CIS 401: Problem Set 2

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

1. You downloaded a large set of data from the web. You type load and the data appears on your workspace as an n-by-2 array. The description of the data said the first column contains the observation times and the second column contains the data values. Missing data is indicated by -999. You want to compute the mean of the data which you can do using the Matlab command mean (type help mean to learn about it). Give some lines of Matlab code that will compute the mean of the data (the second column). You must find a way to exclude the -999's from the computation. Assume that the name of the data array is "data."

2. You've been asked to help plan for a robot exploration of Mars. The planetary geologists running the show have listed a series of points (in Martian latitude and longitude) where they would like the robot to visit. You need to determine whether the robot will have enough fuel to visit all of these points, and this involves turning Martian lat-lon into meters. The function ll2xy.m in the Problem Sets section of the web page does this for Earth. (Please do not use the version of ll2xy.m used in the SSHvel example of Lecture04. It uses nargin which we haven't discussed, yet.) To perform the computations on Mars, you need to substitute the Martian radius for the Earth's radius. Since you expect to perform these kinds of calculations for several different planets or moons (you've applied for a job to work on a lunar rover), how would you modify ll2xy.m so that you can prescribe any radius you want? You should only have to change a couple of lines, and you are not allowed to use the input command to ask for the radius. Be sure your solution will work for any planet or moon.

Programming: More Signal Analysis

Last time you developed a representation of a Fourier series using Matlab matrix-vector operators. Now for the real magic. One of the great achievements of 20th Century numerical analysis was the invention of a fast algorithm to compute a and b–the coefficients for the Fourier series, from an

arbitrary signal[1]. This algorithm, the fast-Fourier transform (FFT), can be used to decompose a signal into it's composite frequencies. If the FFT says that a or b is especially strong at a particular frequency, then we know that the data has some periodicity at that frequency. We will use this to find the dominant frequency of a time series of water temperature.

Clear your workspace and download `scrippsSST.txt` from the course web site. This file contains the monthly mean sea-surface temperature (SST) measured from the Scripps Institute of Oceanograpy's pier in La Jolla, California. Open the file in a text editor. This file contains several columns of data, but we are only interested in column 3 (DecMonth–time in years) and column 5 (SIOSST–the temperature data). First, you must load this into Matlab and then you will extract the two important columns (I'd call one "t" and the other "temp").

4. Describe what you did to scrippsSST.txt in order to get it into Matlab and how you extracted the columns.

Plot temp. Do you notice any patterns? We will now use the Matlab function fft to figure out a and b. y=fft(temp) will return a vector of the same size as temp (length N). If you type whos, Matlab will tell you that y is a "double array (complex)." Complex numbers are a natural way to represent Fourier series since since $e^{ix} = \cos(x) + i\sin(x)$. Most people (me included) find it easier to think in terms of sines and cosines rather than complex exponentials. Fortunately, we can get a and b from y:

```
>> a = real( [y(1); 2*y(2:N/2+1) ] )/N
```

```
>> b = [0; imag( y(2:N/2+1) ) ]*-2/N
```

. The commands real and imag return the real and imaginary parts of a complex number. Once you have a and b, you can verify that the Fourier series described by these vectors is a good approximation to temp using the scheme you developed for problems 5 and 6 on problem set 1. One slight difference: the equation I gave you for problem set 1 assumes that t(1)==0. The more complete equation is below:

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

[1]Actually, Gauss discovered the algorithm in the 19th C, but didn't publish.

Your job is to "package" `fft` in a function (call it "myfft.m") that takes in a signal (vectors of time and data value) and returns `a`, `b`, and the frequencies `f` at which these vectors are defined (In the expression $\sin(2\pi ft)$, f is the frequency). You should also develop another function (call it "FourierMat.m") to which you can pass `a`, `b`, and `f` and a vector of times and it will compute `x` at those times (i.e. implement you answer to problem 5 from before as a function).

5. Paste your version of myfft.m. This function should be properly commented and should perform any error checking.

6. Paste your version of FourierMat.m. This function should be properly commented and should perform any error checking.

You should now be able to prove to yourself that a Fourier series is a good approximation for many functions. But, what's the point? Fourier analysis (aka spectral analysis) is done to learn properties about a signal. The backbone of spectral analysis is the spectrum: a plot of the "energy" at each frequency. The spectral energy is simply $\sqrt{a^2 + b^2}$.

7. Plot the spectrum of the temperature data (DON'T send it to me). Does the spectrum contain any especially energetic frequencies (peaks)? If so at what frequencies do they occur, and how do they relate to the original data?