Adaptive and Array Signal Processing

Homework 06

1. A stationary signal $x[n]$, that is of zero mean, unity power and temporally uncorrelated,

$$\mathrm{E}\left[x[n]\right] = 0, \;\; \mathrm{E}\left[x[n]x^*[n-k]\right] = \begin{cases} 1 & k = 0 \\ 0 & k \neq 0 \end{cases} ,$$

is passed through a channel with time-discrete impulse response $h[k]$, such that the noiseless received signal $u[n]$ reads as

$$u[n] = \sum_{k=0}^{L_h-1} h[k] \cdot x[n-k] \, ,$$

where L_h is the length of the channel's impulse response. For $L_h > 1$ the channel will introduce inter-symbol interference (ISI), that the receiver wishes to remove by means of a linear minimum mean square error (LMMSE) adaptive equalizer that produces the output $y[n]$

$$y[n] = \sum_{m=0}^{M-1} w^*[m] \cdot u[n-m] \, ,$$

where M is the filter length and $w[0], w[1], \ldots, w[M-1]$ are the filter coefficients.

(a) Express the filter output $y[n]$ in terms of the two vectors $\mathbf{w} = [w[0] \; w[1] \; \ldots \; w[M-1]]^{\mathrm{T}}$ and $\mathbf{u}[n] = [u[n] \; u[n-1] \; \ldots \; u[n-M+1]]^{\mathrm{T}}$.

(b) The desired output $d[n]$ of the adaptive filter is chosen to be

$$d[n] = x[n-M+1] \, ,$$

in order to make maximum use of the filter memory. Compute the Wiener solution in terms of a correlation matrix $\mathbf{R}$ and a correlation vector $\mathbf{p}$.

(c) Now compute the correlation coefficients $r[m] = \mathrm{E}\left[u[n] \cdot u^*[n-m]\right]$ and express the correlation matrix $\mathbf{R}$ and correlation vector $\mathbf{p}$ from subtask 1b as a function of the channel impulse response $h[k]$.

(d) For the case $M = L_h = 2$, with $h[0] = a \in \mathbb{C}$, $h[1] = b \in \mathbb{C}$ and zero elsewhere, compute the Wiener solution for $\mathbf{w}$ explicitly in terms of a and b. Now set $b = 0$ and compute the current error $e[n] = d[n] - y[n]$.

(e) Now set the filter length to $M = 6$, and use the channel from subtask 1d to compute the Wiener solution for $a = 4$ and $b = -5$ by using matlab. How large is the residual mean square error (MSE)?

(f) Write a matlab program that will plot the square root of the residual mean squared error as a function of the filter length M in the range $2 \leq M \leq 50$ in steps of one. Note: Use the `semilogy` function for the plotting.

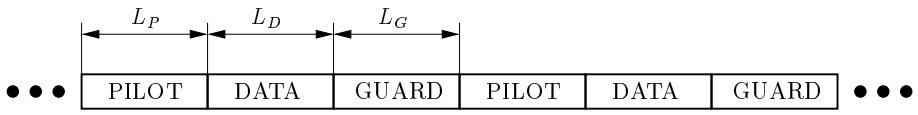

Figure 1: Frame structure of the transmitted data stream

2. The adaptive equalizer from task 1 shall now be implemented using the LMS algorithm. To this end assume the transmitted data stream is structured in frames as depicted in Figure 1. Each frame starts with a training sequence (pilot) of length L_P symbols, which is followed by L_D symbols of actual data. At the end of each frame there is a so called guard interval where the transmitter is quiet (transmitting zeros) for L_G symbols. In the following assume $L_G = L_h + M$. This will let the programmable receiver FIR-filter start in the zero state for each frame, i.e. the simulation can be carried out on a frame by frame basis. The symbols are chosen out of the set $\{-1, +1\}$, i.e. BPSK modulated. The pilot sequence used has length $L_P = 18$, and is defined as:

$$-1 \quad -1 \quad -1 \quad +1 \quad +1 \quad -1 \quad -1 \quad -1 \quad +1 \quad +1 \quad +1 \quad +1 \quad -1 \quad +1 \quad +1 \quad -1 \quad +1 \quad -1$$

where the sequence is transmitted going from left to right. In the following assume $L_D = 150$, $M = 6$, $L_h = 2$, $L_G = 8$ and the channel coefficients $h[0] = 4$ and $h[1] = -5$ as in subtask 1e.

   (a) Write a matlab program that will simulate the advances of LMS adaptive equalization by evaluating the number of bit-errors in 30 successive frames. Note, that the weight update by LMS is only performed during the pilot phases. In each frame initialize the LMS with the weight vector obtained from the previous frame. For the very first frame use the all zero vector as the initial weight vector. Set the step size to 0.0015 for the LMS. When computing bit errors, make sure you look at the data portion of the frame only.

   (b) Compare the final weight vector found by the LMS with the theoretical one derived in subtask 1e.

   Hints:

   - A frame can be generated e.g. like this:
     `data = sign(randn(L_d,1)); frame = [pilot;data;zeros(L_g,1)];`
   - The received signal and filter output can be computed e.g. like:
     `u = conv(h,frame); y = conv(conj(w),u);`
   - Bit detection may be done like this: `bits = sign(real(y))`
   - The desired signal is: `d = [zeros(M-1,1);pilot(1:L_p-M+1)];`
   - Use the `stem` function for plotting the bit errors
   - You can use the LMS program below if you like.

```
function w = LMS(w0,u,d,s)
%w = LMS(w0,u,d,s)
%LMS alorithm
%INPUT
%  w0:   [M,1]  :  Init weight vector, M = Number of taps
%  u :   [N,1]  :  Input data vector
%  d :   [N,1]  :  Desired output data vector
%  s :   [1,1]  :  step-size parameter
%OUTPUT
%  w :   [M,1]  :  final tap weight vector

if (nargin<4) error('Too few parameters'); end;
N = length(u);
M = length(w0);
w = w0;
x = zeros(M,1);
for k=1:N
   x = [u[k];x(1:M-1)];
   e[k] = d[k] - w'*x;
   dw = s * conj(e[k]) * x;
   w = w + dw;
end;
```