School of Mathematics and Statistics
MAST30028 Numerical Methods & Scientific Computing
Week 7

**Drag and drop the folders `Week6`, `Week7` from `L: \MAST30028` to `C:\...\MATLAB` and include it in the path.** Now MATLAB knows how to find the files in `Week7`. The folder includes a pdf file of Chapter 2 in Moler's textbook (`lu.pdf`).

## 1 LU factorization

This relates to material in Lecture 12.

### Exercise Set 1

a. Run the commands

```
A=rand(10,10);
[L,U]=GE(A);
norm(A-L*U)
```



which tests how "well" A has been factorized.

Note: this code is equivalent to Gauss elimination without pivoting. DO NOT use this code unless I tell you to, since you should usually pivot. We'll see later what `norm` does.

b. If the factorization worked well, you can go on to solve a system



```
[m n]=size(A);
b=A*ones(m,1);    % so you know the answer is all 1s
y = L\b;   % solves by forward substitution
x = U\y;   % solves by back substitution
norm(b-A*x)       % the residual
norm(x-ones(n,1))     % the forward error
```



c. MATLAB has a built-in `lu` command. Type `help lu` to see its calling sequence.

Repeat your tests as in parts a, b but with pivoting:

```
[L,U,P]=lu(A);
y = L\(P*b);     % why?
x = U\y;
```

What is the matrix $P$ in the above commands? What does it look like?

d. A textbook version of LU factorization (with pivoting) is `lutx` in the `Week7` folder. Type `help lutx` to see its calling sequence.

An explanation of each part of the code is given in Section 2.7 of Moler's textbook. A pdf file of Chapter 2 is in the `Week6` folder (`lu.pdf`).

Notice that it uses a permutation vector, not a permutation matrix. What is the difference?

What does the statement from lutx help `L*U = A(p,:)` mean? See `help colon` for ideas.

e. Type `lugui` at the command line to start a demo of using row operations to produce the LU factorization of a square matrix. Try both partial and complete pivoting to see how they work.

## 2 Operation counts

### Exercise Set 2

Write some MATLAB M-files to test the claims made in lectures regarding the running times of various algorithms. You might like to test :

a. how does LU factorization (using \ ) compare with solving by using the inverse matrix (i.e. `x = inv(A)*b`)?

b. how does Cholesky factorization, followed by forward and back-substitution, compare with solving a general dense matrix? Try both using `chol`/`lu` and using `\`.

   An example M-file is `benchmark.m`

## 3 Backslash

This relates to material in Lecture 12.

### Exercise Set 3

a. Cholesky factorization only works for symmetric, positive definite matrices. How could you easily generate test matrices for Cholesky factorization?

   A function that implements a simple algorithm for Cholesky factorization is `CholScalar` in `Week7`. By looking at the code, what do you think goes wrong if the matrix is symmetric but not positive definite? *Don't use this function to do Cholesky factorization — use the built-in* `chol` *instead.*

b. The most common way of solving a linear system in MATLAB is the *backslash operator* \ which invokes a polyalgorithm calling various methods depending on the structure of the matrix.

   To get an idea how it works, check out `bslashtx` from the `Week7` folder.

## 4 Vector and matrix norms



This relates to material in Lecture 13.

### Exercise Set 4



a. Compute by hand (or from the command line) $\|\mathbf{b}\|_1, \|\mathbf{b}\|_2, \|\mathbf{b}\|_\infty$ for the vector $\mathbf{b} = (3, -5, 2)$. Check your answers using MATLAB's `norm`.

b. Compute by hand (or from the command line) $\|\mathbf{A}\|_1, \|\mathbf{A}\|_M, \|\mathbf{A}\|_F, \|\mathbf{A}\|_\infty$ for the matrix $\mathbf{A} = \begin{bmatrix} 33 & -17 & 22 \\ 11 & -12 & -1 \\ 0 & -9 & 24 \end{bmatrix}$.



   Check your answers using MATLAB and also find $\|\mathbf{A}\|_2$ with MATLAB.