CIS 401: Problem Set 1

Essential Knowledge–Please give a brief answer (1-2 sentences) for each

1. Assume A is a variable on your workspace. Contrast the information obtained by the commands whos A, length(A), and size(A).

   whos A gives you all the information: name, size, type, and memory used; however, this information would be hard to incorporate into a program.length(A) gives the maximum of any of A's dimensions, and thus, is more useful for vectors. size(A) returns the number of rows and columns of the array, and calling [m,n]=size(A) allows you to capture the size information in variables that can be used later.

2. How would you access every second row in a 10-by-10 matrix A?

   A(2:2:10,:)–Note, I would also accept A(1:2:9,:) since I didn't specify whether to start at row 1 or 2.

3. Construct an index vector to access all of the rows of A (as defined above) *except* for the first, fourth, seventh, and tenth rows.

   There are a couple of ways to do this. This simplest is just to create the index I with the rows you want:

   ```
   >> I=[2,3,5,6,8,9];
   ```

   However, this would be hard if A was really big. In that case another option is to use Matlab's set operators:

   ```
   >> I=1:10;% creates set of all rows of A

   >> I=setdiff(I,1:3:10)% removes [1,4,7,10] from I
   ```

4. What is the dimension of C in C=A*B if A and B have the sizes in the table below? Try figuring them out yourself and then check your answers with Matlab.

	size of A	size of B	size of C
a.	scalar (1-by-1)	2-by-2	2-by-2
b.	4-by-2	scalar	3-by-2
c.	3-by-1	1-by-3	3-by-3
d.	1-by-5	5-by-1	scalar
e.	2-by-4	4-by-2	2-by-2
f.	2-by-4	2-by-4	error

Playing with Matlab: Legendre Polynomials

Programming: Signal Analysis

A common form of data encountered in many fields is a *signal*–a quantity observed at regular intervals in time. Examles include: sound waves (pressure vs. time), earthquakes (displacement of Earth vs. time), fish-catch statistics (number caught vs. year), and stock performance (share value vs. day). Because signal data is so common, many techniques have been developed to analyze this data. In this exercise, you will investigate a common technique known as *Fourier analysis*. The purpose is to reinforce working with arrays and plots and practice using Matlab to explore a new algorithm.

Fourier analysis hinges on the mathematical idea that it is possible to represent many functions as a weighted sum of sines and cosines. Suppose we observed the value of a function x at several points in time separated by time interval dt. Using this notation, we can express the N-point Fourier approximation of x as:

$$x(t(j)) = a_1 + \sum_{k=2}^{N/2+1} a_k \cos(\frac{2\pi(k-1)t(j)}{Ndt}) + b_k \sin(\frac{2\pi(k-1)t(j)}{Ndt})$$

The value of a_k's and b_k's indicate how much the frequency $\frac{k-1}{Ndt}$ contributes to x.

In Matlab, we would represent t and x as vectors. We can also represent a and b as vectors of length $N/2+1$. Since $\cos(0) = 1$ and $\sin(0) = 0$ we can bring a_1 and b_1 inside the sum (the value of b_1 doesn't matter, let's assume it is 0). This gives us:

$$x(t(j)) = \sum_{k=1}^{N/2+1} a_k \cos(\frac{2\pi(k-1)t(j)}{Ndt}) + b_k \sin(\frac{2\pi(k-1)t(j)}{Ndt})$$

If we view a and b as column vectors ($N/2+1$-by-1) the expression starts to resemble multiplication of matrices by vectors:

$$x = C * a + S * b$$

where C and S are p-by-n matrices containing all of the sine and cosine info and a and b are vectors containing the coefficients. x will then be $p - by - 1$ and contain the value of x at each of the p time observations. How do we get C and S? That's your job, but I'll give you some hints. First, notice that the stuff inside both the sin and cos statements is the same, let's call it F. This gives us

$$C = cos(F) S = sin(F)$$

Since cos and sin operate element-by-element, if C is to be p-by-n, F better be p-by-n.

5. What are the variables used to construct F and what are their sizes?

   To build F, we need three scalars (pi, dt, and N) and two vectors. The first vector is the vector of times t, which is p-by-1. The second should contain the $k-1$'s, which should be 1-by-n.

Now that you know the sizes of the variables used to create F, the next step is to figure out how to use them to create a p-by-n matrix. Make sure that the jth row in F contains the $N/2+1$ frequencies each multiplied by $t(j)$.

6. Come up with a Matlab expression for F? If this isn't obvious, try creating some small vectors and playing around.

   First, create the vector with the k-1 values:

   ```
   >> kvec=(1:N/2+1)-1;% or 0:N/2
   ```

   Then, we can create F with an outer product (what else?)

   ```
   >> F=2*pi/N/dt*t*kvec;
   ```

We want the Now that we know what F is, we can begin to explore the Fourier series.

7. Let dt=0.1, and let t=[0:dt:20*dt]. How would you represent

$$x = 3 + 0.5 * \cos(2 * \pi * 2 * t) - 0.75 * \sin(2 * pi * t)$$

using the notation above? I.e. what should a and b be, and how is F composed? To check that your answer is correct, create a vector x by entering the equation above. Then, create a new vector $x2$ using your a, b, and F. The vectors x and $x2$ should be exactly the same. You can examine this graphically using the plot command. plot(t,x,'b') will plot x in blue. You can plot $x2$ in red on top of x by issuing hold on; plot(t,x2,'r') (hold on tells Matlab not to delete the previous plot. To clear the plot and start over, issue clf). You should play around with other possibilities for x to make sure this makes sense.

Looking at the equation for x you should notice that we have three frequencies: $2 * \pi * 0$, $2 * \pi * 1$, and $2 * \pi * 2$. Also notice that these frequencies seem to be missing the $\frac{1}{N*dt}$ component. This suggests that we need to make $N * dt$ equal to 1. Since $dt = 0.1$, $N = 10$. With $N = 10$, kvec will be 1-by-6, and we will have a lot of frequencies we don't need ($2 * \pi * 3$, $2 * \pi * 4$, and $2 * \pi * 5$). What to do? The answer is pretty simple (but a bit tricky): make the values in a and b corresponding to these (or any other missing terms) equal to 0. This means that a and b are

```
>> a=[3, 0, 0.5, 0, 0, 0]';

>> b=[0, -0.75, 0, 0, 0, 0]';
```

Using these arrayas, the complete code for computing x is then

```
>> dt=0.1;t=[0:dt:20*dt]';N=10;

>> kvec=(1:N/2+1)-1;% or 0:N/2

>> F=2*pi/N/dt*t*kvec;

>> x=cos(F)*a+sin(F)*b;
```