MAS3114 MATLAB Assignment 3

Name: Samantha Bennett

UF ID: ####-####

WARNING: You will recieve a zero on this assignment if you fail to enter your name, UF ID, and the seed in the random number generator.

```
clear
% Initialize the random number generator by typing the command
rng(########,'twister')
% where ######## must be your 8-digit UF ID number
```

Exercise 1 -- Solving a linear system Ax = b, where A is 3 x 3

Extra Credit (3 points): Do research online and compare two commands A\b and inv(A)*b.

```
n = randi([5,10]);
n = n*1000;
A = randi([-8, 8], n, n);
b = randi([-8, 8], n, 1);
% Elapsed time for A\b
tic
 A\b;
toc
```

```
Elapsed time is 20.267601 seconds.
```

```
% Elapsed time for inv(A)*b
tic
 inv(A)*b;
toc
```

```
Elapsed time is 78.902776 seconds.
```

- When A is an n x n invertible matrix, which command is faster and which command is more accurate when calculating the unique solution?
- In general, A\b returns ...

1A

```
A = [1 2 3; 4 5 6; 7 8 9]
```

```
A = 3×3
     1     2     3
     4     5     6
     7     8     9
```

```
rref(A)
```

```
ans = 3×3
```

```
     1     0    -1
     0     1     2
     0     0     0
```

A is not invertiable because rref(A) does not have a pivot position in each column and row (rank(A) does not equal n).

Conclusion: det(A) = 0

```
% check determinant of A using MATLAB
det(A)
```

```
ans = 6.6613e-16
```

```
det(sym(A))
```

ans = 0

```
% Use MATLAB code to find the determinant of A transpose
AT = [1 4 7; 2 5 8; 3 6 9];
det(sym(AT))
```

ans = 0

Conclusion:

- $\det A^T =$ det A = 0
- A^T is singular when A is not invertible. Provide a reason based on the matrix A (see the property mentioned in the lecture). Since det(A) = 0, A is singular. Since det(A) = det(A^T), det(A^T) = 0. Therefore, A^T is singular as well.

1B

```
A1 = [1 2 2; 0 1 4; 1 1 -2]
```

```
A1 = 3×3
     1     2     2
     0     1     4
     1     1    -2
```

```
b1 = [5; 5; 0]
```

```
b1 = 3×1
     5
     5
     0
```

```
% Solve the system A1x = b1 using RREF
Partic_Sol(A1, b1)
```

```
Ax = b has infinitely many solutions, and below is
the particular solution by setting free variables as zeros
ans = 3×1
    -5
```

```
     5
     0
```

```
% Solve the system using x=A^(-1)*b
x = A1\b1
```

```
Warning: Matrix is singular to working precision.
x = 3×1
   NaN
   NaN
   NaN
```

```
% Solve the system using Cramer's rule
CramersRule3x3(A1, b1)
```

```
Matrix A is singular, so Cramer's rule cannot be applied.
```

Conclusion: The system $A_1\mathbf{x} = \mathbf{b}_1$ is consistent, and its solution is $x_3\begin{bmatrix} 6 \\ -4 \\ 1 \end{bmatrix} + \begin{bmatrix} -5 \\ 5 \\ 0 \end{bmatrix}$.

1C

```
% repeat the process for A1x = b2
b2 = [5; 1; 0]
```

```
b2 = 3×1
     5
     1
     0
```

```
Partic_Sol(A1, b2)
```

```
Ax = b has no solution.

ans =

     []
```

```
x = A1\b2
```

```
Warning: Matrix is singular to working precision.
x = 3×1
   NaN
   Inf
  -Inf
```

```
CramersRule3x3(A1, b2)
```

```
Matrix A is singular, so Cramer's rule cannot be applied.
```

Conclusion: The system $A_1\mathbf{x} = \mathbf{b}_2$ is inconsistent, and its solution is N/A (has no solution).

1D

```
% repeat the process for A2x = b2
A2 = [1 2 2; 0 1 4; 1 1 2]
```

```
A2 = 3×3
     1     2     2
     0     1     4
     1     1     2
```

```
Partic_Sol(A2, b2)
```

```
Ax = b has a unique solution.
ans = 3×1
    -3
     5
    -1
```

```
x = A2\b2
```

```
x = 3×1
    -3
     5
    -1
```

```
CramersRule3x3(A2, b2)
```

```
ans = 3×1
   -3.0000
    5.0000
   -1.0000
```

Conclusion: The system $A_2\mathbf{x} = \mathbf{b}_2$ is consistent, and its solution is $\begin{bmatrix} -3 \\ 5 \\ -1 \end{bmatrix}$.

Exercise 2 -- Solving a linear system Ax = b, where A is m x n

2A see the file CramersRule.m

2B

```
% Create three random systems and solve each system using the three methods
A1 =  randi([-8,8],6,3)
```

```
A1 = 6×3
     4    -5     6
    -1     8    -5
    -5     8    -4
    -2    -8     4
    -8    -5     8
     8     2     7
```

```
b1 = randi([-8,8],6,1)
```

```
b1 = 6×1
    -7
     5
     8
     7
     8
    -3
```

```
A2 =  randi([-8,8],3,6)
```

```
A2 = 3×6
    -1     2     1     2     7     2
     4     4     0    -5     8    -1
     6     0    -7    -2     4    -8
```

```
b2 = randi([-8,8],3,1)
```

```
b2 = 3×1
     5
     4
     8
```

```
A3 =  randi([-8,8],4,4)
```

```
A3 = 4×4
     8    -3    -2    -7
    -3    -6     2    -4
    -5    -5    -7     5
    -7    -5    -6    -7
```

```
b3 = randi([-8,8],4,1)
```

```
b3 = 4×1
     8
    -3
     2
     4
```

```
% Solve system A1x = b1
Partic_Sol(A1, b1)
```

```
Ax = b has no solution.

ans =

     []
```

```
x = A1\b1
```

```
x = 3×1
   -1.0788
    0.3879
    0.3382
```

```
CramersRule(A1, b1)
```

```
Matrix A is not square, so Cramer's rule cannot be applied.
```

```
% Solve system A2x = b2
Partic_Sol(A2, b2)
```

```
Ax = b has infinitely many solutions, and below is
the particular solution by setting free variables as zeros
ans = 6×1
   -1.9333
```

```
    2.9333
   -2.8000
         0
         0
         0
```

```
x = A2\b2
```

```
x = 6×1
         0
         0
         0
    0.5787
    0.7665
   -0.7614
```

```
CramersRule(A2, b2)
```

```
Matrix A is not square, so Cramer's rule cannot be applied.
```

```
% Solve system A3x = b3
Partic_Sol(A3, b3)
```

```
Ax = b has a unique solution.
ans = 4×1
    0.5119
    0.1375
   -0.9884
   -0.3343
```

```
x = A3\b3
```

```
x = 4×1
    0.5119
    0.1375
   -0.9884
   -0.3343
```

```
CramersRule(A3, b3)
```

```
ans = 4×1
    0.5119
    0.1375
   -0.9884
   -0.3343
```

2C Observe all results from both Exercises 1 and 2.

- When A is invertible, which method(s) returns a unique solution **x**? Cramer's Rule, x = A/b (dividing matrix A by matrix b to solve for x, using the equation Ax = b), partic_soln: using rank(A) == n or the number of pivots columns == number of columns in a matrix, find where the matrix have no soluntions, infinite solutions, or one solution. Then finding the solution, if the matrix only has one, by making a vector of just the last column once the matrix is in rref (the answers).
- When A**x** = **b** is consistent where A is an n x n nonnvertible matrix, , which method(s) returns a correct solution **x**? partic soln: specifics are stated above

- When A**x** = **b** is consistent where A is not a square matrix (m x n), , which method(s) returns a correct solution **x**? partic soln: specifics are stated above

Exercise 3 -- The Rank Theorem

3A

```
A1 % display A1 first
```

```
A1 = 6×3
     4    -5     6
    -1     8    -5
    -5     8    -4
    -2    -8     4
    -8    -5     8
     8     2     7
```

```
N1 = null(A1,'r') % each column represents a vector in a basis for Nul A1
```

```
N1 =

  3×0 empty double matrix
```

```
[B, pivcol] = rref(A1)
```

```
B = 6×3
     1     0     0
     0     1     0
     0     0     1
     0     0     0
     0     0     0
     0     0     0
pivcol = 1×3
     1     2     3
```

```
% write a single command for column space of A1 so that each column represents a vector in a ba
C1 = A1(:, pivcol)
```

```
C1 = 6×3
     4    -5     6
    -1     8    -5
    -5     8    -4
    -2    -8     4
    -8    -5     8
     8     2     7
```

```
% write a single command for row space of A1so that each row represents a vector in a basis for
R1 = B(1:rank(A1), :)
```

```
R1 = 3×3
     1     0     0
     0     1     0
     0     0     1
```

3B

```
% repeat the process for A2 and A3
```

```
%
A = [1 0 -3 0 0 2; 1 0 0 0 0 6; 0 0 0 0 0 4; 0 0 3 0 0 0]
```

```
A = 4×6
     1     0    -3     0     0     2
     1     0     0     0     0     6
     0     0     0     0     0     4
     0     0     3     0     0     0
```

```
% repeat the process for A
N1 = null(A,'r') % each column represents a vector in a basis for Nul A1
```

```
N1 = 6×3
     0     0     0
     1     0     0
     0     0     0
     0     1     0
     0     0     1
     0     0     0
```

```
[B, pivcol] = rref(A)
```

```
B = 4×6
     1     0     0     0     0     0
     0     0     1     0     0     0
     0     0     0     0     0     1
     0     0     0     0     0     0
pivcol = 1×3
     1     3     6
```

```
% write a single command for column space of A1 so that each column represents a vector in a ba
C1 = A(:, pivcol)
```

```
C1 = 4×3
     1    -3     2
     1     0     6
     0     0     4
     0     3     0
```

```
% write a single command for row space of A1 so that each row represents a vector in a basis fo
R1 = B(1:rank(A), :)
```

```
R1 = 3×6
     1     0     0     0     0     0
     0     0     1     0     0     0
     0     0     0     0     0     1
```

```


A2;
N2 = null(A2,'r') % each column represents a vector in a basis for Nul A1
```

```
N2 = 6×3
    1.9667    1.6667    0.6333
   -0.7167   -3.6667   -0.3833
    1.4000    2.0000   -0.6000
    1.0000         0         0
         0    1.0000         0
         0         0    1.0000
```

```
[B2, pivcol] = rref(A2)
```

```
B2 = 3×6
    1.0000         0         0   -1.9667   -1.6667   -0.6333
         0    1.0000         0    0.7167    3.6667    0.3833
         0         0    1.0000   -1.4000   -2.0000    0.6000
pivcol = 1×3
     1     2     3
```

```
% write a single command for column space of A1 so that each column represents a vector in a ba
C2 = A2(:, pivcol)
```

```
C2 = 3×3
    -1     2     1
     4     4     0
     6     0    -7
```

```
% write a single command for row space of A1 so that each row represents a vector in a basis fo
R2 = B2(1:rank(A2), :)
```

```
R2 = 3×6
    1.0000         0         0   -1.9667   -1.6667   -0.6333
         0    1.0000         0    0.7167    3.6667    0.3833
         0         0    1.0000   -1.4000   -2.0000    0.6000
```

```


A3;
N3 = null(A3,'r') % each column represents a vector in a basis for Nul A1
```

```
N3 =

  4×0 empty double matrix
```

```
[B3, pivcol] = rref(A3)
```

```
B3 = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
pivcol = 1×4
     1     2     3     4
```

```
% write a single command for column space of A1 so that each column represents a vector in a ba
C3 = A3(:, pivcol)
```

```
C3 = 4×4
     8    -3    -2    -7
    -3    -6     2    -4
    -5    -5    -7     5
    -7    -5    -6    -7
```

```
% write a single command for row space of A1 so that each row represents a vector in a basis fo
R3 = B3(1:rank(A3), :)
```

```
R3 = 4×4
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

3C

```
rank(A1)
```

```
ans = 3
```

Verify the rank theorem:

1. dim(Col A1) = dim(Row A1) = 3
2. rank(A1) + dim(Nul A1) = 3 Is the sum equal to n? yes, since dim(Nul A1) = 0 and rank(A1) = 3

```
rank(A2)
```

```
ans = 3
```

Verify the rank theorem:

1. dim(Col A2) = dim(Row A2) = 3
2. rank(A2) + dim(Nul A2) = 6 Is the sum equal to n? yes, since dim(Nul A2) = 3 and rank(A2) = 3

```
rank(A3)
```

```
ans = 4
```

Verify the rank theorem:

1. dim(Col A3) = dim(Row A3) = 4
2. rank(A3) + dim(Nul A3) = 5 Is the sum equal to n? yes, since dim(Nul A3) = 1 and rank(A3) = 4

```
rank(A)
```

```
ans = 3
```

Verify the rank theorem:

1. dim(Col A) = dim(Row A) = 3
2. rank(A) + dim(Nul A) = 6 Is the sum equal to n? yes, since dim(Nul A) = 3 and rank(A) = 3

Extra Credit (3 points):

```
A = randi([-8,8],4,6)
```

```
A = 4×6
    -3     0     7    -1     5     6
    -3    -3    -5    -7    -3    -4
     7     1     6    -6     8    -5
     4    -2    -5     6    -4     5
```

```
b = randi([-8,8],4,4)
```

```
b = 4×4
    -1     2    -3     5
    -4    -3     4    -2
```

```
     8     5    -2    -8
    -7    -6    -3     3
```

```
Partic_Sol(A, b)
```

```
Ax = b has infinitely many solutions, and below is
the particular solution by setting free variables as zeros
ans = 6×1
    0.3318
    2.5623
   -0.0873
   -0.6065
         0
         0
```

```
N = null(A,'r')
```

```
N = 6×2
   -0.2810    0.2821
    0.1063    2.2220
   -0.8033   -0.8812
    0.2200   -1.0151
    1.0000         0
         0    1.0000
```

```
% Does the system have infinitely many solutions?
% If not, generate another random system Ax = b until it has infinitely many
%solutions.
```

The general solution to Ax = b is (using the output from Partic_Sol and N)