ELECTRICAL AND ELECTRONICS ENGINEERING INSTITUTE

EE 286: Digital Audio Signal Processing

Exercise 4: Linear Predictive Coding

We can model the human speech system using a source-filter model. The vocal chords act as an oscillator. The mouth cavity, tongue and throat act as filters. We can control a tonal sound, e.g. 'oooh' vs 'aaah'. We can whiten the signal, e.g. 'ssss' and 'ssshhh'.

Linear predictive coding (LPC) is a source-filter analysys-synthesis methodology that approximates sound generation as an excitation (by a pulse train or noise) passing through an all-pole resonant filter. It is extensively used in speech applications. It reduces the amount of data to just a few filter coefficients. Its name is derived from the fact that the output samples are predicted as a linear combination of filter coefficients and previous samples. The difference between the true value and the predicted value is called the residual error.

Speech synthesis model

$$\begin{tikzcd}
\text{vocal cords} \arrow[r] & \boxed{\text{vocal tract}} \arrow[r] & \text{speech} \\
e(n) \arrow[r] & \boxed{H(z)} \arrow[r] & y(n)
\end{tikzcd}$$

where $e(n)$ is the residual error (pulse train produced by the vocal chords), $y(n)$ is the speech signal, and $H(z)$ is an all-pole IIR filter.

$$H(z) = \frac{1}{1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_p z^{-p}}$$

Given a speech signal $y(n)$, the problem is to determine the LPC coefficents a_k's and the residual error $e(n)$.

Procedure:

1. Record the phrase "the quick brown fox jumps over the lazy dog." Use a sampling rate of 8 kHz. Segment the sampled speech into short frames, 30 msec.
2. For each frame, compute the p'th order LPC coefficients. To compute the LPC coefficients,
   a. Let $b(n)$ denote the current frame.
   b. Compute the autocorrelation $r(n)$ of the frame $b(n)$, you may use the `xcorr` function that returns $r(n)$ for $-(M-1) \le n \le (M-1)$ where M is the length of the frame. We need only the values of $r(n)$ for $0 \le n \le p$.
   c. Solve the matrix equation below to determine the LPC coefficients. You may use `toeplitz` function to form matrix R. For a 4th order filter:

$$\begin{bmatrix} r(0) & r(1) & r(2) & r(3) \\ r(1) & r(0) & r(1) & r(2) \\ r(2) & r(1) & r(0) & r(1) \\ r(3) & r(2) & r(1) & r(0) \end{bmatrix} \begin{bmatrix} a(1) \\ a(2) \\ a(3) \\ a(4) \end{bmatrix} = - \begin{bmatrix} r(1) \\ r(2) \\ r(3) \\ r(4) \end{bmatrix}$$

   To verify if your coefficients are correct, use the function `aryule` on the frame $b(n)$.
3. To determine the residual error for each frame, filter the current frame $b(n)$ with the LPC coefficients, e.g. `error_frame=filter(a,1,b)`.
4. Store the LPC coefficients as a p x N (N is the number of frames) matrix, and residual error as a 1-dimensional vector. Moreover, save the residual error as a WAV file.
5. Listen to the residual error. How would you describe the inelligibility signal?

6. Synthesize the speech signal using the computed LPC coefficients and residual error. Save the reconstructed speech as a WAV file. Were you able to recover the original speech?

Questions:

1. Choose the filter order p = 4, 8 and 16. What is the effect of increasing the order of the filter on the reconstructed speech? Can you explain the observations?
2. Increase the frame size to 100 msec. What is the effect of increasing the frame size on the reconstructed speech? Can you explain the observations?
3. Suppose you quantize the residual error with four fixed quantization levels. Reconstruct the speech using this quantized residual error. Listen to the reconstructed speech. Repeat this with eight quantization levels. What is the effect of increasing the quantization levels on the reconstructed speech? Can you explain the observations?

Write your conclusions regarding the effects of different parameters on the performance of LPC on speech signals. What are the trade-offs in maintaining good intelligibility of reconstructed speech?

You can work as a pair. Together with your answers to the questions, submit the Matlab script yoursurname1_ yoursurname1_LPC.m. Compress your files as a zip file then upload in UVLE.