

وزارة التعليـم العـالي
جـــامعة طيبـــــة
كليـــــة العلـــوم
قســم الريــاضيات

MATLAB Workshop
Department of Mathematics

9/7/1436 (2015)

Meeting 3:
Introduction to advanced programming with Matlab
Matlab Programs

Mostafa Zahri
Taibah University
Madinah

First matlab program myfunc.m

Open m-file window type the following code

```
function y = myfunc (x);
y = 2*x.^2 - 3*x + 1;
```

Save this file as 'myfunc.m' in your working directory.
in the command window enter:

```
>> x = -2;              % define the value of x as -2
>> y = myfunc(x);       % evaluate myfunct at x=-2
```

input one value x → output one value y

Multiple inputs multiinput.m

Open m-file window type the following code

```
function y = multiinput (x,y)
y = 2*x.^p - 3*x + 1;
```

Save this file as 'myfunc2d.m' in your working directory.
in the command window enter:

```
>> x = 2; p=3        % define the values  x=2 and p=3
>> y = multiinput (x,p);    % evaluate myfunct at x=-2
```

Multiple inputs x and p→ output one value y

Multiple Ouputs multioutput.m

Type the following code

```
function [a b c] = multioutput (x);
a = x .^2;  b = x .^3; c = x .^3;
```

Save this file as 'multioutput.m' in your working directory.
in the command window enter:

```
>> x = 2;          % define the values  x=2
>> y = multioutput (x);    % evaluate mypowers at x=2
```

One input x → multiple outputs a, b and c

Multiple Input-Ouputs multiinout.m

Type the following code

```
function [a b c] = multiinout(x,p);
a = x .^(2-p);  b = (x+1) .^(2+p); c = (x-3) .^(2*p);
```

Save this file as 'multioutput.m' in your working directory.
in the command window enter:

```
>> x = 2;  p=3        % define the values  x=2 and p=3
>> y = multiinout (x,p);    % evaluate mypowers at x=2
```

Multi input x and p→ multiple outputs a, b and c

Flow Control for…end loop

Type the following code

```
function S = mysumFOR(n)        % gives the sum of the first n integers
S = 0;                          % start at zero
for i = 1: n                    % The loop do n times
S = S + i;                      % add the current integer
end                             % end of the loop
```

Save this file as ' = mysum .m' in your working directory.

```
>> mysum (2)     → ans :  3      %output  (1+2)
>> mysum (4)     → ans :  10     %output  (1+2+3+4)
>> mysum (100)   → ans : 5050    %output  (1+2+…+100)
>> mysum (11111) → ans : 61732716 %output  (1+2+…+11111)
```

input natural n → output the sum 1+2+…+n

Flow Control while…end loop

```
% gives the sum of the first n integers
function S = mysumWHILE(n)
S=0;                          % start at zero
i = 1;                        % start the counter at 1
while i <= n                  % do the task since condition true
S = S + i;                    % add the current integer
i=i+1;                        % update the counter
end                           % end task
```

Flow Control for…end loop: Newton Algorithm

```
function x = mynewton (f,fd ,x0 ,n)
format long                          % prints more digits
x = x0;                              % set x equal to the initial guess x0
for i = 1:n                          % Do n times
x = x - f(x)/ fd(x)                  % Newton 's formula , prints x too
end
```

```
% Solves f(x) = 0 by doing n steps of Newton 's method starting at x0.
% Inputs : f -- the function , input as an inline
% fd -- it 's derivative , input as an inline
% x0 -- starting guess , a number
% n -- the number of steps to do
% Output : x -- approximate the solution in n iteration (steps)
```

The if ..else... end statement : Newton Algorithm

```
% Solves f(x) = 0 by doing n steps of Newton 's method starting at x0.
% Inputs : f -- the function , input as an inline
% f1 -- it 's derivative , input as an inline
% x0 -- starting guess , a number
% tol -- desired tolerance , prints a warning if |f(x)|> tol
% Output : x -- the approximate solution
x = x0; % set x equal to the initial guess x0
```

```
function y = mynewtonif (f,fd ,x0 ,n,tol );
r=10;
disp('#-------- Iter.  Error       Solution ');
for i = 1:n                                          % Do n times
if r > tol
   x = x - f(x)/ fd(x);                              % Newton 's formula
   r = abs (f(x));
   disp ( sprintf ( '\t\t %3d \t %.7f \t  %.7f\t ', i,r,x));
else    break;
end
end
```

The while... end statement : Newton Algorithm

```
function y = mynewtonwhile (f,fd ,x0 ,n,tol );
x = x0; % set x equal to the initial guess x0
r=10;
i=0;
disp('#-------- Iter.  Error       Solution ');
while r > tol % if while statement true iterates again
   x = x - f(x)/ fd(x); % Newton 's formula
   r = abs (f(x));
   disp ( sprintf ( '\t\t %3d \t %.7f \t  %.7f\t ', i+1,r,x));
end
```

Example of running the code

```
>> mynewtonwhile(f,fd ,0.6 ,4,0.000001 )
#-------- Iter.      Error              Solution
          0         0.0021874        0.5669291
                        0             0.0000074              0.5650065
                        0             0.0000000              0.5650000
```

Matlab Format Number

```
% short        5-digit scaled fixed point  3.1416
format short
x_short = rand(1,2)

% short e   5-digit floating point  3.14E+00
format short e
x_short_e = x_short

% short g   Best of 5-digit fixed or floating point     3.1416
format short g
x_short_g = x_short
```

Matlab Format Number

```
% long  15-digit scaled fixed point     3.141592654
format long
x_long = rand(1,2)
 % long e    15-digit floating point     3.14E+00
format long g
x_long_g = x_long
 % long g    Best of 15-digit fixed or floating point    3.141592654
format long e
x_long_e = x_long

% vpa('x',x)d    decimal digits of accuracy
gold = vpa('(1+sqrt(5))/2',50)
```

Matlab Format Number:

type	Result	Example
compact	Suppresses excess line feeds to show more output in a single screen. Contrast with loose.	theta = pi/2 theta=1.5708
hex	Hexadecimal (hexadecimal representation of a binary double-precision number)	400921fb54442d18
long	15-digit scaled fixed point	3.141592654
long e	15-digit floating point	3.14E+00
long g	Best of 15-digit fixed or floating point	3.141592654
loose	Adds linefeeds to make output more readable. Contrast with compact.	theta = pi/2 theta=1.5708
rat	Ratio of small integers	355/113
short	5-digit scaled fixed point	3.1416
short e	5-digit floating point	3.14E+00
short g	Best of 5-digit fixed or floating point	3.1416

Thanks