MA/CS 375
Fall 2002
Lecture 3
Example 2

A is a matrix with 3 rows and 2 columns.

\[
A = \begin{pmatrix}
2.1 & 3.23 \\
4.12 & 1.893 \\
7.1 & 5
\end{pmatrix}
\]
Close Up of Example 2

\[ A(2,1) = 4.12; \]

says:

set the entry of \( A \) in the 2\(^{nd} \) row and 1\(^{st} \) column to 4.12
Observations from Example 2

• In Matlab you do not need to specify the type of the variables unlike C

• Matlab keeps track of the size of matrices.

• Beware – unlike C, Matlab stores data by column first (board explanation)
Example 2 Revisited
We can build $\mathbf{A}$ directly with the following:

$\mathbf{A}$ is a matrix with 3 rows and 2 columns.

$$
\mathbf{A} = \begin{pmatrix}
2.1 & 3.23 \\
4.12 & 1.893 \\
7.1 & 5
\end{pmatrix}
$$
Example 3 in C

• For those of you familiar with C, this is similar code (but not identical)
  
  – double A[3][2] = [[2.1,3.23],[4.12,1.893],[7.1,5.0]];
Vectors

A vector is just a special case of a matrix.

If you require a vector of length 3, then:

```matlab
>> a = zeros(3,1);
>> a
a =
     0
     0
     0
```
Alternative ways to build vectors

```matlab
>> a = [1,2,3,4]'
a =
1
2
3
4

>> a = [1;2;3;4]
a =
1
2
3
4
```
Alternative Ways to Build Vectors

cont

```matlab
>> a = linspace(0, 2, 5)'

a =

   0
   0.5000
   1.0000
   1.5000
   2.0000
```
Now What?

• So we have found a few different ways to build matrices (and vectors).
• Well – we can now do some matrix-vector algebra.
• The following operators are allowed:
  - +, -, *, .*, /, \

Adding Matrices

- Recall if I wish to add two matrices together:
  \[ C_{ij} = A_{ij} + B_{ij} \]

- Where

\[
\begin{pmatrix}
A_{11} & A_{12} & A_{13} & \cdots & A_{1M} \\
A_{21} & A_{22} & A_{23} & \cdots & A_{2M} \\
A_{31} & A_{32} & A_{33} & \cdots & A_{3M} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
A_{N1} & A_{N2} & A_{N3} & \cdots & A_{NM}
\end{pmatrix}
\]

i’th row, j’th column
Matrix Addition in Index Notation

\[
C = A + B = \begin{pmatrix}
A_{11} + B_{11} & A_{12} + B_{12} & A_{13} + B_{13} & \cdots & A_{1M} + B_{1M} \\
A_{21} + B_{21} & A_{22} + B_{22} & A_{23} + B_{23} & \cdots & A_{2M} + B_{2M} \\
A_{31} + B_{31} & A_{32} + B_{32} & A_{33} + B_{33} & \cdots & A_{3M} + B_{3M} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
A_{N1} + B_{N1} & A_{N2} + B_{N2} & A_{N3} + B_{N3} & \cdots & A_{NM} + B_{NM}
\end{pmatrix}
\]
Matrix Addition in Matlab

\[
\begin{bmatrix}
2 & 1 \\
3 & 2
\end{bmatrix}

\begin{bmatrix}
4 & 2 \\
4 & 1
\end{bmatrix}

C = A + B

Random demo on the board, volunteers.
Notes on Matrix Addition

• If I want to add two matrices $A$ and $B$ then
  – the dimensions of $A$ and $B$ must be the same
  – i.e. $\#$ rows of $A = \#$ rows of $B$
    $\#$ columns of $A = \#$ columns of $B$

• what happens when we try to add to matrices of different dimensions in Matlab?

• Guesses?
Error Messages…

```
A = [2,1;3,2;1,4];
B = [4,2;4,1];
C = A+B;
??? Error using ==> +
Matrix dimensions must agree.
```

Volunteer to explain why this is the case.
Matrix Subtraction in Matlab

\[ A = \begin{pmatrix} 2 & 1 \\ 3 & 2 \end{pmatrix} \]

\[ B = \begin{pmatrix} 4 & 2 \\ 4 & 1 \end{pmatrix} \]

\[ C = A - B \]

Random demo on the board, volunteers.
Result of Matrix Subtraction

MATLAB Command Window

```
>> A = [[2,1];[3,2]];
>> B = [[4,2];[4,1]];
>> C = A-B

C =

    -2    -1
   -1     1
```
Matrix Multiplication

- There is a specific definition of matrix multiplication.

\[ \mathbf{C} = \mathbf{A} \ast \mathbf{B} \]

- In index notation:

\[ C_{ij} = \sum_{k=1}^{NcolsA} A_{ik} B_{kj} \]

- i.e. for the \((i,j)\) of the result matrix \(\mathbf{C}\) we take the \(i^{th}\) row of \(\mathbf{A}\) and multiply it, entry wise, with the \(j^{th}\) column of \(\mathbf{B}\)
Example 4
(matrix multiplication)

\[
A = \begin{pmatrix}
2 & 1 \\
3 & 2
\end{pmatrix}, \quad
B = \begin{pmatrix}
4 & 2 \\
4 & 1
\end{pmatrix}, \quad
C = A \ast B
\]

Volunteer?.
Result of Example 4

$$A = \begin{pmatrix} 2 & 1 \\ 3 & 2 \end{pmatrix}$$

$$B = \begin{pmatrix} 4 & 2 \\ 4 & 1 \end{pmatrix}$$

$$C = A \times B$$

MATLAB code:

```matlab
>> A = [2,1;3,2];
>> B = [4,2;4,1];
>> C = A*B;
>> C = A*B
```

```
C =
    12   5
    20   8
```
Matrix Division

• We will save the / and \ operators for later.
Functions in Matlab

- Matlab has a number of built-in functions:
  - cos, sin, tan, acos, asin, atan
  - cosh, sinh, tanh, acosh, asinh, atanh
  - exp, log, sqrt

- They all take matrices as arguments and return matrices of the same dimensions.
- e.g. cos([1 2 3])

- For other functions type: > help matlab\elfun
Example of Function of Vector

```matlab
» cos([1;2;3])
ans =
    0.5403
    0.4161
  -0.9900
```
Special Random Matrix Function

- Say I want to create a 3 by 2 matrix with random entries, with each entry between 0 and 1:

What will happen if I run this again?
Customizing Matlab
Changing Your “Working” Directory

• First make a directory you want to do store your results in on the Desktop

• Click on the … in the box next to the “Current Directory”
cont

• Next click on your directory (which should be somewhere on the desktop)

• For following lectures bring a 3.5 inch floppy diskette
Matlab is Yours to Command

- It is very likely that you will need to use the same string of commands, over and over.

- We can create an M-File which contains a set of commands which will be executed.
  
  - e.g. create a script file mygrid.m containing:

  ```
x1d = linspace( 0, 1, Npts);
y1d = linspace( 0, 1, Npts);
[x2d, y2d] = meshgrid( x1d, y1d);
```
Creating a Matlab M-File

Click on:

File/New/M-File
Editing a Matlab M-File

```matlab
% this is a script
% input: Npts
% output x1d = row vector of length Npts
% yld = row vector of length Npts
% x2d = matrix of size Npts x Npts
% y2d = matrix of size Npts x Npts

x1d = linspace( 0, 1, Npts);
y1d = linspace( 0, 1, Npts);
[ x2d, y2d] = meshgrid( x1d, y1d);
```
Saving Your M-File

Make sure you save it in your “Working Directory”
Calling Your Own *Script*

```matlab
» Npts = 20;
» mygrid;
» plot(x2d,y2d, '+');
»
```
Say we wish to create a function that turns Cartesian coordinates into polar coordinates. We can create a text file with the following text. It can be called like any built-in function.

```matlab
function [ radius, theta] = myfunc( x, y)

    % this is a comment, just like // in C++
    % now create and evaluate theta (in radians)
    theta  = atan2(y,x);

    % now create and evaluate radius
    radius = sqrt( x.^2 + y.^2);
```
Custom Built Function For Matlab

- Make sure you that the Matlab current directory is set to the directory containing `myfunc.m`.

- The arguments to `myfunc` could also have been matrices – leading to two matrices being output.

```matlab
>> [r, angle] = myfunc(2,3)

r =
    3.6056

angle =
    0.9828
```
Constraints on Custom-Built Matlab Functions

• Matlab follows similar *scope* rules to C.

• Without resorting to the Matlab command `global` the only variables available inside the function are those passed as arguments to the function, or created inside the function.

• Just in case you are thinking about using `global` – I consider it poor programming…
Loops in Matlab
Loops in Matlab

• Much like C or Fortran we can use a loop syntax to repeat a sequence of commands.

• HOWEVER, Matlab is an interpreted language and as such is not efficient in executing loop operations.

• In other words, using loops in Matlab is very slow!
Loops in Matlab

One variant of Matlab loop syntax is:

```matlab
for var=start:end
    commands;
end
```
Example of a Matlab Loop

• Say I want to add the numbers from 1 to 10, *without* using the Matlab intrinsic `sum`. 

```
» total = 0;
» for i=1:10
   total = total+i;
end
» total

total = 
   55
```

Summary of Lectures 2 and 3

• We have learnt how to:
  – run matlab
  – create matrices (and hence vectors)
  – set the entries of a matrix
  – add two matrices
  – subtract two matrices
  – multiply two matrices
  – loops
  – creating custom functions..