CHAPTER 6

CORRELATION-BASED METHODS

This chapter reviews nonlinear system identification techniques that are based on cross-correlations. These methods were developed earliest, are well known, and are still in frequent use even though superior methods are now available in most cases. While of interest themselves, the correlation-based methods presented here are also important as the background needed to appreciate the more recent methods discussed in subsequent chapters.

As with previous chapters, a running example will be used to illustrate points raised in the text; each method will be applied to simulated data from the peripheral auditory model used in Chapter 4. In addition, sample applications of each identification technique to physiological systems will be summarized.

6.1 METHODS FOR FUNCTIONAL EXPANSIONS

This section considers methods for estimating the kernels of functional expansion models. In particular, it develops methods for estimating the kernels of a Wiener series model in both the time and frequency domains.

6.1.1 Lee–Schetzen Cross-Correlation

Wiener's work (Wiener, 1958) provided the basis for using cross-correlation functions to estimate the kernels of functional expansion models of nonlinear systems. His method was designed for continuous systems, and therefore estimated correlation functions using analog computations. Subsequently, Lee and Schetzen modified this cross-correlation approach for use in discrete time (Schetzen, 1961a, 1961b). This method, published in the early 1960s (Lee and Schetzen, 1965), is still widespread despite the development

Identification of Nonlinear Physiological Systems, By David T. Westwick and Robert E. Kearney
ISBN 0-471-27456-9

of more precise techniques, such as Korenberg's fast orthogonal algorithm (Korenberg, 1987) in the late 1980s.

Most derivations of the Lee–Schetzen method are based on the Gram–Schmidt orthogonalization used to develop the Wiener series, as described in Section 4.2. Such derivations can be found in textbooks by Marmarelis and Marmarelis (1978) and Schetzen (1980). In contrast, here the method is developed in the framework of orthogonal polynomials to be consistent with the approaches used for the newer techniques in the next two chapters.

Section 4.2.1 demonstrated that the Wiener series may be written as a sum of Hermite polynomial terms. Thus, the output of a second-order Wiener series model can be written in two equivalent forms: one in terms of the Wiener kernels, $\mathbf{k^{(q)}}$:

$$y(t) = \mathbf{k^{(0)}} + \sum_{\tau=0}^{T-1} \mathbf{k^{(1)}}(\tau) u(t-\tau) + \sum_{\tau_1,\tau_2=0}^{T-1} \mathbf{k^{(2)}}(\tau_1, \tau_2) u(t-\tau_1) u(t-\tau_2) - \sigma_u^2 \sum_{\tau=0}^{T-1} \mathbf{k^{(2)}}(\tau, \tau) \tag{6.1}$$

the other in terms of normalized Hermite polynomials, $\mathcal{H}_{\mathcal{N}}^{(q)}$, and their coefficients, $\gamma^{(q)}$:

$$y(t) = \sum_{q=0}^{2} \sum_{\tau_1=0}^{T-1} \cdots \sum_{\tau_q=\tau_{q-1}}^{T-1} \gamma_{\tau_1,\ldots,\tau_q}^{(q)} \mathcal{H}_{\mathcal{N}}^{(q)}(u(t-\tau_1), \ldots, u(t-\tau_q)) \tag{6.2}$$

These two expressions are equivalent, so the Wiener kernels may be transformed into the corresponding polynomial coefficients and vice versa. To accomplish this, recall the development in Section 4.2.1, and equate the zero-, first-, and second-order terms in equations (6.1) and (6.2) to give

$$\mathbf{k^{(0)}} = \gamma^{(0)} \tag{6.3}$$

$$\mathbf{k^{(1)}}(\tau) = \gamma_{\tau}^{(1)} \qquad \text{for } 0 \le \tau < T \tag{6.4}$$

$$\mathbf{k^{(2)}}(\tau, \tau) = \gamma_{\tau,\tau}^{(2)} \qquad \text{for } 0 \le \tau < T \tag{6.5}$$

$$\mathbf{k^{(2)}}(\tau_1, \tau_2) = \frac{\gamma_{\tau_1,\tau_2}^{(2)}}{2} \qquad \text{for } 0 \le \tau_1 < \tau_2 < T \tag{6.6}$$

Kernel values for $\tau_2 < \tau_1$ are obtained based on the symmetry of the second-order Wiener kernel

$$\mathbf{k^{(2)}}(\tau_1, \tau_2) = \mathbf{k^{(2)}}(\tau_2, \tau_1)$$

Thus, estimates of the Wiener kernels may be constructed from estimates of the polynomial coefficients, γ.

Estimates of the polynomial coefficients may be obtained readily using least-squares regression as described in Section 2.4.1.1. Thus, the polynomial coefficients, γ, and

therefore the corresponding Wiener kernels, can be estimated by forming the regression matrix,

$$\mathbf{X}(t,:) = \Big[\mathcal{H}_{\mathcal{N}}^{(0)}\ \mathcal{H}_{\mathcal{N}}^{(1)}(u(t))\ \ldots\ \mathcal{H}_{\mathcal{N}}^{(1)}(u(t-T+1)) \\ \mathcal{H}_{\mathcal{N}}^{(2)}(u(t),u(t))\ \mathcal{H}_{\mathcal{N}}^{(2)}(u(t),u(t-1))\ \ldots\Big] \tag{6.7}$$

Rewriting (6.2) in matrix notation gives

$$\mathbf{y} = \mathbf{X}\boldsymbol{\theta} \tag{6.8}$$

where

$$\boldsymbol{\theta} = \Big[\gamma^{(0)}\ \gamma_0^{(1)}\ \cdots\ \gamma_{T-1}^{(1)}\ \gamma_{0,0}^{(2)}\ \gamma_{0,1}^{(2)}\ \cdots \\ \gamma_{0,T-1}^{(2)}\ \gamma_{1,1}^{(2)}\ \cdots\ \gamma_{T-1,T-1}^{(2)}\Big]^T \tag{6.9}$$

Solving the normal equations

$$\hat{\boldsymbol{\theta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{6.10}$$

provides estimates of the polynomial coefficients.

Most of the computations in the linear regression are associated with forming and inverting the Hessian. These can be avoided if the input is an infinitely long white Gaussian noise sequence since the columns of the regression matrix, $\mathbf{X}$, will be exactly orthogonal. Consequently, the Hessian, $\mathbf{X}^T\mathbf{X}$, will be diagonal and can be inverted simply by inverting the diagonal elements individually. Thus, the kth entry in the parameter vector would be

$$\hat{\theta}_k = \frac{\mathbf{X}(:,k)^T\mathbf{y}}{\mathbf{X}(:,k)^T\mathbf{X}(:,k)}$$

The polynomial coefficients may be estimated sequentially, without matrix inversion. For example, the first element of the estimated parameter vector is given by

$$\hat{\theta}_1 = \frac{\mathbf{X}(:,1)^T\mathbf{y}}{\mathbf{X}(:,1)^T\mathbf{X}(:,1)}$$

From equation (6.7), the first column of $\mathbf{X}$ contains the output of the zero-order Hermite polynomial, which is the constant 1. Thus,

$$\mathbf{X}(t,1) = \mathcal{H}_{\mathcal{N}}^{(0)} = 1$$

so the estimate becomes

$$\hat{\theta}_1 = \frac{\sum_{t=1}^{N} y(t)}{\sum_{t=1}^{N} 1} = \mu_y$$

Thus, the first element in the parameter vector, $\hat{\theta}_1$, is the output mean. It is also the coefficient of the zero-order Hermite polynomial term, and hence the zero-order Wiener kernel,

$$\hat{\mathbf{k}}^{(\mathbf{0})} = \mu_y$$

Similarly, the elements of the first-order Wiener kernel may be read directly from the parameter vector. From equation (6.4), the elements of the first-order Wiener kernel, $\mathbf{k}^{(\mathbf{1})}(\tau)$, are equivalent to the coefficients of the first-order Hermite polynomials, $\gamma_\tau^{(1)}$. These polynomial coefficients are in elements 2 through $T+1$ of the parameter vector, defined in equation (6.9). Thus,

$$\hat{\mathbf{k}}^{(\mathbf{1})} = \left[\hat{\theta}_2\, \hat{\theta}_3 \ldots \hat{\theta}_{T+1}\right]$$

Had the linear regression been solved explicitly, the first-order Wiener kernel could have been read directly from coefficients 2 through $T+1$ of the estimated parameter vector. However, as described above, the Hessian is diagonal so this explicit computation is unnecessary.

Consider the estimate of the value at lag τ in the first-order Wiener kernel, corresponding to element $\tau+2$ of the parameter vector, $\hat{\theta}_{\tau+2}$. Since the Hessian is diagonal, this is given by

$$\hat{\theta}_{\tau+2} = \frac{\mathbf{X}(:, \tau+2)^T \mathbf{y}}{\mathbf{X}(:, \tau+2)^T \mathbf{X}(:, \tau+2)} \tag{6.11}$$

Writing the sums explicitly and substituting $\mathbf{X}(t, \tau+2) = u(t-\tau)$ gives

$$\hat{\theta}_{\tau+2} = \frac{\sum_{t=1}^{N} u(t-\tau) y(t)}{\sum_{t=1}^{N} u^2(t-\tau)} \tag{6.12}$$

Note that the numerator is N times the biased estimate of the input–output cross-correlation. Therefore,

$$\hat{\mathbf{k}}^{(\mathbf{1})}(\tau) = \hat{\theta}_{\tau+2} = \frac{1}{\sigma_u^2} \hat{\phi}_{uy}(\tau) \tag{6.13}$$

and the first-order Wiener kernel estimate is simply the input–output cross-correlation, divided by the input variance.

Theoretically, with infinite data the Hessian will be diagonal and the columns of $\mathbf{X}$ corresponding to the first-order kernel elements will be orthogonal to all remaining columns. In particular, they will be orthogonal to the first column, which generates the output of the zero-order kernel and contains all ones [see equation (6.7)]. Thus, equation (6.13) could be used to estimate the first-order kernel.

In practice, record lengths are finite so the columns of the regression matrix are not perfectly orthogonal. Consequently, the projection of the zero-order kernel onto the first-order kernel will not be exactly zero and will therefore appear as an error in the estimate of the first-order kernel. This error can be eliminated by subtracting the output of the

zero-order kernel, μ_y, from the output before the correlation is computed. Therefore, the first-order kernel is usually estimated from

$$\hat{\mathbf{k}}^{(\mathbf{1})}(\tau) = \frac{1}{\sigma_u^2}\hat{\phi}_{uv^{(0)}}(\tau) \tag{6.14}$$

where $v^{(0)}(t)$ contains the residue that remains after removing the contribution of the zero-order kernel from the output. Thus,

$$\begin{aligned} v^{(0)}(t) &= y(t) - \hat{\mathbf{k}}^{(\mathbf{0})} \\ &= y(t) - \mu_y \end{aligned} \tag{6.15}$$

Next, consider how to estimate the elements of the second-order kernel. In principle, the linear regression, defined by equations (6.7)–(6.10), could be used to estimate the polynomial coefficients, γ, and the second-order kernel estimate could be generated from equations (6.5) and (6.6). However, as with the zero- and first-order kernels, the linear regression may be replaced by a much "cheaper," but theoretically equivalent, cross-correlation function computation.

It is evident from equations (6.5) and (6.6) that the second-order Wiener kernel is determined by the second-order polynomial coefficients, $\gamma^{(2)}_{\tau_1,\tau_2}$. Equation (6.9) shows that these coefficients are in positions $T+2$ through $\frac{1}{2}(T^2+3T)+1$ of the parameter vector, $\boldsymbol{\theta}$. Thus, a cross-correlation based expression, equivalent to the normal equation solution, must be derived to calculate these parameters. This is more difficult than for the lower-order kernels because the expressions for the single- and two-input second-order Hermite polynomials, which give the diagonal and off-diagonal kernel elements, are different. Consequently, the two cases must be dealt with separately.

The first case comprises the diagonal kernel elements and their corresponding polynomial coefficients. These can be estimated from

$$\begin{aligned} \hat{\mathbf{k}}^{(\mathbf{2})}(\tau,\tau) &= \hat{\gamma}^{(2)}_{\tau,\tau} \\ &= \frac{E[\mathcal{H}^{(2)}_{\mathcal{N}}(u(t-\tau),u(t-\tau))y(t)]}{E\left[\left(\mathcal{H}^{(2)}_{\mathcal{N}}(u(t-\tau),u(t-\tau))\right)^2\right]} \end{aligned} \tag{6.16}$$

where the averages over the (infinitely long) data records have been replaced with expected value operators. Using the definition of the second-order Hermite polynomial, the numerator can be expanded as

$$\begin{aligned} E[\mathcal{H}^{(2)}_{\mathcal{N}}(u(t-\tau),u(t-\tau))y(t)] &= E[(u^2(t-\tau)-\sigma_u^2)y(t)] \\ &= E[u^2(t-\tau)y(t)] - \sigma_u^2\mu_y \end{aligned} \tag{6.17}$$

but $u(t)$ is zero mean and stationary, so $\sigma_u^2 = E[u^2(t)] = E[u^2(t-\tau)]$. Thus, the numerator can be written as

$$\begin{aligned} E[u^2(t-\tau)y(t)] - E[u^2(t-\tau)\mu_y] &= E[u^2(t-\tau)(y(t)-\mu_y)] \\ &= E[u^2(t-\tau)v^{(0)}(t)] \end{aligned} \tag{6.18}$$

which is simply the second-order cross-correlation between the input and the zero-order residue, $v^{(0)}(t)$.

The denominator of equation (6.16) can be simplified by multiplying it out and using the properties of products of Gaussian random variables, defined in equation (2.2), to give

$$E[u^4(t-\tau) - 2\sigma_u^2(t-\tau) + \sigma_u^4] = 2\sigma_u^4 \tag{6.19}$$

The estimate in equation (6.16), of the diagonal kernel elements, is the ratio of equation (6.18) and (6.19). Thus

$$\hat{\mathbf{k}}^{(2)}(\tau,\tau) = \frac{1}{2\sigma_u^4}\phi_{uuv^{(0)}}(\tau,\tau) \tag{6.20}$$

The second case to consider deals with the off-diagonal kernel elements and involves estimating the coefficients of the two-input polynomial given by

$$\begin{aligned}\hat{\mathbf{k}}^{(2)}(\tau_1,\tau_2) &= \hat{\gamma}^{(2)}_{\tau_1,\tau_2} \\ &= \frac{E[\mathcal{H}^{(2)}_{\mathcal{N}}(u(t-\tau_1),u(t-\tau_2))y(t)]}{E\left[(\mathcal{H}^{(2)}_{\mathcal{N}}(u(t-\tau_1),u(t-\tau_2)))^2\right]}\end{aligned} \tag{6.21}$$

Expanding the second-order Hermite polynomial, the numerator of equation (6.21) becomes

$$\begin{aligned}E[\mathcal{H}^{(2)}_{\mathcal{N}}(u(t-\tau),u(t-\tau))y(t)] &= E[(u(t-\tau_1)u(t-\tau_2)y(t)] \\ &= \phi_{uuy}(\tau_1,\tau_2)\end{aligned}$$

but the off-diagonal elements of $\phi_{uuy}(\tau_1,\tau_2)$ and $\phi_{uuv^{(0)}}(\tau_1,\tau_2)$ are equal, since

$$\begin{aligned}E[(u(t-\tau_1)u(t-\tau_2)\mu_y] &= \mu_y E[(u(t-\tau_1)u(t-\tau_2)] \\ &= 0 \qquad \text{for } \tau_1 \neq \tau_2\end{aligned} \tag{6.22}$$

Thus, the numerator of equation (6.21) is equal to $\phi_{uuv^{(0)}}(\tau_1,\tau_2)$, which for $\tau_1 = \tau_2$ is the same as the numerator of the diagonal kernel elements.

Expanding the Hermite polynomials, the denominator of equation (6.21) is

$$\begin{aligned}E[u^2(t-\tau_1)u^2(t-\tau_2)] &= E[u^2(t-\tau_1)]E[u^2(t-\tau_2)] + 2E[u(t-\tau_1)u(t-\tau_2)]^2 \\ &= \sigma_u^4 \qquad (\tau_1 \neq \tau_2)\end{aligned}$$

which is twice the value obtained for the diagonal elements. Thus, for $\tau_1 < \tau_2$ we have

$$\gamma^{(2)}_{\tau_1,\tau_2} = \frac{\phi_{uuy_0}(\tau_1,\tau_2)}{\sigma_u^4} \tag{6.23}$$

The off-diagonal kernel elements can be recovered using (6.6):

$$\hat{\mathbf{k}}^{(2)}(\tau_1,\tau_2) = \frac{\gamma^{(2)}_{\tau_1,\tau_2}}{2} = \frac{\phi_{uuv^{(0)}}(\tau_1,\tau_2)}{2\sigma_u^4} \tag{6.24}$$

which is exactly the same as equation (6.20), the expression used to estimate the diagonal kernel values. Since finite length records must be used, the accuracy can be improved by removing the output of the first-order kernel (which is theoretically exactly orthogonal to the output of the second-order kernel) prior to computing the cross correlation. Thus, the second-order kernel estimate is

$$\hat{\mathbf{k}}^{(\mathbf{2})}(\tau_1, \tau_2) = \frac{1}{2\sigma_u^4}\phi_{uuv^{(1)}}(\tau_1, \tau_2) \tag{6.25}$$

where $v^{(1)}(t)$ is the residue remaining after removing the outputs of the first two Wiener kernels from $y(t)$.

$$v^{(1)}(t) = y(t) - \hat{\mathbf{k}}^{(\mathbf{0})} - \sum_{\tau=0}^{T-1} \hat{\mathbf{k}}^{(\mathbf{1})}(\tau)u(t-\tau) \tag{6.26}$$

Similar arguments show that the order q Wiener kernel may be estimated from the order q cross-correlation between the input, $u(t)$, and $v^{(q-1)}(t)$, the residue remaining after removing the outputs of kernels $0 \ldots q-1$ from the output.

6.1.1.1 The Lee–Schetzen Cross-Correlation Algorithm Lee and Schetzen first proposed using cross-correlations to obtain least-squares estimates of the kernels in a series of MIT technical reports (Schetzen, 1961a, 1961b); the approach was disseminated more widely in Lee and Schetzen (1965). The algorithm proceeds as follows:

1. Estimate the order-zero kernel as the mean of the output,

$$\hat{\mathbf{k}}^{(\mathbf{0})} = \frac{1}{N}\sum_{t=1}^{N} y(t) \tag{6.27}$$

2. Subtract $\hat{\mathbf{k}}^{(\mathbf{0})}$ from $y(t)$ to give the zero-order residue,

$$v^{(0)}(t) = y(t) - \hat{\mathbf{k}}^{(\mathbf{0})}$$

3. Estimate the first-order Wiener kernel as the first-order cross-correlation between $u(t)$ and $v^{(0)}(t)$,

$$\hat{\mathbf{k}}^{(\mathbf{1})}(\tau) = \frac{1}{N\sigma_u^2}\sum_{t=1}^{N} u(t-\tau)v^{(0)}(t) \tag{6.28}$$

4. Compute the output of the first-order Wiener kernel by convolution,

$$\hat{y}^{(1)}(t) = \sum_{\tau=0}^{T-1} \hat{\mathbf{k}}^{(\mathbf{1})}(\tau)u(t-\tau)$$

5. Compute the first-order residue,

$$v^{(1)}(t) = v^{(0)}(t) - \hat{y}^{(1)}(t)$$

6. Estimate the second-order Wiener kernel as the second-order cross-correlation between $u(t)$ and $v^{(1)}(t)$, normalized by $2\sigma_u^4$.

$$\hat{\mathbf{k}}^{(2)}(\tau_1, \tau_2) = \frac{1}{2N\sigma_u^4} \sum_{t=1}^{N} u(t - \tau_1)u(t - \tau_2)v^{(1)}(t) \tag{6.29}$$

7. Estimate higher-order kernels similarly. The order q Wiener kernel estimate is the order q cross-correlation between the input, $u(t)$, and the residue remaining after removing the outputs of kernels $0 \ldots q - 1$ from the output, divided by $q!\sigma_u^{2q}$.

Note that this method does not solve the regression explicitly. Rather, it relies on the statistical properties of the input to create an orthogonal expansion that dramatically reduces the number of computations needed to solve the regression. The accuracy of the method depends critically on how well the input approximates an ideal white Gaussian signal.

6.1.1.2 Example: Wiener Kernel Estimates of the Peripheral Auditory Model

This section presents results of simulations that used the Lee–Schetzen cross-correlation algorithm to estimate the Wiener kernels of the peripheral auditory signal processing model, the system used as a running example in Chapter 4. The model is a LNL system consisting of two identical bandpass filters, separated by a half-wave rectifier, as shown in Figure 4.1. The model's first- and second-order Wiener kernels, orthogonalized for two different power levels, are shown in Figure 4.4.

To estimate the Wiener kernels, the model must be driven by a Gaussian white noise signal. This poses problems in discrete-time simulations of nonlinear systems since the nonlinearities may generate components at frequencies above the Nyquist rate. For example, the output of the second-order kernel can contain power between DC and twice the input bandwidth, as shown in equation (6.40), below. Thus, care must be taken to avoid aliasing of these components in discrete-time simulations.

Consequently, for this simulation, the input signal was upsampled to ensure that all computations were performed at a sampling rate much greater than the highest input frequency, to avoid aliasing. The nonlinearity in the peripheral auditory model was represented by an eighth-order polynomial; consequently, the output bandwidth could be at most eight times the input bandwidth. Therefore, the input signal was resampled at 10 times the original sampling rate to ensure that all simulated signals were adequately sampled and that no aliasing would occur. The simulated output was then anti-alias filtered and down-sampled to the original sampling rate. The same anti-aliasing operation was applied to the input signal to ensure that the effects of the anti-aliasing filtering were the same for both input and output.

Two statistically similar data sets were generated: one for identification and one for cross-validation. Both were 100-ms records, sampled at 10 kHz, with white Gaussian inputs having a standard deviation $\sigma_u = 2$. An independent, Gaussian white noise was added to the simulation outputs, after downsampling, to give an output SNR of 10 dB. Figure 6.1 shows 15-ms segments of input–output data from this simulation.

The zero-order Wiener kernel estimate was the output mean, 0.104 in this case. This was subtracted from $z(t)$, to produce the zero-order residue: $v^{(0)}(t)$. Then, the first-order Wiener kernel estimate was computed using equation (6.28); a 16-lag (0–1.5 ms) cross-correlation was computed between $u(t)$ and $v^{(0)}(t)$ and was divided by the input variance.

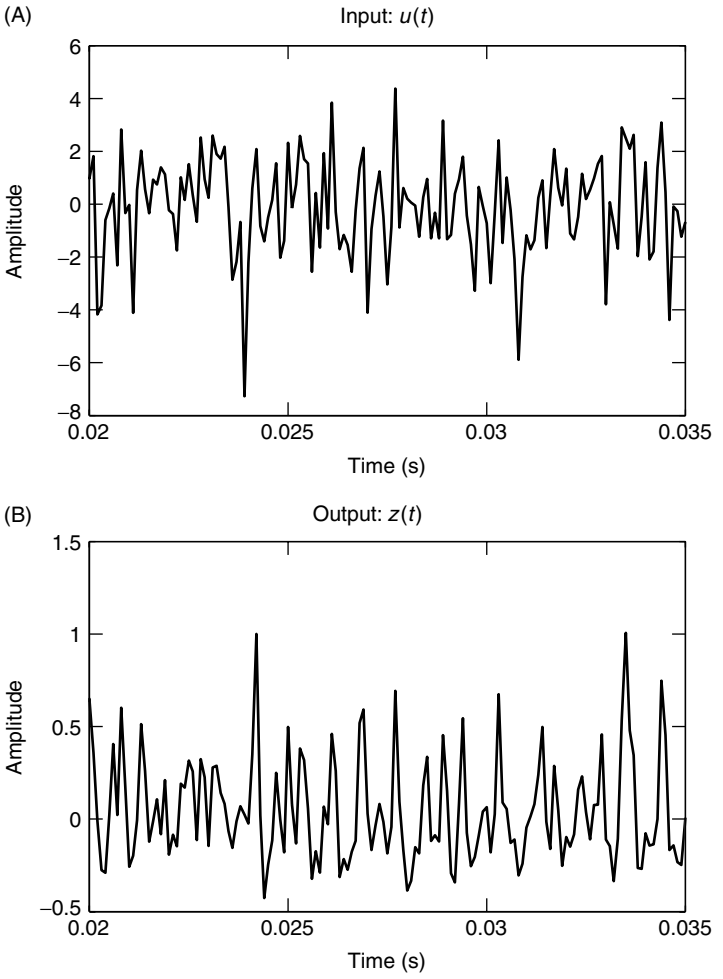

Figure 6.1 Segment of identification data simulated from the peripheral auditory processing model. (A) White Gaussian noise input. (B) Output containing 10 dB of additive white Gaussian measurement noise.

Figure 6.2A shows this kernel estimate; Figure 6.2B shows the zero-order residue, $v^{(0)}(t)$ (solid line), and the output of the first-order Wiener kernel (dashed line). The first-order kernel accounted for 72.4% VAF in the identification data, but only 69.8% VAF in the cross-validation segment.

Next, the first-order residue, $v^{(1)}(t)$, was computed by subtracting the output of the first-order kernel estimate from $v^{(1)}(t)$. This is shown as the solid line in Figure 6.3B. The second-order kernel estimate, Figure 6.3A, was computed from the second-order cross-correlation between $u(t)$ and $v^{(1)}(t)$, as detailed in equation (6.29). This kernel accounted for 56.0% VAF of the first-order residue, raising the in-sample accuracy of the identified model to 87.9%. In the cross-validation data, the second-order kernel accounted for 53.3% of the residual variance, so that the zero- through second-order Wiener kernels accounted for 85.9% VAF.

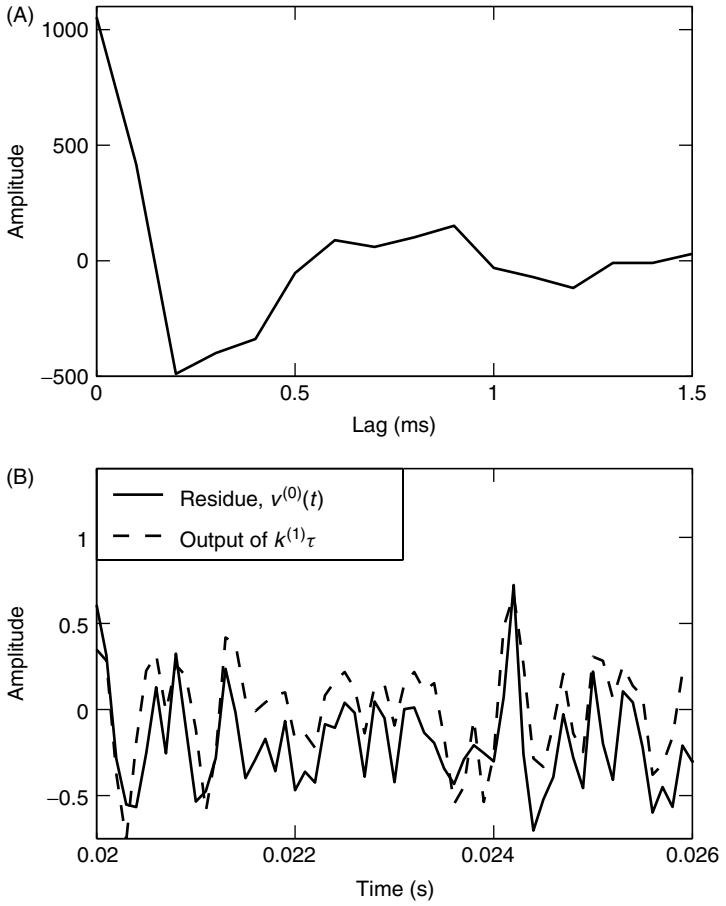

Figure 6.2 Estimate of the first-order Wiener kernel of the peripheral auditory model obtained using Lee–Schetzen cross-correlation. (A) The first-order Wiener kernel estimate. (B) Segments of the zero-order residue (solid line) and the output of the first-order Wiener kernel estimate (dashed line). The first-order Wiener kernel accounted for 69.8% VAF.

6.1.1.3 Application: Early Results from the Catfish Retina The Lee–Schetzen cross-correlation method has been used extensively to study neural systems, where it is often referred to as "white noise analysis." Sakai (1992) and Sakuranaga et al. (1986) provide extensive reviews of neurophysiological applications of the cross-correlation technique.

In early applications, it was common to display the second-order kernel as a contour map, since the technology for producing 3-D perspective plots was not widely available. The figures in this section are reprinted from the original papers and therefore use contour plots. The relationship between these two display formats for second-orders kernels is illustrated in Figure 6.4, which shows a contour plot of a typical second-order kernel and the corresponding 3-D surface plot.

A good example of this work is given in a series of papers by Sakuranaga et al. (Naka et al., 1988; Sakuranaga et al., 1985a, 1985b) that investigated signal encoding

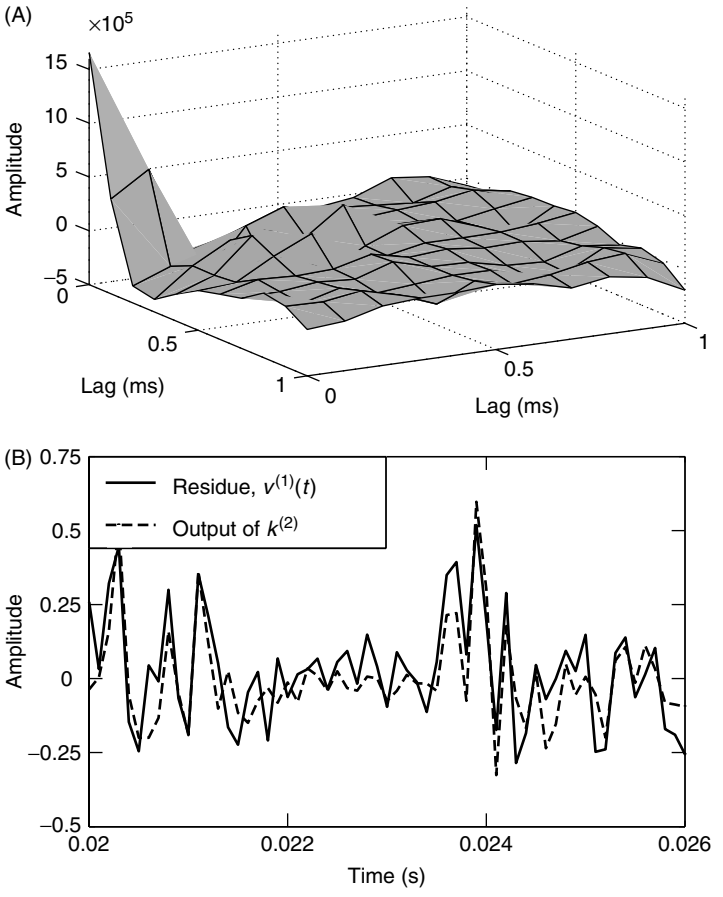

Figure 6.3 Second-order Wiener kernel estimate of the peripheral auditory model obtained with Lee–Schetzen cross-correlation. (A) The second-order Wiener kernel estimate. (B) Segment of the first-order residue (solid line) and the output of the second-order Wiener kernel estimate (dashed line). The second-order Wiener kernel accounted for 53.3% VAF of the first-order residue, raising the overall accuracy of the model to 85.9% VAF.

and transmission in the catfish retina. The retina is a complex neural structure whose operation can be described briefly as follows (Aidely, 1978). Photoreceptors in the retina transduce incident light into electrical activity that then excites the bipolar cells. These in turn excite the ganglion cells, which make up the optic nerve. Interactions between photoreceptors are mediated by two groups of cells: horizontal cells and amacrine cells. Horizontal cells receive input from multiple photoreceptors, and they synapse with several bipolar cells. Amacrine cells, which receive inputs from bipolar cells and from other Amacrine cells, mediate the activity of ganglion cells.

A key division among the amacrine cells concerns their responses to step changes in light intensity. Some act as low-pass filters, producing a sustained response. In the catfish retina, these *sustained amacrine* cells are called *Type-N* amacrine cells. Other amacrine cells produce only a transient response and thus act as high-pass filters. In the catfish retina, these *transient amacrine* cells are known as *Type-C* amacrine cells.

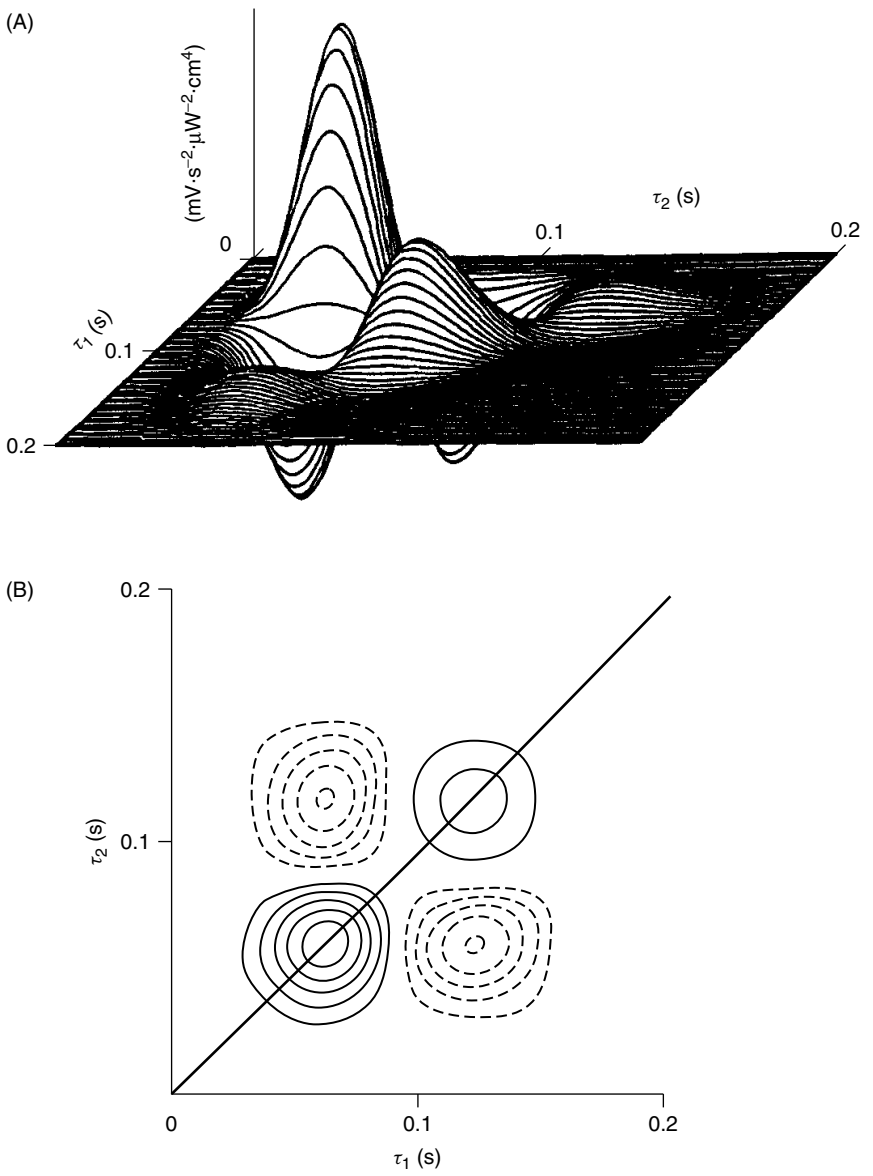

Figure 6.4 Two methods of presenting the second-order kernel. (A) Three-dimensional perspective plot of a second-order kernel. (B) Contour plot of the same kernel. From Naka et al. (1988). Used with permission of BMES.

Experiments were performed using an eye-cup preparation (Baylor et al., 1971; Simon, 1973), where the eyes were removed from their orbits and bisected while the anterior half, containing the lens, was discarded. The vitreous fluid was then drained from the posterior half using tissue paper. The resulting "eye cup" was maintained at room temperature and ventilated with moistened oxygen. Glass micropipette electrodes were advanced through the vitreal surface to record from neurons in the retina.

Outer Retina The first series of experiments (Sakuranaga et al., 1985a) examined signal transmission in the outer retina. Pairs of electrodes were placed approximately 0.4 mm apart, either on the soma and axon of a single horizontal cell or on the somas (or axons) of two separate horizontal cells.

In one set of trials the retina was stimulated with a white-noise-modulated (80-Hz bandwidth) light source, and the resulting voltages at the two electrodes were recorded. The power spectra of this stimulus and response are shown in Figure 6.5B. First- and second-order Wiener kernels were computed between the light input and the voltage outputs using the Lee–Schetzen cross-correlation method. The response of the horizontal cells was found to be almost linear, predicting about 90% VAF of the output voltage. The first-order "light" kernels, illustrated in Figure 6.5A, were low-pass in nature with a bandwidth of about 10 Hz.

In a second set of trials, white noise current, with bandwidth of 80 Hz, was injected into one electrode pair while the resulting voltage change was measured at the other. The

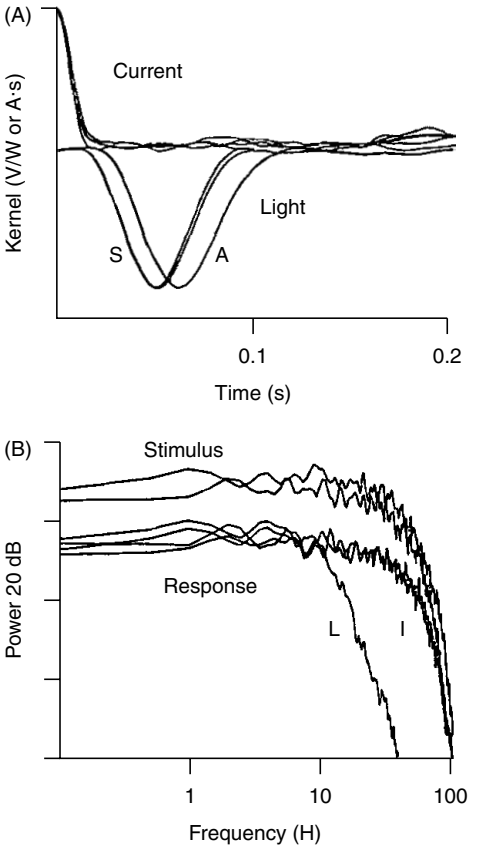

Figure 6.5 White-noise analysis of the horizontal cells in the catfish retina. (A) The first-order Wiener kernels due to light and current inputs. Light kernels labeled "A" were for outputs measured at the cell axon, and those labeled "S" were from the cell soma. (B) The power spectra of the light ("L") and current ("I") inputs and their outputs. From Sakuranaga et al. (1985a). Used with permission of the American Physiological Society.

power spectra of this stimulus and its response are also shown in Figure 6.5B. Wiener kernels computed between the current input and voltage output are shown in Figure 6.5A. This "current" response was also linear, but the bandwidth of the "current" kernel was close to that of the input; this indicates that if the "current" response had any significant dynamics, they occurred at frequencies beyond those interrogated by the input.

Type-C Cells The next set of experiments examined the "light" and "current" responses the Type-C, or transient amacrine, cells (Naka et al., 1988; Sakuranaga et al., 1985c). The response was far from linear for both inputs; the first-order Wiener kernels accounted for less than 20% of the output variance. Adding the second-order kernel increased the prediction accuracy, typically to about 70% VAF.

Figure 6.6A shows a contour plot of the second-order Wiener kernel of the light response for a typical Type-C amacrine cell. The waveforms shown to the left and immediately below the kernel are slices of the kernel taken parallel to horizontal and vertical axes through the kernel's initial positive peak. The horizontal and vertical slices are equal because the kernel is symmetric. The signals plotted on the axis labeled "diagonal cut" are a slice along the kernel diagonal (solid) superimposed on the square of the horizontal slice; evidently, the two traces are very similar.

This behavior led Naka et al. (1988) to conclude that the response of the Type-C cell could be modeled as a Wiener system. To understand the rationale for this, note that equation (4.80) can be used to show that the second-order Wiener kernel of a Wiener

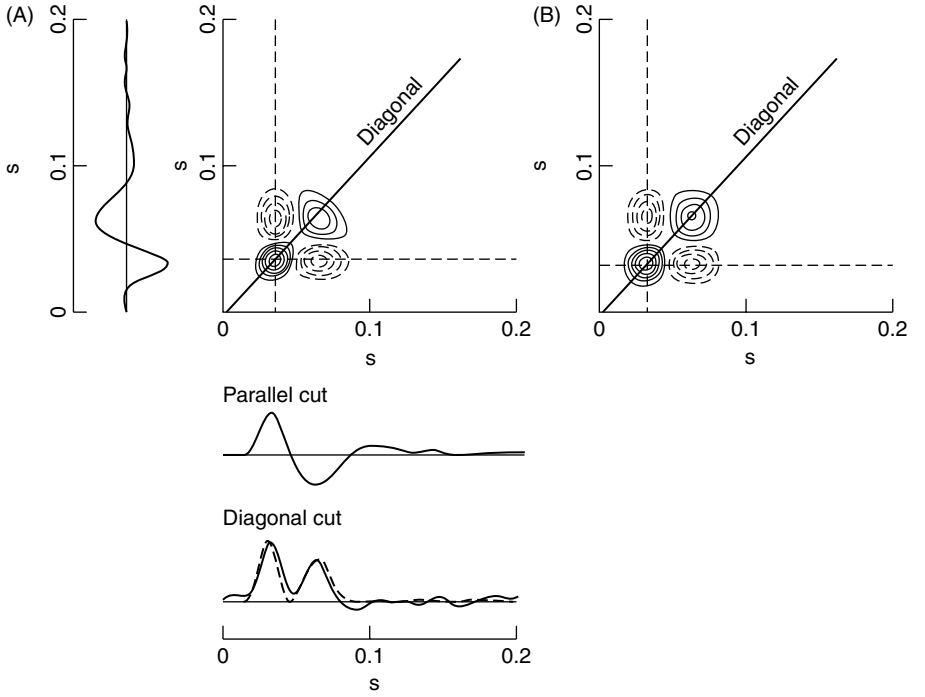

Figure 6.6 Contour plots of second-order kernels. (A) Second-order Wiener kernel estimate for a Type-C cell. (B) Second-order Wiener kernel of a Wiener cascade. From Naka et al. (1988). Used with permission of BMES.

system will have the form

$$\mathbf{k}^{(2)}(\tau_1, \tau_2) = \gamma^{(2)} h(\tau_1) h(\tau_2)$$

Thus a horizontal slice will be

$$\mathbf{k}^{(2)}(\tau, k) = \gamma^{(2)} h(k) h(\tau)$$

and a vertical slice will be

$$\mathbf{k}^{(2)}(k, \tau) = \gamma^{(2)} h(k) h(\tau)$$

That is, the horizontal and vertical slices will be proportional to the IRF of the linear dynamic element and hence to each other. Furthermore, the diagonal element ($\tau_1 = \tau_2$) will be

$$\mathbf{k}^{(2)}(\tau, \tau) = \gamma^{(2)} h(\tau) h(\tau) = c^{(2)} h(\tau)^2$$

which is proportional to the square of the IRF. That is, the horizontal and vertical slices will be proportional to the linear IRF, and the diagonal will proportional to its square.

Further evidence, supporting this conclusion regarding the Type-C cell response, was obtained by computing the second-order Wiener kernel of a Wiener cascade comprising a linear element equal to vertical slice of the second-order kernel, shown at the left of Figure 6.6A, followed by a squarer. The kernel of this simulated system, shown in Figure 6.6B, was very similar to the kernel computed from the experimental data.

Type-N Cells Lastly, the Type-N, or sustained amacrine, cells were studied with the same protocol (Naka et al., 1988; Sakuranaga et al., 1985b) and found to have higher-order nonlinear responses. Although the linear responses for these cells were significant, they usually accounted for no more than 50% VAF. Adding the second-order Wiener kernel increased this by 5–10%; adding the third-order kernel provided an additional 10–20% VAF.

Initially, Sakuranaga and Naka noticed no pattern in the second-order kernels of the Type-N cells (Sakuranaga et al., 1985b). Subsequently, they noted that the structure of the second-order, "light" kernels was consistent with that of an LNL cascade (Naka et al., 1988). Figure 6.7 illustrates the analysis that led to this conclusion. Figure 6.7A shows a typical second-order kernel from a Type-C Amacrine cell, for which the Wiener cascade is thought to be appropriate. Figure 6.7B shows the kernel obtained by convolving this second-order kernel with the impulse response shown between the two kernels. Thus, if the kernel in Figure 6.7A is from a Wiener structure, the kernel in Figure 6.7B is due to an LNL cascade (i.e., a Wiener cascade followed by a linear filter). Figure 6.7C shows the kernel identified from the response of a Type-N cell; it is very similar to that shown in Figure 6.7B. This similarity led Naka et al. (1988) to conclude that the response of a Type-N cell could be represented as an LNL cascade model.

Other Applications The Lee–Schetzen cross-correlation algorithm (Lee and Schetzen, 1965) is still used in neurophysiology (Anzai et al., 1999; Kondoh et al., 1993, 1995; Okuma and Kondoh, 1996; Poliakov et al., 1997; Sakai et al., 1997) where it provides useful structural insights. Nevertheless, as Chapter 7 will show, much better methods have been developed since the Lee–Schetzen algorithm was introduced in 1965.

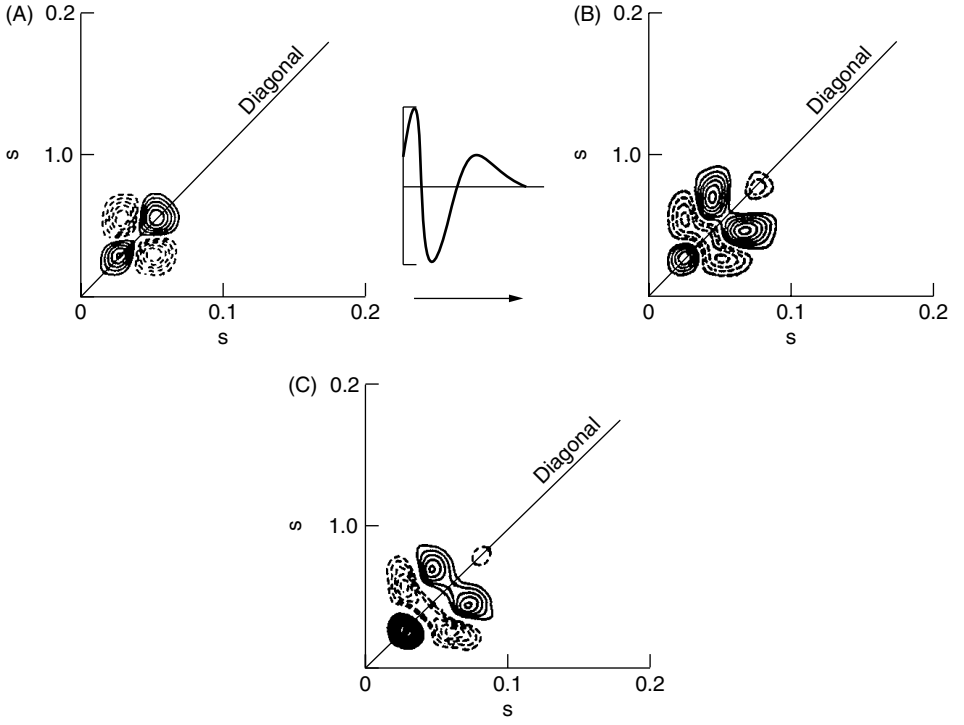

Figure 6.7 Three second-order Wiener kernels. (A) Typical second-order Wiener kernel estimate from a Type-C cell. (B) The Type-C kernel filtered with the impulse response plotted between the panels. (C) Second-order Wiener kernel estimated from a Type-N cell. From Naka et al. (1988). Used with permission of BMES.

6.1.2 Colored Inputs

The Lee–Schetzen method was derived for systems with white inputs. In practice, this will never be the case, although it may be possible to use inputs that are effectively white—that is, whose spectra are flat over the range of frequencies where the system responds. Nevertheless, there are many situations where only inputs with colored spectra can be applied experimentally. In such cases, the terms in the Wiener expansion, equation (4.25), will not be orthogonal; consequently, the Hessian will not be diagonal and the Lee–Schetzen algorithm will give biased results.

To understand the effects of a nonwhite input on the first-order Wiener kernel estimate, consider the first-order cross-correlation:

$$\begin{aligned}\phi_{uy}(\tau) &= E\left[u(t-\tau)y(t)\right] \\ &= E\left[u(t-\tau)\sum_{q=0}^{Q} G_q[\mathbf{k}^{(\mathbf{q})}(\tau_1,\ldots,\tau_m);u(t)]\right]\end{aligned}$$

Note that $u(t-\tau)$ is the output of a first-order Wiener functional and thus it will be orthogonal to the outputs of Wiener functionals of all other orders. Hence, only the G_1

term need be considered. Thus,

$$\begin{aligned}\phi_{uy}(\tau) &= E\left[u(t-\tau)\sum_{j=0}^{T-1}\mathbf{k^{(1)}}(j)u(t-j)\right]\\ &= \sum_{j=0}^{T-1}\mathbf{k^{(1)}}(j)E[u(t-j)u(t-\tau)]\\ &= \sum_{j=0}^{T-1}\mathbf{k^{(1)}}(j)\phi_{uu}(j-\tau)\end{aligned}$$

Therefore, the first-order cross-correlation is no longer equal to the first-order kernel but is equal to the convolution of the input autocorrelation with the first-order Wiener kernel.

Next, consider the effects of a nonwhite input on the second-order Wiener kernel estimate. Recall from Section 4.2.4 that the orthogonalization of the higher-order Wiener kernels changes when the input is nonwhite. Thus, let $y(t)$ be the output of an order Q Wiener series, orthogonalized with respect to a nonwhite input, $u(t)$. As is the usual practice, compute the second-order cross-correlation between the input, $u(t)$, and $v^{(1)}(t)$, the residue remaining after the outputs of the zero- and first-order Wiener kernels have been removed. Thus,

$$\phi_{uuv^{(1)}}(\tau_1, \tau_2) = E\left[u(t-\tau_1)u(t-\tau_2)v^{(1)}(t)\right]$$

Note that $v^{(1)}(t)$ contains terms of order 2 through Q, since the first two terms in the Wiener series have been removed. The product term $u(t-\tau_1)u(t-\tau_2)$ can be expressed as the output of a second-order Wiener operator (or as the sum of a zero-order and second-order Wiener operator, if $\tau_1 = \tau_2$). Therefore, it will be orthogonal to everything in $v^{(1)}(t)$, except the output of the second-order kernel. Thus, if $y^{(2)}(t)$ is the output of the "nonwhite" second-order Wiener functional, given by equation (4.32),

$$y^{(2)}(t) = \sum_{\tau_1,\tau_2=0}^{T-1}\mathbf{k^{(2)}}(\tau_1, \tau_2)u(t-\tau_1)u(t-\tau_2) - \sum_{\tau_1,\tau_2=0}^{T-1}\mathbf{k^{(2)}}(\tau_1, \tau_2)\phi_{uu}(\tau_1-\tau_2)$$

the second-order cross-correlation becomes

$$\begin{aligned}\phi_{uuv^{(1)}}(\tau_1, \tau_2) &= E[u(t-\tau_1)u(t-\tau_2)y^{(2)}(t)]\\ &= \sum_{j_1,j_2=0}^{T-1}\mathbf{k^{(2)}}(j_1, j_2)\Big(E[u(t-\tau_1)u(t-\tau_2)\\ &\qquad \cdot u(t-j_1)u(t-j_2)] - \phi_{uu}(\tau_1-\tau_2)\phi_{uu}(j_1-j_2)\Big)\end{aligned}$$

Since $u(t)$ is Gaussian, the expectation can be expanded as follows:

$$\begin{aligned}E[u(t-\tau_1)u(t-\tau_2)u(t-j_1)u(t-j_2)] = {}&\phi_{uu}(\tau_1-\tau_2)\phi_{uu}(j_1-j_2)\\ &+\phi_{uu}(\tau_1-j_1)\phi_{uu}(\tau_2-j_2)\\ &+\phi_{uu}(\tau_1-j_2)\phi_{uu}(\tau_2-j_1)\end{aligned}$$

Thus,

$$\phi_{uuv^{(1)}}(\tau_1, \tau_2) = 2 \sum_{j_1, j_2=0}^{T-1} \mathbf{k^{(2)}}(j_1, j_2)\phi_{uu}(\tau_1 - j_1)\phi_{uu}(\tau_2 - j_2) \tag{6.30}$$

which shows that the second-order cross-correlation is the two-dimensional convolution of the second-order Wiener kernel with two copies of the input autocorrelation.

The kernel may be obtained, as proposed by Korenberg and Hunter (1990), by deconvolving the input autocorrelation from the rows and columns of the second-order cross-correlation. To do so, first define a matrix, $\mathbf{G}(\tau_1, \tau_2)$, whose columns contain the convolution of the input autocorrelation with the columns of the second-order kernel. Thus,

$$\mathbf{G}(\tau_1, \tau_2) = \sum_{j=0}^{T-1} \mathbf{k^{(2)}}(j, \tau_2)\phi_{uu}(\tau_1 - j), \qquad \tau_1, \tau_2 = 0, \ldots, T-1 \tag{6.31}$$

Then, rewrite equation (6.30) as

$$\phi_{uuv^{(1)}}(\tau_1, \tau_2) = 2 \sum_{j=0}^{T-1} \mathbf{G}(\tau_1, \tau_2)\phi_{uu}(\tau_2 - j) \tag{6.32}$$

Then the jth column of $\mathbf{G}$ can be computed from the deconvolution,

$$\mathbf{G}(:, j) = \tfrac{1}{2}\mathbf{\Phi_{uu}^{-1}}\boldsymbol{\phi}_{\mathbf{uuv^{(1)}}}(:, j) \tag{6.33}$$

where the second-order cross-correlation is shown in boldface to indicate that it is being operated on as a matrix. Multiplying the rows of $\mathbf{G}$ by $\mathbf{\Phi_{uu}^{-1}}$ gives an estimate of $\mathbf{k^{(2)}}(\tau_1, \tau_2)$ and thus solves equation (6.31).

A similar analysis applies to higher-order kernels; the order-q cross-correlation will be the q-dimensional convolution of the order-q kernel with q copies of the input autocorrelation. Estimating the order-q kernel will therefore require multiplying each one-dimensional slice of the order-q cross-correlation, taken parallel to each of the q axes, by the inverse of the autocorrelation matrix.

Section 5.2.3, discussed numerical sensitivity issues surrounding IRF estimation from colored inputs and motivated the development of a pseudo-inverse-based algorithm to stabilize the deconvolution. In estimating an order-q Wiener kernel from data with a colored input, the deconvolution operation must be applied q times. Consequently, the numerical difficulties, outlined in Section 5.2.3, will be compounded.

6.1.2.1 The Repeated Toeplitz Inversion Algorithm

The following algorithm, originally proposed by Korenberg and Hunter (1990), can be used to estimate Wiener kernels of orders 0 through 2 of a nonlinear system subject to nonwhite, Gaussian inputs. The extension to third- and higher-order kernels is straightforward.

1. Compute $\phi_{uu}(\tau)$, the input autocorrelation function, and $\mathbf{\Phi_{uu}^{-1}}$, the inverse of the Toeplitz structured autocorrelation matrix.
2. Estimate the zero-order kernel as the output mean, and compute the zero-order residue, $v^{(0)}(t)$.

3. Use equation (5.10) to estimate the first-order Wiener kernel.

$$\hat{\mathbf{k}}^{(1)}(\tau) = \boldsymbol{\Phi}_{\mathbf{uu}}^{-1}\boldsymbol{\phi}_{\mathbf{uv}^{(0)}} \tag{6.34}$$

and generate the first-order residue

$$v^{(1)}(t) = v^{(0)}(t) - \sum_{\tau=0}^{T-1} \hat{\mathbf{k}}^{(1)}(\tau)u(t-\tau)$$

4. Compute $\phi_{uuv^{(1)}}(\tau_1, \tau_2)$, the second-order cross-correlation between the input and the first-order residue, $v^{(1)}(t)$.
5. The second-order kernel estimate is obtained by multiplying the rows and columns of $\boldsymbol{\phi}_{\mathbf{uuv}^{(1)}}$ by $\boldsymbol{\Phi}_{\mathbf{uu}}^{-1}$:

$$\mathbf{k}^{(2)} = \tfrac{1}{2}\boldsymbol{\Phi}_{\mathbf{uu}}^{-1}\boldsymbol{\phi}_{\mathbf{uuv}^{(1)}}\boldsymbol{\Phi}_{\mathbf{uu}}^{-1} \tag{6.35}$$

6.1.2.2 Example: Wiener Kernel Estimates of the Peripheral Auditory Model Using Colored Inputs To illustrate the effects of a colored input on Wiener kernel estimates, the simulations in the previous section were repeated with the input signals filtered by with a fourth-order, low-pass, Butterworth filter with a cutoff of 3750 Hz (i.e., 0.75 times the Nyquist rate). As before, 10 dB of white Gaussian noise was added to the output.

The first- and second-order Wiener kernels were estimated from 5000 points of input–output data using the repeated Toeplitz inversion method. Figures 6.8A and 6.8C show that the resulting kernel estimates contained substantial high-frequency noise. Nevertheless, they predicted the response well, yielding 89.0% and 78.9% VAF for identification and cross-validation data segments.

The high-frequency noise in kernel estimates can interfere with their interpretation by obscuring important features of the kernel shapes. Indeed, the input used in these simulations was filtered only slightly and the output contained relatively little measurement noise.* Nonetheless, the estimated kernels were very noisy; indeed, it is not possible to determined whether the small peak in the second-order kernel estimate near zero lag in Figure 6.8C is due to the kernel or to estimation noise.

It is common to smooth kernels to suppress estimation noise. Figures 6.8B and 6.8D show the result of smoothing the kernels with a simple three-point, zero-phase filter:

$$h_{smo}(\tau) = \begin{cases} 0.5, & \tau = 0 \\ 0.25, & \tau = \pm 1 \\ 0, & \text{otherwise} \end{cases}$$

It was convolved with the first-order kernel estimate and applied along both the τ_1 and τ_2 directions of the second-order kernel. Smoothing improves the appearance of the kernels, but it also introduces a bias to the kernel estimates that substantially degrades their prediction accuracy. Thus, predictions made with the smoothed kernels accounted for 74.1% and 63.5% in the identification and validation segments, respectively, a decrease of more than 10% in each case. Therefore, even if the kernels are smoothed for presentation, the "raw" kernels should be used for computations.

*Note that the contributions from the third- through eighth-order kernels present in the output will also act as noise when computing the second-order kernel.

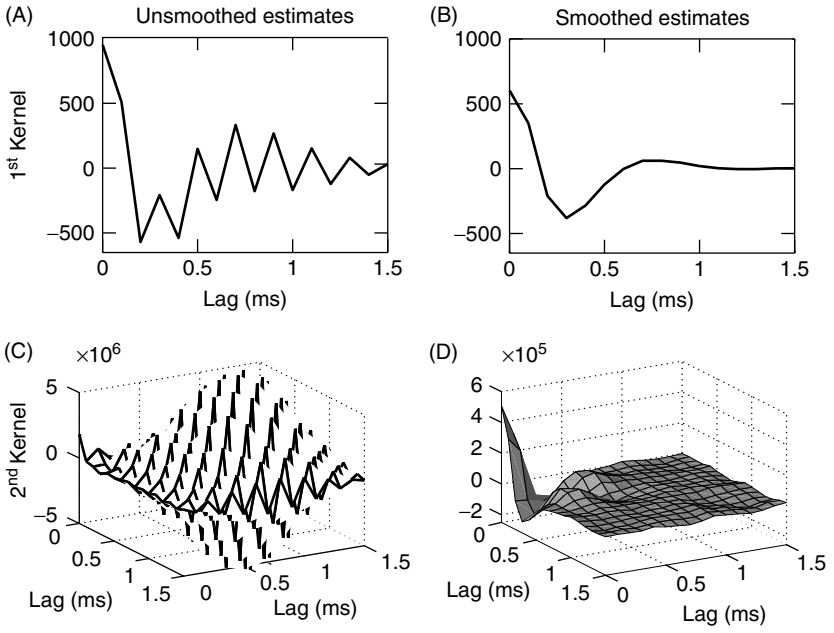

Figure 6.8 Estimates of the first- and second-order Wiener kernels of the peripheral auditory model obtained with a colored input using the repeated Toeplitz inversion algorithm. (A) Raw first-order kernel. (B) Smoothed first-order kernel. (C) Raw second-order kernel. (D) Smoothed second-order kernel. Smoothed estimates were obtained by filtering the raw estimates with a three-point filter. Note the very different z-axis scales for the raw and smoothed second-order kernels in panels C and D.

6.1.3 Frequency Domain Approaches

Section 5.3.2 showed that the cross-power spectrum is the Fourier transform of the cross-correlation. Thus, for a white input, the transfer function of a linear system will be proportional to the input–output cross-spectrum. French and Butz (1973) suggested that higher-order, input–output cross-spectra could be used to estimate the Wiener kernels of nonlinear systems.

To see how, compute the Fourier transform (with respect to lag, τ) of equation (2.18), the biased estimate of the first-order cross-correlation function,

$$\mathfrak{F}\left(\hat{\phi}_{uy}(\tau)\right) = \frac{1}{N}\sum_{\tau=0}^{N}\sum_{i=0}^{N} u(i-\tau)y(i)e^{-j2\pi\tau f/N}$$

Since $e^{-j2\pi(i-i)f/N} = 1$, this simplifies to

$$= \frac{1}{N}\sum_{\tau=0}^{N} u(i-\tau)e^{j2\pi(i-\tau)f/N}\sum_{i=0}^{N} y(i)e^{-j2\pi if/N}$$

$$= \frac{1}{N}U^*(f)Y(f) \tag{6.36}$$

FFT methods calculate the transforms $U(f)$ and $Y(f)$ efficiently; so in practice, cross-correlations are often computed by taking the inverse Fourier transform of equation (6.36).

From equation (6.11) we obtain

$$\hat{\mathbf{k}}^{(1)}(\tau) = \frac{1}{\sigma_u^2}\phi_{uy}(\tau) \tag{6.37}$$

so the cross-spectrum, equation (6.36), divided by σ_u^2, provides a frequency domain estimate of the first-order Wiener kernel.

However, equation (6.36) gives the Fourier transform of the correlation, evaluated between 0 and $N-1$ lags—the length of the data record. This is not desirable since it will produce a highly overparameterized "model" having as many parameters as data points. Moreover, there is no need for such a long correlation function since the system memory will be shorter than the data length. To avoid this, the data record is often segmented into a series of shorter records, the Fourier transform is computed for each segment, and the series of transforms is ensemble-averaged. Thus, if an N point data record is divided into D segments of length N_D, the first-order frequency kernel would be estimated as

$$\hat{\mathbf{K}}^{(1)}(f) = \frac{1}{\sigma_u^2 D N_D}\sum_{d=1}^{D} U_d^*(f)Y_d(f) \tag{6.38}$$

where $U_d(f)$ and $Y_d(f)$ are the Fourier transforms of the dth segment of $u(t)$ and $y(t)$, respectively. Note that the Fourier transforms are taken over lags from 0 to $N_D - 1$. This is sometimes abbreviated as

$$\hat{\mathbf{K}}^{(1)}(f) = \frac{1}{\sigma_u^2 N_D}\langle U^*(f)Y(f)\rangle$$

where the angle brackets denote the ensemble averaging operation. This approach uses the averaged periodogram, described in Section 2.3.5, to estimate the input–output cross-spectrum.

Similar derivations can be used to develop estimates of higher-order Wiener kernels from corresponding high-order cross-spectra. Thus,

$$\hat{\phi}_{uuy}(\tau_1, \tau_2) = \frac{1}{N}\sum_{t=0}^{N} u(t-\tau_1)u(t-\tau_2)y(t) \tag{6.39}$$

is a biased estimate of the second-order cross-correlation (2.27). Taking the two-dimensional Fourier transform with respect to the lag variables, τ_1 and τ_2, gives

$$\mathfrak{F}\left(\hat{\phi}_{uuy}(\tau_1, \tau_2)\right) = \frac{1}{N}\sum_{t=0}^{N}\sum_{\tau_1=0}^{N}\sum_{\tau_2=0}^{N} u(t-\tau_1)u(t-\tau_2)y(t)e^{-j2\pi(\tau_1 f_1+\tau_2 f_2)/N}$$

Multiplying by $e^{-j2\pi(t-t)f_1/N}$ and $e^{-j2\pi(t-t)f_2/N}$, both equal to one, results in

$$\mathfrak{F}\left(\hat{\phi}_{uuy}(\tau_1, \tau_2)\right) = \frac{1}{N}U^*(f_1)U^*(f_2)Y(f_1+f_2) \tag{6.40}$$

As with equation (6.36), equation (6.40) gives the Fourier transform of the kernel, evaluated at lags out to the data length. The resulting frequency domain kernel would be an N by N matrix and clearly have far too many parameters. Consequently, as before, it is common to segment the data and ensemble average their Fourier transforms. The resulting estimate, for segments of length N_D, is

$$\hat{\mathbf{K}}^{(2)}(f_1, f_2) = \frac{1}{2\sigma_u^4 N_D}\langle U^*(f_1)U^*(f_2)Y(f_1 + f_2)\rangle \tag{6.41}$$

Higher-order kernels may be estimated using similar spectral approaches.

This frequency domain implementation of the Lee–Schetzen cross-correlation method (Lee and Schetzen, 1965), as suggested by French and Butz (1973), was intended for use with white inputs. As noted above, for colored inputs, the input autocorrelation must be deconvolved from the cross-correlations to estimate the kernels. This operation is cumbersome for all but the first-order kernel in the time domain. French (1976) recognized that it would be much simpler to perform deconvolution in the frequency domain where time domain deconvolution becomes a frequency domain division. Thus, the frequency kernel estimates can be corrected for nonwhite inputs by dividing by the appropriate input spectrum. The corrected estimates of the frequency kernels are given by

$$\hat{\mathbf{K}}^{(0)} = Y(0) \tag{6.42}$$

$$\hat{\mathbf{K}}^{(1)}(f) = \frac{\langle U^*(f)Y(f)\rangle}{\langle U(f)U^*(f)\rangle} \tag{6.43}$$

$$\hat{\mathbf{K}}^{(2)}(f_1, f_2) = \frac{\langle U^*(f_1)U^*(f_2)Y(f_1 + f_2)\rangle}{2\langle U(f_1)U^*(f_1)\rangle\langle U(f_2)U^*(f_2)\rangle}, \qquad f_1 + f_2 \neq 0 \tag{6.44}$$

where once again the angle brackets denote ensemble averaging.

Note that $\hat{\mu}_y = \mathbf{k}^{(0)} = \mathbf{K}^{(0)} = Y(0)$ and so the zero-order time and frequency domain kernels are identical. Furthermore, it is common practice to subtract the output mean prior to computing the FFTs to increase computational accuracy (French, 1976).

6.1.3.1 Model Validation The original papers by French (1976) and French and Butz (1973) provided no way to assess the accuracy of frequency domain models. One possibility would be to inverse Fourier transform the model output and then use the %VAF, as for time domain models. Alternately, it is possible to compute the coherence between the model output and the measured output (Kearney et al., 1997). This approach will reveal the frequency ranges that are/are not well-modeled by the system. Furthermore, it can be applied kernel by kernel, to reveal which kernels make significant contributions in various frequency ranges.

Note that the coherence between the model output and the measured output does not measure the model accuracy directly. Rather, it provides a measure of how much power in the system output is related linearly to the output predicted by the model, as a function of frequency. That is, it describes how well a linear time-invariant system can model the input–output relation between the observed and predicted outputs, and so it provides a measure of how well the model accounts for nonlinear effects. It is therefore useful to estimate a linear dynamic model between the observed and predicted outputs to determine if there are any unmodeled linear dynamics.

6.1.3.2 Example: Frequency-Domain Kernels of the Peripheral Auditory Model This section demonstrates the estimation of the first- and second-order frequency kernels of the peripheral auditory processing model with a colored input. The input–output data used were the same as those used in the previous section. Five hundred milliseconds (5000 points sampled at 10,000 Hz) of input–output data were used.

The input–output data were divided into 32-point, nonoverlapping segments, resulting in an ensemble of 156 segments of input and output data. These were then detrended, windowed, and Fourier-transformed, using an FFT, and then ensemble-averaged. Kernel estimates were then computed from equations (6.43) and (6.44).

Figure 6.9 illustrates the quantities used to estimate the first-order frequency kernel. Figure 6.9A shows the ensemble estimate of the input autospectrum. Figure 6.9B shows magnitude of the (complex-valued) input–output cross-spectrum, estimated from the ensemble of Fourier-transformed input–output records. Figure 6.9C shows the magnitude of the estimated frequency kernel, computed by dividing the cross-spectrum by the input autospectrum point by point. This is overlayed on the Fourier transform of the system's first-order Wiener kernel. Note the large error above 4000 Hz, where there was little input power.

Although the data segments were 32 points long and yielded 32-point FFTs, the spectra in Figure 6.9 are only 16 points long. This is because the signals used in these computations were real; consequently, the spectra were conjugate symmetric about DC, $S_{uy}(-f) = S^*_{uy}(f)$, and so negative frequency points were redundant and are not shown.

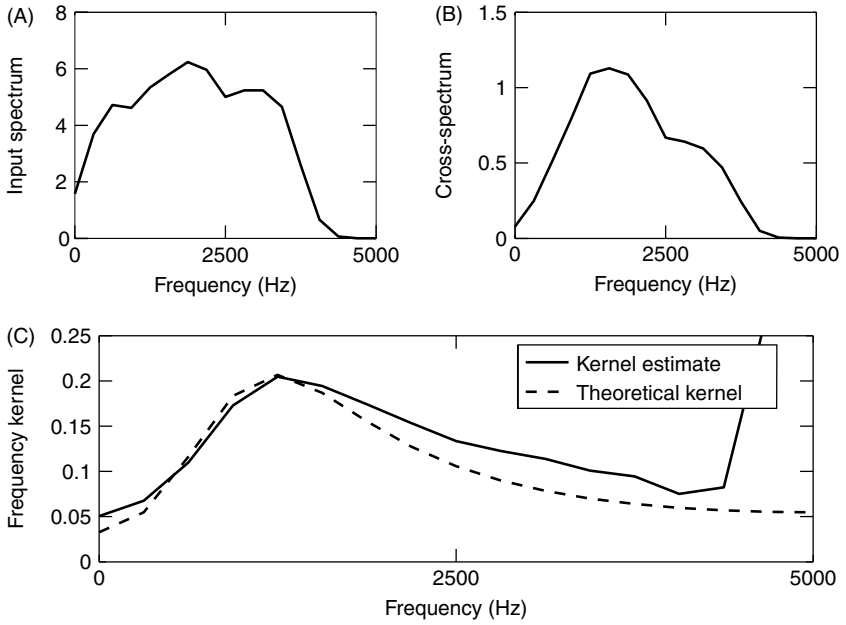

Figure 6.9 Frequency domain estimates of the first-order Wiener kernel of the peripheral auditory model. (A) Magnitude of the input autospectrum. (B) Magnitude of the input–output cross-spectrum. (C) First-order frequency kernel estimate obtained by dividing the cross-spectrum by the input autospectrum (solid line) and the theoretical frequency kernel obtained by Fourier transforming the first-order Wiener kernel.

Figure 6.10 demonstrates the estimation of the second-order frequency kernel. Figure 6.10A shows the square of the input autospectrum, the denominator of equation (6.44). As with the first-order spectra in Figure 6.9, only positive frequencies are shown, since the second-order frequency functions are symmetric with respect to their two arguments, that is,

$$\mathbf{K^{(2)}}(f_1, f_2) = \mathbf{K^{(2)}}(f_2, f_1)$$

and conjugate symmetric about DC,

$$\mathbf{K^{(2)}}(-f_1, f_2) = \mathbf{K^{(2)*}}(f_1, f_2)$$

Thus, the second-order frequency kernel is completely determined by its values in the first quadrant.

Figure 6.10A shows the autospectrum of the input while Figure 6.10B shows the magnitude of the second-order input–output cross-spectrum. Figure 6.10C shows the estimate of the second-order frequency function obtained by dividing the input–output cross-spectrum by the input autospectrum. Figure 6.10D shows the two-dimensional Fourier transform of the system's second-order Wiener kernel. As with the first-order frequency kernel estimate, the second-order kernel contains large errors at frequencies where there is little input power, such as at the two corners (0, 5000 Hz) and (5000 Hz, 0).

The kernel is zero at all points where $f_1 + f_2 \geq 5000$ Hz, the Nyquist frequency. Without this restriction, the second-order kernel could produce an output containing

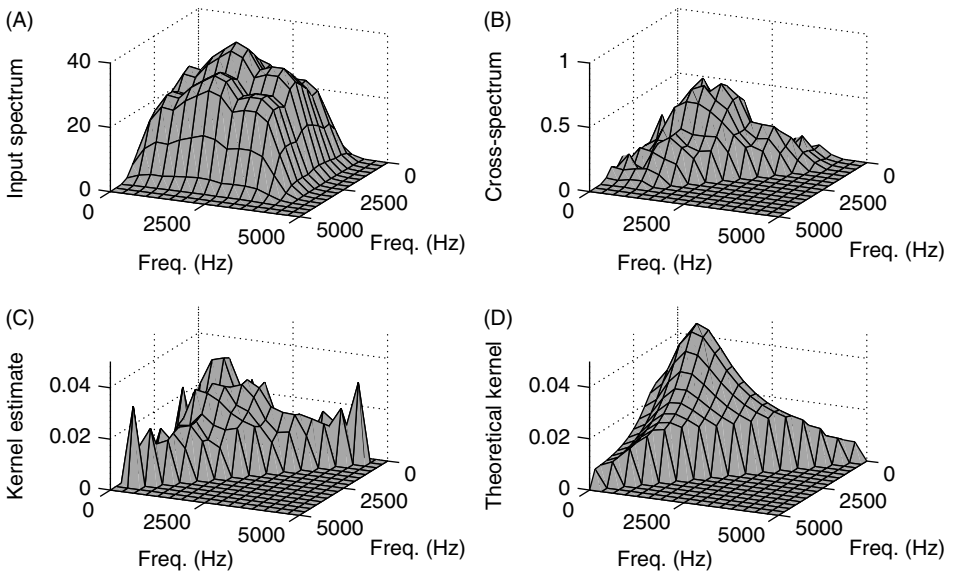

Figure 6.10 Frequency domain estimates of the second-order Wiener kernels of the peripheral auditory model. (A) Magnitude of the product, $S_{uu}(f_1)S_{uu}(f_2)$, the square of the input autospectrum. (B) Magnitude of the second-order input–output cross-spectrum, $S_{uuy}(f_1, f_2)$. (C) Magnitude of the second-order frequency kernel estimate obtained by dividing the spectra shown in A and B. (D) Theoretical frequency kernel obtained by Fourier transforming the second-order Wiener kernel.

frequencies between DC and double the Nyquist frequency, 10,000 Hz in this case. Any components between 5,000 Hz and 10,000 Hz would be aliased, appearing as additional components between 5000 Hz and DC. This is another manifestation of the sampling issues discussed in Section 6.1.1.2.

6.2 BLOCK-STRUCTURED MODELS

Correlation-based algorithms also exist for the identification of block-structured models having the Wiener, Hammerstein, or LNL structures described in Section 4.3. These techniques are based on Bussgang's theorem (Bussgang, 1952), which states the following:

Theorem 6.4 For two Gaussian signals, the cross-correlation function taken after one signal has undergone a nonlinear amplitude distortion is identical, except for a factor of proportionality, to the cross-correlation function taken before the distortion.

To demonstrate this, consider two Gaussian signals $u(t)$ and $x(t)$ with cross-correlation $\phi_{ux}(\tau)$. Let $y(t)$ be the result of applying a polynomial to $x(t)$:

$$y(t) = m(x(t)) = \sum_{q=0}^{Q} c^{(q)} x^q(t) \tag{6.45}$$

The cross-correlation between $u(t)$ and $y(t)$ is then

$$\begin{aligned}\phi_{uy}(\tau) &= E[u(t-\tau)y(t)] \\ &= \sum_{q=0}^{Q} c^{(q)} E[u(t-\tau)x^q(t)]\end{aligned}$$

Both $u(t)$ and $x(t)$ are Gaussian, so using equation (2.3) to evaluate the expectation operation gives

$$\begin{aligned}\phi_{uy}(\tau) &= \sum_{\substack{q=1 \\ q \text{ odd}}}^{Q} c^{(q)}(1 \cdot 3 \cdot \ldots \cdot q)\sigma_x^{q-1}\phi_{ux}(\tau) \\ &= \kappa(\sigma_x, c^{(1)}, \ldots, c^{(Q)})\phi_{ux}(\tau)\end{aligned} \tag{6.46}$$

where

$$\kappa = \sum_{\substack{q=1 \\ q \text{ odd}}}^{Q} c^{(q)}(1 \cdot 3 \cdot \ldots \cdot q)\sigma_x^{q-1}$$

This demonstrates that $\phi_{uy}(\tau)$ is proportional to $\phi_{ux}(\tau)$. Note, however, that unless the polynomial coefficients are either all positive or all negative, there may be some input variances, σ_u^2, for which κ will be zero.

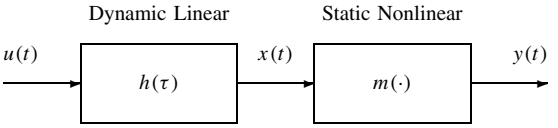

Figure 6.11 Block diagram of a Wiener system.

6.2.1 Wiener Systems

A direct application of Bussgang's theorem arises in the identification of the linear element of a Wiener cascade, shown in Figure 6.11. The output of the Wiener system is given by

$$\begin{aligned} y(t) &= m(x(t)) \\ &= m\left(\sum_{\tau=0}^{T-1} h(\tau)u(t-\tau)\right) \end{aligned}$$

If $u(t)$ is Gaussian, then $x(t)$ will be Gaussian as well. Therefore, by Bussgang's theorem, we have

$$\phi_{uy} = \kappa\phi_{ux}$$

and the IRF of the linear subsystem can be estimated to within a constant of proportionality from

$$\phi_{uy} = \kappa\Phi_{uu}h \tag{6.47}$$

provided that κ is not zero.

6.2.1.1 *Insight from Hermite Polynomials*

It is informative to repeat the foregoing analysis using normalized Hermite polynomials, rather than the power series used in (6.45), to represent the nonlinearity. This gives

$$\begin{aligned} y(t) &= \sum_{q=0}^{Q} \gamma^{(q)}\mathcal{H}_{\mathcal{N}}^{(q)}(x(t)) \\ &= \sum_{q=0}^{Q} \gamma^{(q)}\sigma_x^q\mathcal{H}^{(q)}\left(\frac{x(t)}{\sigma_x}\right) \end{aligned} \tag{6.48}$$

Once the static nonlinearity is represented by a normalized Hermite polynomial, it is straightforward to compute the Wiener kernels of a Wiener cascade. This is because the Wiener system is a special case of the Wiener–Bose model discussed in Section 4.5.4. Recall that the order-q Wiener kernel of a Wiener–Bose model can be computed from the filters in the filter bank and the order-q Hermite polynomial coefficients. Since the Wiener system has only a single filter in the bank, the IRF of the linear element, its first-order Wiener kernel is given by

$$\mathbf{k}^{(\mathbf{1})}(\tau) = \gamma^{(1)}h(\tau) \tag{6.49}$$

where $\gamma^{(1)}$ is the coefficient of the first-order, normalized, Hermite polynomial describing the nonlinearity.

However, from the derivation of the repeated Toeplitz inversion method in equation (6.34), the first-order Wiener kernel is also given by

$$\mathbf{k^{(1)}} = \mathbf{\Phi_{uu}^{-1}} \boldsymbol{\phi}_{\mathbf{uy}}$$

Consequently, the constant of proportionality given in equation (6.47) must be

$$\kappa = \gamma^{(1)}$$

This provides several insights into potential problems with the identification of Wiener systems. First, for an even nonlinearity the first-order Hermite polynomial coefficient, $\gamma^{(1)}$, will be zero by definition. Consequently, the first-order cross-correlation, $\phi_{uy}(\tau)$, will also be zero and the identification will fail.

Second, even if the nonlinearity contains odd terms, the approach will fail if the first-order Hermite coefficient happens to be zero. In such cases, it is useful to change the input power level. This will change the variance of the intermediate signal, σ_x^2, thereby altering the orthogonalization of the Hermite polynomial and giving different coefficients. With an appropriate power level, κ will not be zero and the identification can proceed.

6.2.1.2 Even Nonlinearities For an even nonlinearity, κ will be zero for all input power levels so the first-order correlation cannot be used to identify the linear subsystem's IRF. However, it may be possible to estimate the IRF from the second-order cross-correlation. To see why, consider the second-order cross-correlation, between $u(t)$ and $v^{(0)}(t) = y(t) - \mu_y$, used to estimate the second-order Wiener kernel. Noting that the output of the first-order kernel is zero, equation (6.30) gives

$$\phi_{uuv^{(0)}}(\tau_1, \tau_2) = 2 \sum_{j_1, j_2 = 0}^{T-1} \mathbf{k^{(2)}}(j_1, j_2) \phi_{uu}(\tau_1 - j_1) \phi_{uu}(\tau_2 - j_2)$$

the second-order convolution of the second-order Wiener kernel with two copies of the input autocorrelation. The second-order Wiener kernel of a Wiener system is

$$\mathbf{k^{(2)}}(\tau_1, \tau_2) = \gamma^{(2)} h(\tau_1) h(\tau_2)$$

so that we obtain

$$\phi_{uuv^{(0)}}(\tau_1, \tau_2) = \gamma^{(2)} \phi_{ux}(\tau_1) \phi_{ux}(\tau_2) \tag{6.50}$$

Thus, any nonzero one-dimensional slice of $\phi_{uuv^{(0)}}(\tau_1, \tau_2)$ will be proportional to the cross-correlation measured across the linear element. Deconvolving the input autocorrelation will produce an estimate of the linear IRF, $h(\tau)$.

Alternately, from equation (6.50), it is evident that the second-order Wiener kernel should have only one nonzero eigenvalue and that its eigenvector will be proportional to ϕ_{ux}. Thus, a more robust approach would be to use the principal eigenvector of the whole second-order cross-correlation, rather than simply a one-dimensional slice.

In either case, an unlucky choice of the input power level may result in a nonlinearity whose second-order Hermite polynomial coefficient is zero. Repeating the identification with a different input power level should solve this problem.

6.2.1.3 Fitting the Nonlinearity If a power series is used to represent the nonlinearity, its coefficients may be estimated using a linear regression, as described in Section 2.4.1.1. However, numerical performance can be improved by describing the nonlinearity with an orthogonal polynomial such as the Tchebyshev (Section 2.5.4) or Hermite (Section 2.5.3) polynomials. To do so, first estimate the intermediate signal, $x(t)$, by convolving the IRF estimate, $\hat{h}(\tau)$, with the input, $u(t)$. Then, form a regression matrix appropriate to the polynomial; for an order-Q Tchebyshev polynomial this would be

$$\mathbf{X}(t,:) = \left[1 \; \mathcal{T}^{(1)}(\hat{x}(t)) \; \mathcal{T}^{(2)}(\hat{x}(t)) \; \ldots \; \mathcal{T}^{(Q)}(\hat{x}(t))\right]$$

Finally, use the normal equation (2.33) to solve

$$\mathbf{z} = \mathbf{X}\boldsymbol{\gamma} + \mathbf{v} \tag{6.51}$$

in an MMSE sense, where $\mathbf{z}$ is a vector containing the measured output, $z(t) = y(t)+v(t)$, that contains measurement noise $v(t)$.

6.2.1.4 Iterative Schemes This "one-step" approach to identifying a Wiener system is likely to yield biased estimates of both the IRF and the nonlinearity. To understand why, consider what happens if the static nonlinearity is represented using Hermite polynomials, normalized to the variance of $x(t)$. As argued above, only the first-order Hermite polynomial term will contribute to the first-order input–output cross-correlation. Consequently, when estimating the linear system between the input and output, the contributions from higher-order polynomial terms will be seen as "noise." This "nonlinearity noise" will not be Gaussian and will add to any measurement noise, decreasing the effective signal-to-noise ratio and increasing the variance of the IRF estimate. Furthermore, the estimate of the intermediate signal, $\hat{x}(t)$, used to construct the regressors, $\mathbf{X}$ [see equation (6.51)], is obtained by convolving the input, $u(t)$, with the estimated linear IRF, $\hat{h}(\tau)$. Thus, errors in the IRF estimate will lead to errors in the regression matrix, and consequently the least-squares estimates can be expected to be biased.

Hunter–Korenberg Algorithm Hunter and Korenberg (1986) proposed an iterative solution, which attempts to improve its estimate of the intermediate signal at each stage in the iteration. The algorithm proceeds as follows:

1. Choose the length of the linear filter, and order of the polynomial nonlinearity. Set the iteration number, $k = 0$.
2. Use equation (5.10) to estimate a linear IRF, $\hat{h}_0(\tau)$, between the input, $u(t)$, and output, $z(t)$.
3. Predict the intermediate signal, $\hat{x}_k(t)$, using the convolution:

$$\hat{x}_k(t) = \sum_{\tau=0}^{T-1} \hat{h}_k(\tau) u(t-\tau)$$

4. Fit a polynomial, $\hat{m}_k(\cdot)$, between the estimate of the intermediate signal, $\hat{x}_k(t)$, and the output, $z(t)$, using linear regression (6.51).

5. Compute the output of this model, $\hat{y}_k(t) = \hat{m}_k(\hat{x}_k(t))$, and determine its mean-square error (MSE),

$$V_N(k) = \frac{1}{N}\sum_{t=1}^{N}(z(t) - \hat{y}_k(t))^2$$

6. If $V_N(k) < V_N(k-1)$, the iteration is still converging, so increment k and jump to step 8.
7. If there is no improvement in the MSE, exit and use $\hat{h}_{k-1}(\tau)$ and $\hat{m}_{k-1}(\cdot)$ as estimates for the IRF and static nonlinearity.
8. Estimate the *inverse* of the static nonlinearity, $\hat{m}_k^{-1}(\cdot)$, by using linear regression to fit a polynomial between the output, $z(t)$, and the current estimate of the intermediate signal, $\hat{x}_k(t)$.
9. Construct an alternate estimate of the intermediate signal by transforming the output with the estimated inverse nonlinearity.

$$\hat{x}_{a,k}(t) = \hat{m}_k^{-1}(z(t))$$

10. Compute the next estimate of the linear system, $\hat{h}_k(\tau)$, by fitting a linear system between the input, $u(t)$, and the alternate estimate of the intermediate signal, $\hat{x}_{a,k}(t)$.
11. Go to step 3.

There are several difficulties with this algorithm—mostly related to the inversion of the static nonlinearity required in step 8.

1. First, the nonlinearity must be invertible. This will only be the case when the nonlinearity is a one-to-one function over the range probed by the identification data. Otherwise, the inverse will not exist, and fitting a polynomial between the output and an estimate of the input to the nonlinearity will give unpredictable results.
2. Even if the nonlinearity is invertible, output noise will enter the regression matrix and bias the estimate of the inverse.
3. Output noise will cause problems even if the inverse of the nonlinearity is exactly known. Transforming the measured output with the inverse nonlinearity will generate components where the output noise appears additively, in the first-order terms, and multiplicatively in cross-terms generated by higher-order polynomials. These high-order multiplicative noise terms may bias subsequent estimates of the linear dynamics.

Paulin–Korenberg–Hunter Algorithm Paulin (1993) proposed an approach that avoids inverting the static nonlinearity explicitly. Like the Hunter–Korenberg algorithm, it alternately updates two different estimates of the intermediate signal. The "primary" estimate of the intermediate signal, $\hat{x}_k(t)$, is obtained by filtering the input by the current estimate of the linear dynamic element, $\hat{h}_k(\tau)$. To avoid computing an explicit inverse, Paulin updated the alternate estimate, $\hat{x}_{a,k}(t)$, as follows:

$$\hat{x}_{a,k}(t) = \hat{x}_{a,k-1}(t) + (z(t) - \hat{y}_{k-1}(t)) \tag{6.52}$$

The rationale behind this update can best be explained as follows. The Wiener kernels of the system relating the input, $u(t)$, to the error in the current output, $z(t) - \hat{y}_{k-1}(t)$, will be equal to the errors in the Wiener kernels of the current model. Thus, any error in the linear IRF estimate will lead to a proportional error in the first-order Wiener kernel. Thus, the first-order Wiener kernel between the input and the error in the model output will be proportional to the error in the IRF estimate for the linear element. Hence, using the update given in equation (6.52), and then fitting a linear IRF between $u(t)$ and $\hat{x}_{a,k}(t)$, is equivalent to adding the first-order Wiener kernel of the error model to the current IRF estimate.

This approach has several advantages compared to the direct inversion of the nonlinearity. First, additive noise in $z(t)$ appears as additive noise in $\hat{x}_{a,k}(t)$, since it is never transformed by the inverse of the nonlinearity. Second, since the nonlinearity is never inverted, it need not be invertible. Thus, a wider class of systems may be identified.

Korenberg and Hunter (1999) noted that the iteration proposed by Paulin could become unstable; they suggested that this could be prevented by reducing the size of the correction and by applying the correction to $\hat{x}_{k-1}(t)$, instead of $\hat{x}_{a,k-1}(t)$. This removes the contributions of higher-order terms in the error model, which act as "output noise" in the regression used to estimate the IRF. Thus, Korenberg and Hunter (1999) proposed the update

$$\hat{x}_{a,k}(t) = \hat{x}_{k-1}(t) + \alpha(z(t) - \hat{y}_{k-1}(t)) \tag{6.53}$$

where α, $0 < \alpha \leq 1$, controls the rate of convergence. The algorithm can be stabilized by reducing α. The resulting algorithm proceeds as follows:

1. Choose the length of the linear filter, the order of the polynomial nonlinearity, and a convergence rate, $0 < \alpha \leq 1$. Set the iteration number, $k = 1$.
2. Choose $\hat{x}_{a,0}(t)$ as an initial estimate of the intermediate signal; usually $\hat{x}_{a,0}(t) = z(t)$.
3. Estimate a linear IRF, $\hat{h}_k(\tau)$, between the input, $u(t)$, and the previous estimate, $\hat{x}_{a,k-1}(t)$, of the intermediate signal, using equation (5.10).
4. Produce a "primary" estimate of the intermediate signal using convolution: $\hat{x}_k(t) = \hat{h}_k(\tau) * u(t)$.
5. Fit a polynomial, $\hat{m}_k(\cdot)$, between $\hat{x}_k(t)$ and the output, $z(t)$.
6. Compute the output of this model, $\hat{y}_k(t) = \hat{m}_k(\hat{x}_k(t))$, and the MSE,

$$V_N(k) = \frac{1}{N}\sum_{t=1}^{N}(z(t) - \hat{y}_k(t))^2$$

7. If $V_N(k) > V_N(k-1)$, the error has not improved; exit and use $\hat{h}_{k-1}$ and $\hat{m}_{k-1}$ as the estimated IRF and polynomial. If appropriate, reduce α and restart the identification.
8. If $V_N(k) < V_N(k-1)$, the iteration is still converging, so increment k. Update the alternate estimate of the intermediate signal using equation (6.53).
9. Go to step 3.

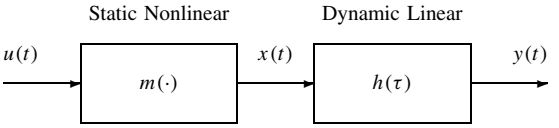

Figure 6.12 Block diagram of a Hammerstein system.

Convergence To our knowledge, none of these algorithms has been proven to converge under all conditions. Characterizing the convergence of these algorithms is a challenging problem that requires first finding the "fixed points" of the iterations—that is, points for which

$$(\hat{h}_{k+1}(\tau), \hat{m}_{k+1}(\cdot)) = (\hat{h}_k(\tau), \hat{m}_k(\cdot))$$

and hence where the algorithm will halt. Once the fixed points have been located, it would then be necessary to characterize their statistics.

6.2.2 Hammerstein Models

Bussgang's theorem may also be used to identify Hammerstein models. Referring to Figure 6.12, the output of a Hammerstein model is given by equation (4.39):

$$y(t) = \sum_{\tau=0}^{T-1} h(\tau) \left\{ \sum_{q=0}^{Q} c^{(q)} u^q(t-\tau) \right\}$$

Thus, the input–output cross-correlation is

$$\phi_{uy}(\tau) = \sum_{j=0}^{T-1} h(j) \left\{ \sum_{q=0}^{Q} c^{(q)} E[u(t-\tau)u^q(t-j)] \right\}$$

Each expectation may be evaluated using equation (2.3), giving

$$\begin{aligned} \phi_{uy}(\tau) &= \sum_{j=0}^{T-1} \phi_{uu}(\tau - j)h(j) \left\{ \sum_{\substack{q=1 \\ q \text{ odd}}}^{Q} c^{(q)}(1 \cdot 3 \cdot \ldots \cdot q)\sigma_y^{q-1} \right\} \\ &= \kappa \sum_{j=0}^{T-1} \phi_{uu}(\tau - j)h(j) \end{aligned}$$

which may be expressed in vector–matrix notation as

$$\boldsymbol{\phi}_{\mathbf{uy}} = \kappa \boldsymbol{\Phi}_{\mathbf{uu}} \mathbf{h}$$

where $\boldsymbol{\Phi}_{\mathbf{uu}}$ is the Toeplitz structured autocorrelation matrix defined in equation (5.9). Thus, the input–output cross-correlation across a Hammerstein cascade is proportional to the cross-correlation taken across the linear subsystem, $h(\tau)$. This provides the means to identify the entire system.

6.2.2.1 Insight from Hermite Polynomials

As with the Wiener system, expanding the nonlinearity using Hermite polynomials normalized with respect to the input variance, σ_u^2, provides insight into the properties of the constant, κ. This expansion gives

$$y(t) = \sum_{\tau=0}^{T-1} h(\tau) \left\{ \sum_{q=0}^{Q} \gamma^{(q)} \mathcal{H}^{(q)}(u(t-\tau)) \right\} \tag{6.54}$$

with the input–output cross-correlation,

$$\phi_{uy}(\tau) = \sum_{j=0}^{T-1} h(j) \left\{ \sum_{q=0}^{Q} \gamma^{(q)} E\left[u(t-\tau) \mathcal{H}^{(q)}(u(t-j)) \right] \right\} \tag{6.55}$$

Since the Hermite polynomials are orthogonal,

$$E\left[u(t-\tau) \mathcal{H}^{(q)}(u(t-j)) \right] = \begin{cases} E[u(t-\tau)u(t-j)], & q = 1 \\ 0, & \text{otherwise} \end{cases}$$

the first-order cross-correlation is

$$\boldsymbol{\phi}_{\mathbf{uy}} = \gamma^{(1)} \boldsymbol{\Phi}_{\mathbf{uu}} \mathbf{h}$$

Thus, the constant of proportionality, κ, will be equal to the first-order Hermite polynomial coefficient, $\gamma^{(1)}$. Provided that $\gamma^{(1)}$ is nonzero, the first-order cross-correlation can be used to estimate the linear IRF. As with Wiener cascade, $\gamma^{(1)}$ may be zero. This may occur either due to an unfortunate choice for the input power level or because the polynomial is even. In the former case, repeating the identification experiment using a different power level should restore identifiability. Alternately, the approach below for even nonlinearities may be applied, provided that the nonlinearity contains at least one even term.

6.2.2.2 Even Nonlinearities

As with the Wiener system, estimates of the linear IRF based on the first-order cross-correlation will fail for even nonlinearities. The IRF estimate for even Hammerstein systems can be derived from the second-order Wiener kernel as was done for the Wiener system. However, for a Hammerstein system, only the diagonal of the kernel need be estimated since

$$\begin{aligned} \phi_{uuv^{(1)}}(\tau, \tau) &= E\left[u^2(t-\tau) \gamma^{(2)} \sum_{j=0}^{T-1} h(j) \mathcal{H}_{\mathcal{N}}^{(2)}(u(t-j)) \right] \\ &= 2\gamma^{(2)} \sigma_u^4 h(\tau) \end{aligned}$$

Again, an unfortunate choice of input power level may result in an orthogonalization for which $\gamma^{(2)} = 0$, in which case the experiment must be repeated with a different input variance.

6.2.2.3 Fitting the Nonlinearity

Once $h(\tau)$ is known, the polynomial coefficients may be estimated using linear regression and the superposition property of linear systems.

Let $\mathbf{U}$ be a matrix whose columns contain polynomials in $u(t)$. For example, for Hermite polynomials the entries in the qth column of $\mathbf{U}$ would be

$$\mathbf{U}(t, q) = \mathcal{H}^{(q-1)}(u(t)) \tag{6.56}$$

The regression matrix is formed by filtering each column of $\mathbf{U}$ with $h(\tau)$, to give

$$\mathbf{W}(t, q) = \sum_{\tau=0}^{T-1} h(\tau)\mathbf{U}(t - \tau, q) \tag{6.57}$$

The polynomial coefficients, γ, are then found from the linear regression

$$\mathbf{z} = \mathbf{W}\boldsymbol{\gamma} + \mathbf{v} \tag{6.58}$$

where $\mathbf{z}$ is a vector containing the measured output, $z(t) = y(t) + v(t)$.

6.2.2.4 Iterative Methods

In theory, the higher-order Hermite polynomial terms in equation (6.54) will be orthogonal to the input. However, in practice, record lengths are finite and input distributions are not perfectly Gaussian, so that the expectations in equation (6.55) will not be identically zero. These nonzero terms will appear as additional output noise, increasing the variance of the correlation estimate and the resulting IRF estimate. These errors will, in turn, generate errors in the regression matrix, $\mathbf{W}$, and bias the estimates of the polynomial coefficients in equation (6.58). As with Wiener systems, iterative methods can be used to reduce the noise in the regression matrix and thus minimize the bias in the polynomial coefficients.

Hunter–Korenberg Iteration for Hammerstein Systems The algorithm proposed by Hunter and Korenberg (1986) uses an iterative approach similar to that employed in the identification of Wiener systems; each iteration attempts to refine two separate estimates of the intermediate signal, $\hat{x}_k(t)$ and $\hat{x}_{a,k}(t)$. The algorithm proceeds as follows:

1. Set the iteration counter, $k = 1$. Estimate a linear system, $\hat{h}_k^{-1}(\tau)$, between the output, $z(t)$, and the input, $u(t)$. This will be proportional to the *inverse* of the linear subsystem and thus will include noncausal components; it must therefore be estimated as a two-sided filter.
2. Compute the "primary" estimate of the intermediate signal, $\hat{x}_k(t)$, with the convolution: $\hat{x}_k(t) = \hat{h}_k^{-1}(\tau) * z(t)$.
3. Fit a polynomial, $\hat{m}_k(\cdot)$, between the input, $u(t)$, and the current estimate of the intermediate signal, $\hat{x}_k(t)$.
4. Predict an "alternate" estimate of the intermediate signal,

$$\hat{x}_{a,k}(t) = \hat{m}_k(u(t))$$

5. Fit a linear system, $\hat{h}_k(\tau)$, between $\hat{x}_{a,k}(t)$ and the output, $z(t)$.
6. Generate the model output, $\hat{y}_k(t) = h_k(\tau) * \hat{x}_{a,k}(t)$, and compute its mean-square error, V_N. Exit if the MSE does not decrease significantly, otherwise continue.
7. Estimate a linear system, $\hat{h}_{k+1}^{-1}(\tau)$, between the output, $z(t)$, and $\hat{x}_{a,k}(t)$, increment the counter, $k = k + 1$, and go to step 2.

6.2.2.5 Convergence The convergence properties of this algorithm have not been established analytically. Westwick and Kearney (2001) used Monte-Carlo simulations to compare the convergence properties of this algorithm to those of an alternate approach using nonlinear optimization (see Chapter 8). They found that the Hunter–Korenberg iteration produced biased results when the identification input was a non-Gaussian and nonwhite signal, even when the output noise level was low. However, for the example studied, the Hunter–Korenberg algorithm produced unbiased estimates of the Hammerstein cascade when the input was either non-Gaussian or nonwhite, but not both.

6.2.2.6 Application: Stretch Reflex Dynamics The *stretch reflex* is the involuntary contraction of a muscle which results from perturbations of its length. Electromyograms from the muscle are often used to assess the activity of a muscle, and so the stretch reflex is often studied as the dynamic relationship between the joint position and the resulting EMG. To identify a stretch reflex model, the joint must be perturbed, with the surrounding joints fixed and the EMG from the relevant muscles measured.

Kearney and Hunter (1983, 1984, 1988) used this approach to examine stretch reflexes in the muscles of the human ankle. Subjects lay supine on a rigid experimental table (Kearney et al., 1983) with their left foot attached to an electrohydraulic actuator by means of a custom-fitted fiberglass boot. They were asked to maintain a constant contraction corresponding to 15% of their maximum voluntary contraction while the actuator, configured as a position servo, produced a pseudo-random position perturbation whose spectrum was flat to 20 Hz and contained significant power up to 50 Hz. The following signals were recorded: joint position, measured by a potentiometer attached to the end of the rotary actuator; joint torque, measured by a strain guage mounted between the actuator and the ankle; and the EMGs over the tibialis anterior (TA) and gastrocnemius–soleus (GS) muscles.

In the first two papers (Kearney and Hunter, 1983, 1984), linear filters were fitted between joint velocities and EMGs. Quasi-linear models were shown to describe the TA stretch reflex quite accurately for each operating point defined by the mean torque, perturbation amplitude, and ankle position. The model parameters did, however, change substantially with the operating point. Quasi-linear models were much less successful for the GS stretch reflex. However, quasi-linear models could be fitted between half-wave rectified velocity and gastrocnemius EMG. This ad hoc approach suggested that a Hammerstein structure might be an appropriate model for the gastrocnemius stretch reflex.

The third paper (Kearney and Hunter, 1988) employed a more systematic approach. Initially, the first- and second-order Wiener kernels between the ankle velocity and gastrocnemius EMG were estimated using the Lee–Schetzen method (Lee and Schetzen, 1965). The first-order kernel accounted for 40% of the EMG variance but the second-order kernel was not significant, accounting for less than 1% of the residual variance, suggesting that higher-order nonlinearities were present. Nevertheless, the form of the second-order kernel gave insight into the potential structure of the system.

Figures 6.13 and 6.14 show the cross-correlation estimates of the first- and second-order Wiener kernels of the stretch reflex EMG, as well as the results of some manipulations used to establish potential model structures. Figure 6.14A shows the second-order kernel viewed orthogonal to the diagonal. It is evident that most of the significant structure occurs at lags where the magnitude of the first-order kernel (Figure 6.13A) is relatively large. Viewing the second-order kernel parallel to its diagonal, as shown in Figure 6.14B, demonstrates that the most significant structure is near the diagonal.

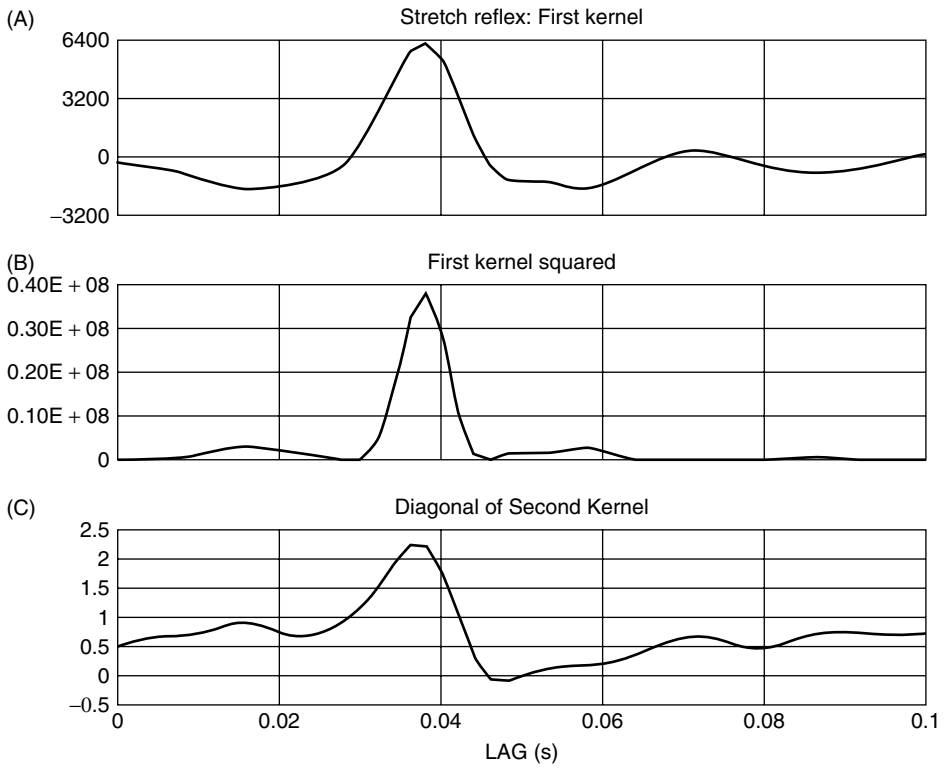

Figure 6.13 Cross-correlation analysis of stretch reflexes. (A) First-order Wiener kernel. (B) The first-order kernel squared. (C) The diagonal of the second-order Wiener kernel. From Kearney and Hunter (1988). Used with permission of BMES.

Compare the first-order kernel (Figure 6.13A) to its square (Figure 6.13B) and to the diagonal of the second-order kernel (Figure 6.13C). If the underlying system were a Hammerstein cascade, the first-order kernel would be expected to be proportional to the diagonal of the second-order kernel. Similarly, if the underlying system were a Wiener cascade, the diagonal of the second-order kernel would be proportional to the square of the first-order kernel. Since the second-order kernel, shown in Figure 6.14, was mostly diagonal, and there was a relatively good correspondence between the first-order kernel and the diagonal of the second-order kernel (Figures 6.13A and 6.13C), the Hammerstein cascade was considered to be an appropriate structure for this system.

Based on these results, a Hammerstein model was identified using the iterative procedure of Section 6.2.2 (Hunter and Korenberg, 1986). The algorithm converged in less than six iterations, resulting in a model which accounted for 62% VAF, as compared to 40% VAF for the linear model. Figure 6.15 shows the elements of the identified Hammerstein model. Figure 6.15A shows the nonlinearity, which resembles the half-wave rectifier used in the earlier ad hoc study (Kearney and Hunter, 1984). Figure 6.15B shows the IRF of the linear element.

Kearney and Hunter noted that the input signal was neither Gaussian nor white. Thus, the assumptions underlying the Lee–Schetzen cross-correlation method (Lee and Schetzen, 1965) for estimating Wiener kernels did not hold. Similarly, Bussgang's theorem,

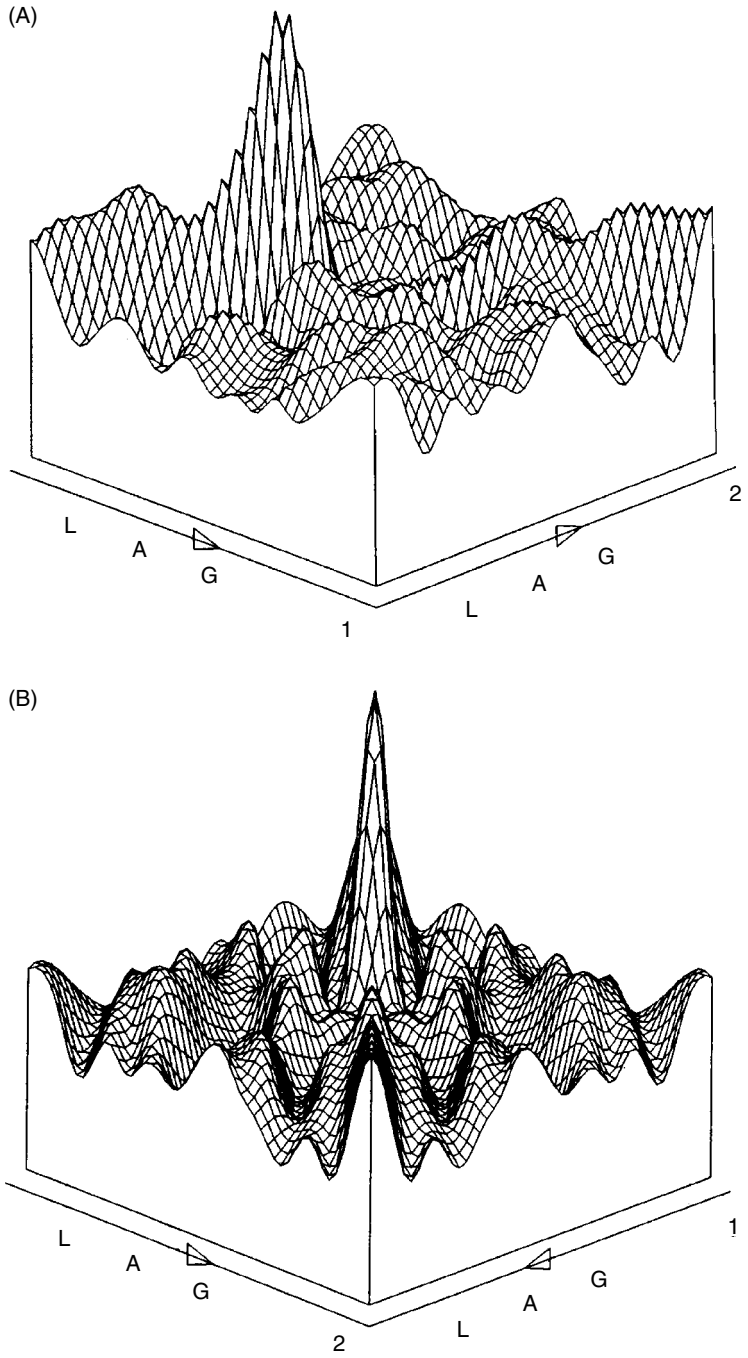

Figure 6.14 Cross-correlation analysis of stretch reflex EMG dynamics. (A) Second-order Wiener kernel viewed orthogonal to the diagonal. (B) Second-order Wiener kernel viewed along the diagonal. From Kearney and Hunter (1988). Used with permission of BMES.

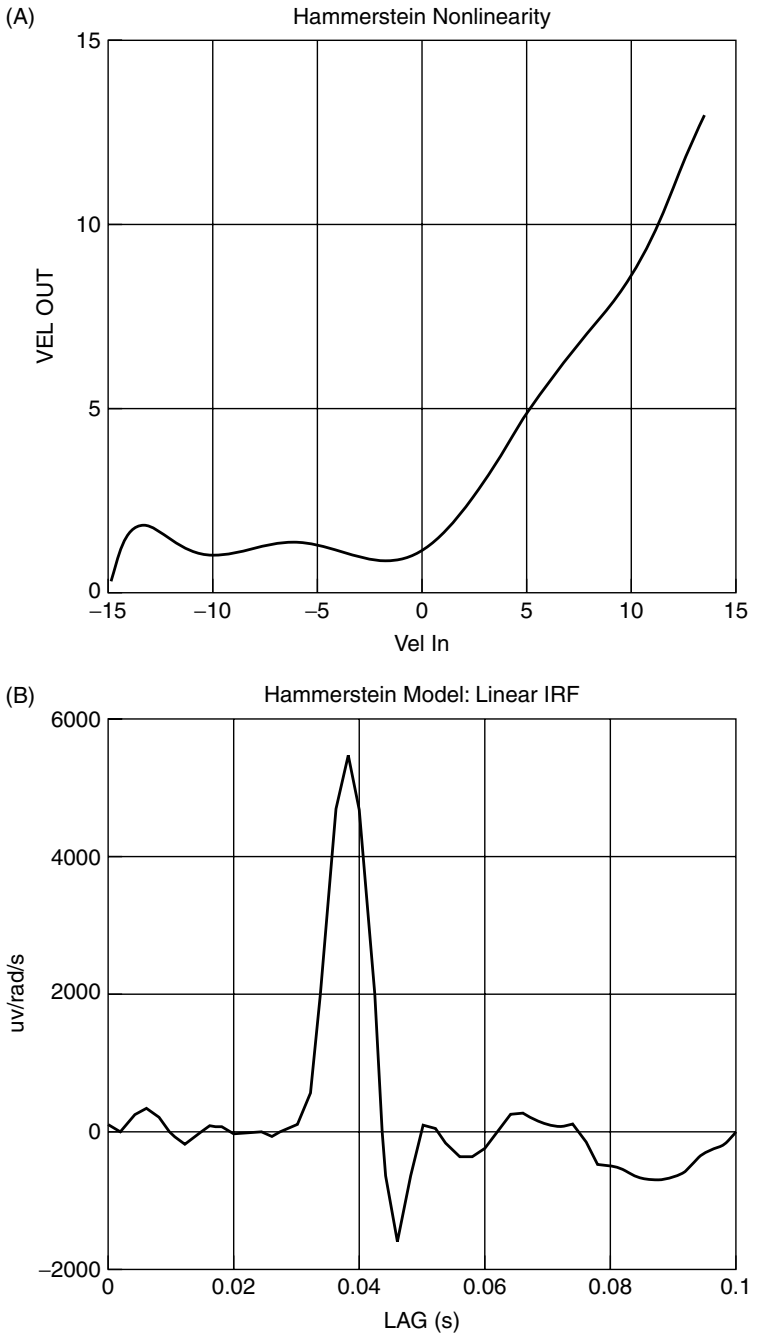

Figure 6.15 Hammerstein model of stretch reflex EMG dynamics. (A) Static nonlinearity estimate. (B) Linear IRF estimate. From Kearney and Hunter (1988). Used with permission of BMES.

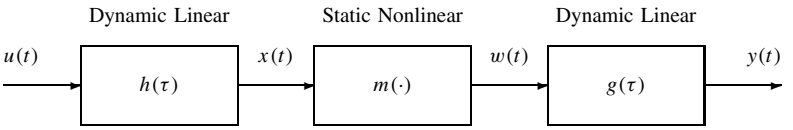

Figure 6.16 Block diagram of an LNL cascade model.

which underlies the iterative Hammerstein fitting approach (Hunter and Korenberg, 1986), would not apply to the experimental input. To assess the impact of these departures, the identified model was driven with a series of inputs, ranging from white Gaussian noise to the measured test input, and the identification was repeated. The Wiener kernels identified from the model, when driven with the experimental input, were found to be very similar to those obtained from the experimental data.

6.2.3 LNL Systems

Cross-correlation-based techniques are also available for LNL models—comprising two dynamic linear filters separated by a static nonlinearity as illustrated in Figure 6.16. These identification methods rely on the observation that the order-q Wiener kernel of an LNL model is proportional to its order-q Volterra kernel, with a constant of proportionality that depends on the kernel order.

This is most readily evident if the output of the LNL cascade is written as

$$y(t) = \sum_{\tau=0}^{T-1} g(\tau) w(t-\tau)$$

Thus, the input–output cross-correlation, used to estimate the first-order Wiener kernel,

$$\begin{aligned}\phi_{uy}(\tau) &= E\left[u(t-\tau)\sum_{j=0}^{T-1} g(j)w(t-j)\right] \\ &= \sum_{j=0}^{T-1} g(j)E[u(t-\tau)w(t-j)] \\ &= \sum_{j=0}^{T-1} g(j)\phi_{uw}(\tau-j)\end{aligned}$$

is the convolution of $g(\tau)$ with the cross-correlation measured across a Wiener system consisting of $h(\tau)$ in series with $m(\cdot)$. The cross-correlation across this Wiener cascade is given by

$$\phi_{uw}(\tau) = \gamma^{(1)} \sum_{j=0}^{T-1} \phi_{uu}(j) h(\tau-j)$$

where $\gamma^{(1)}$ is the first-order coefficient in the normalized Hermite polynomial representation of the static nonlinearity [see equation (6.47)]. Thus,

$$\phi_{uy}(\tau) = \gamma^{(1)} \sum_{j_1=0}^{T-1} g(j_1) \sum_{j_2=0}^{T-1} \phi_{uu}(j_2) h(\tau - j_1 - j_2)$$

Removing the convolution with respect to the input autocorrelation yields the first-order Wiener kernel,

$$\mathbf{k^{(1)}}(\tau) = \gamma^{(1)} \sum_{j=0}^{T-1} g(j) h(\tau - j)$$

From equation (4.44), the first-order Volterra kernel of an LNL cascade is given by

$$\mathbf{h^{(1)}}(\tau) = c^{(1)} \sum_{j=0}^{T-1} g(j) h(\tau - j)$$

where $c^{(1)}$ is the first-order coefficient in the power-series representation of the static nonlinearity. Thus, the first-order Wiener and Volterra kernels of a LNL cascade are proportional to each other:

$$\mathbf{k^{(1)}}(\tau) = \frac{\gamma^{(1)}}{c^{(1)}} \mathbf{h^{(1)}}(\tau)$$

A similar analysis can be used to compute all higher-order Wiener kernels of the LNL cascade. Thus, the qth order Wiener kernel of a LNL cascade is given by

$$\mathbf{k^{(q)}}(\tau_1, \ldots, \tau_q) = \gamma^{(q)} \sum_{j=0}^{T-1} g(j) h(\tau_1 - j) h(\tau_2 - j) \ldots h(\tau_q - j) \tag{6.59}$$

which is proportional to equation (4.44), the qth-order Volterra kernel. The constant of proportionality, $\gamma^{(q)}/c^{(q)}$, depends on the kernel order, the variance of the input signal, and the shape of the nonlinearity.

In particular, the second-order Wiener kernel is given by

$$\mathbf{k^{(2)}}(\tau_1, \tau_2) = \gamma^{(2)} \sum_{j=0}^{T-1} g(j) h(\tau_1 - j) h(\tau_2 - j) \tag{6.60}$$

so that if $\mathbf{k^{(2)}}(:, j)$ is the first nonzero column of the kernel, then

$$h(\tau) = \kappa \mathbf{k^{(2)}}(\tau - j, j)$$

for some proportionality constant, κ.

The second linear element, $g(\tau)$, can then be obtained by deconvolving $h(\tau)$ from the first-order kernel (Korenberg and Hunter, 1996). Once the two linear elements have been identified, the polynomial coefficients may be found using a linear regression.

6.2.3.1 Iterative Methods As with the other block structures, cross-correlation estimation errors will bias the polynomial coefficient estimates. Iterative methods can be used to minimize these effects. Korenberg and Hunter (1986) proposed the following algorithm:

1. Choose $\hat{h}_0(\tau)$, an initial guess for the IRF of the first linear element. For example, one possibility is to use the first nonzero row of the second-order kernel as $\hat{h}_0(\tau)$. Set the iteration counter $k = 0$.
2. Fit a Hammerstein system $(\hat{m}_k(\cdot), \hat{g}_k(\tau))$ between $\hat{x}_k(t) = \hat{h}_k(\tau) * u(t)$ and $y(t)$ using the Hunter–Korenberg iterative algorithm described in Section 6.2.2.
3. If this is the first iteration, or if the model accuracy has improved significantly since the previous iteration, continue. Otherwise, since convergence appears to have stopped, exit.
4. Use a gradient-based minimization (see Chapter 8) to find the optimal linear filter, $\hat{h}_{k+1}$, to precede the Hammerstein system $(\hat{m}_k(\cdot), \hat{g}_k(\tau))$, increment k, and go to step 2.

6.2.3.2 Example: Block-Structured Identification of the Peripheral Auditory Model In this example, techniques for identifying block-structured models will be applied to simulated data from the peripheral auditory model. These data were used previously to demonstrate both the Toeplitz inversion technique (Section 6.1.2.2) and the FFT-based estimation of frequency domain kernels (Section 6.1.3.2). The records consisted of 5000 points (0.5 s) of input–output data. The input was low-pass filtered at 3750 Hz using a fourth-order Butterworth filter. White Gaussian noise was added to the model output, such that the SNR was 10 dB.

The first task in performing a block structured identification is to select an appropriate model structure. In some cases, this can be done using a priori knowledge about the underlying mechanisms. If this is not available, the structural tests, described in Section 4.3, can be used to reduce the number of potential model structures.

To apply these tests, the first- and second-order Wiener (or Volterra) kernels must be estimated. Since the input was nonwhite, Wiener kernels were estimated using the repeated Toeplitz inversion scheme, as in Section 6.1.2.2. Recall that the second-order Wiener kernel estimate, shown in Figure 6.8C, was dominated by high-frequency noise that completely obscured the shape of the underlying kernel. Smoothing the kernel eliminated the noise, but reduced the prediction accuracy of the model. However, the general shape of the kernel was still evident. Since the smoothed second-order kernel is not diagonal, the Hammerstein structure can be ruled out.

For a system to have the Wiener structure, all rows of the second-order kernel must be proportional to the first-order kernel and thus to each other. Again, the high-frequency noise in the unsmoothed estimates of the kernels makes it difficult to use the raw kernel estimates directly. However, the objects being tested, the first-order kernel estimate and slices of the estimated second-order kernel, are all one-dimensional, and thus they can be smoothed using a variation of the pseudo-inverse-based procedure, described in Section 5.2.3.4 for linear IRF identification.

Let $\hat{\mathbf{k}}^{(1)}(\tau)$ be the estimate of the first-order Wiener kernel; and let $\hat{\mathbf{k}}^{(2)}(\tau, j)$, for $j = 0, 1, 2$, be the first three slices of the second-order Wiener kernel estimate. Let $\hat{h}(\tau)$ represent any one of these objects and suppose that it results from using equation (5.10)

to estimate a linear IRF:

$$\hat{\mathbf{h}} = \boldsymbol{\Phi}_{\mathbf{uu}}^{-1}\boldsymbol{\phi}_{\mathbf{uw}}$$

where $\boldsymbol{\phi}_{\mathbf{uw}}$ is the cross-correlation between the input and the output of the filter $h(\tau)$. Since $u(t)$, the input used for the system identification, is not white, $\boldsymbol{\Phi}_{\mathbf{uu}}^{-1}$ would ordinarily be replaced with a low-rank pseudo-inverse,

$$\boldsymbol{\Phi}_{\mathbf{uu}}^{\ddagger} = \mathbf{V_1}\mathbf{S_1}^{-1}\mathbf{V_1}^{T}$$

where $\mathbf{V_1}$ and $\mathbf{S_1}$ are the singular values and vectors retained in the pseudo-inverse, as shown in equation (5.15). Using the pseudo-inverse projects $\hat{\mathbf{h}}$ onto the columns of $\mathbf{V_1}$. Thus, if the pseudo-inverse algorithm had been used to estimate $h(\tau)$, the result would have been

$$\boldsymbol{\Phi}_{\mathbf{uu}}^{\ddagger}\boldsymbol{\phi}_{\mathbf{uw}} = \mathbf{V_1}\mathbf{V_1}^{T}\hat{\mathbf{h}}$$

Therefore, the kernel slices may be smoothed by multiplying each slice by $\mathbf{V_1}\mathbf{V_1}^{T}$. For these data, removing only one singular vector eliminated most of the high-frequency noise. The results of this analysis are shown in Figure 6.17A. The second-order kernel slices are clearly not proportional to each other, so the Wiener structure may be eliminated.

To test for the LNL structure, the first-order kernel was compared to the sum of all columns of second-order kernel:

$$\hat{h}(\tau) = \sum_{j=0}^{T-1} \hat{\mathbf{k}}^{(\mathbf{2})}(\tau, j)$$

Again, noise in the first- and second-order kernel estimates made it impossible to apply the test directly. However, projecting both the first-order kernel estimate and the sum, $\hat{h}(\tau)$, onto the first 15 singular vectors eliminated most of this noise. The result is shown in Figure 6.17B. Since the two traces are similar, the LNL structure cannot be ruled out.

The algorithm of Section 6.2.3.1 was used to fit an LNL cascade to the simulated data. Both IRFs were 16 samples long; the initial guess for the first linear element, $h(\tau)$, was obtained by projecting the first column of the second-order Wiener kernel onto the first 15 singular vectors of the input autocorrelation matrix, the same smoothing procedure as for the structure tests. Models were identified using polynomials of increasing order until increasing the order failed to increase the model accuracy significantly; in this way a sixth-order polynomial was determined to be appropriate.

The results shown in Figure 6.18 are very similar to the elements used in the simulated system and shown in Figure 4.1. Furthermore, the identified model accounted for 88.6% VAF in the identification data and 87.3% in the validation data.

Note that the in-sample accuracy, 88.6% VAF, is slightly better than the 87.9% obtained by the second-order Wiener series model. However, the Wiener series model had 153 parameters while the LNL model had only 39. Thus, despite the fact that the Wiener series model had almost four times as many free parameters as the LNL cascade, it was unable to achieve the same accuracy. The discrepancy is even larger in the cross-validation data, where the Wiener series model accounted for 85.9% VAF, compared to 87.3% for the LNL cascade. This suggests that the Wiener series model was fitting more of the noise in the identification data than the LNL cascade.

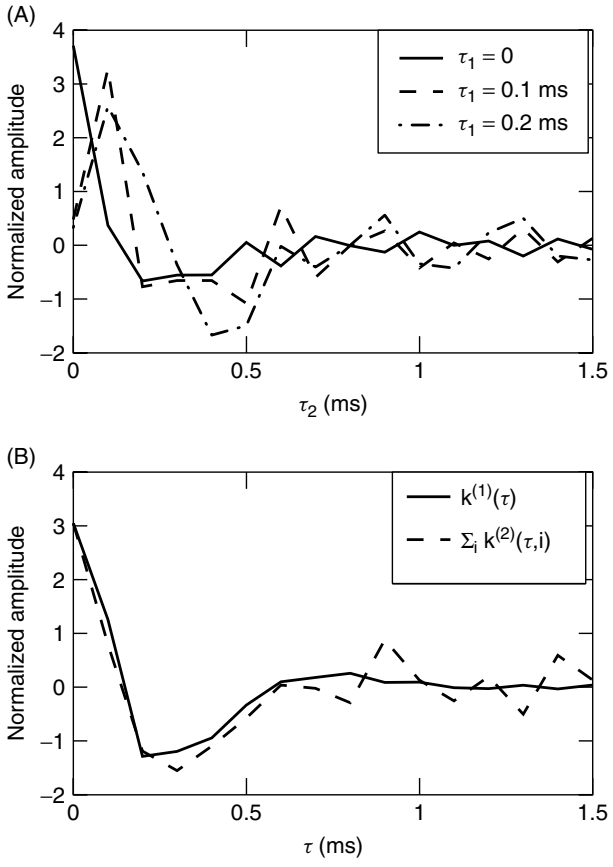

Figure 6.17 Testing for Wiener and LNL structures. (A) The first three columns [i.e., $\mathbf{k^{(2)}}(\tau_1, \tau_2)$ for τ_1 fixed at $0, 0.1$ and 0.2 ms]. The slices are not proportional so the Wiener structure can be eliminated. (B) The first-order kernel and the sum of all columns in the second-order kernel. The similarity between the two (normalized) traces indicates that the LNL structure cannot be ruled out.

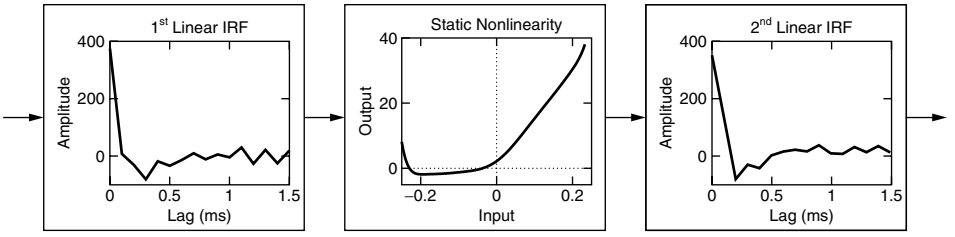

Figure 6.18 Elements of the identified LNL model. Note that the static nonlinearity resembles a half-wave rectifier, for inputs between about -0.2 and 0.2 (arbitrary units). Since the input was Gaussian, and given the variance of the intermediate signal, it is likely that more than 90% of the data points fell into this range.

6.3 PROBLEMS

1. Goussard et al. (1985) proposed a change to the Lee–Schetzen (Lee and Schetzen, 1965) cross-correlation method, in which the output was cross-correlated against Hermite polynomials of delayed inputs, rather than simple products of delayed inputs. Show that this procedure also results in estimates of the Wiener kernels. What are the advantages/disadvantages of this approach?

2. Consider a Hammerstein system, whose nonlinearity is $m(u) = au + bu^2$ and whose linear element has $h(\tau)$ as its IRF. Let the input be a zero-mean Gaussian signal with autocorrelation $\phi_{uu}(\tau)$. Compute the result of using the Lee–Schetzen cross-correlation method to estimate this system's Wiener kernels. What happens if the linear system is high-pass (so that $\sum_{\tau=0}^{\infty} h(\tau) = 0$)? What happens if the input is actually white?

3. Consider a Wiener system, $[h(\tau), m(\cdot)]$, where the nonlinearity is $m(x) = ax + bx^3$. Let the input autocorrelation be $\phi_{uu}(\tau)$. Which polynomial coefficients will result in a zero first-order Wiener kernel? Now, consider the system $[g(\tau), n(\cdot)]$, where

$$g(\tau) = kh(\tau)$$
$$n(x) = \frac{a}{k}x + \frac{b}{k}x^3$$

   What is the first-order Wiener kernel of this system (using the values of a and b found previously)?

6.4 COMPUTER EXERCISES

1. Open the file `ch6/mod1.mat`, which contains an NLDAT object containing an input–output dataset. Is the input White? Gaussian? Estimate the zeroth, first- and second-order Wiener kernels of the system, using the WSERIES object, being careful to use an appropriate memory length. Once you have identified a reasonable Wiener series model, test your kernels to determine Whether the system is a Wiener, Hammerstein or LNL cascade (or none of the above).

2. Repeat question 1 with the additional datasets `mod2.mat` and `mod3.mat`.

3. Open the file `ch6/catfish.mat`. Estimate the Wiener kernels from the data, and test the hypothesis that the data were generated by a Wiener cascade. Identify Wiener cascades using both the HK and PHK methods.

4. Repeat the above with the data contained in `ch6/catfish2.mat`.

5. Open the file `ch6/reflex.mat`, and estimate Wiener kernels from the data. Test the kernels for a Hammerstein structure. Use the HK method to fit a Hammerstein structure to the data. How does the prediction accuracy of the Hammerstein cascade compare to that of the Wiener series model.