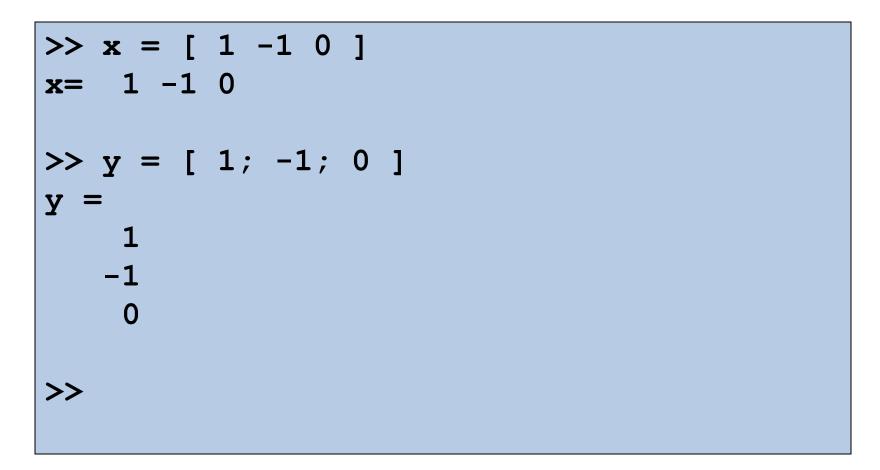
# Mathematical Concepts (G6012)

Lecture 13

Thomas Nowotny Chichester I, Room CI-105 Office hours: Tuesdays 15:00-16:45 T.Nowotny@sussex.ac.uk

#### **DEMO** Vectors in Matlab



#### **DEMO** Matrices in Matlab

>>	A =	[ 1	-1	0;	1	2	3;	3	-1	2	]	
A=												
	1	-1	0									
	1	2	3									
	3	-1	2									
>>	A*y											
ans	s =											
	2											
	-1											
	4											
>>												

# DEMO Accessing elements >> A = [ 1 -1 0; 1 2 3; 3 -1 2 ]; >> A(1,1)

ans = 1 >> A(1) ans =

#### **DEMO** Accessing elements

>> $A = [1 -1 0; 1 2 3; 3 -1 2]$	;
>> A(:,1)	
ans =	
1	
1	
3	
>> A(2:3,1)	
ans =	
1	
3	

#### **DEMO** Transposition

>>	A =	[ 1	-1 (	0;1	2	3;	3	-1	2	]		
A	=											
	1	-1	0									
	1	2	3									
	3	-1	2									
>>	<b>A</b> ′											
an	s =											
	1	1	3									
	-1	2	-1									
	0	3	2									
>>												

#### **DEMO** Scalar product

>>	x=	[	0;	1;	2	]				
<b>x</b> =										
	0									
	1									
	2									
>>	у=	Γ	2;	0;	-1	. ]				
Y=										
	2									
	0									
-	-1									
>>	x′*	ſУ								
ans	5 =									
	-2									

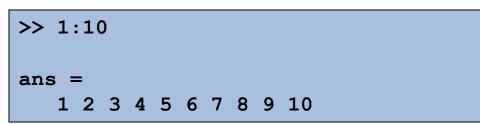
# **DEMO** Errors

• If you try to use the value of an element outside of a matrix, it is an error:

• On the other hand, if you store a value in an element outside of the matrix, the size increases to accommodate the newcomer. Other created spaces are filled with 0.

# **DEMO** Colon operator

 The colon operator, : , is one of MATLAB's most important operators. It occurs in several different forms. The expression 1:10 is a row vector containing the integers from 1 to 10



To obtain non unit spacing, specify an increment. For example:

```
>> 100:-7:50
ans =
   100 93 86 79 72 65 58 51
```

# **DEMO** Built-in functions

- MATLAB provides five functions that generate basic matrices:
  - zeros all zeros
  - ones all ones
  - rand uniformly distributed random elements
  - randn normally distributed random elements
  - eye identity matrix
- Some examples:

>> F=5*ones(3,3)						
F =						
5 5 5						
555						
555						

>> R=randn(4,4)								
R =								
1.0668	0.2944	-0.6918	-1.4410					
0.0593	-1.3362	0.8580	0.5711					
-0.0956	0.7143	1.2540	-0.3999					
-0.8323	1.6236	-1.5937	0.6900					

# **DEMO** MATLAB files and programs

• For example, create a file called factbar.m that contains these MATLAB commands:

```
% investigate the factorial explosion
r=ones(1,6);
for n=2:6
  r(n)=n*(r(n-1);
end;
bar(r);
```

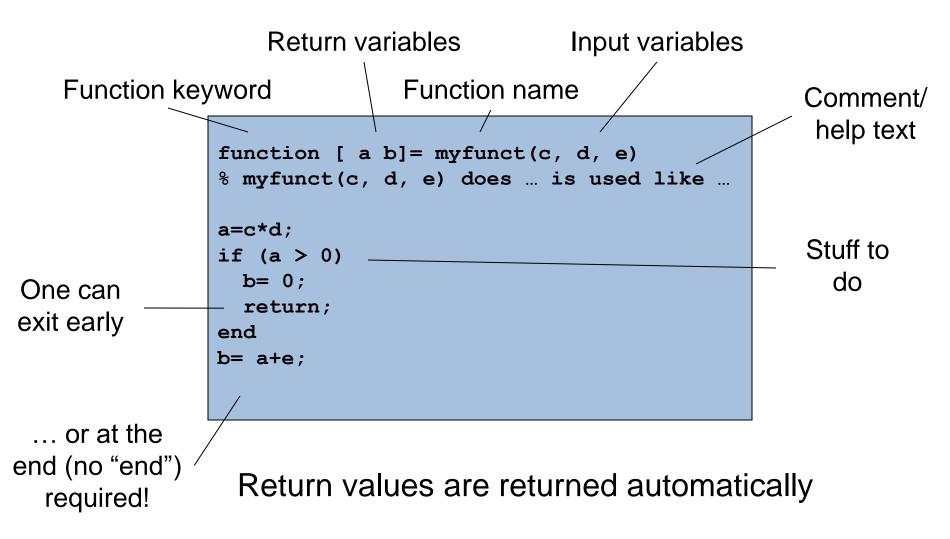
 This is a script (rather than a function) because it doesn't take any inputs or give any outputs.

#### **DEMO** M-file functions

- Functions are M-files that can accept input arguments and return output arguments; the name of the M-file and the function should be the same (WARNING: If they are not the same, the file name overrides!).
- Functions operate on variables within their own workspace

```
function f=myfact(n)
% MYFACT(N) computes N! using an iterative method
f=1;
if (n>1)
  for m=2:n
    f=m*f:
    end;
elseif (n<0)
    error(`negative factorial attempted');
end;</pre>
```

#### M-file functions



# Calling your own functions

function [ a b] = myfunct(c, d, e)

end;

myfunct.m

>> [ apple orange] = myfunct(candy, d, e);

If you do not provide multiple variables for return values, only one of the return values will be considered (and goes into "ans")

### A bit more detail ...

 Variables in Matlab are passed by value, i.e. the content of the variables outside the function remains unchanged

```
function [ a b] = myfunct(c, d, e)
    e= 5;
...
end;
```

#### **DEMO** visualising matrix action

"testMatrixSphere.m" draws a sphere and applies a matrix to it repeatedly

```
>> a= 0.2
>> A= [ cos(a) sin(a) 0;
        -sin(a) cos(a) 0;
        0 0 1]
>> B= [ 0 0 1;
        0 cos(a) sin(a);
        0 -sin(a) cos(a)]
>> testMatrixSphere(A,10);
>> testMatrixSphere(B,10);
>> testMatrixSphere(A*B,10);
```

#### **Rotation matrices**

 Rotation matrices around axes have the general form

$ \left(\begin{array}{cccc} \cos\alpha & \sin\alpha & 0\\ -\sin\alpha & \cos\alpha & 0\\ 0 & 0 & 1 \end{array}\right) $	$ \left(\begin{array}{ccc} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{array}\right) $	$ \left(\begin{array}{rrrrr} 1 & 0 & 0\\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{array}\right) $
Rotation around "z" axis	Rotation around "y" axis	Rotation around "x" axis

• One can combine them with matrix multiplication

• (DEMO)

# File management

- MATLAB uses a search path, or a list of directories, to determine how to execute functions. When we call a standard function, MATLAB executes the first M-file on the path that has the specified name.
- We can override this behaviour using special private directories and sub-functions. The command path shows the search path on any platform.
- MATLAB provides several generic operating system commands for manipulating and managing files:

# File management

Command	Description
what	Return a listing of all M-files in the current directory of folder
dir	List all files in the current directory or folder
ls	Same as dir
type test	Display the M-file test.m in the command window
delete test	Delete the M-file test.m
cd path	Change to directory of folder given by path
chdir path	Same as cd path
cd	Show present working directory or folder (unlike UNIX)
chdir	Same as cd
pwd	Same as cd
which test	Display the directory path to test.m

#### "Toolboxes"

- Functions and scripts can call each other
- A collection of functions/scripts in a directory can form a complex, large program (much like a java .jar library)
- Existing toolboxes are such libraries

#### Alternatives to Matlab

- Python (numpy, scipy and matplotlib)
- Octave
- Mathematica