Linear Prediction of Speech

Yu Cang

Shanghai Jiao Tong University, China

Abstract

A model was established for predicting a speech linearly. Two strategies are developed to determine the coefficients. One is windowing the error and the Cholesky decomposition is applied. The other is windowing the signal and the Topelitz equations are solved iteratively. The two strategies are compared using a sample speech finally.

Keywords: Speech, Prediction, Cholesky, Topelitz

1. Basic Equations

In a simplified situation, a speech can be linearly predicted from the previous p samples as

$$\hat{x}(n) = \sum_{i=1}^{p} a_i x(n-i) \tag{1}$$

where a_i are the linear prediction coefficients. Then the error between the signal $x(n)$ and the predicted value $\hat{x}(n)$ is given as

$$e(n) = x(n) - \hat{x}(n) = -\sum_{i=0}^{p} a_i x(n-i) \tag{2}$$

where $a_0 = -1$. The minimum mean square error(MMSE) is adopted as the principle to determine these coefficients a_i.

The square error of the prediction is defined as

$$E = \sum_n e^2(n) = \sum_n [x(n) - \sum_{i=1}^{p} a_i x(n-i)]^2 \tag{3}$$

To minimize E, each coefficient a_i $(i = 1, 2, ..., p)$ is determined as

$$\frac{\partial E}{\partial a_i} = 0 \tag{4}$$

Which is equivalent to

$$\sum_{j=1}^{p} a_j \sum_n x(n-j)x(n-i) = \sum_n x(n)x(n-i) \tag{5}$$

where $i = 1, 2, ..., p$.

Denote $\phi(i, j)$ as

$$\phi(i, j) = \sum_n x(n-i)x(n-j) \tag{6}$$

it's clear that $\phi(i, j) = \phi(j, i)$, and (5) can be written as

$$\sum_{j=1}^{p} \phi(j, i)a_j = \phi(0, i) \tag{7}$$

where $i = 1, 2, ..., p$.

Hence, it is left to determine $\phi(j, i)$ and then a_j can be resolved. However, there're different ways to determine the bounds of n when calculating $\phi(j, i)$, which led to different strategies of linear prediction.

2. The Autocorrelation Method

The autocorrelation method aims to minimize the error over the whole timespan, and it's assumed that $x(n)$ is 0 when $n \notin [0, N-1]$. Thus, $x(n)$ is windowed with finite length, and the autocorrelation function of $x(n)$ is defined as

$$r(j) = \sum_{n=-\infty}^{+\infty} x(n)x(n-j) \quad (j \in [1, p]) \tag{8}$$

It can be concluded that, from (6), $r(|j-i|) = \phi(j, i)$. Thus, (5) can be expressed as

$$\begin{bmatrix} r(0) & r(1) & r(2) & ... & r(p-1) \\ r(1) & r(0) & r(1) & ... & r(p-2) \\ r(2) & r(1) & r(0) & ... & r(p-3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r(p-1) & r(p-2) & r(p-3) & ... & r(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} r(1) \\ r(2) \\ r(3) \\ \vdots \\ r(p) \end{bmatrix} \tag{9}$$

It can be observed that the coefficient matrix is the so called Toeplitz matrix, where elements are symmetry and each descending diagonal from left to right is constant.

Suppose $a_i^{(k)}$ $(i = 1, 2, ..., k)$ is the solution for $k-th$ iteration for $p = k$, which implies

$$\begin{bmatrix} r(0) & r(1) & r(2) & ... & r(k-1) \\ r(1) & r(0) & r(1) & ... & r(k-2) \\ r(2) & r(1) & r(0) & ... & r(k-3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r(k-1) & r(k-2) & r(k-3) & ... & r(0) \end{bmatrix} \begin{bmatrix} a_1^{(k)} \\ a_2^{(k)} \\ a_3^{(k)} \\ \vdots \\ a_k^{(k)} \end{bmatrix} = \begin{bmatrix} r(1) \\ r(2) \\ r(3) \\ \vdots \\ r(k) \end{bmatrix} \tag{10}$$

Thus, when $p = k + 1$, two equation sets can be constructed as

$$\begin{bmatrix} r(0) & r(1) & ... & r(k) \\ r(1) & r(0) & ... & r(k-1) \\ r(2) & r(1) & ... & r(k-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(k-1) & r(k-2) & ... & r(1) \\ r(k) & r(k-1) & ... & r(0) \end{bmatrix} \begin{bmatrix} a_1^{(k)} \\ a_2^{(k)} \\ a_3^{(k)} \\ \vdots \\ a_k^{(k)} \\ -\lambda \end{bmatrix} = \begin{bmatrix} r(1) - \lambda r(k) \\ r(2) - \lambda r(k-1) \\ r(3) - \lambda r(k-2) \\ \vdots \\ r(k) - \lambda r(1) \\ \sum_{i=1}^{k} a_i^{(k)} r(k+1-i) - \lambda r(0) \end{bmatrix} \tag{11}$$

and

$$\begin{bmatrix} r(0) & r(1) & ... & r(k) \\ r(1) & r(0) & ... & r(k-1) \\ r(2) & r(1) & ... & r(k-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(k-1) & r(k-2) & ... & r(1) \\ r(k) & r(k-1) & ... & r(0) \end{bmatrix} \begin{bmatrix} \lambda a_k^{(k)} \\ \lambda a_{k-1}^{(k)} \\ \lambda a_{k-2}^{(k)} \\ \vdots \\ \lambda a_1^{(k)} \\ 0 \end{bmatrix} = \begin{bmatrix} \lambda r(k) \\ \lambda r(k-1) \\ \lambda r(k-2) \\ \vdots \\ \lambda r(1) \\ \lambda \sum_{i=1}^{k} a_i^{(k)} r(i) \end{bmatrix} \tag{12}$$

Let λ satisfy

$$\sum_{i=1}^{k} a_i^{(k)} r(k+1-i) - \lambda r(0) + \lambda \sum_{i=1}^{k} a_i^{(k)} r(i) = r(k+1) \tag{13}$$

namely

$$\lambda = \frac{r(k+1) - \sum_{i=1}^{k} a_i^{(k)} r(k+1-i)}{\sum_{i=0}^{k} a_i^{(k)} r(i)} \tag{14}$$

where $a_0^{(k)} = -1$. Then $a_i^{(k+1)}$ can be given as

$$\{a_i^{(k+1)}\} = \begin{bmatrix} a_1^{(k)} + \lambda a_k^{(k)} & a_2^{(k)} + \lambda a_{k-1}^{(k)} & ... & a_k^{(k)} + \lambda a_1^{(k)} & -\lambda \end{bmatrix}^T \tag{15}$$

And it can be easily verified that the recursion formula is also valid when $k = 1, 2$, thus this solution is justified by induction.

Further, this is the so-called Levinson-Durbin algorithm, with time complexity $O(n^2)$, and it is much faster than solving it directly where time complexity is $O(n^3)$.

3. The Covariance Method

In constrast to the previous method, the covariance method doesn't windowed the signal. Instead, the amount of points used to calculate the error is fixed. And it aims to minimize the error.

Suppose the amount of points to calculate $r(j)$ is N, then by definition

$$r(j) = \sum_{n=0}^{N-1} x(n)x(n-j) \quad (j = 0, 1, ..., p) \tag{16}$$

Thus, $N + p$ samples are needed for calculating all $r(j)$. And $r(j-i)$ can be expressed as

$$r(j-i) = \sum_{n=0}^{N-1} x(n-j)x(n-i) \triangleq c(j,i) \quad (j = 0, 1, ..., p) \tag{17}$$

Thus, the prediction equation (5) can be written as

$$\begin{bmatrix} c(1,1) & c(1,2) & ... & c(1,p) \\ c(2,1) & c(2,2) & ... & c(2,p) \\ \vdots & \vdots & \ddots & \vdots \\ c(p,1) & c(p,2) & ... & c(p,p) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} c(1,0) \\ c(2,0) \\ \vdots \\ c(p,0) \end{bmatrix} \tag{18}$$

Since $c(i,j) \neq c(i+k, j+k)$, the coefficient matrix above is no longer the Toeplitz matrix, and different scheme should be employed to solve this linear system.

It's clear that $c(i,j) = c(j,i)$, so it can be solved using the Cholesky decomposition. And the coefficient matrix can be decomposed as

$$C = LL^T \tag{19}$$

where L is a lower triangle matrix. Onece L is obtained, targeting a_i can be easily calculated by

$$\begin{aligned} y &= linsolve(L, b) \\ a_i &= linsolve(L^T, y) \end{aligned} \tag{20}$$

4. Numerical Experiment

A segment of audio is taken as the original signal, where $len = 120$ samples are included.

4.1. Comparison between the autocorrelation and covariance

Numerical experiments indicates that the covariance method performs better than the autocorrelation method when p is fixed($p = 6$).

The autocorrelation method is examined in Fig(1). It can be seen from the left figure that the predicted signals fit the original signals well, only with slightly difference. The error between is much more apparent at peak area. Details of performance is given in the right figure.

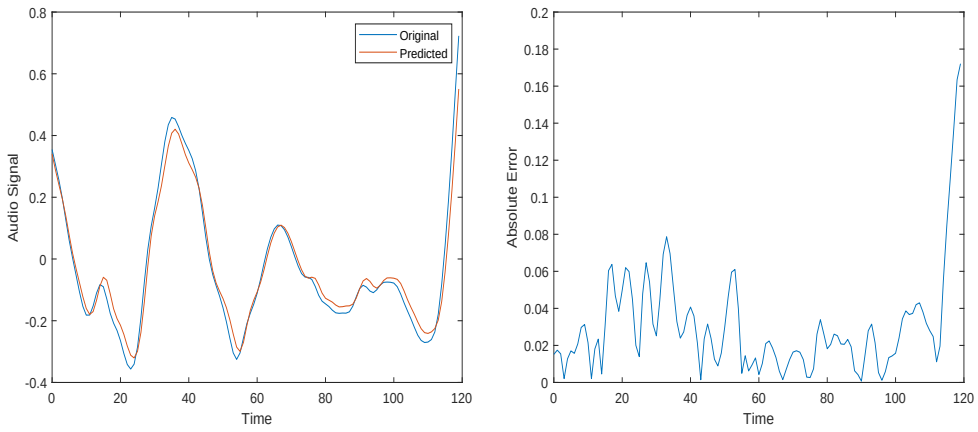

Figure 1: Performance of the Autocorrelation Method

The covariance method is investigated in Fig(2). It can be seen from the left figure that the predicted signals meet the original signals very well and the error between is significantly eliminated compared to the autocorrelation method. It can be seen from the right figure that the error is lower by an order of magnitude.

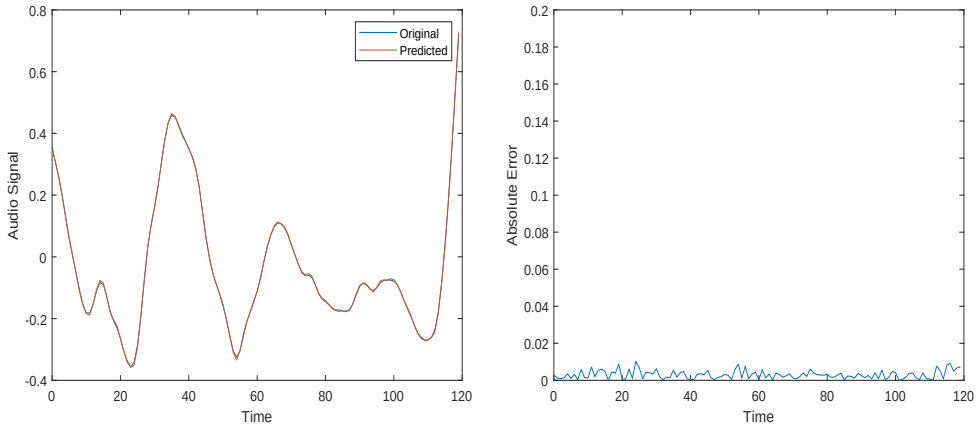

Figure 2: Performance of the Covariance Method

The difference can be explained as the bounds of n are different. There're p more points used in the covariance method when calculating the linear prediction coefficients, and hence, it's more likely that the covariance method out-performs the autocorrelation method.

4.2. Relation between the error and p