Adaptive Filters

Tutorial 2

Prof. Dr.-Ing. H. Puder
Adaptive Systems for Processing of Speech and Audio Signals
Graduate School Computational Engineering

Problem 1 Linear Prediction Filter

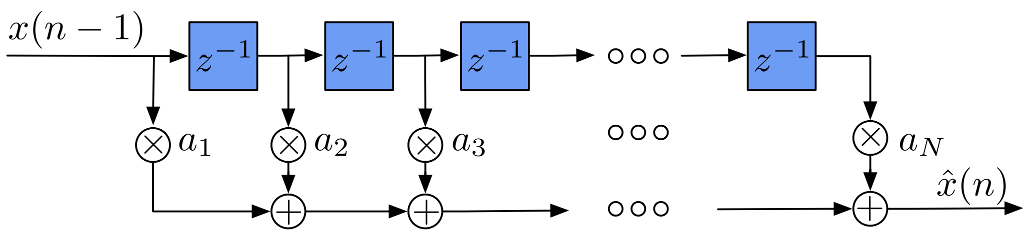

Figure 1: Linear prediction filter.

Consider a linear prediction filter as depicted in Figure 1, where $\hat{x}(n)$ is the estimate of $x(n)$, a_i are the prediction coefficients and N is the length or the order of the prediction filter, respectively. The estimate $\hat{x}(n)$ is given by

$$\hat{x}(n) = \sum_{i=1}^{N} a_i \cdot x(n-i).$$

The optimal values for the prediction coefficients a_i are obtained using the Yule-Walker equation, hence

$$\boldsymbol{a} = \begin{bmatrix} a_1 & a_2 & \cdots & a_N \end{bmatrix}^{\mathrm{T}} = \boldsymbol{R}_{xx}^{-1} \boldsymbol{r}_{xx}(1),$$

where $\mathbf{r}_{xx}$ denotes the auto-correlation vector of the signal $x(n)$ and $\mathbf{R}_{xx}$ the corresponding auto-correlation matrix.

a) Write a function `calculate_pred_coeffs`, which calculates, for a given auto-correlation function (ACF), the prediction coefficients up to the order N. Use the Levinson-Durbin recursion and the order N as stop criterion. Your function should have r_{xx} and N as inputs and should return the prediction coefficients $\mathbf{a}$ and the minimum error signal power E_{min}.

b) You are provided four vectors `r_xx1.mat` to `r_xx4.mat`, containing four different randomly created ACFs. You can load data stored in a `.mat` file to a variable `r_xx` in your workspace, for example by using `r_xx = importdata('r_xx1.mat')`. All given ACFs are of same length $L = 50$ and contain maximum $M = 20$ non-zero elements.

   Select one of the given ACFs and pass it to your self-written function `calculate_pred_coeffs` of a) and plot the prediction coefficients for the order $N = 10$, $N = 30$ and $N = 50$. Also plot E_{min} in dependence of the number of recursion steps for the three different orders. What do you observe?

c) The prediction error is given by

$$\mathbf{e}(n) = x(n) - \hat{x}(n) = x(n) - \sum_{i=1}^{N} a_i \cdot x(n-i)$$
$$= \sum_{i=0}^{N} h_{pef}(i) \cdot x(n-i),$$

with the prediction error filter (pef) coefficients

$$\mathbf{h_{pef}} = \begin{bmatrix} 1 & -a_1 & -a_2 & \cdots & -a_N \end{bmatrix}^{\mathrm{T}}.$$

Write a function `calculate_acf_pred_error` to calculate the ACF $r_{ee}(l)$ of the prediction error in dependence of the ACF $r_{xx}(l)$ and the prediction filter coefficients $\mathbf{a}$. Use a vector notation for your calculations. Hint: How many values of $r_{ee}(l)$ can be non-zero?

d) Use all the previous functions to write a main function `Tut2_ex1_main` with the ACF $r_{xx}(l)$ and the order N as input values. The function should calculate and return the prediction coefficient vector $\mathbf{a}$, the ACF of the error signal $r_{ee}(l)$ and the minimum error signal power E_{min}. Load the same ACF chosen in b). Then use your main function with the different orders $N = 10$, $N = 20$ and $N = 50$ and plot $r_{xx}(l)$ and $r_{ee}(l)$ in one figure for all three cases. Comment on the results.

Problem 2 Prediction Error Filter

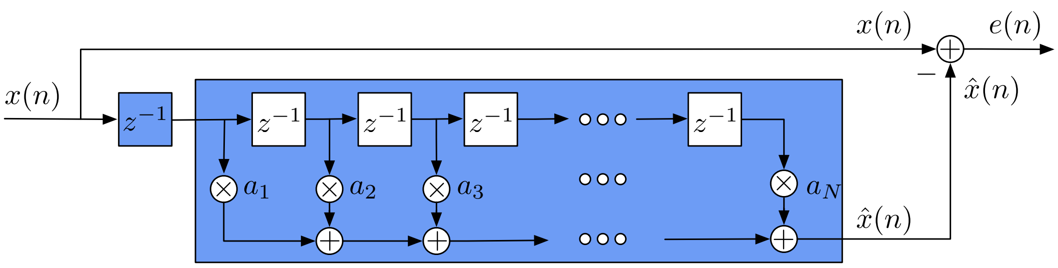

Figure 2: Prediction error filter.

Now we consider a prediction error filter as in Figure 2 and we use a speech signal as input.

a) Load the file `f_1.fol` by using the MATLAB function `load`. The file contains a speech signal with a sampling frequency of 8 kHz. Plot the signal and listen to it. Also estimate the signal power $E\{x^2(n)\} = r_{xx}(0)$.

b) Write a function `pred_error_filter` which uses a signal $x(n)$ and a prediction order N as inputs and calculates the prediction error signal $e(n)$ as output. Your function should calculate the prediction coefficients based on the ACF for the complete signal $x(n)$.
Hint: To calculate the auto-correlation function of a signal, you can use the Matlab function `xcorr`. Also use your previous function `calculate_pred_coeffs`.

c) Use the speech signal f_1 from a) and the function `pred_error_filter` of b) to calculate the error signal and the prediction error gain

$$\frac{E\{x^2(n)\}}{E\{e^2(n)\}} = \frac{r_{xx}(0)}{r_{ee}(0)}.$$

for different prediction orders 1, 2, 4, 8, 16, 32 and 64. Plot the gain values in dB for the different orders. Comment the results. Which are the advantages and the drawbacks of high orders?