Table of Contents


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CODE CHALLENGE 10 - Gaussian Elimination
%
% This challenge is an exercise in applying Gaussian Elimination in
 order
% to solve a system of equations. The system of equations you are
 looking
% to solve is as follows:
%        7x + 3y - 17z = 13
%       -4x + 2z = -2
%        4x + 3y - 9z = -5
%
%
% NOTE: DO NOT change any variable names already present in the code.
%
% Upload your team's script to Gradescope when complete.
%
% NAME YOUR FILE AS Challenge10_Sec{section number}_Group{group
 breakout #}.m
% ***Section numbers are 1 or 2***
% EX File Name: Challenge10_Sec1_Group4.m
%
% STUDENT TEAMMATES
% 1) Zak Reichenbach
% 2) Anna Casillas
% 3) Jack Iribarren
% 4) Lucas House
% 5) Tristan Workman
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Housekeeping

```
clear all;
close all;
clc;
```

Organizing Known Values

```
coeffs1 = [7 3 -17]; % Coefficients of equation 1 in x y z order
coeffs2 = [-4 0 2]; % Coefficients of equation 2 in x y z order
coeffs3 = [4 3 -9]; % Coefficients of equation 3 in x y z order

answers = [13; -2; -5]; % Answers to the three equations

AMat = [coeffs1;coeffs2;coeffs3]; % Creating an A-matrix using the
 given coefficients
[m,n] = size(AMat); % m for the number of rows in your A-matrix, n for
 the number of columns

[L,U] = lu(AMat);
```

Reducing the A-matrix using Recursive Method

Initialize the new A matrix, called AA, and b matrix, called bb

```
aa = AMat;
bb = answers;
a = AMat;
% Using the Recursive Method to reduce our A-matrix, starting at step
 1, row 2
r = 1; % step number
for k = 2:m % Total number of iterations that must be taken (Steps)
    for i = r+1:m % Marker for rows
        for j = r:n % Marker for columns
            scale_factor(j) = a(i,r)/a(r,r);
            aa(i,j) = a(i,j)-a(r,j)*scale_factor(j); % Define the new
 values of the A matrix
        end

        bb(i) = answers(i) - answers(r)*scale_factor(j); % Define the
 new values of the b matrix
    end
    a = aa; % Set the new reference a matrix
    answers = bb; % Set the new reference b matrix
    r = r+1; % Increase the step number
end
```

Solving for variables with Back Substitution

```
aa(1,:) = AMat(1,:);
%Xn = bn/ann
z = answers(3)/aa(3,3);
y = (answers(2) - (aa(2,3) * z)) / aa(2,2);
x = (answers(1)- (aa(1,3) * z) - aa(1,2) * y) / aa(1,1);


% x = nan; % Solving for x
% z = nan; % Back-substituting x to solve for z
```

```
% y = nan; % Back-substituting x and z to solve for y

% Outputting answers
fprintf('The calculated values for our variables are:\n')
fprintf('    x = %.4f\n',x)
fprintf('    y = %.4f\n',y)
fprintf('    z = %.4f\n\n',z)
```

```
The calculated values for our variables are:
    x = -0.7692
    y = -8.2564
    z = -2.5385
```

Sanity Checks

answers one = AMat(1,1)*x+AMat(1,2)*y+AMat(1,3)*z two = AMat(2,1)*x+AMat(2,2)*y+AMat(2,3)*z three = AMat(3,1)*x+AMat(3,2)*y+AMat(3,3)*z bb one = aa(1,1)*x+aa(1,2)*y+aa(1,3)*z two = aa(2,1)*x+aa(2,2)*y+aa(2,3)*z three = aa(3,1)*x+aa(3,2)*y+aa(3,3)*z

BONUS:

```
Solve the system using matrix methods, with the "\" operator and the
"inv()" operator. Compare the time it takes for these two methods with
the time that Gaussian elimination took.
```

```
% Using \ operator
tic
answersBackslash = nan;
timeBackslash = toc;

% Using matrix inverse
tic
answersInverse = nan;
timeInverse = toc;
```

Published with MATLAB® R2019a