MA 3457 / CS 4033
HW #3
Due: Tuesday 11/10 by 11 pm

This assignment is due on **Tuesday 11/10 by 11 pm on Canvas**. Your assignment submission should contain a few files. You should submit all of your Matlab code and it should be properly commented to explain what the code is doing. You can submit as separate m-files saved as HW#Q# OR you can submit a single word or text file with all of the code pasted in (specifying or delineating code for each problem). For additional written work and discussion of problems, this should be a single pdf that is well-organized and either typed or neatly written. (If hand-written, use an app to scan and save as a single pdf). This file should be saved as HW#. To receive full credit on a problem, the code must run with no errors and the written work/discussion of the problem must also be complete. Matlab output should be discussed in the write-up.

1. (10 points) *Cubic Splines*
   Here, we will assume cubic splines for a set of n ordered data points $(x_1, y_1), \ldots, (x_n, y_n)$ are of the form:
   $$S(x) = \begin{cases} s_1(x) & x_1 \le x \le x_2 \\ \vdots & \vdots \\ s_{n-1}(x) & x_{n-1} \le x \le x_n \end{cases}$$
   where the $i = 1, \ldots, n-1$ splines are defined as:
   $$s_i(x) = a_i(x - x_i)^3 + b_i(x - x_i)^2 + c_i(x - x_i) + d_i$$
   It turns out we can use properties of an interpolating function and assumptions of continuity of splines to determine equations for the coefficients in terms of $b_i = M_i/2$ where:
   $$M_{i-1} + 4M_i + M_{i+1} = \frac{6}{h^2}\left(y_{i-1} - 2y_i + y_{i+1}\right), \quad i = 2, \ldots, n-2$$
   with $h = x_i - x_{i-1}$ for $i = 2, \ldots, n$. With Boundary Conditions specified for conditions on M_1 and M_{n-1}, we can write this system of equations as a linear system $A\vec{M} = \vec{R}$ where we need to solve for $\vec{M}$. Once we have $\vec{M}$, we then have $b_i = M_i/2$ and $d_i = y_i$ for $i = 1, \ldots, n-1$. Then c_i and a_i can be determined based on b_i for $i = 1, \ldots, n-1$. Refer to slides 16-17 of power point from Tuesday Nov 1st class.

   (a) (5 points) It turns out that the matrix A is diagonally dominant. Explain what this property is, why you can see it is true for rows $2, \ldots, n-2$, and how it leads to knowing we can solve for coefficients uniquely.

   (b) (5 points) We would call A a sparse matrix since it has many entries that are 0's. As we know from earlier discussions about error, each 0 entry will be represented by its floating point representation in the computer where $0 = fl(0) + \epsilon$. From an error and computational storage standpoint, why does it make sense to use a sparse matrix representation where you only store non-zero entries in the matrix? (Note that Matlab and most programming languages have sparse matrix utilities)

2. (15 points) *Discrete Least Squares*

   (a) (7 points) Write a code that takes an input of D for the degree of the least squares polynomial that you want to fit to the data. The data points are x =[1, 1.1, 1.3, 1.5, 1.9, 2.1], y =[1.84, 1.96, 2.21, 2.45, 2.94, 3.18].

(b) (5 points) Determine the discrete least squares polynomials of degrees 1, 2, and 3 for the data. Create a plot for each polynomial fit. Plot both the data points (using a symbol for data in Matlab such as `'og'` or `'xg'`) and also plot the polynomial at additional points `xeval=linspace(0,3,100)` with the data.

(c) (3 points) Compute the error in each polynomial approximation of degree D, $E_D = \sum_{k=1}^{m}(y_i - p_n(x_i))^2$ for m data points.