EE 473 - *Introduction to Digital Signal Processing*

Final Project

Project Name:

LPC Vocoder Implementation in Python

Project Members:

Sezai Burak Kantarcı - 2020701087

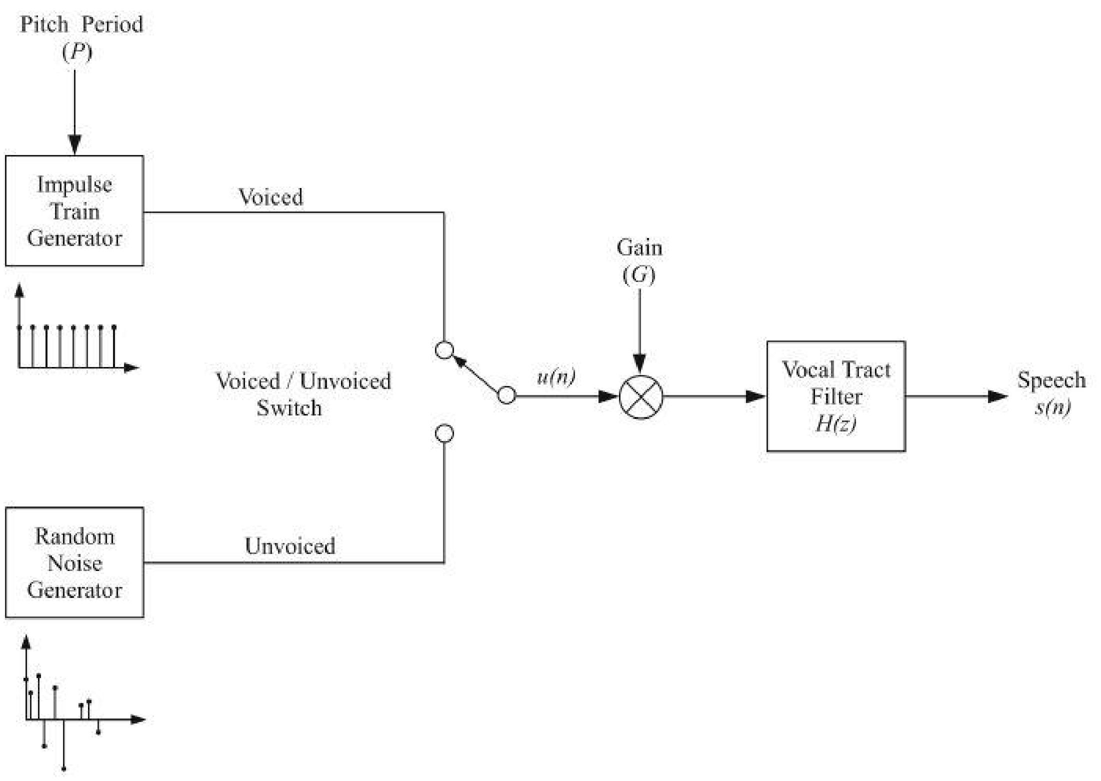

Figure 1: Speech Production Model for LPC [1].

1. Introduction

The development of a parametric model for signal behavior opens up the possibility of using the model for different applications, such as prediction or forecasting, control, and data compression [5].

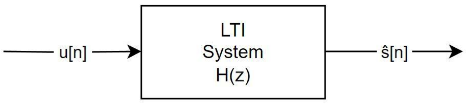

Figure 2: Linear System Model for a signal s[n].

Among the most powerful models in use today, where a signal s_n is thought to be the output of a system with some unknown input u_n such that:

$$s_n = -\sum_{k=1}^{p} a_k s_{n-k} + \sum_{l=0}^{q} b_l u_{n-l} \quad \textbf{(1)}$$

where a_k, $1 \leq k \leq p$ and b_l, $1 \leq l \leq q$ for the hypothesized system.

Equation (1) simply states that the output is a linear function of past outputs and present and past inputs. In other words; s_n is predictable from linear combinations of past outputs and inputs. This is called "***Linear Prediction***".

By taking the Z Transform on both sides with input specified, Equation (1) can be rewritten as:

$$H(z) = \frac{\sum_{k=0}^{q} b_k Z^{-k}}{1 - \sum_{k=1}^{p} a_k Z^{-k}} \quad \textbf{(2)}$$

As a result, the signal is modeled by the value of a_k's and b_k's or by the poles and zeros of $H(z)$, along with knowledge of input [4].

2. Linear Predictive Coding

In Linear Predictive Coding (LPC) -Linear Predictive Analysis when used in the context of speech processing- given a speech sample at time n, $s(n)$ can be approximated as the past speech samples as:

$$s(n) = \sum_{k=1}^{p} a_k s(n-k) + Gu(n)$$

- (3)

In Equation (3), a_k's are linear predictive coefficients, p is the order of the LP filter, $u(n)$ is the normalized excitation and G is the gain of the excitation [2].

Generally, the prediction order is chosen using the relation:

$$p = 2\,x\,(BW + 1) \quad \textbf{(4)}$$

where BW is the bandwidth of speech in kHz.

LPC process can be splitted into two stages, analysis and the synthesis.

2.1. LPC Analysis Filter

As an FIR filter, relation between error signal and the input speech signal for LPC Analysis Filter can be written as:

$$e(n) = s(n) + \sum_{k=1}^{p} a_k s(n-k) \quad \textbf{(5)}$$

$$E(z) = A(z)X(z)\,, \qquad \textbf{(6)}$$

where:

$$A(z) = 1 - \sum_{k=1}^{p} a_k Z^{-k} \qquad \textbf{(7)}$$

2.2. LPC Synthesis Filter

As an IIR Filter, the relation between constructed speech signal and the error signal:

$$s(n) = e(n) + \sum_{k=1}^{p} a_k s(n-k) \qquad \textbf{(8)}$$

$$S(z) = H(z)E(z)\,, \qquad \textbf{(9)}$$

where:

$$H(z) = \frac{1}{A(z)} = \frac{1}{1-\sum_{k=1}^{p} a_k s(n-k)} \qquad \textbf{(10)}$$

This system can be seen on Figure 3.

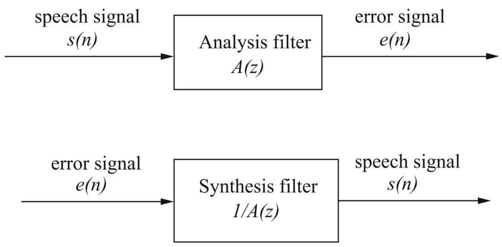

Figure 3: Linear Predictive Coding System [1].

Autocorrelation and Covariance methods can be used to estimate the LPC coefficients (a_k's) such that the energy of the error signal $e(n)$ is minimized.

3. Pitch and Decision

A pitch mark is an energy peak in a speech signal that occurs for a short period of time. Pitch is produced by the vibration of the vocal cord during a speech, and the duration between the peaks is simply called a pitch [3].

In speech, each sample is highly predictable from previous samples. Pitch pulses are generally the only major exception to this predictability [7].

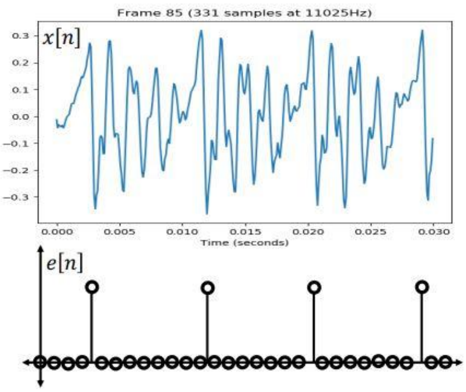

Figure 4: Pitches from a sampled speech and the corresponding error [6].

So the error signal $e(n)$ is a **white noise** for unvoiced speech, and a **pulse train** for voiced speech.

At the end of Section 2, the main principle behind LPC was given. To find coefficients that minimizes error; Autocorrelation is almost exclusively used for speech due to its computational efficiency and inherent stability, while the covariance method does not ensure the stability of the all-pole LP synthesis filter [1].

First, the speech signal is multiplied by a window to get a speech segment. Hamming Window is a commonly used example.

$$s_w(n) = s(n)w(n) \quad \textbf{(11)}$$

Energy of the signal $e(n)$ that is tried to be minimized can be written as:

$$Ep = \sum_{n=-\infty}^{\infty} e^2(n) \quad \textbf{(12)}$$

$$= \sum_{n=-\infty}^{\infty} (s_w(n) + \sum_{k=1}^{p} a_k s_w(n - k))$$

If we take the partial derivatives of Ep with respect to LPC coefficients:

$$\frac{\partial Ep}{\partial a_k} = 0 \quad , \quad 1 \le k \le p \quad \textbf{(13)}$$

which results in p linear equations for p unknown parameters $(a_1, a_2, \ldots, a_p)$:

$$\sum_{k=1}^{p} a_k \sum_{n=-\infty}^{\infty} s_w(n - i)\, s_w(n - k) =$$

$$- \sum_{n=-\infty}^{\infty} s_w(n - i)\, s_w(n)$$

where $\quad , 1 \le i \le p$ **(14)**

These equations can be expressed as autocorrelation function.

$$r_{xx}[j] = \sum_{k=1}^{p} a_k r_{xx}[k - j]$$

where $\quad , 1 \le j \le p$ **(15)**

As the autocorrelation would be the maximum between the neighbor windows from the speech, pitch period can be found after finding autocorrelation, as:

$$P = argmax\, r_{xx}(m)$$

where $\quad P_{min} \le m \le P_{max}$ **(16)**

P_{min} corresponds to a high woman's pitch, around $\frac{Fs}{P_{min}} = 250\ Hz$ and P_{max} corresponds to low man's pitch, around $\frac{Fs}{P_{min}} = 80\ Hz$ [6] .

3.1 Voiced / Unvoiced Decision

In voiced speech and unvoiced speech, windowed speech signals autocorrelation can be given as:

$$x(n)\ voiced: \ r_{xx}(P) \sim r_{xx}(0)$$

$$x(n)\ unvoiced: \ r_{xx}(n) << \delta(n)$$

$$: \ r_{xx}(P) << r_{xx}(0)$$

(17)

So, a voiced/unvoiced decision can be made with:

$\frac{r_{xx}(P)}{r_{xx}(0)} \ge Threshold$ -> Frame is voiced. **(18)**

$\frac{r_{xx}(P)}{r_{xx}(0)} < Threshold$ -> Frame is unvoiced. **(19)**

Practice shows that threshold = 0.25 works reasonably well with speech signals [1].

4. Methodology and Implementation

LPC Transmitter and Receiver block diagrams are given in Figure 5 and Figure 6 respectively.

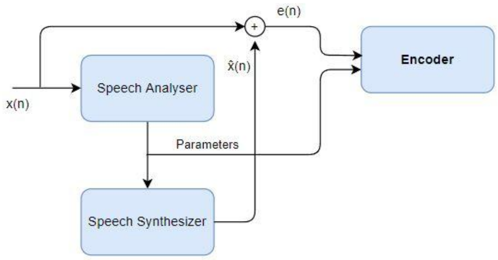

Figure 5: LPC Transmitter.

In the transmitter, sampled input speech signal is passed into an analyzer, which determines the parameters. Synthesizer reconstructs the approximated version of the sampled signal. This reconstructed signal is then compared with the original speech and error signal is calculated [8].

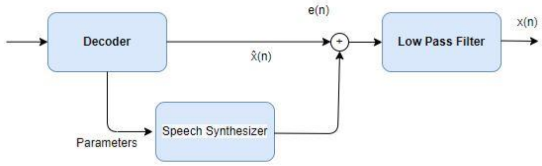

Figure 6: LPC Receiver.

In receiver, LPC coefficients are seperated with the error signal. With the parameters, the synthesizer reconstructs an approximated signal; combined with the error signal to find the original sample signal x(n).

5. Pipeline

The algorithm uses "numpy" and "scipy" packages with their built-in functions. Python language with Anaconda Environment was used as it's ease of use and rich community.

The pseudocode can be given as down below:

<load the speech signal into workspace>
<assign original signal to a modulator variable>
<calculate the carrier as a pulse train>
<reconstruct the signal with the modulator and carrier, using lpc function>
<write all of the sound files>
<end of main>

In order to find LPC coefficients and make the autocorrelation calculation, Levinson-Durbin Recursion was used. Also to make the shift at each sample interchange between Linear Predictive Coding and Line Spectral Pairs was also used.

The source code is tried to be explained at each line with comment lines.

Thank you for a wonderful semester.
Sincerely.

6. References

[1] Linear Prediction Analysis of Speech - Springer - Appendix A - [Link]()
[2] Saranya, A. & Natarajan, Sripriya. (2011). LPC VOCODER using instants of significant excitation and pole focusing. 10.1007/978-3-642-24037-9_18.
[3] T. F. Quatieri, Discrete-Time Speech Signal Processing Principles and Practice, 1st ed., Upper-Saddle River, N. J., 2002.
[4] Oppenheim, Alan V., Schafer, Ronald W. and Buck, John R.. Discrete-Time Signal

Processing. Second : Prentice-hall Englewood Cliffs, 1999.
[5] J. Makhoul, "Linear prediction: A tutorial review," in Proceedings of the IEEE, vol. 63, no. 4, pp. 561-580, April 1975, doi: 10.1109/PROC.1975.9792.
[6] ECE 401 - Lecture 17: LPC speech synthesis and autocorrelation based pitch tracking link
[7] EE.473 - Bogazici University- Prof. Dr. Levent Arslan - Lecture Notes
[8] Linear predictive coding - link