Lecture 5: Functions in Matlab

Functions

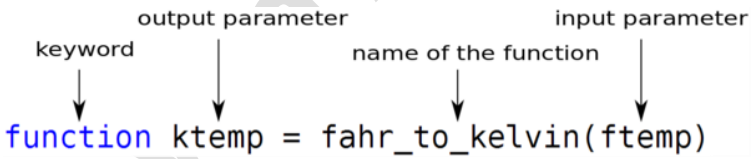
Programs that accept inputs and return outputs

Syntax

```
function [o1,o2, ..., oN] = myfun(i1,i2,i3, ..., iN)
```

```
function [out1, out2] = function_name(in1, in2)
    %FUNCTION_NAME   Function description

    % This section below is called the body of the function
    out1 = something calculated;
    out2 = something else;
end
```



If your function returns one output, you can specify the output name after the function keyword.

```
function myOutput = myFunction(x)
```

If your function returns more than one output, enclose the output names in square brackets.

```
function [one,two,three] = myFunction(x)
```

If there is no output, you can omit it.

```
function myFunction(x)
```

Or you can use empty square brackets.

```
function [] = myFunction(x)
```

If your function accepts any inputs, enclose their names in parentheses after the function name. Separate inputs with commas.
function y = myFunction(one,two,three)

If there are no inputs, you can omit the parentheses.

Note:

The **end** keyword should be used to indicate the end of the function. It is required when any function in the file contains a **nested function** or function used as a **local function** within the script and function file.

Local Functions

Local functions are only available to other functions within the same file.

Example:

```
function [avg, med] = mystats(y)
a= length(y);
avg = mymean(y,a);
med = mymedian(y,a);
end

function a = mymean(v,n) ---- Example of a local function
a = sum(v)/n;
end
```

Example, Define two functions in a file named stat2.m, where the first function calls the second.

```
function [m,s] = stat2(x)
    n = length(x);
    m = avg(x,n);
    s = sqrt(sum((x-m).^2/n));
end

function m = avg(x,n) % Function avg is a local function.

    m = sum(x)/n;
end
```

Call function stat2 from the command line.

```
values = [12.7, 45.4, 98.9, 26.6, 53.1];
[ave,stdev] = stat2(values)
```

Output

```
ave =
   47.3400
stdev =
   29.4124
```

3. Nested Functions

A nested function is a function that is completely contained within a parent function. Any function in a program file can include a nested function.

Example, this function named parent contains a nested function named **nestedfx**:

```
function parent
disp('This is the parent function')
nestedfx

  function nestedfx
    disp('This is the nested function')
  end

end
```

Example:

When parent functions do not use a given variable, the variable remains **local** to the nested function. For example, in this function named main, the two nested functions have their own versions of x that cannot interact with each other:

```
function main
  nestedfun1
  nestedfun2

  function nestedfun1
    x = 1;
  end

  function nestedfun2
    x = 2;
  end
end
```

Example

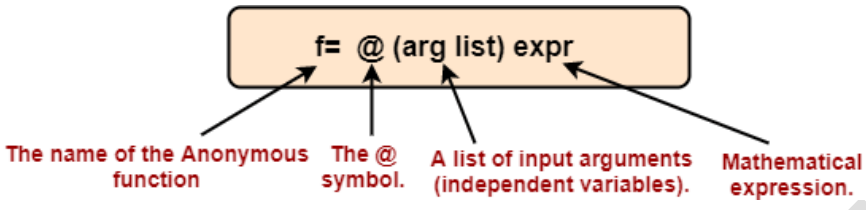
```
function p = makeParabola(a,b,c)
p = @parabola;

  function y = parabola(x)
  y = a*x.^2 + b*x + c;
  end

end
```

Anonymous Function in Matlab

An anonymous function is a simple (one-line) user-defined function that is defined without creating a separate function file (M-file)



where **f** is the function handle. The **input list** can contain a single variable or several variables separated by commas. After creating the function, we can use it with its handle to evaluate the function or pass it as an argument to other functions.

Syntax
Fun=@(argumentlist)expression

Steps to Write Anonymous Function in Matlab

Step 1: First define Matlab handle function by using ' @ ' symbol and input variable , y = @ (x)
Step 2: Write the whole equation next to the function handler variable.
Step 3: Accept the input value inside the output variable, y (0).

Example, create an anonymous function that evaluates and return the area of a circle:

```
>> cirarea = @ (radius) pi * radius .^ 2;
```

```
>> cirarea(4)
ans =
    50.2655
>> cirarea(1:4)
ans =
    3.1416   12.5664   28.2743   50.2655
```

Example

```
mul=@(x,y) x*y;
res1=mul(2,3)
res2=mul(4,3)
res3=mul(4,5)
```

Output:

```
res1=6
res2=12
res3=20
```

We can write anonymous functions with no inputs or multiple inputs and outputs. If the function has **no input** then we can use an empty parenthesis to call the anonymous function.

Example

```
curr= @() datestr(now);
d = curr()
```

Output:

```
d= 22-Oct-2019 11:02:47
```

Examples

```
myfun=@(x,y)(x+y);
x=4
y=7
z=myfun(x,y)
```

Output

```
z=11
```

Advantages of Anonymous Functions in Matlab

- In an anonymous function, we can create any function which is not predefined.
- It can be stored in a variable.
- Anonymous functions can be returned in function.
- It can be pass inside the function.
- These functions cannot be stored in program files, therefore, we can save memory.
- We can store an anonymous function handle so that we can use it again and again whenever required.
- It is easy to represent and implement.

Examples

Example: Function with One Output

```
% Input vector
values = [12, 4, 8.9, 6, 3];

% function return mean of vector c
function m = stat(x)
   n = length(x);
```

```
    m = sum(x)/n;
end
```

```
mean = stat(values)
```

Example: Function with multiple outputs

```
function [m,s] = stat(x)
    n = length(x);
    m = sum(x)/n;
    s = sqrt(sum((x-m).^2/n));
end
```

Call the function from the command line.

```
values = [12.7, 45.4, 98.9, 26.6, 53.1];
[ave,stdev] = stat(values)
```

```
ave =
   47.3400
stdev =
   29.4124
```