DSP Project

Simple echo cancellation

This document provides an outline for a Matlab project to be completed by the end of the course. You are expected to investigate, in detail, methods related to solving the problem. There is a design element to the project, and a quantitative evaluation of the performance of the proposed methods must be performed and presented. You are to write up a comprehensive report (of no more than 8 pages) describing your method and results. You should work in groups of two, although you may work alone if you really want to.

If you wish to propose a project of your own, then please come and talk to me. Project descriptions from previous years are on the course website, and are also options.

Note that this document is still in preparation, and may be added to during the course of the project.

The Task

The aim of this project is to investigate methods of restoring signals that are corrupted by one or more echos. Echos commonly occur over communications channels such as telephone and ADSL lines. A reasonable model of the echo process in the time domain is

$$y[n] = x[n] + a_1 x[n - n_d] + a_2 x[n - 2n_d],$$

where n_d is the echo delay, and a_1 and a_2 are reflection coefficients for the first and second echo. The echo process can be thought of in terms of a system function $H(z) = Y(z)/X(z)$, and the task of echo cancellation is to find the inverse function $G(z) = 1/H(z)$. If all the

parameters in the expression above are known, then this inverse can be found analytically.

In reality one doesn't know the parameters of the echo model, which possibly even change slowly over time. Adaptive filters are therefore used to do the cancellation. Most formulations are beyond the scope of this course, but the LMS algorithm is within reach and is relatively easy to understand. Instead of estimating $H(z)$ and using it to find the $G(z)$ that cancels it, we find $G(z)$ directly.

The echo removal system is modelled as a FIR filter with an unknown impulse response. We estimate the required impulse response by transmitting a known signal $x[n]$ over the channel and observing the corresponding output $y[n]$. (ADSL modems do this during their training stage.) We want a system that takes the echo signal $y[n]$ as input, and outputs the original signal $x[n]$.

The output of the filter at time n is

$$\hat{x}[n] = h[n] * y[n] = \sum_{i=0}^{N} h_i y[n-i],$$

where the notation $h_i = h[i]$ is used for clarity. The desired output is the known transmitted signal value $x[n]$, so the error in the filter output at this instant is $e[n] = x[n] - \hat{x}[n]$. We want to choose the h_i parameters such that the error is minimised.

The squared error is

$$e^2[n] = (x[n] - \hat{x}[n])^2 = (x[n] - \sum_{i=0}^{N} h_i y[n-i])^2,$$

which we can use to work out how h_i should be changed to decrease the error. The derivative of the squared error with regard to the impulse response value h_j is

$$\frac{de^2[n]}{dh_j} = -2\left(x[n] - \sum_{i=0}^{N} h_i y[n-i]\right) y[n-j] = -2e[n]y[n-j].$$

If this derivative is positive, then it means that $e^2[n]$ will become bigger if h_j is increased. We can then decrease the error by reducing h_j.

In general we fix a step size Δ, and for each sample instant we modify each h_j according to

$$h_j \leftarrow h_j + 2\Delta e[n]y[n-j].$$

This is the Least Mean Squares (LMS) algorithm. For sufficiently small Δ and enough training data, the values of h_j will converge to the point where on average $de^2[n]/dh_j = 0$ for each j (i.e. where the squared error is minimised).

Some basic code that uses the LMS algorithm to determine a filter for echo cancellation follows:

```
% Pseudorandom sequence for training
d = 100;  % echo delay
gsig = randn(16000,1);
x = gsig(d+1:end);  % desired echo-free signal
y = gsig(d+1:end) + 0.4*gsig(1:end-d);  % signal with echo

% Train using LMS
M = 3*d;  % FIR filter length
delta = 0.0001;  % LMS training step size
h = zeros(M,1);  % Initial filter IR values
ev = [];
for i=M+1:length(x)
  xn = x(i);  % desired sample value
  yn = y(i-M+1:i);
```



```
  xnhat = h'*yn;  % filter output value
  en = xn - xnhat;  % current error
  h = h + delta*yn*en;  % update

  % Plotting
  ev = [ev abs(en)];
  if rem(i,100)==0  % too slow to plot every iteration
    subplot(2,1,1);  plot(h);  title('Impulse response');
    subplot(2,1,2);  plot(ev);  title('|error|');
    pause(0.001);
  end
end
```

Copy it into a Matlab script, run it, and try to understand it.

Approach

The aim of this project is to explore the problem of echo cancellation. The implementation of the LMS algorithm in the previous section can serve as a starting point, but you can look at alternative methods instead if you prefer.

Some possibilities are:

- Do a theoretical analysis of echo cancellation for known delay and reflection parameters. I briefly discussed an example in class, which can be elaborated upon. In the non-adaptive case one can derive the inverse system analytically.
- Explore the adaptation parameter and the effect it has on rate of convergence.
- Explore the effect of different echo delays on the cancellation process. Since the length of the FIR filter used to cancel the echos

must be substantially longer than the echo delay, the number of filter weights that the LMS algorithm has to estimate increases with increasing delay.

- The LMS formulation does not make use of the knowledge that the signal is corrupted by an echo. It should therefore be possible to use the algorithm as provided for restoring arbitrary channel distortion, as long as it is time invariant. You could explore alternative uses for the cancellation system provided.
- The length of the impulse response required to cancel a long delay is large, so the computational expense becomes high. You could make the computation more manageable by implementing overlap-add or overlap-save FFT-based convolution. Alternatively, I've seen interesting subband approaches which I know little about.