MATLAB Function Example Handout

MatLab is a high performance numeric computing environment, which includes numerical analysis, matrix computation, signal processing, and graphics to provide answers to the most troubling of mathematical problems. This handout provides different examples to show the different aspects of MatLab. This is a very abbreviated handout, and users are strongly encouraged to search through MatLab's extensive help files and demonstrations.

Example 1:

The first example is of a natural response to a simple dynamic system, described by the mathematical equation:

$$f(t) = te^{(-t/k)} \cos(t - \theta)$$

MatLab provides most common trigonometric functions as well as special numerical algorithms for analysis. MatLab also allows users to enter their own functions. To start MatLab, click on the **MatLab** icon in the **Math Programs** group under **All Programs** the **Start** button. Once the program starts, you are ready to create your own **m-file**. An **m-file** is simply a file with the extension **.m** that denotes it is for MatLab use.

– In the **MatLab Command Window**, select the **New Script** button located in the **File** pane of the **Home** tab.

– The **Editor** window will open. You can also open Notepad outside of MatLab to edit m-files.

– To enter the example above, type in the following:

```
function ex1 = fexam1(t)
% fexam1(t)
% the function f(t) = t*e^(-t/k)*cos(pi*t-theta)
% simple example of a natural response to a dynamic system for MatLab Handout

k=3;
theta = 3*pi/4;
ex1 = t.*exp(-t/k).*cos(pi*t-theta);
```

Save this file as **fexam1.m**.

Note: The periods *are* necessary. See the MatLab help handout on common operators and mistakes.

Now that you have an m-file, you are ready to use your function. Make sure that the MatLab command window is in the same directory as your function. To test your function, type **fexam1(1)** in the command window. The screen should show something like this:

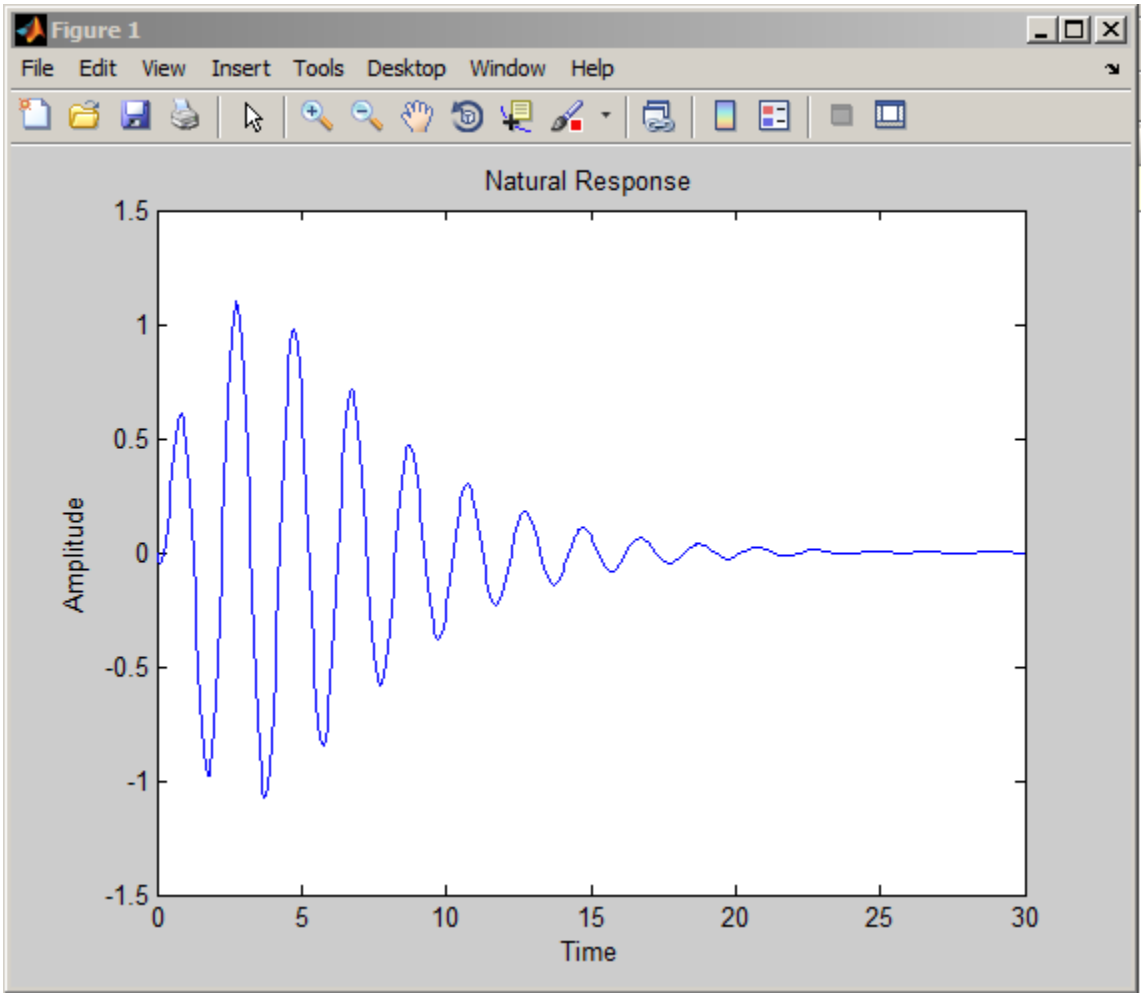
```
» fexam1(1)

ans =

  0.5067

»
```

To do something more useful, let's plot the function. We will plot the same function for the values of t from 0 to 30. To plot this, or any, function, we use **fplot**. To create the plot above, simply type the following at the MatLab command prompt:

```
fplot(@fexam1,[0 30])
```

To add x and y axis labels and a title to the plot, go to the menu in the graph window under **Insert** click **X label,** once you have added the x label text do the same for the **Y label** and **Title**.

The finished plot should look something like this:



Of course, any function can be substituted for **fexam1**, such as **sin**, **cos**, etc.
Note: The at symbol before the function name.

MatLab can do many other useful operations. We will explore only three: **zero finding**, **numerical integration**, and **minima finding**.

Zero Finding:

For this operation, we will use the function **fzero**. Type the following at the MatLab command prompt:

»**fzero(@fexam1, 5)**

ans =

5.2500

This finds a zero for the example function at a value near t=5.

Numerical Integration:

MatLab utilizes a function called **quad**, which is an adaptive recursive form of Simpson's rule. Type the following at the MatLab command prompt:

» **quad(@fexam1,0,3)**

ans =

0.3347

»

This numerically integrates the function from 0 to 3.

Minima Finding:

This function is called **fminbnd**. Type the following at the MatLab command prompt:

» **fminbnd(@fexam1,0,1)**

ans =

0.1256

»

This finds the local minimizer for the function between the boundary values of 0 and 1. This means that at t=0.1256, the amplitude is the smallest for the interval from 0 to 1. To find the value of the function at t=0.1256, simply type the following:

» **fexam1(0.1256)**

ans =

-0.0459