ECE 595 Homework 1 Due: 5 PM, Feb.14

The data file AutoData\_HW1.xlsx contains a subset of automobile MPG (miles per gallon) data along with vehicle information, the number of cylinders, displacement, horsepower, weight, acceleration and the vehicle name. Consider the MPG as the ‘output’ data given the other vehicle information as ‘input’. There are *m= 50* observations in the training data.

1. **Univariate Linear Regression**: Consider the **weight** of each vehicle (column 4) as input and the corresponding MPG (column 6) as output. Get *m*, the length of observations in your code. Normalize the input x as

, *i = 1, 2, …, m*

Obtain a two-parameter linear regression model. You may initialize with [0 0]t for the parameters and terminate after ITNS (=2000, for example) iterations, or when the cost function differs by no more than, say, 1E-4, or whichever occurs first. The cost function is given by

 ,

for the hypothesis,

 with *x0 = 1*.

Using batch gradient descent, update *θ0* and *θ1* simultaneously as given by

 for j = 1, 2 (here).

Make sure to read the correct columns for input and output and augment the input column with 1 for each observation. **Use vectorized operations as much as possible.** Use a learning rate of = 0.01; also, try = 0.05.

(You may try your linear regression code without normalizing *x* and comment on the result.)

You need to turn in the following.

1. Scatter plot of y vs. x. label axes with ‘weight’ and ‘MPG’
2. Linear hypothesis plot – the final straight line – on the scatter plot of y vs. x
3. Plot of J vs. Iteration index
4. Minimum J and the hypothesis parameters
5. Predicted output for the weight *x = 3100*
6. Your code

In MATLAB, using *A = xlsread('AutoData\_HW1.xlsx');* reads all 50 rows of data, except the vehicle information, which is in the last column. Now use only the columns for weight (x) and MPG (y). Sample code for reading data and augmenting the input data is given below.

A = xlsread('AutoData\_HW1.xlsx');

x = A(:,4);

y = A(:,6);

X = [ones(m, 1), data(:,1)]; % Add a column of ones to x;

% DO NOT normalize the column of 1’s!!!

**b. Polynomial Regression and Feature Scaling**: Using the weight as the input variable *x* as *x1*, create features *x2* *= x2*, and *x3= x3*, run a linear regression with 3-features and obtain  for the model  so that the hypothesis can be plotted as a function of the input variable *x*.

Since the input variable *x* (weight) is in the thousands, its squared and cubed values will be on the order of millions and billions. To effectively use these values without slowing down convergence, each of the three features needs to be scaled – you may use feature normalization as defined by

 where  is the mean and is the standard deviation for *j = 1, 2, 3*; Check how  is defined in MATLAB/Octave. You need , which is given by *std(xj,1)*.

Here again, initialize with zeros for the parameters; use = 0.5 and terminate after 1000 iterations.

**Use vectorized operations as much as possible and avoid *for* loops.**

Turn in the following.

1. Cubic hypothesis plot on the scatter plot of y vs. x
2. Plot of J vs. Iteration index
3. Minimum J and theta, the hypothesis parameters
4. Predicted output for *x = 3100*
5. Your code