24-Functions Part 3

text: Chapter 7

ECEGR 101

Engineering Problem Solving with Matlab

Professor Henry Louie

Overview

- Anonymous Functions
- Subfunctions

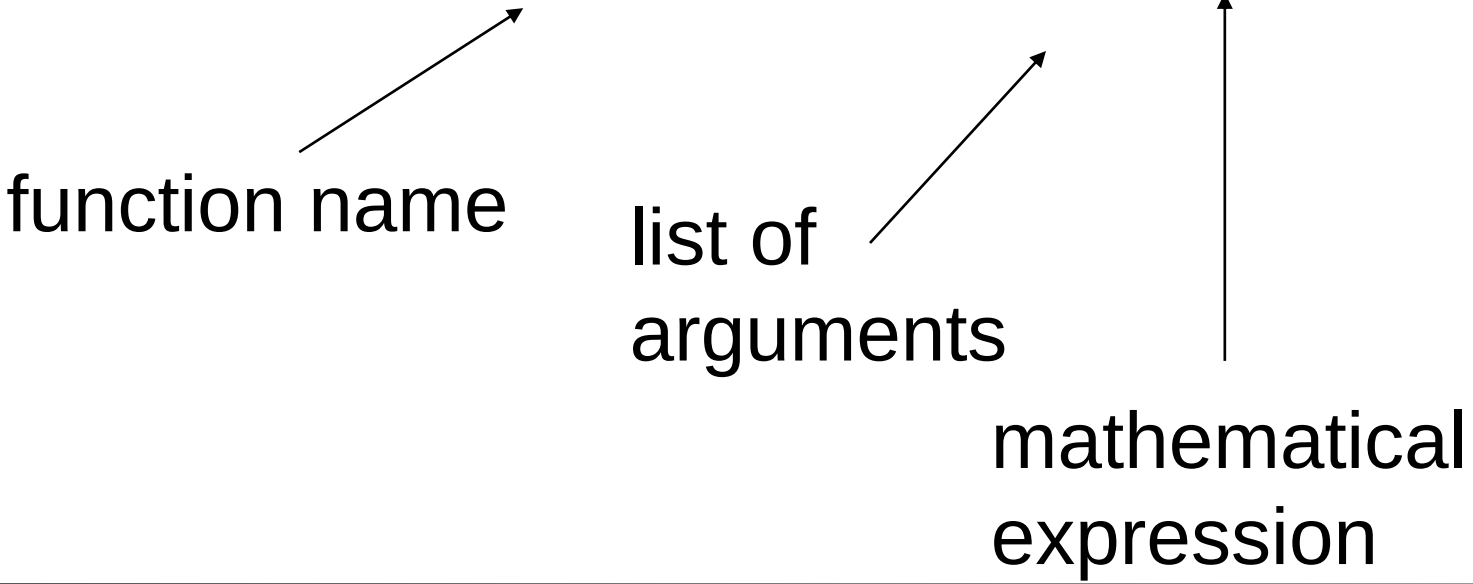
Anonymous Functions

- Used to define simple mathematical expressions which will be used often in the program

  ex. $y = x^2-x+2\sin(x)-x^3$

Anonymous Functions

- Anonymous function is user-defined within the computer code (not in a separate file).
- Used often for math functions (symbolic representation).
- Syntax: fun_name = @ (arglist) expr



Anonymous Functions: Example 1

$$f(t) = e^{-t}\cos(2\pi t)$$

```
decaying_cos = @ (t) exp(-t).*cos(2*pi*t)
```

```
>> decaying_cos = @ (t) exp(-t).*cos(2*pi*t)

>> decaying_cos(1:10)
ans =
    0.3679    0.1353    0.0498    0.0183    0.0067    0.0025    0.0009
    0.0003    0.0001    0.0000
```

Anonymous Functions

- Anonymous functions can be defined for two or more independent variables.

```
>> g = @ (x,y) x.^3 + y.^3 + (x.^2).*y + x.*(y.^2) + x.*y;
>> g
g =
    @(x,y)x.^3+y.^3+(x.^2).*y+x.*(y.^2)+x.*y
>> g(3,4)
ans =
   187
>> g(5)
??? Input argument "y" is undefined.
Error in ==> @(x,y)x.^3+y.^3+(x.^2)*y+x*(y.^2)+x*y
```

Exercise

a) Write a script file that defines two anonymous functions:

$$f(t) = 100\ln(t)\sin(t) \text{ and}$$

$$g(t) = 0.001e^{3t}\sin(30t).$$

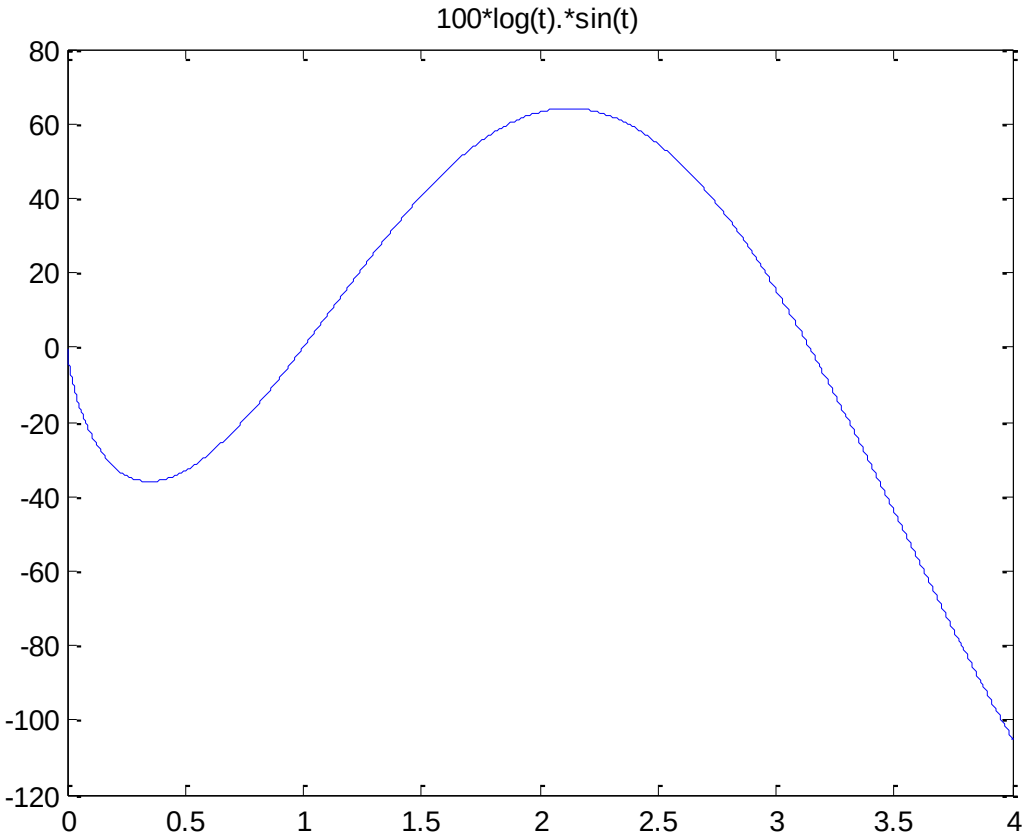
b) Plot functions f(t) and g(t) for t between 0.001 and 4 in separate figures. Make sure that the functions appear smooth on the plots.

c) Plot the function h(t) = f(t) + g(t) over the same t interval in a new figure.

Exercise

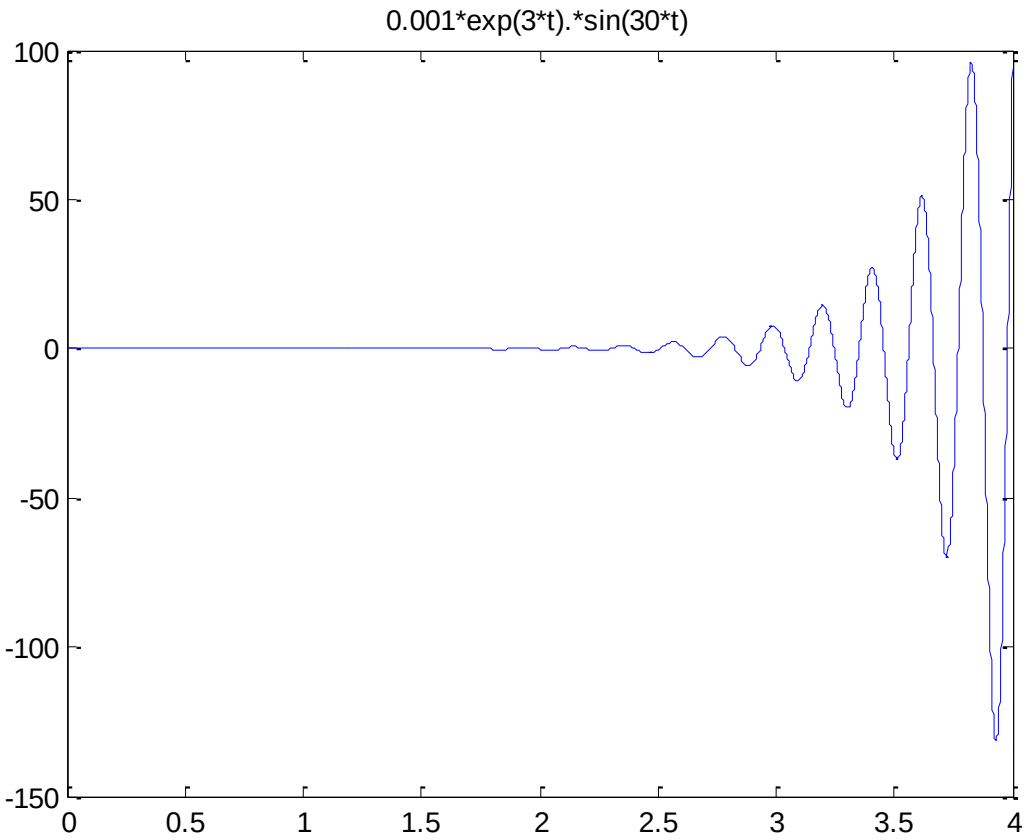
```
t = 0.001:0.001:4;
f = @ (t) 100*log(t).*sin(t);
figure(1)
plot(t, f(t))
title('100*log(t).*sin(t)');
g = @ (t) 0.001*exp(3*t).*sin(30*t);
figure(2)
plot(t, g(t))
title('0.001*exp(3*t).*sin(30*t)');
figure(3)
plot(t, f(t)+g(t))
title('100*log(t).*sin(t) + 0.001*exp(3*t).*sin(30*t)')
```

Exercise

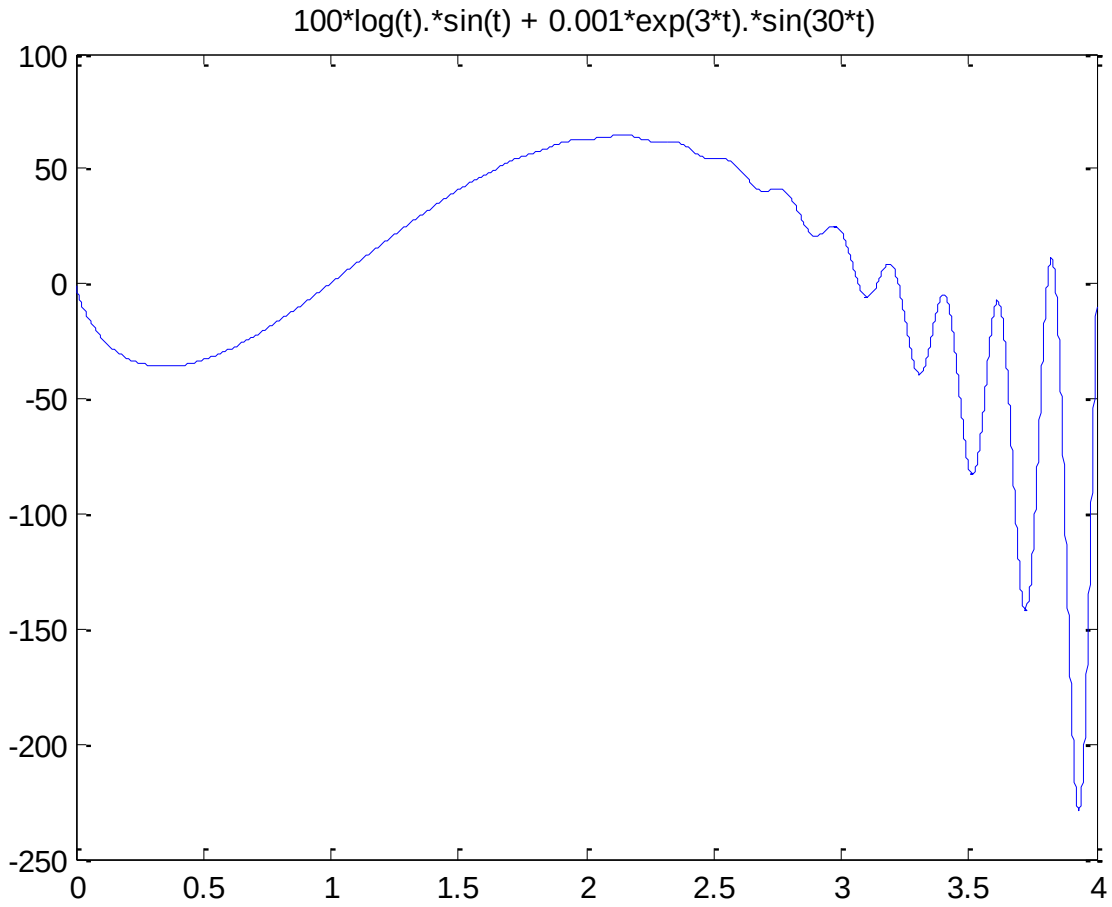



SEATTLE UNIVERSITY
IHS
1891

Exercise


0.001*exp(3*t).*sin(30*t)
100
50
0
-50
-100
-150
0
0.5
1
1.5
2
2.5
3
3.5
4

Exercise


100*log(t).*sin(t) + 0.001*exp(3*t).*sin(30*t)
100
50
0
-50
-100
-150
-200
-250
0
0.5
1
1.5
2
2.5
3
3.5
4

Subfunctions

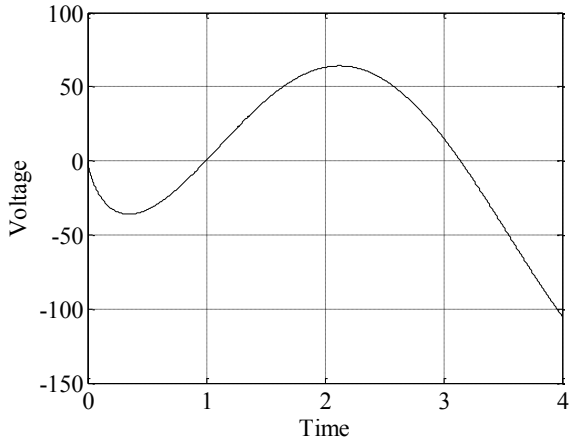
- When several blocks of code are repeated, time can be saved by defining function(s) within functions
- Function(s) defined inside a function only exist (can be called) from that function
- Need to include "end" to separate different functions

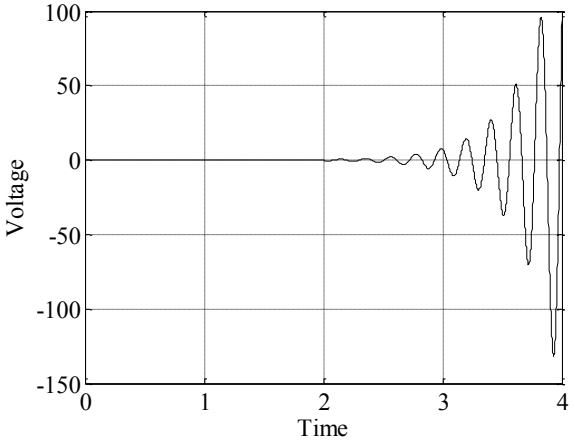
Subfunctions

- Consider a function to create these two plots

```
function [] = data_visualization(t)
f =100*log(t).*sin(t);
figure
plot(t,f,'black')
set(gca,'Fontsize',16,'fontname','times')
grid on
xlabel('Time','Fontsize',16,'fontname','times')
ylabel('Voltage','Fontsize',16,'fontname','times')

g =0.001*exp(3*t).*sin(30*t);
figure
plot(t,g,'black')
set(gca,'Fontsize',16,'fontname','times')
grid on
xlabel('Time','Fontsize',16,'fontname','times')
ylabel('Voltage','Fontsize',16,'fontname','times')
end
```





Subfunctions

- Consider a function to create these two plots

```
function [] = data_visualization(t)
f =100*log(t).*sin(t);
figure
plot(t,f,'black')
set(gca,'Fontsize',16,'fontname','times')
grid on
xlabel('Time','Fontsize',16,'fontname','times')
ylabel('Voltage','Fontsize',16,'fontname','times')

g =0.001*exp(3*t).*sin(30*t);
figure
plot(t,g,'black')
set(gca,'Fontsize',16,'fontname','times')
grid on
xlabel('Time','Fontsize',16,'fontname','times')
ylabel('Voltage','Fontsize',16,'fontname','times')
end
```

These lines of code are very similar.
Shorten code by using a function within the function

Subfunctions

```
function [] = data_visualization(t)
f =100*log(t).*sin(t);
plot_data(t,f)
g =0.001*exp(3*t).*sin(30*t);
plot_data(t,g)

end
```

Must include "end"

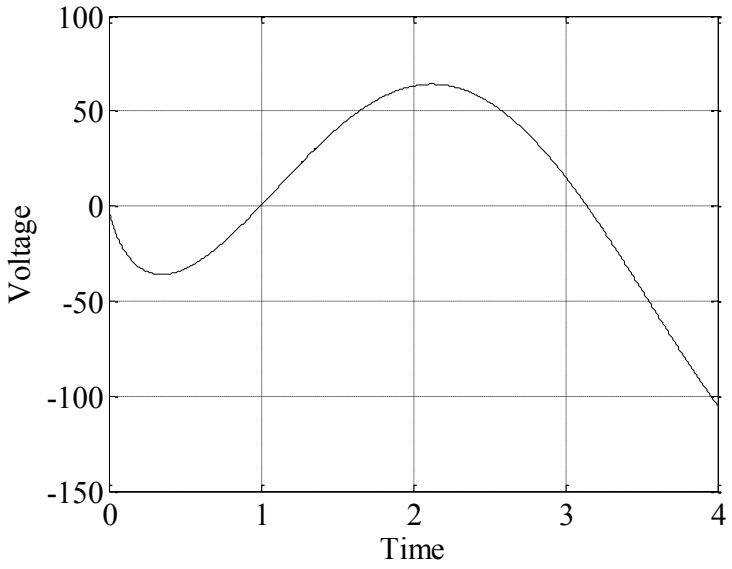
The main function. It calls plot_data, which is defined within it

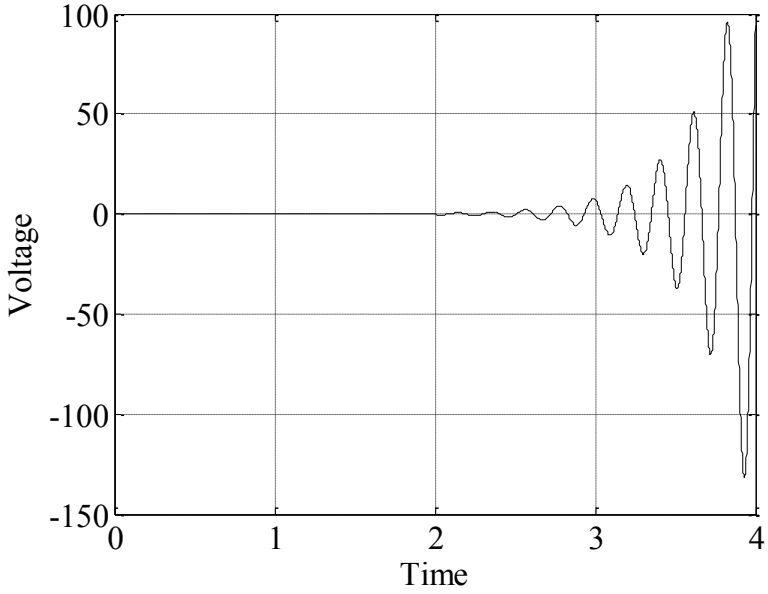
```
function []=plot_data(x,y)
figure
plot(x,y,'black')
set(gca,'Fontsize',16,'fontname','times')
grid on
xlabel('Time','Fontsize',16,'fontname','times')
ylabel('Voltage','Fontsize',16,'fontname','times')
end
```

Must include "end"

This is in the same .m file. It plots the data and formats the graphs

Subfunctions





Subfunctions

- Note: “plot_data” was defined in “data_visualization” and is not available in the command line or other functions

```
>> plot_data(x,y)
Undefined function 'plot_data' for input arguments of type 'double'.
```

Exercise

Write a function whose inputs are the parameters (mean and standard deviation) of a Normal Distribution.

The function then generates 10, 100, 1000 and 10,000 samples from a Normal Distribution with the input parameters and computes and displays the mean, standard deviation and variance of the samples for each distribution. Implement using a subfunction and without loops.

Hint: N=random('Normal',mu,sigma,1000,1);
Creates 1000 samples from a normal distribution with mean mu and standard deviation sigma

Exercise

```
1   function [] = random_stats(mu,sigma)
2   N=random('Normal',mu,sigma,10,1);
3   statistics(N)
4   N=random('Normal',mu,sigma,100,1);
5   statistics(N)
6   N=random('Normal',mu,sigma,1000,1);
7   statistics(N)
8   N=random('Normal',mu,sigma,10000,1);
9   statistics(N)
10  end
11
12  function []=statistics(data)
13  mean_data=mean(data);
14  std_data=std(data);
15  var_data=var(data);
16
17  fprintf('The Mean is %1.3f \n',mean_data)
18  fprintf('The Std. Dev is %1.3f \n',std_data)
19  fprintf('The Variance is %1.3f \n',var_data)
20  disp('====================================')
21  end
```

```
>> random_stats(0,10)
The Mean is -4.414
The Std. Dev is 14.211
The Variance is 201.944
====================================
The Mean is -0.072
The Std. Dev is 10.296
The Variance is 106.008
====================================
The Mean is -0.323
The Std. Dev is 9.833
The Variance is 96.695
====================================
The Mean is 0.049
The Std. Dev is 9.860
The Variance is 97.212
====================================
```