CSE747: Pattern Recognition

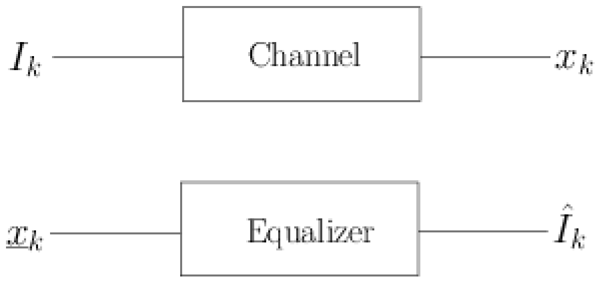
Offline on Channel Equalization

The task of channel equalization is to recover a sequence of transmitted bits at the receiver after they have been distorted by the channel. From the theory of data communication, there are two sources of distortion in the channel: (i) intersymbol interference where an information bit is affected by previously transmitted bits, and (ii) noise, i.e., addition of unwanted signals in the channel. If the transmitted bit is I_k and the received bit is x_k, we can model the effect of channel distortion as follows:

$$x_k = f(I_{k,}\ I_{k-1,}\ \ldots,\ I_{k-(n-1)}) + n_k \quad \ldots\ \ldots\ \ldots\ (1)$$

Here the function $f(.)$ is the channel's impulse response and n_k denotes the added noise. We assume, $f(.)$ as a linear function, i.e., $x_k = \sum_{i=0}^{n-1} h_i I_{k-i} + n_k$ where h_i are constants and depend on the channel. We further assume that noise is normally distributed with zero mean and variance N.

Now, the task of equalizer is to predict the transmitted bit I_k from l successive received bits, $x_{k,}\ x_{k-1,}\ \ldots,\ x_{k-(l-1)} = \underline{x}_k$. We denote the predicted bit with $\hat{I}_k$. We can visualize the full process through the following diagram.



There are several techniques for equalization, i.e., predicting I_k from $\underline{x}_k$ such as cluster based method and using markov chain model. In this assignment you are required to implement that the latter approach, i.e., you will model the equalizer as a 'Markov Chain' using suitable selection of states and then use viterbi algorithm to find the most likely sequence of states that would result in $\underline{x}_k$ and thereby determine the value of $\hat{I}_k$.

Tasks:

1. Build the markov chain model for which we would need the following:
   - Defining the states. Here we define all the possible combination of n bits (n is the number of bits that affects the current received bit) as states at each bit interval.

- Determining the state transition probabilities. Here you need to find out all the possible transitions among states and determine the respective probabilities.
- Determining the observation probabilities. Since we assume the noise is normally distributed, the observation probability should also follow normal distribution. Hence, we need only to find the mean of the distributions since the variance would not be required for our task. (Think yourself why it is so). To determine the mean, we need to go through a training phase where we would transmit a sequence of random bits and then find the means of the received bits associated to each state.

2. After the model is built, we need to test the model i.e., the test phase. Here you will transmit a sequence of k bits, I_k and from the received k bits, x_k you need to estimate the transmitted bit sequence, $\hat{I}_k$ by using viterbi algorithm on your markov chain model. Note that each bit will be transmitted as either +1 or -1 corresponding to its values 1 and 0 respectively.
3. Compare the original signal I and the estimated signal $\hat{I}$ to find out the accuracy.
4. To simulate the channel, you need to define a class whose member will be the channel parameters: (i) the h's of the impulse response, (ii) mean and variance of channel noise. All these parameters are to be taken as input from file. The class will have a function like `transmit` (I_k) which given a bit I_k will return the bit x_k based on equation 1 above. To simulate noise, you need to generate a normal random variable with the given mean and variance.

Requirements:

For building the model, i.e, to determine the prior probability, the transition probabilities, and to determine the cluster means, you will use a given train file "train.txt". It is nothing but a very large bit string, with 0s and 1s appearing randomly. Your program will take as input the channel co-efficients (h_i's), and the variance of the noise introduced in the channel. For example, if there are three co-efficients, then you understand that the transmitted value depends on the previous three bits. So from the train file, you will parse all possible three bit long sub-strings, pass them through the channel and calculate the means of the clusters. In the process, it should come to you naturally how to determine the prior probabilities and the transition probabilities.

For testing, we will provide a file "test.txt", which is a binary string. You should pass the corresponding values through the channel, and then from the transmitted values (that has their own errors), you should determine what was sent. Write the inferred bit-string to a file named "out.txt". Obviously, if you implement correctly, out.txt and test.txt should be the same.

In sessional, we will test with various files as we like, your implementation should work just fine.

Submission deadline

Monday, 11th Week.