Homework I

Neural Network Signal Processing

Rishab Goel

Date 26th January 2016

Problem 1:

Our goal is to do system identification in the following nonlinear system (assumed unknown).
$y(n) = y(n-1)/0.1 + y_2(n-4) + \sin_2 x(n-3)$

We just observe the input x(n) (a white noise Gaussian sequence with unit variance) and the output y(n). Create 2,000 samples of the input x(n) and of the output y(n), starting from a zero initial condition. Select a FIR filter of order M=5 and M=15.IS the filter linear?

Design a Wiener filter ($w=R^{-1}P$) that will approximate the input-output map. Test the quality of the solution with MSE and also by verifying the degree of whiteness of the error signal. Implement also the LMS algorithm, i.e. the iterative solution that uses gradient information.

Estimate the largest stepsize. Estimate the convergence time. Show the difference between batch of 100 samples and on-line learning for the same step-size. Show the weight tracks. Compare the performance of the LMS for the best set of parameters in terms of misadjustment and speed of convergence by varying the stepsize. Experiment with different filter orders to see if the performance improves.

Ans. FIR Filter is a linear filter

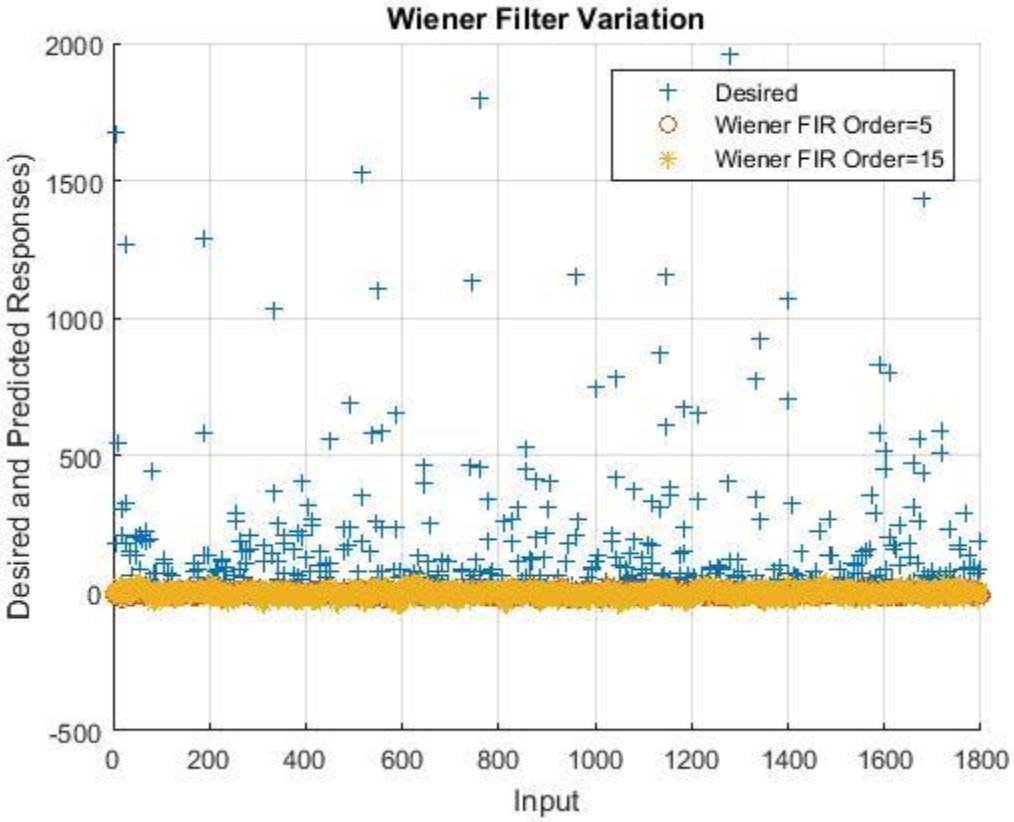
Description:

Wiener Filter is direct computation of the weight matrix of static input sample to predict the desired input. The autocorrelation of the input matrix is inversed and multiplied with cross correlation of the desired to input matrix to get weight matrix. If Autocorrelation is not full rank so inverse won't exist, so adding an Identity with lambda multiplier to make it full rank. MSE for this method for the test data

Correction: We introduced the lags required in the computation of autocorrelation and cross correlation functions to get the impulse response for the order of the respective filter.
Wiener Filter solution: Chosen lambda = 0.001 to ensure inverse exists

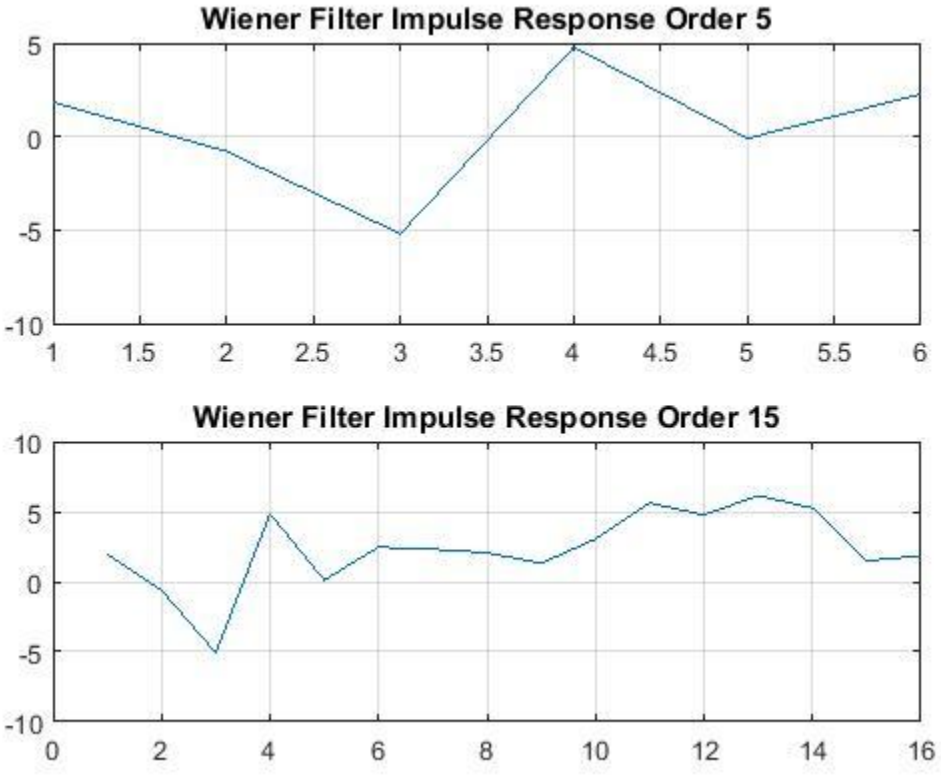
```
[Lx,~] = size(x);
L= L+1;
[Lx,~] = size(x);

rxx = xcorr(x,L-1,'biased')/Lx;      % autocorrelation sequence over (N-1)
lags
R   = toeplitz(rxx(L:-1:1),rxx(L:2*L-1));
%                            covariance matrix
   % cross-correlation vector
[p,~] = size(R);
% If Autocorrelation is not full rank so inverse won't exist
% So Adding a Identity with lambda multiplier to make it full rank
if rank(R) < p
    R = R + lambda * eye(p);
end
```

```
rdx = xcorr(y,x,L-1,'biased')/Lx;    % cross-correlation sequence over (N-1)
lags
P  = (rdx(L:2*L-1));
weights = inv(R)*P;            % Wiener-Hopf solution
pred = filter(weights,1,x);
```

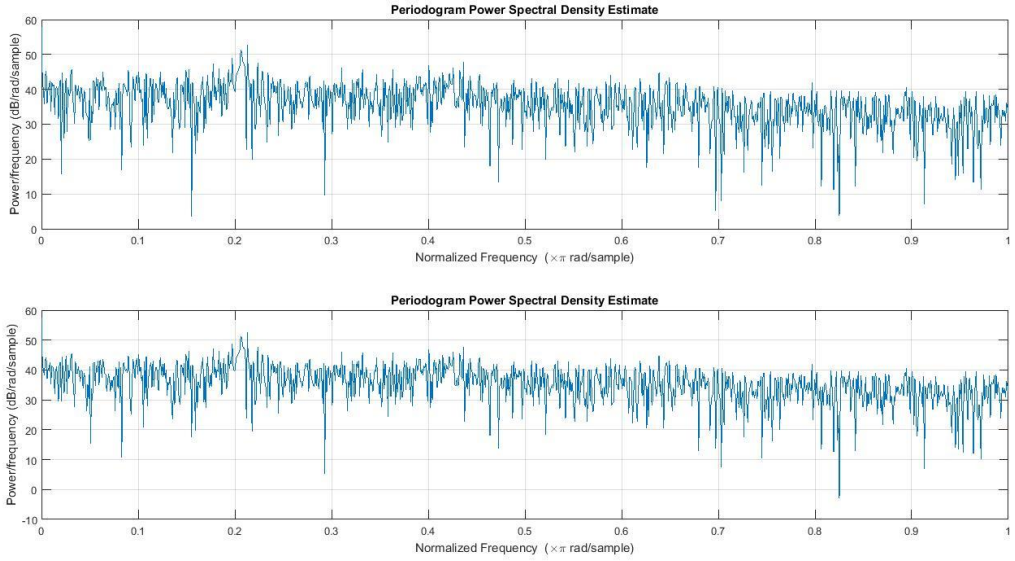


Wiener Filter response for both the order of the filters seems to be concentrated near the region of the response where most the data points are concentrated. Wiener Filter is known to minimize noise and predicted response seems to satisfy that theory as there no spurious variation. The autocorrelation computation is with bias so that autocorrelation does not goes out of bound.

We display the Impulse response of the weights for respective order filter below which were used to predict the above response. For the filter order of five we would have impulse response with six weights and for the filter order of fifteen would have impulse response with sixteen weights.



W_mse5 = 2.2598e+04 W_mse15 = 2.2476e+04 , but we have lower MSE for higher order(15) filter than lower order(5) filter. The degree of whiteness if taken as fft(autocorrelation of input) it gets the same value as MSE. We have plotted power spectral density plot of error which is related which we assume to more closely to the degree of whiteness of the signal. Wiener Filter for this type of data does not appear to be an optimal solution as MSE is too large. In the below graph, the top graph is for filter order =5 and bottom graph is for filter order=15.



The above graph displays the power spectral density plot (or degree of whiteness) for the wiener filter and shows a very similar response.

Online LMS: The weights are modified for each s

```
%% iterating until max iterations with input samples of the order
%% Storing J (MSE) and w_tracks ( Weight tracks)
for iter = 1 : max_iter
   for i = order : N+order-1
            u = x(i:-1:i-order+1 , :);
            e = y(i:-1:i-order+1 , :) - u * w';
            w = w  + (lambda* u' * e)';
            w_tracks(k,:) = w;
            k = k + 1;
    end
    [J(iter), ~] = MeanSquareError(w', x, y);
end
```

Description:

It is clearly observed FIR filter 15 performs better than FIR filter 5.

The Etha0 assumed to start with the training is

```
eta0 = 0.05;%eta0 be 0.5 to increase the convergence rate
R = in_train' * in_train;%% input autocorrelation
e = eig(R); %% eigenvalues of input autocorrelation
trace = trace(R); %% trace of input autocorrelation
% eigen value 'e' of input auto correlation matrix to find range and speed
% of convergence of eta
eta_max = 2/(max(e));
eta = eta0/ trace;
```

The Etha0 was chosen higher as the Jmin started to diverge very fast for higher order FIR filter 15 and lower value is not taken to ensure fast convergence in terms of iterations

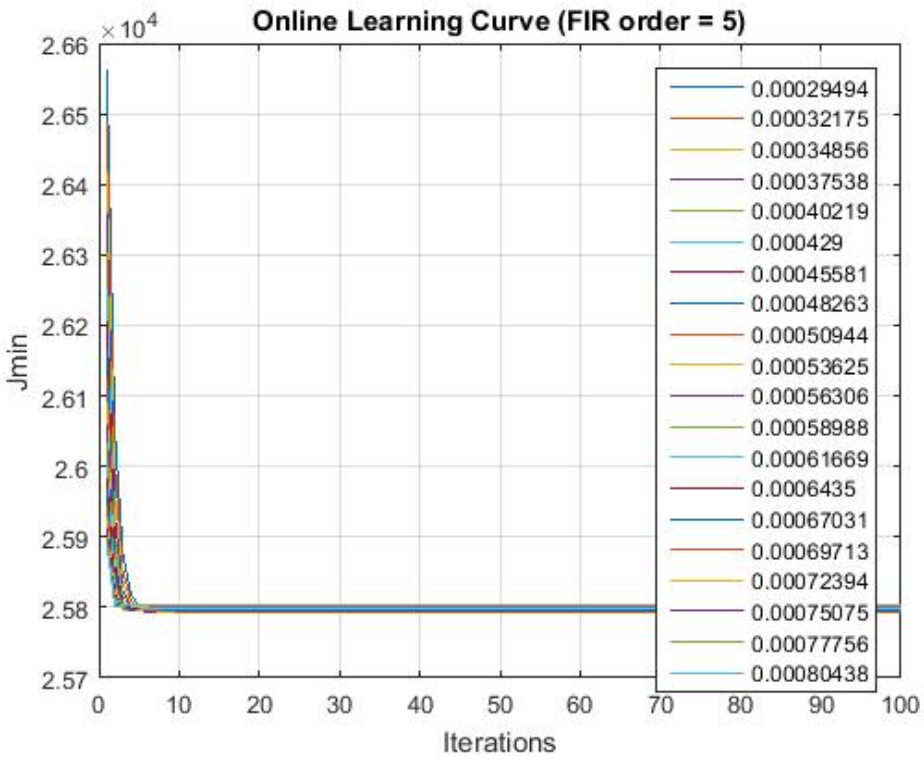
The time constant of adaptation (tau) is calculated using minimum eigenvalues of the input correlation matrix using the following formula for each eta sample:

```
timeconstantofadaptation_5(i) = (1/(eta_val_online5(i)*min(e)));
```

The misadjustment (tau) is calculated using the trace s of the input correlation matrix using the following formula for each eta sample:

```
misadjustment_5(i) = eta_val_online5(i)*trace;
```

Correction : The below graph is the learning curve of online LMS for filter order 5 and displays the variation of Jmin for the 100 iterations for each of the 20 eta samples. The run was given for 30 eta samples but the last 20 samples are only documented in the graph for clarity. The legend in the graph indicates which plot of Jmin corresponds to which eta value. The values quoted below the graph actually are the values for lowest obtained Jmin and their corresponding eta value and other parameters.



The speed of convergence we defined as number of iterations/ time constant of adaptation. Here the iterations = 100

The values generated for Eta (the free parameter) = 2.9494e-04

Time constant of adaptation= 1.8182

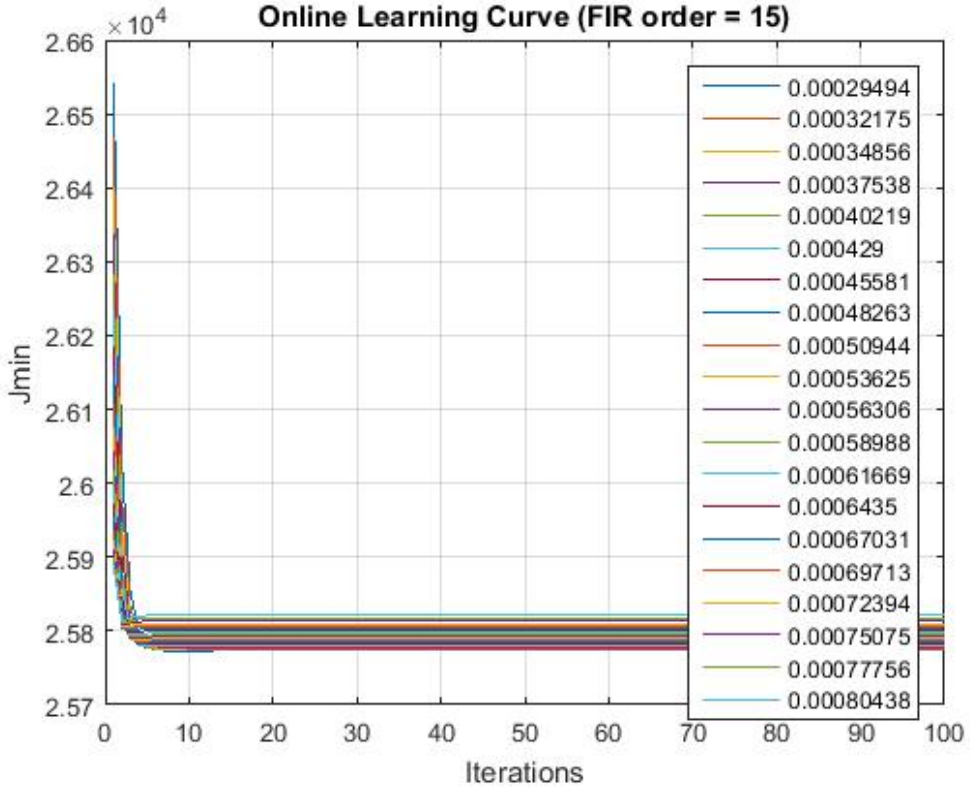
Speed of convergence = 55

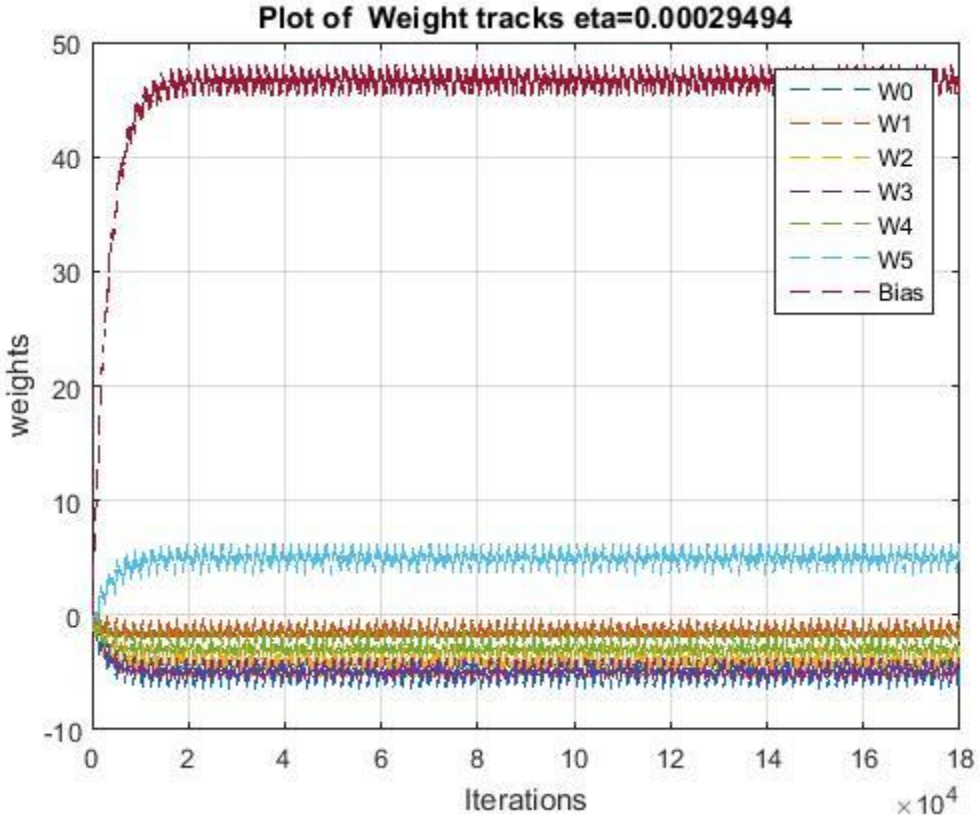
Misadjustment = 0.5500 equivalent to 55 %

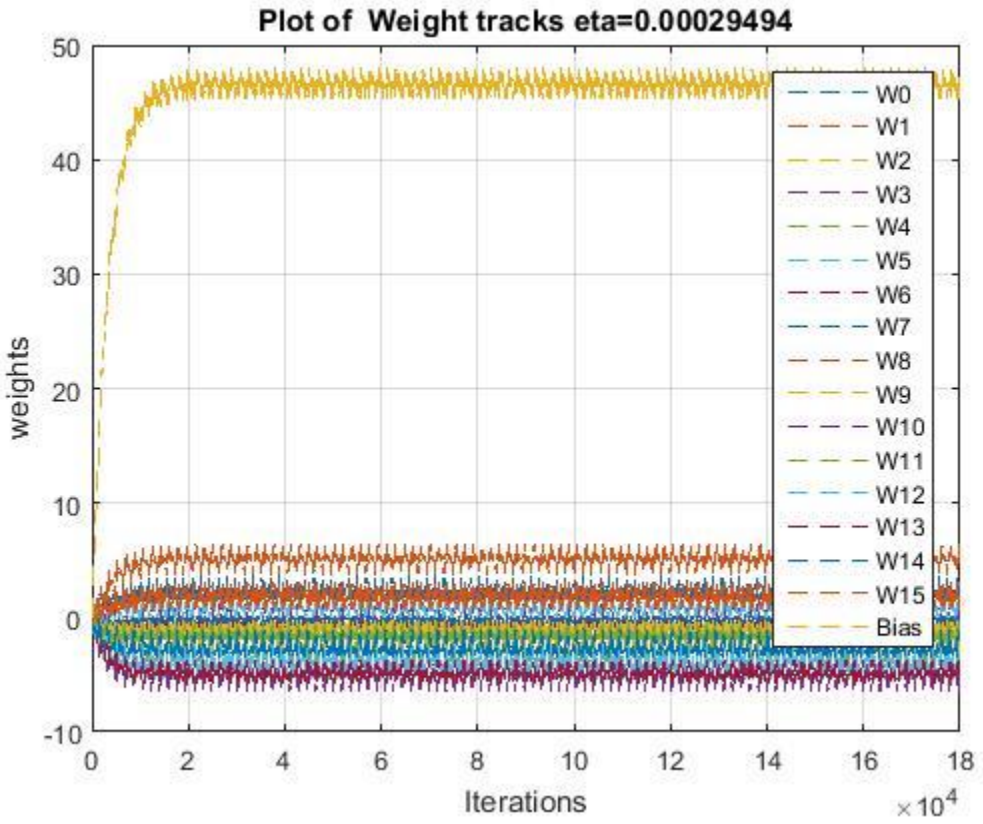
Maximum convergence time 55% misadjustment means a training duration in iterations of 55 times the number of inputs.

For FIR Filter Order 5	**FIR Filter order 15**
Jmin_5 (MSE Train) = 2.5794e+04	Jmin_15 (MSE Train) = 2.5774e+04
Eta (the free parameter) = 2.9494e-04	Eta (the free parameter) =2.9494e-04
MSE_(Test Data) = 2.0803e+04	MSE_(Test Data) = 2.0700e+04

Correction : The graph below is the learning curve of online LMS for filter order 15 and displays the variation of Jmin for the 100 iterations for each of the 20 eta samples. The run was given for 30 eta samples but the last 20 samples are only documented in the graph for clarity. The legend in the graph indicates which plot of Jmin corresponds to which eta value. The filter order 15 displays some diverging Jmin plot but there are displayed to convey the information of faster tendency of divergence for the higher order filter.The values quoted below actually are the values for lowest obtained Jmin and their corresponding eta value and other parameters.


Online Learning Curve (FIR order = 15)
×10^4
Jmin
Iterations
0.00029494
0.00032175
0.00034856
0.00037538
0.00040219
0.000429
0.00045581
0.00048263
0.00050944
0.00053625
0.00056306
0.00058988
0.00061669
0.0006435
0.00067031
0.00069713
0.00072394
0.00075075
0.00077756
0.00080438


Plot of Weight tracks eta=0.00029494
weights
Iterations
×10^4
W0
W1
W2
W3
W4
W5
Bias



Correction : The weight tracks are observed to oscillate more for FIR Filter order 15 compared to 5 which is expected as the weights are modified more often in a higher order filter. The number of weight tracks in the graph is actually the order of filter +2 because order of filter 5 means the time lags ranging from 0 to 5 so six values with additional value of a bias in the plot. The bias is tend to stabilize at a much higher value than all the weights in both the plots. It is also observed the values at which the weights of both the order of filter converges lie in a similar range.

It is also observed that higher order filter LMS tends to converge faster than others and have a tendency to deviate petty faster compared to a lower order filter.

Batch LMS: with batch 100 Filter order 15. The gradient of J is preserved for a batch is averaged out and the weights are modified using that gradient.

```
for iter = 1 : max_iter
    for i = order :batch: N+order-1
        for j = i:batch+i-1
        u = x(i:-1:i-order+1 , :);
        e = y(i:-1:i-order+1 , :) - u * w';
        gradJ(:,j) = u' * e;
        end
        Javg = sum(gradJ')/length(gradJ);
        w = w  + (lambda* Javg);

        k = k + 1;
        w_tracks(k,:) = w;
    end
```

```
    [J(iter), ~] = MeanSquareError(w', x, y);
end
```

Correction : Batch Learning we ran for the filter order of 5 because 15 was not converging after a lot of trail runs we also has to run for 250 iterations and also have to reduce the batch size to 5 to prevent the Jmin from diverging. The below plot indicates the learning curve for the batch training for 20 Jmin which we observed to converge after multiple iterations of the algorithm.

Jmin_B = 2.6057e+04

Eta_val = 1.0725e-04

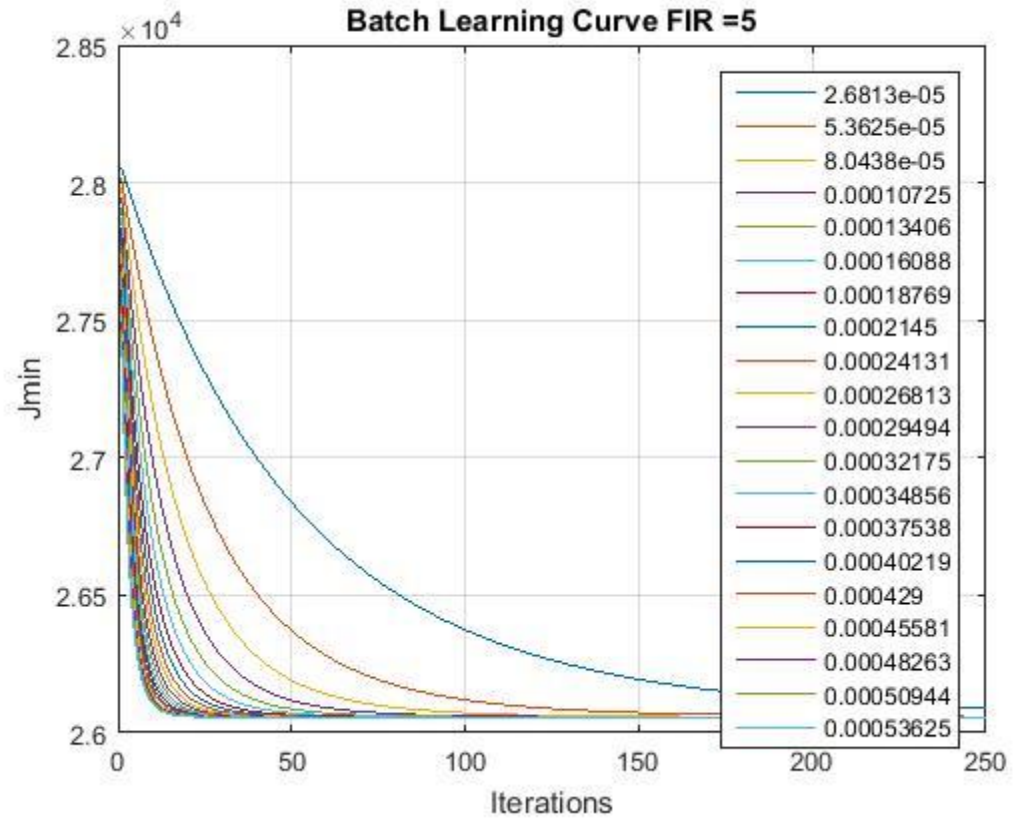
Timeconstantofadaptation = 5

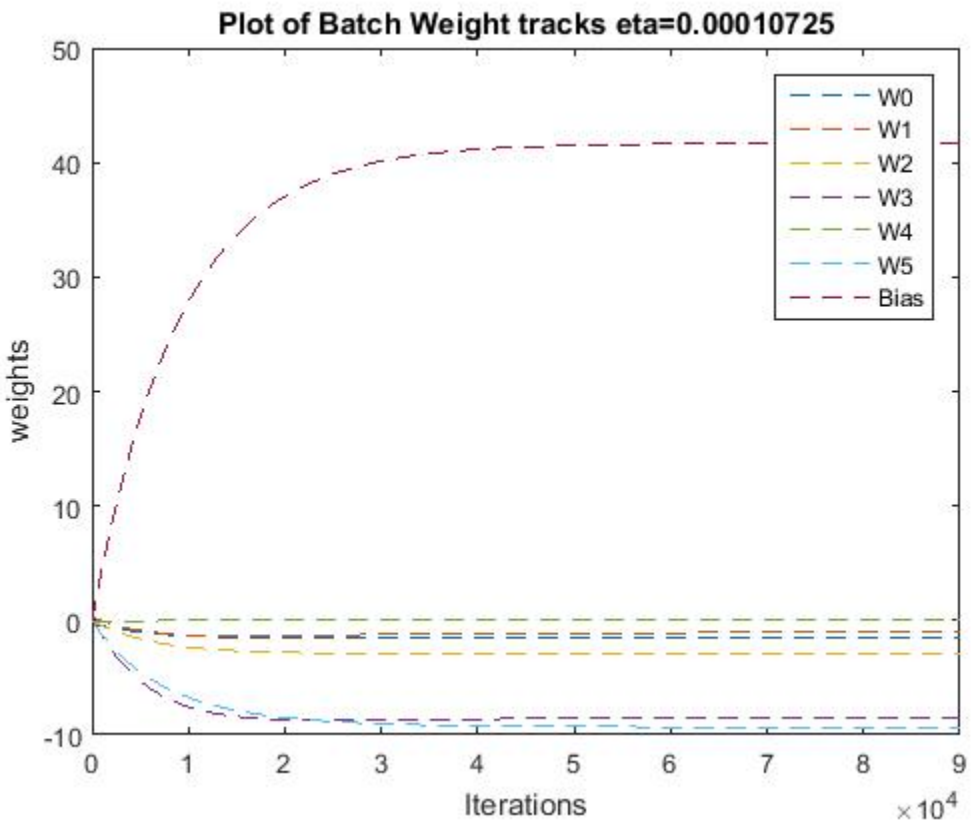
speedofadaptation_batch = 50

Misadjustment = 0.2000 equivalent to 20 %

Maximum convergence time 20% misadjustment means a training duration in iterations of 55 times the number of inputs.

When we test it on the testing data , we observer MSE(= 2.1021e+04) which is slightly lower than online learning for Filter order 5 and 15 Though Jmin (or MSE) for training dataset for a lower eta value is larget than online training cases, but seems like trained model adheres slightly well with testing dataset.





The weights tracks are plotted for the eta value (=1.0725e-04) having the lowest MSE for the training dataset. The weight tracks don't have any oscillation and are much smoother to converge towards a stable value. The weights also converge to different values for the batch training, though having a similar range.

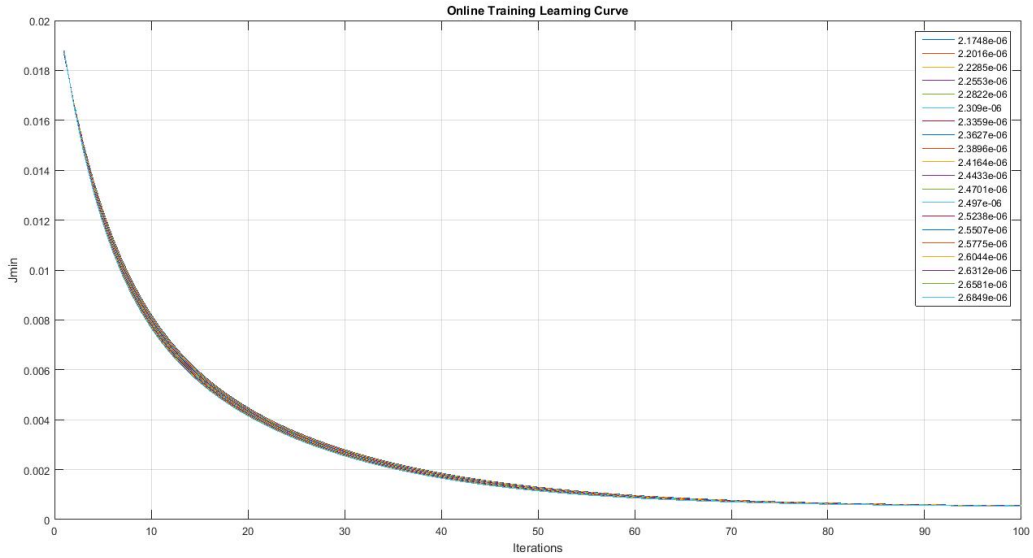
Conclusion: The batch learning is found to be a lower MSE on the test dataset compared to online learning. It is observed the weight tracks also oscillate less in batch training compared to online LMS even less compared to a lower order filter. The batch training could tend to be used to reduce computation and faster processing due to less instances of updating weights, but when it comes to accuracy online training is better. It could get tricky

Problem II

For the obesity dataset in the website, fit the best hyperplane through the data using the optimal solution (least squares) and the iterative solution (LMS). Compare the performance of both and estimate the correlation coefficient. Then test your solution in the unused data (test set) and compare the performance with that of the training set.

Ans: eta0 = assumed is 0.5 again to maintain fast convergence (LMS epochs = 100)

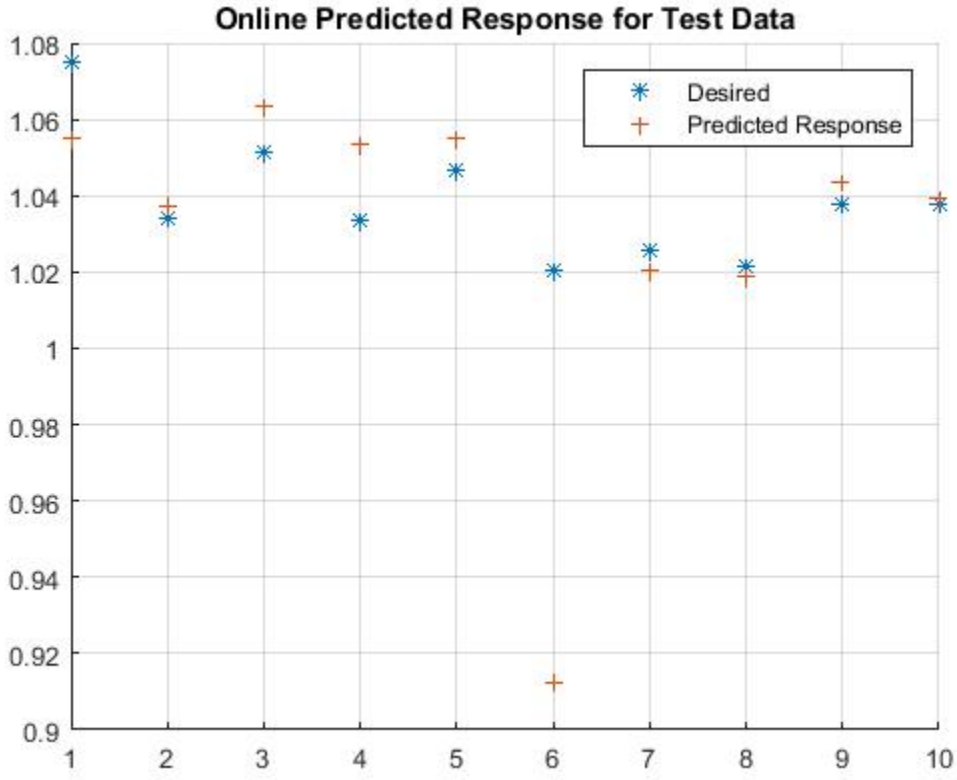
Online Learning Curve for last 20 eta iterations:



Correction: The above figure describes the convergence of the learning curve (Jmin) for online training which is seen to converge after 85 epochs for the online training. In the curve we plot only 20 learning curves with the least Jmin, and the precision for Jmin was kept low. Jmin minimizes 5.3445e-04 for an eta value = 2.6849e-06. The starting eta value is eta0(=0.05)/trace(R) which varies eta*i ; where i is the number of eta iterations.

Coefficient of correlation = { 0.8813, 0.8635, 0.8702, 0.8837, 0.8971, 0.9089, 0.9190, 0.9274, 0.9346, 0.9407, 0.9459, 0.9504, 0.9542, 0.9575, 0.9603, 0.9626, 0.9646, 0.9663, 0.9677, 0.9689} We also observe a positive coefficient of correlation in case of online training which input x varies in same direction of x. It is high even hits as high 0.96 for some cases so respective inputs are petty correlated

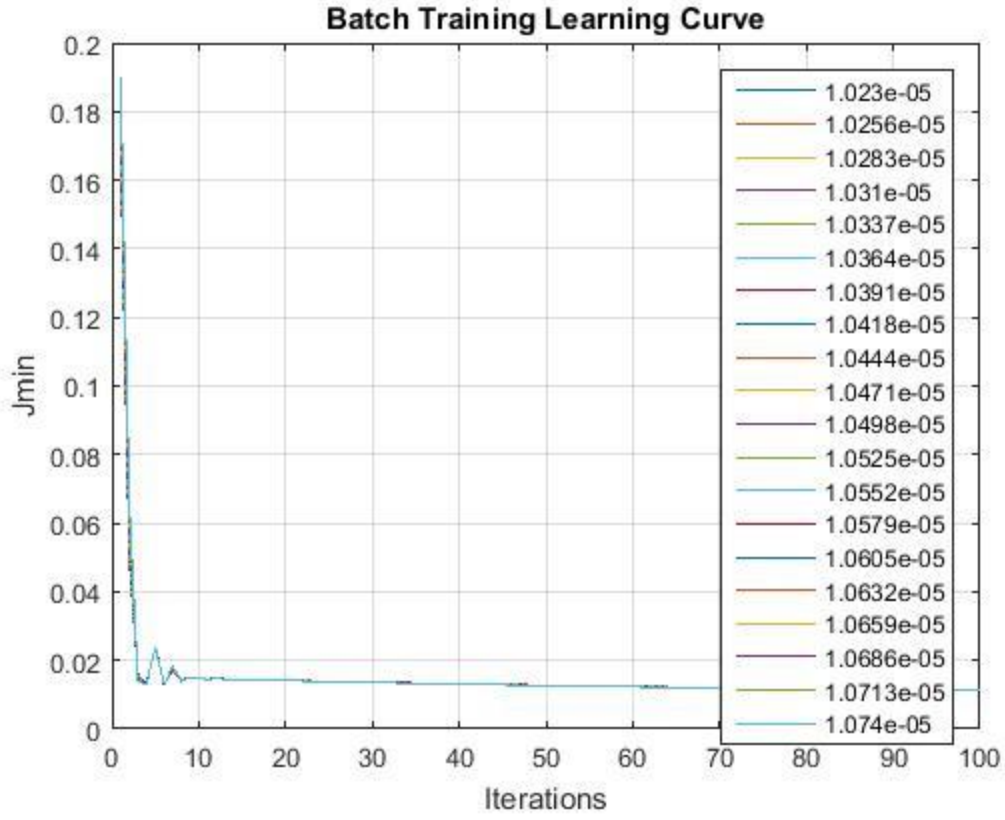
desired output.

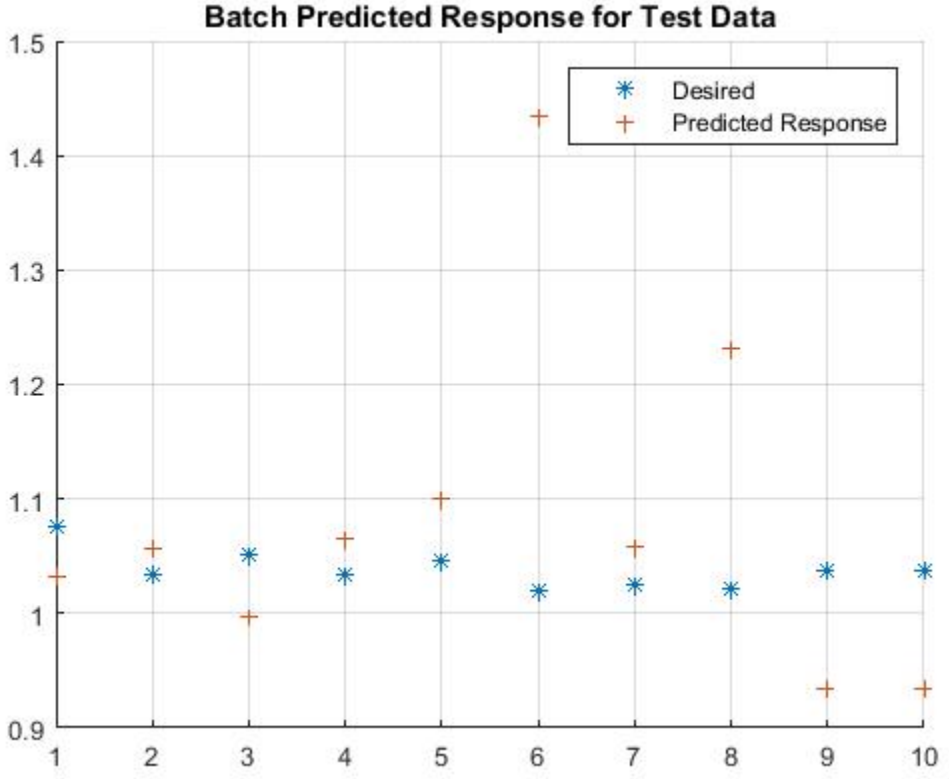


Correction: The above graph shows the predicted responses for the test data of 10 samples from the model trained from the train data. It could be showed clearly that predicted pattern fits petty accurately with the test data. This accuracy is also evident for the MSE of the test Dataset for the above eta= 0.0013

Batch Training Description:

Correction: The below figure describes the convergence of the learning curve (Jmin) for batch training which is seen to converge after 10 epochs with oscillation for the batch training. In the curve we plot only 20 learning curves with the least Jmin, and the precision for Jmin was kept low. Jmin minimizes 0.0112 for an eta value = 1.0740e-05. The eta iterations ran were 400 as we couldn't see Jmin reaching the desired minimum, for the above plot contains the last 20 eta iterations. The Batch training takes a large number of iterations to get a minimal Jmin converging value. It is also observed to oscillate while reaching the stable Jmin for the above output which clearly depicts a less scope for convergence or being stuck in the local minimum. The starting eta value is eta0(=0.05)/trace(R) which varies eta*i ; where i is the number of eta iterations


Batch Training Learning Curve
Jmin
Iterations
1.023e-05
1.0256e-05
1.0283e-05
1.031e-05
1.0337e-05
1.0364e-05
1.0391e-05
1.0418e-05
1.0444e-05
1.0471e-05
1.0498e-05
1.0525e-05
1.0552e-05
1.0579e-05
1.0605e-05
1.0632e-05
1.0659e-05
1.0686e-05
1.0713e-05
1.074e-05


Batch Predicted Response for Test Data
Desired
Predicted Response

Correction : It is to be noted the MSE(0.0248)error for test data in batch training is not too goof. We could also the predicted response for the batch training is smoother and does not get close to drastic changes. Instead maintains a response closer to the average response. worse than MSE (0.0126) error for online training set which is expected the weights are not modified in each iteration.

Coefficient of correlation = { 0.6791, 0.9157, 0.9230, 0.9235, 0.9238, 0.9241, 0.9243, 0.9246, 0.9249, 0.9251, 0.9254, 0.9257, 0.9260, 0.9262, 0.9265, 0.9268, 0.9270, 0.9273, 0.9276, 0.9278}

We also observe a postive coefficient of correlation in case of batch training which input x varies in same direction of x. It is high (0.92 approx) so respective inputs are petty correlated desired output.

Conclusion:

It could be clearly concluded that batch training is worse than online in terms of speed of convergence as it takes much more eta iterations than online. It also converges to higher Jmin for the training dataset. Thus clearly observe a worse MSE for the same training dataset than online training. So the batch requires more iteration, and may not be accurate enough but computation required per iterations by online so it is slightly slow over the course of same iterations.