Underwater Channel Modeling

Chinmay Sahu

Communications, Signal Processing, Networking Lab
Department of Electrical and Computer Engineering
Clarkson Univeristy,Potsdam,13699

May 3, 2019

1 Introduction

Underwater communications are prone to multiple forward scattering events, which create temporal dispersion and inter-symbol interference (ISI). These ISI channels can be modeled as multi-tap FIR filters. The effect of these channels can be mitigated with the help of channel equalization filters. A block diagram representation of the underwater communication system, along with a channel equalizer at the receiver, is shown in Figure 1.

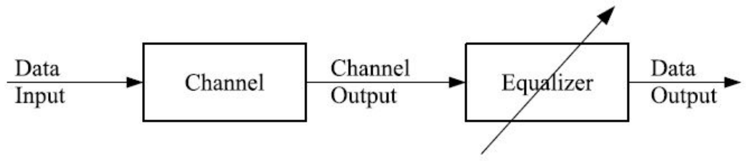


Figure 1: Underwater Digital Communications System. Modulated data is transmitted over a channel. The effects of the channel can be reversed using an equalizer.

1.1 Experimental Setup

Two sets of experiments were conducted. In the first set of experiments, a high quality, low noise, large bandwidth photomultiplier was used in order to obtain a "noise-free" receive sequence. In the second set, an inexpensive photomultiplier was used to model in order to determine feasibility of low-cost infrastructure for underwater digital communications. The experiments were conducted for 10 different turbidity levels. The Channel characteristics can be assumed to not have changed between two set of recordings. Data from high quality recordings were stored in "data1.mat" and data from second setup are in "data2.mat".

1.2 Need for Channel Modeling

To verify the quality of transmission, we estimate the bit error rate for both data sets. Bit error rate (BER) is a measure of the number of bit errors that occur in a given number of bit transmissions and it is usually expressed as a ratio. BER also quantifies the quality of the transmitting device, the receiver, the transmission path and its environment as it takes into consideration factors such as noise, jitter, attenuation, fading, and any error detection and correction schemes used in the interface standard.

The BER for both data sets are estimated and plotted in the Figure 2. The BER of high quality recordings are represented by "Data-1". The BER of low cost device recordings are represented by "Data-2". It can be observed that for high quality photo multiplier recordings represented by Data-1, the BER is as low as zero till channel 5. From channel 6, BER is 0.25. It means 25% of time the received bits are incorrect. For channel 7, it increases to 0.35 and it is close to 0.5 for channel-10. Similarly, for low quality recordings, represented by Data-2, BER remains as low as 0.05 till channel-5. However, it starts to get worse from channel 6 to channel 10. As the recordings are not reliable as high quality recordings, BER is also worse for Data-2 compared to Data-1.

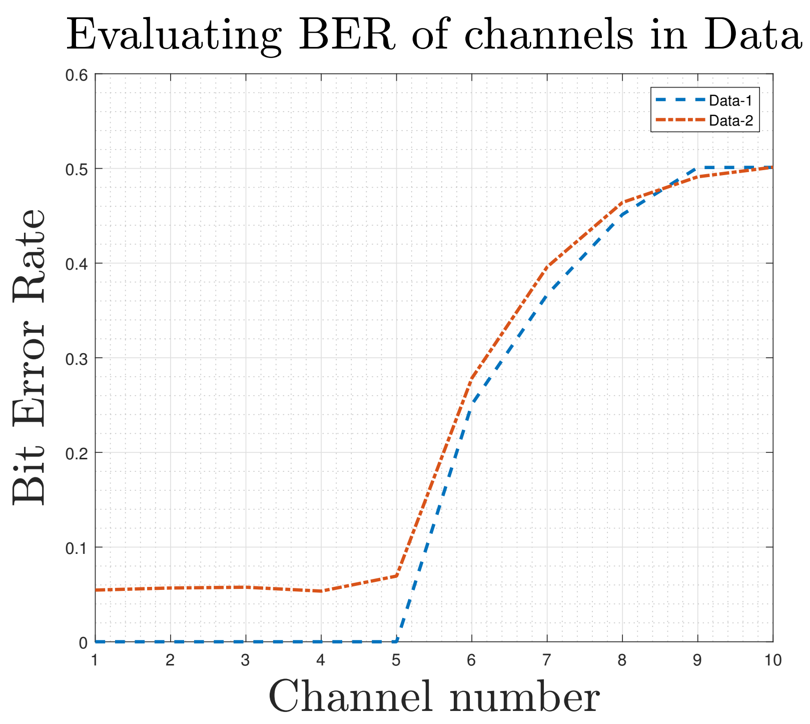
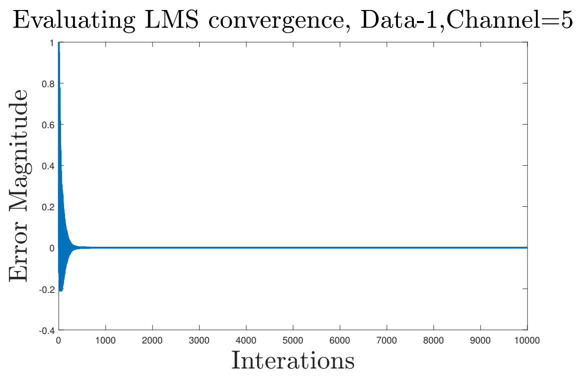


Figure 2: Bit error rates are calculated for all the channels. Higher channel number indicates the higher turbidity in the water. Hence the received message bits at the output of channel will be having higher faulty bits as the channel number increases. We can see that as the channel number increases, BER increases.

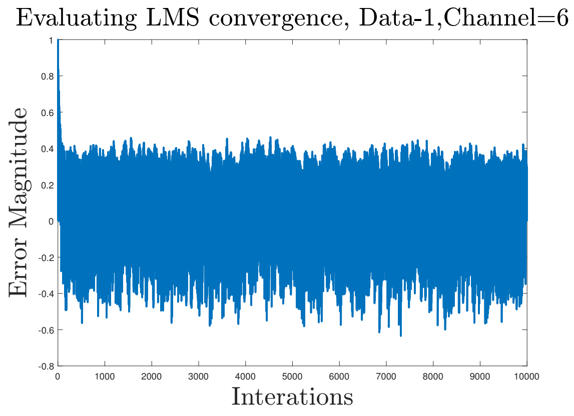
1.3 Channel Estimation and Error Convergence

To evaluate the communication fidelity of channels, LMS filters are deigned for different channels of Data-1. The error convergence for different channels are shown in Figure 3. To limit plots, convergence for channel 5,6,8, and 10 are shown. Up to channel-5, the filter coefficients can be estimated accurately. This means as the turbidity increases, both Data-1 and 2 fidelity is good till channel-5. After channel-5, there are error in filter coefficient estimates, which can be proved by poor error convergence from channel-6 to 10 shown in Figure 3. The LMS error convergence is tested for filter order 2 and step size of 0.01. The error convergence is also tested for higher filter order and varied step sizes. Still, the error did not improve for Channel-6 to 10.

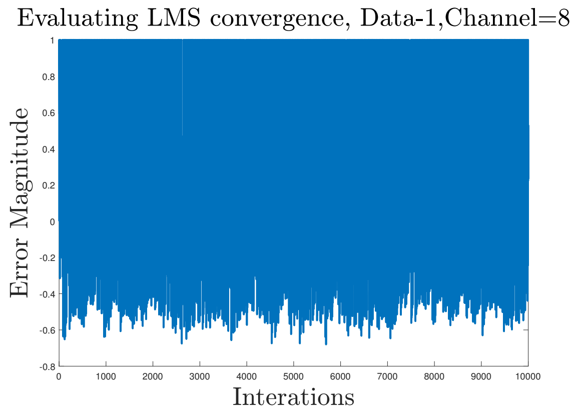
Hence, it is possible to estimate the channels accurately for channel-1 to 5 for Data-1. As underwater turbidity is too high, it is difficult to model channel 6-10. Equalizer is designed by inverting the channel. So having a good estimate of channel filter coefficients indicates a more accurate modeling and representation of equalizers. In this case, we can infer that it is to possible to model equalizers with low BER for channel 1-5. However, for channel 6-10, BER will be poor and the equalizers may not work as expected.



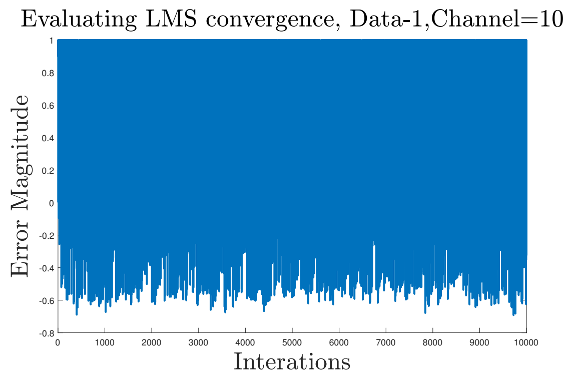
a. Filter Order=2



b. Filter Order=2



c. Filter Order=2



d. Filter Order=2

Figure 3: Error convergence for LMS channel estimation. Each sub figure shows convergence of error for a particular channel as number of iteration increases. Error is converging for channel-5. However, it starts to deteriorate from channel 6 on wards.

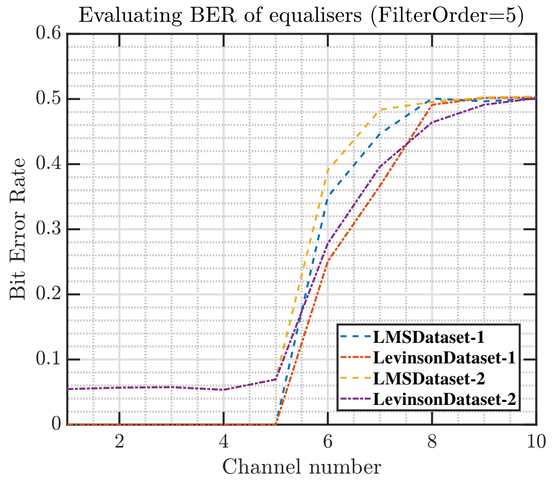
2 Equalizers

Equalization is process to reverse the distortion incurred by a signal while going through a channel. In simple words, equalizer is the inverted model of the channel, so we can receive the message as it is transmitted without any distortion.

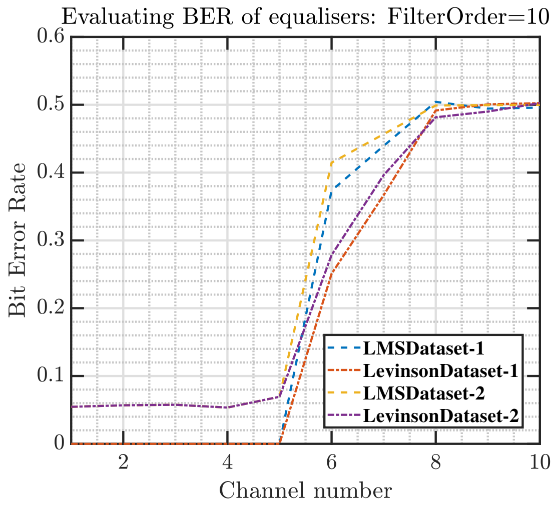
2.1 LMS and Levinson-Durbin (LD) Equalizer

LMS, Levinson-Durbin (LD) [1] equalizers are designed. The BERs for different channels are estimated and plotted as shown in Figure 4. Bit error rates are plotted for both the data-sets for all the channels. This is carried out for filter order 5, 10, 50, and 100. We can see for Data-1, BER for channel-1 to channel-5 is as low as zero for all filter order. For same channels for Data-2, BER is around 0.05, which is not as good as Data-1. This is anticipated as the fidelity of recordings are not as good as Data-1. Hence, the equalizers for Data-2 will have poor BER compared to Data-1. For channel 6 and on wards equalizers BER start increasing from 0.35 to 0.5. Higher BER indicates poor performance. BER of 0.5

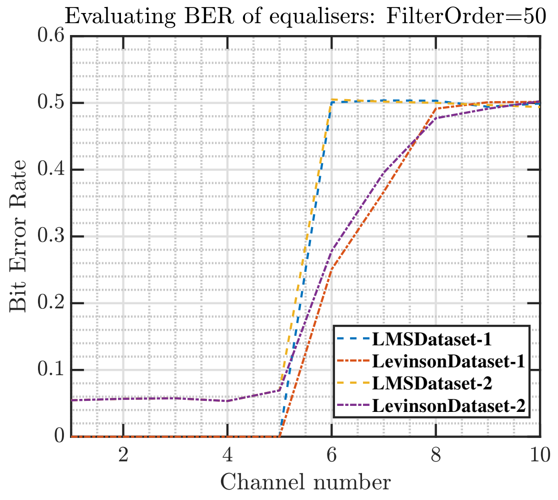
indicates that equalizer is performing poorly all the time. From BERs we can infer that the equalization does not work for channels 7-10 for both Data-1 and Data-2. Also, increasing the filter order seems to not have any impact on improving BER for both Data-1 and Data-2.



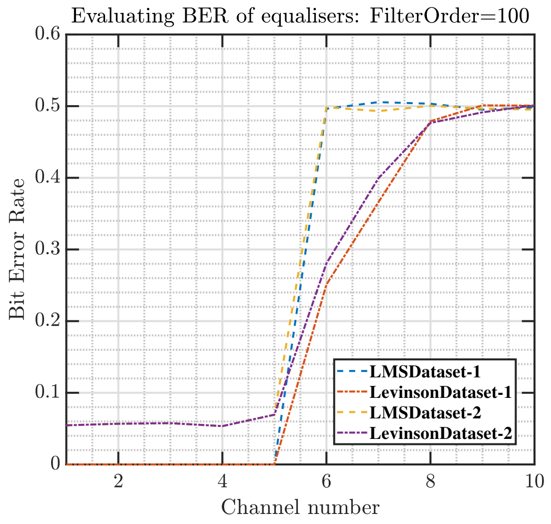
a. Filter Order=5



b. Filter Order=10



c. Filter Order=50



d. Filter Order=100

Figure 4: Performance comparisons of equalizers. The BERs are plotted for LMS and

2.2 Neural Network based Equalizer Design

A neural network based equalizer is designed to test the performance of the equalizers. The neural network model is trained with 10 layers and 10 neurons in each layer. The network training function used the model updates weight and bias values according to the BFGS quasi-Newton method. The bit error rate plot of both data-sets are shown in Figure 5. We can see that for Data-1, equalizers work perfectly for channel 1-5. As the channel estimation is also perfect for channel 1-5, the perfect equalizers can also be designed for channel 1-5.

However, as Data-2 represents low cost photo multiplier recordings, the equalizer designed for them are having BER of 0.12, which is not as good as LMS or LD equalizer. Hence, LMS or LD equalizer seems to be better choice for modeling channel 1-5 for Data-2. BER is really poor for channel 7, 8, 9, and 10. It means all the time equalizer makes a mistake in prediction of correct bit. Given the computation and accuracy, LMS and LD works better for Channel 1-5 design for both the data-sets.

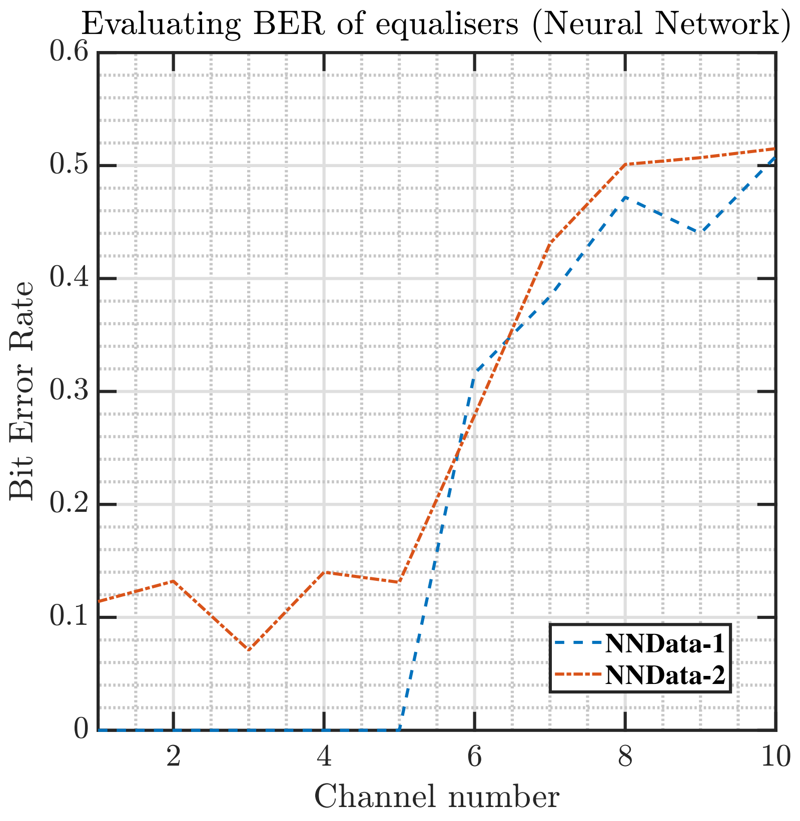


Figure 5: BERs of the Filters after Neural networks modeling

3 Conclusions

Two sets experiments were carried out under water to record the data in "Data-1" and "Data-2". Corresponding channels were modeled using LMS [1]. Then LMS and LD equalizers are designed for both data sets. The bit error rate is considered to check the performance of the equalizers. Then neural network is used to model the channel equalizers. In all the cases, it is observed that the equalizers can be designed for channel 1-5 for both data sets. However, from channel 6-10, the turbidity is so high that it is not possible to estimate the filter coefficients for the equalizer . Hence, the equalizers perform very poorly for those channels. This indicates a threshold for turbidity levels, beyond which photo multiplier recordings are not useful to model channel and it's equalizers.

References

[1] S. S. Haykin, *Adaptive filter theory*. Pearson Education India, 2005.

Appendix

Matlab Code for priliminary BER check for data fidelity

```matlab
% Chinmay Sahu
% sahuc@clarkson.edu
% Last Updated: 04/23/19
% LMS Channel Adaptation
load('data1');
load('data2');

BerEst2=zeros(10,1);
% Estimating current channel quality
for channelNumber=1:10
    dataIn =x2_all(channelNumber,:)'>0.5; % input channel data of nth ...
        channel
    dataOut= y2_all(channelNumber,:)'>0.5; % convolve channel with the ...
        input
    BerEst2(channelNumber)=biterr(dataIn,dataOut)/length(dataIn);
end

figure
grid on
box on
grid minor
hold on
semilogy(1:10,BerEst1,'--','LineWidth',2);
hold on
semilogy(1:10,BerEst2,'-.','LineWidth',2);
% pbaspect([1 1 1])
title('Evaluating BER of channels in Data','FontName','Times New ...
    Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
xlabel('Channel number','FontName','Times New ...
    Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
ylabel('Bit Error Rate','FontName','Times New ...
    Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
% set(gca,'fontsize',20,'ticklength',[0.025 0.05])
set(gca,'FontName','Times New ...
    Roman','FontSize',20,'FontWeight','bold','linewidth',2,'ticklength',[0.025 ...
    0.05],'TickLabelInterpreter', 'latex');
legend( 'Data-1','Data-2')
xlim([1 10])
```

Matlab Code For LMS and Levinson-Durbin Equalizers

```
%% LMS & LevinDurbin Equalizer Design
clc
load('data1');
load('data2');
LmsBerEst1=zeros(10,1);
LevBerEst1=zeros(10,1);
LmsBerEst2=zeros(10,1);
LevBerEst2=zeros(10,1);
filter_order=100;
b1=zeros(filter_order,1);
b2=zeros(filter_order,1);
% LMS Adaptation
for channelNumber = 1:10
    x1 =x_all(channelNumber,:)'>0.5; % input channel data of nth channel
    d1 = y_all(channelNumber,:)'>0.5; % convolve channel with the input
    x2 =x2_all(channelNumber,:)'>0.5; % input channel data of nth channel
    d2 = y2_all(channelNumber,:)'>0.5; % convolve channel with the input

    lms=dsp.LMSFilter(filter_order,'StepSize',0.01,'WeightsOutputPort',true);
    [¬,e,w1]=lms(double(x1),double(d1)); % LMS filter weights
    [¬,e,w2]=lms(double(x2),double(d2)); % LMS filter weights

    lms_filter1=w1./w1(1); % estimater equalizer coefficients
    lms_filter2=w1./w1(1); % estimater equalizer coefficients

    dataOut1=(filter(1,lms_filter1,d1))>0.5; % information bits pass ...
        through equalizer
    dataOut2=(filter(1,lms_filter2,d2))>0.5; % information bits pass ...
        through equalizer
    LmsBerEst1(channelNumber)=biterr(x1,dataOut1)/length(x1); % BER of ...
        channel
    LmsBerEst2(channelNumber)=biterr(x2,dataOut2)/length(x2); % BER of ...
        channel

    %%%LevinsonDurbin
    [a1 ¬]=levinson(autocorr(d1,filter_order),filter_order−1);
    [a2 ¬]=levinson(autocorr(d1,filter_order),filter_order−1);
    b1=a1;
    b2=a2;
    dataOut1=(filter(b1,1,d1))>0.5; % information bits pass through ...
        equalizer
    dataOut2=(filter(b2,1,d2))>0.5; % information bits pass through ...
        equalizer
    LevBerEst1(channelNumber)=biterr(x1,dataOut1)/length(x1); % BER of ...
        channel
    LevBerEst2(channelNumber)=biterr(x2,dataOut2)/length(x2); % BER of ...
        channel
end
% Plot results
figure
```

```
grid on
box on
grid minor
hold on
semilogy(1:10,LmsBerEst1,'--','LineWidth',2);
hold on
semilogy(1:10,LevBerEst1,'-.','LineWidth',2);
hold on
semilogy(1:10,LmsBerEst2,'--','LineWidth',2);
hold on
semilogy(1:10,LevBerEst2,'-.','LineWidth',2);
hold on
% semilogy(1:10,NNBerEst1,'--','LineWidth',2);
% hold on
% semilogy(1:10,NNBerEst2,'-.','LineWidth',2);
% pbaspect([1 1 1])
title(['Evaluating BER of equalisers: ...
    FilterOrder=',num2str(filter_order)],'FontName','Times New ...
    Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
xlabel('Channel number','FontName','Times New ...
    Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
ylabel('Bit Error Rate','FontName','Times New ...
    Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
% set(gca,'fontsize',20,'ticklength',[0.025 0.05])
set(gca,'FontName','Times New ...
    Roman','FontSize',20,'FontWeight','bold','linewidth',2,'ticklength',[0.025 ...
    0.05],'TickLabelInterpreter', 'latex');
legend( ...
    'LMSDataset-1','LevinsonDataset-1','LMSDataset-2','LevinsonDataset-2')
xlim([1 10])
```

Matlab Code For Neural Network based equalizer

```
%% Neural Network based modeling
load data2.mat
%train network

number_of_layers=10;
neurons_per_layer=10;
training_fn='trainbfg';
NNBerEst2=zeros(10,1);

for channel_num=1:10

    number_of_layers=10;
    neurons_per_layer=10;
    training_fn='trainbfg';

    train_data_input=y2_all(channel_num,1:9000)>0.5;
    train_data_output=x2_all(channel_num,1:9000)>0.5;


    hiddensize=neurons_per_layer*ones(1,number_of_layers);
    net=feedforwardnet(hiddensize,training_fn);
    % view(net);
    net=configure(net,train_data_input,train_data_output);
    net=train(net,train_data_input,train_data_output);

    %%Test network
    test_data_input=y2_all(channel_num,9001:10000);
    test_data_output=sim(net,test_data_input);

    x_out=test_data_output>0.5;
    x_ver=x2_all(channel_num,9001:10000)>0.5;

    NNBerEst2(channel_num)=sum(x_out≠x_ver)/1000;
end

figure
grid on
box on
grid minor
hold on
semilogy(1:10,NNBerEst1,'--','LineWidth',2);
hold on
semilogy(1:10,NNBerEst2,'-.','LineWidth',2);
pbaspect([1 1 1])
title('Evaluating BER of equalisers (Neural Network)','FontName','Times ...
    New Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
xlabel('Channel number','FontName','Times New ...
    Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
ylabel('Bit Error Rate','FontName','Times New ...
    Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
```

```
% set(gca,'fontsize',20,'ticklength',[0.025 0.05])
set(gca,'FontName','Times New ...
    Roman','FontSize',20,'FontWeight','bold','linewidth',2,'ticklength',[0.025 ...
    0.05],'TickLabelInterpreter', 'latex');
legend('NNData−1','NNData−2')
xlim([1 10])
```

Matlab Code For LMS error convergence

```
%% LMS channel estimation and it's BER
clc
load('data1');
load('data2');
LmsBerEst1=zeros(10,1);
LevBerEst1=zeros(10,1);
LmsBerEst2=zeros(10,1);
LevBerEst2=zeros(10,1);
filter_order=20;
b1=zeros(filter_order,1);
b2=zeros(filter_order,1);
% LMS Adaptation
for channelNumber = 5:10
    x1 =x_all(channelNumber,:)'>0.5; % input channel data of nth channel
    d1 = y_all(channelNumber,:)'>0.5; % convolve channel with the input
    x2 =x2_all(channelNumber,:)'>0.5; % input channel data of nth channel
    d2 = y2_all(channelNumber,:)'>0.5; % convolve channel with the input

    lms=dsp.LMSFilter(filter_order,'StepSize',0.05,'WeightsOutputPort',true);
    [¬,e,w1]=lms(double(x1),double(d1)); % LMS filter weights
    plot(e,'LineWidth',2);
    title(['Evaluating LMS convergence, ...
        Data−1,Channel=',num2str(channelNumber)],'FontName','Times New ...
        Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
    xlabel('Interations','FontName','Times New ...
        Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
    ylabel('Error Magnitude','FontName','Times New ...
        Roman','FontSize',28,'FontWeight','bold','interpreter','latex');

    [¬,e,w2]=lms(double(x2),double(d2)); % LMS filter weights

    dataOut1=(filter(w1,1,x1))>0.5; % information bits pass through ...
        equalizer
    dataOut2=(filter(w2,1,x2))>0.5; % information bits pass through ...
        equalizer
    LmsBerEst1(channelNumber)=biterr(x1,dataOut1)/length(x1); % BER of ...
        channel
    LmsBerEst2(channelNumber)=biterr(x2,dataOut2)/length(x2); % BER of ...
        channel

    %%%LevinsonDurbin
    [a1,e]=levinson(autocorr(d1,filter_order),filter_order−1);
    [a2 ¬]=levinson(autocorr(d1,filter_order),filter_order−1);
    b1=a1;
    b2=a2;
    dataOut1=(filter(b1,1,d1))>0.5; % information bits pass through ...
       equalizer
    dataOut2=(filter(b2,1,d2))>0.5; % information bits pass through ...
       equalizer
    LevBerEst1(channelNumber)=biterr(x1,dataOut1)/length(x1); % BER of ...
       channel
    LevBerEst2(channelNumber)=biterr(x2,dataOut2)/length(x2); % BER of ...
       channel
end
% Plot results
figure
grid on
box on
grid minor
hold on
semilogy(1:10,LmsBerEst1,'--','LineWidth',2);
hold on
semilogy(1:10,LevBerEst1,'-.','LineWidth',2);
hold on
semilogy(1:10,LmsBerEst2,'--','LineWidth',2);
hold on
semilogy(1:10,LevBerEst2,'-.','LineWidth',2);
hold on
% semilogy(1:10,NNBerEst1,'--','LineWidth',2);
% hold on
% semilogy(1:10,NNBerEst2,'-.','LineWidth',2);
% pbaspect([1 1 1])
title(['Evaluating BER of equalisers: ...
   FilterOrder=',num2str(filter_order)],'FontName','Times New ...
   Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
xlabel('Channel number','FontName','Times New ...
   Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
ylabel('Bit Error Rate','FontName','Times New ...
   Roman','FontSize',28,'FontWeight','bold','interpreter','latex');
% set(gca,'fontsize',20,'ticklength',[0.025 0.05])
set(gca,'FontName','Times New ...
   Roman','FontSize',20,'FontWeight','bold','linewidth',2,'ticklength',[0.025 ...
   0.05],'TickLabelInterpreter', 'latex');
legend( ...
   'LMSDataset-1','LevinsonDataset-1','LMSDataset-2','LevinsonDataset-2')
xlim([1 10])
```