

Math3900
01/16/09
Getting Started

GETTING STARTED:

- **Logging in:** This is what you do to start working on the computer. If your machine seems to be asleep, jiggle the mouse to wake it up. If necessary, hit the "enter" key to get the cursor into the "login name" box. Your login name is made as follows: All login names from classes begin with "c-". If your name were Wolfgang Amadeus Mozart, your login name would be c-mtwa, following the recipe of c-(first letter of last name) (last letter of last name) (first letter of first name)(middle initial). (If you don't have a middle name, your login will only have 3 initials, i.e. c-xxx). If there is more than one person registered during a semester who would have the same login name (say c-mtwa), then they are assigned the login names c-mtwal, c-mtwa2, etc. There might be a list at the front of the lab to find your login if this is all too complicated. Your initial password is the mtwa part of your login name followed by the last four digits of your student ID number, not your social security number. For example, if Herr Mozart's student ID number was 000001234, his initial password would be mtwal234. It would be the same regardless of whether his login name was c-mtwa or c-mtwa3.
- **The xterm window:** Open the xterm window by clicking on the icon of the little computer at the bottom of your screen. Now you can create a folder for the class with the name 'Math3900' by simply entering `mkdir Math3900` at the shell prompt. To see if the folder 'Math3900' exists, enter `ls` at the shell prompt. Now, go into the folder by entering `cd Math3900` at the shell prompt. From now on, you should save all your works for the class in this folder.
- **Logging out:** Just as important as logging in is logging out. You do this when you finish work (or when you leave the room). There is an exit icon to log out.

MATLAB: To initiate a MATLAB session, simply enter `matlab &` at the shell prompt. To terminate a MATLAB session, enter the command `exit` or `quit` at the MATLAB prompt.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Introducing Matlab (adapted from
% http://www-cse.ucsd.edu/~sjb/classes/matlab/matlab.intro.html)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (1) Help and basics

% The symbol "%" is used in front of a comment.

% To get help type "help" (will give list of help topics) or "help topic"

% If you don't know the exact name of the topic or command you are looking for,
% type "lookfor keyword" (e.g., "lookfor regression")

% When writing a long matlab statement that exceeds a single row use ...
% to continue statement to next row.

% When using the command line, a ";" at the end means matlab will not
% display the result. If ";" is omitted then matlab will display result.

% Use the up-arrow to recall commands without retyping them (and down
% arrow to go forward in commands).

% Other commands borrowed from emacs and/or tcsh:
```

```
% C-a moves to beginning of line (C-e for end), C-f moves forward a
% character (C-b moves back), C-d deletes a character, C-k deletes
% the line to the right of the cursor, C-p goes back through the
% command history and C-n goes forward (equivalent to up and down arrows),
% tab command completion.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (2) Objects in matlab -- the basic objects in matlab are scalars,
% vectors, and matrices...
```

```
N      = 5          % a scalar
v      = [1 0 0]   % a row vector
v      = [1;2;3]   % a column vector
v      = v'        % transpose a vector
                    (row to column or column to row)
v      = [1:.5:3]  % a vector in a specified range:
v      = pi*[-4:4]/4 % [start:end] or [start:stepsize:end]
v      = []        % empty vector

m      = [1 2 3; 4 5 6] % a matrix: 1ST parameter is ROWS
                    % 2ND parameter is COLS
m      = zeros(2,3) % a matrix of zeros
v      = ones(1,3)  % a matrix of ones
m      = eye(3)    % identity matrix
v      = rand(3,1) % random matrix with values in [0,1] (see also randn)

v      = [1 2 3];  % access a vector element
v(3)   % vector(number)
                    % Index starts from 1

m      = [1 2 3; 4 5 6]
m(1,3) % access a matrix element
                    % matrix(rownumber, columnnumber)
m(2,:) % access a matrix row (2nd row)
m(:,1) % access a matrix column (1st row)

size(m) % size of a matrix
size(m,1) % number rows
size(m,2) % number of columns

m1     = zeros(size(m)) % create a new matrix with size of m

who     % list of variables
whos   % list/size/type of variables
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (3) Simple operations on vectors and matrices
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (A) Pointwise (element by element) Operations:
```

```

% addition of vectors/matrices and multiplication by a scalar
% are done "element by element"
a      = [1 2 3 4];          % vector
2 * a          % scalar multiplication
a / 4          % scalar multiplication
b      = [5 6 7 8];          % vector
a + b          % pointwise vector addition
a - b          % pointwise vector addition
a .^ 2         % pointwise vector squaring (note .)
a .* b         % pointwise vector multiply (note .)
a ./ b         % pointwise vector divide (note .)
c      = [1; 2; 3; 4];      % a column vector
a + c          % is not valid
a + c'         % is valid

log( [1 2 3 4] )          % pointwise arithmetic operation (natural logarithm)
log10( [1 10 100] )      % base 10 logarithm
round( [1.5 2; 2.2 3.1] ) % pointwise arithmetic operation

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

% (B) Vector Operations (no for loops needed)
% Built-in matlab functions operate on vectors, if a matrix is given,
% then the function operates on each column of the matrix

```

```

a      = [1 4 6 3]          % vector
sum(a)          % sum of vector elements
mean(a)         % mean of vector elements
var(a)          % variance
std(a)          % standard deviation
max(a)          % maximum

```

```

a      = [1 2 3; 4 5 6]    % matrix
a(:)          % vectorized version of the matrix
mean(a)       % mean of each column
max(a)        % max of each column
max(max(a))   % to obtain max of matrix
max(a(:))     % or...

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

% (C) Matrix Operations:

```

```

[1 2 3] * [4 5 6]'          % row vector 1x3 times column vector 3x1
% results in single number, also
% known as dot product or inner product

```

```

[1 2 3]' * [4 5 6]          % column vector 3x1 times row vector 1x3
% results in 3x3 matrix, also
% known as outer product

```

```

a      = rand(3,2)          % 3x2 matrix
b      = rand(2,4)          % 2x4 matrix
c      = a * b              % 3x4 matrix

```

```

a      = [1 2; 3 4; 5 6]    % 3 x 2 matrix

```

```

b          = [5 6 7];           % 1 x 3 vector
b * a      % matrix multiply
a' * b'    % matrix multiply

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%(4) Saving your work

```

```

save mysession           % creates mysession.mat with all variables
save mysession a b      % save only variables a and b

clear all                % clear all variables
clear a b                % clear variables a and b

load mysession           % load session
a
b

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%(5) Relations and control statements

```

```

% Example: given a vector v, create a new vector with values equal to
% v if they are greater than 0, and equal to 0 if they less than or
% equal to 0.

```

```

v          = [3 5 -2 5 -1 0]    % 1: FOR LOOPS
u          = zeros( size(v) );  % initialize

for i = 1:size(v,2)            % size(v,2) is the number of columns
    if( v(i) > 0 )
        u(i) = v(i);
    end
end
u

v          = [3 5 -2 5 -1 0]    % 2: NO FOR LOOPS
u2         = zeros( size(v) );  % initialize
ind        = find( v>0 )        % index into >0 elements
u2(ind)    = v( ind )

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%(6) Creating functions using m-files:
% Functions in matlab are written in m-files. Create a file called
% 'thres.m' In this file put the following 4 lines:

```

```

function res = thres( v )
u          = zeros( size(v) );  % initialize
ind        = find( v>0 )        % index into >0 elements
u(ind)     = v( ind )

```

```

v          = [3 5 -2 5 -1 0]
thres( v ) % call from command line

```

