

Matlab Handout

Nancy Chen
Math 19
Fall 2004

Introduction

Matlab is a useful program for algorithm development, numerical computation, and data analysis and visualization. In this class you will only need Matlab for visualizing and solving systems of three differential equations.

Getting Started

You can use Matlab 6.5 in the computer labs (go to Programs → Math, Statistics, Chemistry) or you can download it from the software downloads page at www.fas.harvard.edu/computing. Note that you can only use Matlab if you're connected to the Harvard network. Also, if you're a Mac user, installing Matlab on a Mac can be a pain, but there are instructions on the website.

The Matlab desktop environment has 3 sub-windows: Current Directory/Workspace, Command History and Command Window.

- Current Directory/Workspace: You can ignore this window. It displays any files in the directory you're working in and any matrices/variables you've created.
- Command History: This window displays all commands you've typed. If you want to repeat a command or a series of commands, you can highlight those lines in this window and press **Enter**.
- Command Window: Here you type everything you want the program to do; in your case you would solve the system of differential equations and plot the trajectories.

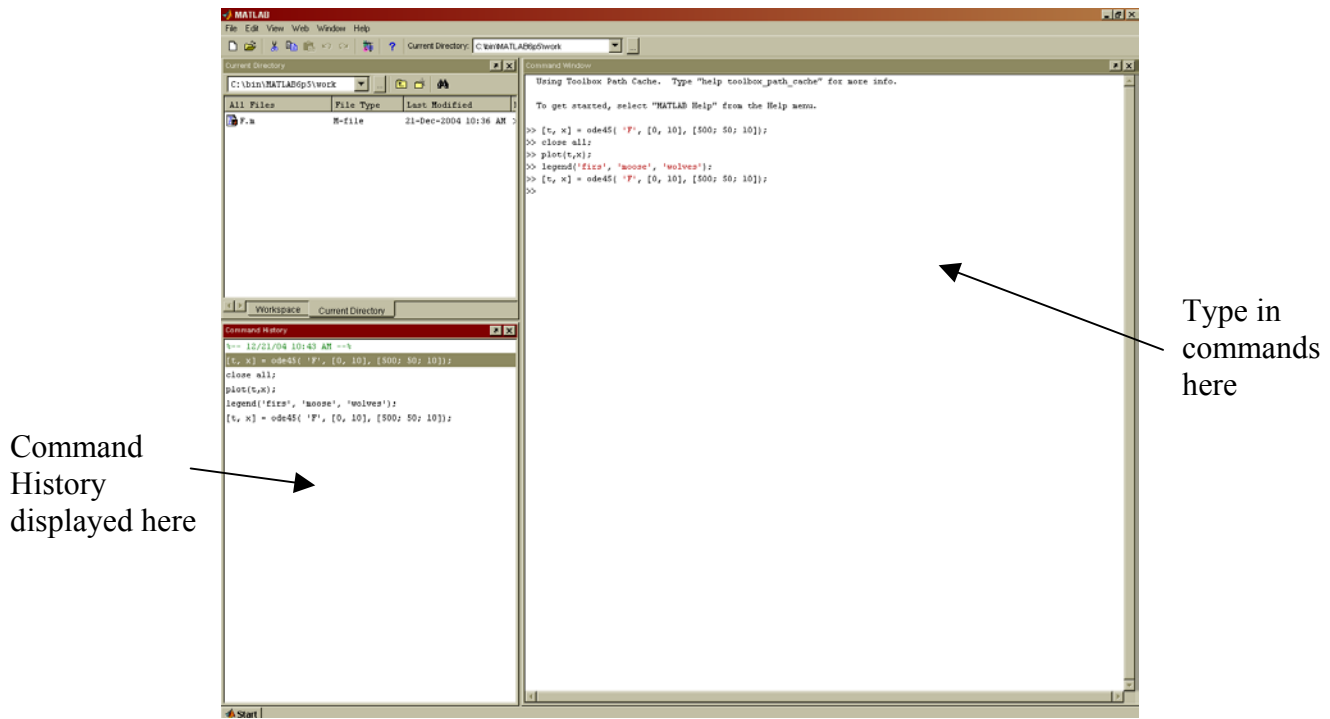
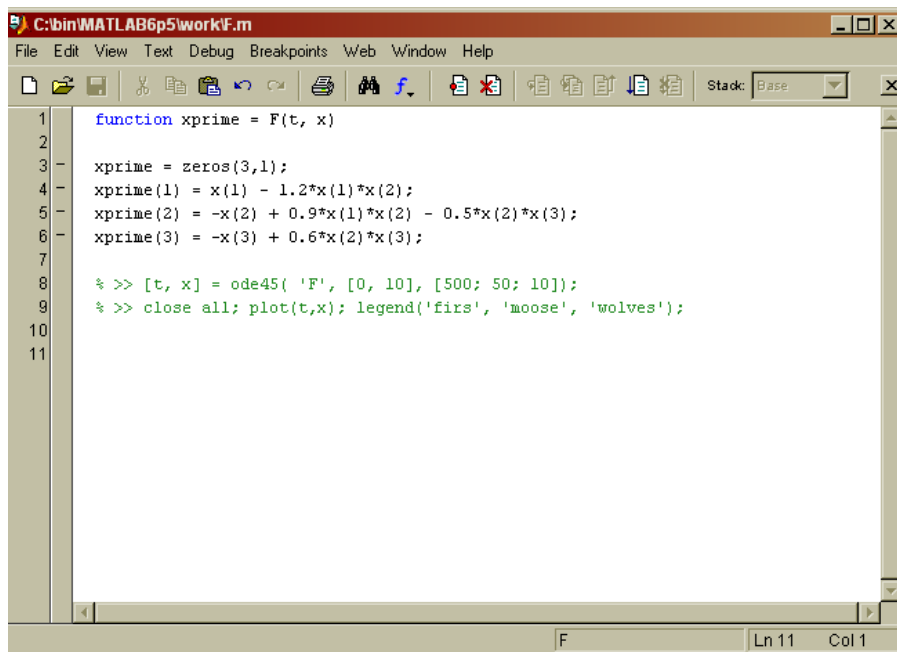


Figure 1: The Matlab desktop environment.

Defining Your System of Equations



```
C:\bin\MATLAB6p5\work\F.m
File Edit View Text Debug Breakpoints Web Window Help
Stack: Base
1 function xprime = F(t, x)
2
3 xprime = zeros(3,1);
4 xprime(1) = x(1) - 1.2*x(1)*x(2);
5 xprime(2) = -x(2) + 0.9*x(1)*x(2) - 0.5*x(2)*x(3);
6 xprime(3) = -x(3) + 0.6*x(2)*x(3);
7
8 % >> [t, x] = ode45('F', [0, 10], [500; 50; 10]);
9 % >> close all; plot(t,x); legend('firs', 'moose', 'wolves');
10
11
F Ln 11 Col 1
```

Figure 2: the m file window for the function defining a fir-moose-wolf system

You need to define your system of differential equations by creating what's called an "m file." I will use a sample fir-moose-wolf system. Click on **New**, and in the new window that just popped up, type in the code for your system:

- On the first line, type "function somename = functionname(variable1, variable2, ...)" somename will be used to define the function in the m file, while functionname will be used whenever you want to do something with that function (e.g. solve it or plot it).
- In the following lines, type the function itself. Here the output of the function is a 3 x 1 matrix (or array). The first line sets the output matrix to 0 initially, and the subsequent 3 lines represent the three differential equations. Note that the variable x is also a matrix, so you can think of x(1) as x, x(2) as y and x(3) as z. You may need to tinker with the coefficients for your project.
- Finally, to make life easier, the commands you will need to use are included as comments (the green lines that start with "%").

When you're done, save it as "functionname.m." Now you're ready to solve and plot.

Graphing

Creating the Figure

Before you begin, make sure you're working in the directory in which you saved your m file. Browse for the correct directory by clicking the "...” button next to **Current Directory** on the toolbar.

First you need to solve your system given initial conditions. You will use an Ordinary Differential Equation (ODE) solver, `ode45`, which uses the Runge-Kutta method to solve a system of differential equations. Type

```
[t, x] = ode45('functionname', [t1, t2], [x0; y0; z0]);
```

where `[var1, var2]` are the variables you want your answer to be, `[t1, t2]` specify the desired time interval, and `[x0; y0; z0]` are the initial values.

The command `plot(t,x);` will open a figure window with the plot of the trajectories. `close all;` closes all open windows.

Adding Legends and Titles

To add a legend, type `legend('name1', 'name2', 'name3');` labeling each line in order. You could also go to the **Insert** menu on the figure, select **Legend**, and then type in the labels.

To add a title, type `title('title you want');` or select **Title** from the **Insert** menu.

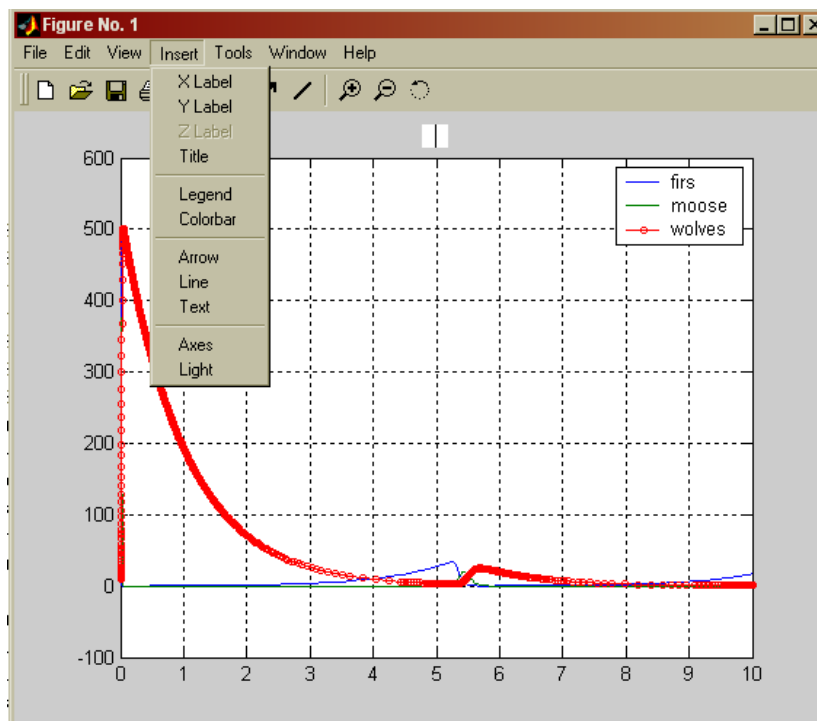


Figure 3: A plot of the fir-moose-wolf system displaying the **Insert** menu.

Editing Axes

You can label the axes using specific functions (xlabel, ylabel, etc.) but it's easiest to select **Axes Properties** from the **Edit** menu of the figure window. Here you can change the axes labels, appearance and range.

Editing Figure Appearance

You will need to be on the cursor mode to do edit the figure lines. You can also add text, lines and arrows to your figure by pressing the appropriate buttons on the toolbar.

Again, the appearance of the lines can be programmed, but to keep things simple, right click on the line you want to change and select **Properties**. From the property window, you can change the color and style of the lines or add markers.

You can add a grid (as in Figure 3) to your graph by typing `grid on` in the command window. `grid off` will erase the grid.

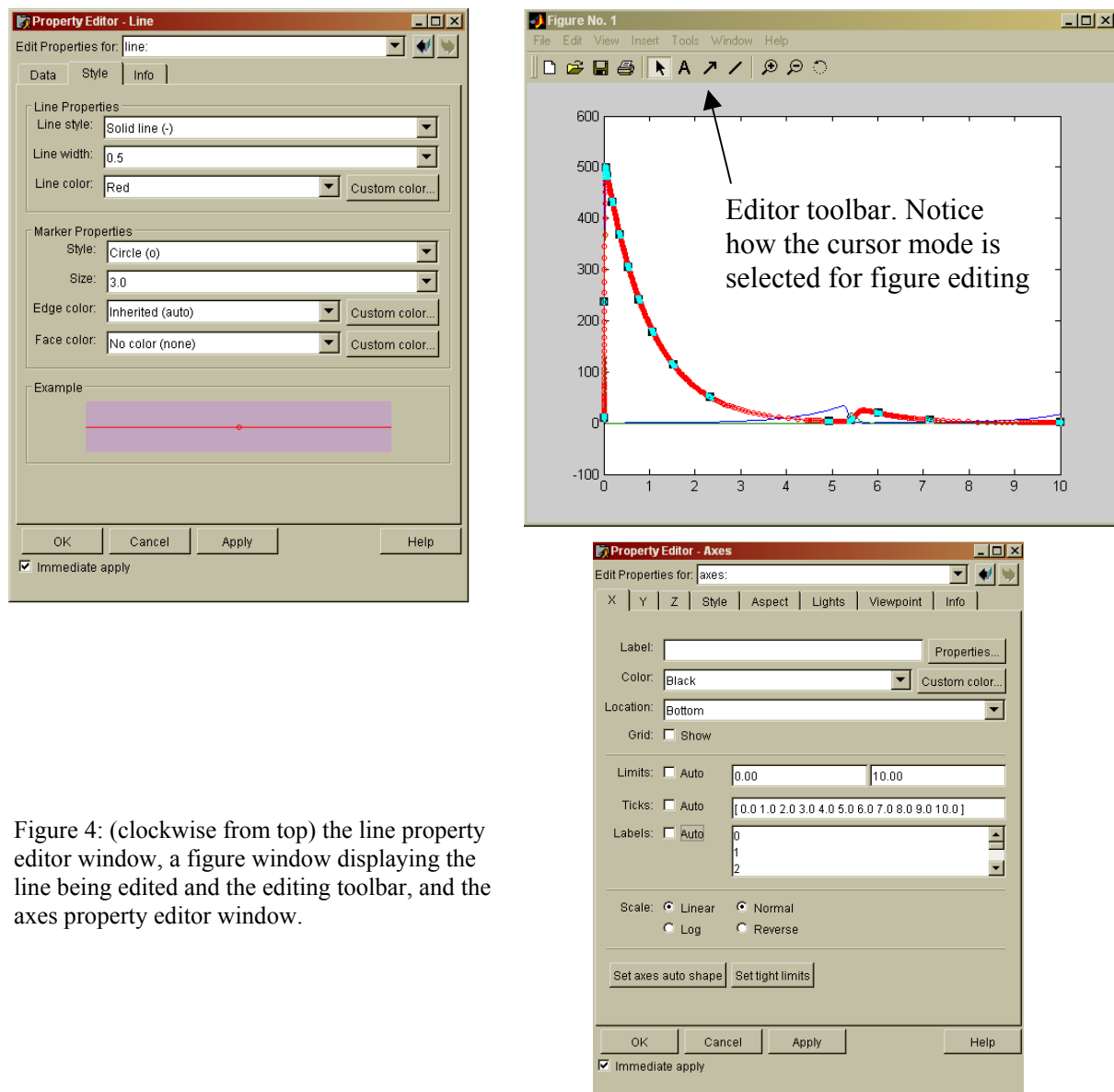


Figure 4: (clockwise from top) the line property editor window, a figure window displaying the line being edited and the editing toolbar, and the axes property editor window.

Graphing Multiple Solutions on a Figure

To add additional plots to an existing graph, use the hold function. After you type hold on, Matlab will add subsequent plots to the selected figure until the command hold off is typed. To select a figure, just click on the figure window or type figure(n), where n designates the desired figure number (the figure number is displayed on the window).

Creating Subplots

The subplot function allows you to have multiple plots in a single window. Type subplot(m,n,p); followed by plot(t,x); to create a m x n matrix of plots. The plot of (t,x) will be displayed in location p. For example, the set of commands:

```
[t, x] = ode45( 'F', [0, 10], [500; 50; 10]);  
[t1, x1] = ode45( 'F', [0, 10], [100; 20; 10]);  
subplot(2,1,1); plot(t,x);  
subplot(2,1,2); plot(t1,x1);
```

gives the series of subplots displayed in Figure 5 (right).

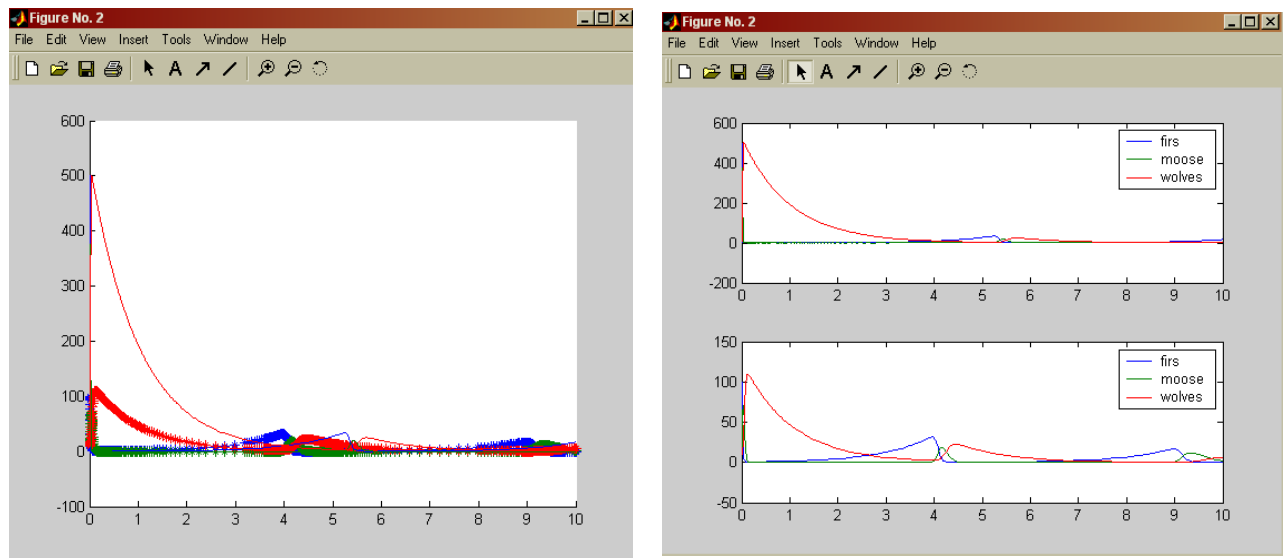


Figure 5: (left) A figure window displaying two plots on one graph. This was done using the hold function. (right) A figure window with two subplots made using the code specified in the text.

Last Words

This is all you really need to know about Matlab for your project. If you have any further questions or if you'd like to learn more about Matlab (it's a pretty neat program), consult the handy Matlab help menu. If that doesn't help you, you're more than welcome to ask me or Prof. Judson for more assistance.