Practical Problems 6 – Matrices & Systems of Equations

1. Use the **eye** function to create a 4 x 5 diagonal matrix called ***A*** with 7's on the leading diagonal.

```
A =

     7     0     0     0     0
     0     7     0     0     0
     0     0     7     0     0
     0     0     0     7     0
```

2. Create a 3 x 3 elementary matrix called ***E*** that adds 3 of the 1st row to the 3rd row of a matrix. Test it with a random 3 x 3 matrix of integers.

```
Original Matrix:
    10    10     2
     2     5     5
    10     9    10

After Row Operation:
    10    10     2
     2     5     5
    40    39    16
```

3. Augment the test matrix from question 2 with a 3 x 3 identity matrix then use Matlab's **rref** function to find it's reduced echelon form.

```
Augmented Matrix:
    10    10     2     1     0     0
     2     5     5     0     1     0
    10     9    10     0     0     1

RREF:
    1.0000         0         0    0.0175   -0.2867    0.1399
         0    1.0000         0    0.1049    0.2797   -0.1608
         0         0    1.0000   -0.1119    0.0350    0.1049
```

4. Create the coefficient matrix for the following linear system, check that its determinant is not 0, then calculate the solution using the inverse matrix method (*hint:* you can find the inverse of a matrix with the **inv** function).

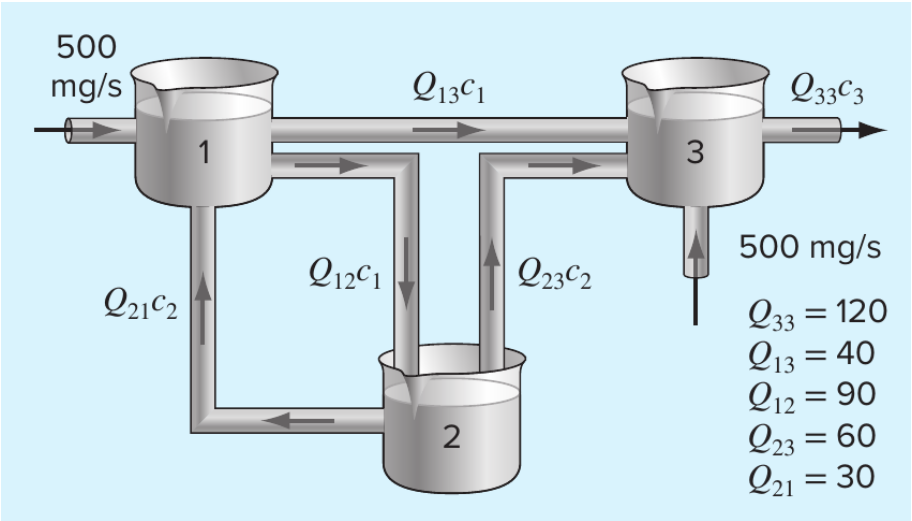
$$3x_1 + 4x_2 = 3$$
$$5x_1 + 6x_2 = 7$$
$$\mathbf{x} = A^{-1}\mathbf{b}$$

```
The solution is:
x1 = 5
x2 = -3
```

5. Solve the system above but using Matlab's left-division operator.

```
The solution is:
x1 = 5
x2 = -3
```

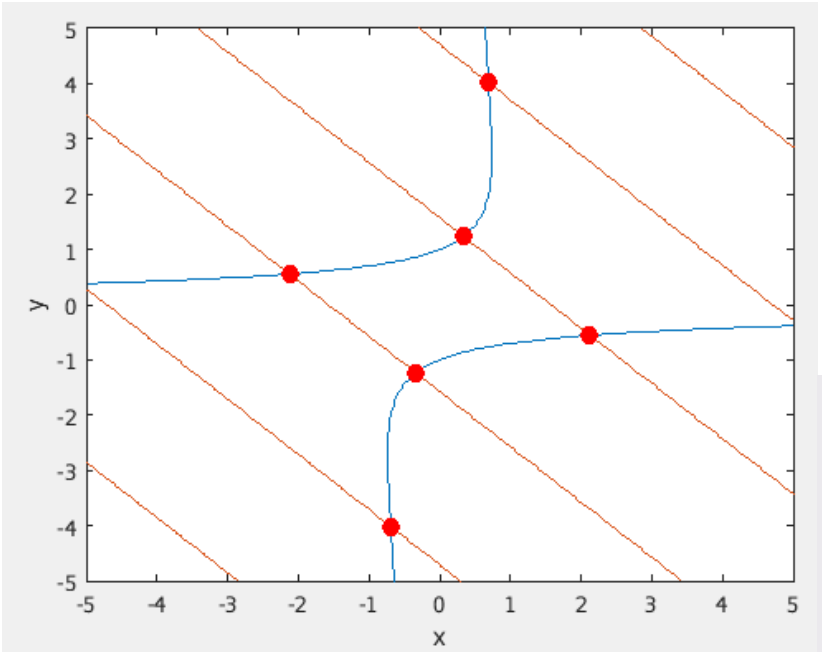
6. There are 3 reactors linked by pipes as shown below. The rate of transfer of chemicals through each pipe is equal to a flow rate (Q, with units of cubic meters per -second) multiplied by the concentration of the reactor from which the flow originates (c, with units of milligrams per cubic meter). If the system is at a steady state, the transfer into each reactor will balance the transfer out. Develop mass-balance equations for the reactors and solve the three simultaneous linear algebraic equations for their concentrations.




```
Command Window
>> Q6_Sol
The value of c1 is 5.0000
The value of c2 is 5.0000
The value of c3 is 8.3333
```

7. Use Matlab's **fsolve** function to find the roots of the system. Since there are infinite solutions you do not need to find every root. Just locate and plot those given in the default window. Plotting is always a good idea to give you an idea of how many roots you are looking for and where to make the initial guesses. Use the **fimplicit** function to plot these implicit functions on the same graph. Define a function at the end of your script closed with keyword **end** that holds the system of equations. Advanced users can also use the ginput() function to pick starting initial guesses.

$$e^{xy} - y^2 = 0$$
$$\cos(x + y) = 0$$




```
The roots are:
    0.6922    4.0202
    0.3387    1.2320
   -2.1254    0.5546
   -0.6922   -4.0202
   -0.3387   -1.2320
    2.1254   -0.5546
```

8. Find a root of the following system. Check your solution.

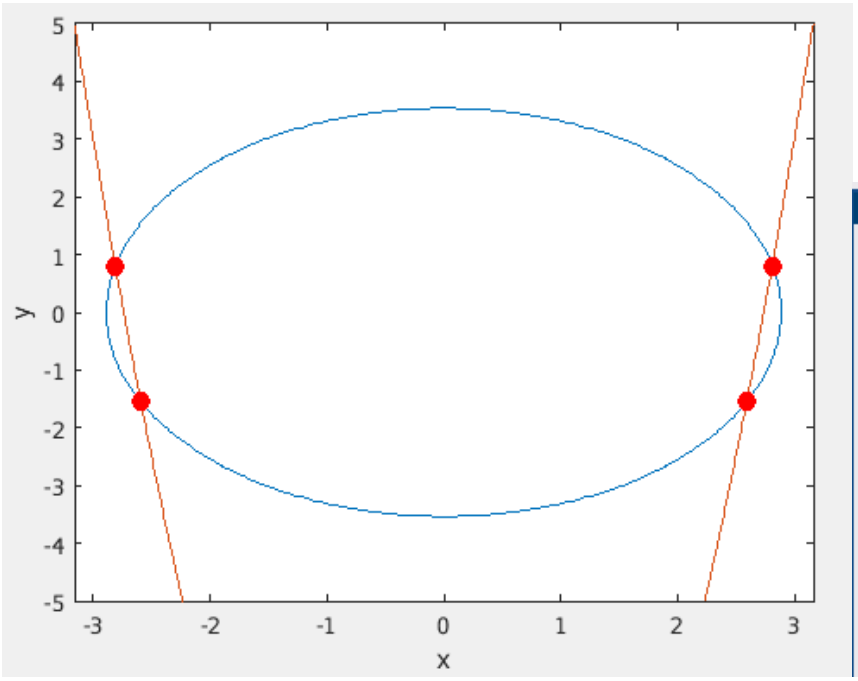
$$3x - \cos yz = \frac{3}{2}$$
$$4x^2 - 625y^2 + 2z = 1$$
$$20z + e^{-xy} = -9$$

```
The roots are:
    0.8333   -0.0352   -0.5015

Substituting into the first equation gives:-4.4409e-16
Substituting into the second equation gives:-1.3567e-13
Substituting into the third equation gives:0
```

9. Write a script that uses the Newton Method for Nonlinear Systems to solve the following nonlinear system. Try to find the roots of the system with your script. Remember you'll have to compute the Jacobian yourself by hand for now in order to put it into your script.

$$3x^2 + 2y^2 = 25$$
$$2x^2 - y = 15$$




```
Command Window
The roots are:
    2.8111
    0.8042

    2.5929
   -1.5542

   -2.5929
   -1.5542

   -2.8111
    0.8042
```

10. Write a script that reduces a 3 x 3 matrix to an echelon form (don't worry about pivoting). Test it on the following matrix:

$$A = \begin{bmatrix} 1 & -3 & 0 \\ 0 & 1 & 3 \\ 2 & -10 & 2 \end{bmatrix}$$

```
Command Window
>> Q10_Sol
     1    -3     0
     0     1     3
     0     0    14
```

11. Write a script that reduces the above matrix to reduced echelon form (RREF).

```
Command Window
>> Q11_Sol
     1     0     0
     0     1     0
     0     0     1
```

12. Write a script that finds the LU factorisation of the matrix ***A*** in question 10 using Gaussian elimination to find ***U*** and the formulae given in lecture notes 6 to find ***L***.

```
Command Window
The original matrix is:
     1    -3     0
     0     1     3
     2   -10     2

A can be written as the product LU where:
L =
     1     0     0
     0     1     0
     2    -4    14

U =
     1    -3     0
     0     1     3
     0     0     1
```

13. Solve the following system using LU factorisation by finding ***L*** and ***U*** as in question 12.

$$\begin{aligned} x_1 - 3x_2 &= -5 \\ x_2 + 3x_3 &= -1 \\ 2x_1 - 10x_2 + 2x_3 &= -20 \end{aligned}$$

```
Command Window
The solution is:
     1
     2
    -1
```

14. Solve the above system using LU factorisation but use Matlab's built in **lu** function. Note you must account for the permutation matrix given by Matlab.

```
Command Window
The solution is:
     1
     2
    -1
```

15. **Challenge Problem**

Write your own function file that calculates the inverse of a square matrix. The input should be a square matrix of any size and it should have sufficient checks to make sure the user inputs an invertible matrix or return an error/warning.

I tested mine with randomly generated matrices then multiplied my result for the inverse with the original matrix to test if I got the identity matrix back in order to verify the solution.

My code includes an error check if the matrix is not square or if the determinant is 0,

```
Command Window

A =

    10     6    10     4    -7
    -7    -8     3     5     4
    10    -2   -10     5   -10
    10     9     7    -2    -5
     0     6     9     3   -10

The inverse is A^(-1):
   -0.0637    0.1103    0.0509    0.1900   -0.0572
    0.6069   -0.5884   -0.1946   -0.8241   -0.0535
   -0.2148    0.2518    0.0393    0.3518    0.0359
    0.7049   -0.5136   -0.1601   -0.9043   -0.0866
    0.3822   -0.2805   -0.1295   -0.4491   -0.1257

Check A*Ainv:

ans =

    1.0000   -0.0000   -0.0000    0.0000    0.0000
    0.0000    1.0000   -0.0000   -0.0000   -0.0000
         0   -0.0000    1.0000    0.0000    0.0000
   -0.0000    0.0000    0.0000    1.0000    0.0000
    0.0000    0.0000   -0.0000         0    1.0000
```