Day One: Welcome to MATLAB

1 Plots

1. In the command window, typing

```
>> x = [1,-3,4,2,-7,8,9]
```

   will save the vector as a variable x. Type the following commands and describe what happens:

```
>> x +1

>> x*10

>>x/5

>>x*x

>>x.^2
```

2. Create a vector containing the numbers from 1 to 10 in order.

3. MATLAB has tools to build list for us, for example the command

```
>> x = 2:10
```

   creates a list of the numbers form 2 to 10 and saves them in the variable x. How would you create a list of all numbers from -100 to 100?

4. Using a:b gives you a list of all the whole numbers (integers) from a to b but what if we want to include decimals? Or we don't want to count by 1? Well, we can just type them in ourselves but we can also use the command a:b:c. This will give us a list

```
[a, a+b, a+2b, ...]
```

   until the list gets to c. For instance, if we want all even numbers from 0 to 10 we could type

```
>> 0:2:10
```

   This will start at 0 and count by 2's up to 10. What commands would you type to make the following lists:

   All odd numbers from 1 to 99?

   All numbers divisible by 8 from 0 to 800?

   A backwards list from 100 to 0?

The list [0, .1, .2, …, 9.8, 9.9, 10]?

5. The function plot(x,y) works as follows: for x = [x1, x2, ..., xn] and y = [y1, y2, ..., yn] it plots the points $(x1, y1)$, $(x2, y2)$ up to (xn, yn) and connects them with lines. For example

```
>> plot([1,0,1],[2,3,-1])
```

plots the lines from (1,2) to (0,3) to (1,-1). Can you plot a triangle?

6. Try plotting a square. You'll notice that nothing seems to happen! That's because MATLAB zooms in so that the square *is* the boundary of the window. We can change the boundary by typing

```
>> axis([xmin, xmax, ymin, ymax])
```

where the x-axis goes from xmin to xmax and the y-axis goes from ymin to ymax. Change the axes so you can see your square.

7. Plot a star.

8. **Color:** There are two ways to change the color of a plot, you can use the command

```
>> plot(x,y, 'Color', [0,0,1])
```

to plot the color specified by , where red, green, blue take on values between 0 (none) and 1 (full). For example blue is [0,0,1], red is [1,0,0] and purple = red + blue is [1,0,1]. Try it now.

For some colors, you can also use shorthand letters, for example

```
>> plot(x,y, 'm')
```

plots in magenta. A full list the shorthand colors can be found in the cheat sheet or in the documentation, by typing doc plot.

9. Plot a yellow star!

10. There are many things you can change about plots. You can change their color, the style of the line, the style of the marker for each point, and more. For example,

```
>> plot([0,2],[2,0], 'b-o')
```

plots a blue line ('b'), dashed ('–') line with each point marked by an 'o'. A full list of commands is given below at the end of this guide. Try to plot your first initial in your favorite color, with your favorite style.

11. Using hold on you can draw multiple times to the same plot. How would you plot a smiley face? Or if you're too baaad for that a skull and cross bones?

12. **Plotting Functions.** To plot the graph of a function in MATLAB we must start with a vector of point we want to plot it over. For instance, to plot $y = x^2$ from -4 to 4 we would type

```
>> x = -4:4

>> y = x.^2

>> plot(x,y)
```

Remember to include to period before the carat! How would you plot $y = x^2 - 2x + 5$?

13. Plot $y = e^x$ for x = -10 to 1. You will have to use the function exp(x) since MATLAB doesn't know what **e** is!

14. Use hold on to plot $y = sin(x)$ and $y = cos(x)$ for x from 0 to 10 on the same plot.

2 Loops: getting a computer to work for you!

A for loop allows us to do something many times, but slightly different each time. Open a new script and type the following

```
for i = [1,3,6,4,2,6,4,5,3]
     i^2
     cos(i)
end
```

What this does is sets i to 1, the first number in the list. It then performs the code inside the loop, ie it computes i squared, computes cosine of i and hits the end. When it hits end it goes back to the beginning and sets i to the second number on this list, in this case 3. It then runs through the code again, sets i to the third number, etc.

Now, the code can be anything. For example, consider

```
for i = 0:10
     x=[0,i]
     y=[10-i,0]
     plot(x,y)
     hold on
end
```

Before running this, what does it do? Well first remember that 0:10 is really . Then describe what the code does. What does the hold on do?

Finally, lets say we want to sum all the number from 1 to 10000. The following code does that by defining a variable total and adding i to it for i each number from 1 to 10000:

```
total = 0
for i = 1 : 10000
     total = total + i
end
```

How can we know that this code actually gave us the right answer?

1. How would you modify to "sum" code to calculate the factorial

$$100!=100 \times 99 \times 98 \times \ldots \times 2 \times 1$$

2. Problem: We want to plot of row of triangles. We now how to make one triangle by specifying he *x*-coordinates in a vector x and the *y*-coordinates in a vector y and plotting them. But we can always move these coordinates around by adding or subtracting from the vectors. For instance, x+1 would shift all the *x*-coordinates by one unit in the x direction.

   The following code draws two triangles sort of next to eachother:

```
x = [0,2,1,0]
y = [0,0,1,0]
plot(x,y)
hold on
plot(x+1,y)
```

   In general, by using plot(x+i,y) we can shift it over by i units. Can you make the two triangles sit so that only a vertex is touching?

3. Now, use a for loop to print 20 triangles next to each other. Can you control their color?

4. Did you know you can put a for loop in a for loop? What do you need to add to the following code to make a grid of triangles?

```
for i=1:10
   for j=1:10
   % your code here!
```

```
   end
end
```

5. The Fibonacci sequence is a sequence of integers often associated with growth in nature. The sequence is generated by starting with $F_1 = 1$, $F_2 = 2$ and then defining the rest of the numbers by

$$F_n = F_{n-1} + F_{n-2}$$

So for example, $F_3 = F_2 + F_1 = 1 + 1 = 2$. The first 6 numbers of the sequence are

F_1	F_2	F_3	F_4	F_5	F_6
1	1	2	3	5	8

What are the first 10 numbers of the Fibonacci sequence?

We want to use MATLAB and a for loop to generate a list of the first 100 numbers of the Fibonacci sequence. To do this, we will define a variable fibSeq and define the first two elements to be 1:

```
fibSeq = 1
fibSeq(2) = 1
```

Now we want to define a for loop

```
for i = 3:100
   % Compute fibSeq(i)
end
```

What code do you need to put in instead of the comment to compute fibSeq(i)?

3 Extra Extra!

1. The function plot3(x,y,z) works like plot except you can plot in 3d! Look at the following code:

```
>> t = 0:.1:10
>> x = sin(t)
>> y = cos(t)
>> z = t
```

```
>> plot3(x,y,z)
```

What do you get? Now, an interesting fact is that if *x,y* and *z* are all functions of sine and cosine they will always form a closed loop since sine and cosine are periodic. For example, something like

```
>> t = 0:.1:10
```

Forms a closed loop, although it's easy to see by rotating the graph that it's just a simple circle. Can you find a set of functions that tie the loop into a knot?

2. What if you want to plot two variable functions, like $z = x^2 - y^2$? This is a little more complicated but we can just use the commands

```
>> [x,y] = meshgrid(-10:.5:10)

>> z = x.^2 - y.^2

>> mesh (x,y,z)
```

Plot the following the functions $z = x^2 + y^2$, $z = xy$, $z = -x^2 + xy + y^2$. Using the MATLAB documentation, what kinds of options do you have to change plots? See if you can make a transparent 3d plot.

Day 1 Cheat Sheet

Reference

lookfor - find functions that match the following key word or phrase

help - find out information about a specific function

doc - like help but bring up the information in the help center.

Vectors/Lists

a:c:b - Create a list of numbers consisting of all the numbers $[a, a + c, a + 2c, \ldots]$ from a to b.

[1,4,6,3,6,4,8,9] Creates a list of numbers.

Arithmetic operations

+, -, *, ^, / - **For Numbers**: Addition, subtraction, multiplication, division and exponentiation respectively.

+, -, .*, .^, ./ - **For Lists**: Term by term addition, subtraction, multiplication, division and exponentiation respectively. You must include the period.

Loops

for i=1:10, do stuff; end - The syntax of a **for** loop. It says set the variable i to 1, then *do stuff*. When you get to end set i to 2 and *do stuff* again. Keep going until after you've set i to 10 and then stop. For example:

```
for i = 1:100

disp(i)

end
```

will display each number from 1 to 100 in order.

Plots

plot(x,y) - For lists x = [x1, x2, ..., xn] and y = [y1, y2, ..., yn] plots the points ($x1, y1$), ($x2, y2$) up to (xn, yn) and connects them with lines.

plot(x,y, 'b–o') - Plots a blue line ('b'), dashed ('–') line with each point marked by an 'o'. A full list of commands is given below.

'Color'

Long Name	Short Name	RGB Triplet
'yellow'	'y'	[1 1 0]
'magenta'	'm'	[1 0 1]
'cyan'	'c'	[0 1 1]
'red'	'r'	[1 0 0]
'green'	'g'	[0 1 0]
'blue'	'b'	[0 0 1]
'white'	'w'	[1 1 1]
'black	'k'	[0 0 0]

'Style'

Short Name	Line Style

'-'	Solid line
'–'	Dashed line
':'	Dotted line
'-.'	Dash-dotted line
'none'	No line

plot(x,y, 'Color', [0,0,1]) - Plots in the color specified by *[red,green,blue]*, where *red, green, blue* take on values between 0 (none) and 1 (full). For example blue is [0,0,1], red is [1,0,0] and purple = red + blue is [1,0,1].

hold on - Don't write over the old plot when drawing a new plot.

hold off - Write over old plots when drawing a new plots.

axis([xmin, xmax, ymin, ymax]) - Scale an already existing plot so that the x-axis goes from *xmin* to *xmax* and the y-axis goes from *ymin* to *ymax*.