telecommunications &
signal processing
laboratory

Minimum Mean-Square Error Filtering: Autocorrelation/Covariance, General Delays, and Multirate Systems

Peter Kabal

Department of Electrical & Computer Engineering
McGill University
Montreal, Canada

Version 0.985 April 2011

2011/04/27

Revision History:

- 2011-04 v0.985: Updated and expanded, generally available on the web
- 2011-03 v0.97: Updated and expanded
- 2011-02 v0.95: Updated and expanded
- 2010-03 v0.83: Initial version

Contents

List of Figures

Part I

Mean-Square Filtering: Theory

Chapter 1

Introduction

These notes outline procedures for solving minimum mean-square error filtering problems. The goal is to find filters which minimize a squared error criterion. The focus will be on discrete-time signals and systems, but in some cases mixed systems such as data transmission systems will entail consideration of both continuous-time and discrete-time signals.

In these notes, the problem is discussed in more generality than in many other expositions; specifically we allow for general filter delays (to accommodate the pitch filtering problem, for instance) and cover both the stochastic case and block-based analyses with a single formalism.

For mean-square error computations, we will only need to use at most second order statistical properties (correlations and means). For the case of a stochastic signals, these notes look at the derivation of the correlation values required for a minimum mean-square error solution. We also examine systems which involve cyclostationary signals (interpolation filters, for instance).

The important linear prediction problem is examined in detail. This includes the setup for non-equally spaced delay values. For the equally spaced delay case, we can develop a rich set of results.

For the least-squares problem, these notes give a generalized view of windowing: windowing the data and/or windowing the error. This view subsumes the traditional special cases, viz. the autocorrelation and covariance methods.

These notes present a number of examples based on "real" signals. With the background developed, the results are obtained with relatively straightforward MATLAB scripts. The results illustrate the useful insights that can be obtained when minimum mean-square error theory is appropriately fleshed out.

These notes discuss a number of topics which are not commonly discussed in textbooks.

- Affine estimation addresses the estimation of signals with non-zero means. Taking the mean into account improves the performance of predictive coding of non-zero mean signals.

- The use of general delays in the filter structure allows a unified analysis for backward predictors and long-term (pitch) predictors.
- The analysis of cyclostationary systems allow for the design of systems which can be modelled as multirate systems. Using this formalism, one can design filters which implement fractional sample delays or sample rate conversion.
- Results are given which show that certain multirate systems involving both interpolation and downsampling can be modelled using a polyphase decomposition. The model operates at a single sampling rate and is time invariant. This approach is to design flexible fractional non-uniformly spaced equalizers for data transmission.
- The notes show that a joint process estimation is related to the Levinson algorithm.
- A least squares formulation which uses both a data and an error window is a generalization which subsumes the conventional autocorrelation or covariance approaches for prediction.
- A method to approximately impose linear constraints on a mean-square error filter.

These notes have been used as part of an advanced course in signal processing. The course consisted of the material covered here, followed by material on adaptive signal processing (LMS filters, for example), and material on power spectral estimation.

Chapter 2

Minimum Mean-Square Error Filtering

Consider a filter with an input $x[n]$ and an output $y[n]$ given by

$$y[n] = \sum_{k=0}^{M-1} w_k^* x[n - D_k], \tag{2.1}$$

where the w_k^* values[1] weight the samples of the input signal at different delays D_k. We require that the delays be distinct.

A conventional causal FIR filter would simply have $D_k = k$ for $k = 0, \ldots, M-1$. We will keep the delays general until later in this document, when it becomes useful to specialize them to get further results. The goal is to find a set of filter coefficients that minimize the squared error between the output of the filter $y[n]$ and a desired signal $d[n]$.

First we write the filtering operation in vector form,

$$y[n] = \mathbf{w}^H \mathbf{x}[n], \tag{2.2}$$

where

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{M-1} \end{bmatrix}, \qquad \mathbf{x}[n] = \begin{bmatrix} x[n - D_0] \\ x[n - D_1] \\ \vdots \\ x[n - D_{M-1}] \end{bmatrix}. \tag{2.3}$$

[1]The use of a conjugate on the filter weights will result in a simpler notation for the final set of equations for the minimum mean-square error problem.

The error is

$$\begin{aligned} e[n] &= d[n] - y[n] \\ &= d[n] - \mathbf{w}^H \mathbf{x}[n]. \end{aligned} \tag{2.4}$$

A block diagram of the system under consideration is shown in Fig. 2.1.

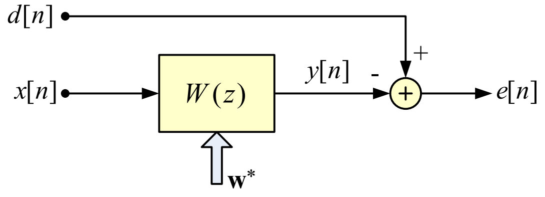


Fig. 2.1 Block diagram for a filtering system

In this chapter, to be able to handle both the stochastic case and block-based squared-error cases with a single formulation, we define an “averaging” operation with a bar over an expression. For the case of ensemble averaging, this is the expectation operator. For other cases, it will signify a sum of squares. In many of the cases we study, the averaging operation will remove the dependency on n. For the case of wide-sense stationary processes, results are reviewed in Appendix B.

Using the averaging notation, the average squared error is

$$\begin{aligned} \varepsilon &= \overline{|e[n]|^2} \\ &= \overline{|d[n]|^2} - \mathbf{w}^H \overline{d^*[n]\mathbf{x}[n]} - \overline{d[n]\mathbf{x}^H[n]}\mathbf{w} + \mathbf{w}^H \overline{\mathbf{x}[n]\mathbf{x}^H[n]}\mathbf{w} \\ &= \overline{|d[n]|^2} - 2\mathrm{Re}[\mathbf{w}^H \mathbf{r}_{\mathbf{x}d}] + \mathbf{w}^H \mathbf{R}_{\mathbf{xx}} \mathbf{w}, \end{aligned} \tag{2.5}$$

with the cross-correlation vector and the correlation matrix defined as, respectively,

$$\mathbf{r}_{\mathbf{u}v} = \overline{\mathbf{u}v^*}, \qquad \mathbf{R}_{\mathbf{uv}} = \overline{\mathbf{u}\mathbf{v}^H}. \tag{2.6}$$

When the signals $\mathbf{u}$ and $\mathbf{v}$ are the same, say equal to $\mathbf{x}$, we get the autocorrelation matrix $\mathbf{R}_{\mathbf{xx}}$.

The ith element of $\mathbf{r}_{\mathbf{u}v}$ is the cross-correlation value $\overline{u_i v^*}$, where u_i is the ith element of $\mathbf{u}$. If $\mathbf{u}$ is a function of time (say $\mathbf{u}[n]$) and/or v is a function of time (say $v[n]$), the time average or ensemble average of the product is over the time index. The i,jth element of $\mathbf{R}_{\mathbf{uv}}$ is the cross-correlation value $\overline{u_i v_j^*}$, where u_i is the ith element of $\mathbf{u}$ and v_j is the jth element of $\mathbf{v}$. If $\mathbf{u}$ is a function of time (say $\mathbf{u}[n]$) and/or $\mathbf{v}$ is a function of time (say $\mathbf{v}[n]$), the time average or ensemble average of the product is over the time index.

Sometimes a distinction is made between *prediction*, *filtering*, and *estimation*. In prediction, the output of the filter depends on only past input values, i.e., in our notation $D_k > 0$. In filtering, the filter is causal; the output depends only on the present and past inputs, $D_k \geq 0$. Estimation includes the case that $D_k < 0$ for some of the delays. Our examples will use causal filters or predictors. However, we will sometimes insert a delay in the desired signal path. In that case, the filter will perform estimation with respect to the delayed desired signal, i.e., the filter will act on input signal values that lead the delayed desired signal.

2.1 Properties of the Autocorrelation Matrix

1. $\mathbf{R}_{\mathbf{xx}}$ is Hermitian symmetric,

$$\begin{aligned} \mathbf{R}_{\mathbf{xx}} &= \overline{\mathbf{x}[n]\mathbf{x}^H[n]} \\ &= \mathbf{R}_{\mathbf{xx}}^H. \end{aligned} \tag{2.7}$$

2. $\mathbf{R}_{\mathbf{xx}}$ is is non-negative definite. One form of the test for non-negativity is to evaluate a quadratic form form for an arbitrary non-zero vector $\mathbf{b}$,

$$\begin{aligned} \mathbf{b}^H\mathbf{R}_{\mathbf{xx}}\mathbf{b} &= \overline{(\mathbf{b}^H\mathbf{x}[n])(\mathbf{x}^H[n]\mathbf{b})} \\ &= \overline{|\mathbf{b}^H\mathbf{x}[n]|^2} \\ &\geq 0. \end{aligned} \tag{2.8}$$

   Since this quadratic form is always real and non-negative, the matrix $\mathbf{R}_{\mathbf{xx}}$ is non-negative definite.

3. We can use non-negativity of correlation matrix to show the non-negativity of the eigenvalues of $\mathbf{R}_{\mathbf{xx}}$. The ith eigenvalue and the ith eigenvector are related by

$$\mathbf{R}_{\mathbf{xx}}\mathbf{v}_i = \lambda_i\mathbf{v}_i. \tag{2.9}$$

   We can form a quadratic form by premultiplying by $\mathbf{v}_i^H$,

$$\mathbf{v}_i^H\mathbf{R}_{\mathbf{xx}}\mathbf{v}_i = \lambda_i\mathbf{v}_i^H\mathbf{v}_i. \tag{2.10}$$

   The eigenvalue λ_i real and non-negative as it is the ratio of a real non-negative quantity and a real positive quantity,[2]

$$\lambda_i = \frac{\mathbf{v}_i^H\mathbf{R}_{\mathbf{xx}}\mathbf{v}_i}{\mathbf{v}_i^H\mathbf{v}_i} \geq 0. \tag{2.11}$$

[2]The ratio is known as the Rayleigh quotient and can be used as the basis of an iteration to find eigenvalues and eigenvectors [10].

2.2 Orthogonality Principle – Wiener-Hopf Equation

We want to find the filter coefficient vector **w** which minimizes the squared error. We cannot simply take the derivative of the (real) squared error with respect to the complex coefficients. Such an operation is not well-defined. Instead we will take derivatives with respect to the real and imaginary parts of the coefficient vector separately and set each of these to zero. To develop an expression for the optimal filter coefficients, we start with the first expression in Eq. (2.5). Then using the chain rule of differentiation,

$$\begin{aligned}\frac{d\varepsilon}{da} &= \overline{e[n]\frac{de^*[n]}{da}} + \overline{e^*[n]\frac{de[n]}{da}} \\ &= 2\text{Re}\Big[\overline{e^*[n]\frac{de[n]}{da}}\Big] \\ &= -2\text{Re}\Big[\overline{e^*[n]\frac{dy[n]}{da}}\Big].\end{aligned} \tag{2.12}$$

The derivatives of $y[n]$ with respect to $\text{Re}[w_k]$ and $\text{Im}[w_k]$ are

$$\frac{dy[n]}{d\,\text{Re}[w_k]} = x[n-D_k], \qquad \frac{dy[n]}{d\,\text{Im}[w_k]} = -jx[n-D_k]. \tag{2.13}$$

We can stack these derivatives in vector form,

$$\frac{dy[n]}{d\,\text{Re}[\mathbf{w}]} = \mathbf{x}[n], \qquad \frac{dy[n]}{d\,\text{Im}[\mathbf{w}]} = -j\mathbf{x}[n]. \tag{2.14}$$

We set the derivatives of the squared error to zero to get the optimal filter coefficients. The derivative with respect to the real part of the coefficient vector is

$$\frac{d\varepsilon}{d\,\text{Re}[\mathbf{w}]} = -2\text{Re}\big[\overline{e^*[n]\mathbf{x}[n]}\big]. \tag{2.15}$$

For the imaginary part of the coefficient vector,

$$\begin{aligned}\frac{d\varepsilon}{d\,\text{Im}[\mathbf{w}]} &= 2\text{Re}\big[\overline{e^*[n]j\mathbf{x}[n]}\big] \\ &= -2\text{Im}\big[\overline{e^*[n]\mathbf{x}[n]}\big].\end{aligned} \tag{2.16}$$

Both sets of derivative equations have real outcomes. Associating these equations with the real and imaginary parts of a complex expression and setting the result to zero,

$$\begin{aligned}\frac{d\varepsilon}{d\,\mathrm{Re}[\mathbf{w}]} + j\frac{d\varepsilon}{d\,\mathrm{Im}[\mathbf{w}]} &= -2\overline{e^*[n]\mathbf{x}[n]} \\ &= \mathbf{0}.\end{aligned} \tag{2.17}$$

This is the orthogonality principle.[3] [4]

Orthogonality Principle:

For the optimal coefficients, the error and the data are orthogonal,

$$\overline{\mathbf{x}[n]e^*_{\mathrm{min}}[n]} = \mathbf{0}. \tag{2.18}$$

We can derive a corollary to the orthogonality principle. Consider the cross-correlation between the filter output and the error,

$$\begin{aligned}\overline{y_{\mathrm{opt}}[n]e^*_{\mathrm{min}}[n]} &= \mathbf{w}^H_{\mathrm{opt}}\overline{\mathbf{x}[n]e^*_{\mathrm{min}}[n]} \\ &= 0.\end{aligned} \tag{2.19}$$

Corollary, Orthogonality Principle:

For the optimal coefficients, the filter output and error are orthogonal,

$$\overline{y_{\mathrm{opt}}[n]e^*_{\mathrm{min}}[n]} = 0. \tag{2.20}$$

Using the orthogonality principle, we can find the optimal filter coefficients by setting the following expression to zero,

$$\begin{aligned}\overline{\mathbf{x}[n]e^*_{\mathrm{min}}[n]} &= \overline{\mathbf{x}[n]\big(d[n] - \mathbf{w}^H_{\mathrm{opt}}\mathbf{x}[n]\big)^*} \\ &= \overline{\mathbf{x}[n]d^*[n]} - \overline{\mathbf{x}[n]\mathbf{x}^H[n]}\mathbf{w}_{\mathrm{opt}} \\ &= \mathbf{r}_{\mathbf{x}d} - \mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\mathrm{opt}}.\end{aligned} \tag{2.21}$$

[3]Two signals are orthogonal if $\overline{x[n]y^*[n]} = 0$. Two signals are uncorrelated if $\overline{x[n]y^*[n]} = \overline{x[n]}\;\overline{y^*[n]}$. If either signal is zero mean, uncorrelatedness is the same as orthogonality.

[4]In this exposition, we end up with the orthogonality principle. An alternate exposition which leads directly to the Wiener-Hopf equations is given in Appendix A. In that appendix, rules for differentiating a scalar value with respect to a complex vector are outlined. The differentiation of the mean-square error with respect to a vector of complex coefficients is used as an example.

Wiener-Hopf Equations:

The optimal filter coefficients are given by solving a set of linear equations involving the correlation for the input data and the cross-correlation between the input signal and the desired signal,

$$\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\text{opt}} = \mathbf{r}_{\mathbf{x}d}. \tag{2.22}$$

The Wiener-Hopf equation is a key result that will reappear in many places in these notes. This equation shows that the general mean-square filtering problem can be solved with knowledge of the autocorrelation of the filter input and the cross-correlation of the input and desired signals. The different cases to which the Wiener-Hopf equation is applied will differ in the way the correlation values needed are calculated.

2.3 Properties of the Optimal Filter Coefficients

From Eq. (2.5), the squared error using the optimal coefficients is

$$\varepsilon_{\text{min}} = \overline{|d[n]|^2} - 2\text{Re}\left[\mathbf{w}_{\text{opt}}^H\mathbf{r}_{\mathbf{x}d}\right] + \mathbf{w}_{\text{opt}}^H\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\text{opt}}. \tag{2.23}$$

From Eq. (2.22), the optimal coefficients satisfy

$$\mathbf{w}_{\text{opt}}^H\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\text{opt}} = \mathbf{w}_{\text{opt}}^H\mathbf{r}_{\mathbf{x}d}. \tag{2.24}$$

Since $\mathbf{R}_{\mathbf{xx}}$ is Hermitian, the lefthand side is real. This shows that the righthand side is also real and so the real operator in Eq. (2.23) is redundant. The squared error using the optimal coefficients can be written in a number of different ways,

$$\begin{aligned}\varepsilon_{\text{min}} &= \overline{|d[n]|^2} - \mathbf{w}_{\text{opt}}^H\mathbf{r}_{\mathbf{x}d} \\ &= \overline{|d[n]|^2} - \mathbf{w}_{\text{opt}}^H\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\text{opt}} \\ &= \overline{|d[n]|^2} - \mathbf{r}_{\mathbf{x}d}^H\mathbf{R}_{\mathbf{xx}}^{-1}\mathbf{r}_{\mathbf{x}d}.\end{aligned} \tag{2.25}$$

The error signal for the optimal coefficients can be written as $e_{\text{min}}[n] = d[n] - y_{\text{opt}}[n]$. But from the corollary to the orthogonality principle, the error $e_{\text{min}}[n]$ is orthogonal to the filter output $y_{\text{opt}}[n]$. This means that the energies are related as

$$\varepsilon_{\text{min}} = \overline{|d[n]|^2} - \overline{|y_{\text{opt}}[n]|^2}. \tag{2.26}$$

It is then evident that the last term in each line of Eq. (2.25) is equal to $\overline{|y_{\text{opt}}[n]|^2}$: *the energy of the filter output is always less than that of the desired signal* and the difference is equal to the error energy.

It can be shown that the squared error for an arbitrary filter with coefficients $\mathbf{w}$ can be expressed in terms of the squared error for the optimal filter,

$$\varepsilon = \varepsilon_{\text{min}} + (\mathbf{w} - \mathbf{w}_{\text{opt}})^H \mathbf{R}_{\mathbf{xx}} (\mathbf{w} - \mathbf{w}_{\text{opt}}). \tag{2.27}$$

Since $\mathbf{R}_{\mathbf{xx}}$ is non-negative definite, the second term adds to the error. This also reiterates the fact that the squared error is minimum for $\mathbf{w} = \mathbf{w}_{\text{opt}}$.

2.3.1 Searching for the Best Coefficients

In some cases, the set of coefficients are restricted to a codebook of possible values. This is the case for quantized coefficients. We will denote one of the sets of quantized coefficients as $\mathbf{w}_i$. The search for the best set of coefficients can be expressed as

$$\begin{aligned}\hat{\mathbf{w}} &= \min_{\mathbf{w}_i}\big(\overline{|d[n]|^2} - 2\text{Re}\big[\mathbf{w}_i^H \mathbf{r}_{\mathbf{x}d}\big] + \mathbf{w}_i^H \mathbf{R}_{\mathbf{xx}} \mathbf{w}_i\big) \\ &= \max_{\mathbf{w}_i}\big(2\text{Re}\big[\mathbf{w}_i^H \mathbf{r}_{\mathbf{x}d}\big] - \mathbf{w}_i^H \mathbf{R}_{\mathbf{xx}} \mathbf{w}_i\big).\end{aligned} \tag{2.28}$$

Using Eq. (2.27), we have another option. We can solve Eq. (2.22) for $\mathbf{w}_{\text{opt}}$ and then find $\hat{\mathbf{w}}$ from

$$\hat{\mathbf{w}} = \min_{\mathbf{w}_i}\big((\mathbf{w}_i - \mathbf{w}_{\text{opt}})^H \mathbf{R}_{\mathbf{xx}} (\mathbf{w}_i - \mathbf{w}_{\text{opt}})\big). \tag{2.29}$$

This approach can be useful for finding a single coefficient (both Eq. (2.27) and Eq. (2.22) are scalar in this case).

2.4 Affine Estimation

The development so far has made no assumptions on whether the signals to be filtered are zero mean or not. For signals with non-zero mean, one can achieve a lower mean square error if the system is modified to subtract out the mean, apply filtering to the resulting signal, and then add back the mean to the output. Why should this produce a lower mean-square error? The answer lies in the fact that the output signal is no longer strictly a linear combination of the inputs – it is created using an affine transformation [5]. An affine transformation is a linear transformation plus a shift. The filter plus the shift term create an output signal as follows: (c.f. Eq. (2.1))

$$y[n] = \sum_{k=0}^{M-1} w_k^* x[n - D_k] + \beta^*. \tag{2.30}$$

This differs from linear filtering with the addition of the constant term β^*. With this extra degree of freedom, the resulting mean-square error can be smaller than for the linear system. The block diagram of the affine filtering system is shown in Fig. 2.2.

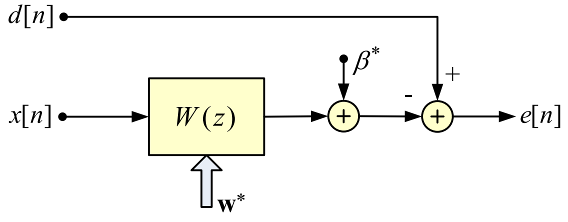


Fig. 2.2 Affine filtering system

We create a new vector of parameters,

$$\mathbf{w}_a = \begin{bmatrix} \mathbf{w} \\ \beta \end{bmatrix}. \tag{2.31}$$

The vector of input values can be augmented with a constant,

$$\mathbf{x}_a[n] = \begin{bmatrix} \mathbf{x}[n] \\ 1 \end{bmatrix}. \tag{2.32}$$

Now the filtering operation can be put back into the vector form (c.f. Eq. (2.2))

$$y[n] = \mathbf{w}_a^H \mathbf{x}_a[n]. \tag{2.33}$$

The error with respect to a desired signal $d[n]$ and the mean-square error have expressions similar to those developed before. The optimal coefficients are found from

$$\mathbf{R}_{\mathbf{x}_a\mathbf{x}_a} \mathbf{w}_{a\text{opt}} = \mathbf{r}_{\mathbf{x}_a d}, \tag{2.34}$$

where the correlation matrix $\mathbf{R}_{\mathbf{x}_a\mathbf{x}_a}$ and the cross-correlation vector $\mathbf{r}_{\mathbf{x}_a d}$ are given by

$$\mathbf{R}_{\mathbf{x}_a\mathbf{x}_a} = \begin{bmatrix} \mathbf{R}_{\mathbf{xx}} & \overline{\mathbf{x}[n]} \\ \overline{\mathbf{x}[n]}^H & 1 \end{bmatrix}, \qquad \mathbf{r}_{\mathbf{x}_a d} = \begin{bmatrix} \mathbf{r}_{\mathbf{x}d} \\ \overline{d[n]}^* \end{bmatrix}. \tag{2.35}$$

The equations for the optimal coefficients can then be written as

$$\begin{bmatrix} \mathbf{R}_{\mathbf{xx}} & \overline{\mathbf{x}[n]} \\ \overline{\mathbf{x}[n]}^H & 1 \end{bmatrix} \begin{bmatrix} \mathbf{w}_{\text{opt}} \\ \beta_{\text{opt}} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{\mathbf{x}d} \\ \overline{d[n]}^* \end{bmatrix}. \tag{2.36}$$

We can split this set of equations into two parts,

$$\begin{aligned} \mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\text{opt}} + \overline{\mathbf{x}[n]}\beta_{\text{opt}} &= \mathbf{r}_{\mathbf{x}d} \\ \overline{\mathbf{x}[n]}^H\mathbf{w}_{\text{opt}} + \beta_{\text{opt}} &= \overline{d[n]}^*. \end{aligned} \tag{2.37}$$

Eliminating β_{opt} from these equations, we find

$$\left(\mathbf{R}_{\mathbf{xx}} - \overline{\mathbf{x}[n]}\;\overline{\mathbf{x}^H[n]}\right)\mathbf{w}_{\text{opt}} = \mathbf{r}_{\mathbf{x}d} - \overline{\mathbf{x}[n]}\;\overline{d^*[n]}. \tag{2.38}$$

We can identify the first term on the lefthand side as the autocorrelation matrix for the mean-removed input signal $x[n] - \overline{x[n]}$. The righthand side is the cross-correlation of the mean-removed input signal and the mean-removed desired signal $d[n] - \overline{d[n]}$. We can express the optimal value of β as,

$$\beta_{\text{opt}} = \overline{d[n]}^* - \overline{\mathbf{x}[n]}^H\mathbf{w}_{\text{opt}}. \tag{2.39}$$

We will simplify the notation and explicitly show constant mean values.

Affine Estimator:

For signals with non-zero means, the optimal affine estimator is given by

$$\begin{aligned} \mathbf{C}_{\mathbf{xx}}\mathbf{w}_{\text{opt}} &= \mathbf{c}_{\mathbf{x}d} \\ \beta_{\text{opt}} &= \overline{d}^* - \overline{x}\sum_{k=0}^{M-1} w_{k\text{opt}}^*, \end{aligned} \tag{2.40}$$

where $\mathbf{C}_{\mathbf{xx}}$ is a covariance matrix ($\mathbf{R}_{\mathbf{xx}}$ with $|\overline{x}|^2$ subtracted from each element) and $\mathbf{c}_{\mathbf{x}d}$ is a cross-covariance vector ($\mathbf{r}_{\mathbf{x}d}$ with $\overline{x}\overline{d}^*$ subtracted from each element).

If the means of the desired signal and the input signal are both zero, we get the same solution as we found before, i.e., $\beta_{\text{opt}} = 0$. For non-zero means, the additional term due to β_{opt} makes the overall solution different from that for the zero-mean case.

An alternate interpretation of the affine filtering system is based on processing zero-mean signals. First we note that β_{opt} in Eq. (2.40) has two parts. The second part subtracts out the mean at the output of the filter; the first adds in the mean of the desired signal. Using these results, we can rearrange the block diagram as shown in Fig. 2.3. Now the filter acts on a mean-removed input, and the mean of the output is shifted to match that of the desired signal. The optimal filter weights remain unchanged from those for the system in Fig. 2.2.

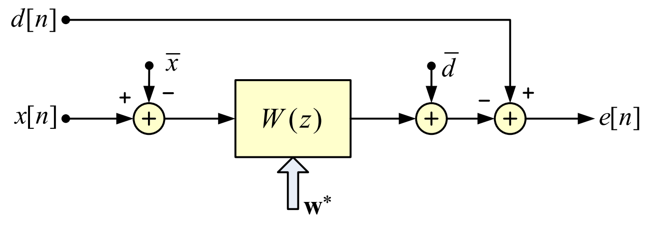


Fig. 2.3 Alternate form of the affine filtering system acting on a zero-mean input

2.5 Multiple Input Signals

Consider the case where two inputs $x_1[n]$ and $x_2[n]$ are separately filtered. The outputs are added to form an estimate of $d[n]$. The setup is shown in Fig. 2.4. The summed output can be written as

$$y[n] = \mathbf{w}_1^H \mathbf{x}_1[n] + \mathbf{w}_2^H \mathbf{x}_2[n], \tag{2.41}$$

where

$$\mathbf{x}_1[n] = \begin{bmatrix} x_1[n - D_{1,0}] \\ x_1[n - D_{1,1}] \\ \vdots \\ x_1[n - D_{1,M_1-1}] \end{bmatrix} \qquad \mathbf{x}_2[n] = \begin{bmatrix} x_2[n - D_{2,0}] \\ x_2[n - D_{2,1}] \\ \vdots \\ x_2[n - D_{2,M_2-1}] \end{bmatrix}. \tag{2.42}$$

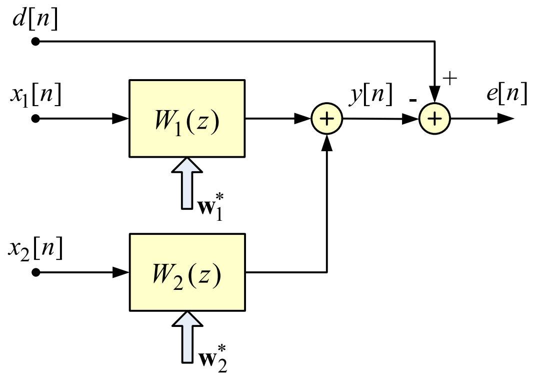


Fig. 2.4 Filtering two signals

The formalism to handle this case is to form an augmented vector containing the two input signals and an augmented weight vector,

$$\mathbf{x}[n] = \begin{bmatrix} \mathbf{x}_1[n] \\ \mathbf{x}_2[n] \end{bmatrix}, \qquad \mathbf{w} = \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix}. \tag{2.43}$$

The set of equations to be solved is then

$$\begin{bmatrix} \mathbf{R}_{\mathbf{x}_1\mathbf{x}_1} & \mathbf{R}_{\mathbf{x}_1\mathbf{x}_2} \\ \mathbf{R}_{\mathbf{x}_2\mathbf{x}_1} & \mathbf{R}_{\mathbf{x}_2\mathbf{x}_2} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{\mathbf{x}_1 d} \\ \mathbf{r}_{\mathbf{x}_2 d} \end{bmatrix}. \tag{2.44}$$

This analysis can be carried over to more than two signals. It is also to be noted that the affine filtering scheme (see Fig. 2.2) is just a specialization of the two input signal case with the second signal being a constant equal to one and the second filter having only one coefficient β.

2.6 Uniqueness of the Wiener-Hopf Equations

A set of linear equations has a unique solution if coefficient matrix (here $\mathbf{R}$) is non-singular. Non-singularity requires that the determinant $|\mathbf{R}|$ be non-zero. The determinant can be expressed as the product of the eigenvalues,

$$|\mathbf{R}| = \prod_i \lambda_i. \tag{2.45}$$

We have shown earlier that for a non-negative definite correlation matrix the eigenvalues are non-negative. For positive (no zeros) eigenvalues, the correlation matrix must be positive definite. One can argue that if the input signal has any amount of uncorrelated noise added, the correlation matrix is theoretically positive definite, and thus has the Wiener-Hopf equations give a unique set of filter coefficients.

In practice, problems can arise. Consider a zero-valued signal. All eigenvalues of the correlation matrix are zero. This case can be caught easily by examining whether $r[0]$ is zero or not. The filter coefficients can then be set to any convenient set of values.

Consider a more general case of an overdetermined set of equations. This can arise, for instance, if the input signal is a sinusoid, and the number of coefficients in the filter is larger than two. For three or more coefficients, there are an infinity of solutions. In the case of prediction using order-recursive algorithms, this case can be detected by observing that with two coefficients, the error falls to zero. At that point in the order recursion, the remaining coefficients can be set to zero, and the recursion halted.

Problems

P2.1 Correlation Matrix and Cross-Correlation Vector

Given an autocorrelation sequence $r_{xx}[l]$ and a cross-correlation sequence $r_{xd}[l]$, find expressions for the elements of the correlation matrix $\mathbf{R}$ and the cross-correlation vector $\mathbf{r}_{\mathbf{x}d}$. These elements should be expressed in terms of the filter delays D_k.

P2.2 Wiener-Hopf Equations Via Complex Differentiation

Consider the matrix-vector expression for the mean-square error (see Eq. (2.5),

$$\varepsilon = \overline{|d[n]|^2} - 2\mathrm{Re}[\mathbf{w}^H \mathbf{r}_{\mathbf{x}d}] + \mathbf{w}^H \mathbf{R}_{\mathbf{xx}} \mathbf{w}, \tag{P2.2-1}$$

derive the Wiener-Hopf equations using the differentiation operations described in Appendix A.

P2.3 Non-Negative Definiteness

In Appendix B, a non-negative definite matrix is defined in terms of a quadratic form,

$$\mathbf{a}^H \mathbf{R} \mathbf{a} \geq 0. \tag{P2.3-1}$$

(a) Show that a correlation matrix is always non-negative definite.

(b) Show that the non-negativity of the quadratic form occurs *if and only if* the eigenvalues of the matrix are non-negative.

P2.4 Augmented Vectors

Consider three vectors of samples: $\mathbf{x}_1[n]$, with N_1 elements, $\mathbf{x}_2[n]$ with N_2 elements, and $\mathbf{x}_3[n]$ with N_3 elements. The vectors are jointly wide-sense stationary. Let $\mathbf{x}[n]$ be the augmented vector,

$$\mathbf{x}[n] = \begin{bmatrix} \mathbf{x}_1[n] \\ \mathbf{x}_2[n] \\ \mathbf{x}_3[n] \end{bmatrix}, \tag{P2.4-1}$$

Give an expression for the correlation matrix for $\mathbf{x}[n]$ as a partitioned matrix, expressed in terms of the correlations and cross-correlations of the constituent vectors. Indicate the sizes of the constituent matrices.

P2.5 Sum of Samples

Consider the sum of two vectors containing samples from jointly wide-sense stationary processes,

$$\mathbf{x}[n] = \mathbf{x}_1[n] + \mathbf{x}_2[n]. \tag{P2.5-1}$$

Express the correlation matrix for $\mathbf{x}[n]$ in terms of the correlations and cross-correlations of the constituent vectors.

P2.6 Completion of Squares

An alternative minimization procedure involves rewriting the expression for the mean-square error in the form

$$\varepsilon = (\overline{|d[n]|^2} - \mathbf{w}_\mathrm{o}^H \mathbf{r}_{\mathbf{x}d}) + (\mathbf{w} - \mathbf{w}_\mathrm{o})^H \mathbf{R}_{\mathbf{xx}} (\mathbf{w} - \mathbf{w}_\mathrm{o}), \tag{P2.6-1}$$

where $\mathbf{w}_\mathrm{o}$ is *any* solution to $\mathbf{R}_{\mathbf{xx}}\mathbf{w}_\mathrm{o} = \mathbf{r}_{\mathbf{x}d}$. Develop this form and then use this form to show that $\mathbf{w}_\mathrm{opt} = \mathbf{w}_\mathrm{o}$ is the set of filter coefficients which minimize the mean-square error.

Chapter 3

Mean-Square Filtering: Stochastic Signals

In this chapter, we will consider stochastic signals for which the averaging operation becomes an statistical expectation over an ensemble of signals. We will first consider a larger perspective on the mean-square estimation problem.

3.1 Mean-Square Estimation

We observe $\mathbf{x}$. It can be shown that the best unrestricted mean-square estimate of $\mathbf{d}$ given the observation is the conditional mean [30, 32]

$$\hat{\mathbf{d}}(\mathbf{x}) = E[\mathbf{d}|\mathbf{x}]. \tag{3.1}$$

This estimate need not be a linear function of the observation. For the case of $\mathbf{x}$ being Gaussian, the best estimate is also linear. Note that the unrestricted estimate obeys the orthogonality principle: the error is orthogonal to the estimate. This implies that the energy of the estimate is always lower than the energy of the desired signal.

For the sequel we will concentrate on linear estimates for wide-sense stationary processes. This specializes the formalism developed in Chapter 2 for stochastic signals.

3.2 Wide-Sense Stationary Processes

We will assume that the signals are wide-sense stationary. Then the correlation values depend only on time differences (correlation lags),

$$E\big[u[k]v^*[l]\big] = r_{uv}[k-l]. \tag{3.2}$$

The correlation matrix for the wide-sense stationary stochastic case is

$$\mathbf{R}_{\mathbf{xx}} = E\big[\mathbf{x}[n]\mathbf{x}^H[n]\big], \tag{3.3}$$

with element k, l equal to

$$E\big[x[n-D_k]x^*[n-D_l]\big] = r_{xx}[D_l - D_k]. \tag{3.4}$$

The correlation matrix then takes on the form

$$\mathbf{R}_{\mathbf{xx}} = \begin{bmatrix} r_{xx}[0] & r_{xx}[D_1 - D_0] & r_{xx}[D_2 - D_0] & \cdots & r_{xx}[D_{M-1} - D_0] \\ r_{xx}[D_0 - D_1] & r_{xx}[0] & r_{xx}[D_2 - D_1] & \cdots & r_{xx}[D_{M-1} - D_1] \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ r_{xx}[D_0 - D_{M-1}] & r_{xx}[D_1 - D_{M-1}] & r_{xx}[D_2 - D_{M-1}] & \cdots & r_{xx}[0] \end{bmatrix} \tag{3.5}$$

The matrix is always Hermitian symmetric since $r_{xx}[k] = r_{xx}^*[-k]$ (see Eq. (3.4)). The elements of the matrix depend on delay differences.

The cross-correlation vector $\mathbf{r}_{\mathbf{x}d}$ for the stochastic case has element k equal to

$$E\big[x[n-D_k]d^*[n]\big] = r_{xd}[-D_k], \tag{3.6}$$

which gives the vector form

$$\mathbf{r}_{\mathbf{x}d} = \begin{bmatrix} r_{xd}[-D_0] \\ r_{xd}[-D_1] \\ \vdots \\ r_{xd}[-D_{M-1}] \end{bmatrix}. \tag{3.7}$$

Since $r_{xd}[-k] = r_{dx}^*[k]$, the cross-correlation vector can also be written in terms of elements of the form $r_{dx}^*[D_k]$. The cross-correlation values in the vector depend directly on the delays (not delay differences).

The optimal coefficients are found from the Wiener-Hopf equations

$$\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\text{opt}} = \mathbf{r}_{\mathbf{x}d}. \tag{3.8}$$

3.2.1 Equally Spaced Delays

A specialization occurs if the delays are equally spaced and ordered, viz., $D_l - D_k = (l-k)p$. Then the correlation matrix takes on the form

$$\mathbf{R_{xx}} = \begin{bmatrix} r_{xx}[0] & r_{xx}[p] & r_{xx}[2p] & \cdots & r_{xx}[(M-1)p] \\ r_{xx}[-p] & r_{xx}[0] & r_{xx}[p] & \cdots & r_{xx}[(M-2)p] \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ r_{xx}[-(M-1)p] & r_{xx}[-(M-2)p] & r_{xx}[-(M-3)p] & \cdots & r_{xx}[0] \end{bmatrix}. \tag{3.9}$$

The matrix is now Toeplitz, i.e., the values down the diagonals are equal. See Appendix C for a discussion of the properties of Toeplitz matrices. We have assumed that the delays are ordered. If the delay values are in a permuted order, the Toeplitz property may be absent.

For equally spaced delays, the cross-correlation vector becomes

$$\mathbf{r_{x}}_{d} = \begin{bmatrix} r_{xd}[-D_0] \\ r_{xd}[-D_0 - p] \\ \vdots \\ r_{xd}[-D_0 - (M-1)p] \end{bmatrix}. \tag{3.10}$$

3.3 Filtering Configurations

The minimum mean-square error solution can be applied to several scenarios. In the following we enumerate three important cases. In each case we need to be able to find an expression for the correlation of the input signal and for the cross-correlation between the input signal and the desired signal.

Equalizer: In Fig. 3.1, we have a system where the filter $W(z)$ is designed to "equalize" the channel response so that the output $y[n]$ is the best match to $d[n]$. The equalizer, in some sense, inverts the effects of the channel. It generally cannot form an exact inverse, both due to its finite length and due to the presence of noise in the input signal. The correlation of the input signal is that of the output of the channel filter. The cross-correlation is that of the input signal with the desired signal. For both, we need to express the correlations in terms of the correlations of a filtered signal.

Canceller: In Fig. 3.2, we have a system where the filter $W(z)$ is designed to match the response of the channel. In doing so, it provides a cancellation in creating the error signal. The correlation of the input signal is that of the given input signal $a[n]$. The cross-correlation is between the desired signal (a filtered version of $a[n]$) and $a[n]$ itself.

Predictor: In Fig. 3.3, we see a prediction system. This system is characterized by having only positive delay values ($D_k > 0$) for the filter. The goal is to get the best estimate of the current input value based on past input values. Prediction will be the focus of Chapter 4.

3.4 Correlation Values

The following sections discuss the calculation of correlation values for special cases.

3.4.1 Filtered Signals

Of some interest to us will be the correlation for filtered wide-sense stationary signals. These filtered signals arise in, for instance, the equalization and cancellation cases outlined above. Let $x[n]$ and $d[n]$ be filtered versions of the same wide-sense stationary signal. They will then be jointly wide-sense stationary. The correlation relationships for filtered random processes are developed in Appendix D. As a specific example, if a signal $a[n]$ is passed through a filter $h[n]$ to give the signal $x[n]$, the autocorrelation for $x[n]$ is given by

$$r_{xx}[k] = r_{hh}[k] * r_{aa}[k], \tag{3.11}$$

where $r_{hh}[k]$ is the deterministic correlation for the filter,

$$r_{hh}[k] = \sum_{m=-\infty}^{\infty} h[m]h^*[m-k]. \tag{3.12}$$

We have now seen that for wide-sense stationary signals, the equations result in the Wiener-Hopf equations,

$$\mathbf{R_{xx}}\mathbf{w}_{\text{opt}} = \mathbf{r}_{\mathbf{x}d}, \tag{3.13}$$

where the matrix $\mathbf{R_{xx}}$ is Toeplitz for equally spaced delays. Such a set of linear equations can be solved with order M^2 operations using the Levinson algorithm. This is in contrast with the order M^3 operations needed to solve a general set of equations (for instance for non-equally spaced delays). The Levinson algorithm is described in Appendix J, and itself runs in parallel with the Durbin recursion described in Appendix I.[1]

[1]In the next chapter, we discuss linear prediction. In the case of one-step prediction, the structure of the cross-correlation vector is such that the Durbin recursion can be used directly.

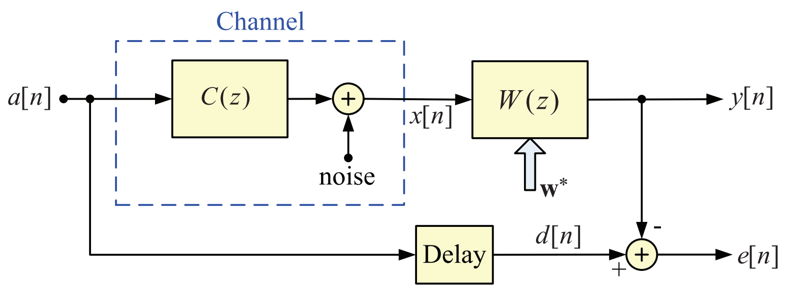


Fig. 3.1 Equalizer

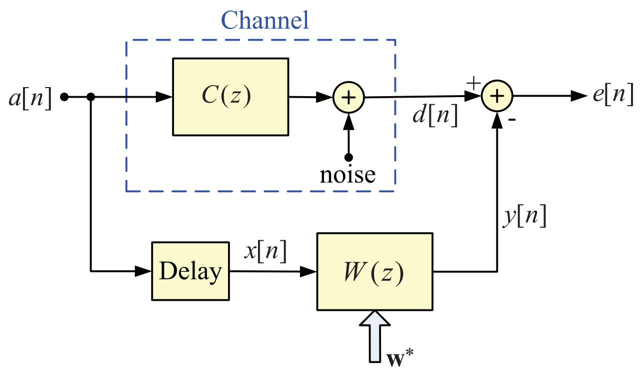


Fig. 3.2 Canceller

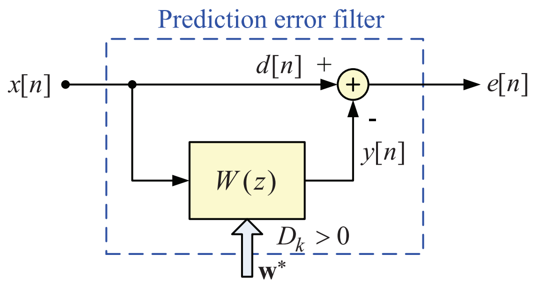


Fig. 3.3 Predictor

3.4.2 Sampled Signals

In many cases, a discrete-time signal is created by sampling a continuous-time signal,

$$x[n] = x(nT). \tag{3.14}$$

Let the autocorrelation for the wide-sense stationary continuous-time signal be $r_{xx}(\tau)$. It is easy to show that the autocorrelation for the discrete-time signal is (see Appendix E)

$$r_{xx}[k] = r_{xx}(kT). \tag{3.15}$$

The autocorrelation function for a sampled signal is found by sampling the autocorrelation function of the continuous-time signal.

The process of sampling a continuous-time signal can be described as a continuous-time to discrete-time (C/D) conversion. The conversion in the other direction needs more elaboration. A discrete-time to continuous-time (D/C) conversion can be modelled as creating a set of dirac delta functions with the area of each delta function being equal to a sample value,

$$x[n] \rightarrow \sum_{k=-\infty}^{\infty} x[k]\delta(t-kT). \tag{3.16}$$

After filtering this sequence of delta functions with a filter with impulse response $h(t)$, the interpolated output is in the form of a pulse amplitude modulated signal,

$$\sum_{k=-\infty}^{\infty} x[k]\delta(t-kT) * h(t) = \sum_{k=-\infty}^{\infty} x[k]h(t-kT). \tag{3.17}$$

3.4.3 Cyclostationary Processes

Another case of interest is that of processes which are jointly cyclostationary. These wide-sense cyclostationary processes have correlation functions which satisfy

$$r_{uv}[n,m] = r_{uv}[n+lN, m+lN], \tag{3.18}$$

for all integer l. For ease of exposition, we are assuming that processes are zero mean. This means that if we calculate a correlation at points n and m, we get the same correlation if we move both pointers by the same multiple of N. To see this more clearly, we can define a correlation function $r_{uv}[k;n]$, with the semicolon separating the arguments indicating that the first argument is a correlation lag value. For a cyclostationary process,

$$\begin{aligned} r_{uv}[k;n] &\overset{\triangle}{=} r_{uv}[n+k,n] \\ &= r_{uv}[n+lN+k,n+lM] \\ &= r_{uv}[k;n+lN]. \end{aligned} \tag{3.19}$$

Consider the modulated signal

$$x[n] = s[n]\cos(\omega_o n + \theta), \tag{3.20}$$

where $s[n]$ is a real process which is zero mean and wide sense stationary. We further assume that the cosine term is periodic with period N. Then it can be shown that $x[n]$ is cyclostationary,

$$\begin{aligned} r_{xx}[n,m] &= E[x[n]x^*[m]] \\ &= E[x[n+N]x^*[m+N]] \\ &= r_{xx}[n+N,m+N]. \end{aligned} \tag{3.21}$$

It is to be noted that we cannot attribute a conventional power spectral density as described later in Section 3.5 to a cyclostationary correlation function (since it is not a function of single time (correlation lag) variable). It is sometimes suggested that a phase randomization can be used. In the case of the modulated signal discussed above, this involves making the phase angle θ in Eq. (3.20) a random value over a 2π interval, rather than a fixed but unknown value. This phase randomization removes important information about the carrier phase, such as would be needed to demodulate the signal. However, by using this randomization artifice, the modulated signal $x[n]$ becomes stationary [37].

Consider an upsampled and interpolation operation which can be written as,

$$x[n] = \sum_{k=-\infty}^{\infty} a[k]h[n-kI], \tag{3.22}$$

where $a[k]$ is a wide-sense stationary process. Such a process can be created by upsampling $a[n]$ (inserting $I-1$ zeros between each sample) and then filtering the result with a filter with impulse response $h[n]$, see Fig. 3.4. The operation shown in the figure models the interpolation process and also the creation of pulse amplitude modulation (PAM) – the pulses $h[n]$ are modulated by the pulse amplitudes $a[k]$. The filtering of the up-sampled sequence can be written as

$$x[n] = \Big(\sum_{k=-\infty}^{\infty} a[k]\delta[n-kI]\Big) * h[n]. \tag{3.23}$$

Carrying out the convolution gives us Eq. (3.22). The correlation function for such filtered processes is developed in Appendix G.

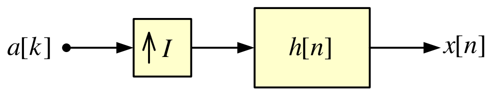


Fig. 3.4 Upsampling and filtering to create a cyclostationary process

The statistics of the cyclostationary process depend on the "phase" of the time index n. The phase can be defined in terms of the modulus operation,

$$p = ((n))_I. \tag{3.24}$$

We can express the filter response in polyphase form (see Appendix F),

$$H(z) = \sum_{p=0}^{I-1} z^{-p} H_p(z^I), \tag{3.25}$$

where the polyphase components (for a causal filter) are

$$\begin{aligned} H_p(z) &= \sum_{k=0}^{M_p-1} h[kI+p]z^{-k} \\ &= \sum_{k=0}^{M_p-1} h_p[k]z^{-k} \end{aligned} \qquad 0 \le p < I. \tag{3.26}$$

The number of coefficients for a polyphase component filter is given by

$$M_p = \Big\lfloor \frac{M-1+I-p}{I} \Big\rfloor. \tag{3.27}$$

where $\lfloor \cdot \rfloor$ gives the smallest integer less than or equal to its argument. Using the polyphase decomposition, a block diagram of the restructured filter is shown in Fig. 3.5.

One can note that the outputs of the polyphase filters (before upsampling) are still stationary. The results given in Appendix G can then be applied to give the cross-correlations between the filter outputs. These filter outputs are then interleaved to give the final system output. The correlation function for the output signal is,

$$r_{xx}[kI+p, lI+q] = r_{h_p h_q}[k-l] * r_{aa}[k-l], \qquad 0 \le p,q < 1, \tag{3.28}$$

where $r_{h_p h_q}[k]$ is the deterministic cross-correlation for the filter responses and $r_{aa}[k]$ is the correlation function for the wide-sense stationary signal $a[n]$.

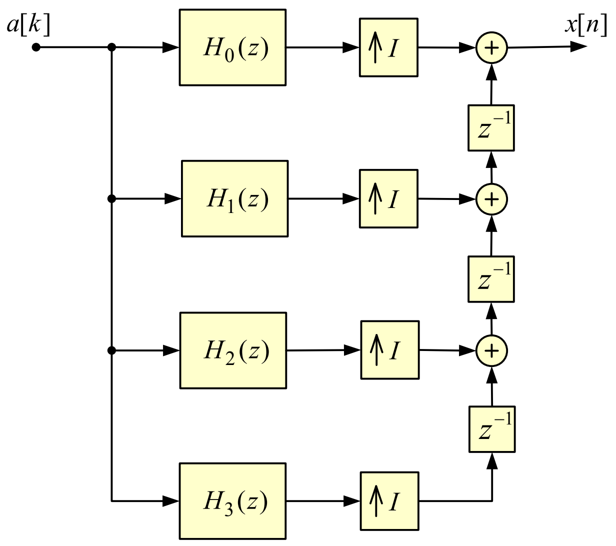


Fig. 3.5 Polyphase decomposition for an interpolation system

Solving for a minimum mean-square error when cyclostationary signals are involved, will sometimes lead to decoupled equations – a separate minimization analysis applies for each phase.

3.5 Power Spectral Density

We have been concerned with designing finite length filters to minimize the mean-square difference between the filter output and a desired signal. We can set describe the statistical properties of the signals in terms of correlations or the corresponding power spectral density (also known as the power spectrum), and cross-correlations or the corresponding cross-power spectral density (also known as the cross-power spectrum).

The cross-power spectral density of two signals $x[n]$ and $y[n]$ is the discrete-time Fourier transform of the cross-correlation of the two signals,

$$S_{xy}(\omega) = \sum_{k=-\infty}^{\infty} r_{xy}[k] e^{-j\omega k}. \tag{3.29}$$

The power spectral density of $x[n]$ is the DTFT of the autocorrelation of $x[n]$,

$$S_{xx}(\omega) = \sum_{k=-\infty}^{\infty} r_{xx}[k] e^{-j\omega k}. \tag{3.30}$$

The power spectral density is non-negative for a "proper" autocorrelation sequence, see Appendix B. This is a consequence of the non-negative definiteness of the autocorrelation sequence.

The power spectral density gives us an alternate description of the correlation. This description of a random signal in terms of the power spectral density is exploited in the examples in Chapter 8 and Chapter 9.

3.5.1 Signals with Discrete Frequency Components

In modelling the power spectral density of a signal, it is useful to be able to have the power spectral density consist of both continuous and discrete frequency components. The discrete frequency components correspond to sinusoidal signal components. A sinusoid of frequency ω_c is of the form

$$c[n] = A\cos(\omega_c n + \theta). \tag{3.31}$$

If θ is a random variable distributed uniformly over an interval of length 2π, we can calculate the correlation function for $c[n]$,

$$\begin{aligned} r_{cc}[n+k,n] &= A^2 E[\cos(\omega_c(n+k)+\theta)\cos(\omega_c n+\theta)] \\ &= \frac{A^2}{2}E[\cos(\omega_c(2n+k)+2\theta)] + \cos(\omega_c k) \\ &= \frac{A^2}{2}\cos(\omega_c k). \end{aligned} \tag{3.32}$$

This sinusoid with randomized phase is wide-sense stationary with the periodic correlation as given above. This same phase randomization was disparaged above, but here as a model for a signal with an unknown phase, it is useful. The power spectral density is the DTFT of the correlation and results in delta functions (discrete components) in the power spectral density [23],

$$S_{cc}(\omega) = \pi\big[\delta(\omega-\omega_c) + \delta(\omega+\omega_c)\big]. \tag{3.33}$$

3.5.2 Frequency Domain Analysis

We have been concerned with designing finite length filters to minimize the mean-square difference between the filter output and a desired signal. We can set up the equations for designing a filter in the frequency domain. The power spectral density of the error sequence can be expressed in terms of the power spectral density of the input to the filter and the cross-power spectral density of the desired signal and the input to the filter,

$$S_{ee}(\omega) = S_{dd}(\omega) - 2\mathrm{Re}[W(\omega)S_{xd}(\omega)] + |W(\omega)|^2 S_{xx}(\omega). \tag{3.34}$$

For a fixed ω, the mean-square error is minimized when

$$W(\omega) = \frac{S_{xd}(\omega)}{S_{xx}(\omega)}. \tag{3.35}$$

The optimal filter found this way is not necessarily causal and is generally infinite in extent.

An application of the frequency domain relationship is filtering using the Discrete Fourier Transform (DFT). In the case of frame-by-frame processing of a time signal, filtering can be done in the DFT domain. The DFT values correspond to samples taken at the points $\omega_k = 2\pi k/N$. The optimal filter response can be found frequency by frequency. Since multiplication in the DFT domain corresponds to *circular convolution*, care must be taken so that the result conforms to linear filtering [12].

Problems

P3.1 Properties of the Correlation

Consider a wide-sense stationary signal with autocorrelation $r[k]$.

(a) Show that $r[0] \geq |r[k]|$, for all k. Hint: Express the correlations in terms of the power spectral density (see Appendix B).

(b) Show that if $r[N] = r[0]$ for some $N \neq 0$, then $r[k]$ is periodic with period N. Hint: Use Schwarz's inequality on the product of the correlation of $x[n+N+l] - x[n+l]$ and the correlation of $x[n]$.

P3.2 Best Unrestricted Mean-Square Estimate

Show that the best unrestricted estimate in the mean-square sense is given by the conditional mean,

$$\hat{\mathbf{d}} = E[\mathbf{d}|\mathbf{x}], \tag{P3.2-1}$$

where $\mathbf{d}$ is a vector of desired values, and $\mathbf{x}$ is a vector of observations.

P3.3 Best Unrestricted Mean-Square Estimate of a Gaussian Process

If the desired vector $\mathbf{d}$ and the data vector $\mathbf{x}$ are zero mean and have jointly Gaussian statistics, show that the best estimate of $\mathbf{d}$ in the mean-square sense is a linear function of the observation $\mathbf{x}$.

P3.4 Frequency Domain Expression

Develop the frequency domain expression for the power spectral density of the error signal at frequency ω, Eq. (3.34).

P3.5 Augmented Equations

Show that the Wiener-Hopf equations defining the tap-weight vector $\mathbf{w}_{\text{opt}}$ and the equation defining the minimum mean-square error ε can be combined into the single matrix equation,

$$\mathbf{R}\begin{bmatrix} 1 \\ -\mathbf{w}_{\text{opt}} \end{bmatrix} = \begin{bmatrix} \varepsilon_{\min} \\ \mathbf{0} \end{bmatrix}. \qquad \text{(P3.5-1)}$$

The matrix $\mathbf{R}$ is the correlation matrix for the vector,

$$\mathbf{x}'[n] = \begin{bmatrix} d[n] \\ \mathbf{x}[n] \end{bmatrix}, \qquad \text{(P3.5-2)}$$

where $d[n]$ is the desired signal and $\mathbf{x}[n]$ is the vector of filter input samples.

P3.6 Equalizer

Consider the equalization system shown in Fig. P3.6-1. The signal $a[n]$ is zero mean and white

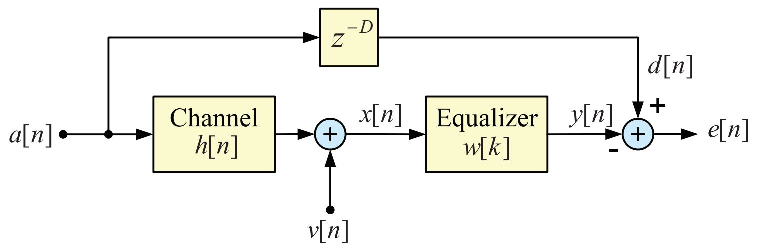


Fig. P3.6-1 Channel Equalizer

with variance σ_a^2. The noise $v[n]$ is zero mean and white with variance σ_v^2 and uncorrelated with $a[n]$. The channel response is causal and FIR with N_h terms. The equalizer has M taps.

(a) In the system diagram $d[n] = a[n-D]$. Discuss how to choose D. Suggest a suitable default value in terms of N_h and M.

(b) Set up the linear equations for finding the optimal equalize coefficients. Use vector-matrix equations of the form $\mathbf{Rw} = \mathbf{p}$, defining each of the symbols. Give an explicit form for the elements of $\mathbf{R}$ and $\mathbf{p}$.

(c) If the input signal $a[n]$ is not white, but specified by an autocorrelation $r_a[k]$ and the noise is specified by an autocorrelation $r_v[k]$, how would you determine the elements of $\mathbf{R}$ and $\mathbf{p}$?

P3.7 Quadrature Processes

Let $a[n]$ and $b[n]$ be zero-mean jointly wide-sense real stationary processes. For a constant ω, form $x[n]$ as

$$x[n] = a[n]\cos(\omega n) + b[n]\sin(\omega n). \tag{P3.7-1}$$

Find the condition on the correlations and cross-correlations of $a[n]$ and $b[n]$ such that $x[n]$ is wide-sense stationary. To do this, express the correlation $E[x[n+k]x^*[n]]$ in terms of the correlations and cross-correlations for $a[n]$ and $b[n]$ and find the relationships which make the result a function only of the time difference k.

P3.8 Quadrature Processes – Redux

This problem examines the quadrature signal in Problem P3.7, this time analyzing it using complex processes. Let $a[n]$ and $b[n]$ be zero-mean jointly wide-sense real stationary processes. Let $c[n]$ be a complex process formed as follows

$$c[n] = a[n] + jb[n]. \tag{P3.8-1}$$

For a constant ω, form $x[n]$ as

$$x[n] = \mathrm{Re}[c[n]\exp(-j\omega n)]. \tag{P3.8-2}$$

Follow the steps below.

(a) Find the correlation for $c[n]$ (i.e., $r_{cc}[k]$)in terms of the correlations and cross-correlations for $a[n]$ and $b[n]$. Is $c[n]$ wide-sense stationary?

(b) Find the correlation for $c^*[n]$ (i.e., $r_{c^*c^*}[k]$) in terms of the correlation for $r_{cc}[k]$ and ultimately in terms of the correlations and cross-correlations for $a[n]$ and $b[n]$.

(c) Find the cross-correlation of $c[n]$ and $c^*[n]$ (i.e., $r_{cc^*}[k]$) in terms of the correlations and cross-correlations for $a[n]$ and $b[n]$.

(d) Find the cross-correlation of $c^*[n]$ and $c[n]$ (i.e., $r_{c^*c}[k]$) in terms of the correlations and cross-correlations for $a[n]$ and $b[n]$..

(e) Find the correlation for $\mathrm{Re}[c[n]\exp(-j\omega n)]$ in terms of the quantities $r_{cc}[k]$, $r_{c^*c^*}[k]$, $r_{cc^*}[k]$, and $r_{c^*c}[k]$. Finally substitute the correlation values for $a[n]$ and $b[n]$.

P3.9 Modulated Signal

Consider the modulated signal (c.f. Eq. (3.20)),

$$x[n] = s[n]\cos(\omega_o n + \theta), \tag{P3.9-1}$$

where $s[n]$ is real, zero mean, and wide sense stationary.

(a) Find the form of the frequency term ω_o which makes the cosine term periodic, with period N.

(b) Show that $x[n]$ is cyclostationary (shift of N).

(c) Show that with complex exponential modulation, $x[n]$ is wide sense stationary,

$$x[n] = s[n]\exp(\omega_o n + \theta). \tag{P3.9-2}$$

What are the conditions on ω_o for this case?

P3.10 Normalized Cross-Power Spectral Density

The coherency of two wide-sense stationary signals $x[n]$ and $y[n]$ can be measured by a normalized cross-power spectral density which is given by the cross-power spectral density normalized by the power spectral densities of the individual signals,

$$C_{xy}(\omega) = \frac{S_{xy}(\omega)}{\sqrt{S_{xx}(\omega)S_{yy}(\omega)}}. \tag{P3.10-1}$$

(a) Show that the normalized cross-power spectral density satisfies $|C_{xy}(\omega)| \leq 1$.

(b) Let the cross-power spectral density of $x[n]$ and $y[n]$ be $C_{xy}(\omega)$. Consider filtering $x[n]$ with a filter $h_x[n]$ to form $x'[n]$. Similarly filter $y[n]$ with filter $h_y[n]$ to form $y'[n]$. Show that $|C_{x'y'}(\omega)| = |C_{xy}(\omega)|$.

Now consider the minimum mean-square error problem with an infinite non-causal filter with impulse response $w[n]$. The filter is chosen to minimize the mean-square value of the error,

$$e[n] = d[n] - \sum_{k=-\infty}^{\infty} w^*[k]x[n-k]. \tag{P3.10-2}$$

(c) Show the DTFT of the optimal filter $W(\omega)$ satisfies

$$W(\omega) = \frac{S_{xd}(\omega)}{S_{xx}(\omega)}. \tag{P3.10-3}$$

(d) Show that the mean-square error for the optimal filter is given by

$$\varepsilon = \frac{1}{2\pi} \int_{-\infty}^{\infty} \left(1 - |C_{xd}(\omega)|^2\right) S_{xx}(\omega)\, d\omega. \tag{P3.10-4}$$

P3.11 Correlation for Bandlimited Processes

For a bandlimited process, the correlation can be written in terms of the integral of the power spectral density,

$$r[k] = \frac{1}{2\pi} \int_{-\omega_c}^{\omega_c} S(\omega) \cos(\omega k)\, d\omega. \tag{P3.11-1}$$

We can make note of an inequality,

$$1 - \cos(\omega k) = 2\sin^2\left(\frac{\omega k}{2}\right) \leq \frac{\omega^2 k^2}{2}. \tag{P3.11-2}$$

With this show that for a process bandlimited to ω_c, the variation in the correlation is bounded as follows

$$r[0] - r[k] \leq r[0] \frac{\omega_c^2 k^2}{2}, \tag{P3.11-3}$$

or equivalently

$$\frac{r[k]}{r[0]} \geq 1 - \frac{\omega_c^2 k^2}{2}. \tag{P3.11-4}$$

Chapter 4

Linear Prediction

Linear prediction using a mean-square error criterion is sufficiently important to warrant a chapter on its own. For this chapter, we will continue assuming that the signals are wide-sense stationary.

4.1 One-Step Predictor

In its simplest guise, a predictor aims to give the best linear prediction from the past data. The desired signal is $d[n] = x[n]$. The so-called *one-step predictor* is a mean-square filter which predicts the value from immediate past data, i.e. the delays are

$$D_k = k + 1, \qquad 0 \le k < M. \tag{4.1}$$

The one-step predictor is shown in Fig. 4.1.

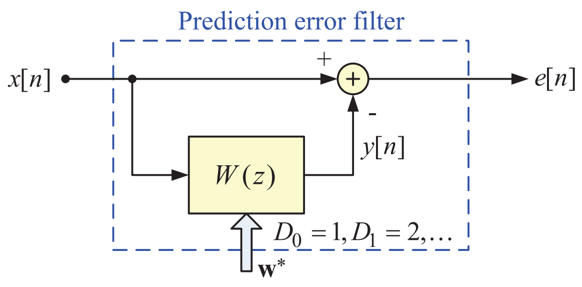


Fig. 4.1 One-step predictor

The minimum mean-square error is obtained by solving (Eq. (2.22) repeated here)

$$\mathbf{R}_{\mathbf{xx}} \mathbf{w}_{\text{opt}} = \mathbf{r}_{\mathbf{x}d}. \tag{4.2}$$

The Toeplitz correlation matrix associated with the minimum mean-square predictor is Eq. (3.9) with spacing $p = 1$,

$$\mathbf{R}_{\mathbf{xx}} = \begin{bmatrix} r_{xx}[0] & r_{xx}[1] & r_{xx}[2] & \cdots & r_{xx}[M-1] \\ r_{xx}[-1] & r_{xx}[0] & r_{xx}[1] & \cdots & r_{xx}[M-2] \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ r_{xx}[-(M-1)] & r_{xx}[-(M-2)] & r_{xx}[-(M-3)] & \cdots & r_{xx}[0] \end{bmatrix}. \tag{4.3}$$

Since for the cross-correlation vector, the desired signal is $x[n]$, we can adopt the notation $\mathbf{r}_{\mathbf{x}x} = \mathbf{r}_{\mathbf{x}d}$, where

$$\mathbf{r}_{\mathbf{x}x} = E\big[\mathbf{x}[n-1]x^*[n]\big] = \begin{bmatrix} r_{xx}[-1] \\ r_{xx}[-2] \\ \vdots \\ r_{xx}[-M] \end{bmatrix}. \tag{4.4}$$

Note that the elements $r_{xx}[-k]$ can be replaced by $r_{xx}^*[k]$. The Wiener-Hopf equations take on a special form, with the cross-correlation vector $\mathbf{r}_{\mathbf{x}x}$ sharing elements with the first row of the autocorrelation matrix. The first row of $\mathbf{R}_{\mathbf{xx}}$ has correlation values from $r_{xx}[0]$ through $r_{xx}[M-1]$, while $\mathbf{r}_{\mathbf{x}d}$ contains correlation terms $r_{xx}^*[1]$ through $r_{xx}^*[M]$. Most terms (aside from conjugation) appear in both, but the $r_{xx}^*[M]$ term only appears only in $\mathbf{r}_{\mathbf{x}x}$. The special structure of the equations allows for an efficient solution of the equations using the Durbin recursion (Appendix I). The Durbin recursion has a computational complexity proportional to M^2, rather than the order M^3 complexity when the autocorrelation matrix is not Toeplitz.

With the optimal predictor coefficients, the mean-square error (Eq. (2.25) specialized to the predictor case) is

$$\varepsilon_{\min} = r_{xx}[0] - \mathbf{w}_{\text{opt}}^H \mathbf{r}_{\mathbf{x}x}. \tag{4.5}$$

In z-transform notation, the prediction filter can be written as

$$W(z) = \sum_{k=0}^{M-1} w_k^* z^{-(k+1)}. \tag{4.6}$$

Here we have explicitly incorporated the delay values of Eq. (4.1). The transfer function relating the input $x[n]$ to the *prediction residual* (error signal) $e[n]$ is

$$A(z) = 1 - W(z). \tag{4.7}$$

This is the prediction error filter shown within the dashed box in Fig. 3.3.

4.2 Augmented Normal Equations

We can set up a set of equations that represents both the filter coefficients and the minimum mean-square prediction error. The predictor coefficients are found from the Wiener-Hopf equations Eq. (2.22). The mean-square error (specializing the first line in Eq. (2.25) for the prediction case) is

$$\varepsilon_{\text{min}} = r_{xx}[0] - \mathbf{r}_{\mathbf{x}x}^{H}\mathbf{w}_{\text{opt}}. \tag{4.8}$$

These two results can be written in terms of an augmented set of equations,

$$\begin{bmatrix} r_{xx}[0] & \mathbf{r}_{\mathbf{x}x}^{H} \\ \mathbf{r}_{\mathbf{x}x} & \mathbf{R}_{\mathbf{xx}} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_{\text{opt}} \end{bmatrix} = \begin{bmatrix} \varepsilon_{\text{min}} \\ \mathbf{0} \end{bmatrix}. \tag{4.9}$$

The matrix in the augmented set of equations is the autocorrelation matrix for a $M+1$ order predictor. The coefficient vector contains the coefficients **a** for the prediction error filter $A(z)$,

$$\mathbf{a} = \begin{bmatrix} 1 \\ -\mathbf{w}_{\text{opt}} \end{bmatrix}. \tag{4.10}$$

This expanded equations are sometimes referred to as the augmented normal equations – the "normal" equations being the Wiener-Hopf equations.

We can write the augmented normal equations as

$$\mathbf{R}_{\mathbf{xx}}^{(a)}\mathbf{a} = \begin{bmatrix} \varepsilon_{\text{min}} \\ \mathbf{0} \end{bmatrix}. \tag{4.11}$$

The augmented correlation matrix $\mathbf{R}_{\mathbf{xx}}^{(a)}$ retains the Toeplitz property.

In spite of the compact form of these combined equations, since the unknowns (the coefficient vector $\mathbf{w}_{\text{opt}}$ and the energy of the prediction residual ε_{min}) appear on opposite sides of the equation, we have to solve first for the coefficients and then use these to find the residual energy.

Figure 4.2 shows a direct form implementation of the prediction error filter using the coefficients from the vector **a**. Note that the coefficients which appear on the filter are the conjugated elements of the vector **a**.

The error signal $e[n]$ is given in the time domain as

$$e[n] = \sum_{k=0}^{M} a_k^* x[n-k] \qquad \text{with } a_0 = 1. \tag{4.12}$$

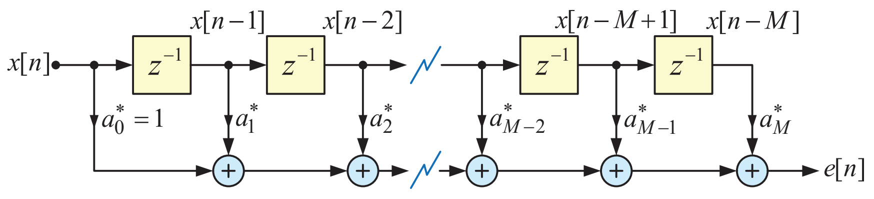


Fig. 4.2 A prediction error filter using the coefficients of the vector $\mathbf{a} = [1 - \mathbf{w}^T]^T$.

Using z-transform notation,

$$A(z) = \sum_{k=0}^{M} a_k^* z^{-k} \qquad \text{with } a_0 = 1. \tag{4.13}$$

Then

$$E(z) = A(z)X(z). \tag{4.14}$$

4.3 Properties of the Prediction Error Filter

4.3.1 Minimum-Phase Property of the Prediction Error Filter

For the optimal predictor, it can be shown that $A(z)$ is minimum-phase, i.e., it has all of its singularities inside the unit circle. A proof by contradiction is given in [29] and a proof using Rouché's Theorem is given in [13]. Another proof using properties of all-pass filters appears in [32]. We summarize that proof here.

We have seen from the Durbin recursion in Appendix I that the recursion uses the reflection coefficients k_m. These are less than unity in magnitude. The step-up procedure written in terms of the z-transform of the mth order prediction error filter is

$$A_m(z) = A_{m-1}(z) + k_m^* z^{-m} A_{m-1}^*(1/z*). \tag{4.15}$$

The proof will proceed by induction. The first order response is $A_1(z) = 1 - k_1^*$. This is clearly minimum-phase if $|k_1| < 1$. Now assume that $A_{m-1}(z)$ is minimum phase and z_i is a zero of $A_m(z)$. Since $A_m(z_i) = 0$, then from Eq. (4.15),

$$\frac{z_i^{-m} A_{m-1}^*(1/z_i^*)}{A_{m-1}(z_i)} = -\frac{1}{k_m^*}. \tag{4.16}$$

The term on the lefthand side can be associated with a causal stable all-pass filter. As shown in Appendix K, the magnitude of an all-pass filter is less than unity only if $|z_i| < 1$. Thus $A_m(z)$ is minimum-phase if $|k_m| < 1$.

Optimal Prediction Error Filters are Minimum-Phase

Any prediction error filter generated from reflection coefficients which are are less than unity in magnitude is minimum-phase. Since the Durbin recursion which finds the optimal prediction error filter produces reflection coefficients which are less than unity in magnitude, optimal prediction error filters are minimum-phase.

Any set of reflection coefficients where each is less than unity in magnitude can be converted to the set of correlations which will generate these reflection coefficients (see the inverse Durbin recursion in Appendix I). The prediction error filter corresponding to those reflection coefficients minimizes the mean-square error for that set of correlations.

An important consequence of the minimum-phase property is that the prediction error filter has a causal stable inverse $1/A(z)$. For a typical coding scenario, the setup is as shown in Fig. 4.3. The analysis filter is $A(z)$. The synthesis filter is $1/A(z)$. The figure shows a signal $q[n]$ being added before the synthesis filter. This can model, for instance, quantization error if the prediction error signal is quantized for transmission. If the synthesis filter is not stable, the output due to the signal $q[n]$ can grow, rendering the system unusable.

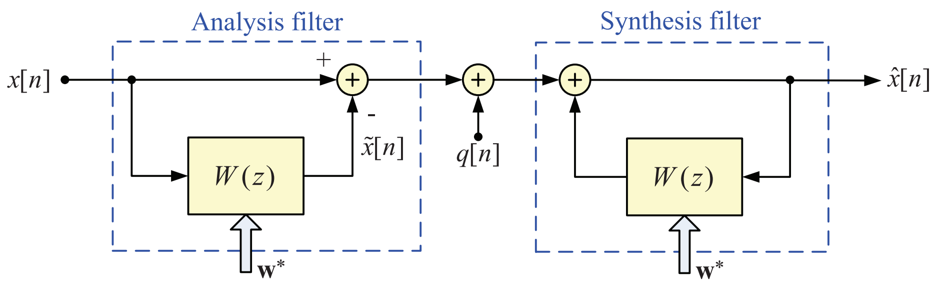


Fig. 4.3 Analysis – synthesis configuration

4.3.2 Constrained Log-Spectrum

The fact that the prediction error filter $A(z)$ has all of its singularities inside the unit circle constrains the log spectrum,

$$\int_{-\pi}^{\pi} \log\left(|A(\omega)|^2\right) d\omega = 0. \tag{4.17}$$

This is a special case of a more general result for any minimum-phase filter, see Appendix K. See Fig. 8.1 for an example.

Zero Mean Log Magnitude

The mean of the log spectrum for a causal, minimum-phase FIR filter is

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} \log\bigl(|A(\omega)|^2\bigr)\,d\omega = 2\log\bigl(|a_0|\bigr). \tag{4.18}$$

For the prediction error filter, $a_0 = 1$.

A causal stable all-pole synthesis filter is given by

$$H(z) = \frac{G}{A(z)}, \tag{4.19}$$

where $A(z)$ has all of its zeros inside the unit circle. The log spectrum of $H(\omega)$ is

$$\log\bigl(|H(\omega)|^2\bigr) = \log\bigl(|G|^2\bigr) - \log\bigl(|A(\omega)|^2\bigr), \tag{4.20}$$

we can apply the constrained log-spectrum result to give

$$\frac{1}{2\pi}\int_{-\pi}^{\pi} \log\bigl(|H(\omega)|^2\bigr)\,d\omega = 2\log\bigl(G/|a_0|\bigr). \tag{4.21}$$

4.4 Backward Prediction

We can examine a backward predictor which estimates the value of $x[n]$ from the samples $x[n+1],\ldots,x[n+M]$. Given that the data is wide-sense stationary, we would expect that the backward predictor would be closely related to the forward predictor examined above. The Wiener-Hopf equations for the backward predictor are obtained by setting the delays to $D_k = k - M$ for $0 \le k < M$. The delay values are negative, corresponding to the use of samples *after* $x[n]$ to predict $x[n]$. With this convention, the first coefficient of the backward predictor is applied to $x[n+M]$ and the last coefficient of the backward predictor is applied to $x[n+1]$,

$$\begin{aligned} y[n] &= \sum_{k=0}^{M-1} w_k^{(B)*} x[n-D_k] \\ &= \sum_{k=0}^{M-1} w_k^{(B)*} x[n+M-k]. \end{aligned} \tag{4.22}$$

The equations to be solved are then (compare with Eq. (3.5) and Eq. (3.7))

$$\begin{bmatrix} r_{xx}[0] & r_{xx}[1] & r_{xx}[2] & \cdots & r_{xx}[M-1] \\ r_{xx}[-1] & r_{xx}[0] & r_{xx}[1] & \cdots & r_{xx}[M-2] \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ r_{xx}[-(M-1)] & r_{xx}[-(M-2)] & r_{xx}[-(M-3)] & \cdots & r_{xx}[0] \end{bmatrix} \begin{bmatrix} w_0^{(B)} \\ w_1^{(B)} \\ \vdots \\ w_{M-1}^{(B)} \end{bmatrix} = \begin{bmatrix} r_{xx}[M] \\ r_{xx}[M-1] \\ \vdots \\ r_{xx}[1] \end{bmatrix}. \tag{4.23}$$

We recognize that the correlation matrix is the same as $\mathbf{R_{xx}}$ which appears in Eq. (4.3). The right-hand side vector can be written in terms of the cross-correlation vector $\mathbf{r}_{\mathbf{x}x}$ defined in Eq. (4.4) for the forward predictor,

$$\mathbf{R_{xx}}\mathbf{w}^{(B)} = \mathbf{J}\mathbf{r}_{\mathbf{x}x}^{*}. \tag{4.24}$$

The exchange matrix **J** (which flips a vector) is introduced in Appendix C.

Using the fact that $\mathbf{R_{xx}}$ is Hermitian and persymmetric (see Appendix C),

$$\begin{aligned} (\mathbf{J}\mathbf{R}_{\mathbf{xx}}^{T}\mathbf{J})\mathbf{w}^{(B)} &= \mathbf{J}\mathbf{r}_{\mathbf{x}x}^{*} \\ \mathbf{J}\mathbf{R}_{\mathbf{xx}}^{T}(\mathbf{J}\mathbf{w}^{(B)}) &= \mathbf{J}\mathbf{r}_{\mathbf{x}x}^{*} \\ \mathbf{R_{xx}}(\mathbf{J}\mathbf{w}^{(B)*}) &= \mathbf{r}_{\mathbf{x}x}. \end{aligned} \tag{4.25}$$

Comparing this equation with $\mathbf{R_{xx}}\mathbf{w} = \mathbf{r}_{\mathbf{x}x}$ for the forward predictor, we see the relationship between the backward predictor and the forward predictor is

$$\mathbf{w}^{(B)} = \mathbf{J}\mathbf{w}^{*}. \tag{4.26}$$

The coefficients of the backward predictor are flipped and conjugated versions of the coefficients of the forward predictor. The mean-squared prediction error for the backward filter is the same as for the forward filter, viz., $\varepsilon_{\min}$.

The coefficients of the backward prediction error filter **b** can be expressed in terms of the coefficients of the forward prediction error filter **a**,

$$\mathbf{b} = \mathbf{J}\mathbf{a}^{*}. \tag{4.27}$$

The augmented version of the equations is (c.f. Eq. (4.11)),

$$\mathbf{R}_{\mathbf{xx}}^{(a)}\mathbf{b} = \begin{bmatrix} \mathbf{0} \\ \varepsilon_{\min} \end{bmatrix}. \tag{4.28}$$

Figure 4.4 shows a direct form implementation of the backward prediction error filter using the coefficients from the vector **b**. Note that the coefficients which appear on the filter are the conjugated elements of the vector **b**, which in turn or the flipped and conjugated versions of the elements of the vector **a**.

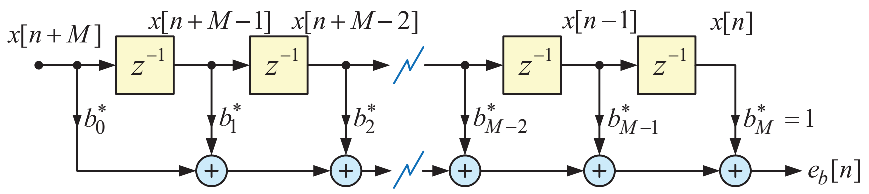


Fig. 4.4 Backward prediction error filter using the coefficients of the vector $\mathbf{b} = \mathbf{J}\mathbf{a}^*$.

The backward error signal $e[n]$ is given in the time domain as

$$e_b[n] = \sum_{k=0}^{M} b_k^* x[n+M-k] \qquad \text{with } b_M = 1. \tag{4.29}$$

Using z-transform notation,

$$B(z) = \sum_{k=0}^{M} b_k^* z^{M-k} \qquad \text{with } b_M = 1. \tag{4.30}$$

Using the fact that $b_k = a_{M-k}^*$, we can write

$$B(z) = z^{-M} A^*(1/z^*). \tag{4.31}$$

The double conjugation (once on z and again on $A(\cdot)$) has the effect of conjugating the coefficients while leaving the terms in z unconjugated. The inversion of z flips the coefficients and the z^{-M} shifts the result so that it is causal.

4.4.1 Lattice Form of the Prediction Error Filter

We have been implicitly assuming a direct form structure for the prediction error filter. A lattice form of the prediction error filter is shown in Fig. 4.5. The filter shown has two lattice stages. Additional lattice stages can be added to create a higher order filter. The coefficients that appear on the filter are the reflection coefficients that are part of the Durbin recursion of Appendix I. It is clear that the lattice structure implements an FIR filter, and furthermore that the first coefficient of the impulse response is unity (due to the the direct path across the top of the filter).

We can set up a coupled recursion to calculate the response of the filter. The forward prediction error signal is shown on the upper branch of the lattice filter. The backward prediction error signal is shown on the lower branch of the lattice. The signal $f_1[n]$ is the prediction error for an optimal one coefficient predictor. The signal $f_2[n]$ is the prediction error for an optimal two

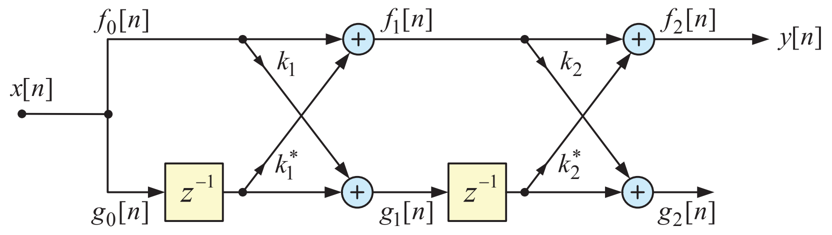


Fig. 4.5 Lattice form prediction error filter

coefficient predictor. The signals $g_m[n]$ which appear on the lower part of the lattice filter are the backward prediction error for the various order backward predictors. Note that $g_M[n]$ is the same as backward error $e_b[n]$ in Fig. 4.4.

Expressed in terms of the z-transform, the coupled recursions to go from order $m-1$ to order m are

$$\begin{aligned} F_m(z) &= F_{m-1}(z) + k_m^* z^{-1} G_{m-1}(z), \\ G_m(z) &= z^{-1} G_{m-1}(z) + k_m F_{m-1}(z). \end{aligned} \tag{4.32}$$

The initial conditions are $F_0(z) = G_0(z) = X(z)$. The response from the input to points on the upper branch will be designated as $A_m(z)$, while the response to points on the lower branch will be designated as $B_m(z)$. These also satisfy a coupled recursion,

$$\begin{aligned} A_m(z) &= A_{m-1}(z) + k_m^* z^{-1} B_{m-1}(z), \\ B_m(z) &= z^{-1} B_{m-1}(z) + k_m A_{m-1}(z). \end{aligned} \tag{4.33}$$

The initial conditions are $A_0(z) = B_0(z) = 1$. It is easily shown that responses on the lower branch are the time-reversed conjugated versions of the responses on the upper branch,

$$B_m(z) = z^{-m} A_m^*(1/z^*). \tag{4.34}$$

Equation (4.33) is the same step-up procedure (converting reflection coefficients to predictor coefficients) which appears as part of the Durbin recursion in Appendix I (Eq. (I.20)). Note that the time-reversal flips zeros inside the unit circle to outside the unit circle. The for a minimum-phase $A_m(z)$, $B_m(z)$ is maximum-phase.

In the next section it is shown that the backward error signals $g_m[n]$ are orthogonal to each other. One interpretation of the lattice implementation is that if the lattice filter uses the reflection coefficients obtained during the Durbin recursion, the filter performs a stage-by-stage optimization using a Gram-Schmidt orthogonalization.

4.4.2 Diagonalization of the Correlation Matrix

The coefficients of the mth order backward prediction error filter will be denoted as $b_{m,k}$, for $0 \leq k \leq m$. The first subscript denotes the filter order. Also note that in the implementation of the filter, it is the conjugated coefficients that are used. The error from the backward prediction error filter of order m will be denoted as $g_m[n]$, for $0 \leq k \leq m$,

$$\begin{aligned} g_0[n] &= b_{0,0}^* x[n] \\ g_1[n] &= b_{1,0}^* x[n] + b_{1,1}^* x[n-1] \\ g_2[n] &= b_{2,0}^* x[n] + b_{2,1}^* x[n-1] + b_{2,2}^* x[n-2] \\ \vdots \quad & \quad \vdots \\ g_M[n] &= b_{M,0}^* x[n] + b_{M,1}^* x[n-1] + \cdots + b_{M,M}^* x[n-M] \end{aligned} \tag{4.35}$$

The coefficients of the backward prediction error filter are the conjugated reversal of the coefficients of the forward prediction error filter, see Eq. (4.27) or Eq. (4.34). Then we can rewrite the $g_m[n]$ signals in terms of the $a_{m,k}$ coefficients,

$$\begin{aligned} g_0[n] &= a_{0,0} x[n] \\ g_1[n] &= a_{1,1} x[n] + a_{1,0} x[n-1] \\ g_2[n] &= a_{2,2} x[n] + a_{2,1} x[n-1] + a_{2,0} x[n-2] \\ \vdots \quad & \quad \vdots \\ g_M[n] &= a_{M,M} x[n] + a_{M,M-1} x[n-1] + \cdots + a_{M,0} x[n-M] \end{aligned} \tag{4.36}$$

Note that $a_{m,0} = 1$. This equation can be written in vector-matrix form as

$$\mathbf{g}[n] = \mathbf{L}_b \mathbf{x}[n], \tag{4.37}$$

where the vectors $\mathbf{g}[n]$ and $\mathbf{x}[n]$ (each $[M+1 \times 1]$) are

$$\mathbf{g}[n] = \begin{bmatrix} g_0[n] \\ g_1[n] \\ \vdots \\ g_M[n] \end{bmatrix}, \qquad \mathbf{x}[n] = \begin{bmatrix} x[n] \\ x[n-1] \\ \vdots \\ x[n-M] \end{bmatrix}, \tag{4.38}$$

and the lower triangular matrix $\mathbf{L}_b$ ($[M+1 \times M+1]$) is

$$\mathbf{L}_b = \begin{bmatrix} a_{0,0} & 0 & 0 & \cdots & 0 \\ a_{1,1} & a_{1,0} & 0 & \cdots & 0 \\ a_{2,2} & a_{2,1} & a_{2,0} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{M,M} & a_{M,M-1} & a_{M,M-2} & \cdots & a_{M,0} \end{bmatrix}. \tag{4.39}$$

Note that the diagonal of this matrix is all ones. Equation (4.37) shows that there is a one-to-one correspondence between the input vector $\mathbf{x}[n]$ and the prediction error $\mathbf{g}[n]$ for the different filter orders.

The cross-correlations for the outputs of the backward prediction error filters of different orders can be written as

$$\begin{aligned} E\big[g_i[n]g_k^*[n]\big] &= E\Big[g_i[n]\sum_{l=0}^{k} a_{k,k-l}^* x^*[n-l]\Big] \\ &= \sum_{l=0}^{k} a_{k,k-l}^* E\big[g_i[n]x^*[n-l]\big] \end{aligned} \tag{4.40}$$

The expectation in the second line is the cross-correlation between the error of an ith order backward prediction error filter and the input signal. For $k < i$, all terms of the sum are zero by the orthogonality principle (see Eq. (2.17)). For $k = i$, this correlation is just the energy of the kth order backward prediction error, i.e., ε_k. For $k > i$, we can just interchange i and k to show that the result is also zero. This analysis shows that the prediction errors for the different filter orders are orthogonal,

$$E\big[g_i[n]g_k^*[n]\big] = \begin{cases} \varepsilon_k, & \text{for } i = k, \\ 0, & \text{for } i \neq k \end{cases}. \tag{4.41}$$

Orthogonality of the Backward Prediction Errors

The triangular transformation from data to backward prediction errors gives the result that the prediction errors for optimal backward predictors of different orders are orthogonal.

Now consider the correlation matrix

$$\begin{aligned} E\left[\mathbf{g}[n]\mathbf{g}^H[n]\right] &= E\left[\mathbf{L}_b\mathbf{x}[n]\mathbf{x}^H[n]\mathbf{L}_b^H\right] \\ &= \mathbf{L}_b E\left[\mathbf{x}[n]\mathbf{x}^H[n]\right]\mathbf{L}_b^H \\ &= \mathbf{L}_b\mathbf{R}_{\mathbf{xx}}\mathbf{L}_b^H \\ &= \operatorname{diag}(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_M). \end{aligned} \tag{4.42}$$

This equation shows that the lower/upper triangular matrices diagonalize the correlation matrix.[1]

Denote the diagonal matrix of prediction mean-square errors as **E**, giving

$$\mathbf{L}_b\mathbf{R}_{\mathbf{xx}}\mathbf{L}_b^H = \mathbf{E}. \tag{4.43}$$

Then we can write the correlation matrix as

$$\mathbf{R}_{\mathbf{xx}} = \mathbf{L}_b^{-1}\mathbf{E}(\mathbf{L}_b^H)^{-1}. \tag{4.44}$$

This form expresses the inverse matrix as the the product of the inverse of a lower triangular matrix (itself lower triangular), a diagonal matrix, and the Hermitian transpose of the inverse of the lower triangular matrix. This factorization is the unique Cholesky factorization of the correlation matrix, see Appendix H. We can also express the inverse of the correlation matrix as

$$\mathbf{R}_{\mathbf{xx}}^{-1} = \mathbf{L}_b^H\mathbf{E}^{-1}\mathbf{L}_b, \tag{4.45}$$

where $\mathbf{E}^{-1} = \operatorname{diag}(1/\varepsilon_0, 1/\varepsilon_1, \ldots, 1/\varepsilon_M)$.

4.5 Linear Predictor Parameters

The process of solving for the predictor coefficients is a mapping from $M+1$ autocorrelation values to M predictor coefficients. Looking at the Wiener-Hopf equations, we see that the solution is insensitive to the scaling of the autocorrelation coefficients, so that if the autocorrelation coefficients are normalized with $r_{xx}[0] = 1$, there are really only M degrees of freedom remaining for the correlation values.

We have parameterized the prediction filter by its direct form coefficients, w_k or the equivalent coefficients in the prediction error filter, a_k. We now survey other possible parameterizations of the predictor.

[1]The correlation matrix $\mathbf{R}_{\mathbf{xx}}$ is $[M+1 \times M+1]$), and is the same as what was referred to earlier as the augmented correlation matrix, see Eq. (4.11).

4.5.1 Reflection Coefficients

The Durbin recursion described in Appendix I, uses a set of intermediate parameters, the reflection coefficients. These are an alternate representation of the filter parameters. As discussed in that appendix, for a positive definite set of equations, the reflection coefficients are bounded by unity in magnitude. From the step-down procedure Appendix I, we see that it is possible to transform a set of predictor coefficients to the corresponding set of reflection coefficients. If the resulting reflection coefficients are all bounded by one in magnitude, the prediction error filter will be minimum phase. In fact this test is the discrete-time version of the Routh-Hurwitz test for left-half plane zeros (see [13] for a discussion). The inverse Durbin procedure of Appendix I converts a set of reflection coefficients to autocorrelation values. The test on the reflection coefficients is also a test as to whether the autocorrelation sequence is positive definite. The reflection coefficients can be used to implement the prediction error filter in a lattice form with the reflection coefficients appearing as gain values in the filter as shown in Section 4.4.1.

4.5.2 Line Spectral Frequencies

Another representation is line spectral frequencies (LSFs) [25]. This is a popular parameterization of the predictor when the predictor coefficients need to be quantized for transmission. These parameters are used in virtually every low rate speech coder.

Consider the formation of two auxiliary polynomials,

$$\begin{aligned} P(z) &= A(z) + z^{-(M+1)} A^*(1/z^*), \\ Q(z) &= A(z) - z^{-M+1} A^*(1/z^*). \end{aligned} \tag{4.46}$$

Note that these are exactly the step-up equations Eq. (4.33), here going from a prediction error filter corresponding to an Mth order predictor to one corresponding to one $M+1$st order filter. For $P(z)$, the reflection coefficient is $k_{M+1} = +1$; for $Q(z)$, the reflection coefficient is $k_{M+1} = -1$. The polynomial $P(z)$ is conjugate symmetric and the polynomial $Q(z)$ is conjugate anti-symmetric.

The roots of $P(z)$ and $Q(z)$ have special properties. We modify the last step of the arguments used earlier to show that having reflection coefficients less than one in magnitude results in a minimum-phase prediction error filter. Now adding a reflection coefficient with magnitude unity results in all of the roots lying on the unity circle. Furthermore, the unit circle roots of $P(z)$ and $Q(z)$ interlace. To show this, we can rewrite Eq. (4.16) for the case at hand,

$$\frac{z_i^{-(M+1)} A^*(1/z_i^*)}{A(z_i)} = -\frac{1}{k_{M+1}^*}. \tag{4.47}$$

The magnitude of the all-pass filter on the lefthand side of the equation is unity when evaluated on the unit circle. The roots of $P(z)$ ($k_{M+1} = 1$) occur whenever the phase of the all-pass filter is an odd multiple of π. The roots of $Q(z)$ ($k_{M+1} = -1$) occur whenever the phase of the all-pass filter is an even multiple of π. In Appendix K it is shown that an all-pass filter (the lefthand side of the equation) has a monotonically decreasing phase. Thus the roots of $P(z)$ and $Q(z)$ are on the unit circle and interlace.

Because the roots lie on the unit circle, they can be uniquely described by their angular frequencies, the line spectral frequencies.[2] These LSF's can then be quantized and coded for transmission. The polynomials $P(z)$ and $Q(z)$ can be reconstructed from the LSF's, which specify the roots of these polynomials. Finally, $A(z)$ can be reconstructed from $P(z)$ and $Q(z)$.

Line Spectral Frequencies

The line spectral frequencies (LSF's) are the angular positions of the unit circle roots of the auxiliary polynomials $P(z)$ and $Q(z)$.

The LSF's can be given in radian units (most commonly) or units in terms of frequency. The trivial roots at $\omega = 0$ or $\omega = \pi$ are usually not included as part of the set of ordered LSF's. For real polynomial coefficients, there will be exactly M LSF's corresponding to roots on the upper half of the unit circle. The roots of $P(z)$ and $Q(z)$ interlace for minimum-phase prediction error filters. Lack of interlacing roots indicates a non-minimum phase filter.

4.6 Correlation Matching Property

Linear prediction has been motivated as a means to minimize the energy in the prediction residual. Linear predictors (which minimize the residual energy) also satisfy a correlation matching property. Consider the causal all-pole filter,

$$H(z) = \frac{G}{A(z)} = \frac{G}{1 - \sum_{k=0}^{M-1} w_k^* z^{-(k+1)}}. \tag{4.48}$$

The input output relationship for such a filter is

$$y[n] = Gx[n] + \sum_{k=0}^{M-1} w_k^* y[n-k-1]. \tag{4.49}$$

[2]The step-up procedure creates fixed roots at $z = +1$ and/or $z = -1$. The remaining roots have conjugate symmetry if the coefficients of $A(z)$ are real.

Now let us multiply each side by $y^*[n-m]$ and take expectations, leading to the following,

$$\begin{aligned} E\big[y[n]y^*[n-m]\big] &= GE\big[x[n]y^*[n-m]\big] + \sum_{k=0}^{M-1} w_k^* E\big[y[n-k-1]y^*[n-m]\big], \\ r_{yy}[m] &= Gr_{xy}[m] + \sum_{k=0}^{M-1} w_k^* r_{yy}[m-k-1]. \end{aligned} \tag{4.50}$$

Making use of the relations for correlations of filtered random processes given in Appendix D, we develop the result in two more steps,

$$r_{hh}[m] * r_{xx}[m] = Gh^*[-m] * r_{xx}[m] + \sum_{k=0}^{M-1} w_k^* \big(r_{hh}[m-k-1] * r_{xx}[m-k-1]\big) \tag{4.51}$$

$$Gh^*[-m] = r_{hh}[m] - \sum_{k=0}^{M-1} w_k^* r_{hh}[m-k-1]. \tag{4.52}$$

In the last step, we have eliminated $r_{xx}[m]$ which is common to all terms. The final result is an equation that involves two different representations for the all-pole filter. The first is given by the filter coefficients w_k; the second is the impulse response of the filter $h[n]$, which also determines the correlation of the impulse response $r_{hh}[m] = h[m] * h^*[-m]$.

If we apply Eq. (4.52) for $m = 0, \ldots, M$, and noting that the filter is causal ($h[n] = 0$ for $n < 0$) and that $h[0] = G$, we get vector-matrix equations of the form,

$$\mathbf{R}_{hh} \begin{bmatrix} 1 \\ -\mathbf{w} \end{bmatrix} = \begin{bmatrix} G^2 \\ \mathbf{0} \end{bmatrix}. \tag{4.53}$$

where

$$\mathbf{R}_{hh} = \begin{bmatrix} r_{hh}[0] & r_{hh}[1] & r_{hh}[2] & \cdots & r_{hh}[M] \\ r_{hh}[-1] & r_{hh}[0] & r_{hh}[1] & \cdots & r_{hh}[M-1] \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ r_{hh}[-M] & r_{hh}[-(M-1)] & r_{hh}[-(M-2)] & \cdots & r_{hh}[0] \end{bmatrix}. \tag{4.54}$$

This vector-matrix equation is exactly of the same form as the augmented normal equations of Eq. (4.9) for the optimal predictor. As we know from Appendix I, there is a one-to-one correspondence between the coefficients and the correlations which were used to solve for the coefficients. If the all-pole filter uses the coefficients which minimize the mean-square prediction error and G^2 is set equal to the energy of the prediction residual $\varepsilon_{\min}$, we get that the correlation of the all-pole filter is the same as that of the original signal for lags 0 through M. The set of linear equations which find the coefficients of an all-pole filter (the output is an auto-regressive process) is known as the Yule-Walker equations (here augmented as for the Wiener-Hopf equations) [13].

Correlations for an All-Pole Synthesis Filter:

Consider an all-pole filter with coefficients derived from an optimal predictor, with gain $G = \sqrt{\varepsilon_{\min}}$. The output of that filter, when driven by zero-mean unit-variance white noise has the same correlation values as the original signal used to derive the filter coefficients for lags 0 through M,

$$r_{hh}[k] = r_{xx}[k], \qquad 0 \le |k| \le M. \tag{4.55}$$

It is however to be noted that this correlation matching occurs only for the $M+1$ correlation values. The recursion in Eq. (4.52) can be used to compute the autocorrelation values of the output signal for lags $M+1$ and larger, given just the correlations for the first $M+1$ lag values. These higher order lag correlations will generally not match those of the signal used to determine the coefficients.

There are approaches (see [24]) which allow for a weighted match to a larger number of correlation terms. For M coefficients, one tries to match more than M correlation values. This is not possible exactly, so a weighting allows one to approximately match the larger set.

4.7 Prediction Gain

The efficacy of a linear predictor can be measured in terms of the prediction gain which is the ratio of the energy of the input signal to the energy of the prediction residual. For an optimal predictor, this is always greater than or equal to one. The prediction gain is equal to one if the the input is uncorrelated, i.e. $r_{xx}[k] = 0$, for $1 \le |k| < M$.

4.7.1 Upper Bound on the Prediction Gain

The prediction gain can be upper-bounded in terms of a spectral flatness measure. First we make note of the fact that the correlation function $r_{xx}[k]$ has as a Fourier transform the power spectral density $S_{xx}(\omega)$. The spectral flatness measure is

$$\gamma_x^2 = \frac{\exp\Bigl(\dfrac{1}{2\pi}\displaystyle\int_{-\pi}^{\pi} \log\bigl(S_{xx}(\omega)\bigr)\, d\omega\Bigr)}{\dfrac{1}{2\pi}\displaystyle\int_{-\pi}^{\pi} S_{xx}(\omega)\, d\omega}. \tag{4.56}$$

The denominator is recognized as $r_{xx}[0]$, the (arithmetic) mean of the power spectral density. The numerator is the geometric mean of the power spectral density. This can be seen from a discrete approximation to the numerator,

$$\begin{aligned}\exp\Big(\frac{1}{2\pi}\int_{-\pi}^{\pi}\log\big(S_{xx}(\omega)\big)\,d\omega\Big) &\approx \exp\Big(\frac{1}{2\pi}\sum_{k=0}^{N-1}\log\big(S_{xx}(\frac{2\pi k}{N})\frac{2\pi}{N}\Big)\\ &= \prod_{k=0}^{N-1}\Big(S_{xx}(\frac{2\pi k}{N})\Big)^{1/N}.\end{aligned} \tag{4.57}$$

The geometric mean is upper-bounded by the arithmetic mean. This means that the spectral flatness measure γ_x^2 lies between 0 and 1. A spectral flatness of one corresponds to a flat (white) spectrum.

The prediction gain for an optimal *infinite* length linear predictor is the inverse of the spectral flatness measure,

$$P_{G\max} = \frac{\dfrac{1}{2\pi}\displaystyle\int_{-\pi}^{\pi} S_{xx}(\omega)\,d\omega}{\exp\Big(\dfrac{1}{2\pi}\displaystyle\int_{-\pi}^{\pi}\log\big(S_{xx}(\omega)\big)\,d\omega\Big)}. \tag{4.58}$$

Since the numerator is the energy in the input signal, the denominator is the energy of the prediction residual for an optimal infinite length predictor. If the power spectral density is zero over an interval, the integral in the denominator evaluates to $-\infty$ and the denominator evaluates to zero. In such a case, for a finite length predictor, the prediction residual energy approaches zero as the predictor length increases. By contrast, if the power spectral density is flat (uncorrelated data), the prediction gain is 1.

Note: Asymptotic Prediction Gain

The derivation of the asymptotic prediction gain can be linked to the asymptotic behaviour of Toeplitz matrices as described in [11].

4.7.2 Error Whitening Effect

The prediction error filter whitens the input signal. If the input signal is already white, the prediction error filter does nothing, i.e., all of its coefficients are zero, since the righthand side vector in Eq. (4.4) for that case is zero.

A general model of stationary stochastic processes, the Wold decomposition [31], states that a stationary process can be decomposed into a predictable (with zero error) and a general linear process. An example of an exactly predictable process is a sinusoid. The general linear process can be created by filtering uncorrelated white noise. It is the filter which creates the correlations in the output signal. Then our prediction error filter is trying to identify the filter used to create the process and remove its effect. To the extent that the predictor order is not large enough to remove all of correlation effects, the prediction residual is not entirely white, but whiter than the input signal.

The power spectral density at the output of the prediction error filter is

$$S_{yy}(\omega) = S_{xx}(\omega)|A(\omega)|^2. \tag{4.59}$$

Now we can relate the spectral flatness of the error to the spectral flatness of the input,

$$\frac{\gamma_e^2}{\gamma_x^2} = \frac{\exp\Bigl(\dfrac{1}{2\pi}\displaystyle\int_{-\pi}^{\pi} \log\bigl(S_{xx}(\omega)|A(\omega)|^2\bigr)\,d\omega\Bigr)}{\exp\Bigl(\dfrac{1}{2\pi}\displaystyle\int_{-\pi}^{\pi} \log\bigl(S_{xx}(\omega)\bigr)\,d\omega\Bigr)} \times \frac{\dfrac{1}{2\pi}\displaystyle\int_{-\pi}^{\pi} S_{xx}(\omega)\,d\omega}{\dfrac{1}{2\pi}\displaystyle\int_{-\pi}^{\pi} S_{ee}(\omega)\,d\omega}. \tag{4.60}$$

Note that the first fraction is unity since the integral of the log of $|A(\omega)|^2$ is zero (see Eq. (4.17)). The second fraction is the prediction gain. This shows that minimizing the mean-square error (maximizing the prediction gain) also maximizes the spectral flatness of the error.

Error Whitening

Minimizing the prediction error is equivalent to maximizing the spectral flatness measure.

4.7.3 Singular Correlation Matrices

A matrix (here $\mathbf{R}$) is singular if for some non-zero vector $\mathbf{v}$, $\mathbf{R}\mathbf{v} = \mathbf{0}$. Pre-multiplying by $\mathbf{v}^H$,

$$\mathbf{v}^H\mathbf{R}\mathbf{v} = 0, \qquad \text{for } \mathbf{v} \neq \mathbf{0}. \tag{4.61}$$

Since this quadratic form yields a zero value, $\mathbf{v}$ exercises the zero mode of "non-negative" definite.

Consider an input signal $x[n]$ applied to an FIR filter with coefficients given by $\mathbf{v}$. The output of the filter is

$$y[n] = \mathbf{v}^H\mathbf{x}[n],, \tag{4.62}$$

The mean-square value of the output signal gives rise to the quadratic form above,

$$E[|y[n]|^2]] = \mathbf{v}^H\mathbf{R}_{\mathbf{xx}}\mathbf{v}. \tag{4.63}$$

If the mean-square error is zero for any non-zero filter $\mathbf{v}$, the matrix $\mathbf{R}_{\mathbf{xx}}$ is non-negative definite, including the zero case. Problem P4.9 explores the situation where the signal $x[n]$ is the sum of complex exponential signals. These complex exponential signals have delta functions in the power spectral density ((see Section 3.5). Such spectra can be described as line spectra. A zero output signal results if the FIR filter $\mathbf{v}$ has zeros at the locations of those delta functions. If we consider

v as a prediction error filter, this shows that a signal with a line spectrum is *perfectly predictable*. In fact since an FIR filter can only annihilate an input power spectral density at a finite number of frequencies, the singular case arises only if the input signal has a line spectrum. For a line spectrum, the power spectral density is zero over an interval, so the prediction gain is infinite. This case also has echoes back to the discussion in Section 4.7.2 on the Wold decomposition; the line spectrum is due to an exactly predictable process.

4.8 Lattice Form of the Synthesis Filter

In Section 4.4.1 (Fig. 4.5), we saw that the prediction error filter could be implemented as an FIR lattice filter with the reflection coefficients appearing as the lattice coefficients. The all-pole synthesis filter can also be implemented as a lattice filter. If we look at the lattice filter in Fig. 4.5, we see that $f_m[n]$ appears across the top of the lattice. If we can instead create $f_0(z)$ from $f_M(z)$, we have implemented the inverse. To go one step down, we rewrite the coupled recursion equations of Eq. (4.32) to move $F_{m-1}(z)$ to the lefthand side of the first equation,

$$\begin{aligned} F_{m-1}(z) &= F_m(z) - k_m^* z^{-1} G_{m-1}(z), \\ G_m(z) &= z^{-1} G_{m-1}(z) + k_m F_{m-1}(z). \end{aligned} \tag{4.64}$$

If we implement these relationships, we get one stage of an all-pole lattice filter as shown in Fig. 4.6. Across the top each stage has $f_m[n]$ as input on the left and $f_{m-1}[n]$ as an output on the right. Across the bottom $g_{m-1}[n]$ enters on the right and $g_m[n]$ exits on the left. At the right hand side of the filter, $g_0[n]$ is set equal to $f_0[n]$.

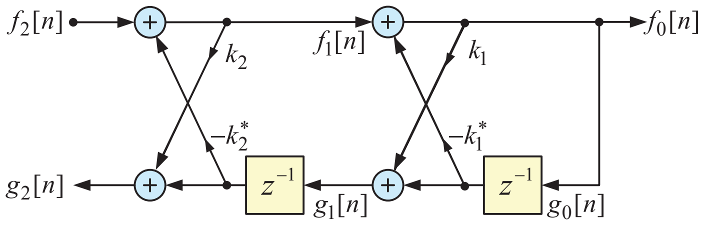


Fig. 4.6 An all-pole lattice filter

We can note that that the transfer function from the input at the top left of the figure to the top right is the all-pole response $1/A_M(z)$. The transfer function from the top right to the bottom left is that of the backward prediction error filter $B_M(z) = z^{-M} A_M^*(1/z^*)$. That means that the transfer function from the top left to the bottom left is an all-pass response.

4.9 Joint Process Estimation

A prediction error filter solves the prediction problem in that it minimizes the mean-square prediction error. Consider a more general filter for minimizing the mean-square error between a desired signal (no longer the same as the input to the filter) and the filter output. This more general filter will be a causal filter that uses the lattice form of the prediction error filter as a subsystem. The coefficients of the lattice subsystem are the reflection coefficients obtained by solving the Durbin recursion based on the statistics of the input signal. The configuration is shown in Fig. 4.7.

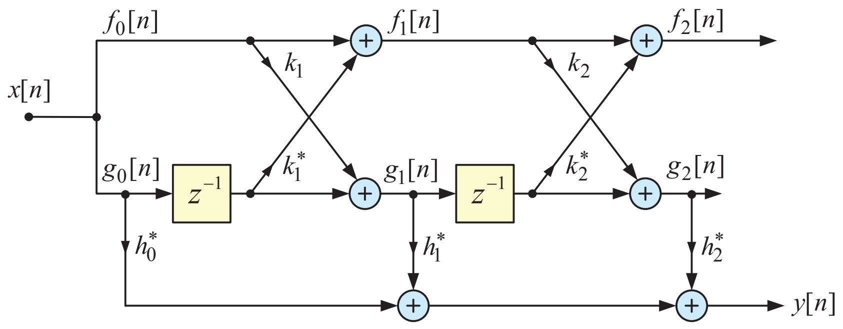


Fig. 4.7 A Joint Process Estimator

On can note that the signals $g_m[k]$ which appear across the lower part of the lattice filter are uncorrelated with each other, see Section 4.4.2. The lattice implementation provides a set of uncorrelated signals that can be used for further processing. The overall filter as shown is no longer a predictor since the input to the first coefficient is not delayed. Note that $g_m[n]$ is a linear combination of $x[n], x[n-1], \ldots, x[n-m]$. So for an M stage lattice, the joint process estimation filter will implement a linear combination of $x[n], x[n-1], \ldots, x[n-M]$.

Earlier we met the case of a bank of filters operating on multiple input signals, see Section 2.5. In a joint process estimation filter, the multiple input signals are the $g_m[n]$ signals. Each filter consists of just a single coefficient. When we go to solve for the optimal filter coefficients, the uncorrelatedness of the $g_m[n]$ signals means that we can solve for each coefficient independently – there is no cross coupling.

The filter output for a filter with $M+1$ coefficients can be written as

$$y[n] = \sum_{k=0}^{M} h_k^* g_k[n]. \tag{4.65}$$

We can write this using vectors as

$$y[n] = \mathbf{h}^H \mathbf{g}[n]. \tag{4.66}$$

The error with respect to the desired signal is

$$e[n] = d[n] - \mathbf{h}^H \mathbf{g}[n]. \tag{4.67}$$

Applying the principles of the minimum mean-square error filtering developed earlier, we can find the optimum filter coefficients from the Wiener-Hopf equation,

$$\mathbf{R}_{gg}\mathbf{h} = \mathbf{r}_{\mathbf{g}d}. \tag{4.68}$$

From earlier results (Eq. (4.42)), we know that

$$\mathbf{R}_{gg} = \operatorname{diag}(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_M) = \mathbf{E}. \tag{4.69}$$

Also from Eq. (4.37) $\mathbf{g}[n] = \mathbf{L}_b \mathbf{x}[n]$, so

$$\mathbf{r}_{\mathbf{g}d} = \mathbf{L}_b \mathbf{r}_{\mathbf{x}d}. \tag{4.70}$$

Then Eq. (4.68) becomes

$$\mathbf{E}\mathbf{h} = \mathbf{L}_b \mathbf{r}_{\mathbf{x}d}. \tag{4.71}$$

The coefficients which appear on the joint process estimation filter can be determined as follows.

1. Calculate or estimate the correlation values $r_{xx}[0], \ldots, r_{xx}[M]$.
2. Solve the Mth order Durbin recursion. Keep the reflection coefficients, predictor coefficients, and the resulting mean-square error for each order $k = 0, \ldots, M$.
3. Set $\mathbf{E}$ to be the $[M+1 \times M+1]$ diagonal matrix of the mean-square errors for each order $(\varepsilon_0, \ldots, \varepsilon_M)$.
4. Fill in the $[M+1 \times M+1]$ lower triangular matrix $\mathbf{L}_b$ with the prediction error filter coefficients (Eq. (4.39)).
5. Calculate or estimate the cross-correlation vector $([M+1 \times 1])$ $\mathbf{r}_{\mathbf{x}d}$.
6. Solve $\mathbf{E}\mathbf{h} = \mathbf{L}_b \mathbf{r}_{\mathbf{x}d}$ or equivalently solve $\mathbf{h} = \mathbf{E}^{-1}\mathbf{L}_b \mathbf{r}_{\mathbf{x}d}$.

The output of the joint process filter is

$$\begin{aligned} y[n] &= \mathbf{h}^H \mathbf{g}[n] \\ &= \mathbf{h}^H \mathbf{L}_b \mathbf{x}[n]. \end{aligned} \tag{4.72}$$

An equivalent direct form filter with coefficients

$$\mathbf{w} = \mathbf{L}_b^H \mathbf{h}, \tag{4.73}$$

will implement the same transfer function.

4.9.1 Order Recursive Computation

Due to the orthogonalization of the backwards error on the lattice part of the joint process estimation filter, the optimal coefficients h_k can be solved for individually. This also means that the calculation of h_m can be done after a Durbin recursion has estimated the optimal predictor of order m. Using the fact that row m of $\mathbf{L}_b$ has coefficients $\mathbf{a}_m^H \mathbf{J}$,

$$h_m = \frac{1}{\varepsilon_m} \mathbf{a}_m^H \mathbf{J} \mathbf{r}_{\mathbf{x}d}^{(m+1)}, \tag{4.74}$$

where the notation $\mathbf{r}_{\mathbf{x}d}^{(m)}$ denotes the m element subvector of $\mathbf{r}_{\mathbf{x}d}$.

4.9.2 Relationship to the Levinson Algorithm

The procedure developed for the joint process estimator is related to the Levinson algorithm. We note that the joint process estimator implements a causal filter, while the Levinson algorithm can solve for a filter with a delay offset. Consider the Levinson algorithm for a causal filter. The coefficient h_m in Eq. (4.74) is equal to the α variable in the first step of the Levinson algorithm of Appendix J.

To get the coefficients of the filter which minimizes the mean-square error, use Eq. (4.73). If this is calculated in an order recursive fashion, the update from a filter with $m-1$ coefficients to one with m coefficients is the second step in the Levinson algorithm.

4.10 Other Predictor Forms

4.10.1 Signal Plus Noise

A generalization of the one-step predictor arises if the signal $x[n]$ is $d[n]$ plus added noise. This is the model we use in one of our later examples (DPCM coding in Section 8.2). If the noise is uncorrelated with $d[n]$, the addition of the noise only affects the autocorrelation matrix,

$$r_{xx}[k] = r_{dd}[k] + r_{nn}[k], \tag{4.75}$$

where $r_{dd}[k]$ and $r_{nn}[k]$ are the correlation terms for the desired signal and noise respectively. If in addition the noise is white, only $r_{xx}[0]$ which is on the diagonal of the autocorrelation matrix, is affected.

4.10.2 Equally Spaced Predictor Coefficients

The one-step predictor has the property that the prediction error filter using the optimal coefficients is minimum-phase. The one-step predictor was defined as having the coefficient delays equally spaced, $D_k = k+1$, $k = 0, \ldots, M-1$. A slightly more general form with coefficient delay spacing p also results in minimum-phase prediction error filter. That this is so can be argued from the fact that the correlation equations are similar in form as for $p = 1$, with the correlation term k being replaced by correlation term kp. This is a manifestation of the result that the correlation for a subsampled sequence is the subsampled correlation.

4.10.3 Non-Minimum Phase Prediction Error Filters

The optimal prediction error filter is not necessarily minimum phase property if the coefficient delays are not equally spaced. This result is noted for the "multi-step" predictor in [29]. A one-step predictor can also give rise to a non-minimum phase prediction error filter if some coefficients are missing or equivalently, fixed to zero.

We can show examples of cases in which simple predictors are non-minimum phase. In these examples, the correlation sequences are positive definite (valid correlation values).

- For the first example, $D_0 = 1$ and $D_1 = 3$. This is a one-step predictor, but with the coefficient at delay 2 constrained to be zero. The equations take on the form shown below,

$$\begin{bmatrix} r_{xx}[0] & r_{xx}[2] \\ r_{xx}[-2] & r_{xx}[0] \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} r_{xx}[-1] \\ r_{xx}[-3] \end{bmatrix}. \tag{4.76}$$

  With correlation values $r_{xx}[0] = 1$, $r_{xx}[1] = -0.32$, $r_{xx}[2] = -0.7$, and $r_{xx}[3] = -0.8$, the optimal predictor is non-minimum phase.

- For the second example, $D_0 = 2$ and $D_1 = 3$. This is a two-step predictor (or a one-step predictor with the coefficient at delay 1 constrained to be zero). The equations take on the form shown below,

$$\begin{bmatrix} r_{xx}[0] & r_{xx}[1] \\ r_{xx}[-1] & r_{xx}[0] \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} r_{xx}[-2] \\ r_{xx}[-3] \end{bmatrix}. \tag{4.77}$$

  With correlation values $r_{xx}[0] = 1$, $r_{xx}[1] = -0.87$, $r_{xx}[2] = 0.55$, and $r_{xx}[3] = -0.13$, the optimal predictor is non-minimum phase.

We note that for these examples the correlation matrix is positive-definite (as it should be) and is Toeplitz. The nature of the correlation matrix by itself does not determine whether the optimal predictor is non-minimum phase.

There might be a question as to why the argument given earlier to show that the optimal prediction error filter is minimum phase does not apply here also. The answer is that there is indeed a minimum-phase prediction error filter which gives a lower mean-square error, but that filter does not have one or more of its coefficients constrained to be zero.

Finally, there are cases with some coefficients constrained to be zero which have optimal prediction error filters which are always minimum phase. There are the cases where the coefficients are equally spaced, $D_k = (k+1)p$, for $k = 0, \ldots, M-1$ (coefficients at the intermediate delays can be considered to be constrained to be zero). For such a filter, the output at a given "phase" depends only on input samples at the same "phase". The p prediction error filters operate in round-robin fashion. Each operates on a subsampled sequence and each has the same optimal coefficients. The prediction error filter is the same as a one-step predictor operating on a subsampled sequence.

4.11 Notes on Positive Definite Correlations

We end this chapter with some notes on correlation sequences. For the predictors with M coefficients, we need access to the $M+1$ correlations $r[0]$ through $r[M]$ to solve the Wiener-Hopf equations. This, of course, does not mean that the correlations for lags larger than M are zero.

The power spectral density $S(\omega)$ corresponding to a correlation sequence can be calculated from the *full* correlation sequence using a DTFT. The power spectral density is real and positive. It is this positivity that can be associated with the positive definite property of the correlation. All "proper" correlations are positive definite.

4.11.1 Correlation Windows

The correlation sequence can be windowed,

$$r_w[k] = w_r[k]r[k]. \tag{4.78}$$

The lag window $w_r[k]$ should be symmetric about lag 0 so that its DTFT is real. The effect of windowing the correlation is to convolve the spectrum of the windows with the power spectral density. Appropriate windows should then have a non-negative spectrum so that the convolution results in a new positive power spectral density which has as its inverse DTFT a positive definite correlation sequence. Examples of appropriate lag windows include triangular and Gaussian windows. Both of these have real non-negative DTFTs.

Consider a truncated correlation sequence. Such a truncated sequence can be considered to be a windowed correlation sequence, where the correlation window is rectangular. The DTFT of a rectangular window is not positive. This means that the truncated correlation sequence is not necessarily a proper positive definite sequence since its DTFT may go negative.

4.11.2 Correlation Subsequences

If we subsample a correlation sequence (starting at 0), the result is also a "proper" correlation sequence.

4.11.3 Correlation Matrices

When we fill in a $[M \times M]$ Toeplitz correlation matrix $\mathbf{R}$, we use only only the first M correlations (lags 0 to $M-1$). If these correlations can be extended to become a full positive definite sequence, the matrix is positive definite,

$$\mathbf{a}^H \mathbf{R} \mathbf{a} > 0. \tag{4.79}$$

This result is true for any non-zero $\mathbf{a}$. An equivalent condition is that all of the eigenvalues of $\mathbf{R}$ are all real and positive. Positivity of the eigenvalues also ensures that the determinant is real and positive.

For a positive definite matrix, all submatrices (starting from the upper left corner) are also positive definite. This result is implicit in the Durbin recursion, since it builds its solution order by order. The reflection coefficients used in the Durbin recursion will only be less than one in magnitude if the correlation matrix is positive definite.

Problems

P4.1 Residual Error

Show that the residual energy for a prediction error filter (not necessarily the one which minimizes the mean-square error) can be expressed as

$$\varepsilon = \mathbf{a}^H \mathbf{R} \mathbf{a}, \tag{P4.1-1}$$

where $\mathbf{a}$ is the vector of the coefficients of the prediction error filter.

P4.2 Prediction Error Filter

The prediction error filter (real coefficients) for an order $m-1$ system is $\mathbf{a}_{m-1}$. Assume that this filter is minimum-phase. The reflection coefficient k_m will be used to generate the prediction error filter for order m. Hint: For a monic polynomial (constant coefficient of unity), the last coefficient is the product of the roots.

(a) Show that if $|k_m| = 1$, the resultant prediction error filter is linear phase.

(b) Show that if $|k_m| = 1$, all of the singularities of the order m prediction error filter lie on the unit circle.

(c) Show that the singularities for the prediction error filter created with $k_m = 1$ interlace with the singularities created with $k_m = -1$.

(d) Show that if $|k_m| > 1$, at least one singularity lies outside the unit circle.

P4.3 Augmented Normal Equations

The augmented normal equations in Eq. (4.9) can be solved using a Lagrange multiplier approach. Note that the **a** vector has one fixed coefficient. Develop a procedure for solving the augmented normal equations by adding a constraint on the first coefficient of **a** to the error expression developed in Problem P4.1.

P4.4 Gram-Schmidt Orthogonalization

Show that the lattice filter implements a Gram-Schmidt orthogonalization. Express the lattice filter coefficients in terms of the inner product which appears in the description of the Gram-Schmidt procedure.

P4.5 Truncated Autocorrelation

Consider an autocorrelation sequence $r[k]$. Since this is an autocorrelation sequence, the DTFT of $r[k]$ is real and non-negative (a power spectral density). A lag window $w[k]$ with DTFT $W(\omega)$ is applied to the sequence.

(a) Find sufficient conditions on the DTFT $W(\omega)$ such that the DTFT of the lag-windowed sequence $w[k]r[k]$ is a valid autocorrelation sequence, i.e., that the power spectral density corresponding to the lag-windowed correlation is real and non-negative.

(b) Consider a symmetrical rectangular lag window. Applying this lag window will truncate the autocorrelation sequence. Note that the DTFT of this lag window is not non-negative. Is this truncated autocorrelation sequence always a valid autocorrelation sequence? If the answer is yes, prove the result. If the answer is no, provide a simple counter example.

(c) Explain why the power spectral density corresponding to the truncated autocorrelation sequence (known as the truncated periodogram) is different from the power spectral density of the inverse of the prediction error filter found by minimizing the mean-square error. What is the relationship between the correlation values for the two cases?

P4.6 Leaky Steepest Descent Algorithm

The mean-square error for a predictor can be written as (see Eq. (2.5))

$$\varepsilon = r_{xx}[0] - 2\mathrm{Re}[\mathbf{w}^H \mathbf{r}_{xx}] + \mathbf{w}^H \mathbf{R}_{\mathbf{xx}} \mathbf{w}. \tag{P4.6-1}$$

To find the predictor coefficients $\mathbf{w}$ which minimize the mean-square error, we took the derivative of ε with respect to $\mathbf{w}$. The derivative was given by (see Eq. (2.17))

$$\frac{d\varepsilon}{d\mathbf{w}} = -2E\big[e^*[n]\mathbf{x}[n]\big]. \tag{P4.6-2}$$

(a) Show that the derivative can also be written as

$$\frac{d\varepsilon}{d\mathbf{w}} = -2(\mathbf{R}_{\mathbf{xx}}\mathbf{w} - \mathbf{r}_{xx}). \tag{P4.6-3}$$

The steepest descent algorithm is a procedure for iteratively solving a set of linear equations. Here the set of linear equations is $\mathbf{R}_{\mathbf{xx}}\mathbf{w} = \mathbf{r}_{xx}$. The steepest descent algorithm updates the weight vector as follows.

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \mu(\mathbf{r}_{xx} - \mathbf{R}_{\mathbf{xx}}\mathbf{w}^{(k)}). \tag{P4.6-4}$$

(b) Show that for a small positive μ, at each step the steepest descent algorithm reduces the mean-square error.

A vector norm and a matrix norm are compatible when

$$\|\mathbf{A}\mathbf{x}\| \le \|\mathbf{A}\|\,\|\mathbf{x}\|. \tag{P4.6-5}$$

Here we will use the Euclidian norm for vectors, $\|\mathbf{x}\| = \mathbf{x}^H\mathbf{x}$. For matrices we use the 2-norm (spectral norm), $\|\mathbf{A}\| = \lambda_{\max}(\mathbf{A})$.

(c) Find the 2-norm of $\mathbf{A} + \mu\mathbf{I}$ in terms of the 2-norm of $\mathbf{A}$.

(d) Using the properties of the vector/matrix norms, find the range of μ which guarantee a monotonically decreasing tap weight error.

A leaky version of the steepest descent algorithm is as follows

$$\mathbf{w}^{(k+1)} = \alpha \mathbf{w}^{(k)} + \mu(\mathbf{r}_{\mathbf{x}x} - \mathbf{R}_{\mathbf{xx}}\mathbf{w}^{(k)}), \tag{P4.6-6}$$

where α is usually a value just below unity. This leaky algorithm can be shown to minimize a modified error criterion which is given by ε plus an additional term.

(e) Find the error criterion minimized by the leaky steepest descent algorithm. Interpret the result in two ways. First as the error term ε augmented with a term which is a constraint using a Lagrange multiplier, and second as a modification of the correlation matrix which augments the diagonal of the matrix (known as ridge regression in the matrix computation literature, or white noise compensation in the case of prediction).

(f) Find a version of the leaky steepest descent algorithm which corresponds to compensation with coloured noise, i.e. noise with a more general set of correlation values given by $r_{nn}[k]$.

P4.7 Signal Plus Noise

In Section 4.10.1, the case of predicting a noisy signal was discussed. As pointed out there, uncorrelated white noise only affects the correlation value $r_{xx}[0]$. Let the variance of the zero-mean noise be σ^2.

(a) For the case of white noise affecting $x[n]$, can one still use the Durbin recursion to find the predictor coefficients?

Now consider coloured noise formed by passing white noise through a filter

$$H(z) = 1 + \alpha z^{-1}. \tag{P4.7-1}$$

(b) If we denote the filtered noise as $\zeta[n]$, what is the correlation for $\zeta[n]$?

(c) Show the form of the correlation matrix $\mathbf{R}_{\zeta\zeta}$.

(d) Show the form of the correlation matrix for signal plus noise (assuming that the noise is independent of the signal).

(e) What is the cross-correlation vector between the signal $\mathbf{x}[n-1]$ and the desired signal ($d[n] = x[n]$)?

(f) For the case of coloured noise, are the matrix and vector which appear in the Wiener-Hopf equation of a form that is appropriate for the use of the Durbin recursion, or do we have to resort to using the more general Levinson algorithm to solve the prediction equations?

P4.8 Partial Correlation Coefficients

The reflection coefficients obtained from the Durbin recursion are also known as the partial correlation coefficients. This nomenclature results from the fact that k_m can be expressed as

$$k_m = \frac{E\left[f_{m-1}[n] g^*_{m-1}[n-1]\right]}{\sqrt{E\left[|f_{m-1}[n]|^2\right] E\left[|g_{m-1}[n-1]|^2\right]}}. \tag{P4.8-1}$$

This fraction is the normalized cross-correlation between the forward and backward errors at the input to stage m on the lattice filter shown in Fig. 4.5.

(a) Assume that all of the previous reflection coefficients have been chosen to minimize the mean-square error, show that the denominator in the expression for k_m is ε_{m-1}.

(b) Show that the reflection coefficient calculated using the normalized cross-correlation is the same as that calculated in the Durbin recursion.

(c) The expression for the mth reflection coefficient minimizes the prediction error at the output of the mth stage whether or not the preceding reflection coefficients are the optimal ones. Given this fact, explain how using this formulation can be useful when the reflection coefficients are calculated and quantized stage-by-stage.

P4.9 Complex Exponential Signals

Consider the sum of N complex exponentials,

$$x[n] = \sum_{k=0}^{N-1} A_k \exp(j\omega_k n), \tag{P4.9-1}$$

with the ω_k values being fixed distinct values.

(a) Consider first the case of a single sinusoid ($N = 1$). Find the form of an $[M \times M]$ correlation matrix for this signal.

(b) What is the largest value of M for which the correlation matrix is non-singular?

(c) Find the coefficients of the optimal predictor of minimum order which predicts the signal with zero error. What is the frequency response of this filter?

Now consider a frequency domain view of the exponential signals.

(d) Find the DTFT of the M sinusoids. This will be a series of M delta functions, see [23].

(e) Now consider a prediction error filter $A(z)$. Find the number and locations of an appropriate number of zeros of that filter such that the prediction error becomes zero. Use the zero locations to find an expression for $A(z)$ in terms of its root factors. This shows that an Nth order prediction system can perfectly predict a signal consisting of N exponentials. Is the prediction error filter minimum-phase? If not, reconcile that fact with the results of Section 4.3.1.

(f) Show that if the predictor is of order M larger than N, there are an infinity of $A(z)$ filters which set the error to zero. What does this say about singularity/non-singularity of the $[M \times M]$ correlation matrix for the signal? Does adding a phase term to each exponential change the solution?

(g) Now consider the sum of N cosines (with frequencies ω_k and phases θ_k). What is the minimum order for the predictor to be able to exactly predict the signal?

P4.10 Joint Process Estimator as a Predictor

In this problem the joint process estimator filter will appear as a subblock of a predictor filter. The input to the joint process filter will be a delayed version of a signal $\tilde{x}[n]$, i.e.

$$x[n] = \tilde{x}[n-1]. \tag{P4.10-1}$$

In the overall filter, the desired signal is $d[n] = x[n]$.

(a) Show that $\mathbf{R}_{xx} = \mathbf{R}_{\tilde{x}\tilde{x}}$.

(b) Show that $\mathbf{r}_{\mathbf{x}d} = \mathbf{r}_{\tilde{\mathbf{x}}d}$.

(c) Find an expression for h_m in terms of $r_{xx}[k]$ and ε_m.

(d) Show that the overall filter response ($\mathbf{w}$) is the same as that for a predictor with $M+1$ coefficients.

Chapter 5

Least Squares Method

With least-squares filters, we seek to minimize the sum of squared errors over an interval of time. The interval of time is selected with a set of windows. In the general setup, windows can be applied to the input signal (a data window), the desired signal (another data window), and/or to the resulting error (error window).

The combination of windows is used to analyze the signal and determine a set of filter coefficients. This is the analysis stage. Once the filter coefficients have been determined, they are applied to a "frame" of data, usually without tapered windowing. Thus the analysis and filtering steps will in general use different data: the analysis stage uses windowed data and the filtering stage uses the original unwindowed data.

In least squares scenarios, the windows advance in steps, with the data at each window position used to analyze the data and determine a set of filter coefficients. This set of filter coefficients is associated with a frame of data. Often the analysis windows associated with a frame of data overlap with adjacent frames of data. This is shown schematically in Fig. 5.1. In some cases such as modelling the correlation or spectrum of the signal, only the analysis step is carried out – there is no filtering using the resulting filter coefficients. In other cases such as in prediction, the filter coefficients derived in the analysis step are subsequently applied to filter the signal.

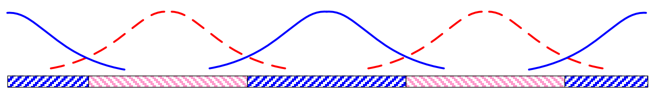


Fig. 5.1 Illustration of frame analysis. Each window is applied to the data to determine a set of filter coefficients which is applied to the data frame in the middle of the analysis interval. The window overlaps frames on either side of the frame to which the filter coefficients will be applied.

For now, we will consider the determination of the filter coefficients for a fixed window position, i.e., we will choose one of the frames as shown in the figure and look at the analysis setup for that frame.

5.1 Analysis Setup

The analysis step involves calculating the correlation values for the windowed data and then solving the Wiener-Hopf equations to get the optimal filter coefficients. The windowed data, windowed desired signal, and windowed error signals are given by

$$\begin{aligned} x_w[n] &= w_x[n]x[n] \\ d_w[n] &= w_d[n]d[n] \\ e_w[n] &= w_e[n]e[n] \end{aligned} \tag{5.1}$$

The windows are most often finite in length. We will assume that they have real-valued coefficients. To determine the optimal filter, the filter is assumed to operate on the windowed data,

$$y[n] = \sum_{k=0}^{N-1} w_k^* x_w[n - D_k]. \tag{5.2}$$

In vector notation, this can be written as

$$y[n] = \mathbf{w}^H \mathbf{x}_w[n], \tag{5.3}$$

where

$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_{M-1} \end{bmatrix}, \qquad \mathbf{x}_w[n] = \begin{bmatrix} x_w[n - D_0] \\ x_w[n - D_1] \\ \vdots \\ x_w[n - D_{M-1}] \end{bmatrix}. \tag{5.4}$$

A block diagram of the system under consideration is shown in Fig. 5.2.

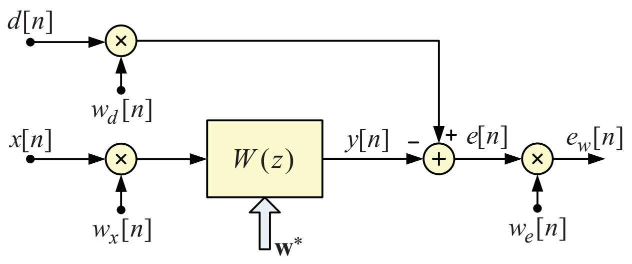


Fig. 5.2 Block diagram for a filtering system with windowed signals

The error signal is

$$\begin{aligned} e[n] &= d_w[n] - y[n] \\ &= d_w[n] - \mathbf{w}^H \mathbf{x}_w[n]. \end{aligned} \tag{5.5}$$

The weighted error signal is

$$e_w[n] = w_e[n]\big(d_w[n] - \mathbf{w}^H \mathbf{x}_w[n]\big). \tag{5.6}$$

We will minimize sum of the weighted error squared over the infinite interval

$$\begin{aligned} \varepsilon &= \overline{|e_w[n]|^2} \\ &= \sum_{n=-\infty}^{\infty} |e_w[n]|^2 \end{aligned} \tag{5.7}$$

In many cases the weighted error signal is actually finite in length and thus there are only a finite number of terms in the sum. This finiteness may occur because of the explicit use of a finite length error window, or because the input and desired signals after windowing are of finite length.

It should be emphasized that the formulation above which includes an equation for filtering of the windowed data and the weighting of the error signal is used only as motivation for solving for the filter coefficients – the actual filtering (if carried out) is done as a separate step. To help in understanding the relative position of the windows with the data, the first time index in the output weighted error signal is set to zero. This establishes a reference point for the windows used in the analysis.

It may be helpful to think of the filter delays as non-negative as they would be for a causal filter. Furthermore, the delays will be ordered $D_0 < D_1 < \cdots < D_{M-1}$.

Covariance Analysis: Consider the case that only an error window is used. No windows are applied to the data or desired signal (or equivalently the these windows are constant for all n). If the error window $w_e[n]$ is of finite length, the sum for the squared error can be truncated. Consider a constant error window of length N,

$$w_e[n] = \begin{cases} 1, & 0 \le n < N, \\ 0, & \text{elsewhere.} \end{cases} \tag{5.8}$$

The limits of the sum of squared errors in Eq. (5.7) will be 0 and $N-1$. We are measuring the error only during that period. Note that for the first few error terms, a causal filter will use data samples before time 0 (past data values).

The term covariance method[1] implies the use of only an error window, which is usually understood to have constant values.

Autocorrelation Analysis: If the data window is finite length and non-zero only in the interval $[0, N-1]$, the filter output will be non-zero only in the interval $[D_0, N-1+D_{M-1}]$, i.e., the first output will occur after a delay of D_0 samples and the last output will occur D_{M-1} samples after the last input applied at time $N-1$. If the desired signal is similarly limited[2] and there is no error window (or equivalently, the error window is constant for all n), this leads to the so-called autocorrelation method. In the autocorrelation method, the data window is often tapered. The filter coefficients determined from the windowed data will generally be applied to the unwindowed data, i.e. the data window is used only in the analysis phase and not in the filtering phase.

Pre-Windowed Analysis: For this case, we can have a one-sided data window that is zero for $n < 0$ and a one-sided error window that is zero for $n \geq N$. This gives the so-called pre-windowed formulation. This combination of data window (and the same window for the desired signal) and error window again results in a finite length output due to the data.

Post-Windowed Analysis: For this case, we can have a one-sided data window that is zero for $n \geq N$ and a one-sided error window that is zero for $n < 0$. This gives the so-called post-windowed formulation. This combination of data window (and the same window for the desired signal) and error window again results in a finite length output due to the data. Note that for the first few error terms, a causal filter will use data samples before time 0 (past data values).

Other Analysis Forms: A weighted covariance formulation arises from the covariance analysis if the error window is allowed to be tapered. Other hybrid forms of analysis occur for simultaneous application of tapered data windows and error windows. An infinite error window can also be used. In that case, special forms such as an exponential window lead to the a recursive least-squares formulation (see [13]).

To simplify the discussion that follows, we will assume that the windowed desired signal, the windowed data signal, and the windowed error signal are of the same length. The windows may have to be zero-padded to achieve this. The weighted error of Eq. (5.6) can be written in vector

[1]The "covariance" name is not based on the standard usage of the covariance function as the correlation function with means removed [28].

[2]If the windowed desired signal is not time limited, it will contribute to the error outside the limits $[0, N-1+D_{M-1}]$. The resulting error signal outside of these limits will not depend on the filter coefficients. Thus for the purpose of determining the optimal filter coefficients, this contribution can be ignored.

form with the elements stacked in a column,

$$\mathbf{e}_w = \mathbf{W}_e\big(\mathbf{W}_d\mathbf{d} - (\mathbf{w}^H\mathbf{X}_w)^T\big). \tag{5.9}$$

The size of the weighted error vector depends on the the analysis method and will for now be left as K. The rectangular data matrix which appears in the formulation has windowed data vectors as rows,[3]

$$\mathbf{X}_w = \Big[\mathbf{x}_w[0] \quad \mathbf{x}_w[1] \quad \cdots \quad \mathbf{x}_w[K-1]\Big], \tag{5.10}$$

The window weights for the desired signal and error signal have been placed in diagonal matrices,

$$\begin{aligned} \mathbf{W}_d &= \operatorname{diag}(\{w_d[n]\}), \qquad [K \times K], \\ \mathbf{W}_e &= \operatorname{diag}(\{w_e[n]\}), \qquad [K \times K]. \end{aligned} \tag{5.11}$$

The sum of the squared errors is

$$\begin{aligned} \varepsilon &= \mathbf{e}_w^H\mathbf{e}_w \\ &= \mathbf{d}^H\mathbf{W}_d\mathbf{W}_e\mathbf{W}_e\mathbf{W}_d\mathbf{d} - 2\mathrm{Re}[\mathbf{w}^H\mathbf{X}_w\mathbf{W}_e\mathbf{W}_e\mathbf{W}_d\mathbf{d}^*] + \mathbf{w}^H\mathbf{X}_w\mathbf{W}_e\mathbf{W}_e\mathbf{X}_w^H\mathbf{w} \\ &= \mathbf{d}_{we}^H\mathbf{d}_{we} - 2\mathrm{Re}[\mathbf{w}^H\mathbf{r}_{\mathbf{x}d}] + \mathbf{w}^H\mathbf{R}_{\mathbf{xx}}\mathbf{w}. \end{aligned} \tag{5.12}$$

We have used the fact that the result is real to write the result in the same form as found in Chapter 2 with the averaging operation now being a sum. The elements of the matrix $\mathbf{R}_{\mathbf{xx}}$ are

$$\mathbf{R}_{\mathbf{xx}}(i,j) = \sum_{n=-\infty}^{\infty} w_e^2[n]\, x_w[n-D_i]x_w^*[n-D_j], \qquad 0 \le i,j < M. \tag{5.13}$$

The elements of the vector $\mathbf{r}_{\mathbf{x}d}$ are

$$\mathbf{r}_{\mathbf{x}d}(k) = \sum_{n=-\infty}^{\infty} w_e^2[n]\, x_w[n-D_k]d_w^*[n], \qquad 0 \le k < M. \tag{5.14}$$

5.2 Covariance Formulation

In the basic covariance method, there is no overlap with adjacent frames. Unlike the autocorrelation formulation, the least squared error calculated during the analysis phase will be the actual squared error when the data in the frame is filtered. For instance, if D_0 is the smallest delay and D_{M-1} is the largest delay, then samples back to $n = -D_{M-1}$ and forward to $n = N-1-D_0$ are needed. For the desired signal, only samples in the interval $[0, N-1]$ are needed. The correlation

[3]The data matrix has conjugated data values and is K by M. This convention will simplify subsequent formulations.

matrix can be expressed as

$$\mathbf{R}_{\mathbf{xx}} = \mathbf{X}\mathbf{X}^H. \tag{5.15}$$

In this expression we use the unwindowed data. The data matrix is then $[M \times N]$,

$$\mathbf{X} = \begin{bmatrix} x[-D_0] & x[1-D_0] & \cdots & x[N-1-D_0] \\ x[-D_1] & x[1-D_1] & \cdots & x[N-1-D_1] \\ \vdots & \vdots & \ddots & \vdots \\ x[-D_{M-1}] & x[1-D_{M-1}] & \cdots & x[N-1-D_{M-1}] \end{bmatrix} \tag{5.16}$$

Element i, j of the correlation matrix $\mathbf{R}_{\mathbf{xx}}$ is

$$\mathbf{R}_{\mathbf{xx}}(i,j) = \sum_{n=0}^{N-1} x[n-D_i]x^*[n-D_j], \qquad 0 \le i,j < M. \tag{5.17}$$

Since the correlation matrix is Hermitian symmetric, only the elements with $0 \le i < M$ and $j \le i$ need be calculated. The rest of the matrix can be filled in from symmetry. The elements of $\mathbf{r}_{\mathbf{x}d}$ are

$$\mathbf{r}_{\mathbf{x}d}(k) = \sum_{n=0}^{N-1} x[n-D_k]d^*[n], \qquad 0 \le k < M. \tag{5.18}$$

Each correlation term is the sum of N products.

5.2.1 Equally Spaced Delays

Let the delays be unit-spaced and ordered, viz., $D_k = D_0 + k$. The data matrix is now Toeplitz, though as we will see, the correlation matrix is not.

$$\mathbf{X} = \begin{bmatrix} x[-D_0] & x[1-D_0] & \cdots & x[N-1-D_0] \\ x[-D_0-1] & x[1-D_0-1] & \cdots & x[N-1-D_0-1] \\ \vdots & \vdots & \ddots & \vdots \\ x[-D_0-(M-1)] & x[1-D_0-(M-1)] & \cdots & x[N-1-D_0-(M-1)] \end{bmatrix} \tag{5.19}$$

Element i, j of the autocorrelation matrix is

$$\mathbf{R}_{\mathbf{xx}}(i,j) = \sum_{n=0}^{N-1} x[n-D_0-i]x^*[n-D_0-j], \qquad 0 \le i,j < M. \tag{5.20}$$

For the covariance formulation, the correlation matrix $\mathbf{R}_{\mathbf{xx}}$ is not Toeplitz in general, even if the delays are equally spaced. This means that neither the Durbin recursion nor Levinson algorithm can be used to solve the Wiener-Hopf equations – one needs to use a more general procedure such as the Cholesky decomposition in Appendix H which applies to positive definite matrices.

The elements of the cross-correlation vector are

$$\mathbf{r}_{\mathbf{x}d}(k) = \sum_{n=0}^{N-1} x[n - D_o - k]d^*[n], \qquad 0 \le k < M. \tag{5.21}$$

With the assumption of equally spaced delays, a recursive computation of the elements or $\mathbf{R}_{\mathbf{xx}}$ down the minor diagonals can be used.

$$\begin{aligned}\mathbf{R}_{\mathbf{xx}}(i+1, j+1) &= \sum_{n=0}^{N-1} x[n - D_0 - i - 1]x^*[n - D_0 - j - 1] \\ &= \mathbf{R}_{\mathbf{xx}}(i, j) + x[-D_0 - i - 1]x^*[-D_0 - j - 1] \\ &\qquad - x[N - 1 - D_0 - i]x^*[N - 1 - D_0 - j]\end{aligned} \tag{5.22}$$

For a large N, finite precision effects can cause computational errors to accumulate during the recursion, even to the extent that the resulting matrix $\mathbf{R}_{\mathbf{xx}}$ is no longer positive definite.

5.2.2 Tapered Error Window

Tapered error windows have been suggested as a means to smooth the evolution of the filter coefficients as the window advances (see [36]). The tapered error window modifies the data matrix $\mathbf{X}$. The modified data matrix is now

$$\mathbf{X}\mathbf{W}_e = \Big[w_e[0]\mathbf{x}[0] \quad w_e[1]\mathbf{x}[1] \quad \cdots \quad w_e[N-1]\mathbf{x}[N-1] \Big]. \tag{5.23}$$

The elements of $\mathbf{R}_{\mathbf{xx}}$ are now, c.f. Eq. (5.17),

$$\mathbf{R}_{\mathbf{xx}}(i, j) = \sum_{n=0}^{N-1} w_e^2[n]x[n - D_i]x^*[n - D_j], \qquad 0 \le i, j < M. \tag{5.24}$$

For a general error window, this expression is no longer amenable to the recursive computation described above. With an error window, the elements of $\mathbf{r}_{\mathbf{x}d}$ are, c.f. Eq. (5.18),

$$\mathbf{r}_{\mathbf{x}d}(k) = \sum_{n=0}^{N-1} w_e^2[n]x[n - D_k]d^*[m], \qquad 0 \le k < M. \tag{5.25}$$

5.3 Autocorrelation Formulation

For the autocorrelation formulation, only a data window is applied – the error window is constant. We need to use the windowed data elements $x_w[n]$. Assume that the data window is non-zero in the interval $[0, N-1]$. The filter output $y[n]$ is then non-zero in the interval $[D_0, N-1+D_{M-1}]$. However, we will use a lower bound of zero giving the signal output range $[0, N-1+D_{M-1}]$. This is done since the error will be non-zero at time zero if the windowed desired signal is non-zero at that time point. The data matrix is now of the form

$$\mathbf{X}_w = \begin{bmatrix} x_w[-D_0] & x_w[1-D_0] & \cdots & x_w[N-1+D_{M-1}-D_0] \\ x_w[-D_1] & x_w[1-D_1] & \cdots & x_w[N-1+D_{M-1}-D_1] \\ \vdots & \vdots & \ddots & \vdots \\ x_w[-D_{M-1}] & x_w[1-D_{M-1}] & \cdots & x_w[N-1] \end{bmatrix} \tag{5.26}$$

Note that the elements in the lower left triangular corner and in the upper right triangular corner are zero. Specifically, any elements $x_w[k]$ with k outside of the interval $[0, N-1]$ are zero. If we eliminate the data samples outside the $[0, N-1]$ interval, the elements of $\mathbf{R_{xx}}$ are given by

$$\mathbf{R_{xx}}(i,j) = \sum_{n=\max(D_i,D_j)}^{N-1+\min(D_i,D_j)} x_w[n-D_i]x_w^*[n-D_j], \qquad 0 \le i,j < M, \tag{5.27}$$

and the elements of $\mathbf{r}_{\mathbf{x}d}$ are given by

$$\mathbf{r}_{\mathbf{x}d}(k) = \sum_{n=D_k}^{N-1} x_w[n-D_k]d_w^*[n], \qquad 0 \le k < M. \tag{5.28}$$

The number of terms in these sums varies, but is at most N for any given element in the matrix or vector.

5.3.1 Equally Spaced Delays

Let the delays be unit-spaced, viz., $D_k = D_0 + k$. The windowed data matrix ($[M \times N-1+M-1+D_0]$) becomes Toeplitz,

$$\mathbf{X}_w = \begin{bmatrix} x_w[-D_0] & x_w[1-D_0] & \cdots & x_w[N-1-D_0+(M-1)] \\ x_w[-D_0-1] & x_w[1-D_0-1] & \cdots & x_w[N-1-D_0+(M-2)] \\ \vdots & \vdots & \ddots & \vdots \\ x_w[-D_0-(M-1)] & x_w[1-D_0-(M-1)] & \cdots & x_w[N-1-D_0] \end{bmatrix}. \tag{5.29}$$

Again the elements in the lower left triangular corner and in the upper right triangular corner are zero due to the finite length of the data window. Each row of this matrix is of the form

$$\mathbf{X}_w(i,:) = \begin{bmatrix} \underbrace{0 \quad \cdots \quad 0}_{D_0+i \text{ zeros}} & x_w[0] & x_w[1] & \cdots & x_w[N-1] & \underbrace{0 \quad \cdots \quad 0}_{M-1-i \text{ zeros}} \end{bmatrix}, \quad i = 0, \ldots, M-1. \tag{5.30}$$

This means each row contains the entire windowed data sequence with zeros added before the sequence and/or added after the sequence. This also means that the correlation matrix $\mathbf{R}_{\mathbf{xx}}$ will be Toeplitz. If we eliminate the data samples outside the $[0, N-1]$ interval, the elements of $\mathbf{R}_{\mathbf{xx}}$ are given by

$$\mathbf{R}_{\mathbf{xx}}(i,j) = \sum_{n=D_0+\max(i,j)}^{N-1+D_0+\min(i,j)} x_w[n-D_0-i]x_w^*[n-D_0-j], \qquad 0 \le i,j < M, \tag{5.31}$$

and the elements of $\mathbf{r}_{\mathbf{x}d}$ are given by

$$\mathbf{r}_{\mathbf{x}d}(k) = \sum_{n=D_0+k}^{N-1} x_w[n-D_k]d_w^*[n], \qquad 0 \le k < M. \tag{5.32}$$

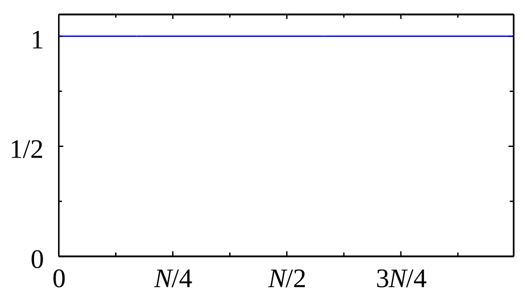
5.3.2 Data Windows for the Autocorrelation Method

The choice of an appropriate data window is important to getting reliable estimates of the correlation values that appear in the correlation matrix $\mathbf{R}_{\mathbf{xx}}$. There is much literature on the subject. Here we will compare some standard choices for finite length windows. Figure 5.3 shows the window shapes and the corresponding frequency responses. A more detailed description of the windows and their application in speech processing appears in [19].

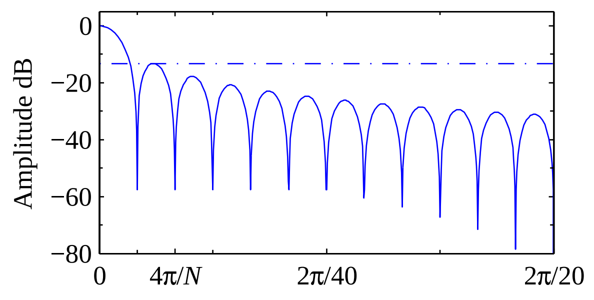
For these responses we notice the following.

- There is a tradeoff between the smoothness of the window and the main lobe width in the frequency domain. The rectangular window has abrupt jumps, but has the narrowest main lobe.
- For the Dolph-Chebyshev window, the sidelobe attenuation was set to match that of the Hamming window (42.7 dB). The constant attenuation sidelobe behaviour of the Dolph-Chebyshev window comes at the cost of an anomalous increase in the end points of the window. Circles are shown at the end points of the Dolph-Chebyshev window in the figure.
- The van Hann window is the smoothest of the windows shown.

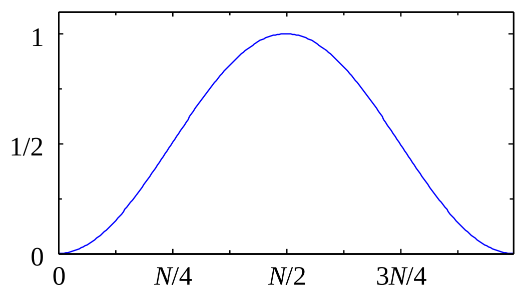
These windows will be compared for their ability to resolve sinusoidal components and for the smoothness of correlation estimates in a later chapter.



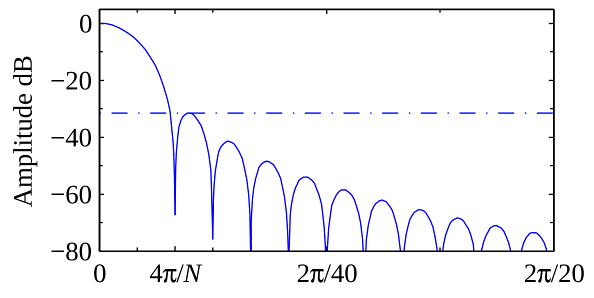
(a) Rectangular window



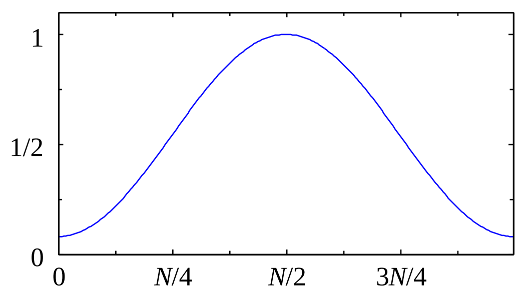
(b) Rectangular window frequency response



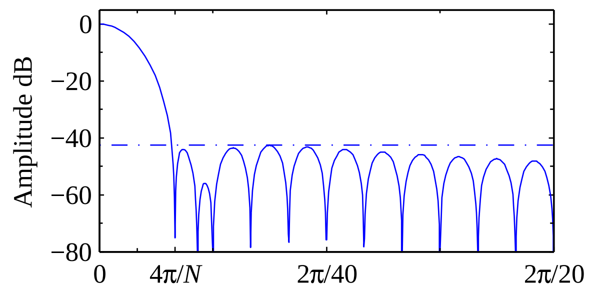
(c) von Hann window



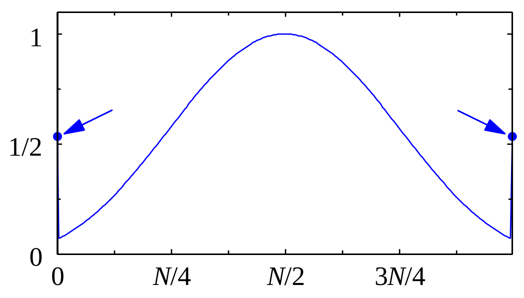
(d) von Hann window frequency response



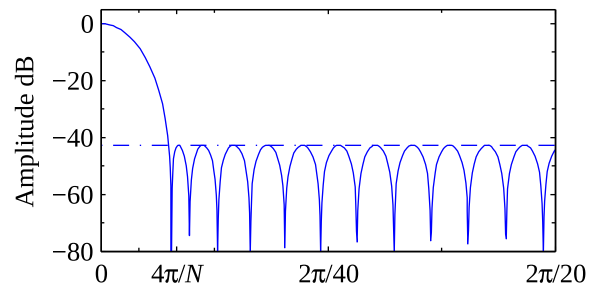
(e) Hamming window



(f) Hamming window frequency response



(g) Dolph-Chebyshev window



(h) Dolph-Chebyshev window frequency response

Fig. 5.3 Several windows ($N = 240$) and their frequency responses. The dashed lines on the frequency responses show the side lobe attenuation. For the Dolph-Chebyshev window, the sidelobe attenuation was chosen to be the same as for the Hamming window.

5.4 Pre-Windowed Analysis

For the pre-windowed formulation, a right-sided data window starting at $n = 0$ and a left-sided error window ending at $n = N - 1$ are applied. The data matrix ($[M \times N]$) is of the form

$$\mathbf{X}_w = \begin{bmatrix} x_w[-D_0] & x_w[1-D_0] & \cdots & x_w[N-1-D_0] \\ x_w[-D_1] & x_w[1-D_1] & \cdots & x_w[N-1-D_1] \\ \vdots & \vdots & \ddots & \vdots \\ x_w[-D_{M-1}] & x_w[1-D_{M-1}] & \cdots & x_w[N-1-D_{M-1}] \end{bmatrix} \tag{5.33}$$

Note that the elements in the lower left triangular corner are zero due to the data window. Specifically, any elements $x_w[n]$ with $n < 0$ are zero. If we eliminate those samples, the elements of $\mathbf{R_{xx}}$ are given by

$$\mathbf{R_{xx}}(i,j) = \sum_{n=\max(D_i,D_j)}^{N-1} x_w[n-D_i]x_w^*[n-D_j], \qquad 0 \le i,j < M, \tag{5.34}$$

and the elements of $\mathbf{r}_{\mathbf{x}d}$ are given by

$$\mathbf{r}_{\mathbf{x}d}(k) = \sum_{n=D_k}^{N-1} x_w[n-D_k]d_w^*[n], \qquad 0 \le k < M. \tag{5.35}$$

The number of terms in these sums varies, but is at most N for any given element in the matrix or vector.

5.5 Post-Windowed Analysis

For the post-windowed formulation, a left-sided data window ending at $n = N - 1$ and a right-sided error window starting at $n = 0$ are applied. The data matrix ($[M \times N + D_{M-1} - D_0]$) is of the form

$$\mathbf{X}_w = \begin{bmatrix} x_w[-D_0] & x_w[1-D_0] & \cdots & x_w[N-1+D_{M-1}-D_0] \\ x_w[-D_1] & x_w[1-D_1] & \cdots & x_w[N-1+D_{M-1}-D_1] \\ \vdots & \vdots & \ddots & \vdots \\ x_w[-D_{M-1}] & x_w[1-D_{M-1}] & \cdots & x_w[N-1+D_{M-1}-D_{M-1}] \end{bmatrix} \tag{5.36}$$

Note that the elements in the upper right triangular corner are zero. Specifically, any elements $x_w[n]$ with $n > N$ are zero. If we eliminate those samples, the elements of $\mathbf{R_{xx}}$ are given by

$$\mathbf{R_{xx}}(i,j) = \sum_{n=0}^{N-1+\min(D_i,D_j)} x_w[n-D_i]x_w^*[n-D_j], \qquad 0 \le i,j < M, \tag{5.37}$$

and the elements of $\mathbf{r}_{\mathbf{x}d}$ are given by

$$\mathbf{r}_{\mathbf{x}d}(k) = \sum_{n=0}^{N-1+D_k} x_w[n - D_k] d_w^*[n], \qquad 0 \le k < M. \tag{5.38}$$

The number of terms in these sums varies, but is at most N for any given element in the matrix or vector.

5.6 Data Matrix Formulation of the Least Squares Problem

The equations to be solved for the least squares problem are

$$\mathbf{R}_{\mathbf{xx}} \mathbf{w}_{\text{opt}} = \mathbf{r}_{\mathbf{x}d}, \tag{5.39}$$

where

$$\begin{aligned} \mathbf{R}_{\mathbf{xx}} &= \mathbf{X}_{we} \mathbf{X}_{we}^H, \\ \mathbf{r}_{\mathbf{x}d} &= \mathbf{X}_{we} \mathbf{d}_{we}^* \end{aligned} \tag{5.40}$$

Here we have defined a doubly weighted data matrix

$$\mathbf{X}_{we} = \mathbf{X}_w \mathbf{W}_e, \tag{5.41}$$

and a doubly weighted desired signal

$$\mathbf{d}_{we} = \mathbf{W}_e \mathbf{W}_d \mathbf{d}. \tag{5.42}$$

The equations to be solved can now be written entirely in terms of the doubly weighted data matrix

$$\mathbf{X}_{we} \mathbf{X}_{we}^H \mathbf{w}_{\text{opt}} = \mathbf{X}_{we} \mathbf{d}_{we}^*. \tag{5.43}$$

The solution can be written as

$$\mathbf{w}_{\text{opt}} = (\mathbf{X}_{we} \mathbf{X}_{we}^H)^{-1} \mathbf{X}_{we} \mathbf{d}_{we}^*. \tag{5.44}$$

This can be expressed using the pseudo-inverse of $\mathbf{X}_h = \mathbf{X}_{we}^H$ which we denote as $\mathbf{X}_h^+$,

$$\mathbf{w}_{\text{opt}}^* = \mathbf{X}_h^+ \mathbf{d}_{we}^*, \tag{5.45}$$

where the pseudo-inverse is given by

$$\mathbf{X}_h^+ = (\mathbf{X}_h^H \mathbf{X}_h)^{-1} \mathbf{X}_h^H. \tag{5.46}$$

Figure 5.2 has been redrawn below to show the doubly weighted desired signal and a weighted filter output. The weighted filter output when the filter uses the optimal coefficients is

$$\begin{aligned} \mathbf{y}^*_{w\,\mathrm{opt}} &= \mathbf{X}^H_{we}\mathbf{w}_{\mathrm{opt}} \\ &= \mathbf{X}^H_{we}(\mathbf{X}_{we}\mathbf{X}^H_{we})^{-1}\mathbf{X}_{we}\mathbf{d}^*_{we}. \end{aligned} \tag{5.47}$$

This can can be interpreted as generating an estimate of the desired signal from the data matrix.

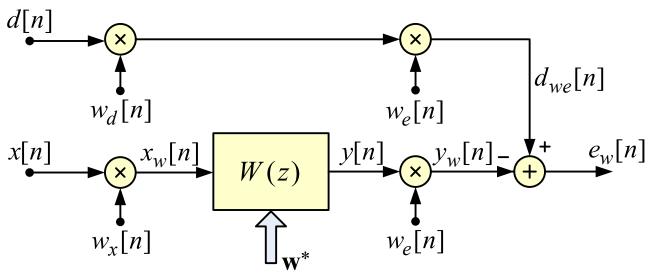


Fig. 5.4 Filtering system showing a doubly weighted desired signal

5.6.1 Projection onto the Data Matrix

An alternate view of the operation shown in Eq. (5.47) is that the filter output is a projection of the desired signal onto the rows of the data matrix,

$$\mathbf{y}_{w\,\mathrm{opt}} = \mathbf{P}\mathbf{d}_{we}, \tag{5.48}$$

where the projection matrix $\mathbf{P}$ is

$$\begin{aligned} \mathbf{P} &= \big(\mathbf{X}^H_{we}(\mathbf{X}_{we}\mathbf{X}^H_{we})^{-1}\mathbf{X}_{we}\big)^* \\ &= \big(\mathbf{X}_h\mathbf{X}^+_h\big)^*, \end{aligned} \tag{5.49}$$

5.6.2 Rank Deficient Cases

We will assume that the data matrix is M by K, where $K \geq M$. If K is not greater than M, padding the data matrix with zero-valued columns will achieve this. The question of interest is then what happens when the data matrix is rank deficient. This answer is given by the singular-value decomposition (SVD).

The SVD for a rectangular matrix $\mathbf{A}$ is [13]

$$\mathbf{U}^H_{[M\times M]}\mathbf{A}_{[M\times K]}\mathbf{V}_{[K\times K]} = \begin{bmatrix} \boldsymbol{\Sigma}_{[P\times P]} & \mathbf{0}_{[P\times(K-P)]} \\ \mathbf{0}_{[(M-P)\times P]} & \mathbf{0}_{[(M-P)\times(K-P)]} \end{bmatrix}. \tag{5.50}$$

For clarity, the matrices are shown with their dimensions as subscripts. In this equation both $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices[4] The matrix $\mathbf{\Sigma}$ is a diagonal matrix with elements $\sigma_0 \cdots \sigma_{P-1}$ along the diagonal, where P is the rank of the matrix A. For a full rank matrix $\mathbf{A}$, the diagonal elements are the square roots of the (non-negative) eigenvalues of the symmetric matrix $\mathbf{A}\mathbf{A}^H$. If $\mathbf{A}$ is rank deficient, some of the eigenvalues of $\mathbf{A}\mathbf{A}^H$ are zero and can be associated with the zero matrices in Eq. (5.50).

The solution the optimal filter weights can be written in terms of a pseudo-inverse where the pseudo-inverse of a rectangular matrix $\mathbf{A}$ is

$$\mathbf{A}^{+} = \mathbf{V} \begin{bmatrix} \mathbf{\Sigma} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \mathbf{U}^H. \tag{5.51}$$

In terms of our original least squares problem

$$\mathbf{w}_{\text{opt}} = \mathbf{X}_h^{+} \mathbf{d}_{we}^{*}. \tag{5.52}$$

In the rank deficient case there are an infinity of solutions which give the smallest sum of squared errors. The pseudo-inverse gives the solution with the smallest sum of squared filter weights.

5.7 Estimation of the Correlation

The methods described above involve applying windows to the data, desired signal signal, and/or the error sequence. With those windows, we can determine the elements of the data matrix. These data are used to compute the correlation matrix and the correlation vector needed for the Wiener-Hopf equations. In this section, we describe an alternate approach. In this approach, we estimate the correlation sequence and then use those values to directly fill in the correlation matrix and correlation vector.

Consider a deterministic cross-correlation calculation of the form

$$r_{uv}[k] = \sum_{m=-\infty}^{\infty} u[m] v^*[m-k]. \tag{5.53}$$

Such a calculation assumes that the sum stays finite. The correlation computed satisfies $r_{uv}[k] = r_{vu}^*[-k]$.

[4]Recall that for a unitary matrix, $\mathbf{U}\mathbf{U}^H = \mathbf{I}$.

5.7.1 Windowed Data

We can assure a finite sum by either applying finite length windows to the signals or using a tapered infinite length window. With windows applied at a fixed position n_o, we can write the correlation sum as

$$r_{uv}[k; n_o] = \sum_{m=-\infty}^{\infty} w_u[m - n_o] u[m]\, w_v[m - n_o - k] v^*[m - k]. \tag{5.54}$$

This again satisfies $r_{uv}[k; n_o] = r_{vu}^*[-k; n_o]$.

Equal Length Windows

If both $w_u[n]$ and $w_v[n]$ are non-zero outside of the interval $[0, N-1]$, then for non-negative values of k,

$$r_{uv}[k; n_o] = \sum_{m=k}^{N-1} w_u[m] u[m + n_o]\, w_v[m - k] v^*[m + n_o - k], \qquad k \geq 0. \tag{5.55}$$

If $u[n]$ and $v[n]$ are jointly wide-sense stationary, the expected value of this correlation sum is

$$\begin{aligned} E\bigl[r_{uv}[k; n_o]\bigr] &= \sum_{m=k}^{N-1} w_u[m] w_v[m - k] E\bigl[u[m + n_o] v^*[m + n_o - k]\bigr] \\ &= \sum_{m=n_o+k}^{n_o+N-1} w_u[m] w_v[m - k]\, r_{uv}[k]. \end{aligned} \tag{5.56}$$

If furthermore the windows are equal to one in the interval $[0, N-1]$,

$$E\bigl[r_{uv}[k; n_o]\bigr] = (N - k) r_{uv}[k], \qquad k \geq 0. \tag{5.57}$$

This shows that for equal length rectangular windows of length N, the average estimated correlation values are modified by a triangular correlation window – the windowed cross-correlation function is biased.

If we go back and look at the autocorrelation method of Section 5.3, we see that the matrix-vector equations are filled in with estimated correlations which can be calculated using Eq. (5.55), with the data window and desired signal window positioned at $n_o = 0$. For the autocorrelation matrix, $u[n] = v[n] = x[n]$ and $k = D_j - D_i$ for $D_j \geq D_i$. For the cross-correlation vector, $u[n] = x^*[n]$, $v[n] = d^*[n]$, and $k = D_k$.

Unequal Length Windows

Now consider that the windows in the basic cross-correlation computation of Eq. (5.54) are of lengths N_u and N_v. Let us calculate the correlation $r_{uv}[k;n_o]$ for lags $0 \le k \le M$ and let N_u be at least $N_v + M$,

$$\begin{aligned} r_{uv}[k;n_o] &= \sum_{m=-\infty}^{\infty} w_u[m+k]u[m+n_o+k]\, w_v[m]v^*[m+n_o] \\ &= \sum_{m=0}^{N_v-1} w_u[m+k]u[m+n_o+k]\, w_v[m]v^*[m+n_o]. \end{aligned} \tag{5.58}$$

Note that the number of terms in the sum is constant.

For constant windows (unity height),

$$r_{uv}[k;n_o] = \sum_{m=0}^{N_v-1} u[m+n_o+k]v^*[m+n_o]. \tag{5.59}$$

Then for jointly wide-sense stationary signals, the expected value of the correlation estimate is

$$E\big[r_{uv}[k;n_o]\big] = N_v r_{uv}[k], \qquad 0 \le k \le M. \tag{5.60}$$

This estimate when scaled by N_v is unbiased.

If we go back and look at the covariance method of Section 5.2, we see that the correlation matrix is filled in with estimated correlations which can be calculated using Eq. (5.59) with the data windows positioned at $n_o = -D_j$ and window lengths $N_v = N$ and $N_u = N + M$. For the autocorrelation matrix, $u[n] = v[n] = x[n]$ and $k = D_j - D_i$ for $D_j > D_i$. Note that the windows are repositioned for each delay D_j, i.e. for each column of the correlation matrix. For the cross-correlation vector, the windows are positioned at $n_o = D_k$ for element k of the vector, with $u[n] = x^*[n]$, $v[n] = d^*[n]$, and $k = D_k$.

5.7.2 Correlation Windows

An alternative to data windows is to apply a lag window to the correlation lag values. Calculate the correlation value at lag k at time n_o as follows,

$$r_{uv}[k;n] = \sum_{m=-\infty}^{n} w_k[m]\, u[n_o-m]v^*[n_o-m-k]. \tag{5.61}$$

The window $w_r[n]$ must render the correlation estimate to be positive definite.

Barnwell [2] suggests a formulation which decomposes the lag window as

$$r_{uv}[k;n_o] = \sum_{m=-\infty}^{\infty} w[k;m]\, u[m+n_o+k]v^*[m+n_o], \tag{5.62}$$

where the window is decomposed such that

$$w[k;m] = w_u[m+k]w_v[m]. \tag{5.63}$$

This in effect applies the time window $w_u[m]$ to $u[n]$ and the time window $w_v[m]$ to $v[n]$. Barnwell's suggestion is to precompute the window functions for each lag k. These combined window functions are applied to the products $u[n+k]v^*[n]$ for each k. In [2], the individual windows are recursive, so the product is also recursive.

Consider the exponential window,

$$w_r[n] = \begin{cases} \lambda^{-n}, & n \le 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5.64}$$

We also require that $\lambda < 1$. The correlation can now be updated recursively,

$$r_{xx}[k;n] = \lambda r_{xx}[k;n-1] + x[n]x^*[n-k]. \tag{5.65}$$

One problem with the exponential window is that it has a rather "fat" tail. This means that it does not "forget" old values fast enough. A second order recursive window can be used. Strohbach [38] suggests a window which is the the cascade of two exponential windows, each with the same forgetting factor,

$$\begin{aligned} r_{xx}[k;n] &= \beta r_{xx}[k;n-1] + t[k;n], \\ t[k;n] &= \beta t[k;n-1] + x[n]x^*[n-k]. \end{aligned} \tag{5.66}$$

Problems

P5.1 Projection Matrix

Consider the projection matrix $\mathbf{P}$ introduced in Section 5.6.1. This matrix has some interesting properties.

(a) Show that $\mathbf{P}$ is Hermitian symmetric.

(b) Show that $\mathbf{P}$ is idempotent, $\mathbf{PP} = \mathbf{P}$. This means that applying $\mathbf{P}$ several times gives the same result as applying it once.

(c) Show that the projection matrix is non-negative definite.

(d) The complementary projection to $\mathbf{P}$ is $\mathbf{P}_c = \mathbf{I} - \mathbf{P}$. Show that $\mathbf{P}_c\mathbf{P} = \mathbf{P}\mathbf{P}_c = \mathbf{0}$.

Show that $\mathbf{P}_c\mathbf{d} = \mathbf{e}_{\text{min}}$, where $\mathbf{e}_{\text{min}}$ is the vector of errors.

(e) Show that the projection is an orthogonal projection of the data onto the desired signal. To do this show that the data is orthogonal to the error,

$$\mathbf{X}^*\mathbf{e}_{\text{min}} = \mathbf{0}. \tag{P5.1-1}$$

P5.2 Double Data Window

In Section 5.7.1 we discussed the estimation of correlation values for the covariance method. In that section, the results came from data windows which were constant but of unequal length. Is there a generalization of the windows (non-flat windows) which can be used to estimate correlation values which correspond to the case of tapered error windows?

P5.3 Covariance-Like Calculation

We see from Section 5.7.1, that estimated correlation values can be used to fill in the matrix and vectors for the Wiener-Hopf equations. However, these correlation values are calculated for different window offsets depending on the column of the correlation matrix or the element of the cross-correlation vector. Here we consider estimation of the correlation values, but at taken at only one window offset, and then used to populate the matrix and the vector.

(a) Is the sequence of correlation estimates for $k = 0, \ldots, M$ positive definite? If not, find a simple sequence of data which gives rise to non-positive definite "correlation" values.

(b) What error condition does the set of equations minimize?

P5.4 Exponential Data Window

Consider the exponential *data* window

$$w_x[l] = \beta^l u[-l]. \tag{P5.4-1}$$

Find the equivalent lag-dependent window which can be applied to the correlations (see Eq. (5.62)).

P5.5 Barnwell Window

Consider the Barnwell data window with the z-transform,

$$W_x(z) = \frac{1}{(1-\beta z)^2}. \tag{P5.5-1}$$

(a) Find the time domain response of the window. Note that window is non-zero to the left of the time origin.

(b) Give a recursive implementation of the data windowing operation.

(c) The window as defined above can be normalized in a number of ways. Find a normalization such that the energy estimate $r_{xx}[0]$ using this data window is unbiased.

(d) The effective window length of a window can be defined as the length of an equivalent rectangular window such that the windowed sequences for both types of windows have the same energy when excited by a white noise input.

P5.6 Correlation Windows

An earlier problem P4.5 explored some aspects of lag windows applied to a correlation sequence. There we found that for a window $w[k]$, non-negativity of the DTFT $W(\omega)$ was desirable. Here we consider two lag windows.

(a) A triangular window (also known as the Bartlett window) is given as

$$w[k] = \begin{cases} 1 - \dfrac{|k|}{L}, & \text{for } |k| \le L, \\ 0, & \text{otherwise.} \end{cases} \tag{P5.6-1}$$

Note that this window does both shaping of the correlation and truncation of the correlation sequence. Find a closed form for $W(\omega)$ which shows that this spectrum is both real and non-negative.

(b) A Gaussian window is given by

$$w[k] = \exp\bigl(-\frac{1}{2}(ak)^2\bigr). \tag{P5.6-2}$$

It may helpful to at first consider the same function with k replaced by the continuous variable t. Find the Fourier transform for $w(t)$. Note that $w(t)$ is an even function and so the Fourier transform $W(f)$ can be written using $\cos(2\pi f t)$ instead of $\exp(j2\pi f t)$. You may have to resort to integral tables (or search for "Fourier transform of a Gaussian" on the Internet) to get an expression for the Fourier transform. The Gaussian response is special, since the Fourier transform also has a Gaussian shape! Find an expression for the DTFT of $w[k]$ in terms of $W(f)$. Since $w[k]$ is a sampled version of $w(t)$, you can express the DTFT of $w[k]$ in terms of the aliased spectrum of $w(t)$ (see [23]). Now argue that $W(\omega)$ is real and strictly positive.

Lag windowing is the multiplication of the lag window with the correlation sequence.

(c) Give an expression for power spectral density of the windowed correlation in terms of the original power spectral density $S_{xx}(\omega)$ and the frequency response of the window $W(\omega)$.

The effect of lag windowing can be assessed in terms of a bandwidth expansion. This is measured as follows. Suppose the power spectral density consists of a single Dirac delta. The lag windowed correlation will correspond to a power spectral density where the delta function is spread out. The bandwidth expansion is then the bandwidth of the spread out function. See [20] for a discussion of bandwidth expansion. For the sequel, we will measure the bandwidth as the width of the response at the -3dB points.

(d) What is the 3 dB bandwidth expansion factor for a triangular lag window as a function of L? There will not be a closed form for this value. Instead find its value numerically for asymptotically large L.

(e) What is the 3 dB bandwidth expansion factor for a Gaussian lag window in terms of the parameter a? Find a closed form when a is large.

P5.7 Data and Desired Signal Scaling

In the formulation of the least-squares approach to finding an optimal filter, we have ignored the "mean" part of mean-square error, i.e. we have calculated the sum of squares rather than the mean of the sum of squares. Show that the solution to the Wiener-Hopf equations is insensitive to (a) scaling of the the data and desired signals, and (b) scaling of the correlation values.

Chapter 6

Least Squares Prediction

Least squares prediction is a fully fledged topic unto itself. From the previous chapter, we have kept the filter delay values somewhat arbitrary. Here we specialize the filter delay values for the predictor case.

6.1 One-Step Predictor

For the one-step predictor,

$$D_k = k + 1, \qquad 0 \leq k < M. \tag{6.1}$$

In the next subsections, we will show the setup for solving the Wiener-Hopf equations using a data matrix formulation.

The scenario we are considering is a frame based analysis as shown schematically in Fig. 5.1. Consider a specific frame of data. For ease of reference, we will denote the samples of the data in the frame as $x[0]$ through $x[N-1]$. If the analysis window extends before and/or after the frame of data, the analysis can refer to samples before the beginning of the frame ($x[n]$ with $n < 0$) and/or after the end of the frame ($x[n]$ with $n >= N$).

The different analysis scenarios will use a data window which is applied to both the desired signal and the signal which is input to the the predictor and/or an error window.

6.2 Autocorrelation Formulation

For the autocorrelation formulation, the data matrix ($[M \times N+M]$) containing windowed data values is

$$\mathbf{X}_w = \begin{bmatrix} 0 & x_w[0] & x_w[1] & \cdots & x_w[M-1] & \cdots & x_w[N-1] & 0 & \cdots & 0 \\ 0 & 0 & x_w[0] & \cdots & x_w[M-2] & \cdots & x_w[N-2] & x_w[N-1] & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & x_w[0] & \cdots & x_w[N-M] & x_w[N-M+1] & \cdots & x_w[N-1] \end{bmatrix}. \tag{6.2}$$

Here we have assumed that the data window is zero outside the interval $[0, N-1]$. The elements of the data matrix are shown as $x_w[n]$, since often a tapered window is used in the autocorrelation method. Each column in the data matrix contains the data values used by the predictor to calculate on output value. The first column is used to predict $x_w[0]$, the second to predict $x_w[1]$, and so on. Note that the first column contains only zeros, since the first non-zero data sample is predicted from the previous zero-valued samples. The last column contains only a single non-zero element.

We also assume (as is usual for the prediction case) that the window applied to the desired signal is the same as that applied to the data. Since the desired signal has to match the length of the filter output sequence, the desired signal window is the same as the data window, but padded out with M zeros. The weighted desired signal ($[M+N \times M]$) is

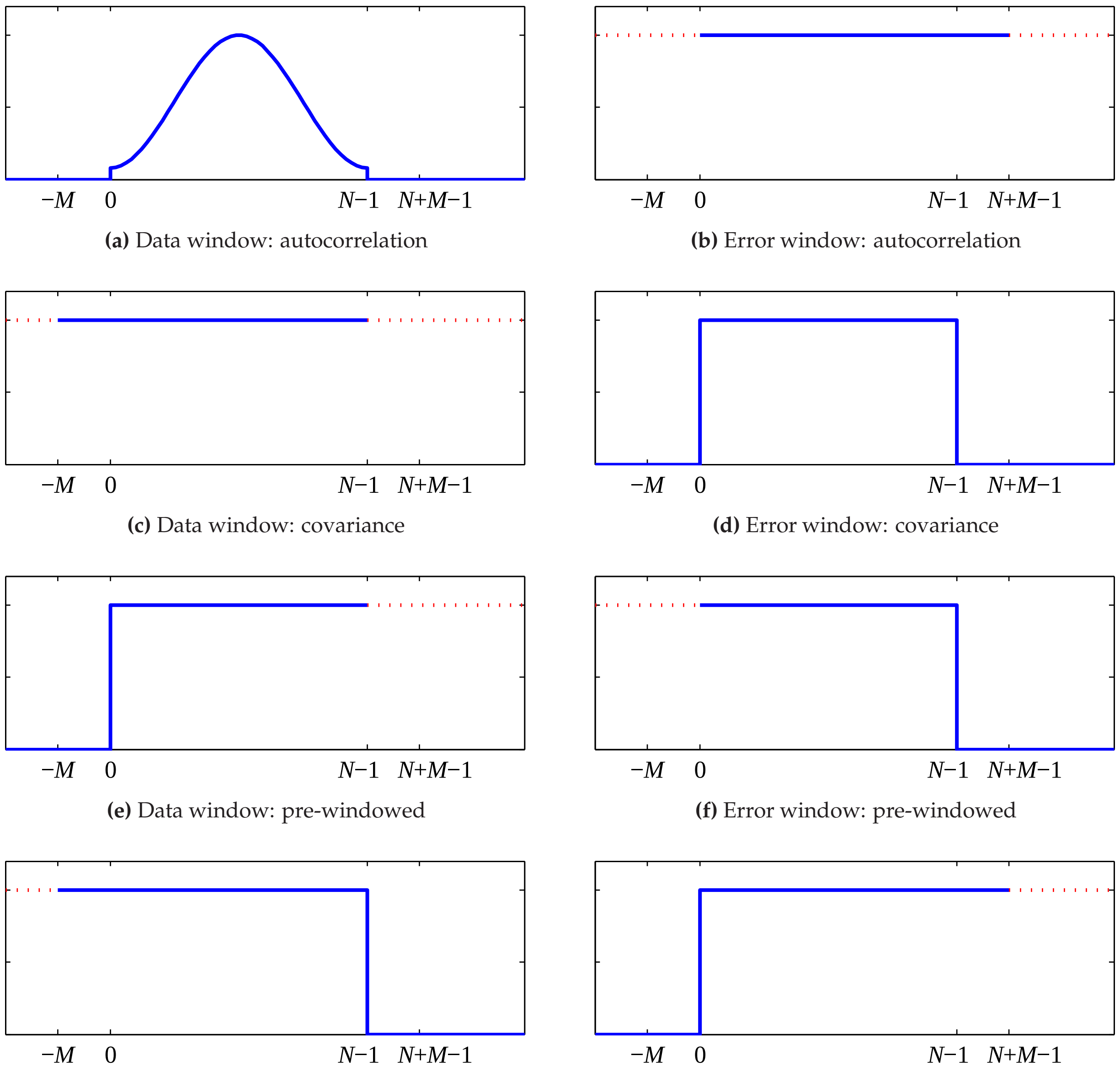
$$\mathbf{d}_w = \begin{bmatrix} x_w[0] & x_w[1] & \cdots & x_w[N-1] & 0 & \cdots & 0 \end{bmatrix}^T. \tag{6.3}$$

The Wiener-Hopf equation can be written as

$$\mathbf{X}_w \mathbf{X}_w^H \mathbf{w}_{\text{opt}} = \mathbf{X}_w \mathbf{W}_d \mathbf{d}_w^*. \tag{6.4}$$

The correlation matrix $\mathbf{X}_w \mathbf{X}_w^H$ is Toeplitz. For the autocorrelation case, the predictor is minimum phase from the structure of the correlation matrix and the cross-correlation vector. This requires that the roots of the prediction error filter lie within the unit circle. In fact for finite blocks of data, the roots can be shown to be inside a circle of radius $\cos\bigl(\pi/(N+M)\bigr)$ [6].

Figure 6.1 shows a schematic representation of the data and error windows for the autocorrelation method. The data window is shown as a Hamming window, but could just as well be a rectangular window. The dotted part of the error window is a region for which the output of the filter is zero and so the value of the error window in that region does not matter.



(a) Data window: autocorrelation

(b) Error window: autocorrelation

(c) Data window: covariance

(d) Error window: covariance

(e) Data window: pre-windowed

(f) Error window: pre-windowed

(g) Data window: post-windowed

(h) Error window: post-windowed

Fig. 6.1 Data and error windows for different analysis procedures. The solid lines show the required window responses. The dotted lines show the “don’t care” values for the window responses.

6.2.1 The Importance of Tapered Windows

For the autocorrelation method, windowing the data has the effect that near the beginning of the block of data, non-zero values are being predicted from zero-valued (due to the data window) data. At the end of the block of data, zero valued samples are being predicted from non-zero valued samples. Of course the predictor coefficients have to be chosen to given the best performance for the whole block of data. One way to reduce the effect of the block end effects is to taper the data. This way the middle of the frame (non-zero data being predicted from non-zero values) are given more prominence.

An alternate motivation for the use of tapered data windows comes from the spectral estimation literature. There the use of multi-tapers, i.e. multiple estimates using different windows, is now more common. The windows used can be chosen based on an attempt to get better frequency resolution. Tapered windows come naturally out of this formalism [39].

6.3 Covariance Formulation

For the basic covariance setup, neither the data nor the desired signal are windowed – only the error sequence is windowed. The data matrix ($[M \times N]$) for the covariance formulation becomes

$$\mathbf{X} = \begin{bmatrix} x[-1] & x[0] & \cdots & x[N-2] \\ x[-2] & x[-1] & \cdots & x[N-3] \\ \vdots & \vdots & \ddots & \vdots \\ x[-M] & x[-(M-1)] & \cdots & x[-(M-N+1)] \end{bmatrix}. \tag{6.5}$$

The desired signal is

$$\mathbf{d} = \begin{bmatrix} x[0] & x[1] & \cdots & x[N-1] \end{bmatrix}^T. \tag{6.6}$$

The second pair of plots in Fig. 6.1 shows a schematic representation of the data and error windows for the autocorrelation method.

The Wiener-Hopf equations are

$$\mathbf{X}_e \mathbf{X}_e^H \mathbf{w}_{\text{opt}} = \mathbf{X}_e \mathbf{d}^*, \tag{6.7}$$

where the error weighted data matrix is denoted by

$$\mathbf{X}_e = \mathbf{X}\mathbf{W}_e. \tag{6.8}$$

6.3.1 Solution Methods for the Covariance Formulation

As noted earlier, the correlation matrix $\mathbf{X}_e\mathbf{X}_e^H$ for the covariance method is *not* Toeplitz. This means that neither the Durbin recursion (Appendix I) nor the Levinson algorithm (Appendix J) can be used to solve for the optimal filter coefficients.

Appendix H discusses the Cholesky decomposition which can be used to solve the equations. It is to be noted that the resulting prediction error filter is not guaranteed to be positive definite. A modification of the covariance method can be used to guarantee the minimum phase property [1] (see also [33] for a description of the method). This approach involves generating a set of pseudo-reflection coefficients while solving the equations using a Cholesky decomposition. While this gives a minimum-phase solution, it does not directly minimize a squared error criterion.

6.4 Post-Windowed Method

The windows for the post-windowed method are shown in the last pair of plots in Fig. 6.1. The data matrix ($[M \times N+M]$) is

$$\mathbf{X}_w = \begin{bmatrix} x[-1] & x[0] & \cdots & x[N-1] & 0 & \cdots & 0 \\ x[-2] & x[-1] & \cdots & x[N-2] & x[N-1] & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x[-M] & x[-(M-1)] & \cdots & x[N-M] & x[N-M+1] & \vdots & x[N-1] \end{bmatrix}. \tag{6.9}$$

The correlation matrix is

$$\mathbf{R_{xx}} = \mathbf{X}_w\mathbf{W}_e\mathbf{W}_e\mathbf{X}_w^H. \tag{6.10}$$

The weighted desired signal vector ($[N+M \times 1]$) is

$$\mathbf{d}_w = \begin{bmatrix} x[0] & x[1] & \cdots & x[N-1] & 0 & \cdots & 0 \end{bmatrix}^T. \tag{6.11}$$

This correlation matrix $\mathbf{R_{xx}}$ is *not* Toeplitz. Nonetheless, the resultant prediction error filter is minimum phase. This is shown in [6] – more specifically that the zeros of the filter lie within a circle of radius $\cos(\pi(N+1))$.[1]

[1]This is a somewhat curious result. For the autocorrelation method, we have a Toeplitz matrix and a cross-correlation vector which allow the Durbin recursion to be used. The resulting predictor is minimum-phase. For the post-windowed case, we lose the Toeplitz property and yet the predictor is minimum-phase.

6.5 Minimum Phase Property

The minimum phase property of the prediction error filter is not present for the covariance method. Why is this property important? In some applications, a prediction residual is created. This prediction residual is then coded for transmission. Such an operation is the heart of Linear Prediction (LP) based speech coders. At the decoder, the approximation to the prediction residual is passed through an LP synthesis filter (see Fig. 4.3). This filter is the inverse to the prediction error filter. As such it will be stable if the prediction error filter is minimum phase. We can model the excitation signal to the synthesis filter as the sum of the original residual signal and a noise signal due to the approximation procedure. The original residual signal undergoes a pole / zero cancellation process with the poles of the synthesis filter being cancelled by the zeroes of the analysis filter. So theoretically, the fact that the synthesis filter is unstable does not matter. However the noise, be it due to numerical precision effects or due to coding, will pass only through the synthesis filter and can be accentuated by an unstable filter.

6.6 Time-Varying Filters

In the analysis-synthesis combination, the filter coefficients change from frame to frame. For time-varying filters, it is important to consider the structure of the filter. Different filter configurations will have different initial conditions at the beginning of the frame. In the case of a prediction error filter, which is an FIR filter, the first M samples of the output will depend on state of the filter which in turn differs for different filter structures.

For the combination of an analysis filter ($A(z)$ and a synthesis filter ($1/A(z)$), it is important that they have compatible structures. For instance, a direct form $A(z)$ with time-varying coefficients will have a direct form $1/A(z)$ with time-varying coefficients as its exact inverse. With time varying coefficients, mixing a lattice form $A(z)$ with a direct form $1/A(z)$ will result in a mismatch.

6.7 Pitch Prediction Filters

What we term pitch prediction filters can also be denoted as L-step predictors. These filters play an important role in modelling the pitch of human speech. The conventional one-step predictor (delays from 1 to M) is often called a short-time predictor, linear prediction (LP) analysis filter, or formant prediction filter. The latter term is applied since the prediction error filter flattens (whitens) the spectrum, thereby removing the peaks due to the resonances of the human vocal tract – these resonances are called the formants. The short-time predictor cannot remove redundancies past its largest delay.

A pitch prediction filter is also termed a long-time predictor. Voiced speech contains pseudo-periodic segments with the repetition period being anywhere from 20 to 140 samples at a sampling rate of 8 kHz. A pitch predictor can be used to reduce these redundancies. An example of a simple one coefficient pitch prediction error filter (delay L) is given by

$$W(z) = w_0^* z^{-L}. \tag{6.12}$$

The pitch prediction error filter is shown in Fig. 6.2.

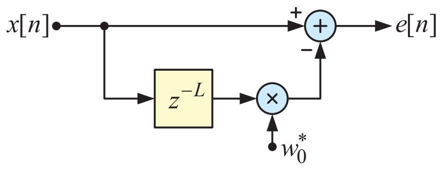


Fig. 6.2 A single coefficient pitch prediction error filter.

In the case of a speech coder, a pitch synthesis filter at the decoder reinserts the pseudo-periodic components into output signal. The pitch filter can be as simple as one coefficient, though often 3 or 5 coefficients around the estimated pitch value are used.[2] The generalization to multiple taps is often of the form

$$W(z) = z^{-L} \sum_{k=0}^{M-1} w_k^* z^{-(k-M_h)}, \tag{6.13}$$

where M_h is chosen to "centre" the pitch filter lags around the nominal pitch lag of L, i.e., for M odd, choose $M_h = (M-1)/2$. Modern speech coders sometimes also use a pitch filter with fractional delay spacing. This can be implemented using fractional delay filters (see the example fractional delay filter design in Section 9.1).

Data windows are generally not used in the formulation of pitch filters. For the one-step predictor, if the block length N used to calculate the needed correlations is significantly larger than maximum lag value of the predictor M, the edge effects of the window play a minor role. For a pitch filter, the maximum lag is the pitch value and is often a significant fraction of the block length. Pitch filters are usually calculated using a covariance formulation. This can lead to pitch synthesis filters which are unstable. However, such filters do indeed model the case when the pitch contribution increases with time.

[2]Reference [21] gives the details of a speech coder based on the ITU-T G.723.1 standard. This coder uses a pitch filter with 5 coefficients.

6.7.1 Pitch Lag Estimation

The pitch lag L for a pitch filter can be determined by a search across allowed values of L. Conceptually, we find the optimal coefficients for each value of L and then determine the mean-square error which results when the filter coefficients are optimized. The pitch lag which gives the lowest mean-square error is chosen.

6.8 Lattice Filter Optimization

A prediction error filter can be implemented in lattice form (repeated here) as shown in Fig. 6.3 The coefficients on this filter are the reflection coefficients found in the Durbin recursion. This filter will be minimum-phase if the reflection coefficients are less than one in magnitude.

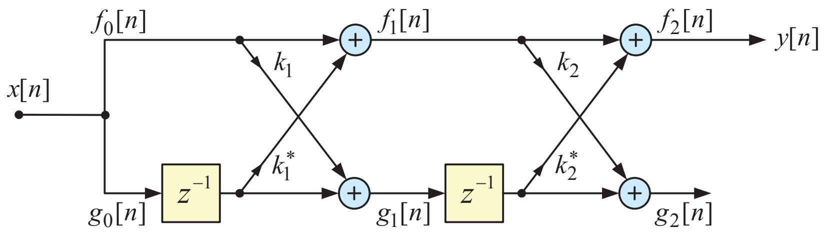


Fig. 6.3 Lattice implementation of a prediction error filter ($M = 2$).

Reference [27] reviews a number of analysis methods based on directly on the lattice form of the prediction error filter. These procedures include ones that calculate the reflection coefficients stage-by-stage as the harmonic or geometric mean of the forward prediction mean-squared error ($E\left[|f_m[n]|^2\right]$) and the backward prediction mean-squared error ($E\left[|g_m[n-1]|^2\right]$) on the lattice. Since these procedures result in reflection coefficients which are bounded by unity, the resultant filter is minimum-phase.

6.9 Discrete All-Pole Model for Periodic Signals

There are limitations in using an all-pole model (the synthesis filter $1/A(z)$) to model speech when the speech is nearly periodic. For periodic inputs, the power spectrum of the signal has discrete components. We can think of this as sampling the power spectrum. Since a sampled spectrum results in time aliasing, the correlation values we compute from the time samples will be aliased. El-Jaroudi and Makhoul [7] have presented the discrete all-pole modelling approach (the term discrete refers to the spectrum having discrete components). The procedure, however, involves a non-linear procedure which tries to find an underlying correlation sequence, which when aliased, matches the observed aliased correlation values [24].

Problems

P6.1 Covariance/Autocorrelation Prediction Errors

(a) Consider a standard covariance approach for finding an M coefficient optimal predictor. Now let the optimum filter coefficients be applied to calculate a prediction residual. Will the prediction gain (in dB) always be positive? Will the prediction error filter be minimum-phase?

(b) Consider an autocorrelation approach for finding an optimal predictor. Let the data window be N samples long and constant over that interval. Let the M filter coefficients be used over the same interval. Will the prediction gain (in dB) always be positive? If not construct a simple example (perhaps with a small number of filter coefficients and a short data frame), for which the energy in the prediction residual is larger than the energy in the signal in the same interval.

(c) If we let the frame size get large with respect to the filter order ($N \gg M$), do the systems tend to give the same solution?

P6.2 Time-Varying Filters

Consider a system which has a prediction error filter $A(z)$, followed by a synthesis filter $1/A(z)$. The M filter coefficients will be determined once per frame of N samples.

(a) Let the prediction error filter be a direct form FIR filter and the synthesis filter be a direct form all-pole IIR filter. These filters will have the same filter coefficients and these coefficients change every N samples. Show that the cascade of these filters is an identity system.

(b) Let the prediction error filter be a lattice FIR filter and the synthesis filter be a direct form all-pole IIR filter. The reflection coefficients used in the lattice filter and the filter coefficients in the synthesis filter change every N samples. Assume that the reflection coefficients and the filter coefficients can be derived from each other using the step-up or step-down procedures outlined in Appendix I. Show that the cascade of these systems shows an anomaly for M samples at the beginning of each frame, after which the system becomes an identity system for the remaining samples in the frame.

Chapter 7

Constrained Minimum Mean-Square Error Filters

In this chapter, we consider finding the coefficients of a filter which minimizes the mean-square error subject to certain constraints.

7.1 Linear Constraints

Let the mean-square error for a filter with coefficients given by the vector $\mathbf{w}$ be $\varepsilon(\mathbf{w})$. First consider a single complex linear constraint on the coefficients,

$$\mathbf{c}^H \mathbf{w} = v. \tag{7.1}$$

We can write this constraint so that the target value is zero,

$$\mathbf{c}^H \mathbf{w} - v = 0. \tag{7.2}$$

Since the quantities in the constraint equation are complex, we require that the real and imaginary components are each zero.

We take a brief digression into a discussion of the method of Lagrange multipliers. We will form an augmented error function using a Lagrange multiplier $\lambda = \lambda_r + j\lambda_i$.

$$\begin{aligned} \tilde{\varepsilon}(\mathbf{w}) &= \varepsilon(\mathbf{w}) + \lambda_r \mathrm{Re}[\mathbf{c}^H \mathbf{w} - v] + \lambda_i \mathrm{Im}[\mathbf{c}^H \mathbf{w} - v] \\ &= \varepsilon(\mathbf{w}) + \mathrm{Re}[\lambda^* (\mathbf{c}^H \mathbf{w} - v)]. \end{aligned} \tag{7.3}$$

Consider taking the derivative with respect to $\mathbf{w}^*$ (see Appendix A), then set the result to zero to get

$$\mathbf{R}\mathbf{w} - \mathbf{r}_{\mathbf{x}d} + \lambda^* \mathbf{c} = \mathbf{0}. \tag{7.4}$$

The augmented equations to solve for both $\mathbf{w}$ and λ are

$$\begin{bmatrix} \mathbf{R} & \mathbf{c} \\ \mathbf{c}^H & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \lambda^* \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{\mathbf{x}d} \\ v \end{bmatrix}. \tag{7.5}$$

7.1.1 Multiple Constraints

We now generalize to K linear constraints (but with K less than M where M is the number of filter coefficients). The constraint equations are

$$\mathbf{C}_c\mathbf{w} - \mathbf{v} = \mathbf{0}. \tag{7.6}$$

The (conjugated) Lagrange multipliers can be placed in a vector $\boldsymbol{\lambda}$ to give us the following set of linear equations,

$$\begin{bmatrix} \mathbf{R} & \mathbf{C}_c \\ \mathbf{C}_c^H & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{\mathbf{x}d} \\ \mathbf{v} \end{bmatrix}. \tag{7.7}$$

This system of equations can be expressed as the coupled equations

$$\mathbf{R}\mathbf{w} + \mathbf{C}_c\boldsymbol{\lambda} = \mathbf{r}_{\mathbf{x}d} \tag{7.8a}$$
$$\mathbf{C}_c^H\mathbf{w} = \mathbf{v}. \tag{7.8b}$$

We can eliminate $\boldsymbol{\lambda}$ to give us a direct solution for $\mathbf{w}$. From Eq. (7.8a) we get

$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{r}_{\mathbf{x}d} - \mathbf{R}^{-1}\mathbf{C}_c\boldsymbol{\lambda}. \tag{7.9}$$

Then substituting for $\mathbf{w}$ in Eq. (7.8b), we get

$$\mathbf{C}_c^H\mathbf{R}^{-1}\mathbf{r}_{\mathbf{x}d} - \mathbf{C}_c^H\mathbf{R}^{-1}\mathbf{C}_c\boldsymbol{\lambda} = \mathbf{v}. \tag{7.10}$$

This then gives us a solution for $\boldsymbol{\lambda}$,

$$\boldsymbol{\lambda} = (\mathbf{C}_c^H\mathbf{R}^{-1}\mathbf{C}_c)^{-1}(\mathbf{C}_c^H\mathbf{R}^{-1}\mathbf{r}_{\mathbf{x}d} - \mathbf{v}). \tag{7.11}$$

We now substitute this value of $\boldsymbol{\lambda}$ into Eq. (7.8a) and solve for $\mathbf{w}$,

$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{r}_{\mathbf{x}d} - \mathbf{R}^{-1}\mathbf{C}_c(\mathbf{C}_c^H\mathbf{R}^{-1}\mathbf{C}_c)^{-1}(\mathbf{C}_c^H\mathbf{R}^{-1}\mathbf{r}_{\mathbf{x}d} - \mathbf{v}). \tag{7.12}$$

Noting that $\mathbf{w}_u = \mathbf{R}^{-1}\mathbf{r}_{\mathbf{x}d}$ is the solution to the minimum mean-square error problem, we can get the alternate expression,

$$\mathbf{w} = \mathbf{w}_u - \mathbf{R}^{-1}\mathbf{C}_c(\mathbf{C}_c^H\mathbf{R}^{-1}\mathbf{C}_c)^{-1}(\mathbf{C}_c^H\mathbf{w}_u - \mathbf{v}). \tag{7.13}$$

7.1.2 Subspace Interpretation of Linear Constraints

For the linearly constrained problem, we can split the space of solutions into subspaces, one constrained and the other unconstrained. The columns of matrix $\mathbf{C}_c$ establish the subspace for the K constraints. The orthogonal complement denoted by $\mathbf{C}_u$ contains the $M-K$ basis (column) vectors for the unconstrained subspace. The weight vector can then be written as

$$\begin{aligned}\mathbf{w} &= \begin{bmatrix}\mathbf{C}_c\,\mathbf{C}_u\end{bmatrix}\begin{bmatrix}\mathbf{q}_c\\ \mathbf{q}_u\end{bmatrix}\\ &= \mathbf{C}_c\mathbf{q}_c + \mathbf{C}_u\mathbf{q}_u\\ &= \mathbf{w}_c + \mathbf{w}_u.\end{aligned} \tag{7.14}$$

where $\mathbf{q}_c$ is $[K \times 1]$ and $\mathbf{q}_u$ is $[M-K \times 1]$. The matrix $\mathbf{C}_u$ is orthogonal to $\mathbf{C}_c$,

$$\begin{aligned}\mathbf{C}_u^H\mathbf{C}_c &= \mathbf{0} \qquad [M-K \times K]\\ \mathbf{C}_c^H\mathbf{C}_u &= \mathbf{0} \qquad [K \times M-K].\end{aligned} \tag{7.15}$$

Equation (7.14) reflects that fact that any vector can be expressed as a linear combination of two vectors, one from a given subspace and the other from the orthogonal complement of that subspace.

The matrix $\mathbf{C}_u$ is the orthogonal complement to $\mathbf{C}_c$ and contains basis vectors for the null space of $\mathbf{C}_c$. The basis vectors can be found by manipulating $\mathbf{C}_c^T$ into a reduced row echelon form (increasing number of leading zeros). The solutions of the homogeneous equations enforcing the orthogonality then gives a set of basis vectors in $\mathbf{C}_u$. Alternate methods include use of the singular-value decomposition or a QR factorization.[1]

[1]The MATLAB routine `null` can be used to find the basis vectors for the null space of $\mathbf{C}_c^T$. This routine can optionally return the basis functions determined using an SVD or using the reduced row echelon form (`rref`).

The linear constraints give us a value for $\mathbf{w}_c$ as shown in the following steps,

$$\mathbf{C}_c^H \mathbf{w} = \mathbf{v} \tag{7.16a}$$

$$\mathbf{C}_c^H(\mathbf{C}_c\mathbf{q}_c + \mathbf{C}_u\mathbf{q}_u) = \mathbf{v} \tag{7.16b}$$

$$\mathbf{C}_c^H\mathbf{C}_c\mathbf{q}_c = \mathbf{v} \tag{7.16c}$$

$$\mathbf{q}_c = (\mathbf{C}_c^H\mathbf{C}_c)^{-1}\mathbf{v} \tag{7.16d}$$

$$\mathbf{w}_c = \mathbf{C}_c(\mathbf{C}_c^H\mathbf{C}_c)^{-1}\mathbf{v} \tag{7.16e}$$

In going from Eq. (7.16b) to Eq. (7.16c), we have used the fact that $\mathbf{C}_c$ is orthogonal to $\mathbf{C}_u$. The final expression finds the component of $\mathbf{w}$ which enforces the constraints.

Note that if the columns of $\mathbf{C}_c$ are orthogonal, the product $\mathbf{C}_c^H\mathbf{C}_c$ is the $[K \times K]$ identity matrix. Indeed, $\mathbf{C}_c$ can be orthogonalized using a QR factorization (MATLAB function `qr`) or using a Gram-Schmidt procedure. The QR factorization returns a matrix $\widetilde{\mathbf{C}}_c$ with orthonormal columns as a factorization of $\mathbf{C}_c$.

$$\mathbf{C}_c = \mathbf{Q}\widetilde{\mathbf{C}}_c, \tag{7.17}$$

where $\mathbf{Q}$ is a unitary $[M \times M]$ matrix, and $\widetilde{\mathbf{C}}_c$ is a $[M \times K]$ matrix. The constraint equations then become

$$\mathbf{C}_c\mathbf{w} = \mathbf{v}, \tag{7.18a}$$

$$\mathbf{Q}\widetilde{\mathbf{C}}_c\mathbf{w} = \mathbf{v}, \tag{7.18b}$$

$$\widetilde{\mathbf{C}}_c\mathbf{w} = \mathbf{Q}\mathbf{v}. \tag{7.18c}$$

We can identify modified constraint values $\tilde{\mathbf{v}} = \mathbf{Q}\mathbf{v}$.

There are two interpretations of the separation of the constrained and unconstrained parts of the problem. The first which appears in Fig. 7.1, shows the input vector $\mathbf{x}[n]$ separately processed by $\mathbf{C}_c$ and $\mathbf{C}_u$. The output of the $\mathbf{C}_c$ (resp. $\mathbf{C}_u$) block is given by $\mathbf{C}_c^H\mathbf{x}[n]$ (resp. $\mathbf{C}_u^H\mathbf{x}[n]$). The M input samples become a transformed vector of length K for the constrained branch and of length $M - K$ in the unconstrained branch. The result of multiplication by the transformed weight vectors results in a single output sample $y[n]$. Once the constraints have been satisfied, $\mathbf{q}_c$ is fixed and minimization of the error involves only $\mathbf{q}_u$. We could then just as well minimize the mean-square error between the desired signal $d[n]$ and the (unconstrained) output of the lower branch.

The separation of the constrained and unconstrained subspaces can be achieved using projection operators. We see from the Equations (7.16) that

$$\mathbf{w}_c = \mathbf{C}_c(\mathbf{C}_c^H\mathbf{C}_c)^{-1}\mathbf{C}_c^H\mathbf{w}. \tag{7.19}$$

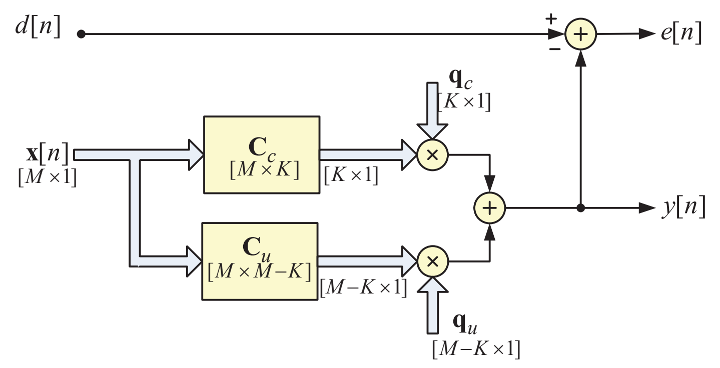


Fig. 7.1 Subspace decomposition based on the constrained and unconstrained subspaces

This equation can be written as

$$\mathbf{w}_c = \mathbf{P}_c\mathbf{w}, \tag{7.20}$$

where the projection operator is

$$\mathbf{P}_c = \mathbf{C}_c(\mathbf{C}_c^H\mathbf{C}_c)^{-1}\mathbf{C}_c^H. \tag{7.21}$$

The complementary projection (from $\mathbf{w}$ to $\mathbf{w}_u$) is

$$\mathbf{P}_u = \mathbf{I} - \mathbf{P}_c. \tag{7.22}$$

Each projection operator is idempotent ($\mathbf{P}_c^n = \mathbf{P}_c$) and the two projections are complementary ($\mathbf{P}_c + \mathbf{P}_u = \mathbf{I}$) and orthogonal ($\mathbf{P}_c\mathbf{P}_u = \mathbf{0}$).

The block diagram in Fig. 7.2 shows the input of length M being processed by two projection operators $\mathbf{P}_c$ and $\mathbf{P}_u$. The output of each projection is of length M. The weight vectors $\mathbf{w}_c$ and $\mathbf{w}_u$ multiply the projection outputs to form a single output sample $y[n]$.

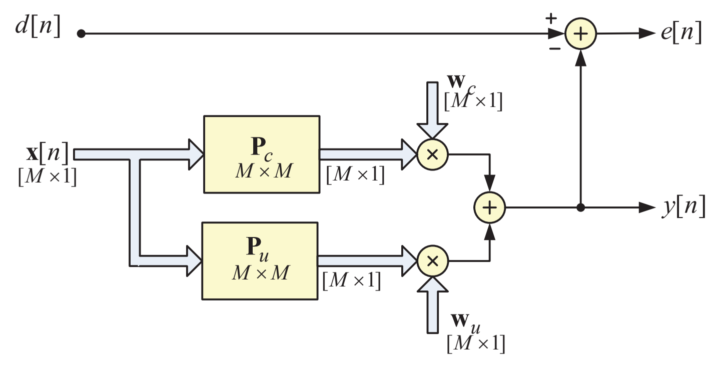


Fig. 7.2 Subspace decomposition based on projecting the constrained and unconstrained subspaces

7.1.3 Approximate Linear Constraints

We will look at means to get the effect of linear constraints without the full machinery of the constraint approaches described above. The basic idea is that conceptually we will add components to the input signal which appear only in the constraint subspace. Giving these components sufficient amplitude (recalling that the mean-square error gives large signals large weights) will allow the constraints to be (closely) satisfied. In the final analysis, these components will lead to correlation values which are added to the true signal correlation values.

Now consider a sequence $\mathbf{x}[n]$ ($[M \times 1]$) applied to $\mathbf{C}_c$. The result is

$$\mathbf{x}_c[n] = \mathbf{C}_c^H \mathbf{x}[n], \tag{7.23}$$

where $\mathbf{x}_c[n]$ is a $[K \times 1]$ vector. The correlation matrix for $\mathbf{x}_c[n]$ is

$$\mathbf{R}_{x_c x_c} = \mathbf{C}_c^H \mathbf{R}_{\mathbf{xx}} \mathbf{C}_c, \tag{7.24}$$

where $\mathbf{R}_{\mathbf{xx}}$ is the correlation matrix for $\mathbf{x}[n]$. We can also create a "desired signal"

$$d_c[n] = \mathbf{x}_c^T[n](\mathbf{C}_c^H \mathbf{C}_c)^{-1}\mathbf{v}. \tag{7.25}$$

The cross-correlation between $\mathbf{x}_c[n]$ and $d_c[n]$ is

$$\begin{aligned} \mathbf{r}_{\mathbf{x}_c d_c} &= E[\mathbf{x}_c[n] d_c^*[n]] \\ &= E[\mathbf{C}_c^H \mathbf{x}[n]\mathbf{x}^H[n]\mathbf{C}_c(\mathbf{C}_c^H \mathbf{C}_c)^{-1}\mathbf{v}] \\ &= \mathbf{C}_c^H \mathbf{R}_{\mathbf{xx}} \mathbf{C}_c (\mathbf{C}_c^H \mathbf{C}_c)^{-1}\mathbf{v} \end{aligned} \tag{7.26}$$

Using the correlation of the subspace signal $\mathbf{x}_c[n]$ and the cross-correlation with $d_c[n]$, the Wiener-Hopf equations become

$$\mathbf{C}_c^H \mathbf{R}_{\mathbf{xx}} \mathbf{C}_c \mathbf{q}_c = \mathbf{C}_c^H \mathbf{R}_{\mathbf{xx}} \mathbf{C}_c (\mathbf{C}_c^H \mathbf{C}_c)^{-1}\mathbf{v}. \tag{7.27}$$

Assuming a full rank $\mathbf{R}_{\mathbf{xx}}$, the equation is satisfied by

$$\mathbf{q}_c = (\mathbf{C}_c^H \mathbf{C}_c)^{-1}\mathbf{v}. \tag{7.28}$$

This is the same result found earlier in Eq. (7.16d).

Now we want to reflect back these results to before the $\mathbf{C}_c$ block. First we note that if we have a signal $\mathbf{x}[n]$ and find its projection to the constraint subspace $\mathbf{P}_c\mathbf{x}[n]$, and then apply $\mathbf{C}_c$, then

$$\begin{aligned}\mathbf{x}_c[n] &= \mathbf{C}_c^H \mathbf{P}_c \mathbf{x}[n] \\ &= \mathbf{C}_c^H \mathbf{x}[n].\end{aligned} \tag{7.29}$$

With this fact we find the contribution to the correlation before the $\mathbf{C}_c$ block

$$\mathbf{R}'_{\mathbf{xx}} = \mathbf{P}_c \mathbf{R}_{\mathbf{xx}} \mathbf{P}_c. \tag{7.30}$$

We note that $\mathbf{P}_c = \mathbf{P}_c^H$. Indeed, if we use $\mathbf{R}'_{\mathbf{xx}}$ instead of $\mathbf{R}_{\mathbf{xx}}$ in Eq. (7.24) or Eq. (7.26), the result is unchanged.

This value of $\mathbf{R}'_{\mathbf{xx}}$ represents the correlation for a signal which affects only the constraint subspace. We can make things even easier, by assuming that the components for the correlation in the constraint space are generated from zero-mean white noise, with variance a^2 projected into the constraint space. Then the augmentation to the correlation is just

$$\mathbf{R}'_{\mathbf{xx}} = a^2 \mathbf{P}_c. \tag{7.31}$$

Note that the projection matrix is non-negative definite, see Problem P5.1. The corresponding augmentation to the cross-correlation is

$$\begin{aligned}\mathbf{r}'_{\mathbf{x}d_c} &= a^2 \mathbf{P}_c \mathbf{C}_c (\mathbf{C}_c^H \mathbf{C}_c)^{-1} \mathbf{v} \\ &= a^2 \mathbf{C}_c (\mathbf{C}_c^H \mathbf{C}_c)^{-1} \mathbf{v}.\end{aligned} \tag{7.32}$$

The Wiener-Hopf equation ($\mathbf{R}'_{\mathbf{xx}}\mathbf{w} = \mathbf{r}'_{\mathbf{x}d_c}$) becomes

$$\begin{aligned}a^2 \mathbf{P}_c \mathbf{w} &= a^2 \mathbf{C}_c (\mathbf{C}_c^H \mathbf{C}_c)^{-1} \mathbf{v} \\ \mathbf{w}_c &= \mathbf{C}_c (\mathbf{C}_c^H \mathbf{C}_c)^{-1} \mathbf{v}\end{aligned} \tag{7.33}$$

We have used Eq. (7.20) to go from the first line to the second line. The last line is the same as Eq. (7.16e).

7.1.4 Minimum-Variance Distortionless Response

A linearly constrained minimum mean-square error filter can be used to estimate the power spectrum of a signal. The problem is set up such that the response of the filter at a frequency ω_c is constrained to be unity (distortionless at ω_c). The overall filter should minimize the mean-square error subject to this constraint. This formulation results in the minimum-variance distortionless response (MVDR). We notice that without the constraint, the filter would have all coefficients be zero. The resulting constrained filter response has a peak at $\omega = \omega_c$, and sidelobe responses away from ω_c are suppressed. We can sweep $\omega + c$ through a range of interest and apply the filter for each value of ω_c to get an estimate of the power spectrum of the input signal at $\omega = \omega_c$.

The constraint equation is

$$\mathbf{C}_c^H \mathbf{w} = v, \tag{7.34}$$

with

$$\mathbf{C}_c = \begin{bmatrix} 1 \\ e^{j\omega_c} \\ e^{j\omega_c 2} \\ \vdots \\ e^{j\omega_c (M-1)} \end{bmatrix}, \tag{7.35}$$

and $v = 1$. We have used a matrix notation $\mathbf{C}_c$ for the constraint matrix for compatibility with the equations developed above. The desired value for the minimization is zero, giving $\mathbf{r}_{\mathbf{x}d} = \mathbf{0}$. Now we can solve the equations directly using Eq. (7.12),

$$\mathbf{w} = \mathbf{R}^{-1}\mathbf{C}_c(\mathbf{C}_c^H \mathbf{R}^{-1}\mathbf{C}_c)^{-1} v. \tag{7.36}$$

Now for the approximate constraint approach applied to MVDR, we set $\mathbf{r}_{\mathbf{x}d}$ to be $\mathbf{r}'_{\mathbf{x}d_c}$ and $\mathbf{R}_{\mathbf{xx}}$ to be $\mathbf{R} + \mathbf{R}'_{\mathbf{xx}}$. The equations to be solved are the standard Wiener-Hopf equations,

$$\begin{aligned} \mathbf{R}_{\mathbf{xx}}\mathbf{w} &= \mathbf{r}_{\mathbf{x}d} \\ (\mathbf{R} + a^2\mathbf{P}_c)\mathbf{w} &= a^2\mathbf{C}_c(\mathbf{C}_c^H\mathbf{C}_c)^{-1}\mathbf{v}. \end{aligned} \tag{7.37}$$

Substituting the value of $\mathbf{C}_c$ into this equation, we get

$$(\mathbf{R} + a^2\mathbf{B})\mathbf{w} = \frac{a^2}{M}\mathbf{C}_c v, \tag{7.38}$$

where $\mathbf{B}$ is an $[M \times M]$ matrix with element $\mathbf{B}_{k,l}$ equal to $e^{j\omega_c(l-k)}$. With an appropriate (large) value for a, the solution to this equation approaches that of the true constrained problem. Note that $\mathbf{B}$ is Toeplitz, so that equations can be solved using the Levinson algorithm (Appendix J).

For a so-called quiet environment with the input signal being white noise ($\mathbf{R} = \sigma^2\mathbf{I}$), a considerable simplification occurs

$$\mathbf{w} = \frac{1}{M}\mathbf{C}_c v. \tag{7.39}$$

It is easy to verify that the frequency response of this filter is unity at $\omega = \omega_c$.

7.2 Augmented Quadratic Error

In this section we consider an augmented squared-error criterion,

$$\varepsilon' = \varepsilon + a\mathbf{w}^H\mathbf{Q}\mathbf{w}. \tag{7.40}$$

With this formulation and a fixed value for a, we can readily see that we can solve the problem by augmenting the correlation matrix,

$$(\mathbf{R}_{\mathbf{xx}} + a\mathbf{Q})\mathbf{w} = \mathbf{r}_{\mathbf{x}d}. \tag{7.41}$$

Indeed this is a generalization of techniques used in LP coding to ensure better spectral fits and/or improve numerical conditioning [20]. The simple case with $\mathbf{Q}$ being an identity matrix is known as *diagonal loading*, *ridge regression*, or *white noise compensation*. In all of these cases, a is fixed, or a fixed fraction of $r_{xx}[0]$.

7.2.1 Eigenfilters

In some cases, the additional quadratic term plays the role of a Lagrange term. Given a signal with correlation matrix $\mathbf{R}_{ss}$ and subject to additive noise (uncorrelated with the signal component) with correlation matrix $\mathbf{R}_{nn}$, we wish to find the filter which maximizes the signal-to-noise ratio at the output of a filter. The powers of the signal and noise components at the output of the filter are

$$\begin{aligned} P_s &= \mathbf{w}^H\mathbf{R}_{ss}\mathbf{w}, \\ P_n &= \mathbf{w}^H\mathbf{R}_{nn}\mathbf{w}. \end{aligned} \tag{7.42}$$

For this case of coloured noise ($\mathbf{R}_{nn}$ is not a scaled identity matrix), we factor $\mathbf{R}_{nn}$ using the Cholesky decomposition (Appendix H)

$$\mathbf{R}_{nn} = \mathbf{L}\mathbf{L}^H. \tag{7.43}$$

Then defining $\mathbf{y} = \mathbf{L}^H\mathbf{w}$, we can write the signal and noise outputs as follows

$$\begin{aligned} P_x &= \mathbf{y}^H\mathbf{A}\mathbf{y}, \\ P_n &= \mathbf{y}^H\mathbf{y}, \end{aligned} \tag{7.44}$$

where $\mathbf{A} = \mathbf{L}^{-1}\mathbf{R}_{\mathbf{xx}}\mathbf{L}^{-H}$. We can now form the signal-to-noise ratio as the Rayleigh quotient,

$$\mathrm{SNR} = \frac{\mathbf{y}^H\mathbf{A}\mathbf{y}}{\mathbf{y}^H\mathbf{y}}. \tag{7.45}$$

The vector $\mathbf{y}$ which maximizes this ratio is the eigenvector $\mathbf{q}_{\max}$ of matrix $\mathbf{A}$ corresponding the largest eigenvalue of $\mathbf{A}$. The coefficients $\mathbf{q}_{\max}$ are known as an eigenfilter. Given $\mathbf{q}_{\max}$, the filter weights are

$$\mathbf{w} = \mathbf{L}^{-H}\mathbf{q}_{\max}. \tag{7.46}$$

This is a constrained problem because maximizing the Rayleigh quotient is equivalent to maximizing the numerator of the fraction, subject to a unity constraint on the denominator. The simpler case of white noise has $\mathbf{R}_{nn} = \sigma_n^2\mathbf{I}$, giving the optimal filter

$$\mathbf{w} = \mathbf{q}_{\max}, \tag{7.47}$$

where now $\mathbf{q}_{\max}$ is an eigenvector of $\mathbf{R}_{\mathbf{ss}}$. Note that this filter does not depend on the value of σ_n^2.

7.2.2 Eigen-Windows

Another eigenfilter appears in window designs. The problem here is to maximize the fraction of energy in the frequency range $[-\beta\pi, \beta\pi]$. The total energy is given by Parseval's relation in the time domain as

$$P_s = \mathbf{w}^H\mathbf{w}. \tag{7.48}$$

The energy within the bandwidth specified is

$$P_w = \frac{1}{2\pi}\int_{-\beta\pi}^{\beta\pi} |W(\omega)|^2 \, d\omega. \tag{7.49}$$

We can define a brick-wall response

$$H_\beta(\omega) = \begin{cases} 1, & |\omega| < \beta\pi, \\ 0, & \beta\pi \le |\omega| \le \pi. \end{cases} \tag{7.50}$$

Now we can write

$$P_w = \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|W(\omega)H_\beta(\omega)\right|^2 d\omega. \tag{7.51}$$

Using the Parseval relationship we calculate the power in the time domain

$$
\begin{aligned}
P_w &= \sum_{n=-\infty}^{\infty} |w[n] * h_\beta[n]|^2 \\
&= \beta^2 \sum_{n=-\infty}^{\infty} |w[n] * \operatorname{sinc}(\beta n)|^2 \\
&= \beta^2 \sum_{n=-\infty}^{\infty} |\sum_{k=0}^{M-1} w[k] \operatorname{sinc}\big(\beta(n-k)\big)|^2 \\
&= \beta^2 \sum_{k=0}^{M-1}\sum_{l=0}^{M-1} w[k]w^*[l] \sum_{n=-\infty}^{\infty} \operatorname{sinc}\big(\beta(n-k)\big)\operatorname{sinc}\big(\beta(n-l)\big) \\
&= \beta \sum_{k=0}^{M-1}\sum_{l=0}^{M-1} w[k]w^*[l] \operatorname{sinc}\big(\beta(k-l)\big).
\end{aligned}
\tag{7.52}
$$

To get to the last line, we have used an identity developed in Problem P7.1. Now we can write P_w in vector-matrix notation,

$$P_w = \mathbf{w}^H \mathbf{D}\mathbf{w}, \tag{7.53}$$

where $\mathbf{D}$ is a an $[M \times M]$ matrix with element $\mathbf{D}_{k,l}$ equal to $\beta \operatorname{sinc}\big(\beta(k-l)\big)$.

With that preliminary work to show that the partial power P_w can be written as a quadratic form, we can solve our maximization problem again using the Rayleigh quotient. The window function with coefficients $\mathbf{w}$ which maximizes the fractional power in the interval $-\beta\pi$ to $\beta\pi$ is the eigenvector corresponding to the largest eigenvalue of $\mathbf{D}$. This is a *discrete prolate spheroidal sequence* (DPSS), also known as a Slepian sequence, and can be calculated using the MATLAB routine `dpss`. Reference [19] shows an example of a window designed using this procedure.

Problems

P7.1 Sinc Function Identity

In this problem we develop an identity involving $\operatorname{sinc}(\cdot)$ functions,

$$\operatorname{sinc}\big(\beta(t-u)\big) = \beta \sum_{n=-\infty}^{\infty} \operatorname{sinc}\big(\beta(n-t)\big)\operatorname{sinc}\big(\beta(n-u)\big). \tag{P7.1-1}$$

Here we have used a normalized sinc function defined by

$$\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x}. \tag{P7.1-2}$$

The proof of this identity will be motivated from reconstruction of a bandlimited function from its samples. Let $y(t)$ be a bandlimited signal which is zero for $|f| > W_y$. Sample $y(t)$ at $t = nT$ to form $y[n]$, where $T \leq 1/(2W_x$, i.e., the sampling theorem is satisfied. Then we can reconstruct $y(t)$ from its samples using an appropriate lowpass filter $h(t)$,

$$y(t) = \sum_{n=-\infty}^{\infty} y[n]h(t - nT). \tag{P7.1-3}$$

Let the filter have an ideal lowpass response with cutoff at $f = W_h$.

(a) Show that $h(t) = 2W_h \operatorname{sinc}(2W_h t)$.

Now we have

$$y(t) = \sum_{n=-\infty}^{\infty} y[n] \operatorname{sinc}(W_h(t - nT). \tag{P7.1-4}$$

Now let $y(t)$ itself be the response of a time shifted ideal lowpass response with cutoff W_y. The shift will be such that the peak of the $y(t)$ occurs at time $t = \tau$.

(b) Express $y(t)$ and $y[n]$ each in terms of the sinc function.

(c) Resolve the constants (W_h, W_y, and T) and manipulate the result to prove the identity. Hint: $\operatorname{sinc}(t)$ is an even function of t.

Part II

Mean-square Filtering: Examples

Chapter 8

Examples – Stationary Stochastic Signals

This chapter illustrates the solution of minimum mean-square error filtering problems for wide-sense stationary stochastic signals.

8.1 Example: Linear Prediction From the Average Speech Spectrum

In this example, we consider the prediction of speech signals based on the long-term average power spectrum of speech signals, modified to reflect realistic filtering conditions. The correlations needed to solve for an optimal predictor are found from the inverse Fourier transform of the power spectrum. From the power spectrum, we can evaluate the maximum prediction gain available when the predictor order grows. Finally, we test the performance of the predictor, designed based on an average spectrum, on real samples of speech.

8.1.1 Power Spectrum to Correlations

The correlations needed for solving for the predictor coefficients are calculated as follows.

- We use the polynomial approximation to the long-term average speech spectrum from [16]. The formula given there is the value at the "mouth reference point". Since scaling does not affect the predictor, we need not alter the scaling. The power spectrum given there is valid for frequencies 100–8000 Hz. The power spectrum goes to zero ($-\infty$ on a dB scale) at zero frequency. To start, we densely sample the spectrum in the interval 0–8000 Hz.

- We modify the spectrum using a shaping filter (the modified IRS filter – sending side, from ITU-T Recommendation P.830 [15]). This filter models the the effect of the transmission from the talker to a telephone central office. The digital filter which implements this response has coefficients taken from ITU-T Recommendation G.191 [17]. We use the 16 kHz sampling rate version of this filter. The frequency response of the filter is sampled and used to modify the long-term average speech spectrum

- We apply a low-pass filter to the spectrum (again taken from ITU-T G.191). This filter is a half-band filter which serves as an anti-aliasing filter before we change the sampling rate.
- The inverse Discrete Fourier Transform (DFT) of the sampled filtered power spectrum gives an autocorrelation sequence at a 16 kHz sampling rate.
- We then add the effect of white noise to the autocorrelation sequence. This is accomplished by modifying the zeroth correlation coefficient to give a signal-to-noise ratio of $\text{SNR}_{\text{dB}} = 60$. We can consider the noise to model quantization effects.
- We then subsample the autocorrelation sequence to get one at an 8 kHz sampling rate. The subsampled autocorrelation sequence is the autocorrelation sequence for a subsampled signal. Note that the combination of adding noise and subsampling (with its attendant small but non-zero aliasing) will fill in the spectral holes in the original power spectrum. The correlation values are used to solve for the optimal predictor coefficients.
- We find the power spectrum corresponding at the 8 kHz sampling rate using a DFT. The power spectrum is used to calculate the upper bound to the prediction gain Eq. (4.58).

The power spectrum for the filtered long-term average speech spectrum is shown in Fig. 8.1. The "curls" at the ends of the spectrum are due to the white noise compensation.

8.1.2 Prediction Gain

Now we can design predictors based on the correlation corresponding to this power spectrum. The prediction gain for different prediction orders is shown in Fig. 8.2. A prediction gain of 0 dB means that the prediction residual has the same energy as the input signal. A positive prediction gain in dB means that the energy of the prediction residual is decreased. The constant line in the figure is the estimate of the prediction gain for an optimal infinite length predictor. The prediction gain was calculated based on Eq. (4.58) using sums to approximate the integrals.

The spectrum of the order 10 prediction error filter is plotted as the lower dashed line in Fig. 8.1. This spectrum has 5 valleys, each corresponding to a pair of zeros in the prediction error filter. Note that as required (see Eq. (4.17), the log spectrum has a zero mean, i.e., it has the same area above and below the 0 dB line. That figure also shows the power spectrum after the prediction error filter (upper dashed line). The power spectrum after the prediction error filter is somewhat flatter (whiter) and has an overall energy which is about 4 dB smaller that the original power spectrum (see the prediction gain for a tenth order filter in Fig. 8.2).

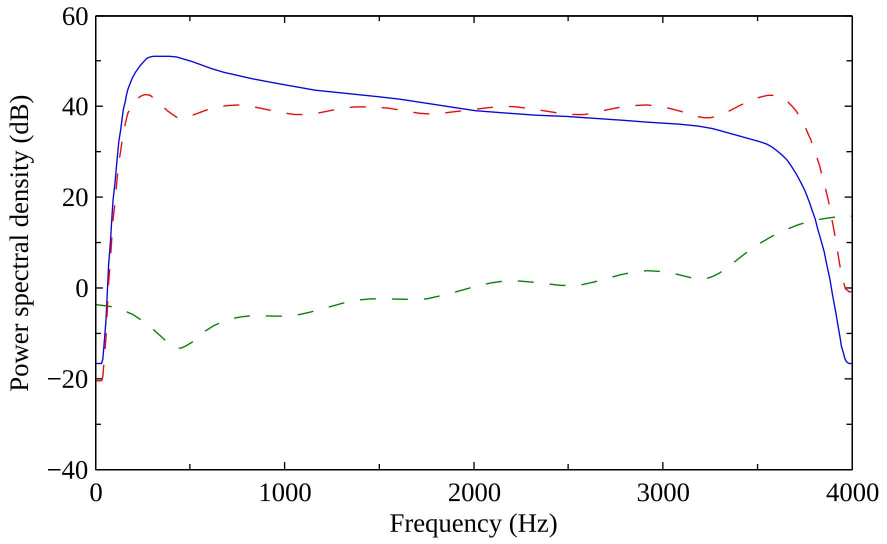


Fig. 8.1 Long-term average power spectrum of filtered speech (solid line). The spectrum is based on ITU-T P.50, filtered and resampled to 8 kHz. The lower dashed line is the frequency response of the prediction error filter designed based on the long-term average power spectrum ($M = 10$). The upper dashed line is the power spectrum after applying the prediction error filter to the long-term average speech power spectrum.

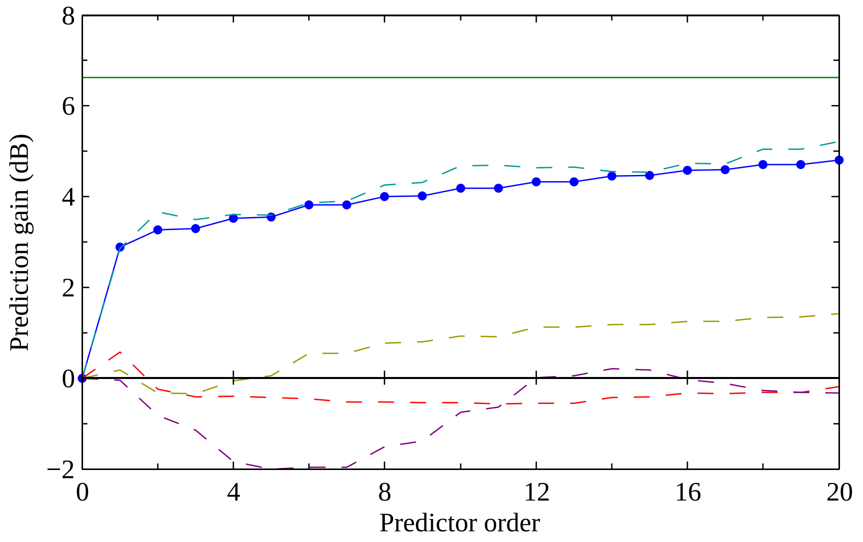


Fig. 8.2 Prediction gain versus predictor order. The solid line with circles is the prediction gain for the spectrum given in Fig. 8.1. The upper flat line is the prediction gain for an infinite order predictor. The dashed lines give the prediction gain for a number of test utterances using the predictors designed for the long-term speech spectrum.

The dashed lines in Fig. 8.2 show the prediction gain achieved when using the predictor coefficients optimized for the long-term average spectrum but applied to sample utterances (2 male and 2 female speakers). The speech has been filtered using the same modified IRS filtering used to shape the long-term average power spectrum. When the dotted lines fall below 0 dB, the prediction residual energy is increased relative to the input signal energy. By chance, one of the test utterances happens to roughly follow the prediction gain curve for the average long-term spectrum.

This example shows that an adaptive predictor is really what is needed – a predictor based on an average spectrum will benefit some signals, but can do harm to other signals. Indeed an adaptive predictor should react to local speech statistics rather than the average spectrum whether that average spectrum is speaker specific or for an ensemble of speakers. Adaptive predictors based on the least squares formulation are shown in Chapter 10.

8.2 Example: DPCM

Consider the Differential Pulse Code Modulation (DPCM) system shown in Fig. 8.3. This is a feedback system around a quantizer. In z-transform notation, the system is described by the following equations,

$$\begin{aligned} \hat{X}(z) &= \hat{E}(z) + \tilde{X}(z), && \text{Output signal,} \\ E(z) &= X(z) - \tilde{X}(z), && \text{Difference signal,} \\ \hat{E}(z) &= E(z) + Q(z), && \text{Quantization.} \end{aligned} \tag{8.1}$$

The quantizer has been modelled as adding a noise $Q(z)$ to its input. It is straightforward to use these relationships to show that

$$\hat{X}(z) = X(z) + Q(z). \tag{8.2}$$

This is the standard relationship for such a system that shows that the reconstructed signal differs from the input signal by the quantization noise introduced on the difference signal $E(z)$. The time-domain equation corresponding to Eq. (8.2) is

$$\hat{x}[n] = x[n] + q[n]. \tag{8.3}$$

The predictor acts on past values of $\hat{x}[n]$ to produce an estimate of $x[n]$. The equations to be solved to determine the predictor coefficients are

$$\mathbf{R}_{\hat{x}\hat{x}}\mathbf{w}_{\text{opt}} = \mathbf{r}_{\hat{x}x}. \tag{8.4}$$

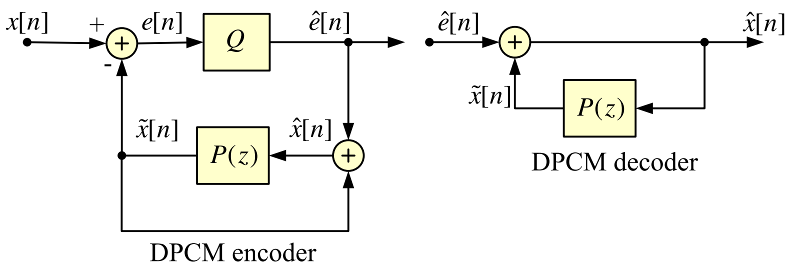


Fig. 8.3 DPCM system

We will assume that the quantization noise is zero-mean, white, with variance σ_q^2, and is uncorrelated with the signals. Then the correlation terms for $\hat{x}[n]$ are

$$r_{\hat{x}\hat{x}}[k] = r_{xx}[k] + \sigma_q^2 \delta_{k0}. \tag{8.5}$$

For the cross-correlation terms, we get

$$r_{x\hat{x}}[k] = r_{xx}[k]. \tag{8.6}$$

The vector-matrix equations to be solved are then

$$(\mathbf{R_{xx}} + \sigma_q^2 \mathbf{I})\mathbf{w}_{\text{opt}} = \mathbf{r}_{\mathbf{x}x}. \tag{8.7}$$

8.2.1 First Order Predictor

For this example, we will consider a first order predictor. The predictor has a coefficient

$$w_0 = \frac{r_{xx}[1]}{r_{xx}[0] + \sigma_q^2}, \tag{8.8}$$

and the mean-square error using this coefficient is

$$\varepsilon = r_{xx}[0] - \frac{r_{xx}^2[1]}{r_{xx}[0] + \sigma_q^2}. \tag{8.9}$$

The mean-square error is the mean-square value of the difference signal $e[n]$.

For our example we will use the following correlation values

$$r_{xx}[0] = 1 + \overline{x}^2, \qquad r_{xx}[1] = \rho + \overline{x}^2, \tag{8.10}$$

where $\overline{x}$ is the signal mean. Let the noise introduced by the quantizer be a fixed fraction of the variance of the input to the quantizer. This means that the quantizer is "centred" around the mean of its input ($\overline{e}$). The variance at the input to the quantizer is $\varepsilon - \overline{e}^2$. The signal-to-noise power ratio at the output of the quantizer is then

$$\eta = \frac{\varepsilon - \overline{e}^2}{\sigma_q^2}. \tag{8.11}$$

8.2.2 Zero Mean Signals

First assume the means are zero. We will examine two sub-cases. In the first, the optimal coefficient takes into account the quantization noise. With the optimal coefficient

$$w_0 = \frac{\rho}{1 + \varepsilon/\eta}, \qquad \text{where } \varepsilon = 1 - \frac{\rho^2}{1 + \varepsilon/\eta}. \tag{8.12}$$

The error appears both on the left side of the expression and in the denominator of the right side. For a given signal-to-noise ratio η, we can solve a quadratic equation for the error ε. The mean-square error is plotted against ρ in Fig. 8.4 (solid lines) with separate curves for signal-to-noise power ratios of 2, 10, and 100. The corresponding predictor gain is $1/\varepsilon$.

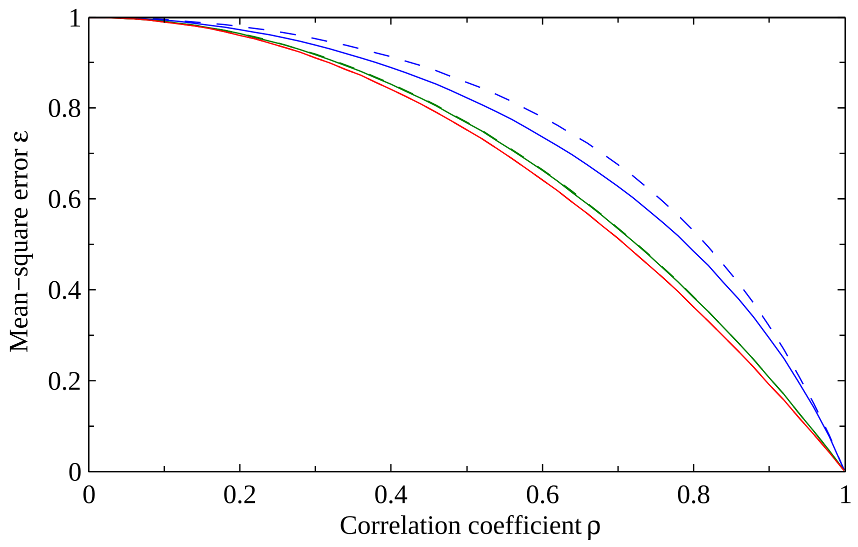


Fig. 8.4 Mean-square error as a function on the correlation coefficient. The solid lines are for the case that the predictor coefficient takes the quantization noise into account. The dashed lines are for the case that the predictor co-efficient is determined assuming no quantization noise. The lines in order from bottom to top for solid lines and separately for dashed lines are for signal-to-noise power ratios of 2, 10, and 100. The dashed lines for all except the first case are obscured by the solid lines.

For the second sub-case, the prediction coefficient will be derived based on the (noise-free) input, but applied in the predictor which has a noisy signal as input. The results are shown in Fig. 8.4 as the dashed lines. For two of the signal-to-noise power ratios, the dashed lines lie essentially on top of the solid lines. This indicates that the suboptimum predictor coefficient is not very suboptimum at moderate to high signal-to-noise ratios.

8.2.3 Non-Zero Mean Signals

Now consider an input signal with non-zero mean. As we saw earlier, we have an affine prediction system if we subtract out the mean before the input to the DPCM system. The mean can be added back in at the decoder. What is the penalty to be paid if we apply the DPCM system directly to the signal with non-zero mean? That there is a difference between the two cases is clear from the following argument. We know that the optimal (single) prediction coefficient is always less than unity (see Eq. (8.8), and use the fact that $r_{xx}[1] \leq r_{xx}[0]$). For the direct application of DPCM, the predicted signal always moves towards zero whether or not the input to the predictor is below or above the mean. For the mean-removed system, the predicted value again moves towards zero, but after the mean is added back in, it in effect moves towards the mean.

We continue using the signal statistics from the previous part of the example, but now consider a non-zero mean $\overline{x}$. Explicitly showing the mean values, the optimal predictor coefficient is

$$w_0 = \frac{\rho + \overline{x}^2}{1 + \overline{x}^2 + (\varepsilon - \overline{e}^2)/\eta}, \qquad \text{where } \varepsilon = 1 - w_0(\rho + \overline{x}). \tag{8.13}$$

The mean $\overline{e}$ is

$$\overline{e} = (1 - w_0)\overline{x}. \tag{8.14}$$

Note that these two equations give a non-linear relation for w_0. The solution for a compatible pair of w_0 and ε values is more complicated now, but can be accomplished with a simple relaxation scheme for a given pair of ρ and η values. We show the results in Fig. 8.5, where the mean has been set to be two times the variance. The vertical axis is the variance of the input to the quantizer ($\varepsilon - \overline{e}^2$). For a given mean value, the difference between the affine predictor (using zero-mean inputs) and the direct predictor decreases as the correlation coefficient approaches one. The difference increases with increasing mean value.

8.3 Example: Feedforward Data Equalizer

Consider the baseband data transmission system shown in Fig. 8.6.[1] The input to the system is a sequence of data $a[k]$. This data is transmitted using pulse amplitude modulation (PAM). The

[1]This is a baseband representation. For systems employing linear modulation, the model will use the baseband equivalent channel, with complex signals and filters.

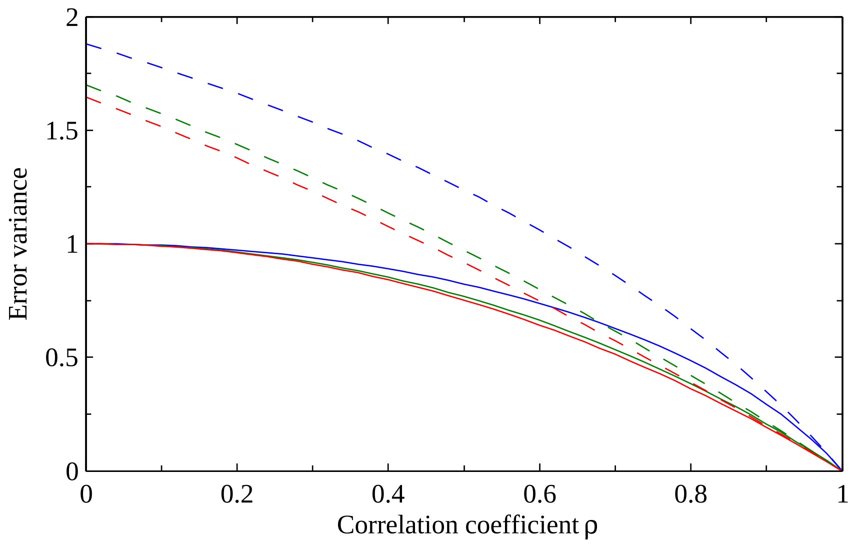


Fig. 8.5 Mean-square error for a non-zero mean signal. The solid lines are for prediction on the mean-removed signal. The dashed lines are for the predictor operating directly on the non-zero mean signal. The mean was set to twice the variance of the input signal. The lines in order from bottom to top for solid lines and separately for dashed lines are for signal-to-noise power ratios of 2, 10, and 100.

signal at the input to the channel is found as the convolution of the output of the discrete-time to continuous-time converter (see Section 3.4.2) and the transmit filter $g_T(t)$,

$$s_T(t) = \sum_{k=-\infty}^{\infty} a[k] g_T(t - kT). \tag{8.15}$$

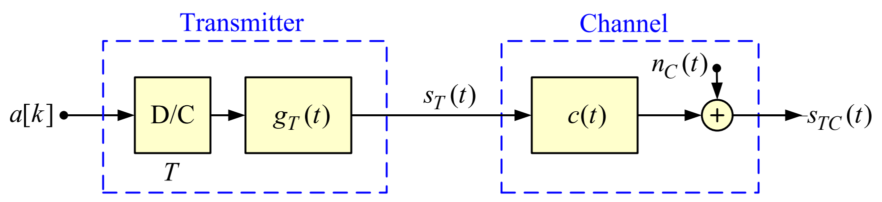


Fig. 8.6 Pulse amplitude modulated data transmitter and channel model

The output of the channel is

$$s_{TC}(t) = \sum_{k=-\infty}^{\infty} a[k] h_{TC}(t - kT) + n_C(t), \tag{8.16}$$

where $h_{TC}(t)$ is the combined response of the transmit filter and the channel filter,

$$h_{TC}(t) = g_T(t) * c(t). \tag{8.17}$$

A linear receiver for the PAM signal is shown in Fig. 8.7. The first block in the receiver is the receive filter $g_R(t)$ which contributes to the shaping of the pulses and rejects out-of-band noise. The output of the receive filter is

$$s_R(t) = \sum_{m=-\infty}^{\infty} a[m]h(t - mT) + n_R(t), \tag{8.18}$$

where $h(t)$ is the combined response of the transmit filter, channel filter, and receive filter,

$$h(t) = g_T(t) * c(t) * g_R(t), \tag{8.19}$$

and $n_R(t)$ is the noise after the receive filter,

$$n_R(t) = n_C(t) * g_R(t). \tag{8.20}$$

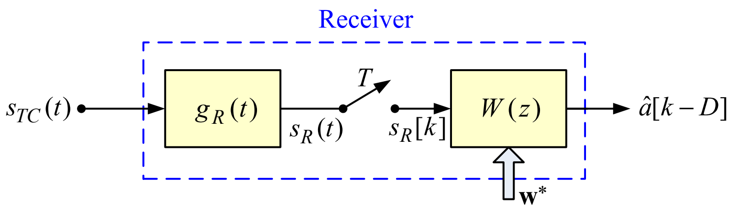


Fig. 8.7 Data receiver with feedforward equalizer

The output of the receive filter is sampled at the data interval T. The discrete-time output of the sampler is processed by an equalizer $W(z)$.[2] The equalizer considered in this example operates at the symbol rate T.

The input to the equalizer is

$$s_R[k] = s_R(kT + t_o), \tag{8.21}$$

where t_o is a sample time offset. For our subsequent analyses, we will set t_o to zero. Any sample time offset will be compensated by a time delay or advance in the channel. The output of the

[2]We refer to this equalizer as a feedforward equalizer since the equalizer does not appear in a feedback path. Later by contrast, we will consider a combined feedforward and decision feedback equalizer.

equalizer is a delayed estimate $\hat{a}[k-D]$ of the data sequence $a[k]$. This is a "soft" decision, i.e., this value has a continuous-valued amplitude. In practice, this signal would be passed on to a decision device which makes a "hard" decision by resolving the value to the nearest allowable data value. Here we find the equalizer which minimizes the mean-square error in the soft decision.

We can now make a discrete-time model of the system is shown in Fig. 8.8. Expanding the input to the equalizer,

$$\begin{aligned} s_R[k] &= \sum_{m=-\infty}^{\infty} a[m]h((k-m)T) + n_R(kT) \\ &= a[k] * h[k] + n_R[k]. \end{aligned} \tag{8.22}$$

In the last part of this equation we have simplified the notation to show discrete-time filters. The noise signal $n_R[n]$ is a discrete-time random signal with autocorrelation equal to the sampled autocorrelation of the continuous-time signal $n_C(t)$ after passing through the receive filter $g_R(t)$. Appendix E has a discussion of how to determine that correlation. The block diagram of the discrete-time system is now in the form that was shown earlier in Fig. 3.1.

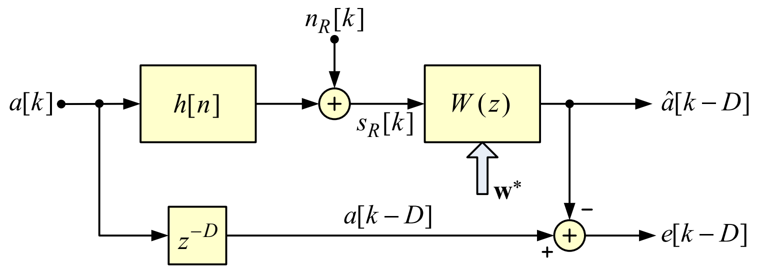


Fig. 8.8 Discrete-time model of the PAM transmitter, channel, and receiver

The figure depicting the discrete-time system also shows the desired signal $a[k-D]$. Now we are ready to find an expression for the coefficients of the minimum mean-square error equalizer. As is by now familiar, we need the autocorrelation matrix of the input signal to the equalizer $s_R[k]$, and the cross-correlation between the input signal $s_R[k]$ and the desired signal $a[k-D]$.

We assume a stationary data sequence $a[k]$ which is uncorrelated with the channel noise. The autocorrelation for the input to the equalizer is (see Appendix D)

$$\begin{aligned} r_{ss}[m] &= E\big[s_R[k]s_R^*[k-m]\big] \\ &= E\big[(a[k]*h[k])(a[k-m]*h[k-m])^*\big] + E\big[n_R[k]n_R^*[k-m]\big] \\ &= r_{aa}[m] * r_{hh}[m] + r_{n_R n_R}[m] \end{aligned} \tag{8.23}$$

where $r_{hh}[m]$ is the (deterministic) correlation for the combined response $h[n]$, $r_{aa}[m]$ is the data autocorrelation, and $r_{n_R n_R}[m]$ is the correlation of the noise signal $n_R[n]$. The cross terms between the data and noise are absent because these signals are uncorrelated. The values $r_{ss}[m]$ are used to populate the Toeplitz matrix $\mathbf{R_{ss}}$.

The cross-correlation terms needed are (see Appendix D)

$$\begin{aligned} r_{sa}[m] &= E\big[s_R[k]a^*[k-D-m]\big] \\ &= E\big[(a[k]*h[k])(a^*[k]*\delta[k-D-m])\big] + E\big[n_R[k]a^*[k-D-m]\big] \\ &= r_{aa}[m]*h[m+D]. \end{aligned} \tag{8.24}$$

The cross terms between the data and noise are absent because these signals are uncorrelated. The cross-correlation values are used to populate the cross-correlation vector $\mathbf{r_s}_a$ (see Eq. (3.7)).

8.3.1 Specializations

We will consider some standard specializations. First, the equalizer will have M equally spaced delays, starting at delay zero. Second, the data will be a zero-mean and have independent identically distributed (iid) values. Third, the noise will be an uncorrelated sequence, independent of the data. Now, the autocorrelation values are

$$r_{ss}[m] = \sigma_a^2 r_{hh}[m] + r_{n_R n_R}[m], \tag{8.25}$$

where $r_{n_R n_R}[m]$ is the correlation of the sampled noise sequence at the input to the equalizer. The autocorrelation matrix is

$$\mathbf{R_{ss}} = \begin{bmatrix} r_{ss}[0] & r_{ss}[1] & r_{ss}[2] & \cdots & r_{ss}[M-1] \\ r_{ss}[-1] & r_{ss}[0] & r_{ss}[1] & \cdots & r_{ss}[M-2] \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ r_{ss}[-(M-1)] & r_{ss}[-(M-2)] & r_{ss}[-(M-3)] & \cdots & r_{ss}[0] \end{bmatrix} \tag{8.26}$$

The cross-correlation is

$$r_{sa}[m] = \sigma_a^2 h[m+D]. \tag{8.27}$$

The delay D is a "centring" parameter used to compensate for delays in the signal path and used to centre the response in the span of the equalizer. The cross-correlation vector used to find the optimal equalizer coefficients is

$$\mathbf{r_s}_a = \sigma_a^2 \begin{bmatrix} h[D] \\ h[D-1] \\ \vdots \\ h[D-(M-1)] \end{bmatrix}. \tag{8.28}$$

8.3.2 Performance of the Feedforward Equalizer

To investigate the performance of the equalized system, we now complete the specification of the filters and the channel model.

Transmit and Receive Filters

The overall response of the transmit and receive filters will be set to have a raised cosine response,

$$G_T(f)G_R(f) = \begin{cases} T, & |f| \le \frac{1-\alpha}{2T}, \\ \frac{T}{2}\Big(1 - \sin\Big(\frac{\pi T}{\alpha}\big(f - \frac{1}{2T}\big)\Big)\Big), & \frac{1-\alpha}{2T} < |f| \le \frac{1+\alpha}{2T}, \\ 0, & \text{elsewhere.} \end{cases} \tag{8.29}$$

where the parameter α (0 to 1), sets the excess bandwidth. This response will result in zero intersymbol interference when sampled properly. When $\alpha = 0$, we have a minimum bandwidth system (a brick-wall filter) with a $\operatorname{sinc}(t/T)$ impulse response. When $\alpha = 1$, we have a full raised cosine response which is twice the bandwidth of the minimum bandwidth system. The corresponding impulse response dies off much more rapidly than the impulse response for the minimum bandwidth system. For the numerical examples, we will assume a compromise value of $\alpha = 0.25$, giving a 25% excess bandwidth. The frequency response and impulse response for the raised-cosine spectrum are shown in Fig. 8.9 and Fig. 8.10, respectively.

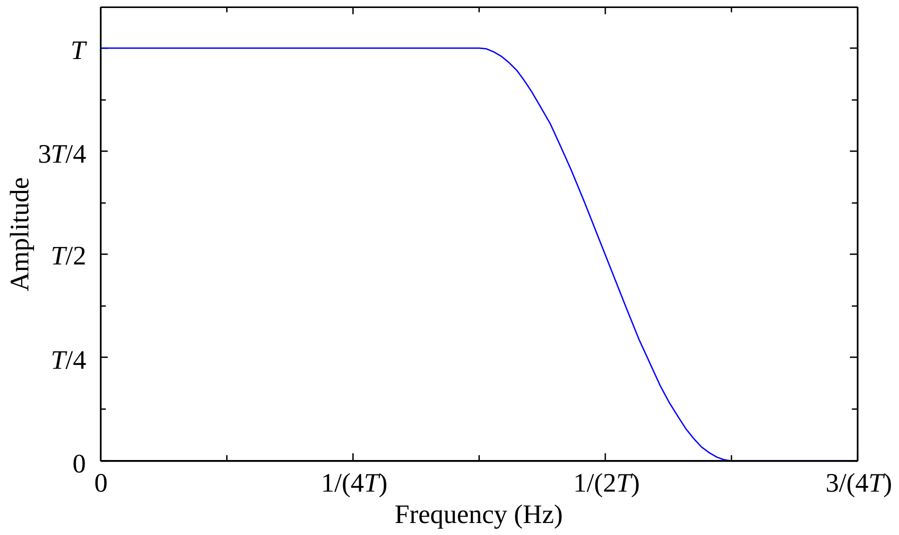


Fig. 8.9 Raised-cosine spectrum ($\alpha = 0.25$)

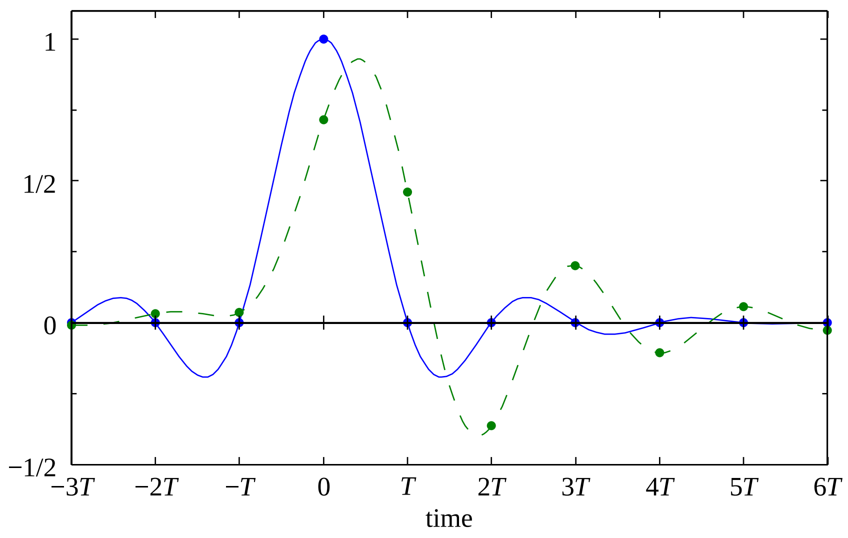


Fig. 8.10 Impulse response for a raised-cosine spectrum ($\alpha = 0.25$) shown with a solid line. The dashed line is the impulse response of a raised-cosine filter in cascade with a channel with a filter with quadratic delay ($\beta_m = 2$). The circled points are the nominal T-spaced sampling times.

The filtering will be equally apportioned between the transmitter and receiver, i.e., each filter will have a square-root raised cosine response,

$$G_T(f) = G_R(f). \tag{8.30}$$

Since we will assume that the additive noise at the channel output is white, the receiver filter is a matched filter to the transmitted pulse shape in the absence of channel distortion.

The actual filters and channel used will have a truncated impulse response. The shift to make these filters causal will introduce a delay. The sum of the delays for the transmitter and receiver filters will be used as part of the value for the delay D which appears in the desired signal. The remaining part of D will be used to centre the response in the equalizer. So if the equalizer has M coefficients, an additional delay of $(M-1)/2$ will ensue (here we assume M is odd).[3]

Signal-to-Noise Ratio

The performance can be expressed in terms of the signal-to-noise ratio (SNR). Here that ratio is the channel input power to the noise power in the Nyquist band ($|f| < 1/(2T)$). The channel input

[3]The case of M even can be handled by introducing a half sample delay in the channel response.

power is the power of the signal $s_T(t)$ at the output of the transmit filter in Fig. 8.6,

$$P_s = \int_{-\infty}^{\infty} S_{aa}(f) |G_T(f)|^2 \, df \tag{8.31}$$

where $S_{aa}(f)$ is the power spectral density for the data sequence. For the case of zero-mean iid data, $S_{aa}(f) = \sigma_a^2/T$. The integral of $|G_T(f)|^2$ for a square-root raised cosine-filter is constant, independent of α. [4] This gives

$$P_s = \frac{\sigma_a^2}{T}. \tag{8.32}$$

With a noise power spectral density of $N_o/2$ watts/Hz, the noise power in the Nyquist band is

$$P_n = \frac{N_o}{2T}. \tag{8.33}$$

Given a value for the signal-to-noise ratio P_s/P_n and a value for σ_a^2, we can easily solve for the noise power spectral density $N_o/2$. To find the noise correlation at the input to the equalizer, we follow the discrete-time modelling outlined in Appendix E. For a given noise spectral density $N_o/2$, we find that the correlation of the noise sequence after the square-root raised-cosine receive filter is given by

$$r_{n_R n_R}[k] = \frac{N_o}{2} \delta[k]. \tag{8.34}$$

Channel Filter Model

In the following, we use channel models from the classic data communications text by Lucky, Salz, and Weldon [26]. The channel is composed of three components,

$$C(f) = e^{-j2\pi\tau f T} A(f) e^{j\Phi(f)}, \tag{8.35}$$

where the first term is a delay of τ samples, the second is a pure amplitude term, and the third is pure phase term. The amplitude component has a linear decrease in log amplitude,

$$A(f) = 10^{A_{\text{dB}}(f)/20}, \qquad A_{\text{dB}}(f) = -2\alpha_m f T, \tag{8.36}$$

[4]The input and output of the transmit filter are cyclostationary. The factor $1/T$ in the expression for $S_{aa}(f)$ comes from imposing a random time offset (uniformly distributed over T) in order to get a stationary signal.

where α_m is the attenuation (in dB) at the half-sampling frequency $f = 1/(2T)$. The phase function $\Phi(f)$ is based on a quadratic group delay function

$$\Phi'(f) = 4\beta_m T(fT)^2. \tag{8.37}$$

The parameter β_m is the group delay in samples at $f = 1/(2T)$. The corresponding phase function (negative derivative of the group delay with respect to radian frequency) is cubic

$$\Phi(f) = -\frac{8\pi}{3}\beta_m(fT)^3. \tag{8.38}$$

Data receivers have means to adjust the the sampling time by an offset. In our model, this sampling time offset can be implemented by changing the channel delay τ. Rather than delaying the sampling time, the same effect is achieved by advancing the channel delay.

Overall Filter

The overall filter-channel response is the product

$$H(f) = G_T(f)C(f)G_R(f). \tag{8.39}$$

A perfect channel is given by $\alpha_m = 0$ dB, $\beta_m = 0$, and $\tau = 0$. The impulse response $h(t)$ then has the T-spaced zero crossings of the raised-cosine response. With phase distortion, the overall impulse response is skewed to the right. This shift is illustrated in Fig. 8.10 for $\beta_m = 2$. With amplitude distortion, the impulse response is distorted but remains symmetrical.

The discrete-time filter $h[n]$ was computed from the frequency response $H(f)$. This entails approximation of the integral of the inverse Fourier transform by a sum. The frequency response function was sampled with $N = 1024$ points with spacing $\Delta f = 1/(NT_s)$, where $T_s = T/N_T$. The oversampling ratio N_T was chosen to be 16. The impulse response $h[n]$ was found by subsampling the approximate inverse transform by the factor N_T. Finally, the correlation $r_{hh}[k]$ is found from $h[n]$.

Mean-Square Error

The signal-to-noise ratio was set to 20 dB and σ_a^2 was set to one. A reference case is a one-tap equalizer ($M = 1$). This single tap is just a gain value. In the absence of channel distortion, the mean-square error at the nominal sampling time is 10^{-2}. The delay D is set to align the peak of the perfect-channel impulse response with the centre of the equalizer.

Consider first the quadratic delay channel with $\beta_m = 2$. The mean-square error is plotted against sample time in Fig. 8.11. We see that the best sampling time for a one-tap equalizer is shifted from zero and that the mean-square error is significantly above the value for no channel distortion. The sampling time shift is consistent with the fact that the impulse response for the overall filter-channel combination is shifted as was shown in Fig. 8.10. When an equalizer with $M = 11$ is used, the equalizer compensates for the channel phase and at the appropriate sampling offset, the error is nearly the same as when the phase distortion is absent.

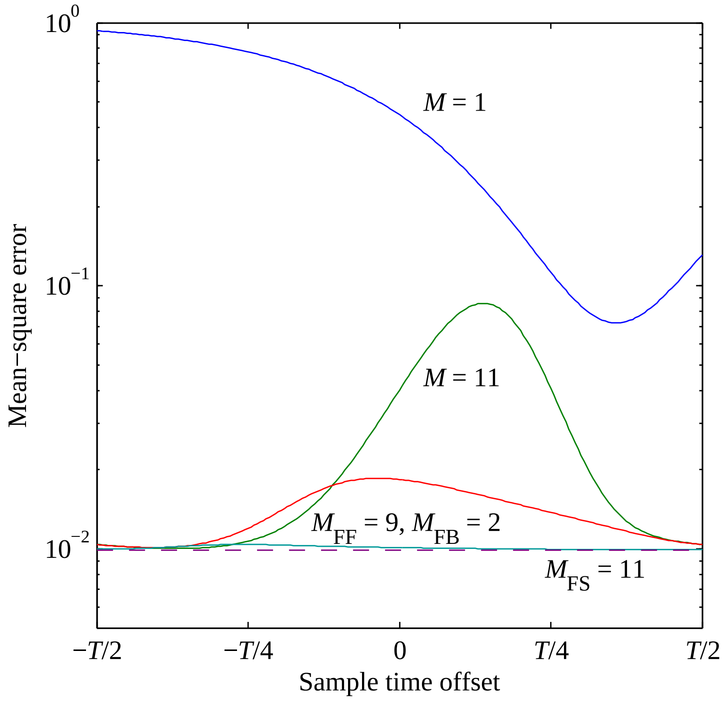


Fig. 8.11 Mean-square error versus sample time for equalization of a quadratic delay channel with $\beta_m = 2$. At the centre of the plot, the upper line is for a feedforward equalizer with 1 coefficient ($M = 1$). The next line down is for a feedforward equalizer with 11 coefficients ($M = 11$). The next line down is for a combination of a feedforward equalizer (9 coefficients) and a decision feedback equalizer (2 coefficients) ($M_{\text{FF}} = 9$, $M_{\text{FB}} = 2$). The lowest solid line is for a fractionally-spaced equalizer with 11 coefficients ($M_{\text{FS}} = 11$). The dashed line (partially obscured by the lowest solid line) shows the mean-square error for a perfect channel.

The second case has the linear attention channel with $\alpha_m = 6$ dB. The plot of mean-square error is shown in Fig. 8.12. The best sample time remains at offset zero. This is consistent with the fact that the impulse response of the filter-channel combination is symmetrical. In this case, even with $M = 11$, the resulting mean-square error stays well above that for the the perfect channel. This can be put down to a noise-enhancement phenomenon [9]. With amplitude distortion, the equalizer adjusts the gain versus frequency response to compensate for the channel attenuation. However in doing so, noise at some frequencies gets enhanced. The optimal solution for the equalizer trades off intersymbol interference (pulse shape compensation) against noise enhancement.

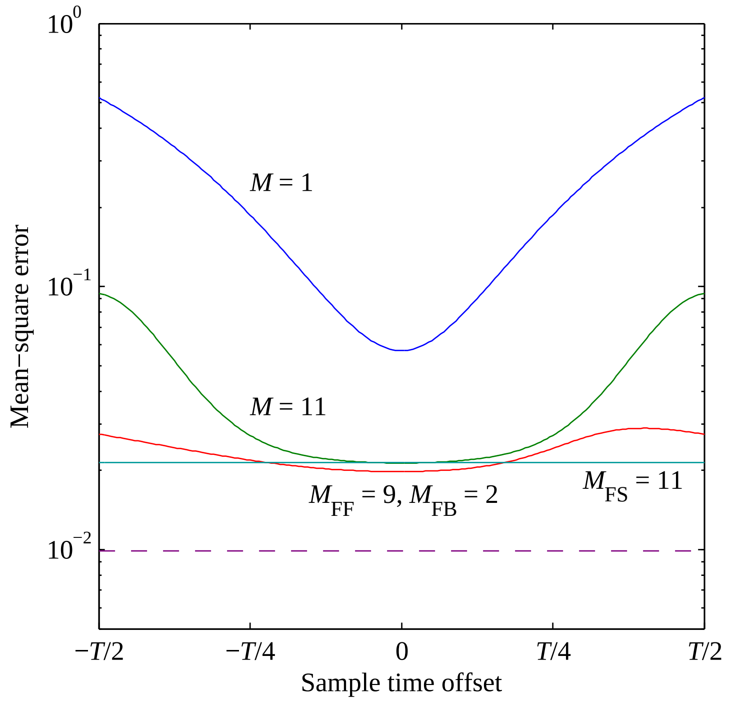


Fig. 8.12 Mean-square error versus sample time for equalization of a linear dB attenuation channel with $\alpha_m = 6$ dB. At the left edge of the plot, the upper line is for a feedforward equalizer with 1 coefficient ($M = 1$). The next line down is for a feedforward equalizer with 11 coefficients ($M = 11$). The next line down is for a combination of a feedforward equalizer (9 coefficients) and a decision feedback equalizer (2 coefficients) ($M_{\mathrm{FF}} = 9$, $M_{\mathrm{FB}} = 2$). The lowest solid line is for a fractionally-spaced equalizer with 11 coefficients ($M_{\mathrm{FS}} = 11$). The dashed line shows mean-square error for a perfect channel.

8.4 Example: Feedforward / Decision Feedback Equalizer

The feedforward equalizer can suffer from noise enhancement. Consider another configuration which uses a combination of a feedforward and decision feedback equalizers [9]. A block diagram of a receiver using this combination is shown in Fig. 8.13. This block diagram includes a new block, the decision block. This block takes the soft decision $\hat{a}[k-D]$ and creates the hard decision $\tilde{a}[k-D]$. We implicitly assume that the decision block is memoryless, i.e., it is a quantizer or slicer. If the decisions are correct then $\tilde{a}[k-D] = a[k-D]$. In the case of correct decisions, the noise is stripped off in the process of creating the hard decisions. These decisions are fed back to an equalizer $W_{\text{FB}}(z)$ in the feedback loop. The feedback equalizer has positive delays. This preserves causality and avoids a delayless loop. In this way, the decision feedback equalizer can exactly, without noise enhancement, cancel intersymbol interference that would be caused by the decoded data. The decision feedback equalizer handles pulse distortion due to the tails of the filter-channel impulse response. The feedforward equalizer shapes the pre-cursor of the filter-channel impulse response.

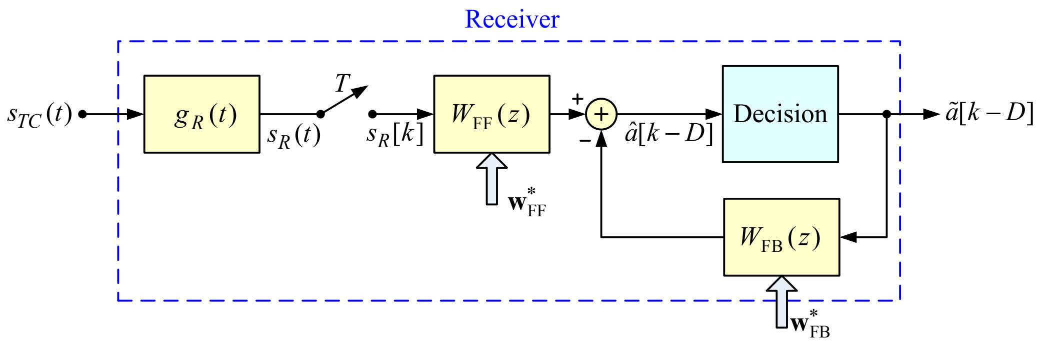


Fig. 8.13 Block diagram of a data receiver using a combination of a feedforward and decision feedback equalizer

However, the decision feedback equalizer can, in the case of incorrect decisions, adversely affect future decisions. It can be argued that if these occur infrequently and if the filter-channel impulse response tail is sufficiently small, the effect of errors can be neglected. To analyze a decision feedback equalizer, we will make the assumption that all decisions are correct. With that assumption, we can rearrange the discrete-time version of the overall system as shown in Fig. 8.14. The decision feedback equalizer has been recast as a canceller.

This system with two equalizers can be viewed as having a receiver with two input signals. The two input signal case was analyzed in Section 2.5. The formalism to handle this case is to form an augmented vector containing the inputs to the two equalizers and an augmented weight

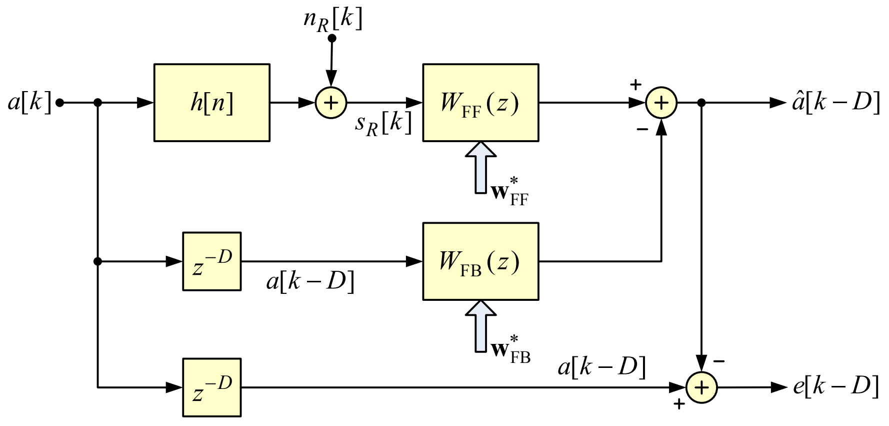


Fig. 8.14 Discrete-time model of a PAM transmission system using a combination of a feedforward and decision feedback equalizer

vector,

$$\mathbf{x}[k] = \begin{bmatrix} \mathbf{s}[k] \\ \mathbf{a}[k-D] \end{bmatrix}, \qquad \mathbf{w} = \begin{bmatrix} \mathbf{w}_{\text{FF}} \\ \mathbf{w}_{\text{FB}} \end{bmatrix}. \tag{8.40}$$

The detailed form of $\mathbf{s}[k]$ and $\mathbf{a}[k]$ will have to wait until we specify the equalizer delays. The set of equations to be solved is then

$$\begin{bmatrix} \mathbf{R_{ss}} & -\mathbf{R_{sa}} \\ -\mathbf{R_{as}} & \mathbf{R_{aa}} \end{bmatrix} \begin{bmatrix} \mathbf{w}_{\text{FF}} \\ \mathbf{w}_{\text{FB}} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{\mathbf{s}a} \\ -\mathbf{r}_{\mathbf{a}a} \end{bmatrix}. \tag{8.41}$$

The negative signs are there to account for the negation of the output of $\mathbf{w}_{\text{FB}}$ before it is summed with the output of $\mathbf{w}_{\text{FF}}$. We can equivalently consider the input to the feedback equalizer to be negated and its output not negated.

8.4.1 Specializations

We will use the same specializations (uncorrelated data, etc.) as given for the feedforward equalizer in Section 8.3.1. Let the equalizers $\mathbf{w}_{\text{FF}}$ and $\mathbf{w}_{\text{FB}}$ have M_{FF} and M_{FB} coefficients, respectively. The delays associated with $\mathbf{w}_{\text{FF}}$ are 0 through $M_{\text{FF}} - 1$. The delays associated with $\mathbf{w}_{\text{FB}}$ are 1 through M_{FB}.

The correlation matrix $\mathbf{R_{ss}}$ is an $[M_{\text{FF}} \times M_{\text{FF}}]$ matrix containing the correlations $r_{ss}[m]$ from Eq. (8.25). The cross-correlation vector $\mathbf{r}_{\mathbf{s}a}$ of the same form as Eq. (8.28).

The correlation matrix $\mathbf{R_{aa}}$ is an $[M_{\text{FB}} \times M_{\text{FB}}]$ matrix containing the correlations of $a[k-D]$. Since the data is uncorrelated,

$$\mathbf{R_{aa}} = \sigma_a^2 \mathbf{I}. \tag{8.42}$$

The cross-correlation vector $\mathbf{r}_{\text{FB}}$ is the cross-correlation between the input to the equalizer ($a[k-D]$) and the desired signal ($a[k-D]$). The elements of $\mathbf{r}_{\mathbf{a}a}$ are then $r_{aa}[m]$ for $m = -1, \ldots, -M_{\text{FB}}$ (see Eq. (3.7)). But these values are all zero, so $\mathbf{r}_{\mathbf{a}a} = \mathbf{0}$.

The matrix $\mathbf{R_{sa}}$ ($[M_{\text{FF}} \times M_{\text{FB}}]$) contains the cross-correlations between the input to $\mathbf{w}_{\text{FF}}$ and the input to $\mathbf{w}_{\text{FB}}$,

$$\mathbf{R_{sa}} = E\{\mathbf{s}[n]\mathbf{a}^H[n-D]\}, \tag{8.43}$$

where

$$\mathbf{s}[n] = \begin{bmatrix} s_R[n] \\ s_R[n-1] \\ \vdots \\ s_R[n-M_{\text{FF}}+1] \end{bmatrix}, \qquad \mathbf{a}[n] = \begin{bmatrix} a[n-1] \\ a[n-2] \\ \vdots \\ a[n-M_{\text{FB}}] \end{bmatrix}. \tag{8.44}$$

The elements of $\mathbf{R_{sa}}$ are the values of the $r_{sa}[m]$ (see Eq. (8.27)) at different delays,

$$\begin{aligned}\mathbf{R_{sa}} &= \begin{bmatrix} r_{sa}[1] & r_{sa}[2] & r_{sa}[3] & \cdots & r_{sa}[M_{\text{FB}}] \\ r_{sa}[0] & r_{sa}[1] & r_{sa}[2] & \cdots & r_{sa}[M_{\text{FB}}-1] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{sa}[2-M_{\text{FF}}] & r_{sa}[3-M_{\text{FF}}] & r_{sa}[4-M_{\text{FF}}] & \cdots & r_{sa}[M_{\text{FB}}+1-M_{\text{FF}}] \end{bmatrix} \\ &= \sigma_a^2 \begin{bmatrix} h[D+1] & h[D+2] & h[D+3] & \cdots & h[D+M_{\text{FB}}] \\ h[D] & h[D+1] & h[D+2] & \cdots & h[D+M_{\text{FB}}-1] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h[D+2-M_{\text{FF}}] & h[D+3-M_{\text{FF}}] & h[D+4-M_{\text{FF}}] & \cdots & h[D+M_{\text{FB}}+1-M_{\text{FF}}] \end{bmatrix}.\end{aligned} \tag{8.45}$$

The matrix $\mathbf{R_{as}} = \mathbf{R}_{\mathbf{sa}}^H$.

Using these values,

$$\begin{bmatrix} \mathbf{R_{ss}} & -\mathbf{R_{sa}} \\ -\mathbf{R}_{\mathbf{sa}}^H & \sigma_a^2 \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{w}_{\text{FF}} \\ \mathbf{w}_{\text{FB}} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{\mathbf{s}a} \\ \mathbf{0} \end{bmatrix}. \tag{8.46}$$

From the lower part of these equations,

$$\mathbf{w}_{\text{FB}} = \frac{1}{\sigma_a^2} \mathbf{R}_{\mathbf{sa}}^H \mathbf{w}_{\text{FF}}. \tag{8.47}$$

Denote by $g[k]$ the convolution of $h[n]$ with the feed forward equalizer (coefficients $\mathbf{w}_{\text{FF}}^*$). Then $\mathbf{w}_{\text{FB}}$ evaluates to $g^*[D+k]$ for $k = 1, \ldots, M_{\text{FB}}$.The feedback equalizer (coefficients $\mathbf{w}_{\text{FB}}^*$) acts to cancel the tail of the combined response of $h[n]$ and the feedforward equalizer.

Substituting the expression for $\mathbf{w}_{\text{FB}}$ into the upper part of the equations,

$$(\mathbf{R_{ss}} - \frac{1}{\sigma_a^2}\mathbf{R_{sa}}\mathbf{R_{sa}}^H)\mathbf{w}_{\text{FF}} = \mathbf{r_s}_a. \tag{8.48}$$

In this equation we can see that we find the feedforward equalizer coefficients from a modified autocorrelation matrix. The modified correlation matrix subtracts out the correlation components which affect the samples that the feedback equalizer removes.

8.4.2 Performance of the Feedforward / Decision Feedback Equalizer

We will use the channel framework set up for the analysis of the performance of the feedforward equalizer in Section 8.3.2. Again, the signal-to-noise ratio was set to 20 dB and σ_a^2 was set to one. The results for a combination of a feedforward equalizer and a feedback equalizer are shown along with the results of a feedforward equalizer in Figs. 8.11 and 8.12. The total number of coefficients has been kept at 11, with 9 for the feedforward equalizer and 2 for the feedback equalizer. For both the quadratic delay channel, the main effect is to make the mean-square error less dependent on the sample time offset. For the linear attenuation channel, the mean-square error is uniformly less than that for the feedforward equalizer, and the mean-square error is much less sensitive to the sample time offset.

The feedback equalizer with our assumptions of no decision errors handles tail of the response within its range perfectly without noise enhancement. In the limit of a long feedback equalizer, the effect of the whole tail is cancelled. The feedforward equalizer is then left to handle the precursors of the response. The feedforward equalizer will manipulate the precursor to minimize it. If this results in a larger tail, no harm; the feedback equalizer will take care of the tail.

8.5 Example: Fractionally-Spaced Equalizer

The equalizers considered above have tap spacing equal to the symbol rate. These equalizers can only affect the response *after* aliasing due to sampling. The optimal linear PAM receiver consists of a matched filter followed by a (infinite length) symbol rate equalizer [8]. The matched filter compensates for channel delay and optimizes the signal-to-noise ratio at the sample times. The matched filter results in a performance is independent of sampling time offset.

A fractionally-spaced equalizer acts as both a matched filter and a symbol rate equalizer. Of course, for a finite length equalizer, the performance falls below that of the ideal system.

The fractionally-spaced equalizer we consider operates after the signal has been sampled with a sampling interval of T/K. We will assume that the receive filter $G_R(f)$ is bandlimited and so K can be chosen to ensure that no aliasing occurs. The output of the fractionally-spaced equalizer is down-sampled to give a symbol-rate output. The block diagram of a PAM receiver using a fractionally spaced equalizer is shown in Fig. 8.15.

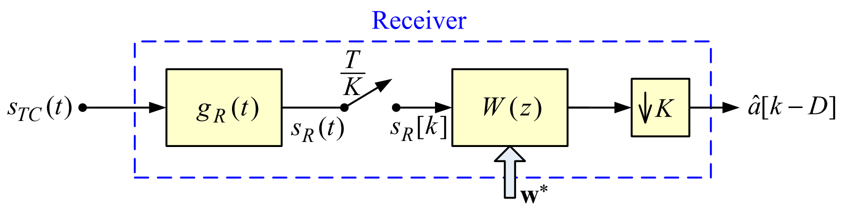


Fig. 8.15 Data receiver with a fractionally-spaced equalizer

Polyphase Representation of the Fractionally-Spaced Equalizer

We can make a discrete-time model of the overall system as shown in Fig. 8.16.

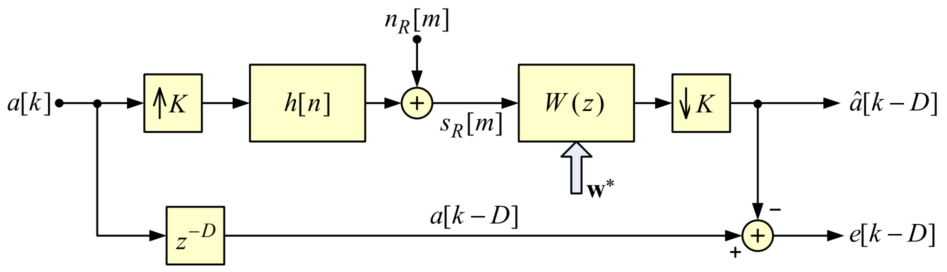


Fig. 8.16 Discrete-time model of a PAM transmission system using a fractionally-spaced equalizer

It is shown in Appendix F (Section F.7.1) that a system with an input upsampled by K, followed by filtering, and then down-sampled by K can be represented in a polyphase form using parallel blocks, each operating at the original rate. For an upsampling ratio of K, $G_T(f)$ and $G_R(f)$ must be bandlimited to $|f| < K/(2T)$. A block diagram of the system under consideration is shown in Fig. 8.17.

The impulse response sampled every T/K is denoted as $h[n]$. The polyphase components of $h_i[n]$ are $h_i[kK+i]$. The polyphase component $h_0[k]$ is identical to the symbol-rate sampled response ($h[k]$ in the preceding examples).

The filter coefficients are also represented with a polyphase decomposition. If only the zeroth branch is present, this is the symbol-rate feedforward equalizer considered in the earlier example.

Note that the system model has a unit delay in the second path. This delay corresponds to a symbol delay (T seconds). This T second delay can be viewed as the sum of a $(K-1)T/K$ delay for the channel and a T/K delay for the equalizer. In order to simplify notation, we will absorb the delay in the filter paths into the filter response $g_i[k]$. The filter $g_i[k]$ is in the same branch as

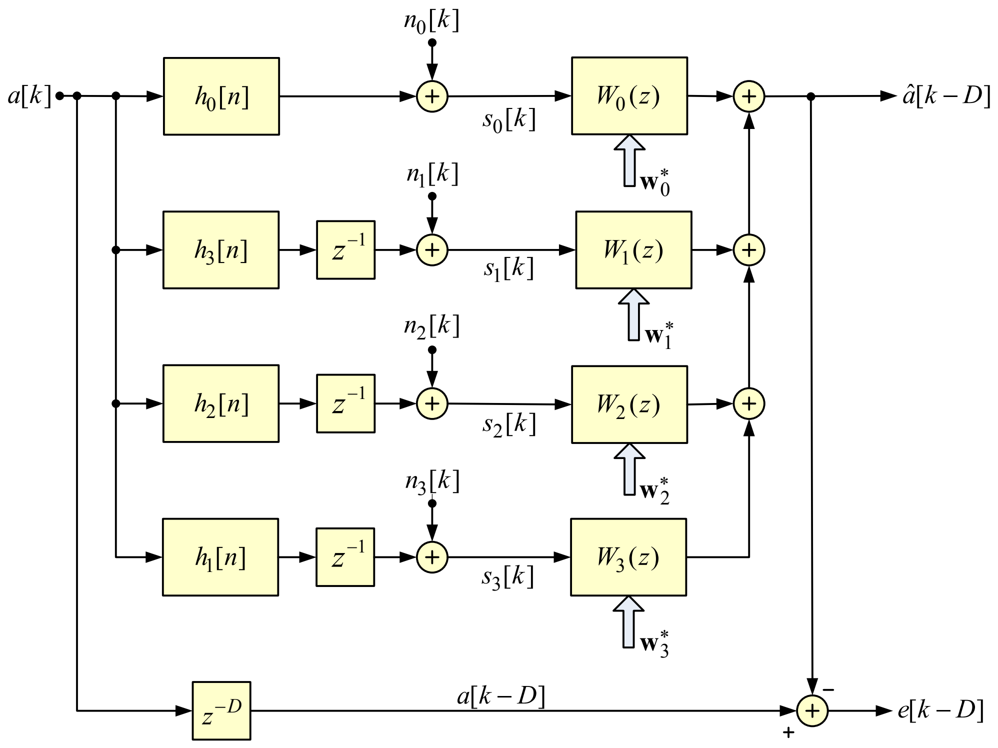


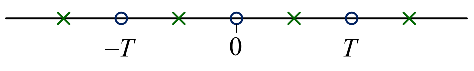
Fig. 8.17 Discrete-time polyphase model of a PAM transmission system using a fractionally-spaced equalizer (shown for $K = 4$)

$W_i(z)$. Then

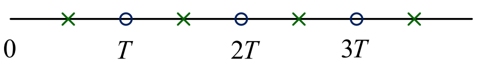
$$g_i[k] = \begin{cases} h_0[k], & \text{for } i = 0, \\ h_{K-i}[k-1], & \text{for } 0 < i < K. \end{cases} \tag{8.49}$$

Equalizer Delays

For this example we will allow for some generality in the delays associated with the fractionally-spaced equalizer. The first part of Fig. 8.18 shows the delays for a $K = 2$ fractionally-spaced equalizer with 7 coefficients. The coefficients of the fractionally-spaced equalizer are at delays of $\{-3/2, -1, -1/2, 0, 1/2, 1, 3/2\}$ (units of symbol rate T), with the reference coefficient indicated by a delay of zero. The desired signal delay D will align the reference point of the overall impulse impulse response to the reference coefficient. The second part of the figure shows a shift of the delays by a multiple of T to make the equalizer causal. The delays are now $\{1/2,\ 1,\ 3/2,\ 2,\ 5/2,\ 3,\ 7/2\}$. For the polyphase equalizer $W_0(z)$, the delays will be at delays of $\{1,\ 2,\ 3\}$. For the polyphase equalizer $W_1(z)$, taking note of the $T/2$ delay already accounted for as noted above, the delays will be at $\{0, 1, 2, 3\}$.



(a) Delays relative to the reference coefficient



(b) Causal delays

Fig. 8.18 Example of delays for a fractionally-spaced equalizer

Noise Model

We still have to specify the noise model that generates the polyphase noise term $n_i[k]$. A model is shown in Fig. 8.19. Let the noise signal after the filter $g_R(t)$ be $n_R(t)$ with correlation $r_{n_R n_R}(\tau)$. The sampled signal is $n_R[n] = n_R(nT/K)$ with correlation $r_{n_R n_R}[m] = r_{n_R n_R}(mT/K)$. Then

$$n_0[k] = n_R[kK], \qquad r_{n_0 n_0}[m] = r_{n_R n_R}[mK]. \tag{8.50}$$

Since a delay does not affect the auto-correlation of a wide-sense stationary signal,

$$r_{n_i n_i}[m] = r_{n_0 n_0}[m]. \tag{8.51}$$

The cross-correlation between $n_i[k]$ and $n_j[k]$ (see Appendix D) is

$$r_{n_i n_j}[m] = \begin{cases} r_{n_R n_R}[mK + i - j], & \text{for } i \geq j, \\ r^*_{n_R n_R}[mK + j - i], & \text{for } i < j. \end{cases} \tag{8.52}$$

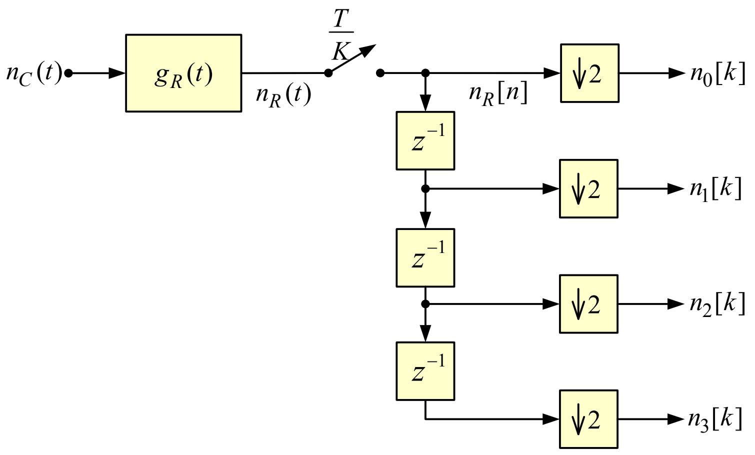


Fig. 8.19 Model for the noise that appears in the polyphase model of a fractionally-spaced equalizer (shown for $K = 4$)

Minimum Mean-Square Error Formulation

The input signal to polyphase component $W_i(z)$ is

$$s_i[k] = a[k] * g_i[k] + n_i[k]. \tag{8.53}$$

This signal is used to form the signal vectors

$$\mathbf{s}_i[n] = \begin{bmatrix} s_i[n - D_i[0]] \\ s_i[n - D_i[1]] \\ \vdots \\ s_i[n - D_i[M_i - 1]] \end{bmatrix}. \tag{8.54}$$

The delays are the delays associated with the coefficients of the polyphase equalizer components.

We are now ready to set up the equations that need to solved in order to get the coefficients of the minimum mean-square error fractional-delay equalizer,

$$\begin{bmatrix} \mathbf{R}_{0,0} & \mathbf{R}_{0,1} & \cdots & \mathbf{R}_{0,K-1} \\ \mathbf{R}_{1,0} & \mathbf{R}_{1,1} & \cdots & \mathbf{R}_{1,K-1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{R}_{K-1,0} & \mathbf{R}_{K-1,1} & \cdots & \mathbf{R}_{K-1,K-1} \end{bmatrix} \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{K-1} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \\ \vdots \\ \mathbf{r}_{K-1} \end{bmatrix}. \tag{8.55}$$

The submatrix $\mathbf{R}_{ij}$ ($[M_i \times M_j]$) is formed as follows

$$\mathbf{R}_{ij} = E\big[\mathbf{s}_i[n]\mathbf{s}_j^H[n]\big]. \tag{8.56}$$

The k, lth element of the submatrix is

$$\begin{aligned} \mathbf{R}_{ij}[k,l] &= E\big[s_i[n - D_i[k]]s_j^*[n - D_j[l]]\big] \\ &= \Big[r_{aa}[m] * r_{g_i g_j}[m] + r_{n_i n_j}[m]\Big]_{m=D_j[l]-D_i[k]}. \end{aligned} \tag{8.57}$$

The kth element of the sub-vector $\mathbf{r}_i$ is

$$\begin{aligned} \mathbf{r}_i[k] &= E\big[\mathbf{s}_i[n - D_i[k]]a^*[n - D]\big] \\ &= \left[r_{aa}[m] * g_i[D - m]\right]_{m=D_i[k]}. \end{aligned} \tag{8.58}$$

8.5.1 Specializations

The data will be zero-mean and have independent identically distributed values. The noise will be an uncorrelated sequence, independent of the data. The autocorrelation values are then

$$\mathbf{R}_{ij}[k,l] = \Big[\sigma_a^2 r_{g_i g_j}[m] + r_{n_i n_j}[m]\Big]_{m=D_j[l]-D_i[k]}. \tag{8.59}$$

The cross-correlation values are

$$\mathbf{r}_i[k] = \sigma_a^2 g_i[D - D_i[k]]. \tag{8.60}$$

8.5.2 Performance of the Fractionally-Spaced Equalizer

We reuse the channel framework developed for the earlier examples. Again, the signal-to-noise ratio was set to 20 dB and σ_a^2 was set to one. We will first consider the case $K = 2$. The fractionally-spaced equalizer will have 11 coefficients. If all of the coefficients are at a $T/2$ spacing, the equalizer will have a time span of $5T$. If all of the coefficients are at a T spacing (a conventional feed-forward equalizer), the equalizer will have a time span of $10T$. We choose a compromise: a 9 coefficient T spaced equalizer with a span of $8T$ with two additional coefficients at the half-sample spacing around the middle of the equalizer. The coefficient delays are $\{-4, -3\ -2, -1, -1/2, 0, 1/2, 1, 2, 3, 4\}$.

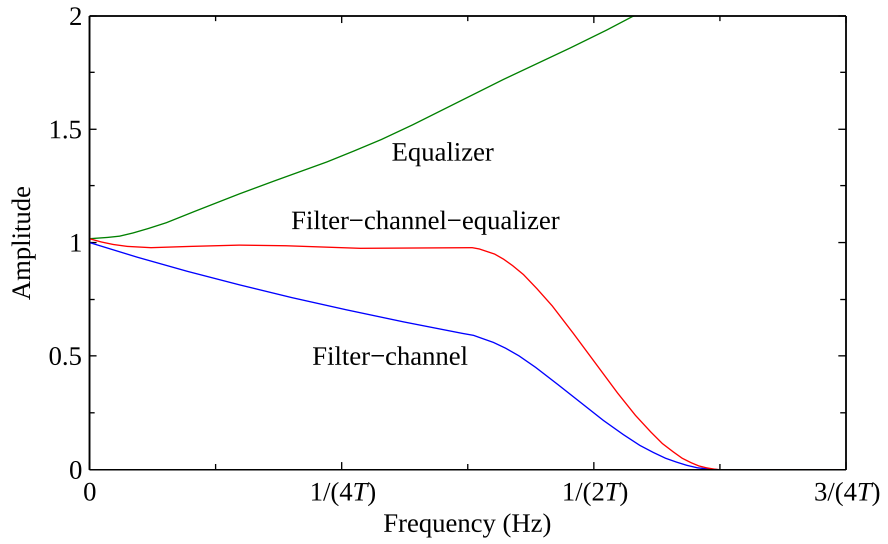
For the quadratic group delay channel, the results are shown in Fig. 8.11. We see that with just two coefficients at the half-sample spacing, the mean-square error is virtually independent of sample time offset. This indicates that the equalizer is able to synthesize a delay to compensate for the sample time offset. For the linear dB attenuation channel, the results are shown in Fig. 8.12. Here again, the results show that the mean-square error is virtually independent of the sample time offset. For this case, the combination feedforward /decision feedback equalizer marginally outperforms the fractionally-spaced equalizer over a range of sample time offsets.

To visualize the equalization performed by the fractionally-spaced equalizer for the linear dB attenuation channel, we plot the frequency characteristics of the fractionally-spaced equalizer. To do this we first form the $T/2$ fractionally-spaced equalizer by merging the polyphase components of the equalizer. In Fig. 8.20 we show the amplitude and group delay response of the equalizer determined at a sampling time offset of $3T/4$. The plot of the amplitude shows that the equalizer response rises with frequency to compensate for the sag in the response due to the linear dB attenuation of the channel. The plot of group delay shows that the group delay is relatively constant at around $4.75T$ over most of the baseband. The 4 sample delay centres the response in the equalizer span, while the $0.75T$ delay compensates for the sampling time offset.

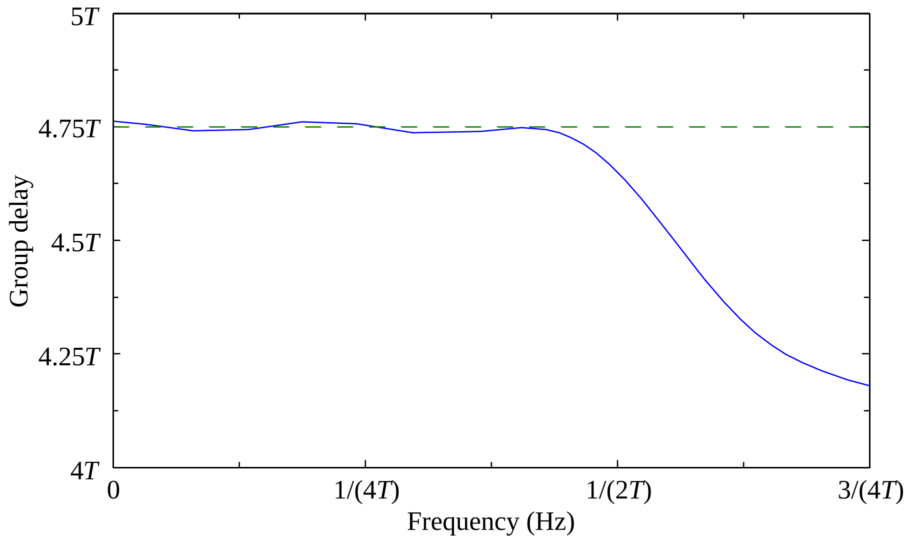
Fractionally-spaced equalizers have the advantage that the mean-square error achieved is virtually independent of sample time offset. In a data receiver, there are two aspects to clock recovery. The first is to determine the sampling rate ($1/T$) to achieve synchronism between the transmitted symbol rate and the rate at which decisions occur. The second is to determine the sample time offset. As we have seen, the mean-square error output of a fractionally-spaced equalizer does not depend strongly on this offset. This means that the data receiver can use an arbitrary offset with almost no penalty.

0.75T-Spaced Equalizer

We consider one more configuration for a fractionally-spaced equalizer. The fractionally-spaced equalizer will have a delay of $0.75T$ between coefficients [40]. This can be accomplished by sampling the received signal with a sampling interval of $0.75T$. Noting that the signal spectrum extends only to $1.25/T$, this equalizer delay spacing is adequate to avoid aliasing. The output of the equalizer is interpolated and resampled at the symbol rate to give the estimates of the data values. Our first change is sample the signal with a sampling interval of $T/4$ and then downsample by a factor of 3 to achieve the sampling interval of $0.75T$. From the results in Section F.7.2 in Appendix F and with the assumption of an appropriately bandlimited signal, downsampling by a factor of 3, filtering (equalizing) and then interpolating back to the original sampling frequency



(a) Amplitude response of the fractionally-spaced equalizer



(b) Group delay response of the fractionally-spaced equalizer

Fig. 8.20 Frequency response of a fractionally-spaced equalizer for the linear dB attenuation channel ($\alpha_m = 6$ dB) at a sampling time offset of $3T/4$. In the amplitude response plot, the curves (top to bottom) are for the equalizer, the filter-channel-equalizer combination, and filter-channel response. In the group delay response plot, the dashed line marks a constant delay of $4.75T$.

can be accomplished with a sparse filter operating at the higher sampling rate.

With the above system equivalences, the equalizer with the $0.75T$ coefficient spacing is reformulated as a $T/4$-spaced equalizer with equalizer delay values set to $\{-3.75, -3, -2.25, -1.5, -0.75, 0, 0.75, 1.5, 2.25, 3, 3.75\}$. This is a $K = 4$ equalizer, but with non-zero coefficients only appearing at delays which are multiples of $0.75T$. The formulation for the minimum-mean square solution then follows directly. The mean-square error for this configuration is not plotted as it is very close to, and essentially lies on top of, the curve for that of the $T/2$-spaced equalizer considered above.

Problems

P8.1 Mean Estimation for DPCM

This problem concerns the DPCM system considered in this chapter. We have seen that the predictor works well for zero-mean inputs. We will investigate an approach to estimating the mean and subtracting it out at the coder, and adding it back in at the receiver. In order for these two operations to track, they must estimate the mean from the reconstructed signal which is available at both the coder and decoder. That signal is $\tilde{x}[n]$ in Fig. 8.3. The gain estimator will use an exponential averaging operation,

$$\hat{x}_{\mathrm{dc}}[n] = \beta\tilde{x}[n] + (1-\beta)\hat{x}_{\mathrm{dc}}[n-1]. \tag{P8.1-1}$$

(a) Show that this estimate is unbiased. What is the time constant of the estimator (in samples) as a function of β.

(b) Find the z-transform of the filter which has as input $\tilde{X}(z)$ and output $\hat{X}_{\mathrm{dc}}(z)$. Denote this filter as $B(z)$.

(c) Draw a block diagram of the system employing this mean estimator.

(d) Show that the filtering at the decoder can be combined into one filter. Give the z-transform of that filter. Where are its poles and zeros?

P8.2 Decision Feedback Equalization

Consider an infinite extent decision feedback equalizer. This equalizer then takes care of all of the pulse tail. How does this modify the formulation for the feedforward equalizer?

Chapter 9

Examples – Cyclostationary Stochastic Signals

This chapter illustrates the solution of minimum mean-square error filtering problems for cyclostationary signals, specifically, systems with sample rate changes.

9.1 Example: Fractional Delay Filter

For this example, we will consider the system shown in Fig. 9.1. The input to the overall system is a continuous-time signal $x(t)$. In the upper path, the signal is sampled to form $x[n]$ and used as input to a filter $W(z)$. In the lower path, the continuous-time signal is delayed before being sampled and used as the "desired" signal. The goal is to minimize the mean-square error between the filter output and the desired signal which has been delayed by τ_d (not necessarily an integer multiple of the sampling interval T).

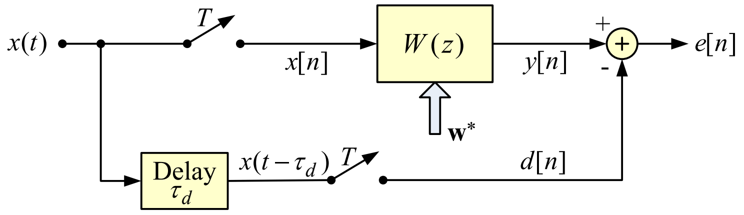


Fig. 9.1 Fractional delay filter

From the diagram of the system, we see that as far as the filter design itself, it operates on stationary continuous-time signals which when sampled are discrete-time stationary signals. Although we have included this example in a chapter on cyclostationary signals, there are no cyclostationary signals involved. However, this fractional delay filter can also be a polyphase component of a system which does involve cyclostationary components. The next example is one such system.

The correlation for a sampled wide-sense stationary signal is equal to the sampled correlation for the continuous-time signal,

$$r_{xx}[n] = r_{xx}(nT), \tag{9.1}$$

where T is the sampling interval.

We want to find the filter coefficients which minimize the mean-square error $e[n]$. Clearly, $W(z)$ should approximate the frequency response corresponding to a pure delay. We will assume for this example that the filter is causal with coefficient delays $D_k = k$, for $0 \le k < M$. The equations to be solved are

$$\mathbf{R_{xx}w} = \mathbf{r}_{\mathbf{x}d}, \tag{9.2}$$

with the elements of the autocorrelation matrix being $r_{xx}((l-k)T)$, and the kth element of the cross-correlation vector being

$$r_k = r_{xx}(\tau_d - kT), \qquad 0 \le k < M. \tag{9.3}$$

For our example we will assume that $x(t)$ has a raised-cosine power spectrum,

$$S_{xx}(f) = \begin{cases} \dfrac{1}{2f_c}, & |f| \le f_c(1-\alpha), \\ \dfrac{1}{4f_c}\Big(1 - \sin\Big(\dfrac{\pi f}{\alpha}(|f| - f_c)\Big)\Big), & f_c(1-\alpha) \le |f| \le f_c(1+\alpha), \\ 0, & f_c(1+\alpha) \le |f|. \end{cases} \tag{9.4}$$

The parameter α controls the roll-off: $\alpha = 0$ gives a brick-wall response cutting off at f_c; $\alpha = 1$ gives a full raised-cosine response which falls off smoothly to zero at $2f_c$. Figure 9.2 shows a raised-cosine frequency response with $\alpha = 1$.

The autocorrelation sequence is the inverse Fourier transform of $S_{xx}(f)$ and can be found in closed form as

$$r_{xx}(\tau) = \frac{\sin(2\pi f_c \tau)}{2\pi f_c \tau}\,\frac{\cos(2\pi\alpha f_c \tau)}{1 - (4\alpha f_c \tau)^2}. \tag{9.5}$$

We set $f_c = f_s/4$, where $f_s = 1/T$ is the sampling rate, and set $\alpha = 1$ as shown in the figure. For our example we will use a filter with $M = 11$ coefficients. We then vary τ_d from 0 to just beyond $(M-1)T$. For each value of τ_d, we find the optimal coefficients and the resulting normalized error, i.e., the error normalized by $r_{xx}(0)$. The result is plotted in Fig. 9.3. As expected, the error falls to zero ($-\infty$ dB) whenever the delay τ_d is an integer multiple of T in the range of the filter. The error has local peaks for τ_d midway between integer multiples of T. The lowest error for a given position relative to an integer multiple of T occurs when there are about the same number of coefficients on either side of the delay value. The error curve is symmetric about $\tau_d = (M-1)/2$.

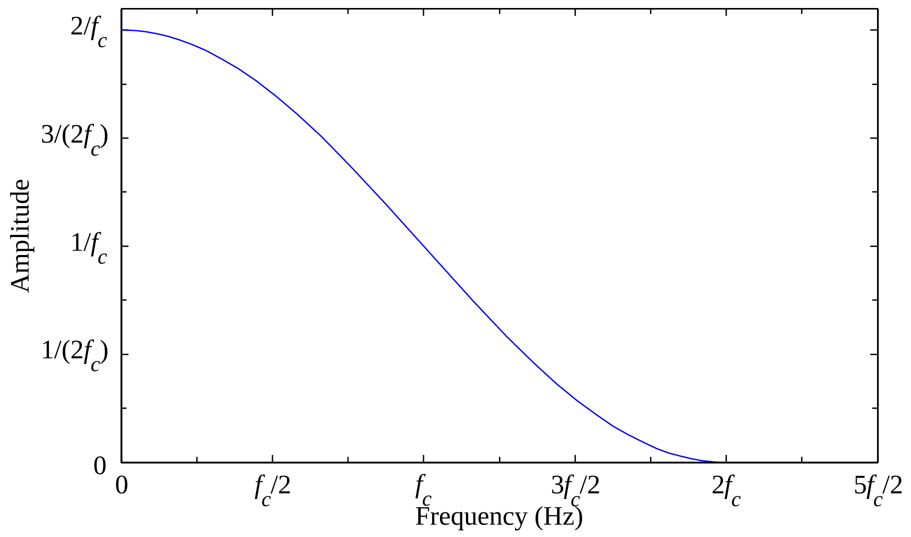


Fig. 9.2 Raised-cosine power spectrum ($\alpha = 1$)

When the delay value goes beyond the range of the filter, the error increases to near 0 dB, i.e., the filter coefficients go to near zero.

The error for the delay $\tau_d = 4.5T$ is marked with a circle on the plot. For the filter designed for that delay, the amplitude response and the group delay response are plotted in Fig. 9.4 and Fig. 9.5, respectively. The goal of a fractional delay filter is to have a flat amplitude response and a flat group delay response. For finite length filters, this is generally not possible. The sample filter approximates unity gain (0 dB) and the desired group delay (4.5 samples).

Some notes on the results.

- The approximations to constant amplitude and constant group delay are best in the regions where the power spectrum is the largest.
- Consider a flipped cross-correlation vector with elements

$$\begin{aligned} r_k^{(B)} &= r_{M-1-k}^* \\ &= r_{xx}^*(\tau_d - (M-1-k)T) \\ &= r_{xx}((M-1-k)T - \tau_d). \end{aligned} \tag{9.6}$$

Flipping the vector changes the delay, the new delay being

$$\tau_d^{(B)} = (M-1)T - \tau_d. \tag{9.7}$$

In the Wiener-Hopf equation, the matrix $\mathbf{R_{xx}}$ does not depend on τ_d. Following a develop-

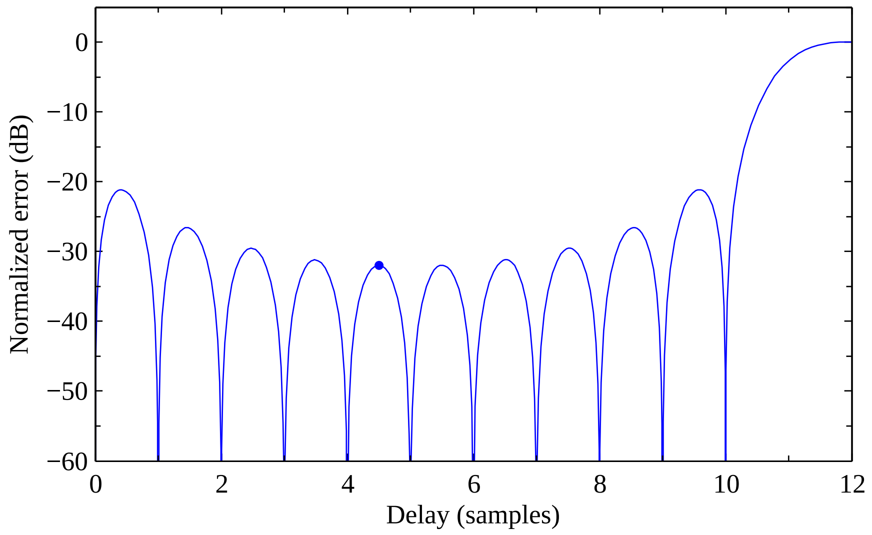


Fig. 9.3 Normalized error for a fractional delay filter ($M = 11$). The frequency response of the filter designed for the delay marked with a circle appears in later figures.

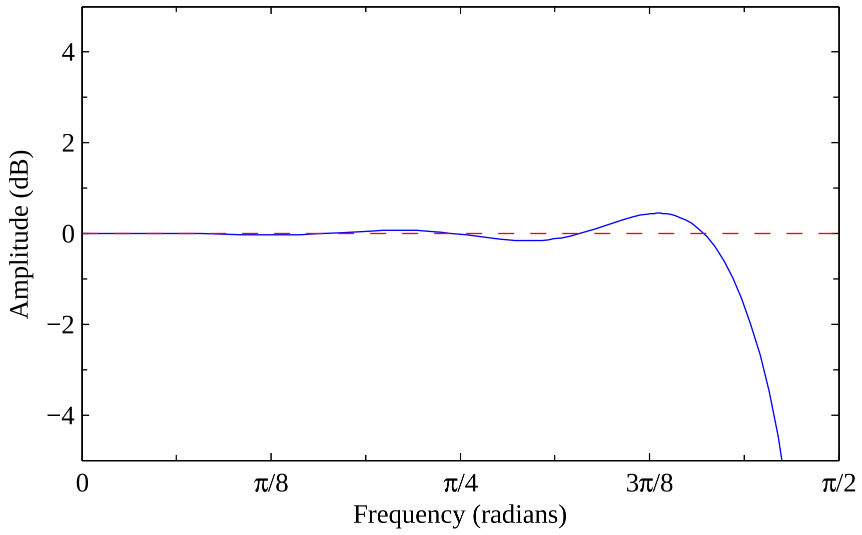


Fig. 9.4 Amplitude response for a fractional delay filter ($M = 11$, designed for $\tau_d = 4.5T$). The dashed line is for an ideal filter.

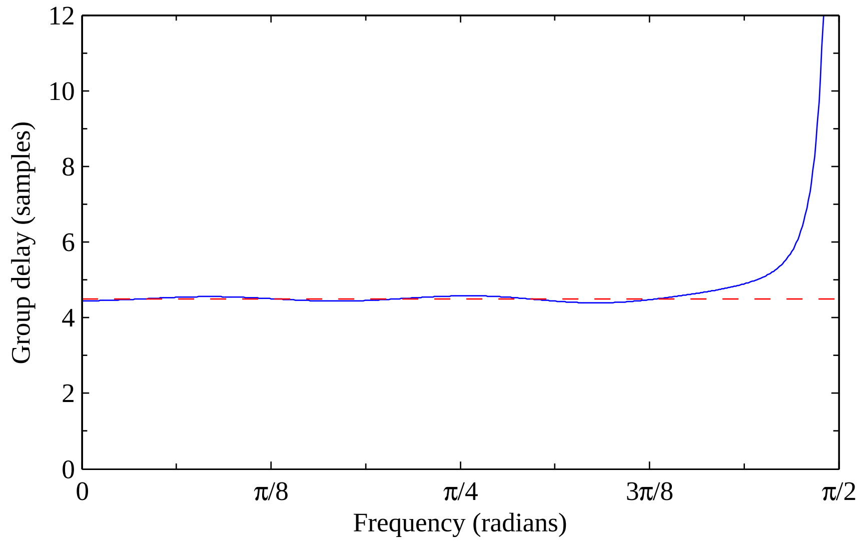


Fig. 9.5 Group delay for a fractional delay filter ($M = 11$, designed for $\tau_d = 4.5T$). The dashed line is for an ideal filter.

ment similar to that for the backward predictor (Section 4.4), we get that the coefficients for the new delay are the flipped version of the coefficients,

$$\mathbf{w}^{(B)} = \mathbf{J}\mathbf{w}^*. \tag{9.8}$$

The new filter response is

$$W^{(B)}(z) = z^{-(M-1)} W^*(1/z^*). \tag{9.9}$$

This means that the group delay functions are related as

$$\tau^{(B)}(\omega) = (M-1)T - \tau(\omega). \tag{9.10}$$

This shows that if we had designed the filter to approximate a delay of 5.5 samples, rather than 4.5 samples, the group delay curve shown in Fig. 9.5 would be a centred around 5.5 samples, but the variations around that value would be inverted. The coefficients for the 5.5 sample delay would be the flipped versions of the coefficients for the 4.5 sample delay.

- We see that if we want a fractional delay of the form $n + 1/2$, we can trade off the delay n against error. For instance, in Fig. 9.3, if we choose a delay of $\tau_d = 1.5$ samples, the error increase by about 3.3 dB.

9.2 Example: Interpolation Filter

An interpolation filter is used as part of a procedure to increase the sampling rate by a factor of I. This is a two-step process. The first step is to increase the sampling rate by inserting $I-1$ zero-valued samples between each existing sample. The second step is to filter the resulting sequence to create the intermediate samples, or equivalently suppress the repetitions of the spectrum. The model for the analysis is shown in Fig. 9.6. The actual interpolator is in the dashed box. We artificially create a subsampled signal $x_s[n]$ which serves as input signal. The signal $x[n]$ with samples at the higher sampling rate will serve as the desired signal. The filter is causal with coefficient delays $D_k = k$, for $0 \le k < M$.

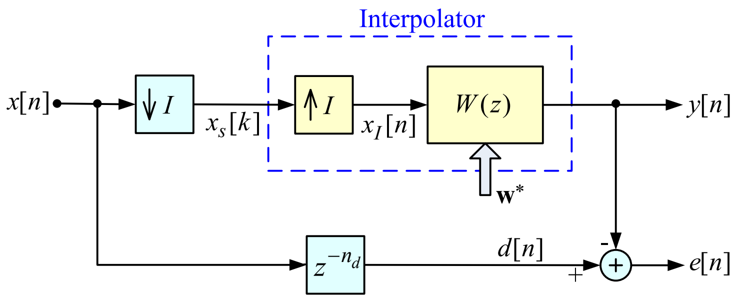


Fig. 9.6 System model for interpolation

We will solve for the filter coefficients twice: once based on the cyclostationary nature of the signals, and again using a polyphase decomposition.

9.2.1 Cyclostationary Formulation

From the discussion earlier in Section 3.2 on cyclostationary processes, it is clear that the up-sampled signal $x_I[n]$ is cyclostationary. See also Appendix G. To determine the optimal filter coefficients we need to evaluate the elements of the autocorrelation matrix,

$$r_{x_I x_I}[n-k, n-l] = \begin{cases} r_{xx}[l-k], & \text{if } ((n-k))_I = ((n-l))_I = 0 \\ 0, & \text{otherwise,} \end{cases} \tag{9.11}$$

where $((n))_I$ is n modulo I For a given n, we get an autocorrelation matrix that is sparse. For each "phase" of n, we get a different autocorrelation matrix. The elements of the cross-correlation vector are

$$\begin{aligned} r_{x_I d}[n-k,n] &= E[x_I[n-k]x^*[n-n_d]] \\ &= \begin{cases} E[x[n-k]x^*[n-n_d]], & \text{if } ((n-k))_I = 0, \\ 0, & \text{otherwise,} \end{cases} \\ &= \begin{cases} r_{xx}[n_d-k], & \text{if } ((n-k))_I = 0, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{9.12}$$

This is a sparse vector, different for each phase of $n - n_d$. The Wiener-Hopf equations are shown schematically for $I = 3$ and $M = 7$ in the following equations (corresponding to phases $p = 0, 1, 2$).

$$\begin{bmatrix} r_{xx}[0] & 0 & 0 & r_{xx}[I] & 0 & 0 & r_{xx}[2I] \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ r_{xx}[-I] & 0 & 0 & r_{xx}[0] & 0 & 0 & r_{xx}[I] \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ r_{xx}[-2I] & 0 & 0 & r_{xx}[-I] & 0 & 0 & r_{xx}[0] \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} r_{xx}[n_d] \\ 0 \\ 0 \\ r_{xx}[n_d - I] \\ 0 \\ 0 \\ r_{xx}[n_d - 2I] \end{bmatrix}, \qquad p = 0. \tag{9.13a}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & r_{xx}[0] & 0 & 0 & r_{xx}[I] & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & r_{xx}[-I] & 0 & 0 & r_{xx}[0] & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} 0 \\ r_{xx}[n_d - 1] \\ 0 \\ 0 \\ r_{xx}[n_d - I - 1] \\ 0 \\ 0 \end{bmatrix}, \qquad p = 1. \tag{9.13b}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & r_{xx}[0] & 0 & 0 & r_{xx}[I] & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & r_{xx}[-I] & 0 & 0 & r_{xx}[0] & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ r_{xx}[n_d - 2] \\ 0 \\ 0 \\ r_{xx}[n_d - I - 2] \\ 0 \end{bmatrix}, \qquad p = 2. \tag{9.13c}$$

There is a different mean-square error associated with each of these phases. This is a manifestation of the fact that the error is cyclostationary.

Noting all of the zeros in these equations, it is clear that we can form a smaller set of equations for each of the I phases. The set of equations for $p = 0$ taken from the example above is

$$\begin{bmatrix} r_{xx}[0] & r_{xx}[I] & r_{xx}[2I] \\ r_{xx}[-I] & r_{xx}[0] & r_{xx}[I] \\ r_{xx}[-2I] & r_{xx}[-I] & r_{xx}[0] \end{bmatrix} \begin{bmatrix} w_0 \\ w_3 \\ w_6 \end{bmatrix} = \begin{bmatrix} r_{xx}[n_d] \\ r_{xx}[n_d - I] \\ r_{xx}[n_d - 2I] \end{bmatrix}. \tag{9.14}$$

The sparse equations above can also be merged (added) to form a single set of equations of size $[M \times M]$. Both the smaller sets of equations and the larger merged equations can be solved using the Levinson algorithm, see Appendix J. However, it is much more efficient to solve the I smaller sets of equations than the larger merged set.[1]

We can calculate an average mean-square error, where the average is over the phases. Doing so will result in the merged set of equations. This is also equivalent to merging is make the phase p random and uniformly distributed. The random phase is equivalent to adding a random integer delay to the input signal $x[n]$ to force stationarity of $x_s[n]$.

9.2.2 Polyphase Decomposition

A good approach in most cases involving cyclostationary signals is to do a polyphase decomposition. We examine this for the current example. First consider a polyphase decompositions of the filter $W(z)$ in Fig. 9.6,

$$W(z) = \sum_{p=0}^{I-1} z^{-p} W_p(z^I). \tag{9.15}$$

The output of the filter is

$$Y(z) = \sum_{p=0}^{I-1} z^{-p} W_p(z^I) X_s(z^I). \tag{9.16}$$

The desired signal can also be written in polyphase form,

$$D(z) = \sum_{p=0}^{I-1} z^{-p} D_p(z^I). \tag{9.17}$$

Now we can write the error, $E(z) = D(z) - Y(z)$, in polyphase form,

$$E(z) = \sum_{p=0}^{I-1} z^{-p} \big[D_p(z^I) - W_p(z^I) X_s(z^I) \big]. \tag{9.18}$$

[1]Note that for the smaller sets of equations, the autocorrelation matrix (except for size) is unaffected by the phase. This can lead to some efficiencies in solving the equations as explored in Problem P9.1.

In this equation, we see that sub-filter $H_p(z)$ affects only the output for phase p. This means that we can minimize the mean-square error for each phase independently.

The error at phase p is

$$e_p[n] = d_p[n] - w_p[n] * x_s[n]. \tag{9.19}$$

The Wiener-Hopf equations for phase p is then

$$\mathbf{R}\mathbf{w}^{(p)} = \mathbf{r}_{\mathbf{x}d}^{(p)}, \tag{9.20}$$

where $\mathbf{R}$ is a Toeplitz matrix of correlations of the appropriate dimensions to match $\mathbf{w}^{(p)}$ containing the correlations $r_{xx}[kI]$, $\mathbf{w}^{(p)}$ is the subsampled coefficient vector, and $\mathbf{r}_{\mathbf{x}d}^{(p)}$ is a vector containing the correlations $r_{xx}[n_d - p - kI]$.

9.2.3 Relationship to Fractional Delay Filters

An earlier example in Section 9.1 outlined the design of a filter which implements a fractional delay. We can consider I such fractional delay filters, each associated with one of the sub-filters in the interpolation filter. The delay τ_d in Fig. 9.1 will take on values $n_d/I, (n_d+1)/I, ..., (n_d+M-1)/I$ for the different sub-filters. The outputs of these I filters will be interlaced (see Section F.6) to form the same output as the interpolation filter.

If as in the formulation for the fractional delay filter, the inputs are derived by sampling a continuous time filter, the conceptual use of fractional delay filters for the sub-filters in an interpolation filter allows for non-integer values of the delay n_d in Fig. 9.6. If the filter has an even number of coefficients, this allows the reference point in the filter to be mid-way between coefficients.

We can note from Fig. 9.3 that the error for a fractional delay filter is largest when the delay is half way between integer delays. When the delay is an integer delay, the error is zero – there is no actual interpolation, only a delay by an integer number of samples.

Subfilters of Different Lengths

Since the fractional delay filters are the sub-filters in an interpolating filter, one is not obligated to use the same length filter for each phase. Consider an overall interpolating filter for $I = 5$ with 29 coefficients. Let the delay n_d be 14 samples. Then the middle coefficient of the filter is the reference point. A standard design would then have subfilters of length 6 for each of the phases $p = 1, 2, 3, 4$. Phase $p = 0$ corresponds to an integer delay and thus only needs a single coefficient. Based on the observation that the mean-square error for a subfilter depends on the non-integer part of the delay, phases 2, and 3 have the largest mean-square error, followed by phases 1 and 4, followed by phase 0 which has zero error. This suggests that one could increase the lengths of the phase 2 and

3 subfilters by 1 each and to compensate decrease the lengths of the phase 1 and 4 subfilters. It is shown in [18], that indeed such a combination can result in a lower *average* mean-square error.

Problems

P9.1 Solving Multiple Sets of Equations

In this problem we explore possible efficiencies in solving multiple sets of Wiener-Hopf equations, where the basic structure of the correlation matrix remains unchanged. Only the right-hand side vector changes. Such systems of equations can each be solved using the Levinson algorithm. The Levinson equation is built around the Durbin recursion. As described in Appendix J, the solution vector for the Levinson algorithm is updated at each iteration of the the Durbin recursion.

The goal of this problem is to provide a Levinson-type equation that can be run after all steps of the Durbin recursion have been run. We will presume that the Durbin recursion saves intermediate results at iteration m.

Describe a post-Durbin version of the Levinson algorithm.

(a) Is there a computational penalty in running the algorithm after the Durbin recursion is run?

(b) Is there a storage penalty in running the algorithm after the Durbin recursion is run? What values need to be saved for use by the Levinson algorithm?

(c) If we have K sets of equations, each of size M, what is the saving in computations in running the Durbin recursion once, followed by running the post-Durbin version of the Levinson algorithm K times, rather than the the imbedded Durbin-Levinson algorithm K times?

Chapter 10

Examples – Least Squares

This chapter applies least-squares processing for a number of examples.

10.1 Example: Choice of Window Shape for Speech Processing

Many different types of windows have been proposed for use in frame-based signal processing. Here we will consider several standard windows and examine their performance in the task of estimating the local energy of a speech signal. In frame-based processing (see Fig. 5.1), we have a window of length N_w which advances by N_{adv} samples between frames. For this example, the window length will be fixed at 30 ms and will be applied to analyze a speech signal sampled at 8 kHz. The window is then 240 samples long.

In many applications, we want to estimate the correlation values of the windowed signal. In this example, we will concentrate on the zero lag correlation, i.e., the energy of the signal. We will need a normalization of the window values. We can take any unnormalized window and create a normalized window as

$$w_n[n] = \frac{w[n]}{\sqrt{\left(\sum_{i=-\infty}^{\infty} w^2[i]\right)}}. \tag{10.1}$$

The normalization means that if the energy of the signal is constant, the following is an unbiased estimate of that constant energy,

$$\hat{E}^2[n] = \sum_{i=-\infty}^{\infty} w_n[i]x[n-i]. \tag{10.2}$$

First consider $N_{\text{adv}} = 1$. With our 240 sample windows, the overlap between estimates is considerable. For this example, we use speech from a male utterance for testing. Fig. 10.1 shows the energy estimates for four different windows: rectangular, Hamming, von Hann, and Dolph-Chebyshev. See Fig. 5.3 for plots of these windows. For the Chebyshev window, the sidelobe attenuation was set to match that of the Hamming window (42.7 dB). Some peculiarities can be noted. The energy estimate using the rectangular window has a superimposed pseudo-periodicity in places. This is due to the fact that as the window advances even a few samples, new energetic samples are suddenly included and/or excluded. For the tapered windows, new energetic samples are gradually incorporated and so the energy track is smoother. For the Dolph-Chebyshev window, the two outlier end points of the window, cause small spikes (just noticeable on the trace) to appear on the energy track. The traces for the von Hann and Hamming window are almost indistinguishable on this plot, although the Hamming window does show slightly higher peaks and deeper valleys.[1] In a study of windows for speech processing [19], the difference between the von Hann and Hamming window shows up as rapid variations in the linear prediction coefficients. Of these windows, the von Hann window shows up as being a good candidate.

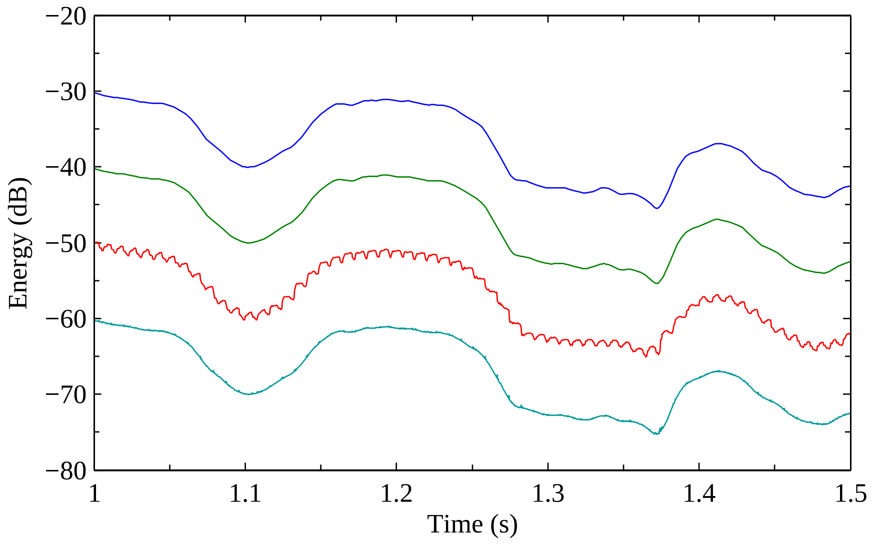


Fig. 10.1 Energy estimates for a portion of a speech signal (8 kHz sampling rate) for different windows (each 240 samples long). The windows are, from top to bottom, von Hann, Hamming, rectangular, and Dolph-Chebyshev (sidelobe attenuation 42.7 dB). For clarity, the traces have been offset from each other by 10 dB.

In practice, windows are not advanced in 1 sample steps. The energy estimates would then be

[1]The traces in the plot consist of 4001 points. In order to reduce the size of the plot file, a process to merge plot vectors was used. The criterion to determine whether adjacent plot vectors could be merged into a single plot vector was based on whether the merged vector could be visually distinguished from the original. Using this procedure, the resulting number of plot vectors for the four traces (top to bottom) was 191, 207, 1090, 1110. These numbers can be interpreted as a measure of the smoothness of the original traces – the smaller the number, the smoother the trace.

subsampled version of the traces shown. For instance, for a typical time advance of 160 samples (20 ms), there would be 50 estimates per second. This subsampling can cause aliasing if a trace is not smooth enough. The lesson from this study is that the frequency response of a time window does not tell the full story – it is important to study the time evolution of the estimates as well.

10.2 Example: Choice of Update Rate

In this example, we employ frame-by-frame prediction. The signal under an analysis window is used to calculate correlation coefficients. These coefficients are then used to calculate a set of prediction coefficients. Each frame of data is then filtered using these prediction coefficients to create a residual (error) signal. The efficacy of the prediction will be measured in terms of a prediction gain which is the ratio of the energy of the original signal to the energy of the residual signal.

For this experiment, four speech files (8 kHz sampling rate) were used. Two were spoken by females and two were spoken by males. The frame sizes varied from 32 to 200 samples. These correspond to frame update rates varying from 250 to 40 frames per second. The windows used for analysis were 240 samples long, centred on the frame. The results for a von Hann window are shown in Fig. 10.2.

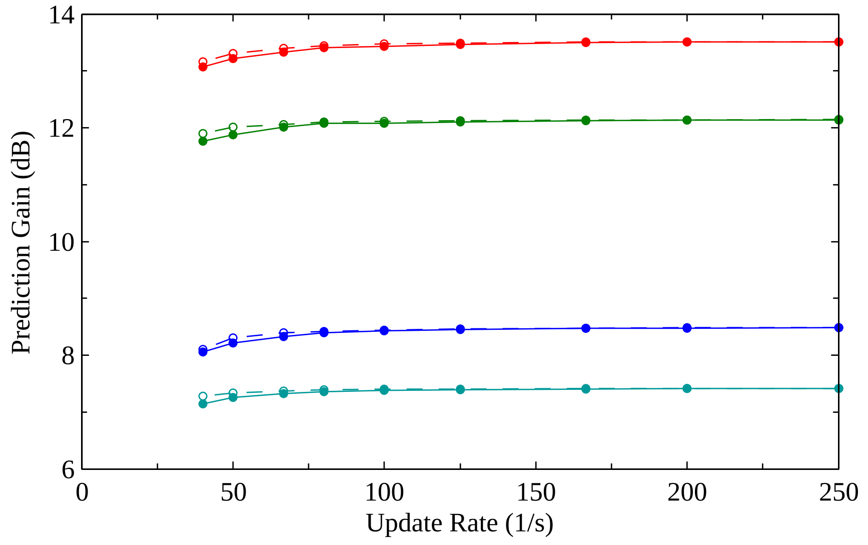


Fig. 10.2 Prediction gain (dB) versus predictor update rate. Each line represents the prediction gain for a different speech file. The solid lines represent the prediction gain when the prediction coefficients are held constant for each frame. The dashed lines represent the prediction gain for the case of using four subframes per frame. Predictor coefficients for the subframes are interpolated between the values obtained once per frame.

The prediction gains shown in the figure can be compared to the results shown in Fig. 8.2 using the same test files. In that plot, the predictor was fixed and determined from the long-term statistics of speech signals. In the present case, the predictor adapts frame-by-frame to the local statistics of the speech signal. The prediction gain is very much dependent on the test file, but the trend in prediction gain as the update rate is varied is the same for all test files.

For frame-by-frame prediction, the prediction gain saturates for frame update rates above about 100/sec. Consider a speech coding scenario. For independent coding of each vector of prediction parameters, the data rate to send predictor information from a coder to a decoder scales with the update rate. Practical speech coders often use an update rate of 50/s, which is on the knee of the curves. The plots are for a von Hann Window. The results are very similar for a Hamming window. The performance for rectangular and Dolph-Chebyshev windows falls uniformly below the values plotted, often by more than 1 dB.

10.3 Example: Interpolation of the Predictor Coefficients

We now modify the system of the previous example so as to employ subframe processing. Each frame will be divided into four subframes. A set of prediction coefficients is determined once per frame. The analysis window used to determine the prediction coefficients is centred on one of the subframes. The predictor coefficients of the intermediate subframes are determined using an interpolation procedure.

The interpolation is carried out as follows. Each vector of prediction coefficients is converted to the equivalent set of line spectral frequencies (LSFs). LSF values for the subframes are determined by linearly interpolating between LSF vectors determined once per frame. The subframe centred under the analysis window uses the coefficients determined from the LP analysis The LSF values of the next subframes are a weighted sum of those coefficients and those of the next analysis frame. As a last step, the interpolated LSF coefficients are transformed to direct form LP coefficients, resulting in one set of LP coefficients for every subframe.

One can consider interpolation of the LP parameters using one of many different transformations. Interpolation of LSF values is often used and has been shown to be a good domain in which to do the interpolation [14]. Another approach to interpolation of the LP coefficients is described in [34].

The prediction gains for a system using interpolation of the prediction coefficients are shown in Fig. 10.2. The dashed lines show the subframe interpolated results. The prediction gain is increased above that of the systems without interpolation for the slow update rates.[2]

[2]Note that the analysis frame is centred around one of the subframes. As such, the positions of the analysis frames differ whether the system uses interpolation or not and so the LP analysis generates different sets of predictor coeffi-

cients. The shift of the analysis window relative to the speech samples will in itself change the results, but averaged over a sufficient number of frames, the prediction gain serves to quantify the benefits of interpolation.

Part III

Projects

Chapter 11

Projects

This chapter suggests a number of projects that can be implemented in Matlab.

11.1 Speech Prediction

Before suggesting particular projects, consider the setup for frame by frame processing of a speech file. The steps involved in a MATLAB simulation are as follows.

- Read the input file into a vector `x` (use `wavread` for wave files).
- For frames of size N, pad the data vector with zeros so that the number of samples is a multiple of N.
- For the kth frame of size N, fill in the data matrix and the desired signal vector for the appropriate analysis method as described in Chapter 6. Use these to calculate the correlation matrix `Rxx` and the cross-correlation vector `rxd`.
- Calculate a prediction coefficient vector for each frame. This involves the solution of linear equation in the form `w = Rxx \ rxd`. The prediction error filter coefficient vector is `a = [1; -w]`.
- Note that you cannot used the MATLAB routine `filter`. This routine uses a different structure than the direct form filter that is implicitly assumed in the formulation of the optimal predictor. The difference is due to the internal structure used by `filter`. The routine shown below implements a simple (but slow) direct form FIR filter. The filtered result will be one frame of the error signal `e`. You can verify the correct operation of the filtering procedure if you use the filter `a=[1; zeros(M,1)]`. With this filter the error signal should be exactly equal to the input signal.

- The prediction gain can be calculated by taking the ratio of the sum of squares in the input signal x to the sum of squares in the error signal e.

```
function [y, FiltMem] = FIRFilt (b, x, FiltMem)
% FIR filter, direct form
% x must be a column vector
% b must be a column vector (length must not change between calls)
% FiltMem must be empty (for initialization) or a column vector returned
% by a previous call

Nmem = length(b) - 1;
if (nargin < 2 || isempty(FiltMem))
  FiltMem = zeros(Nmem, 1);
end

% Extended x vector
xe = [FiltMem; x];

Nx = length(x);
y = NaN(Nx, 1);
for (k = 1:Nx);
  y(k) =  b' * xe(Nmem+k:-1:k);
end

FiltMem = xe(end-(Nmem-1):end);

return
```

Fig. 11.1 MATLAB routine FIRFilt.

11.1.1 Project: Compare Prediction Gains

For speech sampled at 8 kHz, using frames of size $N = 160$ (20 ms), a predictor with $M = 10$ lags, compare the autocorrelation, covariance, and post-windowing approaches. For the autocorrelation method use a rectangular window of length N.

1. Calculate the overall prediction gain for the three methods.
2. Determine the number of frames which have non-minimum-phase prediction error filter. There are a number of ways to do this. You can invoke `roots` and look for roots which are larger than unity in magnitude. Alternately, you invoke the `poly2rc` which converts the direct form coefficients into reflection coefficients.

11.2 Channel Equalization

A MATLAB program for channel equalization needs model can follow the setup in Section 8.3. That section specifies data correlations, transmit and receive filters, the channel specification, and the noise.

11.2.1 Project: Equalizer "Centring"

In this project, we will examine the effect of centring on a feedforward equalizer. By "centring", we mean how many of the equalizer taps precede the reference point and how many follow the reference point. In the examples, the equalizer was centred, i.e., the reference point was in the middle of the equalizer. The delay parameter in the desired signal path can control centring in integer steps. The delay parameter in the channel model can control centring in fractional steps.

1. Set up the feedforward equalizer with the reference point in the middle of the equalizer. With this setup reproduce the curves (mean-square error vs. sampling time offset) shown in the example.

2. Extend the result to change the sampling time offset outside the limits $[-T/2, T/2]$, basically to $-MT/2, MT/2$.

3. Extend the results for distortion values other than the cases shown in the example. The results for the amplitude-only distortion behaves differently from the delay-only distortion case. Provide an explanation for the differences.

Part IV

Appendices

Appendix A

Differentiation with Respect to a Complex Vector

In the development of optimal minimum mean-square error filters, we have need to differentiate an expression for the mean-square error (a real scalar value) with respect to a set of complex coefficients. We can instead think of the optimization over a double set of coefficients – the real and imaginary parts of the coefficients. In optimizing the coefficients, we set the derivatives with respect to the real and imaginary parts of the coefficients to zero. However, sometimes we expand the real scalar value into the sum of scalar conjugate complex values and wish to take the derivatives with respect to these complex values. This appendix will develop a set of differentiation rules which give the same results as handling the real and imaginary parts separately.

A.1 Differentiation of a Complex Value with Respect to a Real Value

The case of differentiation of a complex value is easy – we write the complex value, say z, as

$$z = x + jy. \tag{A.1}$$

The derivative with respect to the real value a is

$$\frac{\partial z}{\partial a} = \frac{\partial x}{\partial a} + j\frac{\partial y}{\partial a}. \tag{A.2}$$

This derivative is well-defined and yields a complex value.

A.2 Differentiation of a Complex Value with Respect to a Complex Value

Now consider $w = a + jb$. The derivative of the complex value z with respect to the complex value w is not well-defined. We can certainly take the derivative of z with respect to each of the real values a and b. If our final goal is to set the derivatives to zero as in many optimization problems, we can set the derivatives individually to zero. For such a purpose, the two derivatives can be combined as [3]

$$\frac{\partial z}{\partial w} = \frac{1}{2}\left(\frac{\partial z}{\partial a} - j\frac{\partial z}{\partial b}\right). \tag{A.3}$$

We have included a scaling factor (1/2) and a negative sign for the second term in the parentheses. The choices have no consequence if the derivative is to be set to zero. They do however give the results that

$$\frac{\partial z}{\partial z} = 1, \qquad \frac{\partial z^*}{\partial z^*} = 1, \tag{A.4}$$

$$\frac{\partial z}{\partial z^*} = 0, \qquad \frac{\partial z^*}{\partial z} = 0. \tag{A.5}$$

Note that the derivative of z^* with respect to z (and vice-versa) is zero. This means that we can consider z^* to be a constant with respect to the the derivative with respect to z.

A.3 Differentiation of a Scalar with Respect to a Vector

The derivative of a scalar (possibly complex) with respect to a vector of complex values is a straightforward extension of the results above – we stack the individual derivatives into a vector of the same shape as **w**,

$$\frac{\partial z}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial z}{\partial w_1} \\ \frac{\partial z}{\partial w_2} \\ \vdots \\ \frac{\partial z}{\partial w_N} \end{bmatrix}. \tag{A.6}$$

A.4 Applications to Vector-Matrix Expressions

We consider several of the vector-matrix forms associated with the mean-square error. The mean-square error can often be expressed in the following form

$$\varepsilon = \sigma_d^2 - 2\mathrm{Re}[\mathbf{w}^H\mathbf{r}] + \mathbf{w}^H\mathbf{R}\mathbf{w}. \tag{A.7}$$

We can develop rules for the derivatives of each of these scalar-valued terms individually. Note that when we set the derivative to zero to find the optimal value of **w**, it matters not whether we take the derivative with respect to **w** or $\mathbf{w}^*$ [3], but of course we have to be consistent in the choice of derivative for each of the terms in the expansion.

A.4.1 Inner Product

Consider the inner product,

$$\mathbf{r}^H\mathbf{w} = \sum_k w_k r_k^*. \tag{A.8}$$

The derivative of this expression with respect to $\mathbf{w}$ is $\mathbf{r}^*$. Now expanding the real part of the inner product in Eq. (A.7),

$$\begin{aligned} a &= -2\mathrm{Re}[\mathbf{w}^H\mathbf{r}] \\ &= -2\mathrm{Re}[\mathbf{r}^H\mathbf{w}] \\ &= -(\mathbf{r}^H\mathbf{w} + \mathbf{r}^T\mathbf{w}^*). \end{aligned} \tag{A.9}$$

Then the derivative of a is

$$\begin{aligned} \frac{\partial a}{\partial \mathbf{w}} &= -(\mathbf{r}^* + 0) \\ &= -\mathbf{r}^*, \end{aligned} \tag{A.10}$$

A.4.2 Quadratic Form

Consider the quadratic form $\mathbf{w}^H\mathbf{R}\mathbf{w}$. We can express this quadratic form as the inner product of $\mathbf{w}^H\mathbf{R}$ and $\mathbf{w}$. The derivative of first term in the inner product is zero since the derivative of $\mathbf{w}^*$ with respect to $\mathbf{w}$ is zero. Thus we can treat $\mathbf{w}^H\mathbf{R}$ as a constant vector in the derivative. Now applying the inner product form as above for the derivative with respect to $\mathbf{w}$,

$$\frac{\partial \mathbf{w}^H\mathbf{R}\mathbf{w}}{\partial \mathbf{w}} = (\mathbf{w}^H\mathbf{R})^T = \mathbf{R}^T\mathbf{w}^*. \tag{A.11}$$

A.4.3 Derivative of the Mean-Square Error

The derivative of the expression for the mean-square error in Eq. (A.7) is the sum of the two terms developed above,

$$\frac{\partial \varepsilon}{\partial \mathbf{w}} = -\mathbf{r}^* + \mathbf{R}^T\mathbf{w}^*. \tag{A.12}$$

Setting this derivative to zero and taking the complex conjugate, we get

$$\mathbf{R}^H\mathbf{w} = \mathbf{r}. \tag{A.13}$$

In the case of mean-square error expression, the matrix $\mathbf{R}$ is Hermitian (required for the mean-square error to be real-valued) allowing for a further simplification of the expression.

If we had taken the derivative with respect to $\mathbf{w}^*$, the result takes on the simpler form immediately,

$$\frac{\partial \varepsilon}{\partial \mathbf{w}^*} = -\mathbf{r} + \mathbf{R}\mathbf{w}. \tag{A.14}$$

Appendix B

Wide-Sense Stationary Processes

In this appendix, we review some of the properties of wide-sense stationary processes [32] These are a partial characterization of strict-sense stationary processes.

B.1 Continuous-Time Process

Consider a continuous-time random process $x(t)$. The autocorrelation for that process is

$$r_{xx}(t,\tau) = E[x(t)x^*(t-\tau)], \tag{B.1}$$

where $E[\cdot]$ is the expectation operator. A wide-sense stationary process is one in which the first and second order statistical properties are a function only of the time delay τ. This means that the mean of $x(t)$ must be a constant, independent of t,

$$E[x(t)] = \mu. \tag{B.2}$$

Likewise, the correlation for a wide-sense stationary process does not depend of t,

$$r_{xx}(\tau) = E[x(t)x^*(t-\tau)]. \tag{B.3}$$

The autocorrelation depends only on the time difference τ. When $\tau = 0$, we get the mean-square value of $x(t)$.

B.2 Discrete-Time Process

Consider a discrete-time process created by sampling $x(t)$,

$$x[n] = x(nT). \tag{B.4}$$

Then is $x(t)$ is wide-sense stationary, the autocorrelation of $x[n]$ will also be wide-sense stationary with respect to the discrete time variable k,

$$\begin{aligned} r_{xx}[k] &= E\big[x[n]x^*[n-k]\big] \\ &= E\big[x(nT)x^*((n-k)T)\big] \\ &= r_{xx}(kT). \end{aligned} \tag{B.5}$$

This shows that the autocorrelation for a sampled signal is the sampled autocorrelation. We can, of course, have discrete-time sequences which are not created by sampling a continuous-time process. These will be wide-sense stationary if the mean of $x[n]$ is constant and the autocorrelation is only a function of the time difference k.

B.3 Properties of the Autocorrelation Function

The autocorrelation function of a wide-sense stationary process from its definition satisfies

$$r_{xx}^*[-k] = r_{xx}[k]. \tag{B.6}$$

This means the correlation sequence is Hermitian symmetric. Not any Hermitian symmetric property is a valid correlation function. As discussed below, the correlation sequence must also be non-negative definite.

B.4 Power Spectral Density

The discrete-time Fourier transform (DTFT) of a wide-sense stationary random signal $x[n]$ is not well defined since the infinite sum defining the DTFT does not converge. This is a consequence of the fact that a wide-sense stationary signal cannot die off – the energy ($r_{xx}[0]$) is constant with time. To examine the frequency domain aspects of wide-sense stationary random signals, we first calculate the auto correlation function.[1] This is now a deterministic function, which in many cases

[1]The Fourier-Stieltjes formulation [32], also referred to as the Cramér, representation [13], use an integrated Fourier transform that can be used to calculate the power spectral density directly.

of interest does have a well defined DTFT – the power spectral density,

$$S_{xx}(\omega) = \sum_{k=-\infty}^{\infty} r_{xx}[k]e^{-j\omega k}. \tag{B.7}$$

The inverse DTFT gives us back the correlation values

$$r_{xx}[k] = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{xx}(\omega)e^{j\omega k}d\omega. \tag{B.8}$$

The power spectral density has the following properties.

1. Since $S_{xx}(\omega)$ is calculated using the DTFT, the power spectral density is a periodic function of ω (period 2π).

2. Since the correlation sequence is Hermitian symmetric, the power spectral density is purely real. If $x[n]$ is real, the power spectral density will also be even.

3. The power spectral density is non-negative,

$$S_{xx}(\omega) \geq 0. \tag{B.9}$$

   This property comes from the non-negative definiteness of the correlation sequence.

B.4.1 Power Spectral Densities are Non-Negative

The non-negativity of the power spectrum can be established once we know that the power spectral density of a filtered process is given by (see Appendix D),

$$S_{yy}(\omega) = |H(\omega)|^2 S_{xx}(\omega). \tag{B.10}$$

First we note that the integral of the power spectral density gives us the energy in the signal.

$$r_{xx}[0] = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{xx}(\omega)\, d\omega. \tag{B.11}$$

Now construct a filter $H(\omega)$ having an ideal bandpass response,

$$H(\omega) = \begin{cases} 1, & \omega_l < \omega < \omega_u, \\ 0, & \text{elsewhere.} \end{cases} \tag{B.12}$$

Then we know that the power spectral density of the output is zero except in the interval $\omega_l < \omega < \omega_u$. The energy in the output signal is

$$r_{yy}[0] = \frac{1}{2\pi} \int_{\omega_l}^{\omega_u} S_{xx}(\omega)\, d\omega \geq 0. \tag{B.13}$$

The area under $S_{xx}(\omega)$ in any interval is non-negative and hence $S_{xx}(\omega) \geq 0$ everywhere.

B.5 Correlation Matrix

Consider a zero-mean wide-sense stationary discrete-time random process. Form a M term (column) vector from the samples,

$$\mathbf{x}[n] = [x[n - D_0)], x[n - D_1], \ldots, x[n - D_{M-1}]^T. \tag{B.14}$$

The correlation matrix ($[M \times M]$) is the outer product of $\mathbf{x}[n]$ with itself,

$$\mathbf{R_{xx}} = E\big[\mathbf{x}[n]\mathbf{x}^H[n]\big], \tag{B.15}$$

where the superscript H represents the Hermitian transpose. Note that the correlation matrix does not depend on n. Expanding the correlation matrix,

$$\mathbf{R_{xx}} = \begin{bmatrix} r_{xx}[0] & r_{xx}[D_1 - D_0] & r_{xx}[D_2 - D_0] & \cdots & r_{xx}[D_{M-1} - D_0] \\ r_{xx}[D_0 - D_1] & r_{xx}[0] & r_{xx}[D_2 - D_1] & \cdots & r_{xx}[D_{M-1} - D_1] \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ r_{xx}[D_0 - D_{M-1}] & r_{xx}[D_1 - D_{M-1}] & r_{xx}[D_2 - D_{M-1}] & \cdots & r_{xx}[0] \end{bmatrix}. \tag{B.16}$$

The correlation matrix for a wide-sense stationary process has a number of important properties.

1. The correlation matrix is Hermitian symmetric,

$$\mathbf{R} = \mathbf{R}^H. \tag{B.17}$$

2. The correlation matrix is non-negative definite, since for every non-zero vector $\mathbf{a}$,

$$\mathbf{a}^H\mathbf{R}\mathbf{a} \geq 0. \tag{B.18}$$

B.5.1 Non-Negative Definite Property

A valid correlation sequence must be non-negative definite. Consider the correlation matrix formed from samples with unit spacing, i.e., the delays appearing above are now $D_k = k$, for $k = 0, \ldots, M-1$,

$$\mathbf{R_{xx}} = \begin{bmatrix} r_{xx}[0] & r_{xx}[1] & r_{xx}[2] & \cdots & r_{xx}[M-1] \\ r_{xx}[-1] & r_{xx}[0] & r_{xx}[1] & \cdots & r_{xx}[M-2] \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ r_{xx}[-(M-1)] & r_{xx}[-(M-2)] & r_{xx}[-(M-3)] & \cdots & r_{xx}[0] \end{bmatrix}. \tag{B.19}$$

This correlation matrix is Toeplitz. This means that the NW-SE diagonals have constant values,

$$\mathbf{R_{xx}}[i,j] = \mathbf{r}[i-j]. \tag{B.20}$$

A Hermitian symmetric Toeplitz matrix ($[M \times M]$) can be fully specified by the M correlation values of the first row.

For the correlation sequence specified for $m = 0, \ldots, M-1$, we have the following properties, each of which is a description of the non-negative definite requirement.

1. The quadratic form using the $[M \times M]$ correlation matrix gives a non-negative result.
2. The symmetry of the correlation matrix means that the eigenvalues of $\mathbf{R}$ are real. A second interpretation of the non-negative definite property is that the the eigenvalues of $\mathbf{R}$ are non-negative.

The entire sequence is non-negative definite (and hence a valid autocorrelation sequence) if the correlation matrix formed from M values in the limit of M large is non-negative definite.

B.5.2 Eigenvalues

The eigenvector $\mathbf{v}$ corresponding to eigenvalue λ satisfies the following

$$\mathbf{R}\mathbf{v} = \lambda\mathbf{v}. \tag{B.21}$$

If we pre-multiply by $\mathbf{v}^H$, we can find the eigenvalue corresponding to an eigenvector,

$$\lambda = \frac{\mathbf{v}^H\mathbf{R}\mathbf{v}}{\mathbf{v}^H\mathbf{v}}. \tag{B.22}$$

The righthand side of this equation is the Rayleigh quotient. This equation shows that the eigenvalue is real and non-negative. In fact it can be shown that the minimum and maximum eigenvalues can be found using the Rayleigh quotient

$$\lambda_{\text{min}} = \min_{\mathbf{v}\neq\mathbf{0}} \frac{\mathbf{v}^H\mathbf{R}\mathbf{v}}{\mathbf{v}^H\mathbf{v}}, \tag{B.23a}$$

$$\lambda_{\text{max}} = \max_{\mathbf{v}\neq\mathbf{0}} \frac{\mathbf{v}^H\mathbf{R}\mathbf{v}}{\mathbf{v}^H\mathbf{v}}. \tag{B.23b}$$

Any Hermitian matrix can be expressed as

$$\mathbf{R} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^H, \tag{B.24}$$

where $\mathbf{Q}$ is a matrix of normalized orthogonal eigenvectors ($\mathbf{Q}^H\mathbf{Q} = \mathbf{I}$) and $\boldsymbol{\Lambda}$ is a diagonal matrix of the corresponding eigenvalues.

The Rayleigh quotient gives another interpretation of the eigenvectors and eigenvalues. Consider filtering the signal $x[n]$ with an FIR filter whose coefficients are given by the vector $\mathbf{v}$. The output of the filter is $y[n] = \mathbf{v}^H\mathbf{x}[n]$. The mean-square value of the output is

$$\begin{aligned} E\big[|y[n]|^2\big] &= E\big[\mathbf{v}^H\mathbf{x}[n]\mathbf{x}^H[n]\mathbf{v}\big] \\ &= \mathbf{v}^H\mathbf{R}\mathbf{v}. \end{aligned} \tag{B.25}$$

The dot product $\mathbf{v}^H\mathbf{v}$ can be thought of as the energy gain of the filter for a white noise input (set $\mathbf{R}$ to be $\sigma^2\mathbf{I}$). Then

$$\frac{E\big[|y[n]|^2\big]}{\mathbf{v}^H\mathbf{v}} = \frac{\mathbf{v}^H\mathbf{R}\mathbf{v}}{\mathbf{v}^H\mathbf{v}}. \tag{B.26}$$

We see that the normalized output energy can be maximized by choosing the filter $\mathbf{v}$ to be the eigenvector of $\mathbf{R}$ corresponding to the largest eigenvalue, and the normalized output energy is the largest eigenvalue. Ditto for the smallest eigenvalue.

The eigenvalues of $\mathbf{R}$ are bounded by the minimum and maximum values of the power spectral density. This can be seen by again considering filtering $x[n]$ with a filter with coefficients $\mathbf{v}$. Then we can write the energy of the output in terms of the power spectrum of the output,

$$\begin{aligned} E\big[|y[n]|^2\big] &= \frac{1}{2\pi}\int_{-\pi}^{\pi} S_{yy}(\omega)\,d\omega \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} |V(\omega)|^2 S_{xx}(\omega)\,d\omega \\ &\leq S_{\text{max}}\frac{1}{2\pi}\int_{-\pi}^{\pi} |V(\omega)|^2\,d\omega \\ &= S_{\text{max}}\mathbf{v}^H\mathbf{v}. \end{aligned} \tag{B.27}$$

The normalized error is then bounded by the maximum of the power spectrum. Since we know that the normalized error is no larger than λ_{max} (compare Eq. (B.26) and Eq. (B.23b)), then $\lambda_{\text{max}} \leq S_{\text{max}}$. Using the analogous result for S_{min},

$$S_{\text{min}} \leq \lambda_i \leq S_{\text{max}}. \tag{B.28}$$

In the limit of large M, the largest eigenvalue approaches the maximum value of the power spectral density, and the smallest eigenvalue approaches the minimum value of the power spectral density. This argument reinforces the relationship between the non-negativity of the eigenvalues and the non-negativity of the power spectral density.

B.6 Cross-Correlation

The correlation between jointly stationary signals is given by

$$r_{xy}[k] = E\big[x[n]x^*[n-k]\big]. \tag{B.29}$$

The cross-power spectral density is defined as

$$S_{xy}(\omega) = \sum_{k=-\infty}^{\infty} r_{xy}[k]e^{-j\omega k}. \tag{B.30}$$

The cross-correlation matrix for vectors containing samples from $x[n]$ and $y[n]$ is

$$\mathbf{R}_{yx} = E\big[\mathbf{x}[n]\mathbf{y}^H[n]\big]. \tag{B.31}$$

Problems

PB.1 Properties of Correlation Matrices

A correlation matrix is formed as

$$\mathbf{R} = E\big[\mathbf{x}[n]\mathbf{x}^H[n]\big]. \tag{PB.1-1}$$

(a) Show that the correlation matrix is non-negative definite.

(b) Show that the eigenvalues of a non-negative definite Hermitian symmetric matrix are real and positive.

Appendix C

Hermitian, Persymmetric, and Toeplitz Matrices

C.1 Symmetric Matrices

A symmetric matrix $\mathbf{A}$ is one that is symmetric about its NW-SE diagonal with elements which satisfy the following relationship

$$a_{ij} = a_{ji}, \qquad 0 \leq i, j < N. \tag{C.1}$$

In matrix form,

$$\mathbf{A}^T = \mathbf{A}. \tag{C.2}$$

A schematic form of a $[3 \times 3]$ symmetric matrix is shown below

$$\begin{bmatrix} a & d & e \\ d & b & f \\ e & f & c \end{bmatrix}.$$

The inverse of a symmetric matrix is also symmetric

$$\mathbf{A}^{-1} = (\mathbf{A}^{-1})^T. \tag{C.3}$$

C.2 Hermitian Symmetric Matrices

A Hermitian symmetric matrix $\mathbf{A}$ is one that is conjugate symmetric about its NW-SE diagonal with elements which satisfy the following relationship

$$a_{ij} = a_{ji}^*, \qquad 0 \leq i, j < N. \tag{C.4}$$

In matrix form,

$$\mathbf{A}^T = \mathbf{A}^*, \qquad \mathbf{A}^H = \mathbf{A}. \tag{C.5}$$

A schematic form of a $[3 \times 3]$ Hermitian symmetric matrix is shown below

$$\begin{bmatrix} A & d & e \\ d^* & B & f \\ e^* & f^* & C \end{bmatrix}.$$

The elements on the diagonal (A, B, and C) are purely real.

The inverse of a non-singular Hermitian symmetric matrix is also Hermitian symmetric,

$$\mathbf{A}^{-1} = (\mathbf{A}^{-1})^H. \tag{C.6}$$

C.3 Exchange Matrix

It is useful to define a matrix which can swap columns or rows of a matrix. The exchange matrix $\mathbf{J}$ defined as

$$\mathbf{J} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \iddots & \vdots & \vdots \\ 0 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix}. \tag{C.7}$$

If $\mathbf{J}$ is used to premultiply a matrix $\mathbf{A}$, the columns of $\mathbf{A}$ are flipped. If $\mathbf{J}$ is used to postmultiply a matrix $\mathbf{A}$, the rows of $\mathbf{A}$ are flipped. Since $\mathbf{JJ} = \mathbf{I}$, then the inverse of the exchange matrix is $\mathbf{J}^{-1} = \mathbf{J}$. Also the exchange matrix is both symmetric and Hermitian symmetric, $\mathbf{J} = \mathbf{J}^H$.

C.4 Persymmetric Matrices

A persymmetric matrix is one which is symmetric about its SW-NE diagonal. A persymmetric matrix has elements which satisfy

$$a_{ij} = a_{N-j-1,N-i-1}, \qquad 0 \le i,j < N. \tag{C.8}$$

Notice that there is no conjugate in this definition. A schematic form of a $[3 \times 3]$ persymmetric matrix is shown below

$$\begin{bmatrix} e & d & a \\ f & b & d \\ c & f & e \end{bmatrix}.$$

Applying the **J** operator to flip rows and columns of a persymmetric matrix, we get the following result.

Persymmetric Matrix:

A persymmetric matrix satisfies

$$\mathbf{JAJ} = \mathbf{A}^T, \text{ or equivalently, } \mathbf{AJ} = \mathbf{JA}^T. \tag{C.9}$$

We will show that the inverse of persymmetric matrix is also persymmetric (a result used in Appendices I and J). First, for *any* non-singular matrix **B**,

$$(\mathbf{B}^{-1})^T = (\mathbf{B}^T)^{-1}. \tag{C.10}$$

For a persymmetric matrix **A**, using Eq. (C.10) and Eq. (C.9),

$$\begin{aligned}(\mathbf{A}^{-1})^T &= (\mathbf{A}^T)^{-1} \\ &= (\mathbf{JAJ})^{-1} \\ &= \mathbf{J}^{-1}\mathbf{A}^{-1}\mathbf{J}^{-1} \\ &= \mathbf{JA}^{-1}\mathbf{J}.\end{aligned} \tag{C.11}$$

We see that $\mathbf{A}^{-1}$ satisfies the symmetry requirements for a persymmetric matrix (Eq. (C.9) with $\mathbf{A}^{-1}$ replacing **A**.

Inverse of a Persymmetric Matrix:

For a persymmetric matrix **A**, the inverse matrix is also persymmetric and satisfies

$$\mathbf{JA}^{-1}\mathbf{J} = (\mathbf{A}^{-1})^T. \tag{C.12}$$

C.5 Toeplitz Matrices

A Toeplitz matrix **T** has elements which satisfy

$$t_{ij} = t_{i-1,j-1}, \qquad 1 \le i,j < N. \tag{C.13}$$

A Toeplitz matrix has equal elements down each diagonal. These diagonals are characterized by $i - j$ being a constant. A Toeplitz matrix is fully specified by giving the first column and first row. If the matrix is also Hermitian symmetric, then either the first column or the the first row is sufficient to specify the full matrix.

A schematic form of a $[3 \times 3]$ Toeplitz matrix is shown below

$$\begin{bmatrix} a & b & c \\ d & a & b \\ e & d & a \end{bmatrix}.$$

Since it has constant values down each diagonal, a Toeplitz matrix is persymmetric. This means that the inverse of Toeplitz matrix is also persymmetric,

$$\mathbf{J}\mathbf{T}^{-1}\mathbf{J} = (\mathbf{T}^{-1})^T. \tag{C.14}$$

However, the inverse of a Toeplitz matrix is not in general Toeplitz.

C.6 Hankel Matrices

A Hankel matrix is the analogue to the Toeplitz matrix but for the NE-SW diagonals. For a Hankel matrix $\mathbf{A}$, $\mathbf{JA}$ is a Toeplitz matrix and $\mathbf{AJ}$ is also a (different) Toeplitz matrix.

A schematic form of a $[3 \times 3]$ Hankel matrix is shown below

$$\begin{bmatrix} c & b & a \\ b & a & d \\ a & d & e \end{bmatrix}.$$

Problems

PC.1 Inverse of a Hankel Matrix

We have shown that the inverse of a Toeplitz matrix is persymmetric. What is the equivalent property of the inverse of a Hankel matrix?

PC.2 Triangular Matrix

Consider a lower triangular matrix $\mathbf{L}$.

(a) Show that the inverse of a lower triangular matrix $\mathbf{L}$ is also lower triangular.

(b) Show that $\mathbf{L}^H$ is upper triangular.

(c) Show that $(\mathbf{L}^H)^{-1}$ is upper triangular.

(d) Show that $(\mathbf{L}^H)^{-1} = (\mathbf{L}^{-1})^H$.

Appendix D

Filtered Random Processes

For some of our examples we need to have at hand results on the correlation properties of filtered random processes. Consider the system in Fig. D.1. The two filters $h_u[n]$ and $h_v[n]$ have a common wide-sense stationary input $x[n]$. The cross-correlation between the two output signals $u[n]$ and $v[n]$ can be written as

$$\begin{aligned} r_{uv}[n, n-k] &= E\big[u[n]v^*[n-k]\big] \\ &= E\Big[\sum_{p=-\infty}^{\infty} h_u[p]x[n-p] \sum_{q=-\infty}^{\infty} h_v^*[q]x^*[n-k-q]\Big] \\ &= \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} h_u[p]h_v^*[q]E\big[x[n-p]x^*[n-k-q]\big] \\ &= \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} h_u[p]h_v^*[q]r_{xx}[k-(p-q)]. \end{aligned} \tag{D.1}$$

We can see that $r_{uv}[n, n-k]$ is a function only of the difference k and so is wide-sense stationary. This gives us the result that correlation functions for filtered wide-sense stationary processes are also wide-sense stationary.

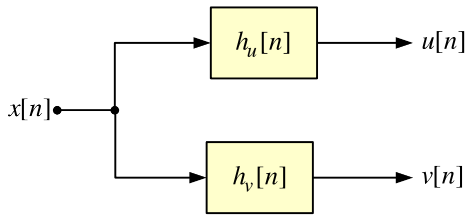


Fig. D.1 Filtered random process

Continuing the development, replace $p-q$ by s and set $q = p - s$,

$$\begin{aligned} r_{uv}[k] &= \sum_{s=-\infty}^{\infty} \Big(\sum_{p=-\infty}^{\infty} h_u[p] h_v^*[p-s] \Big) r_{xx}[k-s] \\ &= \Big(h_u[k] * h_v^*[-k] \Big) * r_{xx}[k]. \end{aligned} \tag{D.2}$$

The term in brackets is a (deterministic) cross-correlation,

$$r_{h_u h_v}[k] = \sum_{m=-\infty}^{\infty} h_u[m] h_v^*[m-k]. \tag{D.3}$$

Using the notation of Chapter 5, we could write this deterministic correlation as $\overline{h_u[m]h_v^*[m-k]}$.

D.1 Special Cases

Some special cases ensue. Consider the case that one of the filters is a no-op filter, e.g., $h_v[n] = \delta[n]$ as shown in the first part of Fig. D.2. Then $v[n] = x[n]$. The cross-correlation between $u[n]$ and $v[n]$ is then really the cross-correlation between $u[n]$ and $x[n]$. Similarly, we can make the other filter a no-op as shown in the second part of the figure. The results for these cases are

$$\begin{aligned} r_{ux}[k] &= h_u[k] * r_{xx}[k], \\ r_{xv}[k] &= h_v^*[-k] * r_{xx}[k]. \end{aligned} \tag{D.4}$$

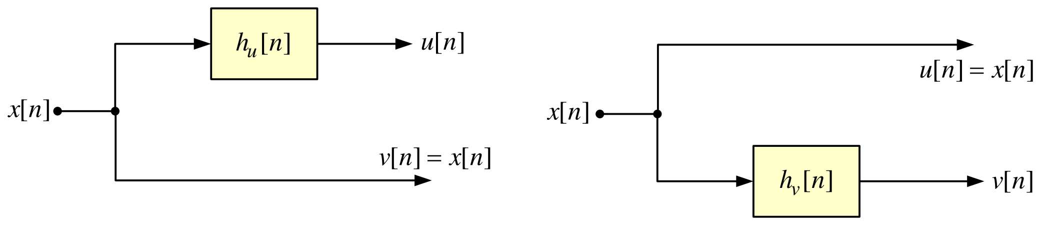


Fig. D.2 Filtered random process, special cases: $v[n] = x[n]$ and $u[n] = x[n]$

Another special case is when the two filters are the same, $h_v[n] = h_u[n] = h[n]$, see Fig. D.3. Then $v[n] = u[n]$. The correlation is then

$$r_{uu}[k] = r_{hh}[k] * r_{xx}[k]. \tag{D.5}$$

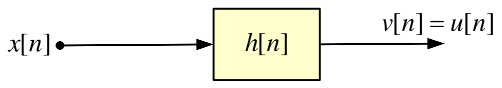


Fig. D.3 Filtered random process, special case: $v[n] = u[n]$

D.2 Frequency Domain Expressions

The cross-power spectral density and the power spectral density are Fourier transforms of the cross-correlation and autocorrelation, respectively. In the frequency domain, the cross-power spectral density corresponding to the result in Eq. (D.2) is

$$S_{uv}(\omega) = H_u(\omega) H_v^*(\omega) S_{xx}(\omega), \tag{D.6}$$

where $S_{xx}(\omega)$ is the power spectral density corresponding to $r_{xx}[k]$. For the special case of a no-op filter as in Eq. (D.4), the cross-power spectral densities of the outputs are

$$\begin{aligned} S_{ux}(\omega) &= H_u(\omega) S_{xx}(\omega), \\ S_{xu}(\omega) &= H_u^*(\omega) S_{xx}(\omega). \end{aligned} \tag{D.7}$$

For the case of a single filter autocorrelation as in Eq. (D.5), the power spectral density of the output is

$$S_{uu}(\omega) = |H(\omega)|^2 S_{xx}(\omega). \tag{D.8}$$

Appendix E

Sampled Random Processes

For some of our examples we need to have at hand results on the correlation properties of sampled random processes. The properties will be used to build models using discrete-time signals. The system under consideration has a random signal as input to a filter as shown in Fig. E.1.

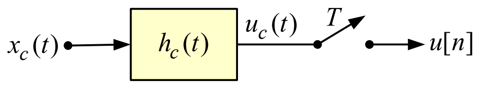


Fig. E.1 Sampled random process

If the signal $x_c(t)$ is wide-sense stationary with autocorrelation $r_{x_c x_c}(\tau)$, the autocorrelation for $u_c(t)$ is given by the convolution (compare with the results for discrete-time signals in Appendix D)

$$r_{u_c u_c}(\tau) = r_{x_c x_c}(\tau) * r_{h_c h_c}(\tau), \tag{E.1}$$

where $r_{h_c h_c}(\tau)$ is the deterministic autocorrelation of the filter impulse response,

$$r_{h_c h_c}(\tau) = \int_{-\infty}^{\infty} h_c(s) h_c^*(s-\tau)\, ds. \tag{E.2}$$

In the frequency domain, the power spectral density relationship is

$$S_{u_c u_c}(f) = |H_c(f)|^2 S_{x_c x_c}(f). \tag{E.3}$$

The discrete-time signal $u[n]$ is formed by sampling $u_c(t)$,

$$u[n] = u_c(nT). \tag{E.4}$$

Consider the autocorrelation of $u[n]$,

$$\begin{aligned} r_{uu}[k] &= E\big[u[n]u^*[n-k]\big] \\ &= E[u_c(nT)u_c^*((n-k)T)] \\ &= r_{u_cu_c}(kT). \end{aligned} \tag{E.5}$$

Autocorrelation for a Sampled Stationary Signal

For a wide-sense stationary continuous-time signal $u_c(t)$, the autocorrelation of the sampled signal $u[n] = u_c(nT)$ is found by sampling the autocorrelation of the continuous-time signal.

$$r_{uu}[k] = r_{u_cu_c}(kT) \tag{E.6}$$

Denote the power spectral density of the discrete-time signal as $S_{uu}(\omega)$ and the power spectral density of the continuous-time signal as $S_{u_cu_c}(f)$. Sampling the correlation results in the aliased response

$$S_{uu}(\omega) = \frac{1}{T}\sum_{k=-\infty}^{\infty} S_{u_cu_c}(\frac{\omega - 2\pi k}{2\pi T}), \tag{E.7}$$

where we have expressed the normalized radian frequency as $\omega = 2\pi f T$.

E.1 Bandlimited Continuous-Time Filter

If $h_c(t)$ is bandlimited, we can sample $u_c(t)$ at an appropriate rate without encountering aliasing. Assume $h_c(t)$ has a bandwidth of f_{max}. Consider the system shown in Fig. E.2. This figure shows $x_c(t)$ applied to an ideal lowpass filter $h_L(t)$ with frequency response

$$H_L(f) = \begin{cases} 1, & \text{if } |f| < f_L, \\ 0, & \text{elsewhere.} \end{cases} \tag{E.8}$$

The impulse response of the lowpass filter is

$$h_L(t) = 2f_L \operatorname{sinc}(2f_L t), \tag{E.9}$$

where $\operatorname{sinc}(x) = \sin(\pi x)/(\pi x)$. The cutoff frequency f_L will be chosen to be greater than or equal to f_{max}. This lowpass filter is redundant in the system of Fig. E.1 since the filter $h_c(t)$ is bandlimited.

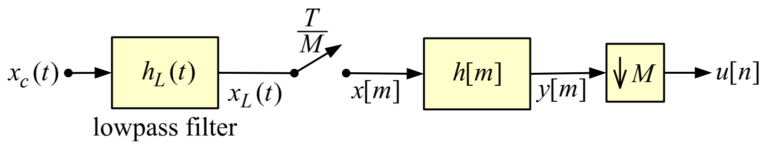


Fig. E.2 Sampled filtered random process

The signal at the output of the ideal lowpass filter is $x_L(t)$, with correlation,

$$\begin{aligned} r_{x_L x_L}(\tau) &= r_{x_c x_c}(\tau) * r_{h_L h_L}(\tau) \\ &= r_{x_c x_c}(\tau) * h_L(\tau). \end{aligned} \tag{E.10}$$

The last relationship is a manifestation of the fact that for the ideal lowpass filter $|H_L(f)|^2 = H_L(f)$ and thus $r_{x_L x_L}(\tau) = h_L(\tau)$. The relationship between power spectral densities is

$$S_{x_L x_L}(f) = \begin{cases} S_{x_c x_c}(f), & \text{for } |f| < f_L, \\ 0, & \text{elsewhere.} \end{cases} \tag{E.11}$$

The lowpass signal $x_L(t)$ is sampled to give $x[m]$. We choose the sampling frequency to an integer multiple of the sampling frequency of the final output signal $u[n]$. The sampling frequency is then M/T. We also set the cutoff of the lowpass filter to be half of the sampling frequency, $f_L = M/(2T)$. The oversampling factor M is chosen so that $M/T \geq 2f_{\max}$. For simplicity of notation, we will denote the sampling interval as

$$T_s = \frac{T}{M}. \tag{E.12}$$

Thus the bandlimited signal $x_L(t)$ is sampled every T_s to form $x[m] = x_c(mT_s)$.

The discrete-time signal $x[m]$ has correlation

$$r_{xx}[k] = r_{x_L x_L}(kT_s). \tag{E.13}$$

The power spectrum of $x[m]$ is

$$S_{xx}(\omega) = \frac{1}{T_s} \sum_{k=-\infty}^{\infty} S_{x_L x_L}\Big(\frac{\omega - 2\pi k}{2\pi T_s}\Big). \tag{E.14}$$

The lowpass filter assures that the shifted version of $S_{x_L x_L}(f)$ do not overlap.

The discrete-time signal $x[m]$ is applied to the discrete-time filter with impulse response $h[m] = T_s\, h_c(mT_s)$. The continuous-time filter $h_c(t)$ has correlation $r_{h_c h_c}(\tau)$. The discrete-time filter $h[m]$

has correlation $r_{hh}[k]$. These two correlations can be related as shown in the following.

The frequency response of the discrete-time filter is

$$H(\omega) = \sum_{k=-\infty}^{\infty} H_c\Big(\frac{\omega - 2\pi k}{2\pi T_s}\Big). \tag{E.15}$$

For the bandlimited case, there is no overlap between the shifted versions of $H_c(f)$. This means that the squared spectrum has the form

$$|H(\omega)|^2 = \sum_{k=-\infty}^{\infty} \Big|H_c\Big(\frac{\omega - 2\pi k}{2\pi T_s}\Big)\Big|^2. \tag{E.16}$$

The inverse transform of the left side of this expression is the correlation $r_{hh}[k]$. The inverse transform of the right side of this expression is T_s times the sampled correlation $r_{h_c h_c}(kT_s)$.

Autocorrelation for a Sampled Filter Response

Let the continuous-time filter with impulse response $h_c(t)$ be bandlimited to less than $1/(2T_s)$. Then the sampled filter response $h[m] = T_s\, h_c(mT_s)$ has a correlation which is related to the correlation of the continuous-time filter as

$$r_{hh}[k] = T_s r_{h_c h_c}(kT_s). \tag{E.17}$$

The correlation of the signal at the output of the filter is

$$r_{yy}[k] = r_{xx}[k] * r_{hh}[k], \tag{E.18}$$

where $r_{hh}[k]$ is filter impulse response correlation. The power spectral density relationship is

$$S_{yy}(\omega) = |H(\omega)|^2 S_{xx}(\omega). \tag{E.19}$$

The last step is to down-sample the output of the filter by a factor M to form $u[n]$. With this configuration, $u[n] = u_c(nT)$, where $u_c(t)$ appears just before the sampler in Fig. E.1. The correlation for $u[n]$ is

$$r_{uu}[k] = r_{yy}[kM]. \tag{E.20}$$

The power spectral density of $u[n]$ is

$$S_{uu}(\omega) = \frac{1}{M} \sum_{k=0}^{M-1} S_{yy}\Big(\frac{\omega - 2\pi k}{M}\Big). \tag{E.21}$$

The discrete-time equivalent model appears in the righthand side of Fig. E.2. The discrete-time signal $x[m]$ with correlation given by $r_{x_L x_L}(kT/M)$ is input to a discrete-time filter $h[m]$, with its output subsampled to form $u[n]$.

E.2 White Noise

Let the input signal $x_c(t)$ be zero-mean white noise. This signal can be characterized by its power spectral density,

$$S_{x_c x_c}(f) = \frac{N_o}{2}, \qquad \text{W/Hz}. \tag{E.22}$$

The correlation for $x_c(t)$ is,

$$r_{x_c x_c}(\tau) = \frac{N_o}{2}\delta(\tau). \tag{E.23}$$

This result reiterates the result that the energy of $x_c(t)$ ($r_{x_c x_c}(0)$) is infinite. The power spectral density of $u_c(t)$ which appears in Fig. E.1 is

$$S_{u_c u_c}(f) = \frac{N_o}{2}|H_c(f)|^2, \tag{E.24}$$

and the corresponding autocorrelation is

$$r_{u_c u_c}(\tau) = \frac{N_o}{2} r_{h_c h_c}(\tau). \tag{E.25}$$

The correlation for the sampled signal $u[n]$ is

$$r_{uu}[k] = \frac{N_o}{2} r_{h_c h_c}(kT), \tag{E.26}$$

and the corresponding power spectral density relationship is

$$S_{uu}(\omega) = \frac{N_o}{2T}\sum_{k=-\infty}^{\infty}|H_c(\frac{\omega - 2\pi k}{2\pi T})|^2. \tag{E.27}$$

A simplified form applies if the aliased squared filter response is a constant, viz.,

$$\frac{1}{T}\sum_{k=-\infty}^{\infty}|H_c(f - \frac{k}{T})|^2 = b. \tag{E.28}$$

The discrete-time noise $u[n]$ is then white with variance

$$\sigma_u^2 = b\frac{N_o}{2}. \tag{E.29}$$

If the filter $H_c(f)$ is bandlimited, we can apply the model shown in Fig. E.2. For white noise, the signal $x[m]$ is itself white, viz.,

$$\begin{aligned} r_{xx}[k] &= r_{x_L x_L}(\tau) * h_L(\tau)\Big|_{\tau=\frac{kM}{T}} \\ &= \frac{N_o}{2}\frac{M}{T}\delta[k]. \end{aligned} \tag{E.30}$$

In the frequency domain,

$$S_{xx}(\omega) = \frac{N_o}{2}\frac{M}{T}. \tag{E.31}$$

The correlation for $y[m]$ is

$$r_{yy}[k] = \frac{N_o}{2}\frac{M}{T}r_{hh}[k]. \tag{E.32}$$

The power spectral density for $y[m]$ is

$$\begin{aligned} S_{yy}(\omega) &= |H(\omega)|^2 S_{xx}(\omega) \\ &= \frac{N_o}{2}\frac{M}{T}|H(\omega)|^2. \end{aligned} \tag{E.33}$$

The down-sampled output signal $u[n]$ has correlation

$$r_{uu}[k] = \frac{N_o}{2}\frac{M}{T}r_{hh}[kM]. \tag{E.34}$$

The power spectral density of $u[n]$ is

$$\begin{aligned} S_{uu}(\omega) &= \frac{1}{M}\sum_{k=0}^{M-1} S_{yy}(\frac{\omega - 2\pi k}{M}) \\ &= \frac{N_o}{2T}\sum_{k=0}^{M-1} |H(\frac{\omega - 2\pi k}{M})|^2. \end{aligned} \tag{E.35}$$

Appendix F

Multirate Signals

In this appendix we review the structures that implement filtering of multirate signals. The basic theory is discussed in, for instance, [35].

F.1 Downsampling

Consider the operation of downsampling. In this procedure, we keep every Mth sample of the original sequence.

$$y[n] = v[nM]. \tag{F.1}$$

In z-transform notation [35],

$$Y(z) = \frac{1}{M} \sum_{p=0}^{M-1} V(W_M^p z^{1/M}), \tag{F.2}$$

where W_M^k is defined as

$$W_M^k = e^{-j2\pi k/M}. \tag{F.3}$$

A common scenario is where the signal is filtered prior to downsampling as in Fig. F.1 The purpose of this filtering is to remove frequency components which might cause aliasing. The output signal for this case is

$$Y(z) = \frac{1}{M} \sum_{p=0}^{M-1} G(W_M^p z^{1/M}) X(W_M^p z^{1/M}). \tag{F.4}$$

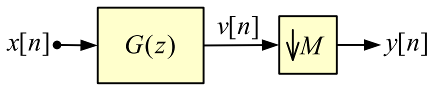


Fig. F.1 Downsampling with anti-alias filtering

F.2 Upsampling

In upsampling, the output signal consists of the original signal with $M-1$ zeros interposed between each sample,

$$v[n] = \begin{cases} x[n/M], & ((n))_M = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{F.5}$$

In z-transform notation,

$$V(z) = X(z^M). \tag{F.6}$$

Commonly an upsampling system includes a filter to "interpolate" between samples as shown in Fig. F.2. The output signal in this case is

$$Y(z) = H(z)X(z^M). \tag{F.7}$$

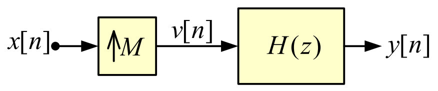


Fig. F.2 Upsampling with interpolation filtering

F.3 The Noble Identities

We note that in downsampling, a filter normally precedes the downsampler (as in Fig. F.1) and in upsampling, a filter normally follows the upsampler (as in Fig. F.2). These filters *cannot* generally be moved across the downsampler or upsampler. There are cases in the following polyphase configurations in which the filter is on the wrong side of an upsampler or downsampler. The so-called noble identities [35] can be used to move the filter to the other side of the upsampler or downsampler.

Figure F.3 shows the noble identity for downsampling. Note that in the upper figure, the filter is on the *wrong* side of the downsampler. Moving the filter across the downsampler causes the frequency response to repeat. The filter $G(z^M)$ will not prevent aliasing.

Figure F.4 shows the noble identity for upsampling. Note that in the upper figure, the filter is on the *wrong* side of the upsampler. Moving the filter across the upsampler causes the frequency response to repeat. The filter $H(z^M)$ will not interpolate between samples.

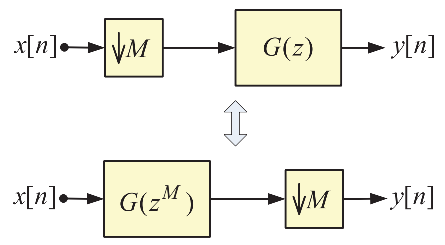


Fig. F.3 Noble identity for downsampling

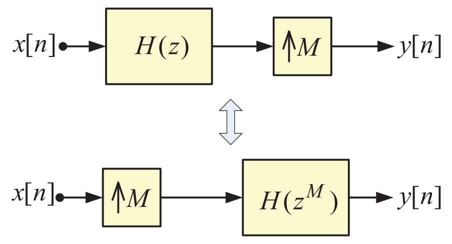


Fig. F.4 Noble identity for upsampling

F.4 Polyphase Decomposition

Any filter $H(z)$ can be decomposed into M polyphase components,

$$\begin{aligned} H(z) &= \sum_{n=-\infty}^{\infty} h[n]z^{-n} \\ &= \sum_{p=0}^{M-1} z^{-p} \sum_{m=-\infty}^{\infty} h[mM+p]z^{-mM} \\ &= \sum_{p=0}^{M-1} z^{-p} H_p(z^M), \end{aligned} \tag{F.8}$$

where the polyphase components of the filter are

$$H_p(z) = \sum_{m=-\infty}^{\infty} h[mM+p]z^{-m}, \qquad 0 \le p < M. \tag{F.9}$$

Two versions of the polyphase decomposition for a filter are shown in Fig. F.5.

Here we have applied a polyphase decomposition to a filter. The same form of polyphase decomposition can also be applied to a signal $X(z)$,

$$X(z) = \sum_{p=0}^{M-1} z^{-p} X_p(z^M). \tag{F.10}$$

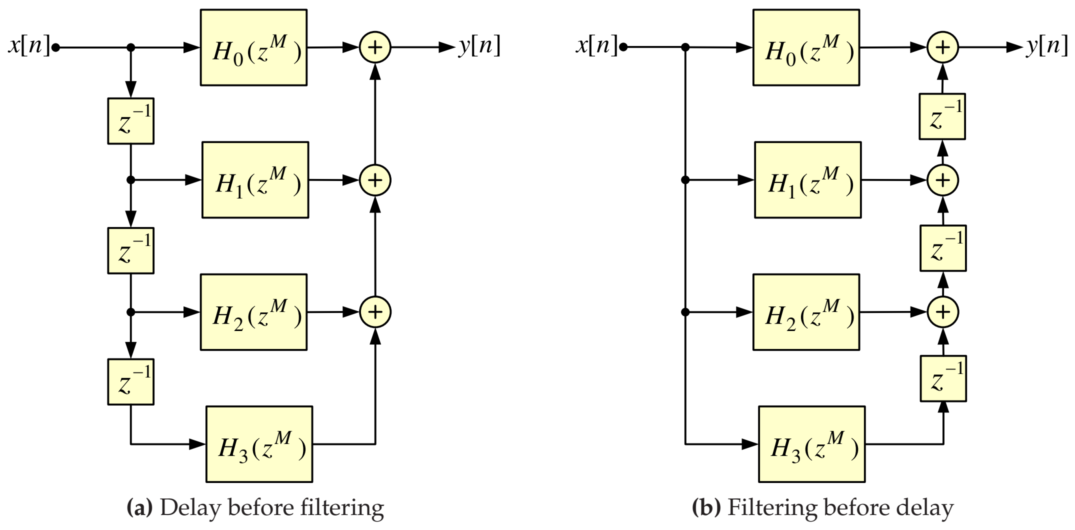


Fig. F.5 Two versions of a polyphase decomposition of a filter, shown for $M = 4$

We will not use the polyphase form of signals in this appendix. Section 9.2.2 uses a polyphase decomposition of both a filter and a signal in order to design an interpolation filter.

F.5 Polyphase Downsampling

The polyphase decomposition in Fig. F.5(a), can be applied to the filter in the downsampling system shown in Fig. F.1. The resulting system is shown in Fig. F.6. In this figure, the downsampling operation has been moved into each branch, and then the order of filtering and downsampling has been interchanged with the appropriate modification to the filter (see Section F.3). One can note that the delays and downsampling operations before the polyphase filter components act as a commutator, directing input samples into the different branches in a round-robin fashion. The sample at time 0 is directed to the upper branch; the next sample is directed to the bottom branch; the next sample is directed to the second to bottom branch; and so on.

F.6 Polyphase Interpolation

The polyphase decomposition in Fig. F.5(b) can be applied to the filter in the interpolation system shown in Fig. F.2. The resulting system is shown in Fig. F.7. In this figure, the upsampling operation has been moved into each branch, and then the order of upsampling and filtering has been interchanged with the appropriate modification to the filter (see Section F.3). One can note that the delays and downsampling operations after the polyphase filter components act as a commutator, interleaving samples from the different branches into the output.

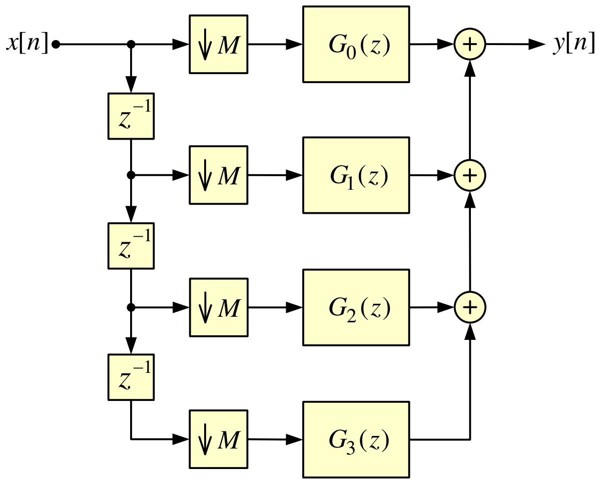


Fig. F.6 Downsampling with a polyphase decomposition, shown for $M = 4$

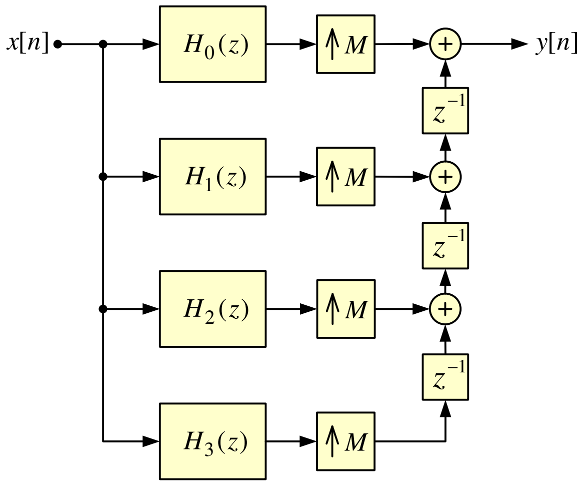
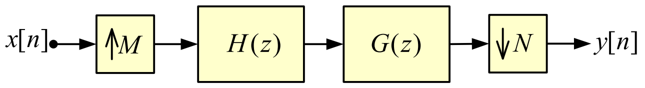


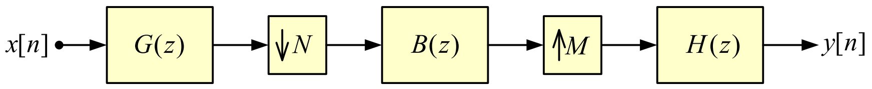
Fig. F.7 Interpolation with a polyphase decomposition, shown for $M = 4$

F.7 Combined Upsampling / Downsampling

The upsampling and downsampling operations can be cascaded as shown in Fig. F.8 to achieve a sampling rate change by a rational factor. The form in part (a) has upsampling followed by filtering, followed by downsampling, and is the more standard configuration for sample rate change. The filter $H(z)$ serves to interpolate the samples, while the second filter $G(z)$ can serve as an anti-aliasing filter before downsampling. In the form in part (b), $G(z)$ can serve as an anti-aliasing filter before processing using the filter $B(z)$ at the lower intermediate rate. The filter $H(z)$ is an interpolating filter.



(a) Upsampling – filtering – downsampling



(b) Downsampling – filtering – upsampling

Fig. F.8 Two forms of cascaded interpolation and downsampling

An interesting case arises if the upsampling and downsampling ratios are the same.

F.7.1 Upsampling – Filtering – Downsampling

The system combining upsampling and downsampling of Fig. F.8(a) with $N = M$ has an output

$$Y(z) = \frac{1}{M} X(z) \sum_{p=0}^{M-1} G(W_M^p z^{1/M}) H(W_M^p z^{1/M}). \tag{F.11}$$

Define the filter related term

$$F(z) = \frac{1}{M} \sum_{p=0}^{M-1} G(W_M^p z^{1/M}) H(W_M^p z^{1/M}). \tag{F.12}$$

Now using a M-phase decomposition of both $G(z)$ and $H(z)$,

$$F(z) = \frac{1}{M} \sum_{q=0}^{M-1} \sum_{r=0}^{M-1} H_q(z) G_r(z) z^{-(q+r)/M} \sum_{p=0}^{M-1} W_M^{-p(q+r)}. \tag{F.13}$$

The last sum takes on one of two values,

$$\sum_{p=0}^{M-1} W_M^{-p(r+q)} = \begin{cases} M, & ((r+q))_M = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{F.14}$$

The non-zero value occurs if $r = q$ when $q = 0$, or $r = M - q$ when $1 \le q < M$. Using this result

$$F(z) = H_0(z)G_0(z) + z^{-1} \sum_{q=1}^{M-1} H_q(z)G_{M-q}(z). \tag{F.15}$$

This decoupled result is illustrated in Fig. F.9. Note that all signal flows are now at one rate – this is not a multirate system. This result was generated analytically, but can also be developed strictly through block diagram manipulations. When the polyphase interpolator and polyphase downsampler are cascaded, the commutators act such that the samples from one branch of the interpolator are shunted over to only one branch of the downsampler.

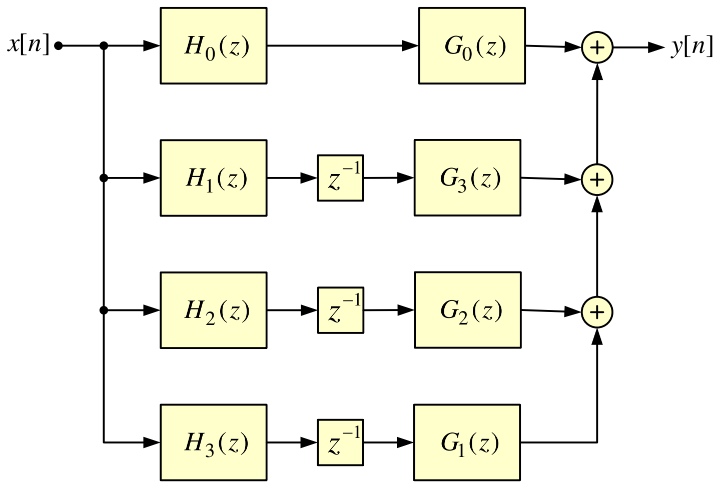
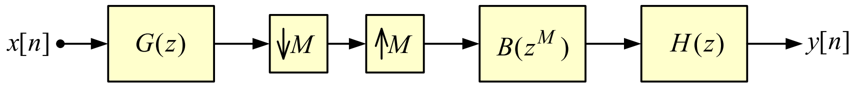


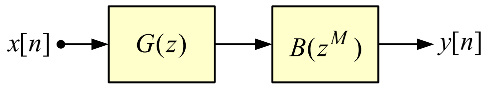
Fig. F.9 Polyphase decomposition for interpolation followed by downsampling, shown for $M = 4$

F.7.2 Downsampling – Filtering – Upsampling

We now consider the system in Fig. F.8(b) with $N = M$. The first step is to redraw the system as shown in Fig. F.10(a). Note that we can freely interchange the filter blocks $G(z^M)$ and $H(z)$. We will assume (1) that $G(z)$ bandlimits the input signal, or that the input signal is already bandlimited, thereby avoiding aliasing in the downsampling operation, and (2) that $H(z)$ is an ideal interpolation filter. With these assumptions the downsampling operation, followed by the upsampling operation, followed by ideal interpolation is an identity operation. This leads to the simplified system shown in Fig. F.10(b). This is no longer a multirate system.



(a) General case: downsampling – filtering – Upsampling



(b) Special case: downsampling – filtering – upsampling

Fig. F.10 Cascaded downsampling, filtering, and interpolation. In the case that the input to the downsampler is bandlimited, and the filter $H(z)$ is an ideal interpolation filter, the second system gives an identical output to the first.

Problems

PF.1 Noble Identities

Prove both the upsampling and downsampling noble identities of Section F.3.

Appendix G

Filtered Cyclostationary Processes

In this appendix, we examine filtered cyclostationary processes. The model will consist of a wide-sense stationary process $x[n]$ being upsampled to become $x_I[n]$ and then that signal being filtered by $h[n]$.

The signal $x_I[n]$ is created from $x[n]$ by inserting $I-1$ zeros between each sample of $x[n]$,

$$x_I[n] = \begin{cases} x[n/M], & n = lM, \\ 0, & \text{otherwise.} \end{cases} \tag{G.1}$$

The filtered signal is

$$\begin{aligned} y[n] &= \sum_{k=-\infty}^{\infty} x_I[k]h[n-k] \\ &= \sum_{i=-\infty}^{\infty} x[i]h[n-iI]. \end{aligned} \tag{G.2}$$

If we let $n = kI + p$ with $0 \leq p < m$, then

$$\begin{aligned} y[kI+p] &= \sum_{i=-\infty}^{\infty} x[i]h[(k-i)I+p], \\ y_p[k] &= \sum_{i=-\infty}^{\infty} x[i]h_p[k-i]. \end{aligned} \tag{G.3}$$

This is in fact a polyphase decomposition of the upsampling system, see Appendix F for more information. The output signal $y[n]$ is formed by interleaving the $y_p[n]$ signals. This is shown explicitly in Fig. G.1.

The intermediate signals $y_p[n]$ are wide-sense stationary signals. The cross-correlations be-

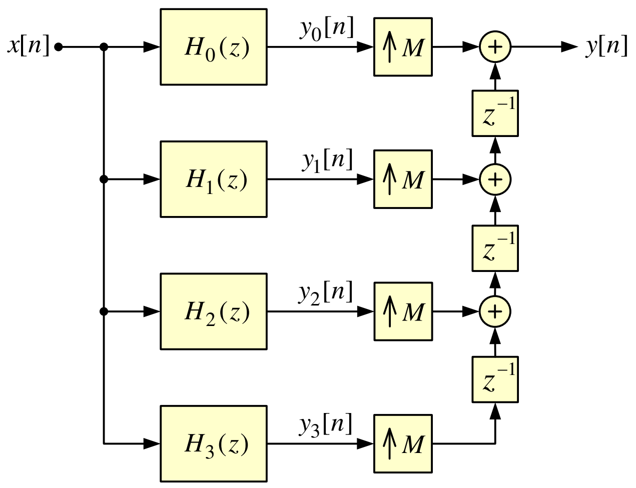


Fig. G.1 Polyphase form of an upsampling system, shown for $M = 4$

tween the $y_p[n]$ signals are

$$E\big[y_p[k]y_q^*[l]\big] = E\big[(h_p[k] * x[k])\,(h_q[l] * x[l])^*\big]. \tag{G.4}$$

Using the results for filtered stationary signals (Appendix D),

$$E\big[y_p[k]y_q^*[l]\big] = r_{h_p h_q}[k-l] * r_{xx}[k-l]. \tag{G.5}$$

For a given p and q, this is wide-sense stationary.

The correlation of $y[n]$ depends on the "phase" of n where the phase of n is given by the modulus $p = ((n))_I$. This means that $y[n]$ is not stationary, but is instead cyclostationary, even though the component signals $y_p[n]$ are jointly wide-sense stationary. For a given n and m, write $n = kI + p$ and $m = lI + q$,

$$\begin{aligned} E\big[y[n]y^*[m]\big] &= E\big[y[kI+p]y^*[lI+q]\big] \\ &= E\big[y_p[k]y_q^*[l]\big] \\ &= E\big[y_p[k+1]y_q^*[l+1]\big] \\ &= E\big[y[n+I]y^*[m+I]\big]. \end{aligned} \tag{G.6}$$

Going from the second line to the third line, we have made use of the stationarity of $y_p[n]$.

A special case occurs if the filter $h[n]$ is an identity filter. In that case we are looking at the cyclostationary correlation of $y[n] = x_I[n]$,

$$E\left[x_I[n]x_I^*[m]\right] = \begin{cases} r_{xx}[(n-m)/I], & \text{if } ((m))_I = ((n))_I = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{G.7}$$

Appendix H

Cholesky Factorization

Consider a Hermitian symmetric positive definite matrix $\mathbf{R}$. The Cholesky decomposition forms a lower/upper (LU) triangular factorization of the matrix $\mathbf{R}$,

$$\mathbf{R} = \mathbf{L}\mathbf{U}, \tag{H.1}$$

where $\mathbf{L}$ is a lower triangular matrix and $\mathbf{U}$ is the upper triangular matrix formed as a Hermitian transpose of $\mathbf{L}$ ($\mathbf{U} = \mathbf{L}^H$).

H.1 Factorization

Expanding the matrix product in Eq. (H.1) for $0 \le i, j < M$,

$$\begin{aligned} \mathbf{R}(i,j) &= \sum_{k=0}^{M-1} \mathbf{L}(i,k)\mathbf{U}(k,j) \\ &= \sum_{k=0}^{M-1} \mathbf{L}(i,k)\mathbf{L}^*(j,k) \\ &= \sum_{k=0}^{\min(i,j)} \mathbf{L}(i,k)\mathbf{L}^*(j,k) \end{aligned} \tag{H.2}$$

In second line, we have replaced $\mathbf{U}$ by $\mathbf{L}^H$. In the third line, we used the fact that $\mathbf{L}$ is triangular to change the upper limit of the sum. We will solve for the elements of $\mathbf{L}$ row by row ($i = 0, \ldots M-1$), then for each row, column by column ($j = 0, \ldots, i$). Rewrite the product as

$$\mathbf{L}(i,j)\mathbf{L}^*(j,j) = \mathbf{R}(i,j) - \sum_{k=0}^{j-1} \mathbf{L}(i,k)\mathbf{L}^*(j,k) \qquad 0 \le j \le i. \tag{H.3}$$

As we work on row i, all elements of $\mathbf{L}$ in the triangular wedge above row i are known. As we work across the row, all elements to the left in the same row are also known. Then for $0 \leq j < i$,[1]

$$\mathbf{L}(i,j) = \frac{\mathbf{R}(i,j) - \sum_{k=0}^{j-1} \mathbf{L}(i,k)\mathbf{L}^*(j,k)}{\mathbf{L}(j,j)} \qquad 0 \leq j < i. \tag{H.4}$$

and for $j = i$,

$$\mathbf{L}(i,i) = \sqrt{\mathbf{R}(i,i) - \sum_{k=0}^{i-1} |\mathbf{L}(i,i)|^2}. \tag{H.5}$$

Note that the quantity inside the square root is purely real since the diagonal elements of a Hermitian symmetric matrix are real. Furthermore, the quantity inside the square root is positive.

Here we do not go into the details of the several other ways to carry out the factorization. For further information see [10] (for the case of real-valued matrices).

Cholesky Factorization

Every Hermitian symmetric positive definite matrix can be *uniquely* factored into a product of a lower triangular and an upper triangular matrix. The matrices are Hermitian transposes of each other. Furthermore, the elements on the diagonals of the factors are positive.

We can also find a a LDU (lower triangular, diagonal, upper triangular) factorization,

$$\mathbf{R} = \mathbf{LDU}. \tag{H.6}$$

In this case, the lower triangular and upper triangular matrices have unity values on the diagonal, and the diagonal matrix contains positive values.

H.2 Solving Linear Equations Using the Cholesky Decomposition

Consider a set of linear equations with a Hermitian symmetric positive definite coefficient matrix $\mathbf{R}$,

$$\mathbf{Rc} = \mathbf{b}. \tag{H.7}$$

Define the auxiliary vector

$$\mathbf{g} = \mathbf{Uc}. \tag{H.8}$$

[1]We have anticipated that $\mathbf{L}(j,j)$ is purely real and omitted the conjugation for the term in the denominator.

The solution of the equations proceeds by first solving a triangular set of equations by back substitution,

$$\mathbf{L}\mathbf{g} = \mathbf{b}. \tag{H.9}$$

The solution vector $\mathbf{g}$ is substituted into the triangular set of equations given by Eq. (H.8), which is solved for the coefficients $\mathbf{c}$.

H.3 Cholesky Decomposition for Covariance Approach Prediction

Consider the case that the matrix $\mathbf{R}$ is a correlation matrix, for instance for the covariance approach to least-squares prediction. The equations to be solved are

$$\mathbf{R}\mathbf{w} = \mathbf{r}, \tag{H.10}$$

and the resulting mean-square error is

$$\varepsilon = \overline{|d[n]|^2} - \mathbf{w}^H\mathbf{R}\mathbf{w}. \tag{H.11}$$

The correlation matrix is factored as $\mathbf{R} = \mathbf{L}\mathbf{U}$. Let $\mathbf{g}$ be the auxiliary vector used in solving the linear equations,

$$\mathbf{g} = \mathbf{U}\mathbf{w}. \tag{H.12}$$

Then the mean-squared error can be expressed as

$$\varepsilon = \overline{|d[n]|^2} - \mathbf{g}^H\mathbf{g}. \tag{H.13}$$

One property of the LU decomposition of a positive definite matrix is a nesting of the solution. Denote $\mathbf{R}_m$ as the $[m \times m]$ submatrix obtained by keeping only the first m rows and first m columns of $\mathbf{R}$. The lower triangular matrix in the LU decomposition of $\mathbf{R}_m$ is itself a submatrix of $\mathbf{L}$. This nesting implies that the vector $\mathbf{g}$ also nests and that the mean-square error is a non-decreasing function of the filter order.

Problems

PH.1 Positivity of the Diagonal Elements

The discussion on the positivity of the diagonal elements of the triangular matrices needs elaboration. In this problem, you will flesh out a full proof.

(a) Show that the trace of a triangular matrix is equal to the sum of the eigenvalues.

(b) Show that the upper left submatrix of **L** is the triangular decomposition of the upper left submatrix of **R**.

PH.2 Computational Complexity

(a) Estimate the computation complexity of the LU decomposition. Do this for the case of matrices with real elements. The computational complexity should be a function of M and should count square roots, divides, adds (or subtracts), and multiplies separately.

(b) Estimate the additional computational complexity to solve the linear equations after the LU decomposition is available.

PH.3 LDU Decomposition

Develop a Cholesky decomposition for an LDU decomposition, where the lower triangular (and upper triangular) matrix has unity values on the diagonal. Show that this procedure avoids square root operations. Is there a computational complexity penalty relative to the LU decomposition?

Appendix I

Durbin Recursion

Consider a Hermitian symmetric $[M \times M]$ Toeplitz matrix $\mathbf{R}_M$. We want to solve the Wiener-Hopf equations,

$$\mathbf{R}_M \mathbf{w}_M = \mathbf{r}_M. \tag{I.1}$$

The matrix $\mathbf{R}_M$ is both Hermitian symmetric and persymmetric (see Appendix C). This matrix is completely defined by its first row, $\big[r[0], r[1], \ldots, r[M-1]\big]$. The righthand side vector $\mathbf{r}_M$ is

$$\mathbf{r}_M = \begin{bmatrix} r^*[1] \\ r^*[2] \\ \vdots \\ r^*[M] \end{bmatrix}. \tag{I.2}$$

An efficient approach to solving the Toeplitz equations is through an order recursive procedure, the Durbin algorithm.[1] Given the solution for order $m-1$, we want to find the solution for order m. We can extend the $[(m-1) \times (m-1)]$ correlation matrix $\mathbf{R}_{m-1}$ to the matrix $\mathbf{R}_m$ by adding a column to the right and a row at the bottom,

$$\mathbf{R}_m = \begin{bmatrix} \mathbf{R}_{m-1} & \mathbf{J}\mathbf{r}^*_{m-1} \\ \mathbf{r}^T_{m-1}\mathbf{J} & r[0] \end{bmatrix}, \tag{I.3}$$

where the exchange matrix $\mathbf{J}$ is described in Appendix C. The order m Wiener-Hopf equations can

[1]We follow the notation of [10] and refer to the solution method as the Durbin recursion if the righthand side vector shares elements with the autocorrelation matrix. The Levinson algorithm (Appendix J) solves the general righthand side problem by wrapping another computational layer around each iteration of the Durbin procedure.

be written as

$$\begin{bmatrix} \mathbf{R}_{m-1} & \mathbf{J}\mathbf{r}^*_{m-1} \\ \mathbf{r}^T_{m-1}\mathbf{J} & r[0] \end{bmatrix} \begin{bmatrix} \mathbf{v}_{m-1} \\ -k_m \end{bmatrix} = \begin{bmatrix} \mathbf{r}_{m-1} \\ r^*[m] \end{bmatrix}, \tag{I.4}$$

where

$$\mathbf{w}_m = \begin{bmatrix} \mathbf{v}_{m-1} \\ -k_m \end{bmatrix}. \tag{I.5}$$

The top $m-1$ rows of the matrix-vector equations give

$$\mathbf{R}_{m-1}\mathbf{v}_{m-1} - k_m\mathbf{J}\mathbf{r}^*_{m-1} = \mathbf{r}_{m-1}. \tag{I.6}$$

Solving for $\mathbf{v}_{m-1}$,

$$\begin{aligned} \mathbf{v}_{m-1} &= \mathbf{R}^{-1}_m\mathbf{r}_{m-1} + k_m\mathbf{R}^{-1}_{m-1}\mathbf{J}\mathbf{r}^*_{m-1} \\ &= \mathbf{w}_{m-1} + k_m\mathbf{J}(\mathbf{R}^{-1}_{m-1})^T\mathbf{r}^*_{m-1} \\ &= \mathbf{w}_{m-1} + k_m\mathbf{J}(\mathbf{R}^{-1}_{m-1})^*\mathbf{r}^*_{m-1} \\ &= \mathbf{w}_{m-1} + k_m\mathbf{J}\mathbf{w}^*_{m-1}. \end{aligned} \tag{I.7}$$

To go from the first line to the second line, we have used the fact that the inverse of a Toeplitz matrix (here $\mathbf{R}_{m-1}$) is persymmetric (see Appendix C) and that for a persymmetric matrix $\mathbf{A}$, $\mathbf{AJ} = \mathbf{JA}^T$. To go from the second line to the third line, we have used the fact that the inverse of a Hermitian symmetric matrix is Hermitian symmetric.

We still need an expression for k_m. From the last row of Eq. (I.4),

$$\mathbf{r}^T_{m-1}\mathbf{J}\mathbf{v}_{m-1} - k_m r[0] = r^*[m]. \tag{I.8}$$

Now eliminating $\mathbf{v}_{m-1}$ using the last line of Eq. (I.7),

$$\mathbf{r}^T_{m-1}\mathbf{J}(\mathbf{w}_{m-1} + k_m\mathbf{J}\mathbf{w}^*_{m-1}) - k_m r[0] = r^*[m]. \tag{I.9}$$

Solving for k_m,

$$k_m = -\frac{r^*[m] - \mathbf{r}^T_{m-1}\mathbf{J}\mathbf{w}_{m-1}}{r[0] - \mathbf{r}^T_{m-1}\mathbf{w}^*_{m-1}}. \tag{I.10}$$

We recognize the denominator in Eq. (I.10) as the prediction residual energy for the order $m-1$ predictor, ε_{m-1} (see Eq. (4.8)), giving

$$k_m = -\frac{r^*[m] - \mathbf{r}^T_{m-1}\mathbf{J}\mathbf{w}_{m-1}}{\varepsilon_{m-1}}. \tag{I.11}$$

Now that we have an expression for $\mathbf{v}_{m-1}$ and an expression for k_m, we can write

$$\mathbf{w}_m = \begin{bmatrix} \mathbf{w}_{m-1} + k_m \mathbf{J}\mathbf{w}^*_{m-1} \\ -k_m \end{bmatrix}. \tag{I.12}$$

A further savings in computation ensues if we develop a recursion for the prediction residual energy. The prediction residual energy for the order m predictor is

$$\begin{aligned} \varepsilon_m &= r[0] - \mathbf{r}^T_m \mathbf{w}^*_m \\ &= r[0] - \begin{bmatrix} \mathbf{r}^T_{m-1} & r^*[m] \end{bmatrix} \begin{bmatrix} \mathbf{v}^*_{m-1} \\ -k^*_m \end{bmatrix} \\ &= r[0] - \begin{bmatrix} \mathbf{r}^T_{m-1} & r^*[m] \end{bmatrix} \begin{bmatrix} \mathbf{w}^*_{m-1} + k^*_m \mathbf{J}\mathbf{w}_{m-1} \\ -k^*_m \end{bmatrix} \\ &= \left(r[0] - \mathbf{r}^T_{m-1}\mathbf{w}^*_{m-1}\right) + k^*_m \left(r^*[m] - \mathbf{r}^T_{m-1}\mathbf{J}\mathbf{w}_{m-1}\right) \\ &= \varepsilon_{m-1} - k^*_m k_m \varepsilon_{m-1} \\ &= \varepsilon_{m-1}(1 - |k_m|^2). \end{aligned} \tag{I.13}$$

The intermediate variables k_m are known as reflection coefficients or sometimes as the partial correlation (parcor) coefficients. These coefficients can be used to implement a predictor in a lattice form [35]. From the last line of Eq. (I.13), the magnitude of reflection coefficient k_m can be expressed as

$$|k_m|^2 = 1 - \frac{\varepsilon_m}{\varepsilon_{m-1}}. \tag{I.14}$$

Since the residual energies are non-increasing with order, $|k_m| \leq 1$.

The Durbin recursion generates an optimal forward predictor $\mathbf{w}_m$ and an optimal backward predictor $\mathbf{J}\mathbf{w}^*_m$ at the same time. Note how Eq. (I.12) combines the forward predictor with the backward predictor scaled by the reflection coefficient. This is perhaps easier to see if we examine the prediction error filter coefficient vector,

$$\mathbf{a}_m = \begin{bmatrix} 1 \\ -\mathbf{w}_m \end{bmatrix}. \tag{I.15}$$

Then the Eq. (I.12) can be written as

$$\mathbf{a}_m = \begin{bmatrix} \mathbf{a}_{m-1} \\ 0 \end{bmatrix} + k_m \begin{bmatrix} 0 \\ \mathbf{J}\mathbf{a}^*_{m-1} \end{bmatrix}. \tag{I.16}$$

Indeed Eq. (I.10) can also be expressed compactly in terms of the vector $\mathbf{a}_{m-1}$,

$$k_m = -\frac{\mathbf{r}_m^T \mathbf{J} \mathbf{a}_{m-1}}{\varepsilon_{m-1}}. \tag{I.17}$$

Durbin Recursion

Given the autocorrelation sequence $[r[0], r[1], \ldots, r[M+1]]$, such that the Toeplitz matrix $\mathbf{R}$ is positive definite, solve the Wiener-Hopf equations. Initialization: Set $\varepsilon_0 = r[0]$ and set the predictor coefficient vector $\mathbf{w}_0$ and the correlation vector $\mathbf{r}_0$ to be empty. Then for each m, $m = 1, \ldots, M$, repeat the following steps.

$$k_m = -\frac{r^*[m] - \mathbf{r}_{m-1}^T \mathbf{J} \mathbf{w}_{m-1}}{\varepsilon_{m-1}} \qquad \text{or} \qquad k_m = -\frac{\mathbf{r}_m^T \mathbf{J} \mathbf{a}_{m-1}}{\varepsilon_{m-1}} \tag{I.18}$$

$$\varepsilon_m = \varepsilon_{m-1}(1 - |k_m|^2) \tag{I.19}$$

$$\mathbf{w}_m = \begin{bmatrix} \mathbf{w}_{m-1} + k_m \mathbf{J} \mathbf{w}_{m-1}^* \\ -k_m \end{bmatrix} \qquad \text{or} \qquad \mathbf{a}_m = \begin{bmatrix} \mathbf{a}_{m-1} \\ 0 \end{bmatrix} + k_m \begin{bmatrix} 0 \\ \mathbf{J} \mathbf{a}_{m-1}^* \end{bmatrix} \tag{I.20}$$

Conjugated Filter Coefficients

The reader is reminded that the filter coefficients appear in the actual filter structures as conjugated quantities. For instance the elements of **w** appear on a direct form predictor as conjugated quantities. However, the reflection coefficients which do appear in the lattice forms of the filters appear there as both a conjugated and non-conjugated value, see for instance Fig. 4.5. The reader is also warned that the definition of the reflection coefficients is not entirely consistent in the literature – sometimes they are the conjugates of what is used here; sometimes they are the negatives of the convention used here.

I.1 Step-up Procedure

The Durbin recursion uses a step-up procedure to calculate the filter coefficients from the reflection coefficients (k_m). The step-up procedure can be used in isolation from the rest of the Durbin recursion.

Step-Up Procedure

Given a set of reflection coefficients, the predictor coefficients can be calculated with the recursion.

$$\mathbf{w}_m = \begin{bmatrix} \mathbf{w}_{m-1} + k_m \mathbf{J}\mathbf{w}^*_{m-1} \\ -k_m \end{bmatrix} \qquad \text{or} \qquad \mathbf{a}_m = \begin{bmatrix} \mathbf{a}_{m-1} \\ 0 \end{bmatrix} + k_m \begin{bmatrix} 0 \\ \mathbf{J}\mathbf{a}^*_{m-1} \end{bmatrix}. \tag{I.21}$$

I.2 Step-down Procedure

We can also define a step-down procedure which generates a set of reflection coefficients from a vector of prediction error filter coefficients. From Eq. (I.16), we can write

$$\mathbf{J}\mathbf{a}^*_m = \begin{bmatrix} 0 \\ \mathbf{J}\mathbf{a}^*_{m-1} \end{bmatrix} + k^*_m \begin{bmatrix} \mathbf{a}_{m-1} \\ 0 \end{bmatrix}. \tag{I.22}$$

Then from Eq. (I.16) and Eq. (I.22),

$$\mathbf{a}_m - k_m \mathbf{J}\mathbf{a}^*_m = (1 - |k_m|^2) \begin{bmatrix} \mathbf{a}_{m-1} \\ 0 \end{bmatrix}. \tag{I.23}$$

Now we can solve for the lower order prediction error coefficients, giving us the step-down procedure.

Step-down Procedure

Given a set of filter coefficients, this algorithm generates the reflection coefficients k_m. We identify k_m as the last coefficient of $\mathbf{a}_m$. Then we apply the step-down procedure to find $\mathbf{a}_{m-1}$.

$$\begin{bmatrix} \mathbf{a}_{m-1} \\ 0 \end{bmatrix} = \frac{\mathbf{a}_m - k_m \mathbf{J}\mathbf{a}^*_m}{1 - |k_m|^2}. \tag{I.24}$$

As long as $|k_m|$ is not unity, the coefficients of the next lower order filter can be determined from and the procedure repeated to find the next lower order reflection coefficient.

I.3 Reflection Coefficients to Correlations

The reflection coefficients can be used to find correlation values which would have generated them. The reflection coefficients will be used in a step-up procedure to create both the predictor coefficients and the correlation values.

Inverse Durbin Recursion

Given a set of reflection coefficients, we can find both correlation values and filter coefficients. Initialization: Set $\varepsilon_0 = r[0]$ and $\mathbf{r}_0$ and $\mathbf{w}_0$ to be empty vectors. (If we set ε_0 to unity, we get normalized correlation values.) The inverse Durbin recursion uses the steps of the Durbin recursion for $m = 1, 2, \ldots,$

$$\varepsilon_m = \varepsilon_{m-1}(1 - |k_m|^2), \tag{I.25}$$

$$\mathbf{w}_m = \begin{bmatrix} \mathbf{w}_{m-1} + k_m \mathbf{J}\mathbf{w}^*_{m-1} \\ -k_m \end{bmatrix}, \tag{I.26}$$

$$r^*[m] = k_m \varepsilon_{m-1} + \mathbf{r}^T_{m-1}\mathbf{J}\mathbf{w}_{m-1}, \tag{I.27}$$

$$\mathbf{r}_m = \begin{bmatrix} \mathbf{r}_{m-1} \\ r^*[m] \end{bmatrix}. \tag{I.28}$$

The procedures discussed show that for positive definite correlations (reflection coefficients less than unity in magnitude), there are one-to-one correspondences between the correlation values, the filter coefficients, and the reflection coefficients. Given any one of these, the others can be uniquely determined.

- Correlations to reflection coefficients and predictor coefficients: Durbin recursion.
- Reflection coefficients to predictor coefficients: Step-up procedure.
- Predictor coefficients to reflection coefficients: Step-down procedure.
- Reflection coefficients to correlations: Inverse Durbin recursion.
- Predictor coefficients to correlations: Step-down procedure to get the reflection coefficients, followed by the inverse Durbin recursion.

Problems

PI.1 Computational Complexity of the Durbin Recursion

Consider the Durbin recursion for a real signal. This means that the all quantities involved will be purely real.

(a) Consider going from order $m-1$ to order m. Count the number of multiplies, adds, and divides necessary.

(b) What are the totals of the number of multiples, adds, and divides for the Durbin recursion, ending up at order M?

Appendix J

Levinson Algorithm

Consider a symmetric $[M \times M]$ Toeplitz matrix $\mathbf{R}_M$. We want to solve the linear equations,

$$\mathbf{R}_M \mathbf{c}_M = \mathbf{b}_M. \tag{J.1}$$

The matrix $\mathbf{R}_M$ is both symmetric and persymmetric (see Appendix C).

The Toeplitz equations can be solved using through an order-recursive procedure, the Levinson algorithm. Given the solution for order $m-1$, we find the solution for order m. The $[m \times m]$ correlation matrix can be written in terms of the $[(m-1) \times (m-1)]$ correlation matrix,

$$\mathbf{R}_m = \begin{bmatrix} \mathbf{R}_{m-1} & \mathbf{J}\mathbf{r}^*_{m-1} \\ \mathbf{r}^T_{m-1}\mathbf{J} & r[0] \end{bmatrix}, \tag{J.2}$$

where the vector $\mathbf{r}_{m-1}$ is the same vector that appears in the Durbin algorithm of Appendix I, and the exchange matrix $\mathbf{J}$ is also used in Appendix I. We will track two parallel algorithms, the Durbin recursion and the modifications to adapt the solution to the general righthand side vector $\mathbf{b}_M$. For the Durbin form we solve the $[m \times m]$ equations

$$\mathbf{R}_m \mathbf{w}_m = \mathbf{r}_m. \tag{J.3}$$

For the Levinson, we need to solve the following equations (shown with the partitioned $\mathbf{R}_m$ matrix),

$$\begin{bmatrix} \mathbf{R}_{m-1} & \mathbf{J}\mathbf{r}^*_{m-1} \\ \mathbf{r}^T_{m-1}\mathbf{J} & r[0] \end{bmatrix} \begin{bmatrix} \mathbf{z}_{m-1} \\ \alpha \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{m-1} \\ b[m] \end{bmatrix}. \tag{J.4}$$

The first $m-1$ rows of Eq. (J.4) give us,

$$\mathbf{R}_{m-1}\mathbf{z}_{m-1} + \alpha \mathbf{J}\mathbf{r}^*_{m-1} = \mathbf{b}_{m-1}, \tag{J.5}$$

and then solving for $\mathbf{z}_{m-1}$,

$$\begin{aligned}\mathbf{z}_{m-1} &= \mathbf{R}_{m-1}^{-1}\mathbf{b}_{m-1} - \alpha\mathbf{R}_{m-1}^{-1}\mathbf{J}\mathbf{r}_{m-1}^{*} \\ &= \mathbf{c}_{m-1} - \alpha\mathbf{J}\mathbf{w}_{m-1}^{*},\end{aligned} \tag{J.6}$$

where $\mathbf{c}_{m-1}$ is the result of the Levinson algorithm of order $m-1$. As in the corresponding step in the Durbin recursion, we have used the fact that $\mathbf{R}_m^{-1}$ is both persymmetric and Hermitian. Now the last row of Eq. (J.4) gives us

$$\begin{aligned}\mathbf{r}_{m-1}^{T}\mathbf{J}\mathbf{z}_{m-1} + \alpha r[0] &= b[m], \\ \mathbf{r}_{m-1}^{T}\mathbf{J}(\mathbf{c}_{m-1} - \alpha\mathbf{J}\mathbf{w}_{m-1}^{*}) + \alpha r[0] &= b[m].\end{aligned} \tag{J.7}$$

Finally solving for α gives

$$\begin{aligned}\alpha &= \frac{b[m] - \mathbf{r}_{m-1}^{T}\mathbf{J}\mathbf{c}_{m-1}}{\mathbf{r}_{m-1}^{T}\mathbf{w}_{m-1}^{*}} \\ &= \frac{b[m] - \mathbf{r}_{m-1}^{T}\mathbf{J}\mathbf{c}_{m-1}}{\varepsilon_{m-1}}.\end{aligned} \tag{J.8}$$

In the second line, the denominator has been replaced by ε_{m-1} from the first line of Eq. (I.13). We can solve for α using only vectors available at iteration $m-1$. This value of α is used to find $\mathbf{z}_{m-1}$ using Eq. (J.6). Finally the solution at iteration m is

$$\mathbf{c}_m = \begin{bmatrix}\mathbf{z}_{m-1} \\ \alpha\end{bmatrix}. \tag{J.9}$$

Levinson Algorithm

The mth stage of the Levinson algorithm operates on the the output of the Durbin recursion for order $m-1$. From the Durbin recursion, we have available $\mathbf{w}_{m-1}$ and ε_{m-1}. From the previous iteration of the Levinson algorithm, we have available $\mathbf{c}_{m-1}$. Now calculate

$$\alpha = \frac{b[m] - \mathbf{r}_{m-1}^{T}\mathbf{J}\mathbf{c}_{m-1}}{\varepsilon_{m-1}}, \tag{J.10}$$

$$\mathbf{c}_m = \begin{bmatrix}\mathbf{c}_{m-1} - \alpha\mathbf{J}\mathbf{w}_{m-1}^{*} \\ \alpha\end{bmatrix} \qquad \text{or} \qquad \mathbf{c}_m = \begin{bmatrix}\mathbf{c}_{m-1} \\ 0\end{bmatrix} + \alpha\mathbf{J}\mathbf{a}_{m-1}. \tag{J.11}$$

Next run the Durbin recursion to bring its values up to order m, ready for the next iteration of the Levinson algorithm.

J.1 Alternate Formulation

If we omit the substitution of $\mathbf{c}_{m-1}$ in the expression for $\mathbf{z}_{m-1}$, the second term in the numerator for α, Eq. (J.8) becomes $\mathbf{w}_{m-1}^T \mathbf{J} \mathbf{b}_{m-1}$. Now we can write

$$\begin{aligned} \alpha &= \frac{b[m] - \mathbf{w}_{m-1}^T \mathbf{J} \mathbf{b}_{m-1}}{\varepsilon_{m-1}} \\ &= \frac{\mathbf{a}_{m-1}^T \mathbf{J} \mathbf{b}_m}{\varepsilon_{m-1}}. \end{aligned} \tag{J.12}$$

Problems

PJ.1 Computational Complexity of the Levinson Algorithm

Consider the Levinson algorithm for a real signal. This means that the all quantities involved will be purely real.

(a) Consider going from order $m-1$ to order m. Count the number of multiplies, adds, and divides above those required by the Durbin recursion.

(b) What are the totals of the number of multiples, adds, and divides for the Levinson recursion, ending up at order M?

Appendix K

Minimum Phase and All-Pass Systems

This appendix examines the properties of minimum-phase and all-pass filters. The analysis deals with the case of *complex* filter coefficients. The discussion here has been taken from [22], but tailored to give results which are used in the present document. Our focus will be on FIR minimum-phase filters. The reference gives a more general treatment of minimum phase filters with rational transfer functions and gives examples of the decomposition of general filters into the cascade of a minimum phase and an all-pass filter.

K.1 FIR Filter Response

An FIR filter can be written as the product of its root factors,

$$\begin{aligned} H(z) &= Gz^{-n_0} \prod_{k=1}^{M} (1 - z_k z^{-1}) \\ &= Gz^{-n_0} \prod_{k=1}^{M} (1 - r_k e^{j\theta_k} z^{-1}), \end{aligned} \tag{K.1}$$

where in the last part of the equation the poles have been expressed as $z_k = r_k e^{j\theta_k}$.

K.1.1 Amplitude Response

The magnitude-squared response of a filter $H(z)$ can be evaluated from the z-transform as

$$\begin{aligned} |H(\omega)|^2 &= H(\omega)H^*(\omega) \\ &= H(z)H^*(1/z^*)\big|_{z=e^{j\omega}} . \end{aligned} \tag{K.2}$$

The z-transform relationship for the second term can be seen to result in

$$H^*(1/z^*) = \sum_{n=-\infty}^{\infty} h_n^* z^n. \tag{K.3}$$

The double conjugation (once on z and again on $H(\cdot)$) has the effect of leaving the z^n terms unconjugated. Only the coefficients are conjugated. The response $H^*(1/z^*)$ has an expansion in positive powers of z, i.e. the time response is flipped about $n = 0$.

For the filter of Eq. (K.1), the magnitude-squared response is

$$\begin{aligned} |H(\omega)|^2 &= |G|^2 \prod_{k=1}^{M} (1 - z_k z^{-1})(1 - z_k^* z) \Big|_{z=e^{j\omega}} \\ &= |G|^2 \prod_{k=1}^{M} (1 - 2\mathrm{Re}[z_k e^{-j\omega}] + |z_k|^2) \\ &= |G|^2 \prod_{k=1}^{M} (1 - 2r_k \cos(\omega - \theta_k) + r_k^2). \end{aligned} \tag{K.4}$$

The frequency dependent terms in the magnitude-squared response are $\sin\omega$ and $\cos\omega$. For a system with real coefficients, the magnitude-squared response is symmetrical about $\omega = 0$ and can be expressed as a polynomials in $\cos\omega$.

K.1.2 Phase Response

The phase response of a filter with frequency response $H(\omega)$ is[1]

$$\arg[H(\omega)] = \tan^{-1}\left(\frac{H_I(\omega)}{H_R(\omega)}\right), \tag{K.5}$$

where $H_R(\omega)$ and $H_I(\omega)$ are the real and imaginary parts of the frequency response, respectively.

For the response given by Eq. (K.1), the phase response is

$$\begin{aligned} \arg[H(\omega)] &= \arg[G] - n_0\omega + \sum_{k=1}^{M} \tan^{-1}\left(\frac{-\mathrm{Im}[z_k e^{-j\omega}]}{1 - \mathrm{Re}[z_k e^{-j\omega}]}\right) \\ &= \arg[G] - n_0\omega + \sum_{k=1}^{M} \tan^{-1}\left(\frac{r_k \sin(\omega - \theta_k)}{1 - r_k \cos(\omega - \theta_k)}\right). \end{aligned} \tag{K.6}$$

For a system with real coefficients, the phase response is anti-symmetrical about $\omega = 0$.

[1]Some authors define the phase with a negative sign. Here, we will refer to the negative of the phase as the *phase lag*.

K.1.3 Group Delay Response

The group delay response is the negative derivative of the phase response[2]

$$\begin{aligned}\tau_g(\omega) &= -\frac{d\arg[H(\omega)]}{d\omega} \\ &= \frac{H_I(\omega)\dfrac{dH_R(\omega)}{d\omega} - H_R(\omega)\dfrac{dH_I(\omega)}{d\omega}}{H_R^2(\omega)+H_I^2(\omega)}.\end{aligned} \tag{K.7}$$

A more compact form of the same result is

$$\tau_g(\omega) = -\mathrm{Im}\left[\frac{dH(\omega)/d\omega}{H(\omega)}\right]. \tag{K.8}$$

Since the group delay is the derivative of the phase lag, the area under the group delay curve over a frequency interval of 2π is equal to the change in the phase lag over the same interval.

Consider the group delay for just one of the terms in the response given in Eq. (K.1),[3]

$$\begin{aligned}\tau_g^{(z_k)}(\omega) &= \frac{|z_k|^2 - \mathrm{Re}[z_k e^{-j\omega}]}{1-2\mathrm{Re}[z_k e^{-j\omega}] + |z_k|^2} \\ &= \frac{r_k^2 - r_k\cos(\omega-\theta_k)}{1-2r_k\cos(\omega-\theta_k)+r_k^2}.\end{aligned} \tag{K.9}$$

The overall group delay for the rational transfer function is

$$\begin{aligned}\tau_g(\omega) &= n_o + \sum_{k=1}^{M}\frac{|z_k|^2 - \mathrm{Re}[z_k e^{-j\omega}]}{1-2\mathrm{Re}[z_k e^{-j\omega}] + |z_k|^2} \\ &= n_o + \sum_{k=1}^{M}\frac{r_k^2 - r_k\cos(\omega-\theta_k)}{1-2r_k\cos(\omega-\theta_k)+r_k^2}.\end{aligned} \tag{K.10}$$

The group delay response has terms in $\sin\omega$ and $\cos\omega$. For a system with real coefficients, the group delay response is symmetrical about $\omega = 0$ and can be expressed as a ratio of polynomials in $\cos\omega$. As a ratio of polynomials, the group delay is often better behaved than the phase response – the phase response can have phase jumps and ambiguities of multiples of 2π. Thus it may be preferable to characterize a system or filter by the group delay response rather than the phase response.

[2]The derivation of the expression for the group delay response uses the identity $d\tan^{-1}(y/x)/du = (x\,dy/du - y\,dx/du)/(x^2+y^2)$.

[3]The result uses the relations $d\mathrm{Re}[z_k e^{-j\omega}]/d\omega = \mathrm{Im}[z_k e^{-j\omega}]$ and $d\mathrm{Im}[z_k e^{-j\omega}]/d\omega = -\mathrm{Re}[z_k e^{-j\omega}]$.

K.2 Minimum-Phase Systems

A causal filter is said to *minimum-phase* if all of its zeros are inside the unit circle. A *maximum-phase* system has all of its zeros outside the unit circle. A system with a zero on the unit circle is not strictly minimum-phase.

The expansion of $H(z)$ in Eq. (K.1) can be split into three parts.

$$H(z) = H_{\text{min}}(z) H_{\text{uc}}(z) H_{\text{max}}(z), \tag{K.11}$$

where $H_{\text{min}}(z)$ is minimum-phase, $H_{\text{uc}}(z)$ has unit circle zeros, and $H_{\text{max}}(z)$ is maximum-phase. A minimum-phase system has only the first factor. For this discussion, we have set the delay factor n_0 which appeared in Eq. (K.1) to zero.[4]

K.2.1 Log-Magnitude Response of a Minimum-Phase Filter

Consider a causal stable minimum phase filter with real coefficients,

$$H(z) = \frac{B(z)}{A(z)}. \tag{K.12}$$

Both $B(z)$ and $A(z)$ are minimum-phase. The log-magnitude-squared response is

$$\log\bigl(|H(\omega)|^2\bigr) = \log\bigl(|B(\omega)|^2\bigr) - \log\bigl(|A(\omega)|^2\bigr). \tag{K.13}$$

The average log-spectrum of, say, $|B(\omega)|^2$ is

$$\bar{B}_{2L} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \log\bigl(|B(\omega)|^2\bigr)\, d\omega. \tag{K.14}$$

We will write this integral as a contour integral with $z = \exp(\omega)$ and the contour being the unit circle traversed in the counter clockwise direction as ω goes from $-\pi$ to π.

$$\begin{aligned}
\bar{B}_{2L} &= \frac{1}{2\pi j} \oint_C \log\bigl(|B(z)|^2\bigr) \frac{1}{z}\, dz \\
&= \frac{1}{2\pi j} \oint_C \log\bigl(B(z)\bigr) \frac{1}{z}\, dz + \frac{1}{2\pi j} \oint_C \log\bigl(B^*(1/z^*)\bigr) \frac{1}{z}\, dz \\
&= \frac{1}{2\pi j} \oint_C \log\bigl(B(1/z)\bigr) \frac{1}{z}\, dz + \frac{1}{2\pi j} \oint_C \log\bigl(B^*(1/z^*)\bigr) \frac{1}{z}\, dz.
\end{aligned} \tag{K.15}$$

[4]Having $n_0 < 0$ would violate causality. Having $n_0 > 0$ gives a zer o at infinity which will prevent the response from being minimum phase.

We have used Eq. (K.2) to express the magnitude squared as a product of $B(z)$ and $B^*(1/z^*)$. This product becomes a sum after taking the log. From the second line to the third, we have replaced $B(z)$ by $B(1/z)$ since on the unit circle, the integral of $\log\bigl(B(\omega)\bigr)$ is the same as the integral of $\log\bigl(B(-\omega)\bigr)$.

Consider the first contour integral in the last line of Eq. (K.15). The singularities of $\log\bigl(B(z)\bigr)$ are the poles of $B(z)$ (at the origin) and the zeros of $B(z)$ (inside the unit circle). Then the singularities of $\log\bigl(B(1/z)\bigr)$ are all outside the unit circle. The contour in the integral "encloses" the region inside the circle and this region is analytic. We can now apply the Cauchy integral formula [4],

$$f(z_o) = \frac{1}{2\pi j} \oint_C \frac{f(z)}{z - z_o}\, dz. \tag{K.16}$$

In our case, $f(z) = \log\bigl(B(1/z)\bigr)$ and $z_o = 0$. Then with $B(1/z) = b_0 + b_1 z + b_2 z^2, \ldots$, the integral evaluates to $\log(b_0)$. A similar argument can be applied to the second contour integral in Eq. (K.15) resulting in the value $\log(b_o^*)$. Then

$$\begin{aligned} \bar{B}_{2L} &= \log(b_0) + \log(b_0^*) \\ &= 2\log\bigl(|b_0|\bigr). \end{aligned} \tag{K.17}$$

Finally applying the same reasoning to the integral of $\log\bigl(|A(\omega)|^2\bigr)$, the mean of the log-magnitude-squared spectrum is

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} \log\bigl(|H(\omega)|^2\bigr)\, d\omega = 2\log\bigl(|b_0/a_0|\bigr). \tag{K.18}$$

For an integral involving $\log\bigl(|H(\omega)|^n\bigr)$, the mean is scaled by $n/2$,

$$\frac{1}{2\pi} \int_{-\pi}^{\pi} \log\bigl(|H(\omega)|^n\bigr)\, d\omega = n\log\bigl(|b_0/a_0|\bigr). \tag{K.19}$$

This expression is valid for all integer n. For a magnitude curve in dB, the mean value is

$$\bar{H}_{\text{dB}} = 20\log_{10}\bigl(|b_0/a_0|\bigr). \tag{K.20}$$

K.3 All-Pass Filters

An all-pass filter is a filter for which the magnitude of the frequency response is constant. All-pass filters, except for a trivial filters, are IIR.

Start with the simplest non-trivial all-pass filter, which can form one section of a causal, stable all-pass filter,

$$H_{\text{all}}^{(k)}(z) = \frac{z^{-1} - \alpha_k^*}{1 - \alpha_k z^{-1}}. \tag{K.21}$$

Here we have expressed the all-pass filter in terms of its pole location α_k. Since this filter is causal and stable, $|\alpha_k| < 1$.

A general all-pass filter can be formed as the cascade of all-pass sections

$$H_{\text{all}}(z) = \prod_{k=1}^{K} \frac{z^{-1} - \alpha_k^*}{1 - \alpha_k z^{-1}}. \tag{K.22}$$

If the all-pass filter has real coefficients, then for each complex root α_k, there must be a corresponding conjugate root α_k^*.

From Eq. (K.22) we can see that an all-pass filter is characterized by having a numerator polynomial which has the coefficients of the denominator polynomial, conjugated and in reverse order,

$$H_{\text{all}}(z) = \frac{z^{-K} D^*(1/z^*)}{D(z)}. \tag{K.23}$$

Then from Eq. (K.2), we see that any filter of this form has constant magnitude

$$|H_{\text{all}}(\omega)|^2 = 1. \tag{K.24}$$

K.3.1 Phase Response of an All-Pass Filter

If α_k is expressed as $r_k e^{j\theta_k}$, the phase response for an all-pass section is

$$\begin{aligned} \arg[H_{\text{all}}^{(k)}(\omega)] &= \arg[e^{-j\omega} - \alpha_k^*] - \arg[1 - \alpha_k e^{-j\omega)}] \\ &= \arg[e^{-j\omega}] + \arg[1 - \alpha_k^* e^{j\omega}] - \arg[1 - \alpha_k e^{-j\omega}] \\ &= -\omega - 2\tan^{-1}\left(\frac{r_k \sin(\omega - \theta_k)}{1 - r_k \cos(\omega - \theta_k)}\right) \end{aligned} \tag{K.25}$$

The denominator inside the $\tan^{-1}(\cdot)$ function is positive for $r_k < 1$ (corresponding to a stable, causal all-pass filter). The numerator can change sign, and so the phase contribution due to the $\tan^{-1}(\cdot)$ function is limited to $\pm\pi/2$ (and $2\tan^{-1}(\cdot)$ is limited to $\pm\pi$). The $\tan^{-1}(\cdot)$ function contributes an oscillation around the linear phase term – the phase is monotonic downward (as shown in the discussion of the group delay response which follows). The phase response for each all-pass section decreases by 2π as ω increases by 2π.

The phase response for the overall filter is the sum of the phases for each section. Thus the overall phase is monotonic downward and decreases by $2\pi K$ as ω increases by 2π.

$$\arg[H_{\text{all}}(\omega)] = \phi - K\omega - 2\sum_{k=1}^{K} \tan^{-1}\left(\frac{r_k \sin(\omega - \theta_k)}{1 - r_k \cos(\omega - \theta_k)}\right) \tag{K.26}$$

For an all-pass section, real roots have $\theta_k = 0$, or $\theta_k = \pm\pi$. For an overall all-pass filter with real coefficients, for each section with a complex root, there is another section with the conjugate root. These complex conjugate pairs of roots conspire to make the phase anti-symmetrical about $\omega = 0$ and result in an overall filter with real coefficients.

K.3.2 Group Delay Response of an All-Pass Filter

The group delay of the all-pass filter can be determined from Eq. (K.26),

$$\begin{aligned} \tau_g(\omega) &= K + 2\sum_{k=1}^{K} \frac{r_k \cos(\omega - \theta_k) - r_k^2}{1 - 2r_k \cos(\omega - \theta_k) + r_k^2} \\ &= \sum_{k=1}^{K} \frac{1 - r_k^2}{1 - 2r_k \cos(\omega - \theta_k) + r_k^2} \end{aligned} \tag{K.27}$$

In the second expression, we see that for $r_k < 1$, the group delay for an all-pass filter is the ratio of positive quantities and hence is always positive. The denominator of the group delay response for a filter section has a minimum at $\omega = \theta_k$, contributing a peak in the group delay response of that section at that frequency.

Positivity of the group delay means that the phase is monotonically moving downward as a function of frequency. Since the phase decreases by $2\pi K$ as ω advances by 2π, the area under the group delay curve for a frequency interval spanning 2π is $2\pi K$. This same result can be stated as: the average group delay of an all-pass filter is K samples. Noting this, each term in the sum in the first line of Eq. (K.27) must have an average value of zero.

K.3.3 Magnitude System Response of an All-Pass Filter

We already know that an all-pass filter has a frequency response which is constant in magnitude. As a generalization of this result, consider the magnitude of $H_{\text{all}}(z)$ where z is not necessarily on the unit circle. This response is the product of first order responses as shown in Eq. (K.22). The magnitude-squared of each first-order section is

$$|H_{\text{all}}^{(k)}(z)|^2 = \frac{1 - 2\text{Re}[\alpha_k^* z] + |\alpha_k|^2|z|^2}{|z|^2 - 2\text{Re}[\alpha_k^* z] + |\alpha_k|^2} = \begin{cases} > 1, & |z| > 1, \\ 1, & |z| = 1, \\ < 1, & |z| < 1. \end{cases} \tag{K.28}$$

The overall filter then also obeys these inequalities,

$$|H_{\text{all}}(z)|^2 = \begin{cases} > 1, & |z| > 1, \\ 1, & |z| = 1, \\ < 1, & |z| < 1. \end{cases} \tag{K.29}$$

We note that the magnitude response is constant for z on the unit circle. The magnitude response is *not* constant for z on a circle with non-unity radius.

The magnitude calculated above is that of $H_{\text{all}}(z)H_{\text{all}}^*(z)$. Using the form of the all-pass Eq. (K.23) we find that for an all-pass filter $H_{\text{all}}(z)H_{\text{all}}^*(1/z^*) = 1$ for all z. The two expressions agree for $|z| = 1$.

References

[1] B. S. Atal and M. R. Schroeder, “Predictive Coding of Speech Signals and Subjective Error Criteria”, *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 27, no. 3, pp. 247–254, June 1979.

[2] T. P. Barnwell, “Recursive Windowing for Generating Autocorrelation Coefficients for LPC analysis”, *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 29, no. 5, pp. 1062–1066, Oct. 1981.

[3] D. H. Brandwood, “A Complex Gradient Operator and its Application in Adaptive Array Theory”, *Proc. IEE*, vol. 130, no. 1, pp. 11–16, Feb. 1983.

[4] R. V. Churchhill and J. W. Brown, *Complex Variables and Applications*, 5th ed., McGraw-Hill, 1990 (ISBN 978-0-07-010905-6).

[5] M. H. A. Davis and R. B. Vinter, *Stochastic Modelling and Control*, Chapman and Hall, 1985 (ISBN 978-0-412-16200-8).

[6] P. Delsarte, Y. Genin, and Y. Kamp, “Stability of Linear Predictors and Numerical Range of a Linear Operator”, *IEEE Trans. Inform. Theory*, vol. 33, no. 3, pp. 412–415, May 1987.

[7] A. El-Jaroudi, and J. Makhoul, “Discrete All-Pole Modeling”, *IEEE Trans. Signal Processing*, vol. 39, no. 2, pp. 411–423, Feb. 1991.

[8] T. Ericson, “Structure of Optimum Receiving Filters in Data Transmission Systems”, *IEEE. Trans. Inform. Theory*, vol. 17, no. 3, pp. 352–353, May 1971.

[9] R. D. Gitlin, J. F. Hayes, and S. B. Weinstein, *Data Communications Principles*, Plenum Press, 1992 (ISBN 978-0-306-43777-9).

[10] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Third Edition, Johns Hopkins University Press, 1996 (ISBN 978-0-8018-5414-9).

[11] R. M. Gray, “Toeplitz and Circulant Matrices: A Review”, *Foundations and Trends in Communications and Information Theory*, vol. 2, No. 3, pp. 155-239, 2006 (on-line at `http://ee.stanford.edu/~gray`).

[12] H. Gustafsson, S. E. Nordholm, and I. Claesson, "Spectral Subtraction Using Reduced Delay Convolution and Adaptive Averaging", *IEEE Trans. Speech, Audio Processing*, vol. 9, no. 8, pp. 799–807, Nov. 2001.

[13] S. Haykin, *Adaptive Filter Theory*, Fourth Edition, Prentice-Hall, 2002 (ISBN 978-0-13-090126-2).

[14] T. Islam and P. Kabal, "Partial-Energy Weighted Interpolation of Linear Prediction Coefficients", *Proc. IEEE Workshop Speech Coding* (Delavan, WI), pp. 105–107, Sept. 2000.

[15] ITU-T, "Subjective Performance Evaluation of Telephone-Band and Wideband Digital Codecs", Recommendation P.830, Feb. 1996.

[16] ITU-T, "Artificial Voices", Recommendation P.50, Sept. 1999.

[17] ITU-T, "Software Tools for Speech and Audio Coding Standardization", Recommendation G.191, Sept. 2005.

[18] P. Kabal and E. Dubois, "Interpolating and Nyquist Filters with Constraints at Certain Frequencies", *Signal Processing*, vol 24, no. 2, pp. 117–126, Aug. 1991.

[19] P. Kabal, "Time Windows for Linear Prediction of Speech", Technical Report, Dept. Electrical & Computer Engineering, McGill University, Nov. 2009 (on-line at `http://www-mmsp.ece.mcgill.ca/Documents/Reports`).

[20] P. Kabal, *Ill-Conditioning and Bandwidth Expansion in Linear Prediction of Speech*, Technical Report, Dept. Electrical & Computer Engineering, McGill University, Feb. 2003 (on-line at `www-mmsp.ece.mcgill.ca/Documents/Reports`).

[21] P. Kabal, *ITU-T G.723.1 Speech Coder: A Matlab Implementation*, Technical Report, Dept. Electrical & Computer Engineering, McGill University, Oct. 2009 (on-line at `www-mmsp.ece.mcgill.ca/Documents/Reports`).

[22] P. Kabal, *Minimum-Phase & All-Pass Filters*, Technical Report, Dept. Electrical & Computer Engineering, McGill University, Mar. 2011 (on-line at `www-mmsp.ece.mcgill.ca/Documents/Reports`).

[23] P. Kabal, *Frequency Domain Representations of Sampled and Wrapped Signals*, Technical Report, Dept. Electrical & Computer Engineering, McGill University, Mar. 2011 (on-line at `www-mmsp.ece.mcgill.ca/Documents/Reports`).

[24] P. Kabal and W. B. Kleijn, "All-Pole Modelling of Mixed Excitation Signals", *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing* (Salt Lake City, UT), pp. 97-100, May 2001.

[25] P. Kabal and R. P. Ramachandran, "The Computation of Line Spectral Frequencies Using Chebyshev Polynomials", *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 34, no. 6, pp. 1419–1426, Dec. 1986.

[26] R. W. Lucky, J. Salz, and E. J. Weldon, Jr., *Principles of Data Communications*, McGraw-Hill, 1968 (ISBN 978-0-07-038960-1).

[27] J. Makhoul, “Spectral Linear Prediction: Properties and Applications”, *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 25, no. 5, pp. 423–428, Oct. 1977.

[28] J. D. Markel and A. H. Gray, Jr., *Linear Prediction of Speech*, Springer, 1976 (ISBN 978-387-07563-1).

[29] W. F. G. Mecklenbräuker, “Remarks on the Minimum Phase Property of Optimal Prediction Error Filters and Some Related Questions”, *IEEE Signal Processing Letters*, vol. 5, no. 4, pp. 87–88, April 1998.

[30] S. J. Orfanidis, *Optimum Signal Processing*, Second Edition, MacMillan, 1988 (ISBN 978-0-02-389380-3).

[31] A. Papoulis, “Predictable Processes and Wold’s Decomposition: A Review”, *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 33, no. 4, pp. 933–938, Aug. 1985.

[32] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, Fourth Edition, McGraw-Hill, 2002 (ISBN 978-0-07-366011-0).

[33] R. P. Ramachandran and P. Kabal, “Pitch Prediction Filters in Speech Coding”, *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 37, no. 4, pp. 467–478, April 1989.

[34] W. Pereira and P. Kabal, “Improved Spectral Tracking Using Interpolated Linear Prediction Coefficients”, *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing* (Orlando, FL), pp. I-262–I-264, May 2002.

[35] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing*, 4th ed., Prentice-Hall, 2007 (ISBN 978-0-13-187374-2).

[36] S. Singhal and B. S. Atal “Improving Performance of Multi-Pulse Coders at Low Bit Rates”, *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing* (San Diego, CA), pp. 9–12, March 1984.

[37] B. J. Skinner, F. M. Ingels, and J. P. Donohoe “Stationary and Cyclostationary Random Process Models”, *Proc. IEEE Southeastcon* (Miami, FL), pp. 450–454, April 1994.

[38] P. Strobach, *Linear Prediction Theory*, Springer-Verlag, 1990 (ISBN 978-0-387-51871-8).

[39] D. J. Thomson “Multiple-Window Spectral Estimates”, contribution in “Highlights of Statistical Signal and Array Processing”, A. Hero, ed., *IEEE Signal Processing Magazine*, vol. 15, no. 5, pp. 30–32, Sept. 1998.

[40] G. Ungerboeck, “Fractional Tap-Spacing Equalizer and Consequences for Clock Recovery in Data Modems”, *IEEE Trans. Commun.*, vol. 24, no. 8, pp. 856–864, Aug. 1976.