LMS Adaptive Filter
Matlab Exercise

Signal Processing in Telecommunications 1

TABLE OF CONTENTS

1. INTRODUCTION

This document gives the background information and guidelines for completing the project work for the course , Signal Processing in Telecommunication 1. This programming exercise is
a required element of the course. This assignment is graded out of 12 points and will contribute to the final homework grade. All the required documentation (see Section 6) should be returned to the course locker below the course information board. Remember to put your contact information on the cover page, i.e., name, study and group number and e-mail address. If the document is returned after the deadline the grade for the document has to be decreased by one for each delayed day! If you have questions or problems related to the exercise you can contact the course assistant.

The project work should be solved using the Matlab programming environment [1]. Matlab is available, for example, on the computers at the Computing Center. On those machines you start Matlab by typing *use matlab,* and then *matlab* at the UNIX prompt. Useful commands for online information about the available Matlab functions are *lookfor <keyword>* and then *help <command>*.

The project may be completed in one- or two-person groups. All groups should register with the assistant (via email, for example), so that they obtain a group number and corresponding channel parameter values. Students who choose to work in groups of two must still hand in two separate and distinct reports, so that it is seen that each student has understood and contributed to the project. The reports will be graded separately.

In the following sections the general system framework, the description of the prototype Matlab program, a sample demo and the requirements for how the equalizer task should be accomplished are presented.

2. GENERAL SYSTEM FRAMEWORK

Figure 1 shows the general discrete-time transmission system you will be working with in this exercise.

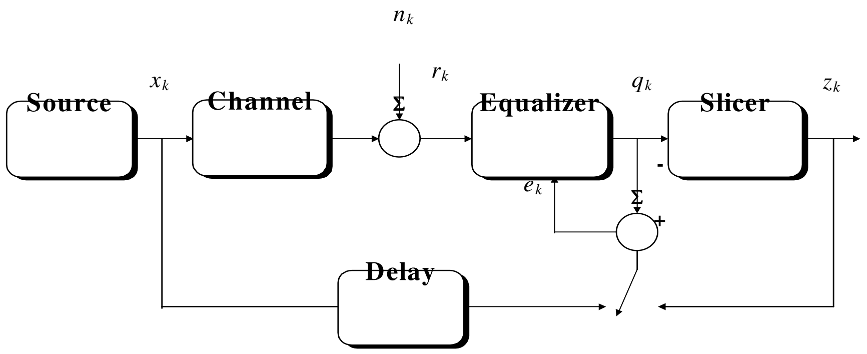

Figure 1 Adaptive equalizer in a chain of the transmission system

The *source block* transmits Binary Phase Shift Keying (BPSK) symbols $x_k \in \pm 1 (k = 1,..., K)$ with equal probability. The total number of transmitted symbols is denoted as *K*. The *channel block* introduces intersymbol interference (ISI) using a finite impulse response (FIR) type of channel model. The transfer function of the channel $H(z)$ can be expressed as

$$H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + \ldots + h_{L-1} z^{-(L-1)} \qquad (1)$$

where h_l (l=0, …, L-1) are the values of the sampled impulse responses and L is the number of channel filter coefficients. At the output of the channel, a noise sequence n_k is added. This noise is assumed to be additive white Gaussian noise (AWGN) with variance $\sigma_n^{\,2}$. The sum of the channel output and the noise sequence forms the received signal r_k , which is fed into the *equalizer block*. Finally, the equalizer output q_k is fed to the *slicer* to obtain the estimate z_k of the transmitted data symbol x_k.

The *equalizer block* performs the equalization of the channel. Different performance criteria can be utilized in the equalization [2]. In this exercise work you will design an adaptive equalizer using the mean square error (MSE) criterion. The minimization of the cost function can be performed adaptively by applying the stochastic gradient (SG) or the least mean square (LMS) algorithm. The LMS algorithm update of the equalizer coefficient vector is given by [3]

$$\mathbf{c}_{k+1} = \mathbf{c}_k + 2\mu e_k \mathbf{r}_k \qquad (2)$$

where $\mathbf{r}_k = [r_k \ \ r_{k-1} \ \ldots \ r_{k-(N-1)}]^T$ is the input vector, $\mathbf{c}_k$ is the weight vector, e_k is the error signal and μ is the step size. This step size μ controls the adaptation speed of the adaptive filter. The number of the adaptive filter coefficients has been denoted as N*.

In order to implement the adaptive equalizer, we need to generate a reference signal for the adaptive algorithm. For the initial adaptation of the filter coefficients we need at the receiver to be able to generate a replica of the transmitted data sequence. This known sequence is referred to as the training sequence. During the training period the desired signal is used as a reference signal and the error signal is defined as $e_k = x_{k-D} - q_k$ (see Figure 1). After the training period, the equalization can be performed in decision-directed manner. This mode can be utilized if the channel can be assumed to be time-variant. Therefore, it can be assumed that the decisions in the slicer output are correct most of the time and the slicer decisions can be used as reference signal. In the decision-directed mode, the error signal is defined as $e_k = z_k - q_k$ (see Figure 1). The mean square error (MSE) for the filter in the kth time instant is defined as

$$MSE_k = E\left[\left|e_k\right|^2\right] \qquad (3)$$

In simulations the expectation in (3) is evaluated through averaging over I independent runs, in order to be able to view the convergence of the equalizer as a function of the time. After the convergence of the adaptive equalizer, we can compute the final *MSE* by time averaging if the process is ergodic. However, it is important that the initial transient of the equalizer is discarded. Otherwise, you will end up with a too large value of the *MSE*.

*(Note that the notation is slightly different from the notation used in the lectures.)

3. MATLAB PROGRAM DESCRIPTION

This section describes parts of the sample MATLAB program you work with in this exercise. The program can be divided into four parts: initialization, transmitter, channel, and equalizer corresponding to the blocks in Figure 1. The initialization part controls the system parameters and can be modified in order to study their impact on the equalizer performance. The values presented here are the same as for the sample demo shown in the next section.

```
%%%-------------------------------------------------------------------
%%%                      Initialization
%%%-------------------------------------------------------------------
clear all;                          % Clear all the variables.
K=500;                              % Number of symbols.
I=200;                              % Number of independent runs.
N=11;                               % Number of filter coefficients.
MSE=0*ones(1,K);                    % MSE for adaptive filter.
sigma2_n=0.001;                     % Noise power.
hn=[0.2194 1.0000 0.2194];          % Channel impulse response.
my_max=1/(N*(sum(hn.^2)+sigma2_n)); % Maximum step size (Eq 5).
my=my_max/2;                        % Step size parameter.
D=7;                                % Delay factor.
```

The transmitter part generates the data and reference signals. Note that the reference signal is delayed by simply inserting zeros in the beginning of the data sequence.

```
%%%-------------------------------------------------------------------
%%%                      Transmitter
%%%-------------------------------------------------------------------
x=sign(randn(1,K));                 % Generate BPSK data.
ref_x=[0*ones(1,D),x];              % Generate reference signal.
```

The channel parts introduce ISI through an FIR filter and white noise is added to the channel output.

```
%%%-------------------------------------------------------------------
%%%                      Channel
%%%-------------------------------------------------------------------
for i=1:I,                          % Run all the indepedent runs.
  ['Iteration:',mat2str(i)]         % Iteration index.
  channel_data=filter(hn,1,x);      % Distort the signal.
  n=sqrt(sigma2_n)*randn(1,K);      % Generate white noise sequence.
  channel_output=channel_data+n;    % AWGN channel.
```

The equalizer part is where you will insert your equalizer. The weight vector **c** can be initialized with zeros. After running through all the independent iterations the results can be analyzed.

```
%%%-------------------------------------------------------------------
%%%                      Equalizer
%%%-------------------------------------------------------------------
  [c,e,q,z]=lms(channel_output,ref_x,my,N);    % Linear LMS equalizer.
  MSE=MSE+abs(e).^2;                           % Mean Square Error(MSE).
end
```

4. SAMPLE DEMO

In this section we present a demo example for the adaptive equalization to aid your own work. This demo example uses a channel model described in [4] and the impulse response of cosine type h_l (l=0, …, 2) given by

$$h_l = \frac{1}{2}\left[1+\cos\left(\frac{2\pi}{W}(l-1)\right)\right] \tag{4}$$

where the parameter W controls the amplitude distortion (W=2.9). The values of the parameters used are given in Table 1 below. The value of the step size μ can be determined in the following way [4][5]. The step size μ should be selected smaller than the maximum step size $\mu_{\max}$ in order to avoid instability. However, too small values result in slower convergence. The maximum value can be calculated from

$$\mu_{\max} = \frac{1}{Trace[\mathbf{R}]} = \frac{1}{N\left(\sum_{l=0}^{L-1} h_l^2 + \sigma_n^2\right)} \tag{5}$$

For the demo system, the step size $\mu = \mu_{\max}/3$ was selected in order to get fast enough convergence.

Table 1 System parameters for the sample demo

Parameter	Value
Number of independent runs I	200
Number of transmitted symbols K	500
Number of filter coefficients N	11
Noise power σ_n^2	0.001
Step size μ ($\mu_{\max}/3$)	0.0276
Channel coefficients [h_0 h_1 h_2]	[0.2194 1.0000 0.2194]

Figure 2 shows MSE for adaptive filter. This plot confirms that the equalizer converges close to the noise level. Figure 3 shows the impulse responses for the channel, the weight coefficients of the adaptive filter after the convergence, and the convoluted impulse response of the channel and equalizer. The impulse response of the channel is divided into the causal and non-causal parts and is symmetric about l = 1. The equalizer response is also symmetric about n = 8. As a rule of thumb, the delay factor D should be selected near the value $D = (N+L)/2 = 7$. Figure 4 shows the respective frequency responses. Flat frequency response confirms that the equalizer has enabled to equalize the channel. Figure 5 shows the discrete time signal representations for the data, for the data after the channel, for the data after the equalizer and for the slicer output data. The effect of the equalizer on the transmitted data can be concluded from these subplots.

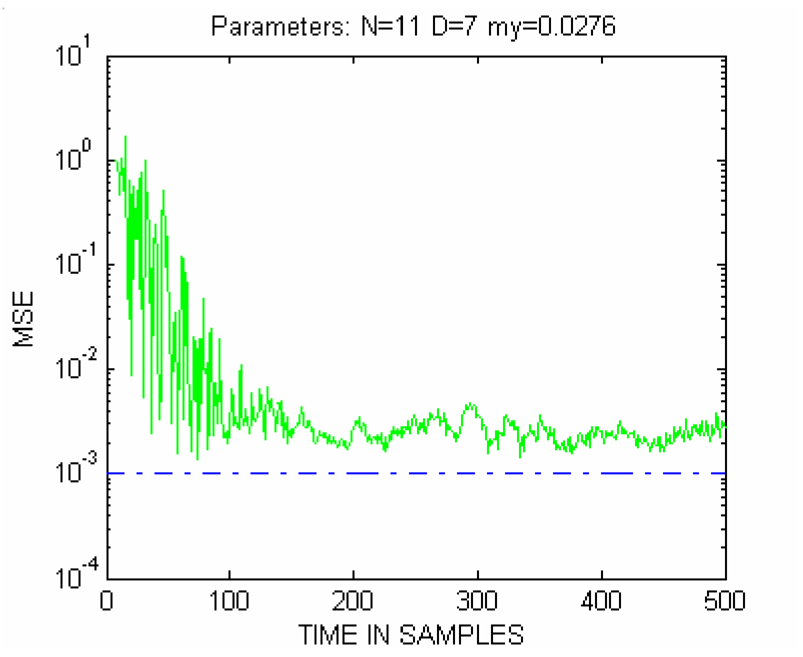

Figure 2 MSE as a function of time instant *k*

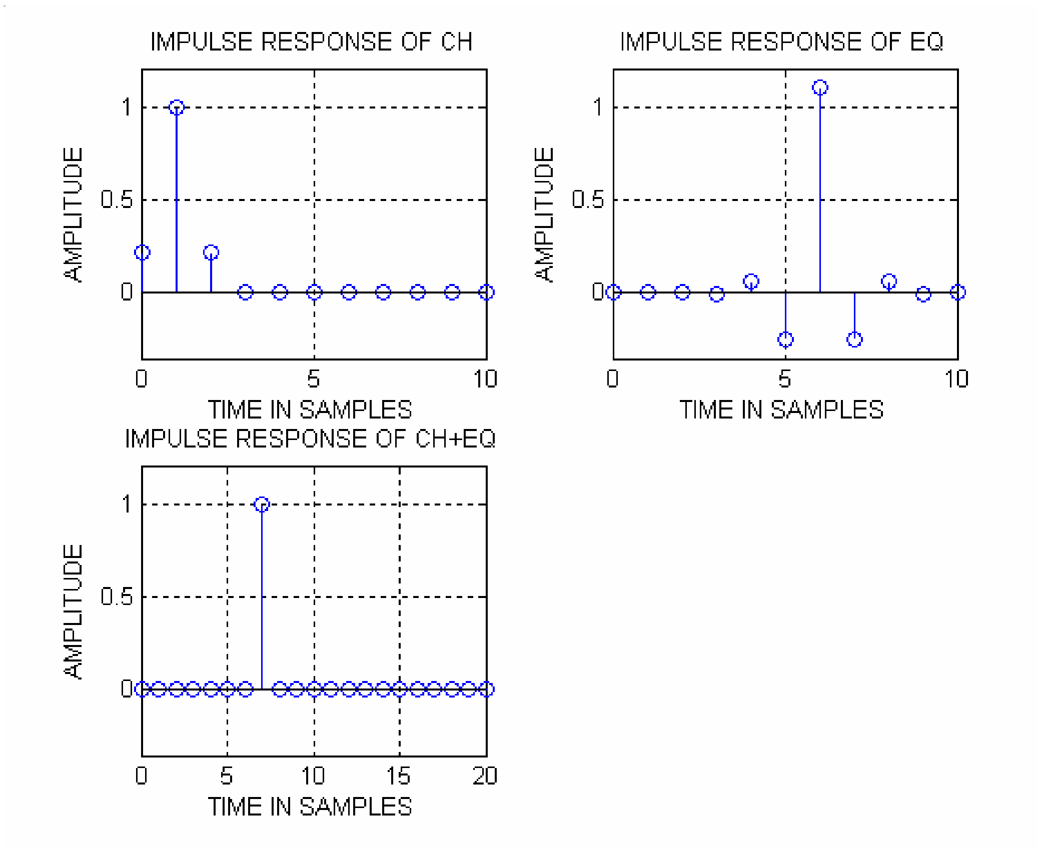

Figure 3 Impulse responses of the channel, the equalizer, and the channel plus equalizer

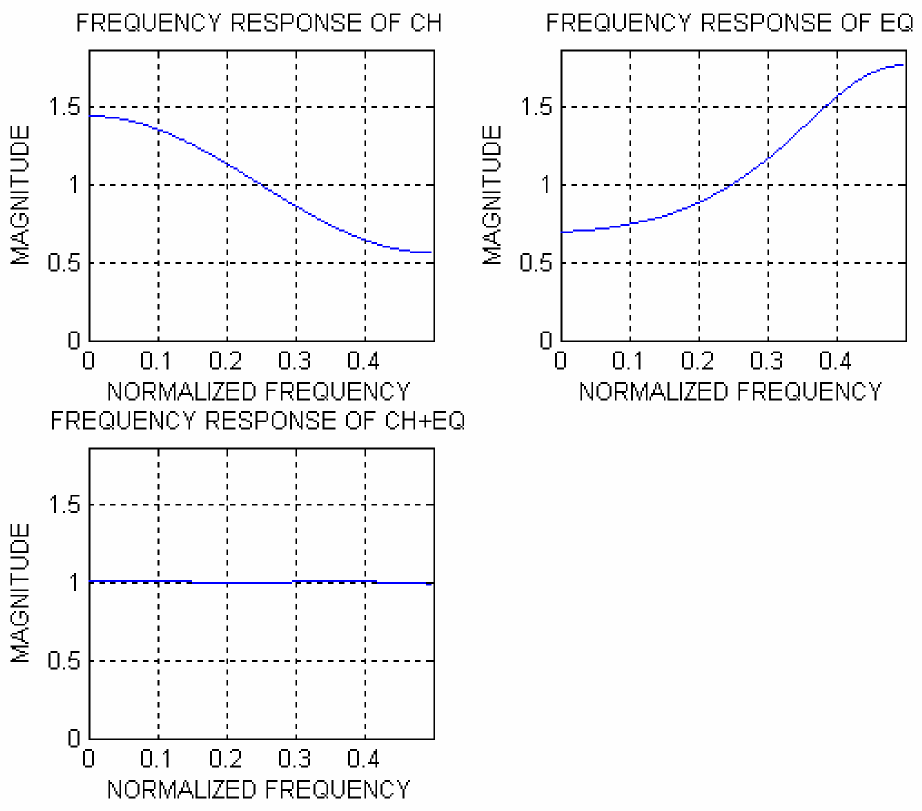

Figure 4 Frequency responses of the channel, the equalizer and the channel plus equalizer

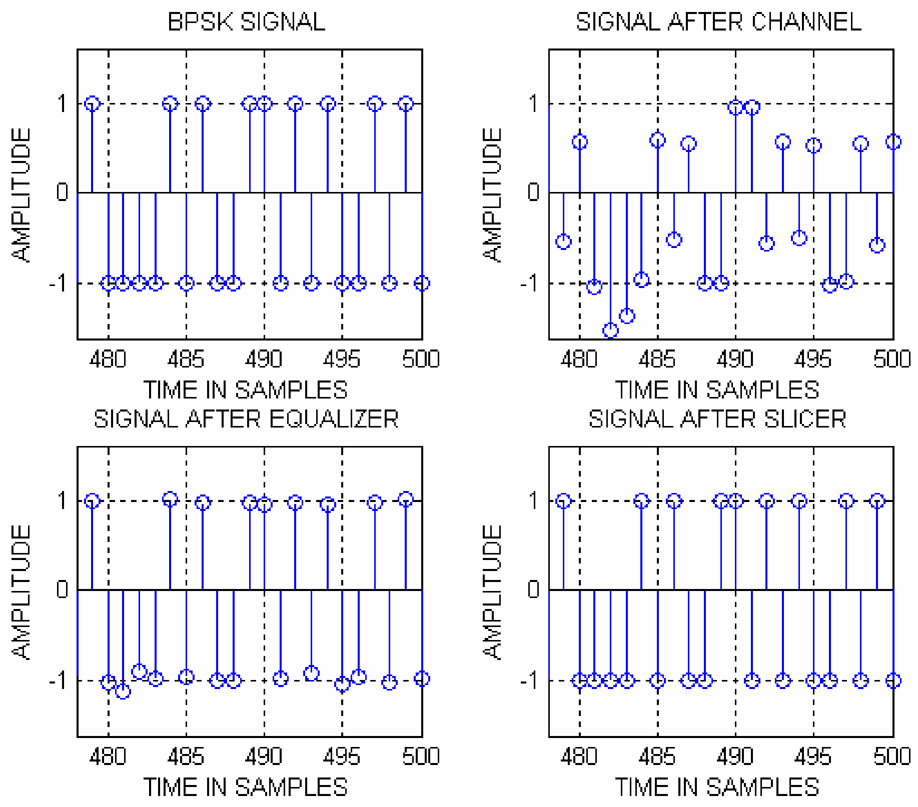

Figure 5 Discrete time signal for transmitted data, the data after the channel, the data after equalizer and the data at the slicer output

5. EXERCISE REQUIREMENTS

In this section the exercise tasks for each group are specified. It is recommended that the prototype Matlab program be used. You are not strictly restricted to this though, so that you may freely modify it. Use the channel coefficients h_l (l=0, …, L-1) according to your group number as specified in Table 2 below.

Table 2 Coefficients of the channel model

Group No	h_0	h_1	h_2	h_3	h_4	h_5	h_6
1	1	-0.5	0.25	-	-	-	-
2	1	-0.3	0.07	0.09	-0.02	-	-
3	1	-0.1	-0.14	0.149	-0.0125	-0.0175	0.003
4	1	0.2	-0.08	-	-	-	-
5	1	0.4	-0.19	-0.046	0.012	-	-
6	1	0	-0.19	0.094	0	-0.01216	0.00192
7	1	0.2	-0.15	-	-	-	-
8	1	-0.2	-0.07	0.092	-0.024	-	-
9	1	0.1	-0.04	0.053	-0.0027	0.00108	-0.00216
10	1	-0.4	0.16	-	-	-	-
11	1	-0.1	0.13	0.012	0.0144	-	-
12	1	-0.2	0.15	-0.002	0.0145	-0.00132	0.000144
13	1	0.3	0.09	-	-	-	-
14	1	0.2	0.07	-0.006	0.0009	-	-
15	1	0.6	0.09	0.01	-0.0057	0.00072	-0.000054
16	1	-0.1	0.01	-	-	-	-
17	1	0.3	-0.09	0.01	-0.0006	-	-
18	1	0.1	-0.11	0.04	-0.0062	0.00052	-0.000024
19	1	0.4	-0.06	-	-	-	-
20	1	0.2	-0.1	0.028	-0.0024	-	-
21	1	0.4	-0.09	0.002	0.0062	-0.00132	0.000072
22	1	-0.2	0.04	-	-	-	-
23	1	0	-0.03	0.014	-0.0012	-	-
24	1	0.3	0.07	0.005	0	0.00104	-0.00012
25	1	0.2	-0.03	-	-	-	-
26	1	0.5	0.13	0.011	-0.003	-	-
27	1	0	0.13	0.071	0.024	0.00425	-0.00075
28	1	0.3	0.1	-	-	-	-
29	1	-0.2	0.2	0.025	0.025	-	-
30	1	0	0.08	0.081	0.014	0.003	-0.002
31	1	-0.5	0.25	-	-	-	-
32	1	-0.3	0.07	0.09	-0.02	-	-
33	1	-0.1	-0.14	0.149	-0.0125	-0.0175	0.003
34	1	0.2	-0.08	-	-	-	-
35	1	0.4	-0.19	-0.046	0.012	-	-
36	1	0	-0.19	0.094	0	-0.01216	0.00192

37	1	0.2	-0.15	-	-	-	-
38	1	-0.2	-0.07	0.092	-0.024	-	-
39	1	0.1	-0.04	0.053	-0.0027	0.00108	-0.00216
40	1	-0.4	0.16	-	-	-	-

Exercise specifications:

1. Implement a linear LMS-based equalizer block by adding the necessary parts into the given Matlab program.

2. Use, for example, N = 11 filter coefficients and fix the noise variance to the small value of σ_n^2=0.001.

a. What is the largest step size $\mu_{\max}$ which still guarantees the convergence of filter coefficients in your problem?

b. Find the optimal delay factor D for your system (function *grpdelay* might also be useful). Has the delay factor any influence on the results and how?

c. Experiment with the step size μ. How the convergence is affected by the selection of the step size?

d. How does the change in the number of filter coefficients N affect the results?

e. How does the increase in the noise level affect the performance?

Give MSE plots in each case. It is recommended when studying the effect of a parameter that the rest of the parameters are kept fixed. The MSE curves can be combined into a single plot for ease of comparison in each parameter study case.

3. Choose a step size smaller than $\mu_{\max}$ (for example $\mu_{\max}/5$) and use a large number of symbols to make sure that the equalizer has converged.

a. Plot the impulse response of the channel, equalizer and convoluted response of the channel and equalizer by using, for example, 2N data points.

b. Plot the frequency responses of the channel, equalizer and the channel plus equalizer. What kind of results should we get in the ideal case and explain why the equalizer can not always achieve these aims?

c. For the last 2N symbols after that the equalizer has converged, plot the transmitted data, the data after the channel, the data after the equalizer, and the data at the slicer output. What is the effect of the equalizer on the received symbols? How can the effect of choice of the delay factor D be seen in these plots?

4. Finally, do some experiments by using decision directed mode. How does the MSE behave when switched to the decision directed mode at different times before convergence? Give some illustrative MSE plots.

6. DOCUMENTATION

The project work should be documented. Explain in sufficient detail what you have done, how and why, so that the document shows that you have understood the problem. The document should include the answers to the questions and the necessary simulation plots. Also, add the listing of your Matlab program(s) for LMS equalizer with comments as part of your document in the Appendix. When documenting, you can use the format of this document for both the text part and plots. In order to import the current plot in the encapsulated postscript form from the MATLAB environment to the Word document the command *print -deps -tiff picture.eps* is recommended. Additional comments how the exercise work could be improved are welcome. Good luck!

REFERENCES

[1] MATHWORKS Inc, *MATLAB High-performance Numeric Computation and Visualization Software, User's Guide*, South Natick, MA, USA, 1994.

[2] E. A. Lee and D. G. Messerschmitt, *Digital Communication*, Kluwer Academic Publishers, second edition, Boston, 1994.

[3] S. U. H. Qureshi, "Adaptive Equalization", *Proceedings of the IEEE*, Vol. 73, No. 9, Sep. 1995, pp. 1349-1287.

[4] S. Haykin, *Adaptive Filter Theory*, Prentice Hall, third edition, New Jersey, 1996.

[5] P. S. Diniz, *Adaptive Filtering: Algorithms and Practical Implementation*. Kluwer Academic Publishers, Boston, MA, USA, 1997.