Lab: Introduction to Matrices and Plots

1 Matrices

Since MATLAB is short for "matrix laboratory" you may have guessed MATLAB can be used to work with matrices. In fact, MATLAB thinks in matrices! Before we begin to learn MATLAB commands regarding matrices, let's briefly review some matrix notation.

In this document both matrices and vectors will be denoted by bold font. Please do not confuse command names with matrices. Scalars or constant numbers will not be in bold font and will usually be lower case letters. Inverses will be denoted with using the $^{-1}$ notation. In MATLAB there is no bold font or exponent notation and we will not use any of this notation when we are working with the command lines.

1.1 General Matrix Notation

In MATLAB everything is designated as a matrix. Even single numbers are considered to be 1×1 matrices. To see this type:

```
>> a = 7;
>> whos
```

Notice the "size" category returns the value 1×1, which is in matrix form! To enter matrices use square brackets around the entire matrix, spaces for column designations and semicolons for row designations. To enter the matrices:

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} -1 \\ 4 \\ 2 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} -1 & 0 & 2 \\ 4 & 7 & 1 \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$$

Use the following commands:

```
>> clear A B C D
>> A = [1 2 3; 4 5 6]
>> B = [-1; 4; 2]
```

```
>> C = [-1 0 2; 4 7 1]
>> D = [1 2; 4 3]
```

To pull out a single entry from the matrix, use the matrix name, row and column number. If we wanted to look at the value in matrix **A** in the second row and third column, we would type

```
>> A(2,3)
```

This would return the value "6" which is the number in the second row and third column. This is also handy, if you have an incorrect value in your matrix. You could always clear the matrix and retype it in. This method will just save you some time and maybe even aggravation. Assume we really wanted the second row, third column of matrix **A** to be -6 instead of 6. We can change it by using

```
>> A(2,3) = -6
```

Now we have fixed our matrix, without having to reenter it. MATLAB even allows you to look at entire rows and columns by using a colon. To pull off the second row of matrix **A**, type

```
>> A(2,:)
```

and for the first column, type

```
>> A(:,1)
```

When you are working with matrices and especially vectors you may need to take the transpose of them. There are two methods for doing this in MATLAB. The first is to use the **transpose** command and the second method uses an apostrophe.

```
>> transpose(B)
>> B'
```

Another useful command is the **size** command. This will return the rows and columns of a matrix to you in a row vector. The first entry is the number of rows and the second entry is the number of columns. With vectors you can also use the **length** command.

```
>> size(A)
>> size(B)
>> length(B)
```

To create a matrix of zeros (the additive identity matrix) or the identity matrix use the commands **zeros** and **eye** respectively. In the code below we create a 2×4 matrix of zeros and a 3×3 identity matrix.

```
>> zeros(2,4)
>> eye(3,3)
```

You can also create a matrix of all ones with the command **ones** and a square matrix of random numbers with the **rand** command.

```
>> ones(2,3)
>> rand(5)
```

1.2 Addition and Multiplication

To add matrices use the + and to multiply use *, just as you would for numbers. Notice that MATLAB will give you an error that reads:

??? Error using ==> plus
Matrix dimensions must agree.

when the dimensions of your matrices do not allow for addition or multiplication. Try working with these matrices.

```
>> A+C
>> A+B
>> A*B
>> A*C
```

1.3 Inverses

MATLAB is able to find inverses of matrices, when they exist, by using $\wedge(-1)$

```
>> A∧(−1)
```

We should check this to make sure it is true.

```
>> clear all
>> A = [1 2; 4 3]
>> inA = A ∧(−1)
>> A*inA
>> inA*A
```

Okay, now what happens if there is no inverse?

```
>> B = [1 2; 4 8];
>> inB = B ∧(−1)
```

This results in the following error message:

Warning: Matrix is singular to working precision.

You will also get an error message if you try to take the inverse of a non-square matrix. This brings us to our last topic for this matrix lesson, determinants. To find the determinate of a matrix use the **det** command.

```
>> det(A)
>> det(B)
```

Now you know why we could not find the inverse of matrix **B** (i.e. $\det(\mathbf{B}) = 0$). Similar to using finding inverses with MATLAB, if you try to find the determinant of a non-square matrix you will get an error.

2 Graphing

There are several different graphing options in MATLAB. We will be working with the **plot** command for this lesson. The **plot** command has many options within itself. These can be viewed through **help**.

```
>> help plot
```

As you can see, there are several different plotting colors, shapes, line forms , etc. You may have to refer to the help menu every now and then until you become familiar with these options. There is also another method to adjust your plots that we will discuss at the end of this section.

In order to plot a function, MATLAB requires you to have two vectors of the same lengths. One vector is a list of the "x" values and the second vector is the corresponding "y" values. In the first example we will work with the **sin** and **cos** commands. First, we need to create a vector of x-values. Here is a command that will let you determine the beginning and end points as well as the lengths of the spatial divisions in between. For lack of creativity, let's label this vector of x-values **x** and lets graph our trig functions from -2π to 2π with our interval broken up into lengths of $\frac{\pi}{10}$.

```
>> clear all
>> x = -2*pi:pi/10:2*pi
```

Now we need to define our functions. Here let **y1** be the vector of y-values that correspond to the sine of our x-value vector **x** and let **y2** be the y-values for cosine.

```
>> y1 = sin(x)
>> y2 = cos(x)
```

Now we can plots these two functions! Here are several plots for this sine function with slight variations. We will clear the figure in between each variation.

```
>> plot(x,y1)
>> clf
```

```
>> plot(x,y1,'k')
>> clf
>> plot(x,y1,'--r')
>> clf
>> plot(x,y1, 'xr')
>> clf
>> plot(x,y1,'cv-', 'linewidth',5)
```

If we want to plot both functions at the same time, or any number of functions at the same time, we have to use a different method. In order to use this method, we need to enter the next several commands before they are evaluated. Remember to use **shift enter** in between these lines.

```
>> figure(1)
clf
hold on
plot(x, y1, 'b','linewidth',3)
plot(x, y2,'c','linewidth',4)
title('Sine and Cosine Functions');
xlabel('X values');
ylabel('Function values:  f(x)');
hold off
```

We can even get fancier...

```
>> figure(2)
clf
hold on
set(gca,'fontsize',20,'YTickLabels',[],'XTickLabels',[])
grid on
plot(x, y1, 'b','linewidth',3)
plot(x, y2,'c','linewidth',4)
axis([-2*pi 2*pi -2 2])
legend('sin(x)','cos(x)');
title('Sine and Cosine Functions')
xlabel('X values')
ylabel('Function values:  f(x)')
hold off
```

There are many different techniques one can use to plot these function. With the following code we will look at some different functions and use a different method to define our "x values". This time we will look at two linear functions.

```
>> clear all
>> x = linspace(-2,10,50);
>> y = -3*x +1;
```

```
>> z = 4*x -6;
>> figure(3)
clf
hold on
set(gca,'fontsize',10)
grid minor
plot(x, y, 'pr:','linewidth',1)
plot(x, z,'*m-.','linewidth',1)
axis([-2 10 -50 50])
legend('y','z');
title('A Couple of Lines')
xlabel('X values')
ylabel('Function values:  f(x)')
hold off
```

These are just examples of basic graphing techniques. To find more about graphing you can use the graphics tutorial.

3 Practice Problems

1. Create the following matrices:

   (a) $\mathbf{A} = \begin{bmatrix} -1 & 0 \\ 4 & 5 \end{bmatrix}$

   (b) $\mathbf{B} = \begin{bmatrix} \frac{1}{2} \\ 3 \end{bmatrix}$

   (c) $\mathbf{C} = \begin{bmatrix} 2.5 & 1.07 \\ 5.5 & -0.3 \end{bmatrix}$

   (d) $\mathbf{D} = \begin{bmatrix} 2 & 6 \\ -4 & 3 \end{bmatrix}$

   (e) $\mathbf{F} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 2 & 1 & 1 & 1 \\ 1 & 1 & 1 & 3 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$

   (Hint: Use the **ones** command and then alter two of the entries)

2. Perform the following manipulations, if possible

   (a) $\mathbf{A} + \mathbf{C}$

   (b) $\mathbf{A} * \mathbf{B}$

   (c) $\mathbf{B} * \mathbf{D}$

   (d) $\mathbf{A}^5$

(e) $\mathbf{A} * \mathbf{C} * \mathbf{D} * \mathbf{B}$

3. Does matrix $\mathbf{F}$ have an inverse? If so, find it.

4. Plot the function $y(x) = 3 + 2sin(\pi * x)$ from -1 to 10 with a blue line and hexagon points.

5. Plot the function $f(x) = 0.5e^x$ in red and the function $g(x) = 0.859x + 0.5$ in green from $x = 0$ until $x = 1$ using the following specifications:

   (a) font size 15, title, legend, x-axis label, y-axis label

   (b) font size 10, title, legend, set the y-axis to be between 0 and 1.5