Lab: Math 350

The function `vander1.m` is on our class website.

1. First, we look at polynomial interpolation. The $x-$values will be 40 evenly spaced points in the interval $[-4,7]$, and the $y-$values will be a noisy sine function. For example,

   ```
   x=linspace(-4,7,40);
   x=x';  %Makes x a column
   y=sin(x)+0.1*randn(size(x));  %0.1 is the standard deviation of noise
   ```

   (a) Use Matlab's `vander` command to build the polynomial that interpolates the data. Show the original data (red asterisks) and then evaluate the polynomial at 200 points in the interval $[-4,7]$ and plot the curve on top of the data.

   (b) Use our `vander1` command to build polynomials of degrees 5, 15 and 20 and repeat the plot that you previously obtained (one plot for each polynomial).

   (c) For each of the examples, was the error on the data increasing or decreasing with the degree of the polynomial (Hint: The error on the data using the full Vandermonde matrix should be zero since we are interpolating).

   ```
   z=polyval(a1,x);  %a1 is the vector of coefficients
   Err1=sum((z-y).^2);
   ```

2. Is there a way that we can quantify the "badness" that we see with increasing the degree of the polynomial?

   The error on the *training data* decreases to zero by increasing the degree of the polynomial. However, the error on the *other points in the interval* start to *increase* with the degree of the polynomial.

   Idea: Break up the data into two sets. On one set, "train" the model (finding the polynomial coefficients). Use the second set of data to measure the error of the model. What we should see is that the error begins to decrease, then at some point starts to increase (with increasing polynomial degree).

   Download the file `LabNov18.m` and try it out. Find the best degree for your data.