MATH1019 Linear Algebra and Statistics for Engineers
Laboratory Session 7

Learning outcomes for this session

At the end of this session, you will be able to

1. Solve systems by Gaussian Elimination (i.e. reduction to echelon form).
2. Calculate ranks.
3. Solve systems directly by using `A\b`.

Overview

1. Using matlab to enhance our understanding of solving systems of equations and calculating ranks.
2. Calculate ranks.
3. Some of the matlab commands introduced in this lab: `rank`, `lu`

From row operations to rank and solving linear systems

1. In the command window define the matrix,
`AI =eye(3)`
Matlab is not really designed to do something as basic as row operations. But you can make it do so, as in the following examples.

```
AI([1 3],:) = AI([3 1],:)  % swaps row1 and row3
P=AI
```

The matrix P is called a permutation matrix. Show the P^2 (in matlab `P*P`) is the identity matrix.
Define,

```
M= [1,2,3,4;5,6,7,8;9,10,11,12]
```

Let's now do some other row operations, performing them on the matrix M. The comments, after the %, are in notation closer to that in your lectures.

```
M0=M;  % save a copy of M as the next commands modify M
            % and will use M0 in a later exercise
M(2,:) = M(2,:)-5*M(1,:)  % row2 <- row2-5*row1
M(3,:) = M(3,:)-9*M(1,:)  % row3 <- row3 -9*row1
M(3,:)= M(3,:)-2*M(2,:)  % row3 <- row3-2*(the_present_state_of)row2
```

The rank of a matrix is the number of nonzero rows in the matrix when it's in row echelon form. Run the matlab command `rank(M)` and see that it agrees with what you have found with your row operations.

2. Define the matrices,

```
M1= [1,2,3,4;5,6,7,8;9,10,11,11]
M2= [1,2,3,4;5,6,7,8;9,10,12,12]
```

What are their ranks? As you have learnt in lectures, there are some nice properties of ranks. See also
https://en.wikipedia.org/wiki/Rank_(linear_algebra)

3. The system of linear equations $A\boldsymbol{x} = \boldsymbol{b}$ can be solved by reducing the augmented matrix $A|\boldsymbol{b}$ to row echelon form. Consider next systems $A_j\boldsymbol{x} = \boldsymbol{b}_j$ where the A_j matrices and the $\boldsymbol{b}_j$ column vectors are defined as follows.

```
A0=M0(:,[1:3]);  b0=M0(:,4)
A1=M1(:,[1:3]);  b1=M1(:,4)
A2=M2(:,[1:3]);  b2=M2(:,4)
```

(Note the M_j matrices are the augmented matrices of the systems $A_j\boldsymbol{x} = \boldsymbol{b}_j$, and you have already found their ranks.) Considerations of rank are useful in telling us if a system of linear equations $A\boldsymbol{x} = \boldsymbol{b}$ has solutions, and if so, how many. For example,
• if the rank of the augmented matrix $A|\boldsymbol{b}$ is not equal to the rank of the matrix A, there is no solution;
• if A is square (i.e. $n \times n$) and both A and the augmented matrix $A|\boldsymbol{b}$ are full rank (i.e. both ranks are equal to n), there is a unique solution.
Use the `rank` command to decide which of the systems have solutions.
Just one of the matrices A_j is full rank. For it find the solution of $A_j\boldsymbol{x} = \boldsymbol{b}_j$ first from Gaussian Elimination using row operations, and after that with a command of the form,

```
x= A\b
```

4. In engineering applications you rarely have to do hand calculations for solving linear systems. Matlab's `x=A\b` will do it for you. It is all nice and simple when A is a square matrix of full rank, but there are subtleties to watch in other circumstances. (Later in MATH1019 you will learn what matlab `x=A\b` does when A is not square.) Gaussian elimination is fundamental, simple (just the arithmetic can be boring) and it underpins matlab's solution methods (and is the first stage of Gauss-Jordan, `rref` next lab). Gaussian elimination in matlab is its `lu` command, the basic idea for which is that every square matrix A can be written as the product of 3 nice matrices, $A = P\,L\,U$ where P is a permutation matrix (swaps of rows), L is a unit lower triangular matrix and U is an upper triangular matrix. See, for example,
https://en.wikipedia.org/wiki/LU_decomposition
You could, if you wanted, follow the methods of their 2 by 2 matrix example, with this 2 by 2 matrix. Start with Gaussian elimination,

```
A = [8,3;4,6];   A0=A;  b=[-2;8];
A(2,:)= A(2,:)-(1/2)*A(1,:)
```

This gives a row echelon form (an upper triangular matrix). Gaussian elimination in matlab is lu factorization:

```
[L,U,P] = lu(A0)
```

Look at the upper triangular factor, U. It is the same as you found with the row operation above. Let's now consider the principles involved in solving $A\boldsymbol{x} = \boldsymbol{b}$ when A, as in our example, factors as $A = LU$. The basic idea of LU-factorization, and of Gaussian elimination (the same thing), is that solving triangular systems is easy. So if one solves $L\boldsymbol{y} = \boldsymbol{b}$ and then $U\boldsymbol{x} = \boldsymbol{y}$ the $\boldsymbol{x}$ so found solves $A\boldsymbol{x} = \boldsymbol{b}$. Now use this LU-factorization method to solve $A\boldsymbol{x} = \boldsymbol{b}$.