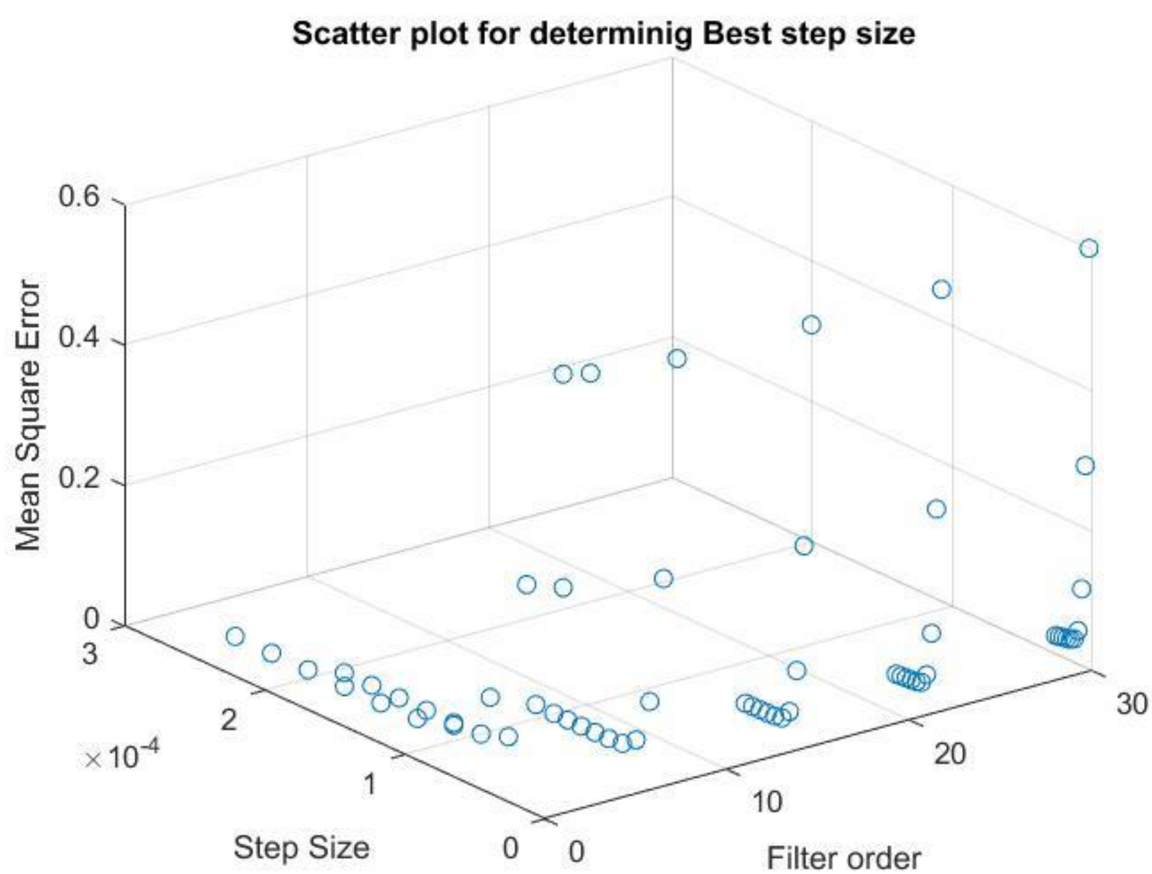
Assignment 3
Name : Harshit Vijayvargia , UFID :19355645

Question1. (3 points) In this problem, the two hyper parameters are the Filter order and the step-size. Repeat the validation of these two parameters and produce a three-dimensional plot of the MSE versus step-size and Filter order. Is the step size related to the Filter order or independent of it?

Solution : The equation for Stochastic descent in LMS is :

$\hat{w}(n + 1) = \hat{w}(n) + \mu u(n)e^*(n)$.

The orders I took for my analysis are: 3,4,8,15,22,30 and the step size which I have chosen varies from (0 to 2/Max Eigen (R)). Guessing the step size is tricky. I took the maximum Eigen value of Auto-correlation matrix R to determine this range but in various cases it is not possible to obtain. In such situations I would have taken trace of matrix R at each sample. After this for each filter order and step size I calculated the parameter w using LMS gradient descent algorithm. This parameter calculation was done using training set. Once I obtain parameter w, I used validation set to determine MSE. This is the Scatter plot I obtain:



The filter order and the step size at which I obtained the minimum MSE are 3 and step size (0.000129981348466126)

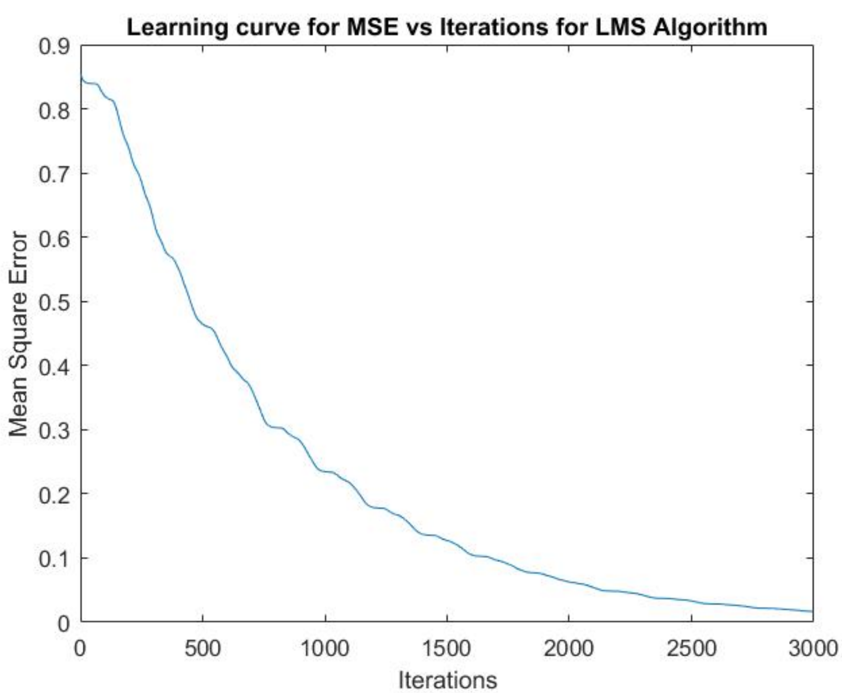
Part-2: Dependency of step size on filter order:

As we know that in order for converging our parameter to optimal parameter our step size should be in a range (0 to 2/Lmax(Eigen value)) where Lmax(Eigen value) is the maximum Eigen value of Auto- correlation matrix R. The step sizes which I have taken are in this range since the step size for convergence depends on a factor Eigen spread(Lmax/Lmin). This Eigen spread is dependent on order of the filter(As order of the filter determines Matrix R). So we can say that the step size at which we come closer to convergence is dependent on order of the filter. But if we keep the filter order fixed then what we observe is that the final parameter value is dependent on step size only.

Question:2) Plot the learning curve of the filter for the best step-size. Pick two other step-sizes

to show that the filter learns slower when compared to the best step-size?

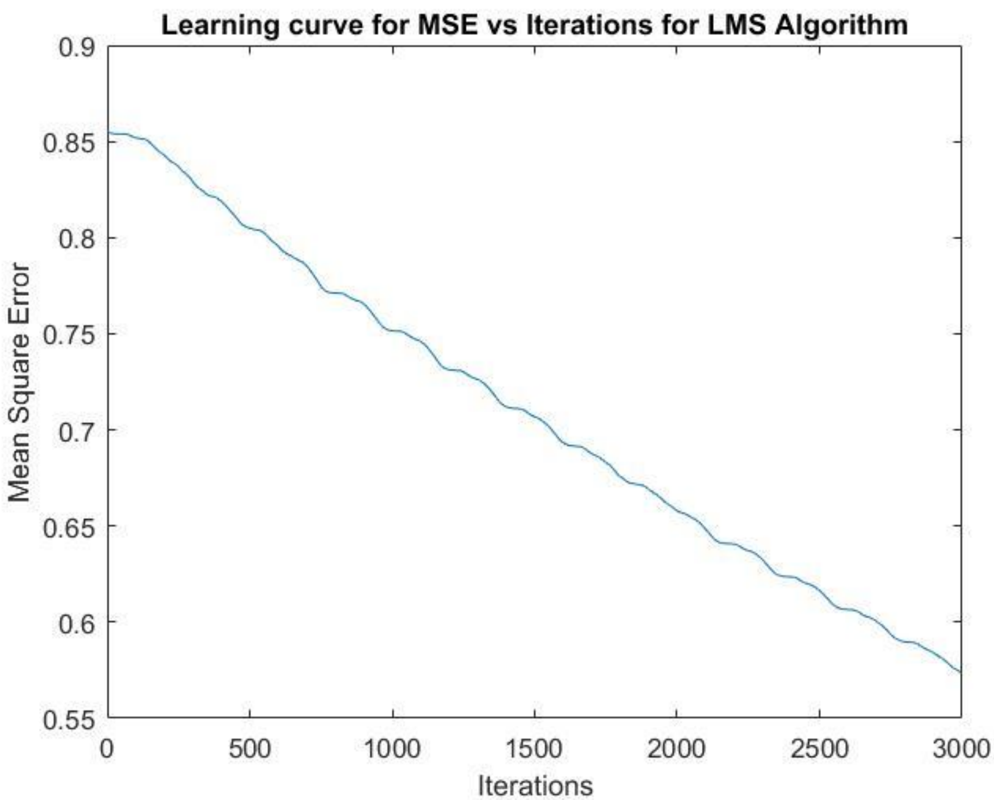
Ans: The Learning curve which I obtained for filter order 3 and step size 0.000259962696932252 (hyper parameters where my MSE was minimum) is :



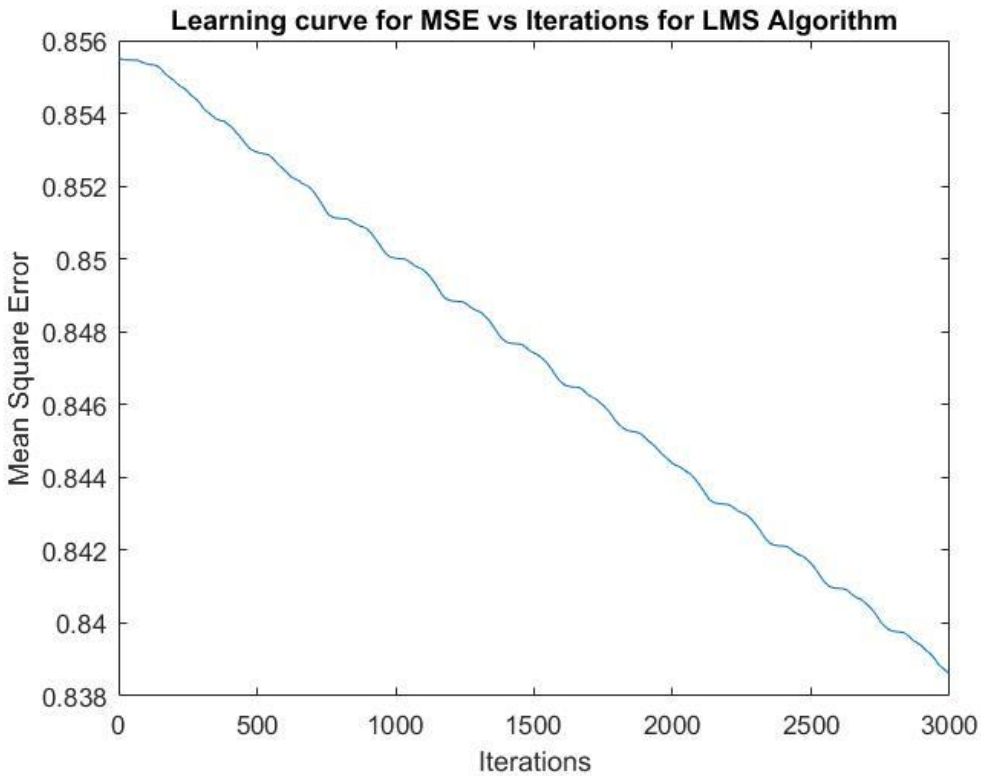
The algorithm we are using is stochastic. Here we are calculating the statistics at each sample which is introducing gradient noise in our algorithm. The observation which I made on comparing this Learning curve with other curves is that My algorithm is learning fast with higher step size because the number of iterations I have are limited so to converge fast higher step size will be better but in the long run or if I would be having more iterations then smaller step size would have given more accuracy.

For same filter order and for two different step sizes less than mine I obtained the following Learning curves: Clearly it can be seen that learning rate for less step size curves are less i.e they will take more iterations to converge.

Order 3 : step Size : 2.59962696932252e-05



Order 3: Step Size : 1.29981348466126e-06 (Decreasing it more) :



3) Repeat the problem using the normalized MSE algorithm. Which algorithm provides the best performance for real world signals? Which algorithm should you be using for regression? Provide answers for both questions and corresponding explanations?

Solution :

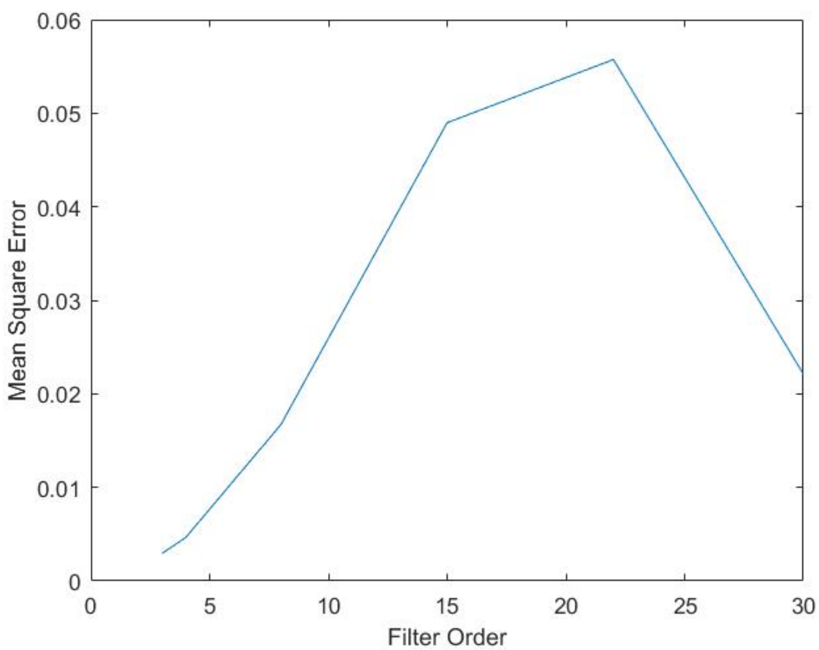
For Real World Signal Normalized LMS will provide better performance. The reasons for this is:

1) Here in our analysis we have assumed that our input signal is stationary which is not the case usually in real world. In real world the amplitude of the input signal can increase the gradient noise in our algorithm thus affecting step size. For Example when we are using filter for noise cancellation, channel equalization etc. To avoid this, we scale our step size by dividing it with the variance of input signal. So in this case we are not choosing step size manually but at each sample calculating the trace of R and taking (d/(trace(R) + c) as

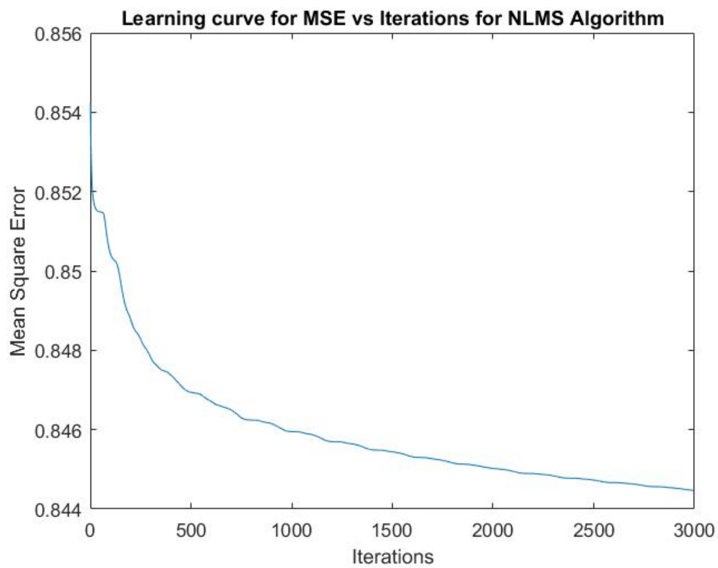
step size, here c & d are constants. Constant c ensures that our step size does not gets zero when trace(R) is zero.

2) When we don't have much statistics information of the input then it is better to use NLMS as it calculates statistics at each sample.

The Plot which I get here for different order of filter and there MSE is: It can be observed that MSE increases with filter order but decrease after a certain point.



The Learning curve I obtain for NLMS for filter order 3 is :



We observe that this is smoother compared to Learning curve obtained in LMS. It can be observed that NLMS converges fast.

Part b: For regression It depends on data as well as our performance requirements. If we are given less statistics about data, then we will use NLMS. If we have to choose between LMS vs NLMS then if data is non-stationary, NLMS will be better. Normally it is observed that NLMS converges faster than LMS.
For higher step size NLMS and LMS gave same average steady state error but NLMS converged faster. When I increased the step size of both then it was observed that convergence speed was same for both but for NLMS gave lower average steady state error. **To conclude NLMS is better but complexity of NLMS is more**.

4)Finally, compare the results of the best LMS with the results of homework 2. Given

the performance and the computation cost, which algorithm would you use? Why?

Solution :

The MSE obtained using analytical approach was 0.000000326 and the MSE obtained through LMS is 0.0009152. Accuracy for analytical approach is better but performance wise LMS will be better for time series data.

Since the data is Non – stationary LMS/NLMS will give better performance. We used analytical approach in homework2 considering our data was stationary.

In general difference between two methods can be given as:

In Analytical approach the weights which we were obtaining were exact weights for obtaining minimum MSE value (fixed trajectory) but in LMS our weights are predicted weights as we are doing stochastic gradient descent so there is randomness in this algorithm and thus if performance is the key then I will use analytical approach or Normal equation to determine optimal parameters.

If input size is too large and data is Non-Stationary and our concern is computation cost then we will use LMS/NLMS as in analytical approach we have to calculate inverse of matrix which can take a lot of time.