MATLAB Simulation Exercise

Statistical Signal Processing II

RLS ALGORITHM

Group 13

Athmajan Vivekananthan

University of Oulu
Finland

February 21, 2024

1 Background

The filtering operation in the RLS algorithm is constituted by the excitation of the priori estimation error computation followed by the adaptive operation of RLS algorithm where the tap weights are incrementally changed with the priori estimate error times the time varying gain vector. One important algorithmic benefit in RLS algorithm is that the necessity to inverse the correlation matrix has been replaced by the a simple scalar division which contributes greatly for complex applications. The convergence of the LMS and RLS algorithm has been also analyzed in this exercise. [1] [2]

2 Results and Discussion

The Matlab code for Task 1 has been attached in Section 3. After exploring performances LMS and RLS algorithms for channel equalization based on convergence and Symbol Error Rate (SER) the following results have been presented in this exercise.

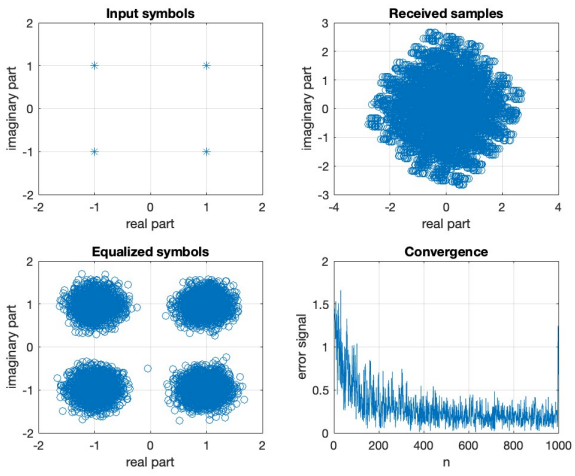

Figure 1: Input, Received, Equalized Samples and Convergence of LMS algorithm

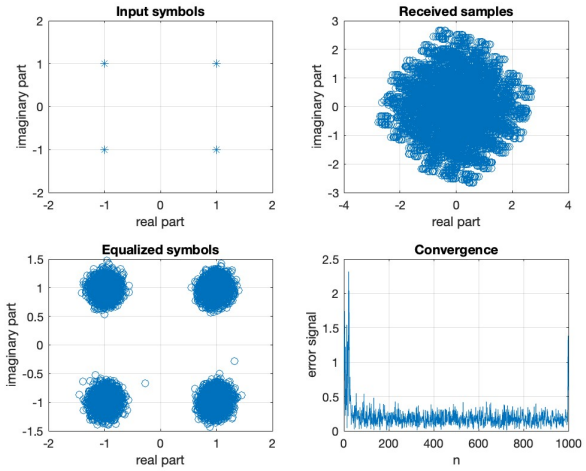

Figure 2: Input, Received, Equalized Samples and Convergence of RLS algorithm with forgetting parameter = 1

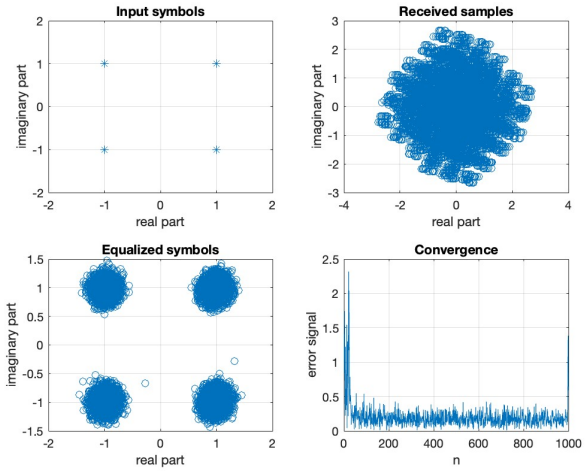

Figure 3: Input, Received, Equalized Samples and Convergence of RLS algorithm with forgetting parameter = 0.98

As per the results presented in Table 1 and Figures 2 and 3 it can be seen that the SER values and the convergence of RLS algorithm depends on the forgetting parameter. Since this has no relation with the LMS algorithm it is unaffected. The performance of the algorithms are also notable affected

Table 1: SER values for LMS and RLS algorithms with changing λ parameter with fixed tap length

λ	LMS SER	RLS SER
0.5	0.0098	0.2944
0.98	0.0098	0.0179
1.0	0.0098	0.0037
2.0	0.0098	0.2235

by the length of the filters.

This phenomenon can be seen with the observations in convergence when changing the tap length as seen in Figure 4. As per calculation formula [2] for RLS algorithm it can be seen that the time averaged correlation matrix is uniquely defined by the input sequence. However the measurement error is independent of the input vector. Also the measurement error has zero mean. Deriving from these it can be shown that the convergence of the error in the mean value is marked at the number of taps in the RLS adaptive filter. Unlike LMS algorithm the filter does not have to converge asymptotically.

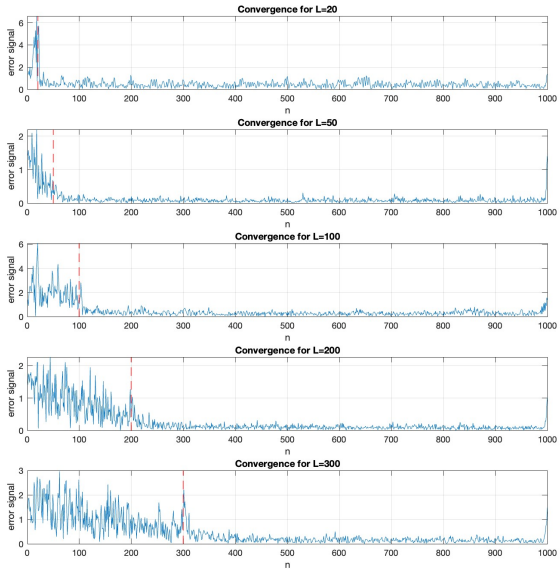

Figure 4: Convergence of RLS algorithm with varying tap length

3 Matlab Code

```
clear all;
close all;
clc;

N=1e4;    % number of data samples
T=1e3;    % number of training symbols
dB=25;     % SNR in dB value

rng(42);

L=20; % length of the filter

lamb = 1;
mu = 0.01;
d_1 = 0.004;

%%%%% QPSK TRANSMISSION %%%%%%%
TxS=round(rand(1,T))*2-1;
TxS=TxS+1i*(round(rand(1,T))*2-1);

%%%%%%%% Channel %%%%%%%%%%%%%%%
ChL=5;  % length of the channel(ChL)
Ch=randn(1,ChL)+1i*randn(1,ChL);
Ch=Ch/norm(Ch);

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

EqD= round((L+ChL)/2);  %delay for equalization

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

%%%%%%%% RECEIVED SIGNAL %%%%%%%%%%%%%%%
u=x+v;
% w
```

```
%%%%%%%% WRITE RLS AND QR-RLS ALGORITHM FOR EQUALIZATION HERE %%%%%%%%%%

u  = [zeros(1,EqD) u zeros(1,L) ];
%y

w_hat_LMS = zeros(L,1);
w_hat_RLS = zeros(L,1);
en_LMS = zeros(T,1);
en_RLS = zeros(T,1);

pn = eye(L)/d_1;

for n=1:T
    un= transpose(u(n+L-1:-1:n));

    % LMS algorithm
    en_LMS(n) = TxS(n) - w_hat_LMS'*un;
    w_hat_LMS = w_hat_LMS + mu*un*en_LMS(n)';

    % RLS algorithm
    kn = ((1/lamb)*pn*un)/(1 + (1/lamb)*un'*pn*un);
    pn = (1/lamb)*(pn-kn*un'*pn);
    en_RLS(n) = TxS(n) - w_hat_RLS'*un;
    w_hat_RLS = w_hat_RLS + kn*en_RLS(n)';


end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
TxS=round(rand(1,N))*2-1;
TxS=TxS+1i*(round(rand(1,N))*2-1);

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

%%%%%%%%% NOISE %%%%%%%%%%%%%%%%
v=randn(1,N);
v=v/norm(v)*10^(-dB/20)*norm(x);
```

```
%%%%%%%% RECEIVED SIGNAL %%%%%%%%%%%%%%%
u=x+v;

%%%%%%%%%%%%% EQUALIZATION %%%%%%%%%%%%%%%%%%

un=zeros(L,N);
u = [zeros(1,EqD) u zeros(1,L) ];
for n=1:N
    un(:,n)= transpose(u(n+L-1:-1:n));
end

%% LMS
sb_LMS=w_hat_LMS'*un;   % recieved symbol estimation

%SER(decision part)
sb1=sb_LMS/norm(w_hat_LMS);  % normalize the output
sb1=sign(real(sb1))+1i*sign(imag(sb1));  %symbol detection
sb2=sb1-TxS(1:length(sb1));  % error detection
SER=length(find(sb2~=0))/length(sb2); %  SER calculation
disp(['LMS SER:  ',num2str(SER)]);

figure
% plot of transmitted symbols
    subplot(2,2,1),
    plot(TxS,'*');
    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_LMS,'o');
    grid, title('Equalized symbols'), xlabel('real part'), ylabel('imaginary part')

% convergence
    subplot(2,2,4),
```

```
    plot(abs(en_LMS));   % en is priori estimation error
    grid, title('Convergence'), xlabel('n'), ylabel('error signal')

%% RLS
sb_RLS=w_hat_RLS'*un;   % recieved symbol estimation

%SER(decision part)
sb1=sb_RLS/norm(w_hat_RLS);  % normalize the output
sb1=sign(real(sb1))+1i*sign(imag(sb1));  %symbol detection
sb2=sb1-TxS(1:length(sb1));  % error detection
SER=length(find(sb2~=0))/length(sb2); %  SER calculation
disp(['RLS SER:  ',num2str(SER)]);

figure
% plot of transmitted symbols
    subplot(2,2,1),
    plot(TxS,'*');
    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_RLS,'o');
    grid, title('Equalized symbols'), xlabel('real part'),
    ylabel('imaginary part')

% convergence
    subplot(2,2,4),
    plot(abs(en_RLS));   % en is priori estimation error
    hold on
    xline(L, 'r--', 'LineWidth', 1);
    grid, title('Convergence'), xlabel('n'), ylabel('error signal')
    hold off
```

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