1TE651: Signal Processing
Homework Assignment 1: Acoustic Echo Cancellation

September 28, 2015

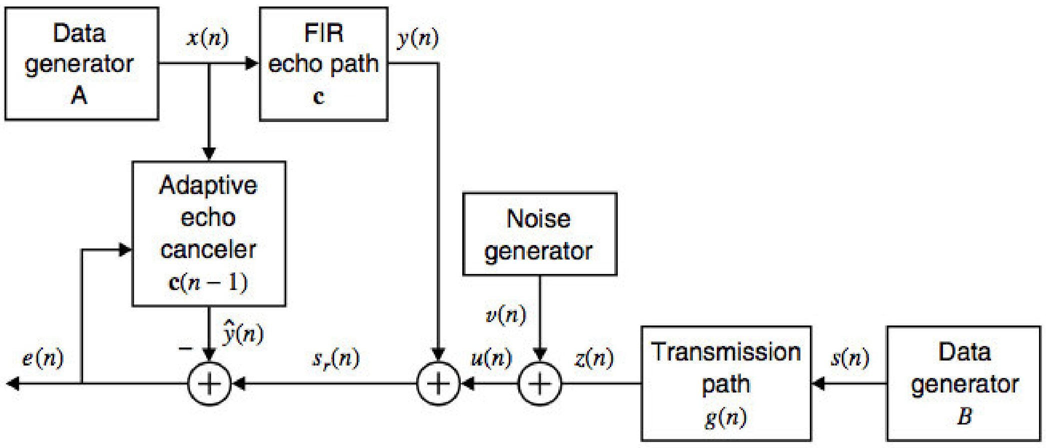

Figure 1: Echo Cancellation Block Diagram

1 Echo Cancellation

PLEASE READ CAREFULLY: In this lab you will implement an echo cancellation algorithm for an acoustic signal. To perform echo cancellation, you are going to use two algorithms, namely the Least Mean Square (LMS) and Recursive Least Square (RLS). The software should preferably be written in MATLAB. This lab can be done in a group of 2 or 3 students. The submission of the written lab report should contain an explanation of the design and implementation of the LMS and RLS algorithm for the acoustic echo cancellation procedure. You are requested to attach a number of plots and implement the two algorithms as mentioned in the detailed instructions to follow. You will be provided some audio files for the source signal. You are also required to include a copy of the source code. The submission deadline is midnight, October 30, 2015. Please submit your report as a hardcopy or a soft copy via e-mail to Subhra.Dey@signal.uu.se.

2 System Model

Figure 1 above illustrates a system with full duplex transmission. In a practical scenario, the far end signal $s(n)$ as shown in the figure passes through a channel with impulse response $g(n)$ and contaminated by noise $\nu(n)$. On the other hand the near end signal $x(n)$ gets reflected and forms the echo component $y(n)$ which is added into the signal to form a mixed signal $s_r(n)$ at the receiver. The echo distorts the far-end signal and the objective of echo cancellation is to remove the echo $y(n)$ as best as possible.

The best way to tackle this problem is to form a replica of the echo and then subtract it from the incoming signal. We can model the echo as the result of an echo path between the transmitter and the receiver. For baseband data transmission this echo path is basically linear and varies very slowly with time. The echo can be simulated by passing the audio signal through an finite impulse response (FIR) filter as below:

$$y(n) = c_1 x(n-d_1) + c_2 x(n-d_2) + \ldots + c_M x(n-d_M) \tag{1}$$

where $d_1, \ldots, d_M$ are the delays of the various echo paths in number of samples. Note that in order for the echoes to be heard, the delays should be greater than a certain interval, generally 40 milliseconds (as a guideline). The replica of the echo signal can be obtained in the receiver by implementing an adaptive filtering algorithm using LMS and RLS to estimate the coefficients $c_i,\ i = 1, 2, \ldots, M$. For details on these two algorithms and how to use them for estimating the parameters, please refer to the lecture notes.

Threre are several other things that need to be remembered while choosing the parameters $c_i,\ i = 1, 2, \ldots, M$ to model the aforementioned system. Firstly, the gain of the echo paths $c(n)$ should be decreasing in order as the delay increases. The noise added to the near end signal $\nu(n)$ is zero-mean additive white gaussian in nature. The variance of the noise σ^2 should be chosen reasonably low.

3 Algorithms

As mentioned previously, there are two algorithms based on LMS and RLS, that you are required to use to accomplish the acoustic echo cancellation. Find a brief description and the Pseudo-codes for these two algorithms below.

3.1 LMS Algorithm

The least mean square algorithm is used to estimate the echo path coefficient and more generally to replicate the echo in the receiver, so that it can be removed. The convergence of the algorithm relies heavily on the chosen step size μ for the algorithm. The larger the step size is, the algorithm converges faster, but is more sensitive to any system noise. The pseudo-code for the LMS algorithm is given below.

```
Summary of the LMS algorithm
Design Parameters
$\mathbf{x}(n)$ = input data vector at time $n$
$y(n)$ = echo component at time $n$
$\mathbf{c}(n)$ = filter coefficient vector for the echo paths at time $n$
$M$ = number of coefficients
$\mu$ = step-size parameters
$0 < \mu \ll \frac{1}{\sum_{k=1}^{M} \mathbf{E}\{|x_k(n)|^2\}}$
Initialize
$\mathbf{c}(0) = \mathbf{x}(0) = 0$
Computation
For $n = 0, 1, 2, ...$ compute,
$\hat{y}(n) = \hat{\mathbf{c}}^T(n-1)\mathbf{x}(n)$
$e(n) = s_r(n) - \hat{y}(n)$ (note that the received signal $s_r(n)$ contains $y(n)$).
$\hat{\mathbf{c}}(n) = \hat{\mathbf{c}}(n-1) + 2\mu\mathbf{x}(n)e(n)$
Until end of audio file
```

Here ^{T} denotes transpose and $\hat{\mathbf{c}}(n)$ denotes the estimated parameter vector of the echo path coefficients at iteration n.

3.2 RLS Algorithm

The Recursive least square algorithm is another algorithm for estimating the echo coefficients. This convergence of the algorithm depends on the initial value of a recursive matrix, which can be chosen as ρI. The value of ρ must be chosen significantly large implying low confidence in the initial assumption. The pseudo-code of the RLS algorithm is given below.

```
Summary of the RLS algorithm
Initialization
$\hat{\mathbf{c}}(0) = 0 \quad \mathbf{P} = \rho\mathbf{I}$
$\rho$ = a large positive constant
Adaptive gain computation
For $n = 0, 1, 2, ...$ compute,
```

$\tilde{\mathbf{K}}_\lambda(n) = \mathbf{P}(n)\mathbf{x}(n) = \frac{\mathbf{P}(n-1)\mathbf{x}(n)}{\lambda+\mathbf{x}^T(n)\mathbf{P}(n-1)\mathbf{x}(n)}$
$\mathbf{P}(n) = \lambda^{-1}\left\{\mathbf{P}(n-1) - \frac{\mathbf{P}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)P(n-1)}{\lambda+\mathbf{x}^T(n)\mathbf{P}(n-1)\mathbf{x}(n)}\right\}$
Filtering
$e(n) = s_r(n) - \hat{\mathbf{c}}^T(n-1)\mathbf{x}(n)$ (note that the received signal $s_r(n)$ contains $y(n)$).
Coefficient updating
$\hat{\mathbf{c}}(n) = \hat{\mathbf{c}}(n-1) + \tilde{\mathbf{K}}(n)e(n)$
Until end of audio file

4 Instructions

For the system mentioned above:

- Load the two audio files "vocal.wav" and "drumloop.wav". You may find the MATLAB command "audioread" useful in this respect. Note that you can also obtain the sampling frequency of this audio file by properly using the "audioread" function. You can use one of these audio files as the near end signal and the other one as the far end signal. Please make sure that both of the audio files have the same number of samples for convenience. If one of them has less number of samples than the other, you can pad it with required number of zeroes to make it the same length as the other one. Listen to the audio signals using the "sound" command.

- Generate the echo $y(n)$ from the provided far end signal $x(n)$ using (1) while choosing the coefficients c_i and the delays d_i appropriately. You can choose between 3-5 echo paths. Plot the original far end signal and echo signal. Also listen to the original audio signal and the echo.

- Generate the noise contaminated near end signal $u(n)$ by adding zero-mean Gaussian noise of an appropriate variance σ^2 to the near end signal obtained from the audio file. Assume the channel $g()$ to be unity. Plot the noise contaminated signal, and also listen to it.

- Generate the mixed signal $s_r(n)$ of noise contaminated far end signal added with the near end echo. Plot the mixed audio signal and listen to it.

- Estimate of the echo coefficients $c(n)$ using both the LMS and RLS algorithms. In the first instance, assume that the delay elements $d_1, \ldots, d_M$ of the echo component are known, since you generated them yourself. Note that in this case you only have to estimate the M coefficients. For the second case assume the delays are unknown, but assume that the maximum delay d_M is less than some suitably chosen number N_M. In such a scenario, the estimated FIR filter should have tap coefficients from one to the maximum delay component. In the output of your estimation algorithm, many of these coefficients not representing any real delay components should be close to zero or very small.

- Remove the echo from the mixed audio signal $\hat{y}(n)$ using the echo coefficient estimates $\hat{c}(n)$. Plot the recovered signal. Compare it with the noise contaminated far end signal by plotting the squared error between them. Also determine the average squared error (averaged over all samples) in this case. Listen to the recovered signal also. Do you observe or hear much difference between the recovered signal with LMS as opposed to RLS?

- Compare the echo added to the signal, $y(n)$, with its estimate $\hat{y}(n)$ by plotting the squared error between them. Also determine the time-averaged squared error in this case. Listen to the two signals as well.

- In the case where the delays are known, plot the evolution of the M estimated echo-path coefficients along with their true values against the time axis or the iteration number for both LMS and RLS. Comment on the convergence rates of LMS and RLS.

- Compare the averaged square error in the parameter estimates for RLS and LMS. Comment on the difference.

- While running the parameter estimation algorithms for LMS and RLS, compute the time required to run the algorithm (please make sure you only compute the CPU time for only the parameter estimation algorithms). Comment on the difference.

Note: You can use the following book (available electronically via the Uppsala University Library) as a reference in addition to the lecture notes.
Manolakis, Ingle and Kogon, *Statistical and Adaptive Signal Processing: Spectral Estimation, Signal Modeling, Adaptive Filtering and Array Processing*, Artech House, 2005. (Refer to Chapter 10).