MATLAB Simulation Exercise

Statistical Signal Processing II

LMS ALGORITHM FOR CHANNEL EQUALIZATIONS

Group 13

Athmajan Vivekananthan

University of Oulu
Finland

February 14, 2024

1 Task 1(a) 1(b)

The Matlab Code for this section has been listed in section 4. Observation has been made for this section by first changing μ while keeping T constant, then changing T while keeping μ constant. The performance of the LMS algorithm has been quantified by using Symbol Error Rate (SER) values for each of the scenarios. [1] [2]

1.1 Changing μ

Values for μ were changed in the range of 0.1 to 0.2. The resulting SER values for each of the step size values has been plotted in Figure 1. It is evident that there is an upper bound for the step size for the LSM algorithm to provide SER that is realistic. Somewhere closer to 0.189 and above the SER values are 1. This effect can be seen in the convergence of the error signal. As shown in Figure 2 for very small values of μ the convergence takes a long time but for very huge values of μ the convergence does not happen as seen in Figure 4. This effect can be explained using the convergence satisfying range for μ, which should be $0 < \mu < 2$. However, the upper bound does not even reach 2 in this case.

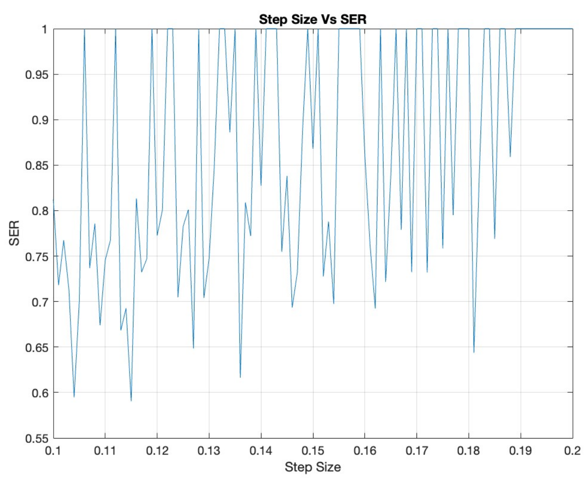

Figure 1: Step Size Vs SER values

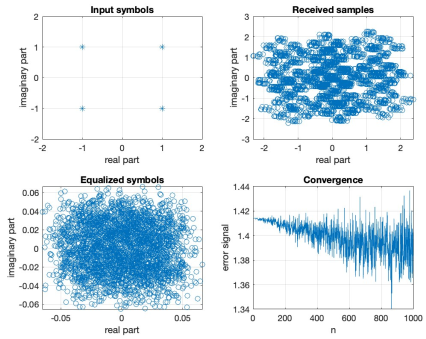

Figure 2: Symbol Output and Error Convergence for $L = 20$ and $\mu = 0.00001$

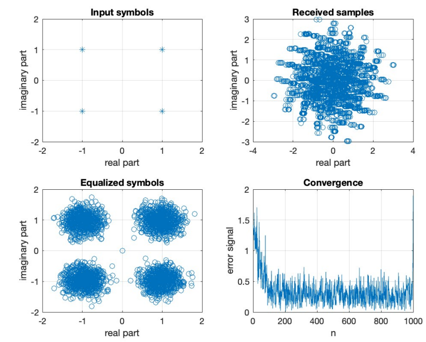

Figure 3: Symbol Output and Error Convergence for $L = 20$ and $\mu = 0.01$

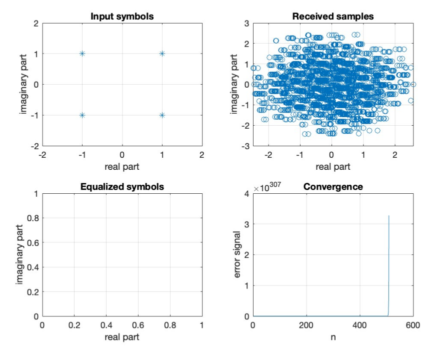

Figure 4: Symbol Output and Error Convergence for $L = 20$ and $\mu = 0.2$

1.2 Changing T

The performance of the LMS algorithm was testing for multiple T values against the SER values while keeping the step size fixed at $\mu = 0.01$. The results are presented in Table 1. It can be seen there is a sweet spot in terms of SER related to the number of known desired samples. When T is very low the model is under-fitted and produces high SER values. Then around $T = 1000$ the model seems to provide SER values which are preferable and when increasing to larger known samples the model is over-fitted producing in poor SERs.

Table 1: SER (Symbol Error Rate) for Different Values of T

T	SER
10	0.7473
100	0.7587
1000	0.7370
10^4	0.7423
10^5	0.7523
10^6	0.7590

2 Task 1(c)

Now the length of the equalizer is first reduced to 10 and then increased to 30 to study the effect on the performance of the LMS algorithm in terms of error convergence and SER values. As seen in Figure 5 it can be observed that realistic SER values are obtained for larger step sizes compared to the case of $L = 20$. But when looking at convergence Figures 7, 8, 9, is it evident that convergence is not that much impacted. However for longer equalizers ($L = 30$) the performance largely sensitive on the granularity of the step size as seen in Figure 6.

2.1 Changing μ

As seen on 2 there is a slight improvement in the SERs because of the complexity of the equalizers increasing with the number of taps. This potential better performance can be reasoned because with more taps the equalizer can capture more features of the channel and mitigate noise. However the realistic SER values and convergence range for μ is reduced with larger tap counts or equalizer lengths because the complex design of the equalizer needs granular step sizes.

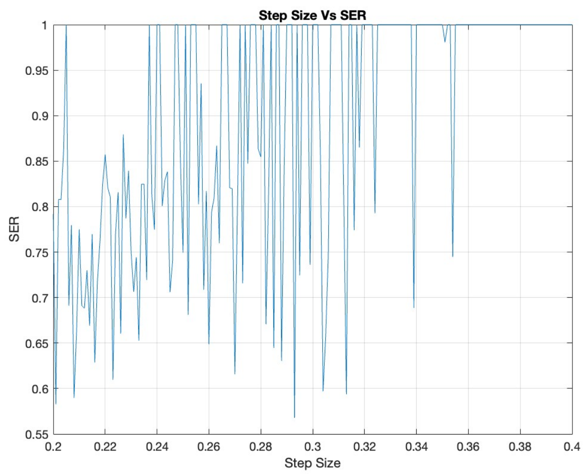

Figure 5: Step Size Vs SER values for $L = 10$

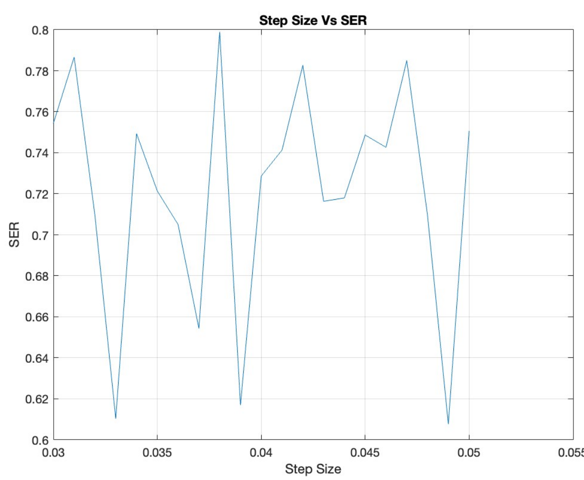

Figure 6: Step Size Vs SER values for $L = 30$

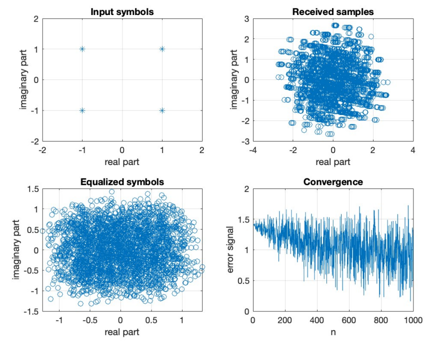

Figure 7: Symbol Output and Error Convergence for $L = 10$ and $\mu = 0.0001$

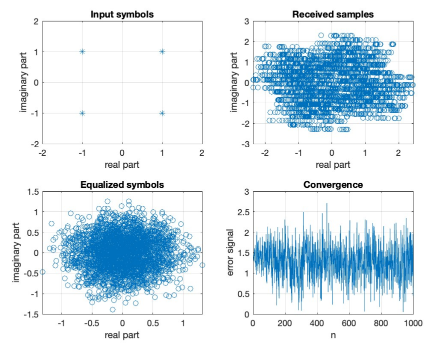

Figure 8: Symbol Output and Error Convergence for $L = 10$ and $\mu = 0.001$

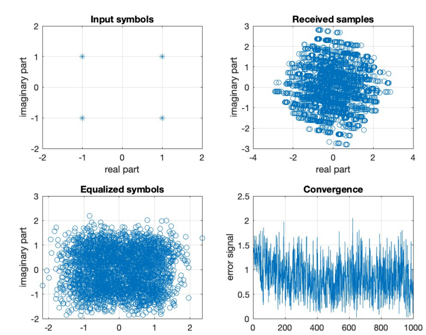

Figure 9: Symbol Output and Error Convergence for $L = 10$ and $\mu = 0.01$

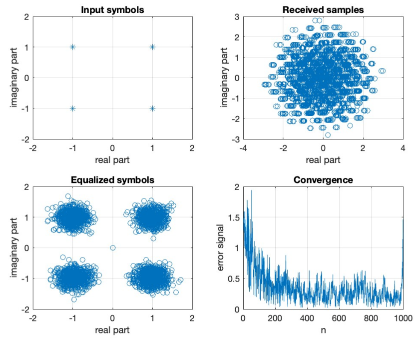

Figure 10: Symbol Output and Error Convergence for $L = 30$ and $\mu = 0.01$

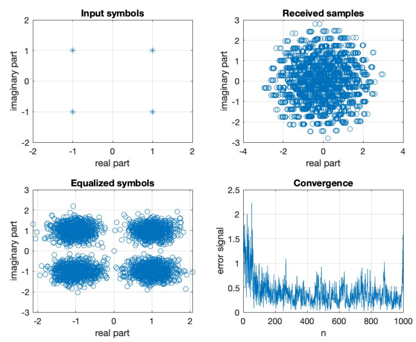

Figure 11: Symbol Output and Error Convergence for $L = 30$ and $\mu = 0.02$

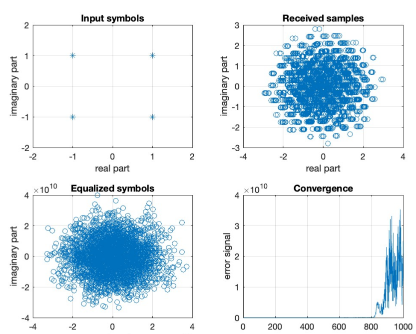

Figure 12: Symbol Output and Error Convergence for $L = 30$ and $\mu = 0.04$

Table 2: SER (Symbol Error Rate) for Different Values of μ and L

μ	$L = 10$	$L = 20$	$L = 30$
0.0001	0.7837	0.7583	0.7567
0.001	0.7950	0.7590	0.7570
0.01	0.7820	0.7587	0.7563
0.1	0.6347	0.6680	0.6953
0.2	0.7883	0.7833	1
0.3	0.7313	1	1
1	1	1	1
1.5	1	1	1
2	1	1	1
3	1	1	1

2.2 Changing T

Based on the observations presented in Table 3 it can be seen that when the complexity of the equalizer is increased by longer equalizer, having more known samples result in over-fitting of the model.

Table 3: SER (Symbol Error Rate) for Different Values of T and L

T	SER for $L = 20$	SER for $L = 10$	SER for $L = 30$
10	0.7473	0.8533	0.7380
100	0.7587	0.8430	0.8110
1000	0.7370	1	1
10^4	0.7423	1	1
10^5	0.7523	1	1
10^6	0.7590	1	1

3 Task 2

Table 4: NLMS - SER (Symbol Error Rate) for Different Values of μ and L

μ	$L = 10$	$L = 20$	$L = 30$
0.0001	0.7870	0.7577	0.7563
0.001	0.7897	0.7580	0.7563
0.01	0.7957	0.7600	0.7563
0.1	0.7890	0.7587	0.7563
0.2	0.7697	0.7587	0.7563
0.3	0.7653	0.7590	0.7563
1	0.6847	0.7693	0.7563
1.5	0.5727	0.7920	0.7530
2	0.6317	0.7970	0.7187
3	0.7553	0.6070	0.7720
6.8	1	0.6190	0.7533
8	1	1	0.7900
8.5	1	1	1

Table 5: NLMS - SER (Symbol Error Rate) for Different Values of T and L

T	SER for $L = 20$	SER for $L = 10$	SER for $L = 30$
10	0.7417	0.7663	0.7487
100	0.7397	0.7753	0.7460
1000	0.7600	0.7957	0.7597
10^4	0.7437	0.7177	0.7453
10^5	0.7560	0.7043	0.7523
10^6	0.7583	0.7927	0.7590

It can be seen when tap input vector is normalized in terms of Euclidean norm of the tap input vector, the realistic equalizer functions for a multitude of step sizes and also for larger number of known desired signals. This is due to the fact that this algorithm has been changed now to a time varying step size within the algorithm itself.

One of the major advantage is that normalized LMS exhibits a rate of convergence that is potentially faster than that of the standard LMS.

One of the major disadvantages of normalized LMS compared to the standard LMS is that numerical computational complexities may occur when the tap input vector is small. Specifically, due to the division by the power of the tap input vector when normalizing, it may cause exploding weights calculated for the step sizes.

This theory is evident as seen in Figure 14, which shows that the output is poor in Normalized LMS for small tap inputs compared to Standard LMS as shown in Figure 13.

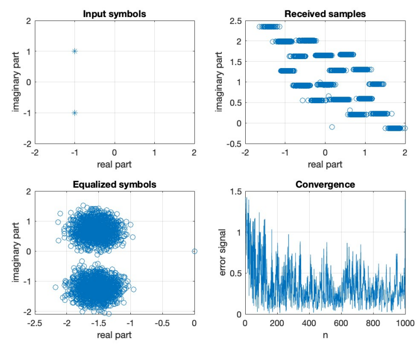

Figure 13: Standard LMS for small tap inputs

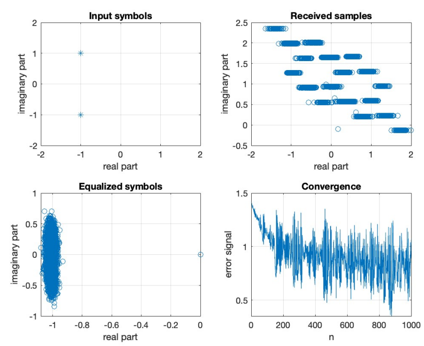

Figure 14: NLMS for small tap inputs

4 Matlab Code

```
    clear all;
close all;
clc;
rng(42);
T=1E3;    % number of training samples
N=3e3;    % number of data samples
SNRdB=25;     % SNR in dB value
L=20; % length for smoothing(L+1)
ChL=5;  % length of the channel(ChL+1)
mu = 0.01;  % Step size of the LMS algorithm
%SERlist = zeros(size(0.03:0.001:0.05));
ath = 1;
%for mu=0.03:0.001:0.05




%% Training phase
% QPSK symbols to transmit
d=round(rand(1,T))*2-1;
d=d+1i*(round(rand(1,T))*2-1);

% Channel
Ch=randn(1,ceil(ChL/2))+1i*randn(1,ceil(ChL/2));
Ch=[Ch Ch(end-1:-1:1)];
Ch=Ch/norm(Ch);


% signal filtered by channel
x= filter(Ch,1,d);


% Noise
v=randn(1,T);
v=v/norm(v)*10^(-SNRdB/20)*norm(x);

% Received signal
u=x+v;
```

```
%%%%%%%%%%%%%% WRITE LMS ALGORITHM FOR EQUALIZATION HERE %%%%%%%%%%%%%%%%%
EqD= round((L+ChL)/2); %delay of input desired response

% Initialize filter coefficients
w_hat = zeros(L, 1);

% Initialize error signal
e = zeros(T, 1);
u  = [zeros(1,EqD) u zeros(1,L) ];

for n=1:T
    un= transpose(u(n+L-1:-1:n));
    e(n) = d(n)-w_hat'*un;

    w_hat = w_hat + mu*un*e(n)';
end

tapInputs = un;

%% Transmission phase
d=round(rand(1,N))*2-1;
d=d+1i*(round(rand(1,N))*2-1);

% signal filtered by FIR delay line channel
x= filter(Ch,1,d);

% Noise
v=randn(1,N);
v=v/norm(v)*10^(-SNRdB/20)*norm(x);

% Received signal
u=x+v;


%%%% Add proper padding (zeros) to u %%%%%%%%



Un=zeros(L,N);% Un is the matrix such that n-th column
```

```
                % corresponds to vector:
                % u(n) = [u(n) u(n-1) ... u(n-L+1)]
for n=1:N-L
    Un(:,n) = flip(transpose(u(n:n+L-1)));
end

sb=w_hat'*Un;   % recieved symbol estimation where w_hat is LMS filter in training

%SER(decision part)
sb1=sb/norm(w_hat);  % normalize the output
sb1=sign(real(sb1))+1i*sign(imag(sb1));  %symbol detection
sb2=sb1-d(1:length(sb1));  % error detection
SER=length(find(sb2~=0))/length(sb2); %  SER calculation
disp(SER);
disp(ath);
SERlist(ath) = SER;
ath = ath +1 ;
%end

% plot of transmitted symbols
    subplot(2,2,1),
    plot(d,'*');
    grid,title('Input symbols');  xlabel('real part'),ylabel('imaginary part')
    axis([-2 2 -2 2])

% plot of received symbols
    subplot(2,2,2),
    plot(u,'o');
    grid, title('Received samples');  xlabel('real part'), ylabel('imaginary part')

% plots of the equalized symbols
    subplot(2,2,3),
    plot(sb,'o');
    grid, title('Equalized symbols'), xlabel('real part'), ylabel('imaginary part')

% convergence of LMS algorithm
    subplot(2,2,4),
    plot(abs(e));   % e is estimation error in LMS algorithm
    grid, title('Convergence'), xlabel('n'), ylabel('error signal')
```

References

[1] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory.* Prentice Hall International, 1993.

[2] S. Haykin, *Adaptive Filter Theory.* Pearson, 2014. [Online]. Available: https://books.google.fi/books?id=J4GRKQEACAAJ