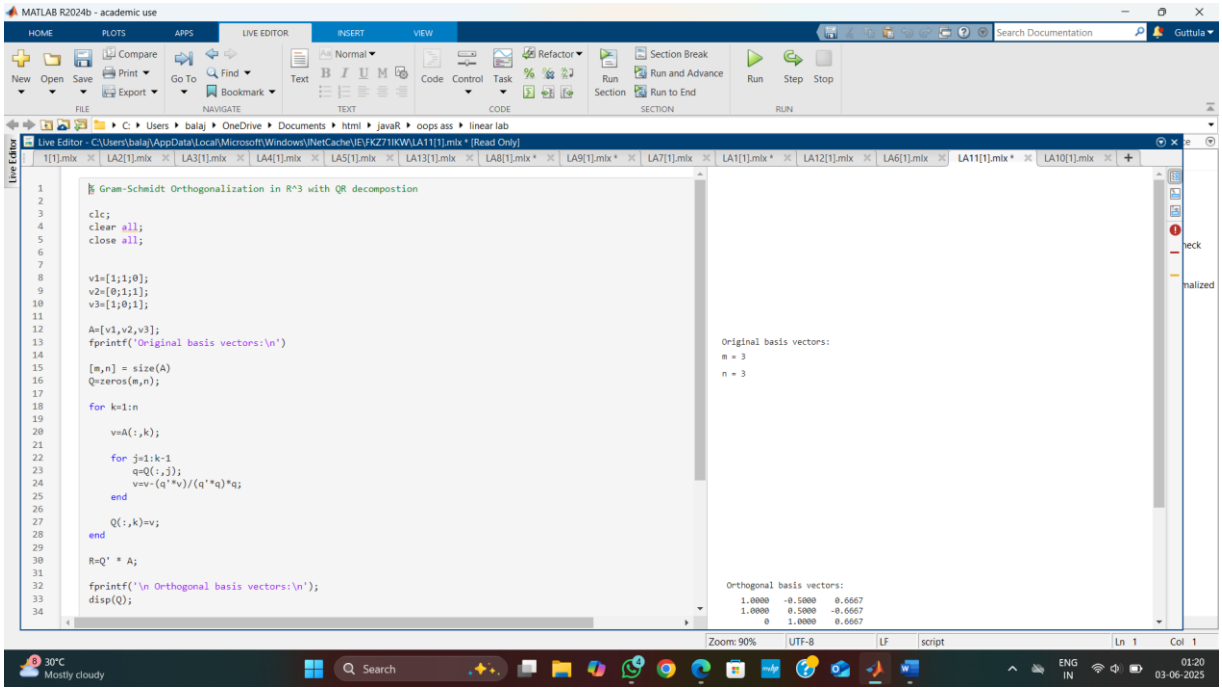
ORTHOGONALIZATION

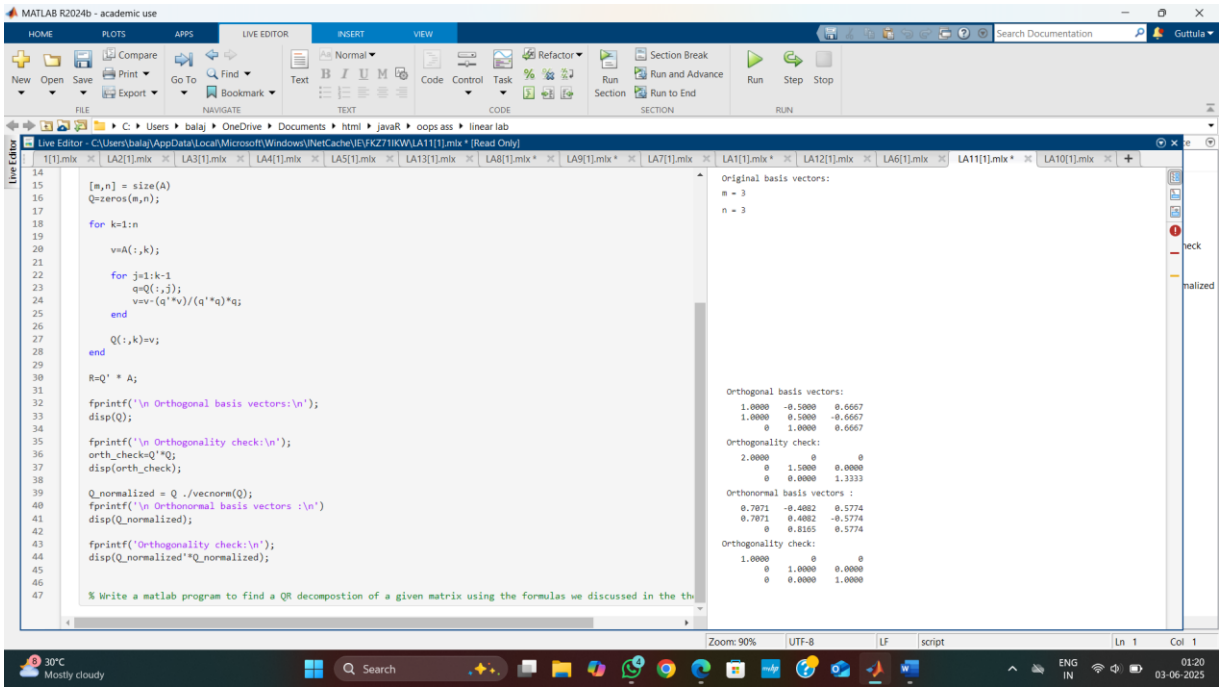
10.


MATLAB R2024b - academic use
% Gram-Schmidt Orthogonalization in R^3
clc;
clear all;
close all;
v1=[1;1;0];
v2=[0;1;1];
v3=[1;0;1];
A=[v1,v2,v3];
fprintf('Original basis vectors:\n')
[m,n] = size(A)
Q=zeros(m,n);
for k=1:n
v=A(:,k);
for j=1:k-1
q=Q(:,j);
v=v-(q'*v)/(q'*q)*q;
end
Q(:,k)=v;
end
fprintf('\n Orthogonal basis vectors:\n');
disp(Q);
fprintf('\n Orthogonality check:\n');
orth_check=Q'*Q;
disp(orth_check);
Original basis vectors:
m = 3
n = 3
Orthogonal basis vectors:
1.0000 -0.5000 0.6667
1.0000 0.5000 -0.6667
0 1.0000 0.6667
Orthogonality check:
2.0000 0 0


MATLAB R2024b - academic use
A=[v1,v2,v3];
fprintf('Original basis vectors:\n')
[m,n] = size(A)
Q=zeros(m,n);
for k=1:n
v=A(:,k);
for j=1:k-1
q=Q(:,j);
v=v-(q'*v)/(q'*q)*q;
end
Q(:,k)=v;
end
fprintf('\n Orthogonal basis vectors:\n');
disp(Q);
fprintf('\n Orthogonality check:\n');
orth_check=Q'*Q;
disp(orth_check);
Q_normalized = Q ./vecnorm(Q);
fprintf('\n Orthonormal basis vectors :\n')
disp(Q_normalized);
fprintf('Orthogonality check:\n');
disp(Q_normalized'*Q_normalized);
% * Exercise Questions
% For any given positive integer n write a matlab program to find an
% orthonormal basis from a given basis for R^n.
Original basis vectors:
m = 3
n = 3
Orthogonal basis vectors:
1.0000 -0.5000 0.6667
1.0000 0.5000 -0.6667
0 1.0000 0.6667
Orthogonality check:
2.0000 0 0
0 1.5000 0.0000
0 0.0000 1.3333
Orthonormal basis vectors :
0.7071 -0.4082 0.5774
0.7071 0.4082 -0.5774
0 0.8165 0.5774
Orthogonality check:
1.0000 0 0
0 1.0000 0.0000
0 0.0000 1.0000

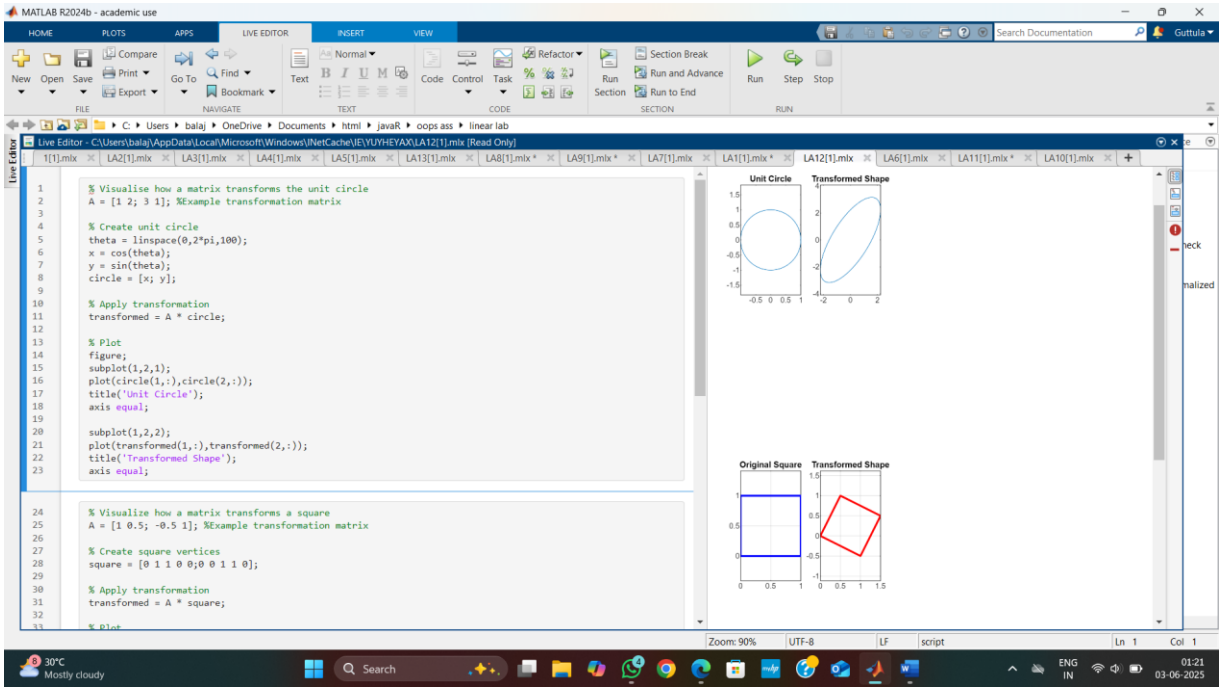
11.

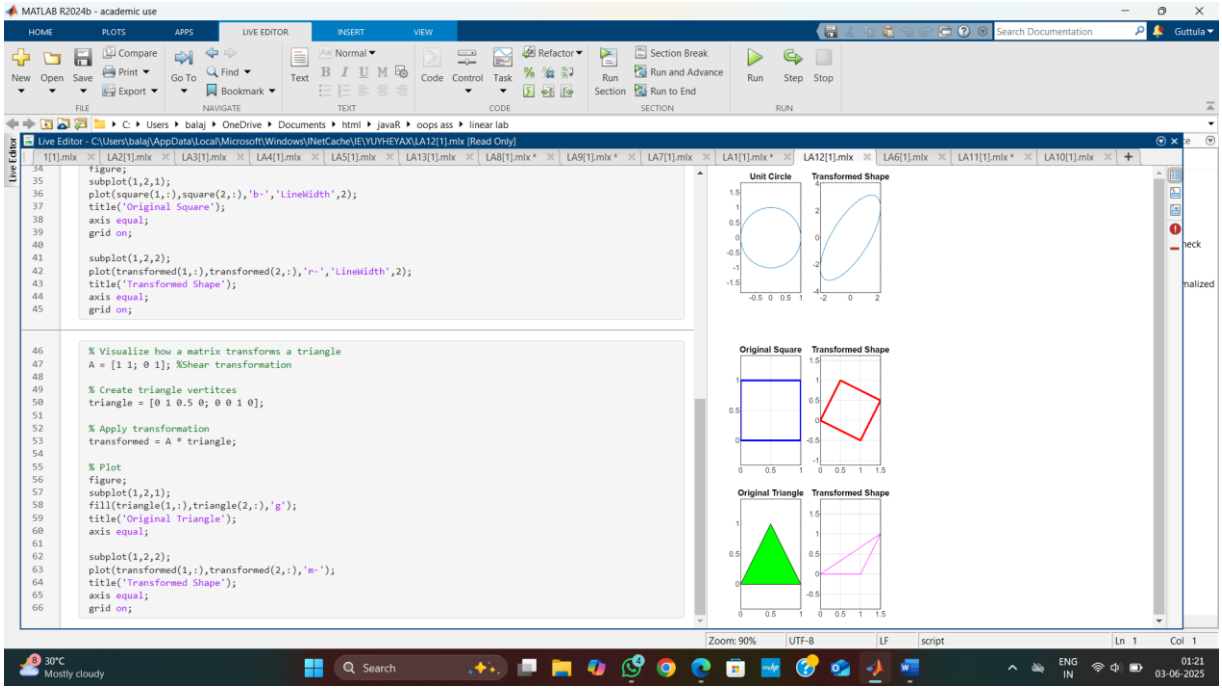




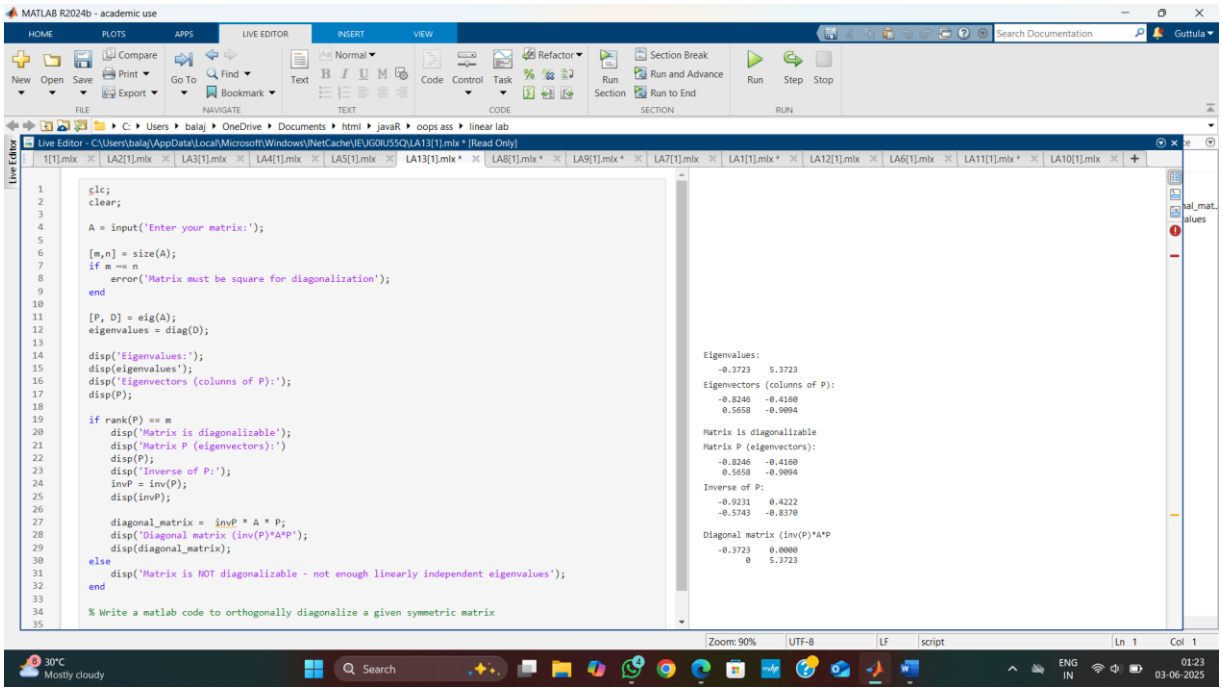
SHAPES GRAPHS

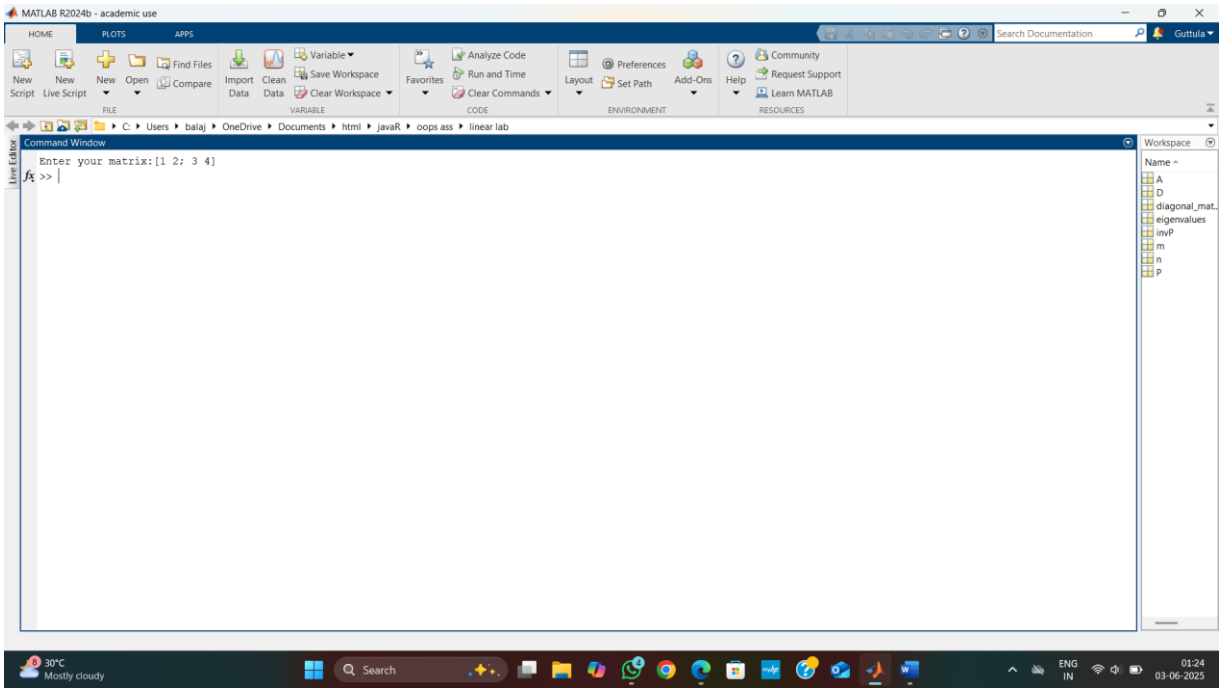
12.


MATLAB R2024b - academic use
% Visualise how a matrix transforms the unit circle
A = [1 2; 3 1]; %Example transformation matrix
% Create unit circle
theta = linspace(0,2*pi,100);
x = cos(theta);
y = sin(theta);
circle = [x; y];
% Apply transformation
transformed = A * circle;
% Plot
figure;
subplot(1,2,1);
plot(circle(1,:),circle(2,:));
title('Unit Circle');
axis equal;
subplot(1,2,2);
plot(transformed(1,:),transformed(2,:));
title('Transformed Shape');
axis equal;
% Visualize how a matrix transforms a square
A = [1 0.5; -0.5 1]; %Example transformation matrix
% Create square vertices
square = [0 1 1 0 0;0 0 1 1 0];
% Apply transformation
transformed = A * square;
Unit Circle
Transformed Shape
Original Square
Transformed Shape


MATLAB R2024b - academic use
figure;
subplot(1,2,1);
plot(square(1,:),square(2,:),'b-','LineWidth',2);
title('Original Square');
axis equal;
grid on;
subplot(1,2,2);
plot(transformed(1,:),transformed(2,:),'r-','LineWidth',2);
title('Transformed Shape');
axis equal;
grid on;
% Visualize how a matrix transforms a triangle
A = [1 1; 0 1]; %Shear transformation
% Create triangle vertitces
triangle = [0 1 0.5 0; 0 0 1 0];
% Apply transformation
transformed = A * triangle;
% Plot
figure;
subplot(1,2,1);
fill(triangle(1,:),triangle(2,:),'g');
title('Original Triangle');
axis equal;
subplot(1,2,2);
plot(transformed(1,:),transformed(2,:),'m-');
title('Transformed Shape');
axis equal;
grid on;
Unit Circle
Transformed Shape
Original Square
Transformed Shape
Original Triangle
Transformed Shape

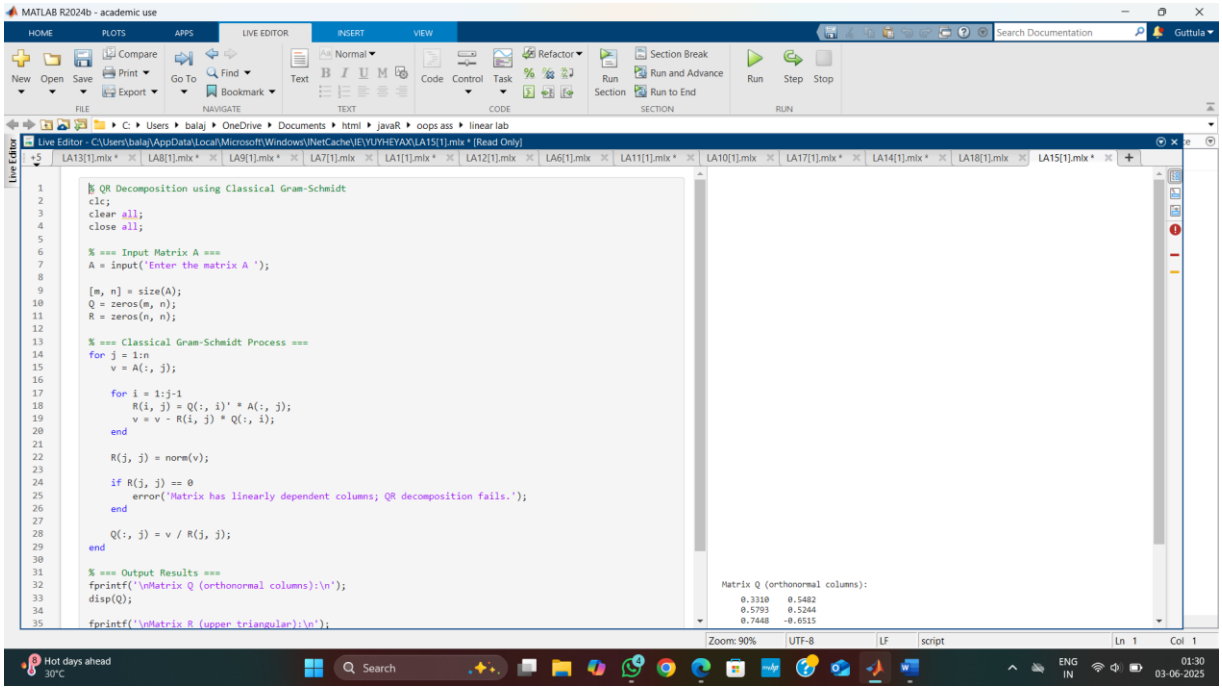
13.

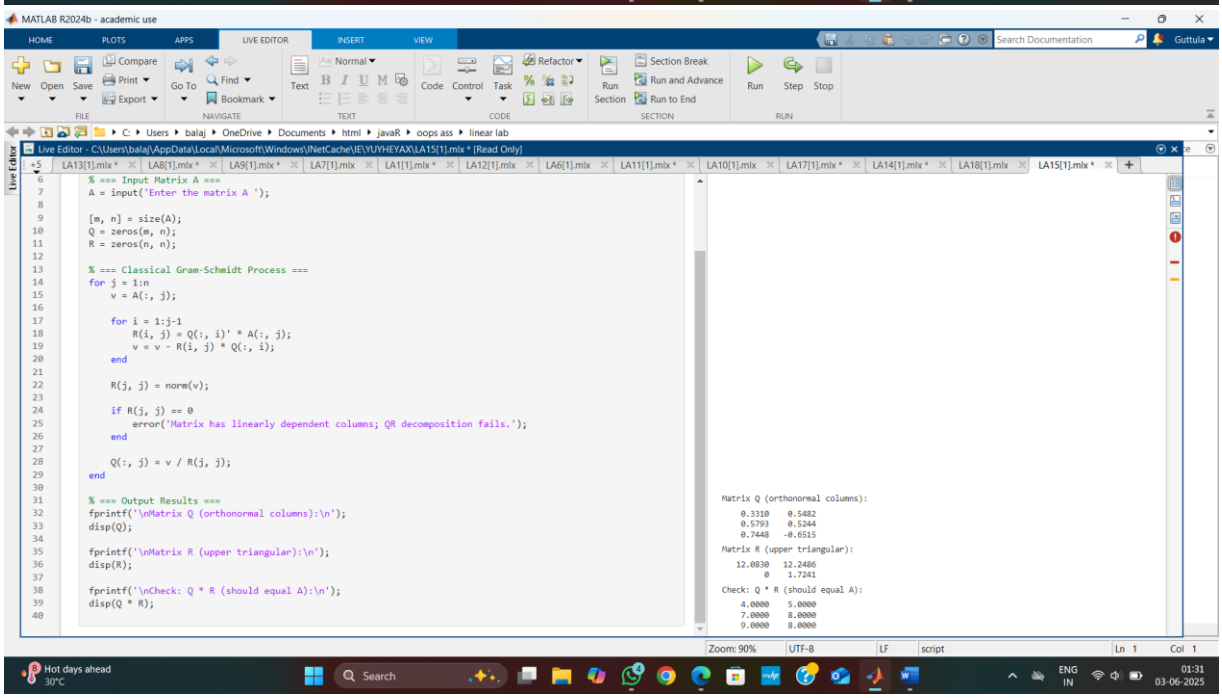

MATLAB R2024b - academic use
HOME
PLOTS
APPS
LIVE EDITOR
INSERT
VIEW
Search Documentation
Guttula
New
Open
Save
Compare
Print
Export
Go To
Find
Bookmark
Text
Normal
Code
Control
Task
Refactor
Run Section
Section Break
Run and Advance
Run to End
Run
Step
Stop
FILE
NAVIGATE
TEXT
CODE
SECTION
RUN
C: ▸ Users ▸ balaj ▸ OneDrive ▸ Documents ▸ html ▸ javaR ▸ oops ass ▸ linear lab
Live Editor - C:\Users\balaj\AppData\Local\Microsoft\Windows\INetCache\IE\JG0IU55Q\LA13[1].mlx * [Read Only]
1[1].mlx
LA2[1].mlx
LA3[1].mlx
LA4[1].mlx
LA5[1].mlx
LA13[1].mlx *
LA8[1].mlx *
LA9[1].mlx *
LA7[1].mlx
LA1[1].mlx *
LA12[1].mlx
LA6[1].mlx
LA11[1].mlx *
LA10[1].mlx
clc;
clear;
A = input('Enter your matrix:');
[m,n] = size(A);
if m ~= n
error('Matrix must be square for diagonalization');
end
[P, D] = eig(A);
eigenvalues = diag(D);
disp('Eigenvalues:');
disp(eigenvalues');
disp('Eigenvectors (colunns of P):');
disp(P);
if rank(P) == m
disp('Matrix is diagonalizable');
disp('Matrix P (eigenvectors):')
disp(P);
disp('Inverse of P:');
invP = inv(P);
disp(invP);
diagonal_matrix = invP * A * P;
disp('Diagonal matrix (inv(P)*A*P');
disp(diagonal_matrix);
else
disp('Matrix is NOT diagonalizable - not enough linearly independent eigenvalues');
end
% Write a matlab code to orthogonally diagonalize a given symmetric matrix
Eigenvalues:
-0.3723 5.3723
Eigenvectors (colunns of P):
-0.8246 -0.4160
0.5658 -0.9094
Matrix is diagonalizable
Matrix P (eigenvectors):
-0.8246 -0.4160
0.5658 -0.9094
Inverse of P:
-0.9231 0.4222
-0.5743 -0.8370
Diagonal matrix (inv(P)*A*P
-0.3723 0.0000
0 5.3723
Zoom: 90%
UTF-8
LF
script
Ln 1
Col 1
30°C
Mostly cloudy
Search
ENG
IN
01:23
03-06-2025

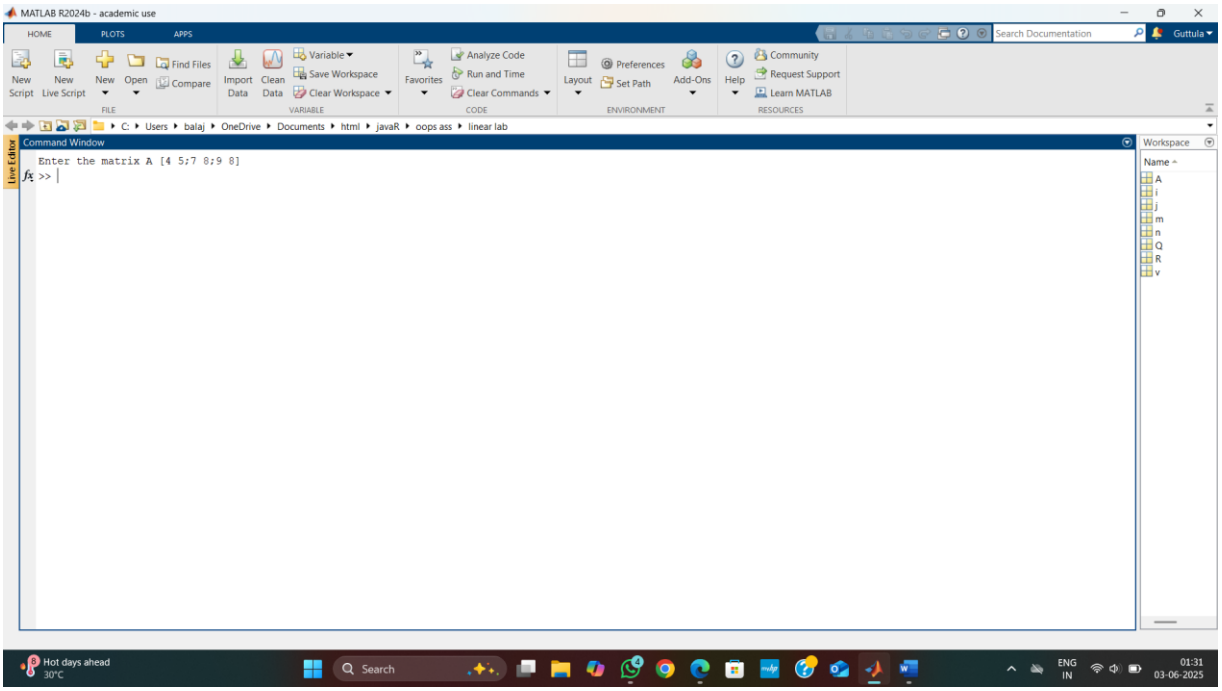

MATLAB R2024b - academic use
HOME
PLOTS
APPS
Search Documentation
Guttula
New Script
New Live Script
New
Open
Find Files
Compare
Import Data
Clean Data
Variable
Save Workspace
Clear Workspace
Favorites
Analyze Code
Run and Time
Clear Commands
Layout
Preferences
Set Path
Add-Ons
Help
Community
Request Support
Learn MATLAB
FILE
VARIABLE
CODE
ENVIRONMENT
RESOURCES
C: ▸ Users ▸ balaj ▸ OneDrive ▸ Documents ▸ html ▸ javaR ▸ oops ass ▸ linear lab
Command Window
Enter your matrix:[1 2; 3 4]
>>
Workspace
Name
A
D
diagonal_mat..
eigenvalues
invP
m
n
P
30°C
Mostly cloudy
Search
ENG
IN
01:24
03-06-2025

QUESTIONS ASKED:

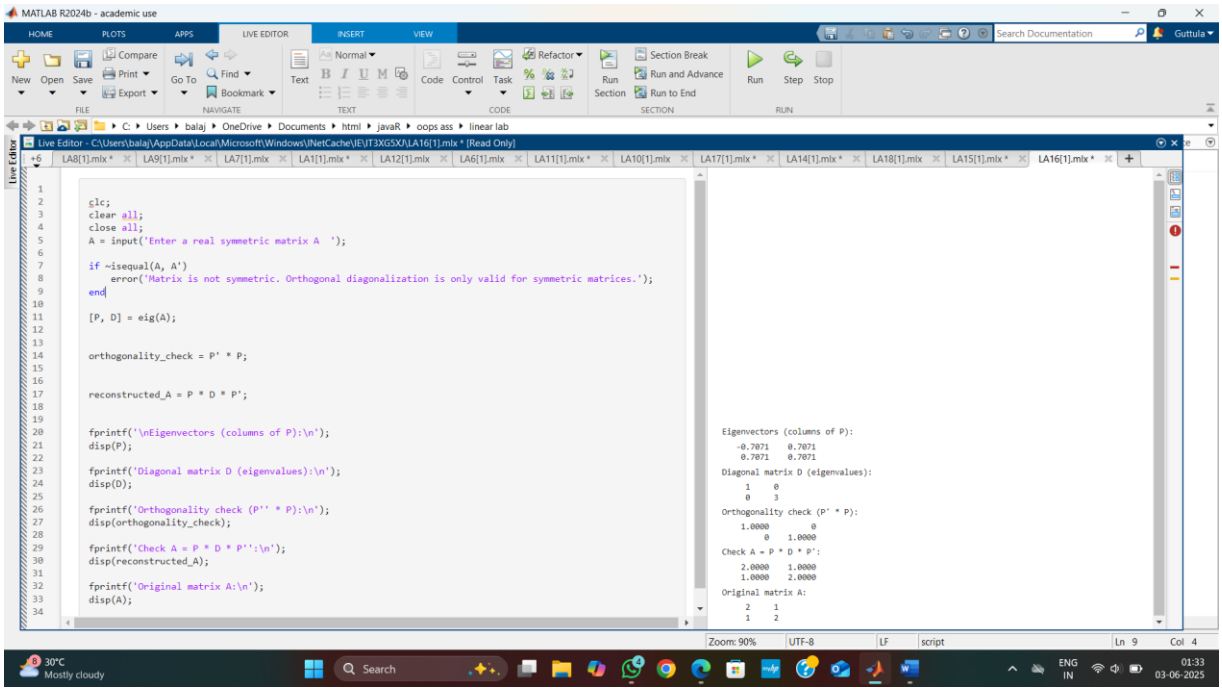
1.WRITE A MATLAB PROGRAM TO FIND A QR DECOMPOSTION OF A GIVEN MATRIX USING THE FORMULAS WE DISCUSSED IN THE THEORY SESSION

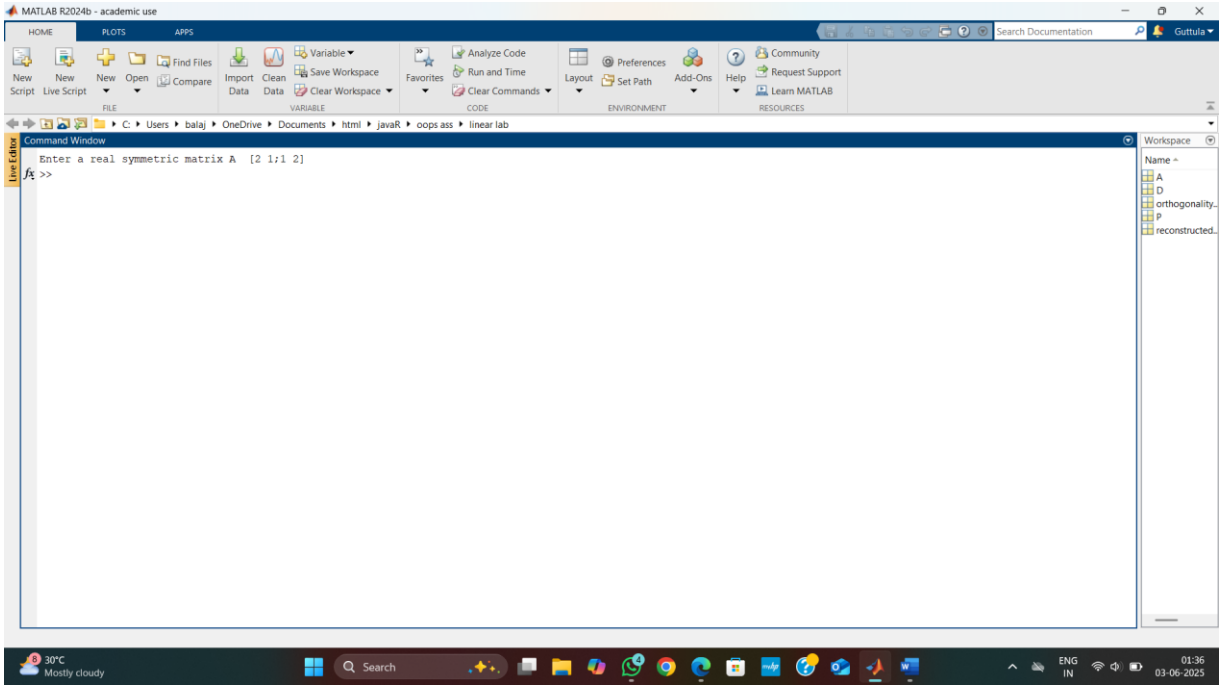




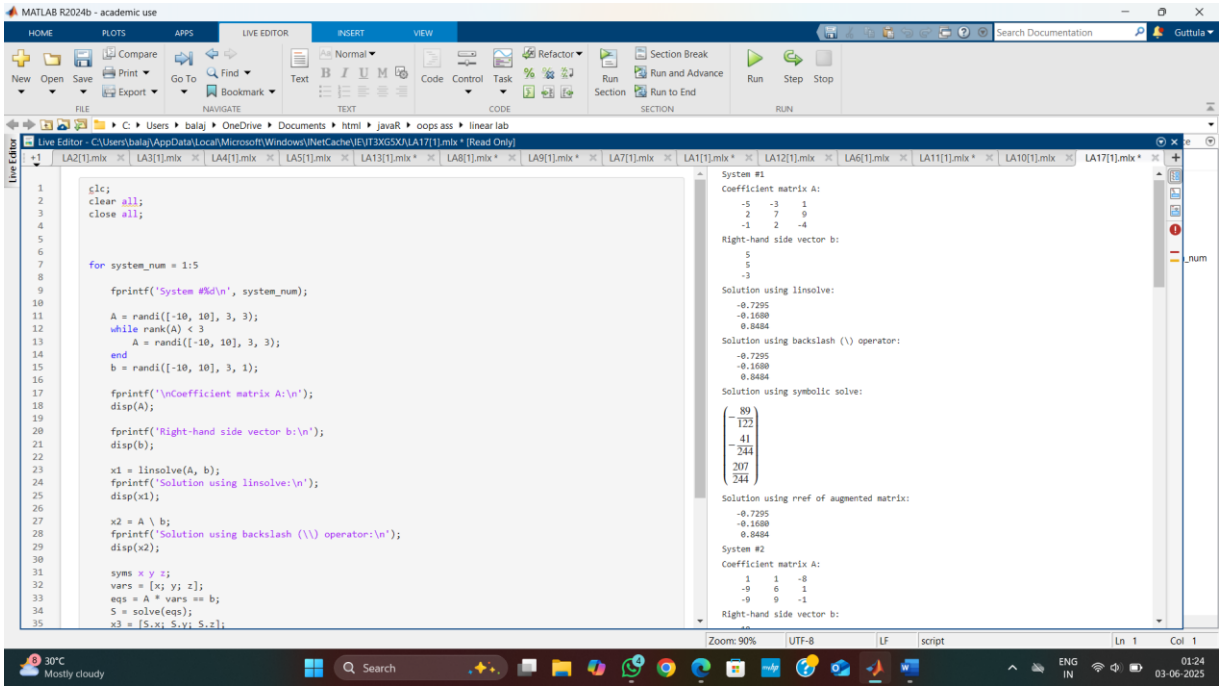


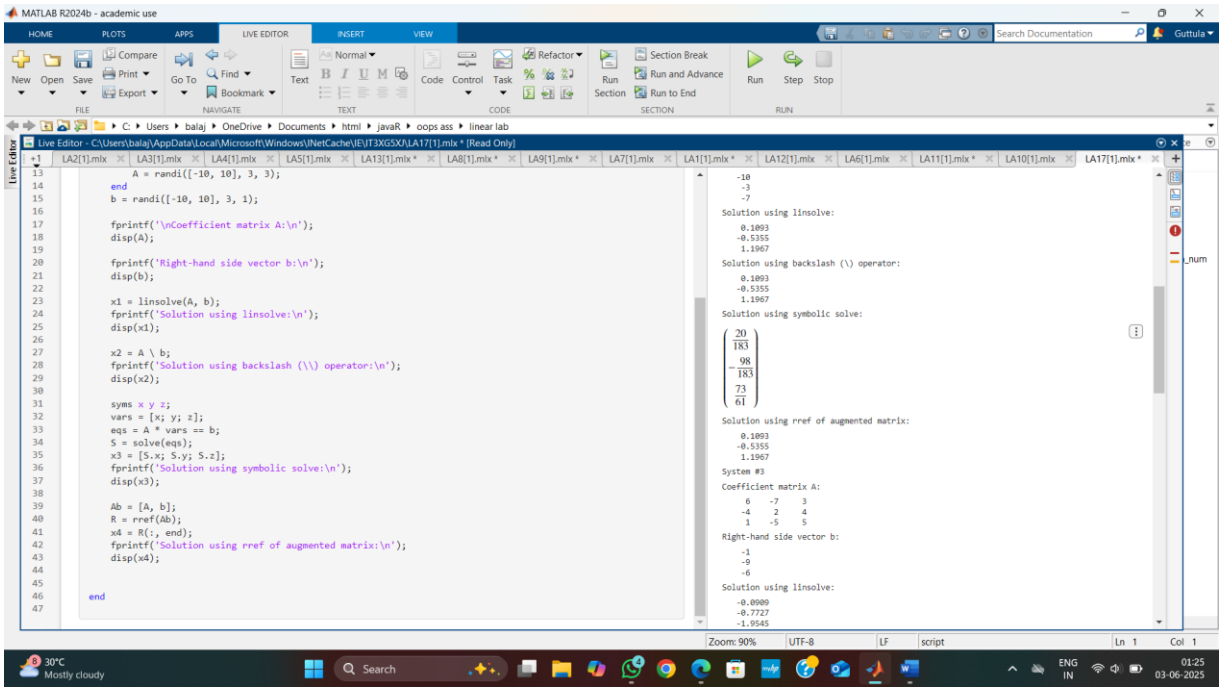
2. WRITE A MATLAB CODE TO ORTHOGONALLY DIAGONALIZE A GIVEN SYMMETRIC MATRIX

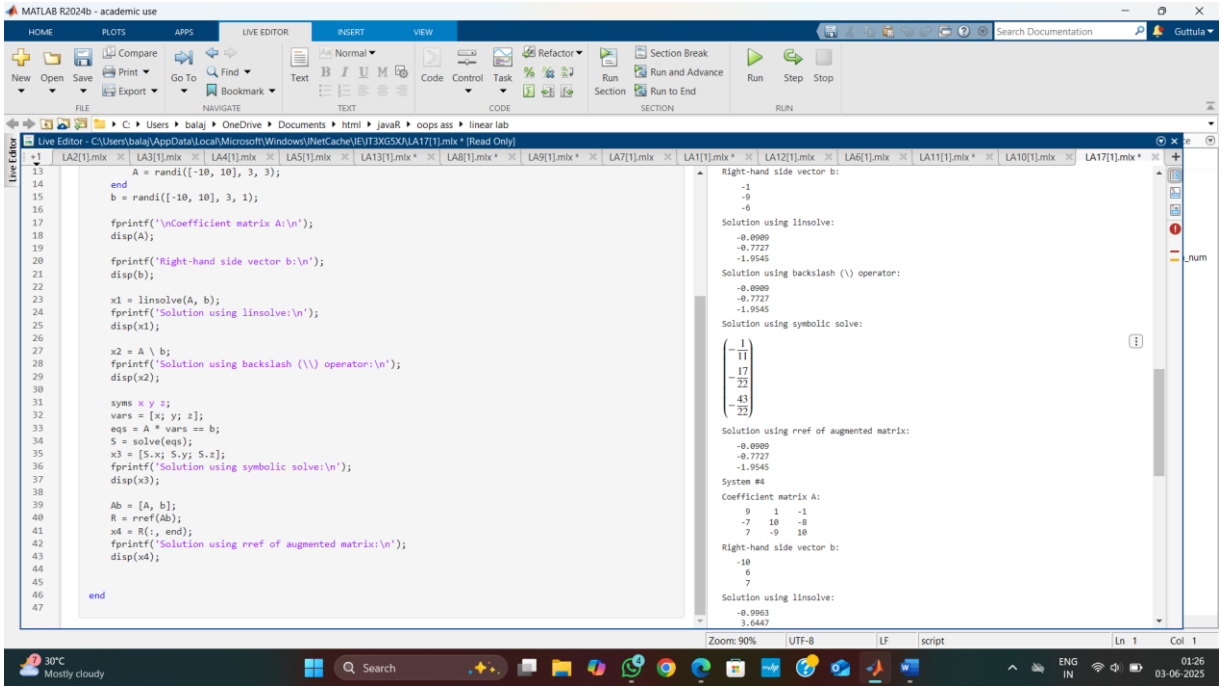


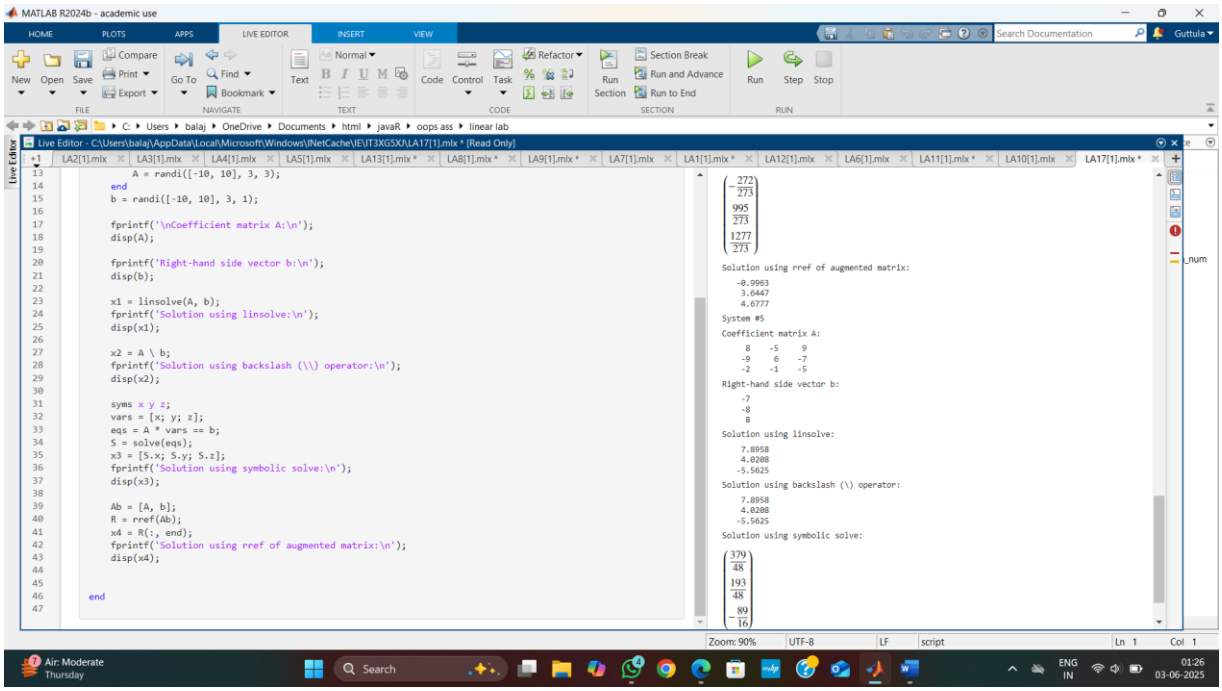


3. FOR ANY 5 SYS OF LINEAR EQN THAT WE SOLVEDSOLVE IT USING LINSOLVE BACKSLASH SOLVE AND ALSO FIND RREF

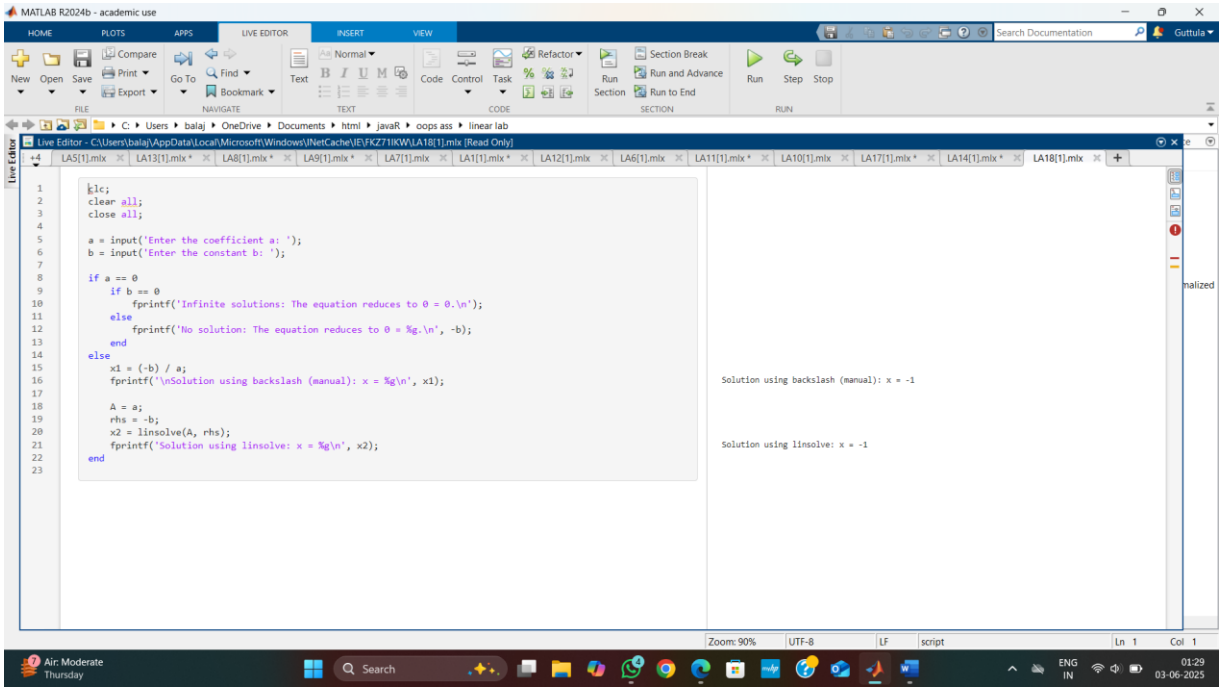



MATLAB R2024b - academic use
Live Editor - C:\Users\balaj\AppData\Local\Microsoft\Windows\INetCache\IE\IT3XG5XJ\LA17[1].mlx * [Read Only]
A = randi([-10, 10], 3, 3);
end
b = randi([-10, 10], 3, 1);
fprintf('\nCoefficient matrix A:\n');
disp(A);
fprintf('Right-hand side vector b:\n');
disp(b);
x1 = linsolve(A, b);
fprintf('Solution using linsolve:\n');
disp(x1);
x2 = A \ b;
fprintf('Solution using backslash (\\) operator:\n');
disp(x2);
syms x y z;
vars = [x; y; z];
eqs = A * vars == b;
S = solve(eqs);
x3 = [S.x; S.y; S.z];
fprintf('Solution using symbolic solve:\n');
disp(x3);
Ab = [A, b];
R = rref(Ab);
x4 = R(:, end);
fprintf('Solution using rref of augmented matrix:\n');
disp(x4);
end
-10
-3
-7
Solution using linsolve:
0.1093
-0.5355
1.1967
Solution using backslash (\) operator:
0.1093
-0.5355
1.1967
Solution using symbolic solve:
Solution using rref of augmented matrix:
0.1093
-0.5355
1.1967
System #3
Coefficient matrix A:
6 -7 3
-4 2 4
1 -5 5
Right-hand side vector b:
-1
-9
-6
Solution using linsolve:
-0.0909
-0.7727
-1.9545
Zoom: 90%
UTF-8
LF
script
Ln 1
Col 1
30°C
Mostly cloudy
Search
ENG
IN
01:25
03-06-2025

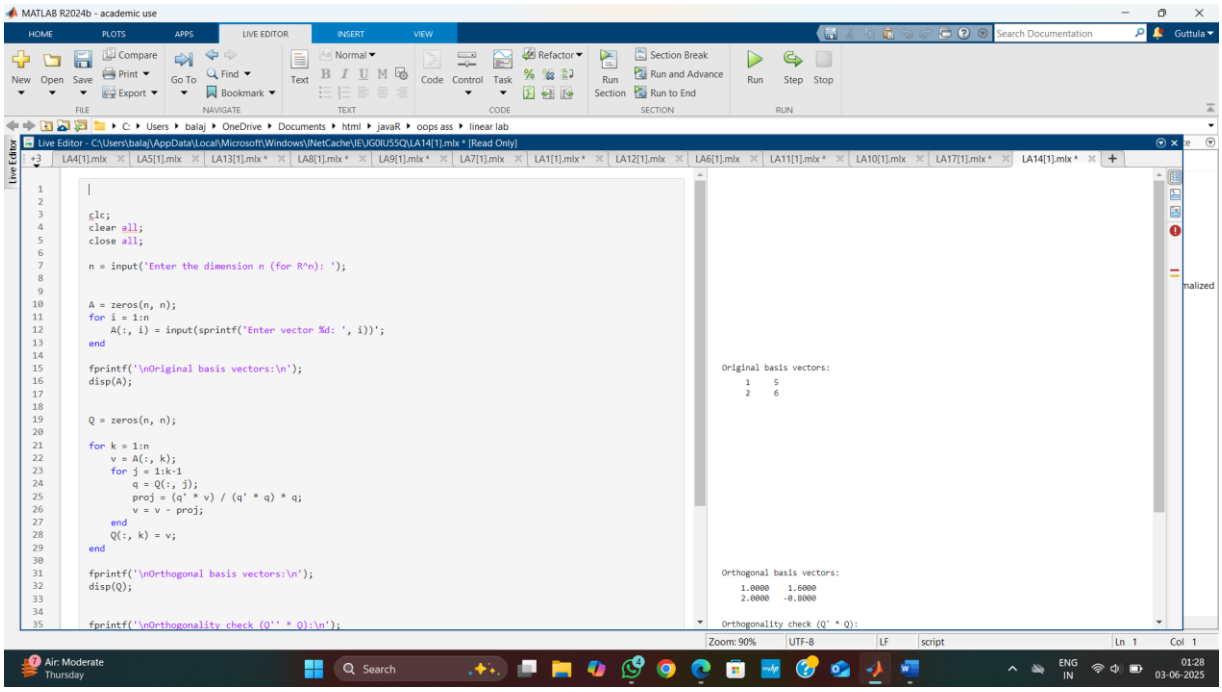

MATLAB R2024b - academic use
Live Editor - C:\Users\balaj\AppData\Local\Microsoft\Windows\INetCache\IE\IT3XG5XJ\LA17[1].mlx * [Read Only]
A = randi([-10, 10], 3, 3);
end
b = randi([-10, 10], 3, 1);
fprintf('\nCoefficient matrix A:\n');
disp(A);
fprintf('Right-hand side vector b:\n');
disp(b);
x1 = linsolve(A, b);
fprintf('Solution using linsolve:\n');
disp(x1);
x2 = A \ b;
fprintf('Solution using backslash (\\) operator:\n');
disp(x2);
syms x y z;
vars = [x; y; z];
eqs = A * vars == b;
S = solve(eqs);
x3 = [S.x; S.y; S.z];
fprintf('Solution using symbolic solve:\n');
disp(x3);
Ab = [A, b];
R = rref(Ab);
x4 = R(:, end);
fprintf('Solution using rref of augmented matrix:\n');
disp(x4);
end
Right-hand side vector b:
-1
-9
-6
Solution using linsolve:
-0.0909
-0.7727
-1.9545
Solution using backslash (\) operator:
-0.0909
-0.7727
-1.9545
Solution using symbolic solve:
Solution using rref of augmented matrix:
-0.0909
-0.7727
-1.9545
System #4
Coefficient matrix A:
9 1 -1
-7 10 -8
7 -9 10
Right-hand side vector b:
-10
6
7
Solution using linsolve:
-0.9963
3.6447
Zoom: 90%
UTF-8
LF
script
Ln 1
Col 1
30°C
Mostly cloudy
Search
ENG
IN
01:26
03-06-2025

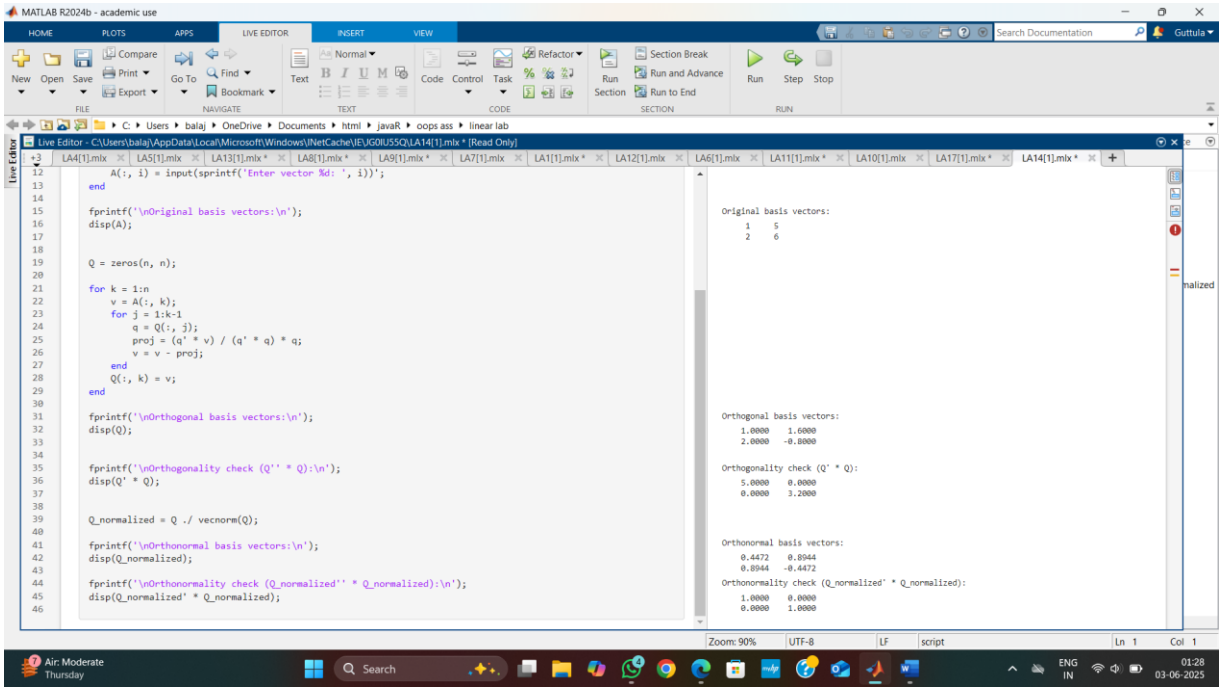


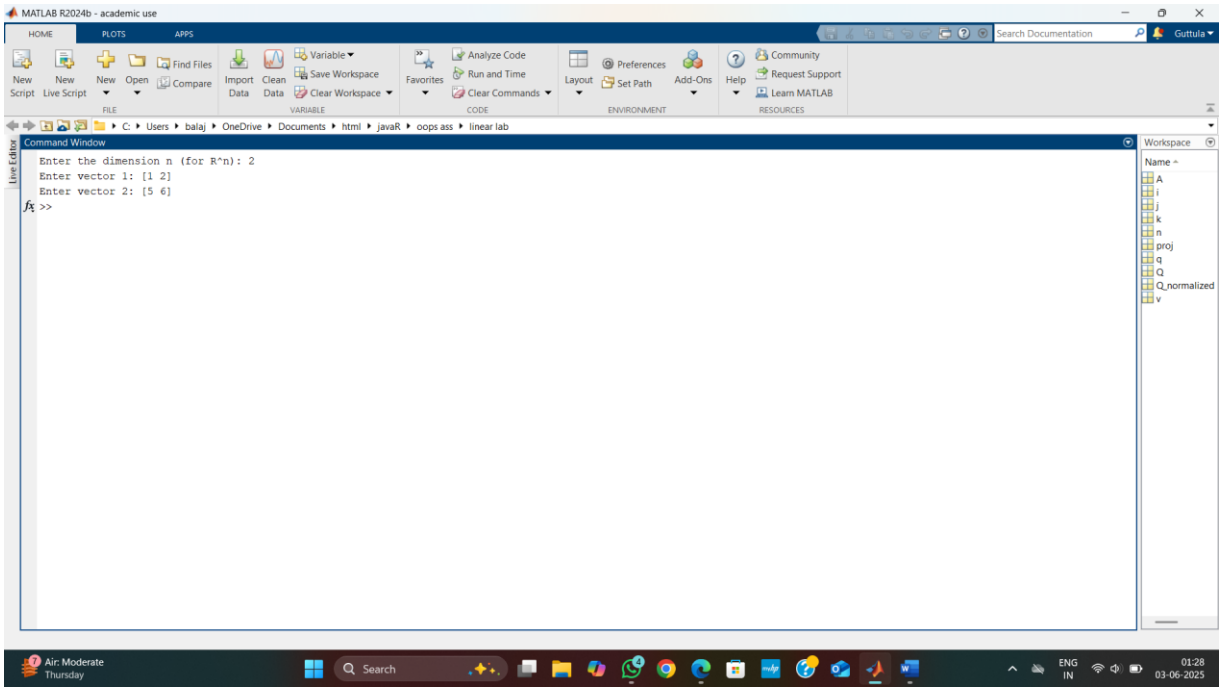
4.SOLVE THE EQUATION AX+B=0 FOR THE VARIABLE X USING THE COMMAND LINSOLVE OR BACKSLASH OPERATOR



5. FOR ANY GIVEN POSITIVE INTEGER N WRITE A MATLAB PROGRAM TO FIND AN ORTHONORMAL BASIS FROM A GIVEN BASIS FOR R^N.






MATLAB R2024b - academic use
HOME
PLOTS
APPS
Search Documentation
Guttula
New Script
New Live Script
New
Open
Find Files
Compare
Import Data
Clean Data
Variable
Save Workspace
Clear Workspace
Favorites
Analyze Code
Run and Time
Clear Commands
Layout
Preferences
Set Path
Add-Ons
Help
Community
Request Support
Learn MATLAB
FILE
VARIABLE
CODE
ENVIRONMENT
RESOURCES
C: ▸ Users ▸ balaj ▸ OneDrive ▸ Documents ▸ html ▸ javaR ▸ oops ass ▸ linear lab
Live Editor
Command Window
Enter the dimension n (for R^n): 2
Enter vector 1: [1 2]
Enter vector 2: [5 6]
fx >>
Workspace
Name
A
i
j
k
n
proj
q
Q
Q_normalized
v
Air: Moderate
Thursday
Search
ENG
IN
01:28
03-06-2025