

Stochastic State-Space Modeling with Applications

Jim White
jvwhite.com
james.v.white@gmail.com

Last revision: May 24, 2010

Contents

1	Baseline CV Algorithm	3
1.1	Introduction	3
1.2	Canonical-Variates Analysis	5
1.3	Least-Squares Parameter Estimation	6
1.4	Model Selection	7
1.5	Concise Summary	8
2	CV Algorithm Extensions	10
2.1	Choice of Input Parameters	10
2.2	Transfer-Function Modeling	11
2.3	Modeling Rank-deficient Processes	14
2.4	Pooling Multiple Time Series	14
2.5	Software	16
2.6	Topics for Future Research	16
2.6.1	Generalized Local Past and Future	17
2.6.2	Nongaussian State-Space Models	19
2.6.3	Nonlinear State-Space Models	19
2.6.4	Updating Model Parameters	19
2.6.5	Modeling Processes With Deterministic Components	20
3	Applications of State-Space Models	21
3.1	Estimation of Power Spectra and Covariances	21
3.1.1	Power Spectra	22
3.1.2	Covariance Sequences	25
3.2	State-Space Representations and Reduced-Order Modeling	26
3.3	Data Simulation	27
3.4	Kalman Filtering	27
3.4.1	State-space Models for Noisy Observations	27
3.4.2	Kalman Filtering Algorithm	29
3.5	Optimal Detection and Classification	30
3.5.1	Probabilistic Formulation	30
3.5.2	Optimal Classifier Algorithm	32

A Performance Objectives	36
A.1 CV Analysis	36
A.2 Least-Squares Parameter Estimation	37
A.3 Model Selection	38
B Derivation of Baseline CV Algorithm	39
B.1 Useful Notation	39
B.2 Fundamentals of Least-Squares Estimation	40
B.3 Derivation of Baseline CV Algorithm	41
B.3.1 CV Analysis	41
B.3.2 Least-Squares Parameter Estimation	43

Chapter 1

Baseline CV Algorithm

This report provides a detailed discussion of the baseline canonical-variate (CV) algorithm [1, 2], which is a direct descendent of the original algorithm developed by Lawrence Larimore [3, 4, 5] for automatically constructing linear state-space models based on vector time series data. This report provides the theoretical justification for the CV algorithm and specifies the algorithm in full detail. The report also shows how state-space models provide useful estimates of power spectra and lagged covariances when classical approaches fail. Finally, the report explains how state-space techniques have been used for optimally detecting signals in colored noise and classifying data sequences based on Kalman filtering.

1.1 Introduction

A *time series* is a sequence of observed m -vectors: $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N$, where at time $t \in \{1, 2, \dots, N\}$

$$\mathbf{y}_t = \begin{pmatrix} y_t(1) \\ y_t(2) \\ \vdots \\ y_t(m) \end{pmatrix} \quad (1.1)$$

The elements of \mathbf{y}_t are real numbers. If the original data vectors, say \mathbf{y}'_t , contain complex numbers, then the real and imaginary parts are listed separately in \mathbf{y}_t :

$$\mathbf{y}_t = \begin{pmatrix} \Re \mathbf{y}'_t \\ \Im \mathbf{y}'_t \end{pmatrix} \quad (1.2)$$

By so listing the real and imaginary parts, it is possible to model complex processes that have nonzero covariances between $\Re \mathbf{y}'_t$ and $\Im \mathbf{y}'_t$.

A *state-space model in innovations form* for the process generating the time series has the following state equations for all integer values of t :

$$\mathbf{x}_{t+1} = \Phi \mathbf{x}_t + G \mathbf{e}_t \quad (1.3)$$

$$\mathbf{y}_t = H \mathbf{x}_t + \mathbf{e}_t \quad (1.4)$$

$$C = \text{cov}(\mathbf{e}_t) \quad (1.5)$$

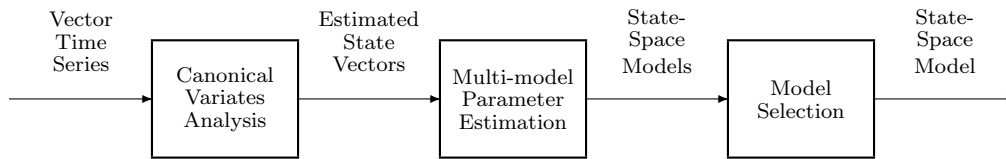


Figure 1.1: Baseline CV modeling algorithm.

In Eq. 1.3, \mathbf{x}_t is the $n \times 1$ *state vector*, Φ is the $n \times n$ *state transition matrix*, G is the $n \times m$ *noise-gain matrix*, and \mathbf{e}_t is the $m \times 1$ *innovations vector*. The innovations process is modeled as white noise. The state vector and innovations vector are internal variables; only the output of the model, \mathbf{y}_t , is observed. This is called *output modeling*. In Eq. 1.4, \mathbf{y}_t is expressed as the sum of two terms. The first term, which contains the $m \times n$ *output matrix* H , is that part of \mathbf{y}_t which is correlated with the past and depends only on the current state vector. The second term, the innovations \mathbf{e}_t , is that part of \mathbf{y}_t which is uncorrelated with the past. In Eq. 1.5, C is the $m \times m$ *innovations covariance matrix*, which is defined as $C = E\mathbf{e}_t\mathbf{e}_t^T$, where the E denotes expectation, and the superscript T denotes matrix transpose.

The baseline algorithm described in this chapter solves the simplest problem in state-space modeling for time series data. This problem is stated as follows. Three inputs are given: (1) a time series of m -vectors; (2) two positive integers, p and f , which determine the magnitude of the computations and define the reach of the *local past* and *local future* in the baseline CV analysis (these terms will be defined later); and (3) a positive integer, n_{\max} , which is the upper limit on the number of state variables being considered. (The number of state variables is bounded by the reach of the local past and future: $n_{\max} \leq \min[mp, mf]$.)

Given these inputs, the problem is to find optimal numerical values for the state-space parameters (n, Φ, G, H, C) . The optimal model is defined as that model, among the set of possible ones based on p and f and having $n \leq n_{\max}$, which is expected to be the most accurate from an information-theory point of view. Additional information about this criterion of optimality is provided in [1, 6].

As depicted in Fig. 1.1, the CV modeling algorithm consists of three steps: (1) a canonical-variates analysis to estimate canonical states; (2) a least-squares analysis to compute model parameters; and (3) an AIC analysis to select the optimal model. These steps are explained in sections 1.2 to 1.4. A *concise summary* of the complete algorithm is provided in section 1.5.

Extensions to the baseline CV algorithm are discussed in chapter 2. Criteria for choosing appropriate values for the *input parameters* (p, f, n_{\max}) are discussed in section 2.1. Section 2.2 discusses the application of CV to *transfer-function modeling*. Modeling *rank-deficient processes* is treated in section 2.3. The *pooling* of multiple time series is discussed in section 2.4. Current *software tools* for CV state-space modeling are described in section 2.5. Topics for *future research* are suggested in section 2.6. Section 3 discusses several applications of state-space models: estimating power spectra and covariances from time series; reduced-order modeling; data simulation, Kalman filtering, and optimal detection and classification. Appendix A discusses the *performance objectives* with respect to which the CV modeling algorithm is optimal, and appendix B provides a *derivation* of the baseline CV algorithm.

1.2 Canonical-Variates Analysis

A statistical covariance analysis of the observed time series is performed to determine the best (minimum-variance) linear transformations of vectors of past observations for the purpose of predicting vectors of future observations. This analysis is a generalization of principal components analysis in multivariate statistics, and is known in statistics as an analysis of canonical correlations. Intermediate quantities appearing in the analysis are vectors of *canonical variates*, which are statistics having nice canonical properties. In the CV modeling algorithm, these canonical variates are estimated state variables that form the core of the analysis.

The CV analysis consists of the matrix calculations described in this section. For $t = p + 1$ to N , define the *local past vector* \mathbf{z}_t^- :

$$\mathbf{z}_t^- := \begin{pmatrix} \mathbf{y}_{t-1} \\ \mathbf{y}_{t-2} \\ \vdots \\ \mathbf{y}_{t-p} \end{pmatrix} \quad (1.6)$$

For $t = 1$ to $N - f + 1$, define the *local future vector* \mathbf{z}_t^+ :

$$\mathbf{z}_t^+ := \begin{pmatrix} \mathbf{y}_t \\ \mathbf{y}_{t+1} \\ \vdots \\ \mathbf{y}_{t+f-1} \end{pmatrix} \quad (1.7)$$

The dimension of \mathbf{z}_t^- is $mp \times 1$, and the dimension of \mathbf{z}_t^+ is $mf \times 1$. Therefore, the parameters p and f control the dimensions of the local past and future vectors. They also limit the possible number of state variables in the models: $n_{\max} \leq \min(fm, pm)$, where m is the dimension of \mathbf{y}_t .

The next set of calculations is most easily described in terms of the *sample covariance matrix* $S(\mathbf{a}_t, \mathbf{b}_t)$, which is defined for two vector quantities \mathbf{a}_t and \mathbf{b}_t :

$$S(\mathbf{a}_t, \mathbf{b}_t) := \frac{1}{N - f - p + 1} \sum_{t=p+1}^{N-f+1} \mathbf{a}_t \mathbf{b}_t^T \quad (1.8)$$

Note that the sample covariance is defined with summation limits that depend on p and f . This is done so that when the covariances between the local future and local past are computed, the summation does not run off the ends of the time series.

Given the vectors \mathbf{z}_t^- and \mathbf{z}_t^+ computed using Eqs. 1.6 and 1.7, and given the definition of the sample covariance in Eq. 1.8, the *sample covariances between the local past and local future* are computed. These are the $mp \times mp$ matrix

$$C_{pp} = S(\mathbf{z}_t^-, \mathbf{z}_t^-) \quad (1.9)$$

the $mf \times mp$ matrix

$$C_{fp} = S(\mathbf{z}_t^+, \mathbf{z}_t^-) \quad (1.10)$$

and the $mf \times mf$ matrix

$$C_{ff} = S(\mathbf{z}_t^+, \mathbf{z}_t^+) \quad (1.11)$$

The next set of calculations involves the Cholesky factorizations of C_{pp} and C_{ff} . The *right Cholesky factor* of a positive-definite matrix A , denoted $[A]_R$, enters the right-handed factorization of A :

$$A \equiv [A]_R^T [A]_R \quad (1.12)$$

In contrast, the *left Cholesky factor*, denoted $[A]_L$, provides the left-handed factorization:

$$A \equiv [A]_L [A]_L^T \quad (1.13)$$

The right-handed and left-handed factors are related by the identity $[A]_R \equiv [A]_L^T$.

Given the definitions of the right and left Cholesky factors, the matrices C_{ff} and C_{pp} are factored, inverted, and then used to pre-multiply and post-multiply C_{fp} to compute the $mf \times mp$ matrix P :

$$P = [C_{ff}]_L^{-1} C_{fp} [C_{pp}]_R^{-1} \quad (1.14)$$

As Appendix B shows, P is the optimal estimator of the local future, given the local past, *after* the observed data have been standardized by suitable linear transformations to a new coordinate system in which they are dimensionless and have unity covariance matrices.

Having computed matrix P , it is factored using the *singular value decomposition* (SVD):

$$P = U \Sigma V^T \quad (1.15)$$

where U is an $mf \times mf$ orthogonal matrix, V is an $mp \times mp$ orthogonal matrix, and Σ is an $mf \times mp$ matrix. The diagonal elements, $\Sigma(k, k)$ for $k = 1$ to $\min(mf, mp)$, are the singular values of P arranged in order, from largest to smallest. The other elements of Σ are zero. With this ordering of the singular values, the estimated state vectors, defined below, are arranged in the order of their importance for predicting the future from the past.

Finally, the sequence of estimated state vectors, each of dimension n_{\max} , is computed for $t = p + 1$ to $N + 1$:

$$\hat{\mathbf{x}}_t = V(1:mp, 1:n_{\max})^T [C_{pp}]_L^{-1} \mathbf{z}_t^- \quad (1.16)$$

where $V(1:mp, 1:n_{\max})^T$ denotes the transpose of the $mp \times n_{\max}$ submatrix consisting of the first n_{\max} columns of $mp \times mp$ matrix V .

1.3 Least-Squares Parameter Estimation

The parameter matrices for a state-space model having n state variables are denoted (Φ^n, G^n, H^n, C^n) . These parameters are computed for a family of models having orders $n = 1$ to n_{\max} . Let $\hat{\mathbf{x}}_t^n$ denote the subvector consisting of the first n elements of $\hat{\mathbf{x}}_t$:

$$\hat{\mathbf{x}}_t^n := \begin{pmatrix} \hat{x}_t(1) \\ \hat{x}_t(2) \\ \vdots \\ \hat{x}_t(n) \end{pmatrix} \quad (1.17)$$

The calculations are started by computing the sample covariance of the time series:

$$C_{yy} = S(\mathbf{y}_t, \mathbf{y}_t) \quad (1.18)$$

For model orders $n = 1$ to n_{\max} , the following covariance matrices are computed:

$$C_{yx}^n = S(\mathbf{y}_t, \hat{\mathbf{x}}_t^n) \quad (1.19)$$

$$C_{xx}^n = S(\hat{\mathbf{x}}_t^n, \hat{\mathbf{x}}_t^n) \quad (1.20)$$

$$C_{x_1x}^n = S(\hat{\mathbf{x}}_{t+1}^n, \hat{\mathbf{x}}_t^n) \quad (1.21)$$

$$C_{x_1y}^n = S(\hat{\mathbf{x}}_{t+1}^n, \mathbf{y}_t) \quad (1.22)$$

The model parameter matrices are then computed:

$$H^n = C_{yx}^n \quad (1.23)$$

$$\Phi^n = C_{x_1x}^n \quad (1.24)$$

$$G^n = (C_{x_1y}^n - \Phi^n H^{nT})(C_{yy} - H^n H^{nT})^{-1} \quad (1.25)$$

The innovations covariance matrix for the n th model, C^n , is computed as

$$C^n = \frac{1}{N-p} \sum_{t=p+1}^N \mathbf{e}_t^n \mathbf{e}_t^{nT} \quad (1.26)$$

where the \mathbf{e}_t^n are computed recursively for $t = p+1$ to N as follows. First, a state vector of dimension n is initialized using the first n elements of the estimated state vector computed in Eq. 1.16 for the starting time $t = p+1$:

$$\mathbf{x}_{p+1}^n = \hat{\mathbf{x}}_{p+1}^n \quad (1.27)$$

Then, the innovations are computed for $t = p+1$ to N :

$$\mathbf{e}_t^n = \mathbf{y}_t - H^n \mathbf{x}_t^n \quad (1.28)$$

$$\mathbf{x}_{t+1}^n = (\Phi^n - G^n H^n) \mathbf{x}_t^n + G^n \mathbf{y}_t \quad (1.29)$$

1.4 Model Selection

The Akaike information criterion for Gaussian models is used to determine which model is expected to be most accurate. The AIC statistic is computed for each candidate model. For the zeroth-order ($n = 0$) model, which has no state variables because the time series is modeled as white noise, the innovations covariance is $C^0 = C_{yy}$, and the AIC is

$$\text{AIC}(0) = (N-p) \log_e \det(C^0) + m(m+1) \quad (1.30)$$

where \det denotes the determinant. For the remaining candidate models, having $n = 1$ to n_{\max} , the AIC is computed as

$$\text{AIC}(n) = (N-p) \log_e \det(C^n) + 4mn + m(m+1) \quad (1.31)$$

The optimal model, which is the one that is expected to be most accurate, has the smallest AIC. Let n denote the number of states in this model. If $n = 0$, then a white-noise model is optimum, and the innovations covariance is $C = C^0$. (The other parameters (Φ, G, H) are undefined for the white-noise model.) If $n > 0$, then $(\Phi, G, H, C) = (\Phi^n, G^n, H^n, C^n)$, which completes the state-space modeling algorithm.

1.5 Concise Summary

The computations described in the previous sections are summarized here. The inputs are (1) a time series of m vectors, \mathbf{y}_t for $t = 1$ to N , (2) the positive integers p and f , which define the reach of the local past and local future, and (3) the positive integer $n_{\max} \leq \min(fm, pm)$, which is the maximum number of state variables considered.

1. For $t = p + 1$ to N , define the *local past vector* \mathbf{z}_t^- :

$$\mathbf{z}_t^- := \begin{pmatrix} \mathbf{y}_{t-1} \\ \mathbf{y}_{t-2} \\ \vdots \\ \mathbf{y}_{t-p} \end{pmatrix} \quad (1.32)$$

2. For $t = 1$ to $N - f + 1$, define the *local future vector* \mathbf{z}_t^+ :

$$\mathbf{z}_t^+ := \begin{pmatrix} \mathbf{y}_t \\ \mathbf{y}_{t+1} \\ \vdots \\ \mathbf{y}_{t+f-1} \end{pmatrix} \quad (1.33)$$

3. Compute the three sample covariance matrices:

$$C_{pp} = S(\mathbf{z}_t^-, \mathbf{z}_t^-) \quad (1.34)$$

$$C_{fp} = S(\mathbf{z}_t^+, \mathbf{z}_t^-) \quad (1.35)$$

$$C_{ff} = S(\mathbf{z}_t^+, \mathbf{z}_t^+) \quad (1.36)$$

4. Compute the inverses of the left and right Cholesky factors of C_{ff} and C_{pp} and form the matrix

$$P = [C_{ff}]_L^{-1} C_{fp} [C_{pp}]_R^{-1} \quad (1.37)$$

5. Compute the SVD of P :

$$P \equiv U \Sigma V^T \quad (1.38)$$

6. Using the first n_{\max} columns of V , compute the estimated state vectors for $t = p + 1$ to $N + 1$:

$$\hat{\mathbf{x}}_t = V(1:n_{\max}, 1:n_{\max})^T [C_{pp}]_L^{-1} \mathbf{z}_t^- \quad (1.39)$$

7. Compute the sample covariance for the time series:

$$C_{yy} = S(\mathbf{y}_t, \mathbf{y}_t) \quad (1.40)$$

8. Compute the following sample covariance matrices and model parameter matrices for model orders $n = 1$ to n_{\max} :

$$C_{yx}^n = S(\mathbf{y}_t, \hat{\mathbf{x}}_t^n) \quad (1.41)$$

$$C_{xx}^n = S(\hat{\mathbf{x}}_t^n, \hat{\mathbf{x}}_t^n) \quad (1.42)$$

$$C_{x_1x}^n = S(\hat{\mathbf{x}}_{t+1}^n, \hat{\mathbf{x}}_t^n) \quad (1.43)$$

$$C_{x_1y}^n = S(\hat{\mathbf{x}}_{t+1}^n, \mathbf{y}_t) \quad (1.44)$$

$$H^n = C_{yx}^n \quad (1.45)$$

$$\Phi^n = C_{x_1x}^n \quad (1.46)$$

$$G^n = (C_{x_1y}^n - \Phi^n H^{n\Gamma})(C_{yy} - H^n H^{n\Gamma})^{-1} \quad (1.47)$$

9. For model orders $n = 1$ to n_{\max} , compute the innovations for $t = p + 1$ to N : the initial condition is

$$\mathbf{x}_{p+1}^n = \hat{\mathbf{x}}_{p+1}^n \quad (1.48)$$

and the recursion is

$$\mathbf{e}_t^n = \mathbf{y}_t - H^n \mathbf{x}_t^n \quad (1.49)$$

$$\mathbf{x}_{t+1}^n = (\Phi^n - G^n H^n) \mathbf{x}_t^n + G^n \mathbf{y}_t \quad (1.50)$$

10. Compute the innovation covariance matrices: for the zero-order model, $C^0 = C_{yy}$; and for model orders $n = 1$ to n_{\max} ,

$$C^n = \frac{1}{N-p} \sum_{t=p+1}^N \mathbf{e}_t^n \mathbf{e}_t^{n\Gamma} \quad (1.51)$$

11. Compute the AIC statistics for model order $n = 0$ to n_{\max} :

$$\text{AIC}(n) = (N-p) \log_e \det(C^n) + 4mn \quad (1.52)$$

12. The model having the smallest AIC is the one expected to be most accurate.

Chapter 2

CV Algorithm Extensions

2.1 Choice of Input Parameters

The AIC for selecting models is rigorous for Gaussian modeling of long time series. A long time series is one containing many more observed numbers than there are parameters being estimated in the state-space models. This concept is made more precise in the following. A time series of N vectors, each containing m real numbers, consists of mN scalars. In the baseline CV algorithm, only $N - f - p + 1$ time steps are used in computing covariance matrices (to avoid running off the ends of the data set). Therefore, there are $m(N - f - p - 1)$ scalars. In contrast, the number of independent parameters that define an n -state model is $2mn + m(m + 1)/2$. (Reference [1] contains a derivation of this result.) Therefore, the average number of degrees of freedom in each estimated parameter is

$$d = \frac{N - f - p - 1}{2n + (m + 1)/2} \quad (2.1)$$

A time series may be considered to be long for the purposes of developing an n -state model when $d \geq 20$. For example, if the time series consists of 100 scalars ($m = 1$ and $N = 100$), then $d = \frac{100}{2n+1}$. This time series is long for models having $n \leq 2$ state variables.

As another example, suppose that a time series consists of 300 4-vectors ($N = 300$ and $m = 4$). Then $d = \frac{300}{2n+5/2}$. This time series is long for models having $n \leq 6$.

The lengths of the local past and local future, p and f , determine the resolution with which the baseline CV algorithm models narrow-band structure in the time series. Usually $p = f$. Two constraints limit the magnitude of these parameters: (1) the larger p and f are, the less data are available for computing covariance matrices, because they are formed from outer products for times $t = p + 1$ to $N - f + 1$ to avoid running off the ends of the data; (2) the larger p and f are, the more computation that is required, because the dimensions of the covariance matrices become larger.

In practice, with long time series, progressively larger values for p and f are tried (with $p = f$). For each choice, the power spectrum of the resulting model is computed. By comparing these spectra, it may be verified that p and f have been set large enough so that increases have not significantly changed the spectrum.

The effects of different choices for p and f are depicted in Fig. 2.1, which shows the power spectra of three models corresponding to different choices for p and f with $p = f$. These models

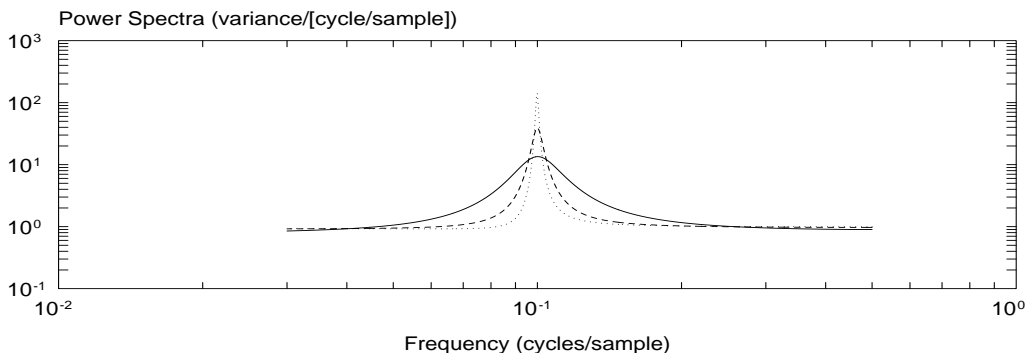


Figure 2.1: Comparison of effects of p and f on model power spectra ($p = f$, solid: $p = 5$, dashed: $p = 10$, dotted: $p = 20$).

were developed using a time series from the following stationary process:

$$y_t = \sqrt{2} \sin(2\pi t/10 + \theta) + n_t \quad (2.2)$$

where $t = 1, 2, \dots, 1000$, the random variable θ is uniformly distributed on the interval $[0, 2\pi)$, and the white noise n_t is Gaussian with a standard deviation of 1.0. The mean-square signal-to-noise ratio of this sinewave in noise is also 1.0. The spectrum of the process has a delta function at frequency $f = 0.1$ cycles/sample and a white-noise floor at a spectrum value of 1.0. The broadest spectrum in Fig. 2.1 corresponds to a model developed using $p = 5$, while the narrower, more accurate, spectra correspond to using $p = 10$ and $p = 20$.

An automatic approach to choosing p and f , which was first suggested by Akaike, is to model the time series using autoregressive models for orders $n = 1$ to some high order, e.g., $n_{\max} = \frac{N-5(m+1)}{20m}$, which corresponds to 10 degrees of freedom per model parameter. The best of these models is selected by determining which model has the smallest AIC. The order of this model is then used as the value for p and f in the CV algorithm. The merit of this approach is that it can be easily automated.

2.2 Transfer-Function Modeling

The time-series modeling discussed in the previous sections is *output modeling*. Transfer-function modeling, in contrast, is *input-output modeling*, because both input time series and output time series are observed. The application of CV analysis to input-output modeling is straight forward [3]. The state-space equations are augmented by adding terms corresponding to the *observed input vectors* \mathbf{u}_t of dimension q , for $t = 1$ to N :

$$\mathbf{u}_t = \begin{pmatrix} u_t(1) \\ u_t(2) \\ \vdots \\ u_t(q) \end{pmatrix} \quad (2.3)$$

The input-output state equations, in innovations form, are

$$\mathbf{x}_{t+1} = \Phi \mathbf{x}_t + B \mathbf{u}_t + G \mathbf{e}_t \quad (2.4)$$

$$\mathbf{y}_t = H \mathbf{x}_t + D \mathbf{u}_t + \mathbf{e}_t \quad (2.5)$$

$$C = \text{cov}(\mathbf{e}_t) \quad (2.6)$$

where the $n \times q$ matrix B and the $m \times q$ matrix D model the direct effects of inputs on the states and outputs, respectively. In Eqs. 2.4 and 2.5, \mathbf{u}_t is a known vector at each time, and \mathbf{e}_t is the innovations vector with covariance matrix C . This model simultaneously accounts for the dependence of \mathbf{y}_t on the known sequence \mathbf{u}_t as well as the part of \mathbf{y}_t that is not influenced by \mathbf{u}_t .

The z-transform of the input process is defined as

$$\mathbf{U}(z) := \sum_{t=0}^{\infty} \mathbf{u}_t z^{-t} \quad (2.7)$$

The z-transforms of the innovations and output processes are denoted $\mathbf{E}(z)$ and $\mathbf{Y}(z)$, respectively. The outputs may now be decomposed into two parts, one caused by \mathbf{u}_t and the other caused by \mathbf{e}_t :

$$\mathbf{Y}(z) \equiv \mathbf{Y}_u(z) + \mathbf{Y}_e(z) \quad (2.8)$$

where each term is defined in terms of an appropriate transfer function:

$$\mathbf{Y}_u(z) \equiv \mathbf{T}_u(z) \mathbf{U}(z) \quad (2.9)$$

$$\mathbf{Y}_e(z) \equiv \mathbf{T}_e(z) \mathbf{E}(z) \quad (2.10)$$

These transfer functions are computed from Eqs. 2.4 to 2.6 and are given by the following formulas, in which I_n denotes the $n \times n$ identity matrix:

$$\mathbf{T}_u(z) = H(zI_n - \Phi)^{-1}B + D \quad (2.11)$$

$$\mathbf{T}_e(z) = H(zI_n - \Phi)^{-1}G + I_m \quad (2.12)$$

For any normalized frequency f , the power spectral density matrix¹ of \mathbf{Y}_e is computed as follows:

$$\mathcal{S}(f) = \mathbf{T}_e(e^{i2\pi f}) C \mathbf{T}_e(e^{i2\pi f})^H \quad (2.13)$$

where the superscript H denotes the Hermitian transpose. The *folding frequency* of this spectral density is $f = 1/2$ cycle per sample.

The CV algorithm for determining the parameters (n, Φ, B, C, D, G, H) is a straight-forward application of the approach used for output modeling in chapter 1. The local past is augmented to include p_y output vectors and p_u input vectors:

$$\mathbf{z}_t^- = \begin{pmatrix} \mathbf{y}_{t-1} \\ \mathbf{y}_{t-2} \\ \vdots \\ \mathbf{y}_{t-p_y} \\ \mathbf{u}_{t-1} \\ \mathbf{u}_{t-2} \\ \vdots \\ \mathbf{u}_{t-p_u} \end{pmatrix} \quad (2.14)$$

¹The spectral density matrix is defined in section 3.1.

Usually in practice, $p_y = p_u$. The local past \mathbf{z}_t^- is now an $(mp_y + qp_u) \times 1$ vector, because the output vectors are m -dimensional and the input vectors are q -dimensional. The *reach* of the local past is defined to be

$$p := \max(p_y, p_u) \quad (2.15)$$

and the maximum number of state variables is bounded as follows:

$$n_{\max} \leq \min(mf, mp_y + qp_u) \quad (2.16)$$

The rest of the CV analysis proceeds as summarized in steps 2 through 5 of section 1.5. Step 6 is replaced with the following calculation, which accounts for the fact that the dimension of \mathbf{z}_p^- is $d_p = mp_y + qp_u$:

6. Using the first n_{\max} columns of V , compute the estimated state vectors for $t = p + 1$ to $N + 1$:

$$\hat{\mathbf{x}}_t = V(1:d_p, 1:n_{\max})^T [C_{pp}]_L^{-1} \mathbf{z}_t^- \quad (2.17)$$

The *least-squares parameter estimation* for input-output modeling involves several additional sample covariance matrices that account for the inputs and their correlations with the estimated state vectors. Steps 7 to 9 in section 1.5 are replaced with the following:

7. Compute the sample covariances for the output and input time series:

$$C_{yy} = S(\mathbf{y}_t, \mathbf{y}_t) \quad (2.18)$$

$$C_{uu} = S(\mathbf{u}_t, \mathbf{u}_t) \quad (2.19)$$

$$C_{yu} = S(\mathbf{y}_t, \mathbf{u}_t) \quad (2.20)$$

8. Compute the following sample covariance matrices and model parameter matrices for model orders $n = 1$ to n_{\max} , with I_n denoting the $n \times n$ identity matrix:

$$C_{yx}^n = S(\mathbf{y}_t, \hat{\mathbf{x}}_t^n) \quad (2.21)$$

$$C_{xx}^n = S(\hat{\mathbf{x}}_t^n, \hat{\mathbf{x}}_t^n) \quad (2.22)$$

$$C_{x_1x}^n = S(\hat{\mathbf{x}}_{t+1}^n, \hat{\mathbf{x}}_t^n) \quad (2.23)$$

$$C_{x_1y}^n = S(\hat{\mathbf{x}}_{t+1}^n, \mathbf{y}_t) \quad (2.24)$$

$$C_{xu}^n = S(\hat{\mathbf{x}}_t^n, \mathbf{u}_t) \quad (2.25)$$

$$C_{x_1u}^n = S(\hat{\mathbf{x}}_{t+1}^n, \mathbf{u}_t) \quad (2.26)$$

$$M = \begin{bmatrix} I_n & C_{xu}^n \\ C_{xu}^{nT} & C_{uu}^n \end{bmatrix} \quad (2.27)$$

$$[H^n \ D^n] = [C_{yx}^n \ C_{yu}^n] M^{-1} \quad (2.28)$$

$$[\Phi^n \ B^n] = [C_{x_1x}^n \ C_{x_1u}^n] M^{-1} \quad (2.29)$$

$$C_{ee}^n = C_{yy} - [H^n \ D^n] M [H^n \ D^n]^T \quad (2.30)$$

$$G^n = (C_{x_1y}^n - [C_{x_1x}^n \ C_{x_1u}^n] [H^n \ D^n]^T) (C_{ee}^n)^{-1} \quad (2.31)$$

9. For model orders $n = 1$ to n_{\max} , compute the innovations for $t = p + 1$ to N . The initial condition is

$$\mathbf{x}_{p+1}^n = \hat{\mathbf{x}}_{p+1}^n \quad (2.32)$$

and the recursion is

$$\mathbf{e}_t^n = \mathbf{y}_t - H^n \mathbf{x}_t^n - D^n \mathbf{u}_t \quad (2.33)$$

$$\mathbf{x}_{t+1}^n = (\Phi^n - G^n H^n) \mathbf{x}_t^n + (B^n - G^n D^n) \mathbf{u}_t + G^n \mathbf{y}_t \quad (2.34)$$

For input-output modeling, step 11 is replaced by the following, to account for the additional parameters involving inputs:

11. Compute the AIC statistics for model order $n = 0$ to n_{\max} :

$$\text{AIC}(n) = (N - p) \log_e \det(C^n) + 2(nq + 2mn + mq + m(m + 1)/2) \quad (2.35)$$

2.3 Modeling Rank-deficient Processes

A process is *numerically rank-deficient* when at least one of the sample covariance matrices, C_{pp} and C_{ff} , has a zero or extremely small positive eigenvalue. Rank deficiency occurs when part of a process can be predicted linearly with no error (or very little error) from observations of some other part of the process. For example, \mathbf{y}_t may be a scalar process consisting of a sinewave having a known frequency. All values of this signal can be predicted from just two consecutive observations, because the signal satisfies a 2nd-order linear difference equation. Another example of numerical rank deficiency is a time series which is the output of a digital filter that has multiple zeros at the folding frequency. Such filters are sometimes used for antialiasing.

The rigorous way of extending the baseline CV algorithm to deal with rank-deficient processes is straight forward: (1) replace the Cholesky decomposition by an alternative decomposition (such as one using the singular value decomposition) that works even if an eigenvalue of C_{pp} or C_{ff} is very small; and (2) replace all matrix inverses with Moore-Penrose pseudo inverses. This rigorous extension of the baseline CV algorithm will increase the computational requirements.

A simpler way of dealing with rank-deficient processes is to simply add a very small amount of white noise to the observations before starting the modeling process. This approach makes the process full-rank and has a clearly understood effect on the resulting model that is easy to assess.

2.4 Pooling Multiple Time Series

The CV modeling algorithm is easily extended to the combining, or pooling, of information from multiple time series that are to be modeled by a single process. For example, when bad data are removed from a time series, the resulting data set consists of multiple time-series fragments. Another need for pooling occurs when many time series can be collected from a process, but each series is short compared to the period of the lowest frequency at which the process is to be modeled. Then it is necessary to pool these short time series to obtain enough data to model the process at the lowest frequencies of interest.

The pooling of multiple time series is straight forward. Suppose there are n_s time series in the pool of data. The sample covariances between the local past and future, for the k th time series, are

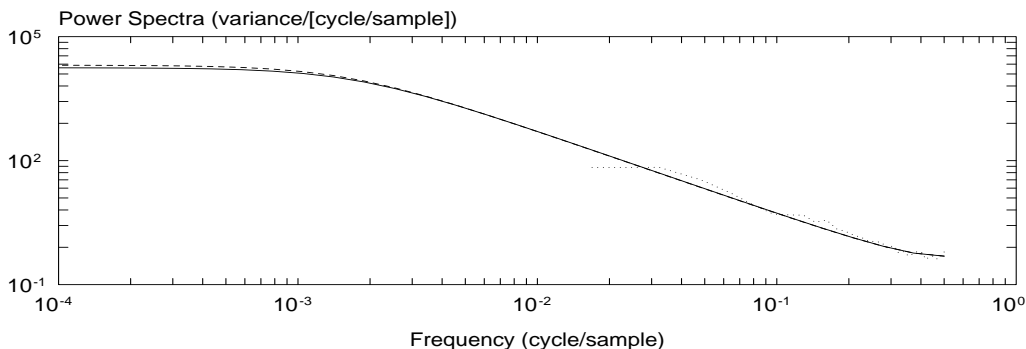


Figure 2.2: Power spectra of true model (dashed), estimated model (solid) derived from pooled time series, and Welch-FFT method (dotted).

denoted $C_{pp}(k)$, $C_{fp}(k)$, and $C_{ff}(k)$. The number of outer products that are summed to compute each of these covariance matrices for the k th time series is denoted N_k . Then the *pooled covariances*, which are to used in the CV analysis, are computed by summing the covariances for each series:

$$N_{\text{all}} = \sum_{k=1}^{n_s} N_k \quad (2.36)$$

$$C_{pp} = \frac{1}{N_{\text{all}}} \sum_{k=1}^{n_s} N_k C_{pp}(k) \quad (2.37)$$

$$C_{fp} = \frac{1}{N_{\text{all}}} \sum_{k=1}^{n_s} N_k C_{fp}(k) \quad (2.38)$$

$$C_{ff} = \frac{1}{N_{\text{all}}} \sum_{k=1}^{n_s} N_k C_{ff}(k) \quad (2.39)$$

A similar Equation is used to compute the innovation covariance matrix C^n for the model with n state variables:

$$C^n = \frac{1}{N_{\text{all}}} \sum_{k=1}^{n_s} N_k C^n(k) \quad (2.40)$$

The reaches of the local past and local future, p and f , are chosen to be compatible with the length of the shortest time series in the pool, i.e., $p + f < N_{\text{min}}$, where N_{min} is the number of vectors in the shortest time series.

An example of pooling is illustrated in Fig. 2.2. Three power spectra are compared: the dashed curve is the spectrum of the true model; the dotted curve is the spectrum estimate obtained using the Welch FFT-based algorithm working on a single 1500-point time series divided into nonoverlapping 64-point segments; and the solid curve is the spectrum of the state-space model obtained by pooling the collection of short time series obtained by cutting the 1500-point series into 30 nonoverlapping pieces, each 50 points long. The correlation time of the process generating

the time series is 100 samples, which is consistent with the corner frequency of the true spectrum at $1/(2\pi 100) = 1.59 \times 10^{-3}$ cyc/samp. The Welch method working with 64-point segments can not resolve the low-frequency behavior of the process, because the segments are shorter than the correlation time. However, the CV algorithm yields an accurate state-space model, in spite of the fact that short 50-point time series were used, because the information in the pool of time series spanning 15 correlation times is effectively combined. It should be noted that the pooling algorithm does not require that the short time series originate from a single long realization. There is no assumed continuity of sample paths between the time series forming the data pool.

2.5 Software

The MATLAB software provides the following capabilities:

Time Series \implies State-Space Model Functions that implement the baseline CV modeling algorithm

Time Series \implies Whiten Time Series Functions that use a state-space model to whiten time series and compute innovations covariance matrices

Whiten Time Series \implies Gaussian Log-likelihood A function for computing the log-likelihood of a Gaussian state-space model, given a whitened time series

Time Series \implies Short-Term Power Spectra Functions that compute and display, in three-dimensions, the temporal evolution of short-term power spectra

State-Space Model \implies State-Space Model Functions that convert a state-space model in one form into an equivalent model in a different form. Functions that compute reduced-order approximations of state-space models in innovations form, and functions that convert back and forth between discrete-time and continuous-time models

State-Space Model \implies Power Spectra and Spectral Coherence Functions that compute and plot the power spectra, spectral coherencies, group delays, and phase lags of processes from their state-space models

State-Space Model \implies Lagged Covariances A function that computes the lagged covariance sequence of a process from its state-space model

Lagged Covariances \implies State-Space Model A function that computes the parameters of a state-space model from a lagged covariance sequence

State-Space Model \implies Discrimination Information Functions that compute the information rate for discriminating against an incorrect state-space model, given the correct model.

2.6 Topics for Future Research

Several topics for future research in CV modeling that appear to be especially promising are discussed in the sections 2.6.1 to 2.6.5

2.6.1 Generalized Local Past and Future

Consider a modeling problem in which extremely long time series are available. The accuracy of the CV modeling algorithm is then determined by how large one allows the reaches of the local past and local future to be. The larger these reaches, the greater the accuracy. (Of course, once the reaches are large enough, increasing them produces very little accuracy improvement. For example, once the reaches exceed the longest correlation time of a process, the reaches are as large as they need to be for most applications.)

Suppose that an observed process has a very long correlation time, e.g., because it contains significant narrow-band signals. Then a very long reach is needed for the highest accuracy modeling. But this means that the dimensions of the covariances involving the local past and future will be correspondingly large, which in turn means that a very large amount of computation is needed to compute these covariances. *By generalizing the definitions of the local past and future, as described below, it is possible to provide very large reaches with relatively small dimensions for the covariances of the past and future.*

Output modeling will be discussed for simplicity. The generalization of this discussion to input-output modeling is straight forward. As is appropriate in most applications, let the reaches of the local past and future be equal ($p = f$). Because the local past and future are defined by Eqs. 1.6 and 1.7, the dimensions of the local past and future are equal to each other and proportional to the reach p :

$$\dim = mp \tag{2.41}$$

where m is the dimension of each vector in the time series. Therefore, when the reach is doubled, so are the dimensions of the covariance matrices between the past and future. The proportionality between dimension and reach in Eq. 2.41 can be avoided by generalizing the definitions of the local past and future, which will be explained by a simple example.

In this example, the desired reach is $p = 100$. The conventional way of defining the local past uses 100 *consecutive* past observations:

$$\mathbf{z}_t^- := \begin{pmatrix} \mathbf{y}_{t-1} \\ \mathbf{y}_{t-2} \\ \vdots \\ \mathbf{y}_{t-100} \end{pmatrix} \tag{2.42}$$

which is a $100m \times 1$ matrix. Now consider a generalization of the local past based on *progressive*

averaging of consecutive past observations:

$$\mathbf{z}_t^- := \begin{pmatrix} \mathbf{y}_{t-1} \\ \mathbf{y}_{t-2} \\ \vdots \\ \mathbf{y}_{t-10} \\ \sum_{k=11}^{12} \mathbf{y}_{t-k} \\ \sum_{k=13}^{14} \mathbf{y}_{t-k} \\ \vdots \\ \sum_{k=19}^{20} \mathbf{y}_{t-k} \\ \sum_{k=21}^{25} \mathbf{y}_{t-k} \\ \sum_{k=26}^{30} \mathbf{y}_{t-k} \\ \sum_{k=31}^{40} \mathbf{y}_{t-k} \\ \sum_{k=41}^{50} \mathbf{y}_{t-k} \\ \sum_{k=51}^{70} \mathbf{y}_{t-k} \\ \sum_{k=71}^{100} \mathbf{y}_{t-k} \end{pmatrix} \quad (2.43)$$

This is a $21m \times 1$ matrix, which is nearly 1/5th the size of the vector in Eq. 2.42. Each summation in Eq. 2.43 is a lowpass operation on part of the data within the reach of the local past. The summations extend over progressively larger domains, and the resulting lowpass filtering becomes more extreme, for samples further into the past. The idea behind this is that, for the purposes of predicting the future, only the lower-frequency part of the distant past is relevant. The high frequency part of the distant past is not expected to contribute significantly to future behavior for most processes. A similar line of reasoning leads to a similar generalization of the local future:

$$\mathbf{z}_t^+ := \begin{pmatrix} \mathbf{y}_t \\ \mathbf{y}_{t+1} \\ \vdots \\ \mathbf{y}_{t+9} \\ \sum_{k=10}^{11} \mathbf{y}_{t+k} \\ \sum_{k=12}^{13} \mathbf{y}_{t+k} \\ \vdots \\ \sum_{k=18}^{19} \mathbf{y}_{t+k} \\ \sum_{k=20}^{24} \mathbf{y}_{t+k} \\ \sum_{k=25}^{29} \mathbf{y}_{t+k} \\ \sum_{k=30}^{39} \mathbf{y}_{t+k} \\ \sum_{k=40}^{49} \mathbf{y}_{t+k} \\ \sum_{k=50}^{69} \mathbf{y}_{t+k} \\ \sum_{k=70}^{99} \mathbf{y}_{t+k} \end{pmatrix} \quad (2.44)$$

which is also a $21m \times 1$ matrix.

This approach to generalizing the local past and future raises a number of questions that future research could answer. For example, what are good rules for choosing the summation limits and number of summations? What is a practical approach to optimizing these selections with respect to a reasonable performance objective? How much of a computational advantage does this approach

realize for selected representative processes that are typical of specific types of data encountered in current applications?

The above method of extending the reach of the local past and future is effective for those time series in which progressively lower-frequency components are significantly correlated with each other over progressively larger time spans. However, there are time series in which narrowband components have long-range correlations. In this case, the lowpass filtering used above is expected to be less effective than an appropriate bandpass filtering. These thoughts lead to several questions for future research. How should the definition of the local past and future be tailored to the properties of the time series? Viewing the local past and future as linear functions of the observed data, what is an effective algorithm for determining the impulse responses of these functions. How can the *definition* of the local past and future be formulated as a least-squares problem that leads to a useful optimal solution?

2.6.2 Nongaussian State-Space Models

The baseline CV algorithm selects, from the candidate models being considered, that state-space model which is expected to have the largest Gaussian likelihood on time series that are independent of the one used for modeling. The algorithm can be generalized by enlarging the set of candidate models to include ones with nongaussian innovations. An example of the benefits of doing this, when the time series consists of the sum of a Gaussian process plus an outlier process, is given in [7].

This generalization of the baseline CV algorithm is easy to implement, because the linear dynamics of the models is being retained. Only the set of hypothesized distributions for the innovations is being enlarged. Every step in the baseline algorithm is retained, but some additional AIC statistics are computed during the last stage of selecting a model. For example, to handle time series having outliers that produce innovations with heavy tails, an AIC statistic for Cauchy-distributed innovations may be computed in addition to the Gaussian AICs already being computed.

2.6.3 Nonlinear State-Space Models

The linear state-space model structure defined by Eqs. 1.3 to 1.5 can be extended to polynomial nonlinearities. The baseline CV algorithm remains largely unchanged, except for the definition of the local past, which is defined using polynomial functions of the observed data. This nonlinear local past usually has a much larger dimension than a linear local past having the same reach. Therefore, the use of progressive filtering, as described in section 2.6.1, to reduce the dimension of the local past, may be extremely useful for nonlinear modeling. It is also likely that the extension to nongaussian innovations described in section 2.6.2 will also be appropriate to use in combination with nonlinear dynamics.

2.6.4 Updating Model Parameters

In some applications, it is appropriate to model a time series as coming from a time-varying process. In many cases, the time variations are slow enough to justify the use of a sequence of local time-invariant models. Each model represents the behavior of the data in a subsequence of the time series. The time series is divided into segments that may, or may not, overlap with each other. Then

a separate state-space model is computed for each segment. This approach leads to a sequence of models with each one being determined independently of the others.

Instead of the above discrete approach to updating the model parameters, an incremental approach may be better in some applications. An incremental approach may use less computing and may be more accurate when the process being modeled evolves smoothly. Future research is needed to develop algorithms for incremental updating of model parameters.

2.6.5 Modeling Processes With Deterministic Components

An example of a time series with a deterministic component is

$$y_t = A \sin(2\pi f_0 t + \theta) + v_t \quad (2.45)$$

where v_t is white noise, and the sinewave has three parameters: A is the amplitude; f_0 is the frequency; and θ is the phase. If the sinewave parameters are known, then the sinewave is a deterministic component. Even when the sinewave parameters are modeled as random variables, the sinewave is a deterministic waveform on each sample path, because the random parameters of the sinewave are independent of time.

To pursue the idea of deterministic components in more detail, the following model is considered: the sinewave parameters A and θ are independent random variables and are independent of the noise; the amplitude A has zero mean and unit variance; the noise v_t has unit variance; the frequency f_0 is known; and the phase θ is uniformly distributed on the interval $[0, 2\pi)$. Then the power spectrum of y_t is $S_{yy}(f)$ for $|f| \leq 1/2$:

$$S_{yy}(f) = \frac{1}{2}\delta(f - f_0) + \frac{1}{2}\delta(f + f_0) + 1 \quad (2.46)$$

where $\delta(f - f_0)$ is a delta function at $f = f_0$. This power spectrum has two delta functions. But finite dimensional state-space models have power spectra given by rational functions that can not represent delta functions. Therefore, this process can not be represented exactly, for $t = \dots, -2, -1, 0, 1, 2, \dots$, by a state-space model having a finite number of states. However, we can represent the process by a state-space model for $t = 1, 2, \dots$ by introducing appropriate *random initial conditions* for the model. The delta functions arise because of the sinewave, and the sinewave is the solution of a time-invariant linear difference equation. Therefore, by defining the initial states of the model to be random variables with appropriate means and covariances, we can exactly represent 2nd-order consequences of the random sinewave parameters A and θ .

Although the above example concerns a sinewave, deterministic components may also take the form of ramps, parabolas, higher-degree polynomials, damped exponentials, etc. In each case, the deterministic component can be modeled as the solution of a finite-dimensional system of difference equations in state-space form, with appropriate random initial conditions on the state vector.

This discussion has shown that the modeling of deterministic components in a time series involves the selection of appropriate initial conditions. In the case of modeling for power spectrum estimation, an appropriate mean vector and covariance matrix for the initial state vector in the model should be specified. The baseline CV algorithm does not address this problem. Therefore, this is a significant topic for future research. A recent paper [2] discusses a solution to this problem for the class of autoregressive models.

Chapter 3

Applications of State-Space Models

This chapter surveys a variety of important applications for state-space models. The intention is to describe briefly the nature of each application and to provide key definitions and algorithms.

3.1 Estimation of Power Spectra and Covariances

For a stationary zero-mean discrete-time vector process \mathbf{y}_t , the power spectral density matrix $S_{yy}(f)$ (which contains the autospectra and cross-spectra) and the covariance sequence $C_{yy}(t)$ (which contains the autocorrelations and the lagged covariances) are defined as follows for frequencies $-1/2 < f \leq 1/2$ and lags $t = \dots, -2, -1, 0, 1, 2, \dots$:

$$S_{yy}(f) := \sum_{t=-\infty}^{\infty} C_{yy}(t) e^{-i2\pi ft} \quad (3.1)$$

$$C_{yy}(t) := \mathbf{E} \mathbf{y}_k \mathbf{y}_{k-t}^T \quad (3.2)$$

where \mathbf{E} denotes expectation. In theory, the correlation sequence can be computed from the power spectrum as follows:

$$C_{yy}(t) = \int_{-1/2}^{1/2} S_{yy}(f) e^{i2\pi ft} df \quad (3.3)$$

However, if the parameters of a state-space model (Φ, G, H, C) are known (e.g., by using the CV modeling algorithm), then the power spectrum and covariance sequence *can be computed algebraically*. In fact, this is usually the preferred method of estimating power spectra and covariances from short time series.

First the CV algorithm is used to develop a state-space model from the time series. Then the power spectrum and covariance sequence of the model are used as estimates. The advantage of this approach, for short time series, is that much more accurate estimates are obtained, because far fewer parameters are estimated. Furthermore, the resolution of the spectral estimate is not limited by the assumption of periodicity that is inherent in the FFT approaches. Finally, the model-based approach produces both the desired estimates and also a model for the process. This model is frequently crucial to the optimal solution of related estimation, detection, and classification problems. The model can also be used to generate simulated data for testing algorithms with realistic data having known statistical properties. Several of these topics are discussed in later sections.

3.1.1 Power Spectra

For the state-space model given by Eqs. 1.3 to 1.5, the power spectrum is computed at any normalized frequency f as follows:

$$S_{yy}(f) = \mathbf{T}(e^{i2\pi f})C\mathbf{T}(e^{i2\pi f})^H \quad (3.4)$$

where the superscript H denotes a complex conjugate transpose, and the transfer-function matrix is

$$\mathbf{T}(z) = H(zI_n - \Phi)^{-1}G + I_m \quad (3.5)$$

In Eq. 3.5, I_n is the n th-order identity matrix, n is the number of state variables in the model, and m is the dimension of the vector \mathbf{y}_t .

An example of estimating power spectra from vector time series will now be discussed. This discussion will include a comparison of the CV state-space approach with the classical Welch-FFT approach to spectrum estimation. The data used in this example form a time series generated by a state-space process having known properties. The process is defined by specific state-space parameters (m, n, Φ, G, H, C) , where $n = 4$ state variables and $m = 2$ outputs. This model is used to generate a vector times series 4096 time steps in length. The number 4096 is chosen, because a classical FFT-based technique for spectrum estimation will be considered, and an integer power of 2 favors the use of FFTs.

The baseline CV algorithm, with a local past and local future of $p = f = 20$, is used with a limit of $n_{\max} = 6$ state variables. The resulting estimated model parameters, $(m, n', \Phi', G', H', C')$, are then used with Eqs. 3.4 and 3.5 to compute the power spectral density matrix of the process defined by this model for selected frequencies. The autospectra are diagonal entries of the spectral density matrix, and the cross spectra are off-diagonal entries. In Fig. 3.1 the estimated auto spectra (solid curves) are compared with the true auto spectra (dashed curves). The small errors in the estimated spectra confirm that the CV algorithm has accurately modeled the process which generated the time series.

The accuracy of the CV-based estimates of the cross spectra is depicted in Fig. 3.2, which compares the estimated squared coherence (solid) between channels 1 and 2 with the true coherence (dashed). The squared coherence $\mathcal{C}_{jk}(f)$ between channels j and k of a vector process is a function of frequency f and is defined as follows:

$$\mathcal{C}_{jk}(f) = \frac{|S_{jk}(f)|^2}{S_{jj}(f)S_{kk}(f)} \quad (3.6)$$

where $S_{jk}(f)$ is the cross-spectral density between channels j and k at frequency f . The squared coherence takes values on the unit interval. The integral

$$\int_{-1/2}^{1/2} \mathcal{C}_{jk}(f) df$$

is the fraction of total variance in process j that can be explained by an optimal linear transformation of process k . Because of symmetry, j and k can be interchanged in this interpretation. The high accuracy of the CV-based coherence estimate is evident from Fig. 3.2.

The advantage of CV model-based spectral estimation over nonparametric FFT-based estimation is clear by comparing Figs. 3.1 and 3.2 with the results of using the Welch-FFT algorithm,

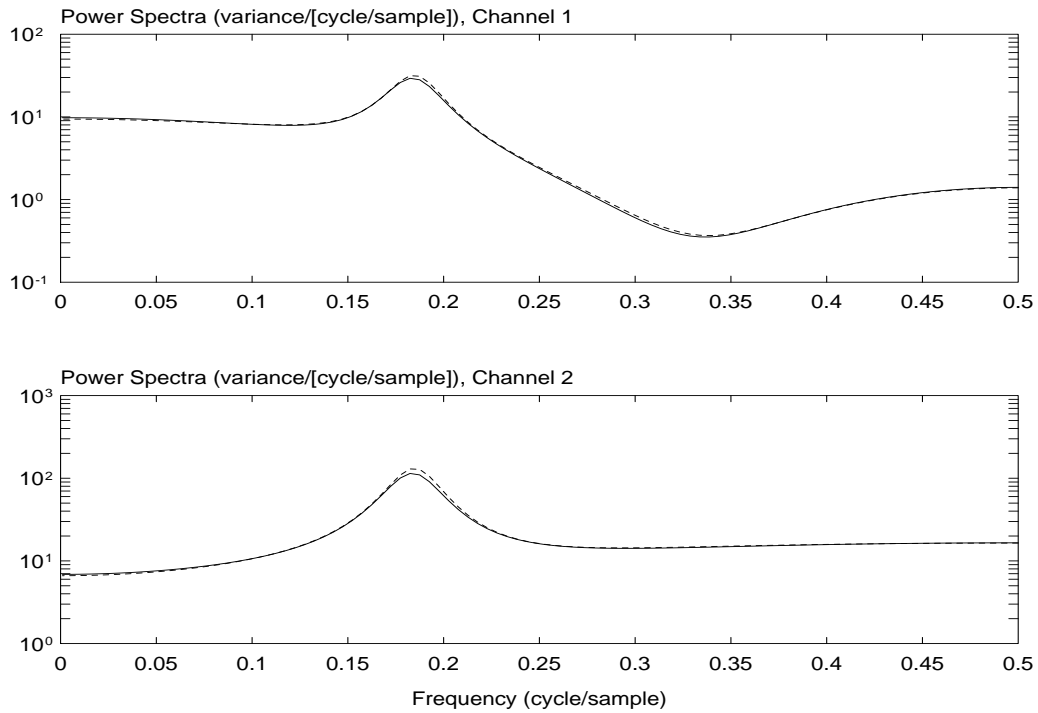


Figure 3.1: Comparison of true (dashed) and estimated (solid) auto spectra for channels 1 and 2, based on CV modeling.

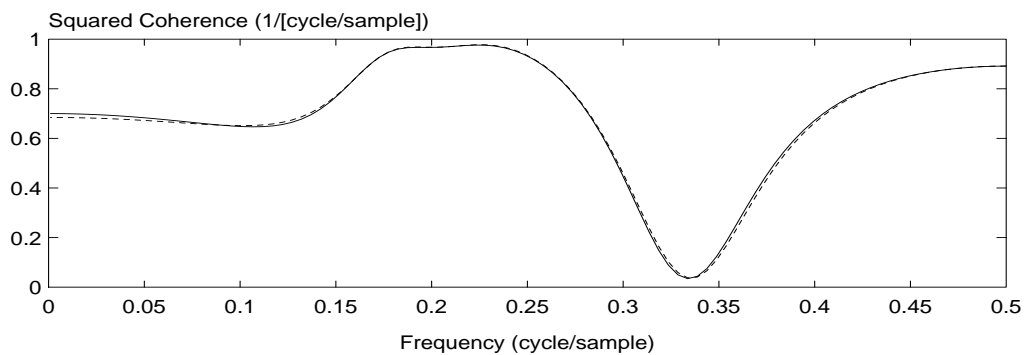


Figure 3.2: Comparison of true (dashed) and estimated (solid) squared coherence between channels 1 and 2, based on CV modeling.

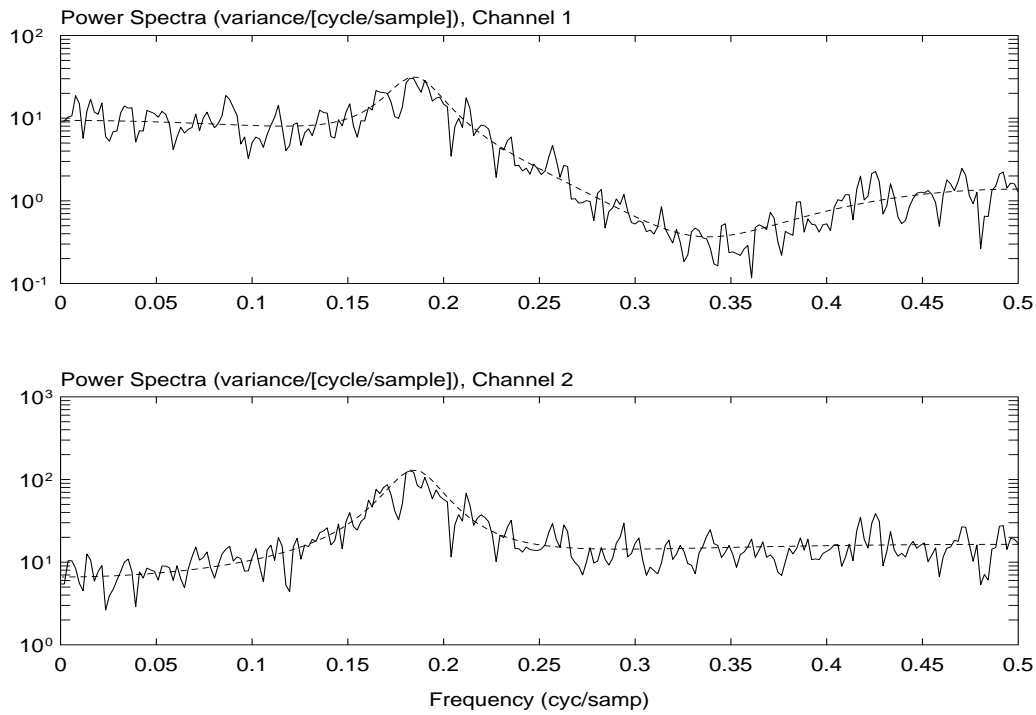


Figure 3.3: Comparison of true (dashed) and estimated (solid) auto spectra for channels 1 and 2, based on Welch-FFT method.

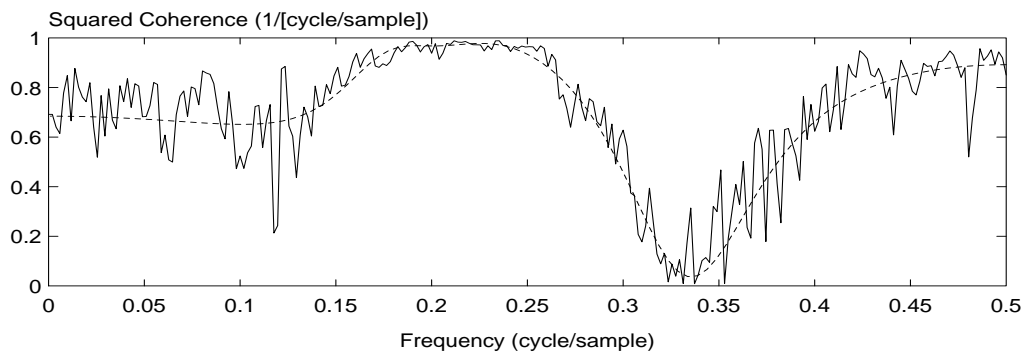


Figure 3.4: Comparison of true (dashed) and estimated (solid) squared coherence between channels 1 and 2, based on Welch-FFT method.

which are shown in Figs. 3.3 and 3.4. In applying the Welch algorithm, nonoverlapping 512-point data segments are used. The two methods of spectral estimation have similar trends, but the FFT-based estimate is much noisier.

3.1.2 Covariance Sequences

The covariance sequence has the property that, for all integer lags t ,

$$C_{yy}(-t) \equiv C_{yy}(t)^T \quad (3.7)$$

Therefore, an algorithm is given for computing the sequence for nonnegative lags only. The covariances for negative lags can be computed using Eq. 3.7.

For the state-space model given by Eqs. 1.3 to 1.5, the steady-state lagged covariance sequence $C_{yy}(t)$ for $t \geq 0$ is computed in three stages. In this section, the state sequence \mathbf{x}_t is modeled as having zero mean. The first stage is to compute the steady-state zero-lag state covariance matrix

$$C_{xx} := \mathbf{E}\mathbf{x}_t\mathbf{x}_t^T \quad (3.8)$$

This covariance matrix satisfies the discrete-time Lyapunov equation

$$C_{xx} = \Phi C_{xx} \Phi^T + GCG^T \quad (3.9)$$

A simple, but not necessarily efficient, procedure for computing the steady-state covariance is to use successive approximations: set $C_{xx}^1 = I_n$; and then, for $k = 1, 2, \dots$

$$C_{xx}^{k+1} = \Phi C_{xx}^k \Phi^T + GCG^T \quad (3.10)$$

The iterations stop when either a matrix norm of the difference $C_{xx}^{k+1} - C_{xx}^k$ is less than some accuracy threshold, or the number of iterations has reached a predetermined limit. It should be noted that not all state-space models have steady-state state covariances. For example, some systems are unstable, and in others not all states are correlated with the innovations process. There are advanced techniques for solving Eq. 3.9 that are computationally much more efficient (and much more complex) than using successive approximations. An example is the “dlyap” function in the MATLAB Control Toolbox, which provides the desired covariance matrix directly: $C_{xx} = \text{dlyap}(\Phi, GCG^T)$.

The second stage is to compute the lagged state covariance sequence, $C_{xx}(t)$, and the lagged covariance between the state vector and the innovations vector, $C_{xe}(t) := \mathbf{E}\mathbf{x}_{t+t'}\mathbf{e}_{t'}^T$, for lags $t = 1, 2, \dots$. For the first lag,

$$C_{xx}(1) = \Phi C_{xx} \quad (3.11)$$

$$C_{xe}(1) = GC \quad (3.12)$$

For all subsequent lags, with $t = 1, 2, \dots$,

$$C_{xx}(t+1) = \Phi C_{xx}(t) \quad (3.13)$$

$$C_{xe}(t+1) = GC_{xe}(t) \quad (3.14)$$

The third stage is to compute the lagged covariance sequence of the observed process, $C_{yy}(t)$, for lags $t = 0, 1, 2, \dots$:

$$C_{yy}(0) = HC_{xx}H^T + C \quad (3.15)$$

$$C_{yy}(t) = HC_{xx}(t)H^T + HC_{xe}(t) \quad (3.16)$$

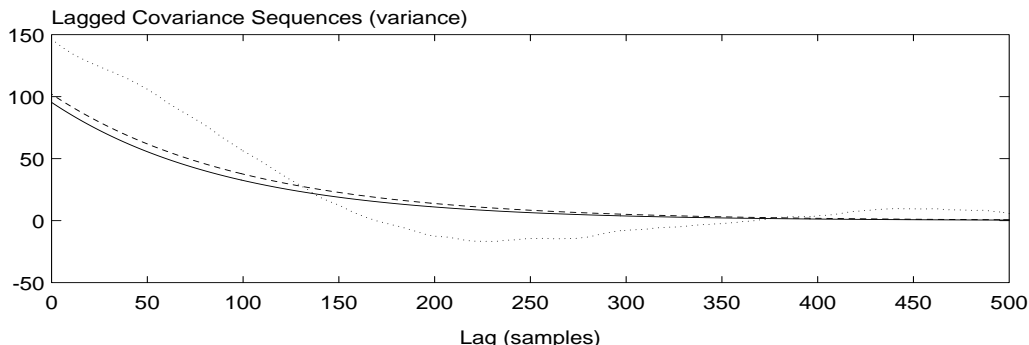


Figure 3.5: Comparison of true (dashed), model-based (solid), and nonparametrically estimated (dotted) lagged covariance sequences.

As an example of applying this algorithm, a long time series (from a first-order process having a correlation time of 100 samples) was cut into thirty short time series, each 50 samples long. A state-space model was then computed using the pooling approach in section 2.4, and the covariance sequence of the resulting model was computed using the above covariance algorithm. The covariance sequence of the true process was also computed and plotted for comparison with the empirically estimated covariance sequence in Fig. 3.5. The true (dashed) and model (solid) sequences are in close agreement. In contrast, the classical nonparametric estimate of the covariance sequence (dotted), which was computed from the original long time series of 1500 samples, has relatively low accuracy. The classical estimate is poor, because the data are highly colored, and because the number of parameters being estimated is comparable to the number of data in the time series.

3.2 State-Space Representations and Reduced-Order Modeling

The lagged covariance sequence of a process may be known, e.g., from a theoretical analysis or a previous modeling effort. How can a state-space model be developed from this covariance sequence? This is the problem of developing a state-space representation for the process from its covariance behavior. For some processes, a finite-dimensional state-space representation does not exist. In such cases, an approximate model is sought. Even if a state-space model exists that exactly represents the given covariances, a reduced-order approximation to this model may be needed. For example, a computationally constrained application may require that the state-space model contain no more than 5 state variables, but the original model has 10 state variables.

The baseline CV modeling algorithm is easily modified to use a finite lagged covariance sequence, $C_{yy}(t)$ for $t = 0, 1, 2, \dots, t_{\max}$, as input data, instead of a time series \mathbf{y}_t . The algorithm modification consists in using the given values for $C_{yy}(t)$ to construct the covariance matrices C_{pp} , C_{fp} , C_{ff} for the local past and future, instead of the use of sample covariances in the baseline algorithm. This is straight forward and has been implemented in a MATLAB function called “ssmc.”

The CV algorithm automatically develops a complete family of models having from 0 to n_{\max} state variables. Each model is optimal for the number of state variables it uses, where optimality

is defined as minimizing the cost functions described in Appendix A. Therefore, the selection of a reduced-order model is straight forward: that model having the acceptable number of states is selected from the family of candidates. The differences between this selected model and other models can be analyzed by computing and then comparing power spectra and covariance sequences for the models of interest.

3.3 Data Simulation

State-space models are easily used to generate time series. The resulting simulated data are useful for testing algorithms and for performing Monte Carlo simulations. The data generation algorithm presented here produces realizations of a Gaussian process having the state-space parameters (Φ, G, H, C) . More precisely, the algorithm computes a time series $\{\mathbf{y}_t\}_{t=1}^N$ consisting of N vectors that satisfy the following state equations with a given initial state vector:

$$\mathbf{x}_{t+1} = \Phi \mathbf{x}_t + G \mathbf{e}_t \quad (3.17)$$

$$\mathbf{y}_t = H \mathbf{x}_t + \mathbf{e}_t \quad (3.18)$$

$$C = \text{cov}(\mathbf{e}_t) \quad (3.19)$$

The algorithm consists of the following steps:

1. Select the state-space model of interest, (Φ, G, H, C)
2. Select the input parameter N , which is the number of output vectors to be generated
3. Select an initial $n \times 1$ state vector \mathbf{x}_1 . In many applications, an appropriate choice is $\mathbf{x}_1 = \mathbf{0}$
4. Compute the sequence of states and output vectors, \mathbf{x}_{t+1} and \mathbf{y}_t , for $t = 1, 2, \dots$, using equations 3.17 and 3.18, in which the \mathbf{e}_t 's are independent zero-mean Gaussian m -vectors having covariance matrix C . These vectors can be computed as $\mathbf{e}_t = [C]_L \mathbf{n}_t$, where $[C]_L$ is the left Cholesky factor of C (i.e., $C \equiv [C]_L [C]_L^T$), and $\{\mathbf{n}_t\}$ are independent gaussian m -vectors with covariance matrix I_m .

3.4 Kalman Filtering

A Kalman filter is a recursive algorithm for computing the expected values of the state variables and their covariance matrices, conditioned on the observed time series, given a linear Gaussian state-space model for the observations. Section 3.4.1 describes how a master state-space model can be constructed for time series that are linear combinations of already-modeled subprocesses, each having an innovations-form state-space representation. The resulting master model is in an augmented state-space form, not an innovations form. In section 3.4.2, the Kalman filtering equations are given for this augmented form of model.

3.4.1 State-space Models for Noisy Observations

The purpose of this section is to develop a state-space notation for noisy observations. The observed time series are modeled as linear combinations of processes for which innovations-form models have already been determined.

Suppose that a sensor (having m output channels) observes a linear transformation of a vector process \mathbf{y}'_t having the state-space parameters (Φ', G', H', C') . The sensor output is noisy and may contain interfering signals. Let process \mathbf{y}''_t denote this possible combination of noise and interfering signals, and let the state-space parameters of \mathbf{y}''_t be (Φ'', G'', H'', C'') . Then the sensor observations can be modeled as the following m -dimensional process:

$$\mathbf{z}_t = L'\mathbf{y}'_t + L''\mathbf{y}''_t \quad (3.20)$$

where matrix L' is $m \times m'$, vector \mathbf{y}'_t is $m' \times 1$, matrix L'' is $m \times m''$, and vector \mathbf{y}''_t is $m'' \times 1$. Each process can be expressed in terms of its own state vector and innovations process:

$$\mathbf{z}_t = L'H'\mathbf{x}'_t + L''H''\mathbf{x}''_t + L'\mathbf{e}'_t + L''\mathbf{e}''_t \quad (3.21)$$

where matrix H' is $m' \times n'$, vector \mathbf{x}'_t is $n' \times 1$, matrix H'' is $m'' \times n''$, vector \mathbf{e}'_t is $m' \times 1$, and vector \mathbf{e}''_t is $m'' \times 1$. To simplify Eq. 3.21 and reduce it to a standard state-space form, the matrices H and J are defined:

$$H := \begin{bmatrix} L'H'(m \times n') & L''H''(m \times n'') \end{bmatrix} \quad (3.22)$$

$$J := \begin{bmatrix} L'(m \times m') & L''(m \times m'') \end{bmatrix} \quad (3.23)$$

where matrix H is $m \times n$, matrix J is $m \times m_{12}$, and the number $m_{12} := m' + m''$. Now Eq. 3.21 can be rewritten as

$$\mathbf{z}_t = H\mathbf{x}_t + J\mathbf{n}_t \quad (3.24)$$

where the state and noise vectors are defined as follows:

$$\mathbf{x}_t := \begin{pmatrix} \mathbf{x}'_t \\ \mathbf{x}''_t \end{pmatrix} \quad (3.25)$$

$$\mathbf{n}_t := \begin{pmatrix} \mathbf{e}'_t \\ \mathbf{e}''_t \end{pmatrix} \quad (3.26)$$

It is straight forward to verify that these state and noise vectors satisfy the following state equation

$$\mathbf{x}_{t+1} = \Phi\mathbf{x}_t + G\mathbf{n}_t \quad (3.27)$$

where the transition matrix and gain matrix are defined as follows:

$$\Phi := \begin{bmatrix} \Phi'(n' \times n') & 0(n' \times n'') \\ 0(n'' \times n') & \Phi''(n'' \times n'') \end{bmatrix} \quad (3.28)$$

$$G := \begin{bmatrix} G'(n' \times m') & 0(n' \times m'') \\ 0(n'' \times m') & G''(n'' \times m'') \end{bmatrix} \quad (3.29)$$

$$C := \begin{bmatrix} C'(m' \times m') & 0(m \times m'') \\ 0(m'' \times m') & C''(m'' \times m'') \end{bmatrix} \quad (3.30)$$

A general state-space model for sensor data has been derived. For comparison with the innovations-form Eqs. 1.3 to 1.5, the state equations for noisy observations are reproduced here:

$$\mathbf{x}_{t+1} = \Phi\mathbf{x}_t + G\mathbf{n}_t \quad (3.31)$$

$$\mathbf{z}_t = H\mathbf{x}_t + J\mathbf{n}_t \quad (3.32)$$

$$C = \text{cov}(\mathbf{n}_t) \quad (3.33)$$

There is just one important difference between these equations and the innovations-form equations used previously in this document: the output equation has the output gain matrix J multiplying \mathbf{n}_t . Therefore, these are *not* innovations-form equations. Nevertheless, they are a compact representation that is suited for solving optimal filtering, detection, and classification problems.

3.4.2 Kalman Filtering Algorithm

Let \mathbf{x}_t denote the actual state of the observed process at time t and let $\mathbf{Z}_t := (\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_t)$ denote all observed vectors up to, and including, the current time. The *optimal Kalman filter* for the observed process is a recursive algorithm for computing the following conditional expected values when the process \mathbf{n}_t is Gaussian white noise. All of the following expected values are conditioned on observing \mathbf{Z}_t .

- The expected value of the current state vector, which is denoted $\mathbf{x}_t^+ := \mathbb{E}\mathbf{x}_t | \mathbf{Z}_t$
- The expected value of the next state vector, which is denoted $\mathbf{x}_{t+1}^- := \mathbb{E}\mathbf{x}_{t+1} | \mathbf{Z}_t$
- The covariance matrix of the current state vector, which is denoted $P_t^+ := \mathbb{E}(\mathbf{x}_t - \mathbf{x}_t^+)(\mathbf{x}_t - \mathbf{x}_t^+)^T | \mathbf{Z}_t$
- The covariance matrix of the next state vector, which is denoted $P_{t+1}^- := \mathbb{E}(\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^-)(\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^-)^T | \mathbf{Z}_t$

For initial conditions at time $t = 1$, the Kalman filter is given the initial expected state vector, \mathbf{x}_1^- , and the initial state covariance matrix, P_1^- . The filter is also given the model parameters (Φ, G, H, J, C) .

A Kalman filtering algorithm for observations modeled by Eqs. 3.31 to 3.33, which is based on [10], consists of the following two steps:

1. Compute the constant matrices M , L , and T :

$$M = \Phi - GCJ^T(JCJ^T)^{-1}H \quad (3.34)$$

$$L = G[C - CJ^T(JCJ^T)^{-1}JC]G^T \quad (3.35)$$

$$T = GCJ^T(JCJ^T)^{-1} \quad (3.36)$$

2. For $t = 1, 2, \dots$, compute the filter's innovation vector \mathbf{e}_t , its innovation covariance C_t , and the Kalman gain matrix K_t :

$$\mathbf{e}_t = \mathbf{z}_t - H\mathbf{x}_t^- \quad (3.37)$$

$$C_t = HP_t^-H^T + JCJ^T \quad (3.38)$$

$$K_t = P_t^-H^TC_t^{-1} \quad (3.39)$$

Compute the updated state mean \mathbf{x}_t^+ and covariance matrix P_t^+ :

$$\mathbf{x}_t^+ = \mathbf{x}_t^- + K_t\mathbf{e}_t \quad (3.40)$$

$$P_t^+ = P_t^- - K_tHP_t^- \quad (3.41)$$

Compute the propagated state mean \mathbf{x}_{t+1}^- and covariance matrix P_{t+1}^- :

$$\mathbf{x}_{t+1}^- = M\mathbf{x}_t^+ + T\mathbf{z}_t \quad (3.42)$$

$$P_{t+1}^- = MP_t^+M^T + L \quad (3.43)$$

It is worth noting that the covariance-type quantities (C_t, K_t, P_t^+, P_t^-) can be computed *before* the observations, \mathbf{z}_t , are available. This is useful when the accuracy of a proposed filter is being studied prior to its deployment. Also, these quantities can be precomputed so that the computations are minimized while the observations are processed.

3.5 Optimal Detection and Classification

This section describes how detection and classification problems can be formulated and solved within a probabilistic state-space framework. Although individual hypotheses may be represented by gaussian state-space models, *mixtures* of these models can be used to model extremely nongaussian states of uncertainty. As long as the mixtures are composed of gaussian state-space building blocks, the optimal solution to detection and classification problems can be computed without approximation using Kalman filtering.

3.5.1 Probabilistic Formulation

Suppose that a vector time series \mathbf{z}_t is observed for $t = 1, 2, \dots, N$. Also suppose that there are q hypotheses about the time series. According to the first hypothesis, \mathcal{H}_1 , the time series is generated by a state-space model with parameters (Φ_1, G_1, H_1, C_1) . According to the second hypothesis, \mathcal{H}_2 , the time series is generated by another state-space model with parameters (Φ_2, G_2, H_2, C_2) , and so on for all q hypotheses. In this formulation, the time series is considered to be generated by exactly one of these hypotheses. In a *classification* problem, the objective is to decide which hypothesis generated the time series. In a *detection* problem, a choice between just two possibilities is made. For detection, either there are just two hypotheses or the hypotheses are partitioned into two sets. One set consists of null hypotheses regarding the absence of target signals, while the other set consists of hypotheses about the presence of target signals. The detection problem is a special case of the classification problem. Therefore, the classification problem is discussed in the remainder of this section.

An *optimal classifier* is an algorithm that computes the posterior probability of each hypothesis, given the observations, and then makes a decision that maximizes the expected value of a utility function. The utility function is selected to represent the relative usefulness of different decisions, including the costs (negative utilities) of different types of incorrect decisions. The utility function, $U(\mathcal{D}, \mathcal{H})$, assigns a real number to each pair $(\mathcal{D}, \mathcal{H})$, where \mathcal{H} denotes that hypothesis which actually generated the observations, and \mathcal{D} denotes the decision that was chosen by the decision rule in the classifier. In this discussion we consider d possible decisions. The utility function can be any real-valued function defined for all possible pairs of \mathcal{D} and \mathcal{H} . Since the hypotheses are numbered from 1 to q , and the decisions are numbered from 1 to d , the variables \mathcal{H} and \mathcal{D} take these same integer values, e.g., $\mathcal{H} = 3$ and $\mathcal{D} = 4$ means that the third hypothesis generated the observations, but they were classified using the fourth decision. Therefore, without loss of generality, the utility function can be represented by a $d \times q$ matrix U , where $U(j, k)$ is the utility of decision j when the

data are generated by hypothesis k . The *expected utilities* of all possible decisions are represented by a column matrix \bar{U} , in which $\bar{U}(j)$ is the expected utility of the j th decision. The matrix \bar{U} is computed using the column matrix \mathbf{p} of posterior probabilities for the hypotheses:

$$\bar{U} = U\mathbf{p} \quad (3.44)$$

where $\mathbf{p}(k) = \Pr(\mathcal{H}_k|\mathbf{Z}_N)$. The optimal classification decision is $\mathcal{D} = j$, where

$$j = \arg \max_{k \in \{1, 2, \dots, d\}} \bar{U}(k) \quad (3.45)$$

As an example of a utility function, consider the case in which there are as many decisions as there are hypotheses, and the k th decision is to classify the data as coming from the k th hypothesis. The matrix $U = I$ states that each correct decision yields one unit of utility, and each incorrect decision yields zero utility. There is nothing in this utility function to distinguish one incorrect decision from another. This utility function yields the decision rule to choose that hypothesis which has the highest posterior probability. This utility function also minimizes the probability of error.

To compute the posterior probabilities of the hypotheses, $\Pr(\mathcal{H}_k|\mathbf{Z}_N)$ for $k = 1$ to q , Bayes's rule is used:

$$\Pr(\mathcal{H}_k|\mathbf{Z}_N) = \frac{\ell_k \Pr(\mathcal{H}_k)}{\text{normalizing constant}} \quad (3.46)$$

where $\Pr(\mathcal{H}_k)$ is the *prior probability* of the k th hypothesis, and ℓ_k is the *likelihood* of the k th hypothesis. The likelihood is defined to be that number which is obtained by substituting the particular time series at hand into the joint probability density of the observations under the hypothesis that the time series was generated by the k th model:

$$\ell_k := p(\mathbf{Z}_N|\mathcal{H}_k) \quad (3.47)$$

The normalizing constant in Eq. 3.46 is chosen so that the posterior probabilities sum to unity.

The likelihoods tend to be either very large or very small numbers, which may cause overflow and underflow during computations. Therefore, *log-likelihoods* are computed in practice, and a logarithmic version of Bayes's rule is used instead of Eq. 3.46. The log-likelihoods are all shifted prior to exponentiation to avoid underflow or overflow. (This shifting corresponds to multiplicative scaling of the likelihoods, which is always permissible, because the normalizing constant is selected as the last step in the computations.) The recommended version of Bayes's rule is as follows, for $k = 1$ to q :

$$L_k := \log \ell_k - \max_{j \in \{1 \dots q\}} \log \ell_j \quad (3.48)$$

$$\Pr(\mathcal{H}_k|\mathbf{Z}_N) = \frac{\exp(L_k) \Pr(\mathcal{H}_k)}{\text{normalizing constant}} \quad (3.49)$$

The prior probabilities are selected based on the physical interpretation of each hypothesis and its relation to the alternative hypotheses. There is no automatic way of choosing the priors, because they contain important information about the hypotheses before the current data set is processed. The priors are usually chosen so that, in the event that the time series at hand turns out to be invalid (e.g., because of a sensor failure), the priors are a reasonable representation of our uncertainties about the hypotheses.

3.5.2 Optimal Classifier Algorithm

An optimal classifier algorithm is presented in this section for the important case where each hypothesis is a Gaussian state-space model. This applies to a large class of problems, because Gaussian mixtures can accurately represent prior uncertainties that are extremely nongaussian. The most interesting parts of the optimal classifier are the log-likelihood computations. These can be efficiently implemented using q Kalman filters, one filter for each hypothesis, as explained in [11].

The idea behind using a Kalman filter to compute the likelihood of a Gaussian state-space model is straight forward. It is based on numbers called incremental likelihoods that can be computed efficiently by using the Kalman filter. To see this, the likelihood of the k th model, given the time series $\mathbf{Z}_N = (\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_N)$, is written in terms of *incremental likelihoods* of the form $p(\mathbf{z}_j | \mathbf{Z}_{j-1}, \mathcal{H}_k)$:

$$p(\mathbf{Z}_N | \mathcal{H}_k) \equiv p(\mathbf{z}_N, \mathbf{Z}_{N-1} | \mathcal{H}_k) \quad (3.50)$$

$$p(\mathbf{Z}_N | \mathcal{H}_k) \equiv p(\mathbf{z}_N | \mathbf{Z}_{N-1}, \mathcal{H}_k) p(\mathbf{Z}_{N-1} | \mathcal{H}_k) \quad (3.51)$$

$$p(\mathbf{Z}_N | \mathcal{H}_k) \equiv p(\mathbf{z}_N | \mathbf{Z}_{N-1}, \mathcal{H}_k) p(\mathbf{z}_{N-1} | \mathbf{Z}_{N-2}, \mathcal{H}_k) p(\mathbf{Z}_{N-2} | \mathcal{H}_k) \quad (3.52)$$

$$p(\mathbf{Z}_N | \mathcal{H}_k) \equiv p(\mathbf{z}_N | \mathbf{Z}_{N-1}, \mathcal{H}_k) p(\mathbf{z}_{N-1} | \mathbf{Z}_{N-2}, \mathcal{H}_k) \dots \\ p(\mathbf{z}_2 | \mathbf{Z}_1, \mathcal{H}_k) p(\mathbf{z}_1 | \mathcal{H}_k) \quad (3.53)$$

By defining the *incremental log-likelihood* of the k th hypothesis at time t ,

$$i_t(k) := \log p(\mathbf{z}_t | \mathbf{Z}_{t-1}, \mathcal{H}_k) \quad (\text{for } t > 1) \\ i_1(k) := \log p(\mathbf{z}_1 | \mathcal{H}_k) \quad (\text{for } t = 1) \quad (3.54)$$

the logarithm of the k th likelihood can be written as follows:

$$\log \ell_k \equiv \log p(\mathbf{Z}_N | \mathcal{H}_k) = \sum_{t=1}^N i_t(k) \quad (3.55)$$

The Kalman filter for the k th hypothesis is used to compute the incremental log-likelihoods in this sum, as explained in the following.

The Kalman filter computes the one-step-ahead expected value of the state vector, \mathbf{x}_t^- , and its covariance matrix, P_t^- , given the previous observations, \mathbf{Z}_{t-1} . From these quantities, the conditional expected value and covariance of \mathbf{z}_t , given \mathbf{Z}_{t-1} , is computed (where the fact is used that \mathbf{e}_t is Gaussian white noise with covariance C_k):

$$\mathbf{E}\mathbf{z}_t | \mathbf{Z}_{t-1} = H_k \mathbf{x}_t^- \quad (3.56)$$

$$\text{cov}(\mathbf{z}_t | \mathbf{Z}_{t-1}) = H_k P_t^- H_k^T + J_k C_k J_k^T \quad (3.57)$$

On the basis of these expressions, the incremental log-likelihood of the k th Gaussian hypothesis can be written as follows:

$$i_t(k) = -\frac{1}{2} \log \det(2\pi C_{zz,t}) - \frac{1}{2} (\mathbf{z}_t - \mathbf{m}_t)^T C_{zz,t}^{-1} (\mathbf{z}_t - \mathbf{m}_t) \quad (3.58)$$

$$\mathbf{m}_t := \mathbf{E}\mathbf{z}_t | \mathbf{Z}_{t-1} \quad (3.59)$$

$$C_{zz,t} := \text{cov}(\mathbf{z}_t | \mathbf{Z}_{t-1}) \quad (3.60)$$

In summary, the optimal classifier algorithm consists of the following steps.

1. Given: the vector time series $\mathbf{Z}_N = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_N\}$
2. Define q hypotheses about the observed data, $\{\mathcal{M}_i\}_{i=1}^q$, where the parameters of \mathcal{M}_i are $(\Phi_i, G_i, H_i, J_i, C_i, \mathbf{x}_1^-, P_1^-)$. The symbol \mathbf{x}_1^- denotes the expected value of the state vector at initial time $t = 1$, and P_1^- is the covariance matrix of the initial state vector. The state equations for these hypotheses are given by Eqs. 3.31 to 3.33.
3. Choose prior probabilities for the q hypotheses, $\{\Pr(\mathcal{H}_i)\}_{i=1}^q$ to model the initial uncertainty.
4. Define d candidate decisions and an associated $d \times q$ utility-function matrix U , where $U(j, k)$ is the utility of making decision j when the data are generated by hypothesis k .
5. For each the q hypotheses, $k = 1, 2, \dots, q$, compute the log-likelihood statistic $\log \ell_k = \sum_{t=1}^N i_t(k)$, where the incremental log-likelihoods, $i_t(k)$, are computed using a Kalman filter for the k th hypothesis. The equations are summarized here for convenient reference. For $t = 1, 2, \dots, N$,

$$i_t(k) = -\frac{1}{2} \log \det(2\pi C_{zz,t}) - \frac{1}{2} (\mathbf{z}_t - \mathbf{m}_t)^\top C_{zz,t}^{-1} (\mathbf{z}_t - \mathbf{m}_t) \quad (3.61)$$

$$\mathbf{m}_t := H_k \mathbf{x}_t^- \quad (3.62)$$

$$C_{zz,t} := H_k P_t^- H_k^\top + J_k C_k J_k^\top \quad (3.63)$$

where the expected value of the next state vector and its covariance matrix, \mathbf{x}_{t+1}^- and P_{t+1}^- , are computed recursively as follows:

$$\mathbf{x}_{t+1}^- = M \mathbf{x}_t^+ + T \mathbf{z}_t \quad (3.64)$$

$$P_{t+1}^- = M P_t^+ M^\top + L \quad (3.65)$$

$$\mathbf{x}_t^+ = \mathbf{x}_t^- + K_t \mathbf{e}_t \quad (3.66)$$

$$P_t^+ = P_t^- - K_t H P_t^- \quad (3.67)$$

In these equations, the following definitions have been used:

$$\mathbf{e}_t = \mathbf{z}_t - H \mathbf{x}_t^- \quad (3.68)$$

$$C_t = H P_t^- H^\top + J C J^\top \quad (3.69)$$

$$K_t = P_t^- H^\top C_t^{-1} \quad (3.70)$$

$$M = \Phi - G C J^\top (J C J^\top)^{-1} H \quad (3.71)$$

$$L = G [C - C J^\top (J C J^\top)^{-1} J C] G^\top \quad (3.72)$$

$$T = G C J^\top (J C J^\top)^{-1} \quad (3.73)$$

6. For each hypothesis, $k = 1, 2, \dots, q$, compute the posterior probability $\mathbf{p}(k) = \Pr(\mathcal{H}_k | Z_N)$:

$$L_k := \log \ell_k - \max_{j \in \{1, \dots, q\}} \log \ell_j \quad (3.74)$$

$$\mathbf{p}(k) = \frac{\exp(L_k) \Pr(\mathcal{H}_k)}{\text{normalizing constant}} \quad (3.75)$$

where the normalizing constant is selected so that $\sum_k \mathbf{p}(k) = 1$.

7. Compute the vector of expected utilities, \bar{U} , and choose the optimal decision, say the j th, which has the maximum expected utility:

$$\bar{U} = U\mathbf{p} \tag{3.76}$$

$$j = \arg \max_{k \in \{1, 2, \dots, d\}} \bar{U}(k) \tag{3.77}$$

Bibliography

- [1] White, J.V., *Stochastic State-Space Modeling*, Engineering Memorandum EM-2300, TASC, Reading MA, May 1984.
- [2] Sailor, R.V., "Signal Processing," in *Geoid and Its Geophysical Interpretations*, edited by Petr Vanek and Nikolaos T. Christou, CRC, 1993, pp 147-186.
- [3] Larimore, W., "System Identification, Reduced-Order Filtering and Modeling via Canonical Variate Analysis," *Proceedings of the 1983 American Control Conference*, San Francisco, June 1983, pp. 445-451.
- [4] White, J.V., "Stochastic State-Space Models from Empirical Data," *Proceedings ICASSP 83*, vol. 1, IEEE International Conference Acoustics, Speech and Signal Processing, Boston, April 1983, pp. 243-246.
- [5] Prasad, S. and Hari, K.V.S., "Improved ARMA Spectral Estimation Using the Canonical Variate Method," *IEEE Trans. Acoustics, Speech, Signal Proc.*, Vol. ASSP-35, No. 6, June 1987, pp 900-903.
- [6] Konishi, S. and Kitagawa, G., *Information Criteria and Statistical Modeling*, Springer, New York, 2008.
- [7] Akaike H., "Use of Statistical Models for Time Series Analysis," *Proceedings of ICASSP 86*, International Conference on Acoustics, Speech and Signal Processing, Tokyo, 1986, pp. 3147-3155.
- [8] Ben-Israel, A. and Greville, T.N.E., *Generalized Inverses: Theory and Applications*, John Wiley & Sons, New York, 1974, pp. 246, example 21.
- [9] Gelfand, I.M. and Yaglom, A.M., "Calculation of the Amount of Information About a Random Function Contained in Another Such Function," *Amer. Math. Soc. Translations*, series 2, vol. 12, pp. 199-246, 1959.
- [10] Anderson, B.D.O., and Moore, J.B., *Optimal Filtering*, Prentice-Hall, Inc., Englewood Cliffs, p. 120, 1979.
- [11] Scheppe, F., "Evaluation of Likelihood Functions for Gaussian Signals," *Transaction of the IEEE on Information Theory*, vol.. IT-11, no. 1, pp. 61-70, January 1965.

Appendix A

Performance Objectives

The CV approach to state-space modeling involves three stages of computation: (1) CV analysis of the local past and local future; (2) least-squares parameter estimation; (3) selection of the most accurate model. Each of these stages minimizes its own performance objective. This appendix defines these performance objectives and discusses their physical interpretations.

A.1 CV Analysis

In the CV analysis, estimated state vectors, containing n state variables, are computed from the local-past vectors:

$$\hat{\mathbf{x}}_t^n = L^n \mathbf{z}_t^- \quad (\text{A.1})$$

where L^n is an $n \times mp^*$ matrix, with either $p^* = p$ or $p^* = mp_y + qp_u$, depending on whether output modeling, or input-output modeling, is being considered.

A state vector contains all the information from the past of the observed process for the purpose of predicting the future. For Gaussian processes, a state vector is a linear function of the past. Therefore, the estimator in Eq. A.1 is linear. Because the analysis is based on finite time series of observations, the estimator uses a finite part of the past. The matrix L^n should be chosen to maximize the amount of information in $\hat{\mathbf{x}}_t^n$ about the future. However, there are an infinite number of linear transformations that do that. To define a unique optimal estimator, three nice canonical properties are put down as requirements:

1. Given the number of state variables, n , the vector $\hat{\mathbf{x}}_t^n$ is optimal for predicting the local future \mathbf{z}_t^+ . That is, for an appropriate $mf \times n$ matrix M^n , the following estimates of the local future are optimal in a weighted least-squares sense (which is defined below):

$$\hat{\mathbf{z}}_t^+ = M^n \hat{\mathbf{x}}_t^n \quad (\text{A.2})$$

2. The estimated state vector $\hat{\mathbf{x}}_t^n$ is *standardized* for convenience so that the n state variables are orthogonal to each other and have unit sample variances. This implies that the sample covariance matrix

$$S(\hat{\mathbf{x}}_t^n, \hat{\mathbf{x}}_t^n) = I_n \quad (\text{A.3})$$

where I_n is the n th-order identity matrix. (Note that S is defined in Eq. 1.8 using specific limits on the summation that prevent the computations from running off the ends of the time series.)

3. The state variables in $\hat{\mathbf{x}}_t^n$ occur in the order of their importance for predicting the future as measured by the error criterion defined below. The first state variable is most important, the second state variable is second most important, etc.

The error in the n -state estimate $\hat{\mathbf{z}}_t^+$ of the local future is

$$\epsilon^n = \hat{\mathbf{z}}_t^+ - \mathbf{z}_t^+ \quad (\text{A.4})$$

The resulting weighted sum-squared error is defined as

$$J_n := \sum_{t=p+1}^{N-f+1} \epsilon^{nT} W^{-1} \epsilon^n \quad (\text{A.5})$$

The weighting matrix W is chosen so that, asymptotically as the number of observations N increases, minimizing J_n (with respect to matrices L^n and M^n) is equivalent to maximizing the amount of information in $\hat{\mathbf{x}}_t^n$ about the local future \mathbf{z}_t^+ . This objective is achieved by choosing W to be the sample covariance matrix of the local future:

$$W = S(\mathbf{z}_t^+, \mathbf{z}_t^+) \quad (\text{A.6})$$

Note that this choice of W makes J_n independent of the units in which the observations are expressed. Also, J_n can be thought of as the sum of squares of the *fractional errors*.

A.2 Least-Squares Parameter Estimation

Two least-squares problems are solved in computing the model parameter matrices (Φ^n, G^n, H^n, C^n) for an n -state model. The first problem is to determine H^n in the model output equation

$$\mathbf{y}_t = H^n \hat{\mathbf{x}}_t^n + \mathbf{e}_t^n \quad (\text{A.7})$$

where the innovation vector \mathbf{e}_t^n is the error in using H^n to estimate \mathbf{y}_t from $\hat{\mathbf{x}}_t^n$. Given the estimated states, H^n is selected to minimize the total energy in the innovation vectors. Specifically, H^n is selected to minimize

$$J_n^{(1)} := \sum_{t=p+1}^{N-f+1} \mathbf{e}_t^{nT} \mathbf{e}_t^n \quad (\text{A.8})$$

It is interesting to note that a weighted sum of innovation energy could also be used to define $J_n^{(1)}$, but the resulting H^n would be the same, regardless of which positive-definite weighting matrix were used. The reason the weighting matrix is unimportant, is that this is an unconstrained minimization in which there are enough parameters in H^n to minimize the total energy in each element of \mathbf{e}_t^n independently of the other elements.

Given the estimated state vectors $\hat{\mathbf{x}}_t^n$ from the CV analysis and the innovation vectors resulting from the optimal choice of H^n in Eq. A.7, the parameter matrices Φ^n and G^n are chosen to minimize $J_n^{(2)}$, which is the total energy in the residual vectors \mathbf{r}_t^n in the following equation:

$$\hat{\mathbf{x}}_{t+1}^n = \Phi^n \hat{\mathbf{x}}_t^n + G^n \mathbf{e}_t^n + \mathbf{r}_t^n \quad (\text{A.9})$$

$$J_n^{(2)} := \sum_{t=p+1}^{N-f+1} \mathbf{r}_t^{nT} \mathbf{r}_t^n \quad (\text{A.10})$$

The reason that the residual vectors in Eq. A.9 are nonzero is that the estimated state vectors, $\hat{\mathbf{x}}_t$, only approximately satisfy the state propagation Eq. 1.3. The matrices Φ and G are selected in Eq. A.9 to minimize the residual error that results from using the estimated states as if they were actual state vectors.

Although $J_n^{(2)}$ is an unweighted error measure, there is no loss of generality in using it, because the optimal values of Φ^n and G^n are independent of whichever positive-definite weighting matrix might be used.

A.3 Model Selection

The goal of model selection is to choose from the family of candidate models, $\{\Phi^n, G^n, H^n, C^n\}_{n=0}^{n_{\max}}$, that particular model which is the most accurate. The accuracy of a candidate model is measured in information-theoretic terms by the *discrimination information* (DI) for recognizing the truth and discriminating against the candidate. Therefore, DI is a measure of model *inaccuracy*. The goal is to select that model which minimizes the DI. (DI is also known as Kullback information, Kullback-Leibler information, cross entropy, and directed divergence.) Unfortunately, DI can not be computed when the truth is unknown. Therefore, an unbiased estimate of DI, called the Akaike Information Criterion (AIC), is computed instead for models with Gaussian innovations. Actually, the AIC is an affine transformation of estimated DI, which is sufficient for identifying which model is expected to be most accurate.

For a Gaussian state-space model having n state variables, the AIC is computed as follows:

$$\text{AIC} = (N - p) \log_e[\det(C^n)] + 4mn + m(m + 1) \quad (\text{A.11})$$

where C^n is the innovations covariance matrix of the model, and $(N - p)$ is the number of innovations vectors that were used in computing C^n . The first term on the right side of Eq. A.11 measures how small the innovations are on average, which is a measure of how well the model can do one-step-ahead predictions on the observations. The second term increases with the complexity of the model as measured by the number of state variables, and the third term is the number of parameters in the innovations covariance matrix. These two terms compensate for the artificially small value of the first term, which is caused by the fact that the model parameters were chosen to minimize the sample variances of the innovations. If a new time series, independent of the time series used for computing model parameters, were available for computing the innovations covariance matrix C^n , then the AIC would not be used, because the first term by itself is then an unbiased estimate of model inaccuracy.

Appendix B

Derivation of Baseline CV Algorithm

B.1 Useful Notation

This section establishes useful matrix notation for doing linear least-squares calculations and proofs. Consider the vector time series \mathbf{a}_t and a related time series \mathbf{b}_t , for $t = 1, 2, \dots, N$. The vectors are real and have dimensions m_a and m_b , respectively. Let

$$\hat{\mathbf{b}}_t = M\mathbf{a}_t \tag{B.1}$$

where $\hat{\mathbf{b}}_t$ denotes an *estimate* of \mathbf{b}_t , and the *estimator* M is an $m_b \times m_a$ matrix. The mean-square error (MSE) of this estimate, *over the domain* $\mathcal{D} = \{t_{\min}, t_{\min} + 1, \dots, t_{\max}\}$, is defined as

$$\text{MSE}_{\mathcal{D}} := \frac{1}{|\mathcal{D}|} \sum_{t=t_{\min}}^{t_{\max}} \mathbf{e}_t^T \mathbf{e}_t \tag{B.2}$$

where the error $\mathbf{e}_t := \hat{\mathbf{b}}_t - \mathbf{b}_t$, and $|\mathcal{D}| := t_{\max} - t_{\min} + 1$.

Corresponding to this MSE, it is useful to define the $m_a \times m_b$ *sample covariance matrix with respect to \mathcal{D} for the time series \mathbf{a}_t and \mathbf{b}_t* :

$$S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{b}_t) := \frac{1}{|\mathcal{D}|} \sum_{t \in \mathcal{D}} \mathbf{a}_t \mathbf{b}_t^T \tag{B.3}$$

This notation draws attention to the domain \mathcal{D} , because it plays an important role in the selfconsistent formulation of the state-space modeling problem.

The sample covariance operator $S_{\mathcal{D}}$ has the following properties (where tr denotes the trace of a matrix, A and B are matrices of appropriate sizes, and \mathbf{c}_t denotes a vector time series of appropriate dimension and length):

- The MSE property:

$$\text{tr} S_{\mathcal{D}}(\mathbf{e}_t, \mathbf{e}_t) \equiv \text{MSE}_{\mathcal{D}} \tag{B.4}$$

- The transpose property:

$$S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{b}_t) \equiv S_{\mathcal{D}}(\mathbf{b}_t, \mathbf{a}_t)^T \tag{B.5}$$

- The transformation property:

$$S_{\mathcal{D}}(A\mathbf{a}_t, B\mathbf{b}_t) \equiv AS_{\mathcal{D}}(\mathbf{a}_t, \mathbf{b}_t)B^{\top} \quad (\text{B.6})$$

- The linearity property:

$$S_{\mathcal{D}}(\mathbf{a}_t + \mathbf{b}_t, \mathbf{c}_t) \equiv S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{c}_t) + S_{\mathcal{D}}(\mathbf{b}_t, \mathbf{c}_t) \quad (\text{B.7})$$

B.2 Fundamentals of Least-Squares Estimation

The CV approach to state-space modeling is based on the fundamental theorem of linear least-squares estimation for time series:

Theorem 1 *Given: (1) Two time series, \mathbf{a}_t and \mathbf{b}_t , for $t \in \mathcal{T} := \{1, 2, \dots, N\}$, having dimensions m_a and m_b , respectively. (2) A domain $\mathcal{D} \subseteq \mathcal{T}$. (3) The matrix $M = S_{\mathcal{D}}(\mathbf{b}_t, \mathbf{a}_t)S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{a}_t)^{\dagger}$, where \dagger denotes the Moore-Penrose pseudo inverse.*

Assertions: (1) The estimate $\hat{\mathbf{b}}_t = M\mathbf{a}_t$, for $t \in \mathcal{D}$, is optimal among all linear estimates in the sense that the resulting $MSE_{\mathcal{D}}$ is minimized. (2) Among all such optimal estimators, M is unique in having the minimum Frobenius norm, $\|M\|_{\text{F}} := \sqrt{\text{tr}MM^{\top}}$. (3) The observed vector \mathbf{a}_t and the resulting estimation error \mathbf{e}_t are orthogonal: $S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{e}_t) = 0$. (4) The sample error-covariance matrix is given by $S_{\mathcal{D}}(\mathbf{e}_t, \mathbf{e}_t) = S_{\mathcal{D}}(\mathbf{b}_t, \mathbf{b}_t) - S_{\mathcal{D}}(\mathbf{b}_t, \mathbf{a}_t)S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{a}_t)^{\dagger}S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{b}_t)$.

When the time series \mathbf{a}_t has an invertible sample covariance matrix, the following corollary is useful.

Corollary 1 *Given: (1) The existence of the inverse matrix $S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{a}_t)^{-1}$. (2) The matrix $M = S_{\mathcal{D}}(\mathbf{b}_t, \mathbf{a}_t)S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{a}_t)^{-1}$.*

Assertions: (1) The estimate $\hat{\mathbf{b}}_t = M\mathbf{a}_t$, for $t \in \mathcal{D}$, is both optimal and unique among all linear estimates in the sense that the resulting $MSE_{\mathcal{D}}$ is minimized and there is no other linear estimator that achieves this $MSE_{\mathcal{D}}$. (2) The sample error-covariance matrix satisfies the equation $S_{\mathcal{D}}(\mathbf{e}_t, \mathbf{e}_t) = S_{\mathcal{D}}(\mathbf{b}_t, \mathbf{b}_t) - S_{\mathcal{D}}(\mathbf{b}_t, \mathbf{a}_t)S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{a}_t)^{-1}S_{\mathcal{D}}(\mathbf{a}_t, \mathbf{b}_t)$.

When there is a rank constraint on a linear estimator, the following theorem on *best matrix approximations of given rank* from [8] applies. In this theorem, $R_r^{m \times n}$ denotes the set of real $m \times n$ matrices of rank r , and $\|\cdot\|_{\text{F}}$ denotes the Frobenius norm.

Theorem 2 *For a given $A \in R_r^{m \times n}$ and an integer k , $1 \leq k \leq r$, a best rank- k approximation of A is a matrix $A_{(k)} \in R_k^{m \times n}$ satisfying*

$$\|A - A_{(k)}\|_{\text{F}} = \inf_{X \in R_k^{m \times n}} \|A - X\|_{\text{F}} \quad (\text{B.8})$$

Let $A = U\Sigma V^{\top}$ be the singular value decomposition of A , with the singular values in the diagonal $m \times n$ matrix Σ arranged in descending order, with the largest $\sigma_1 = \Sigma(1, 1)$. Let $U_{(k)}$ and $V_{(k)}$ denote the submatrices consisting of the first k columns of U and V , respectively. Let $\Sigma_{(k)}$ be the diagonal $k \times k$ submatrix consisting of upper left corner of Σ .

Then a best rank- k approximation of A is

$$A_{(k)} = U_{(k)}\Sigma_{(k)}V_{(k)}^{\top} \quad (\text{B.9})$$

which is unique if, and only if, the k th and the $(k+1)$ st singular values of A are distinct:

$$\sigma_k \neq \sigma_{k+1} \quad (\text{B.10})$$

The approximation error of $A_{(k)}$ is

$$\|A - A_{(k)}\|_F = \left(\sum_{i=k+1}^r \sigma_i^2 \right)^{1/2} \quad (\text{B.11})$$

B.3 Derivation of Baseline CV Algorithm

The baseline CV algorithm for state-space modeling is based on corollary 1 and theorem 2. A derivation of the first two stages of the algorithm is presented in sections B.3.1 and B.3.2.

B.3.1 CV Analysis

Given: (1) The vector time series \mathbf{y}_t for $t = 1$ to N , where each vector has dimension m . (2) Two positive integers, p and f , which define the reach of the local past and local future, respectively. (3) A positive integer, $n_{\max} \leq \min(mp, mf)$, which is the maximum number of state variables being considered. The objective of the CV analysis is to determine a sequence of estimated state vectors corresponding to the time series. The estimated vectors are to be linear functions of the local past, as expressed by Eq. A.1, because this is a practical and reasonable approach when using finite time series for linear modeling. The estimated state vectors are to be *optimal* in the sense that when n state variables are considered, they minimize the weighted sum-squared error criterion defined by Eq. A.5 (assuming that the best possible matrix M^n is used in Eq. A.2). The estimated state vectors are also to be *canonical* (for mathematical convenience) in the sense that they satisfy Eqs. A.3 and have their states arranged in the order of their importance for predicting the local future.

The following notation is used for the covariances involving the local past and future:

$$C_{pp} := S_{\mathcal{D}}(\mathbf{z}_t^-, \mathbf{z}_t^-) \quad (\text{B.12})$$

$$C_{fp} := S_{\mathcal{D}}(\mathbf{z}_t^+, \mathbf{z}_t^-) \quad (\text{B.13})$$

$$C_{ff} := S_{\mathcal{D}}(\mathbf{z}_t^+, \mathbf{z}_t^+) \quad (\text{B.14})$$

The derivation of the CV analysis starts by replacing the weighted error criterion in Eq. A.5 with an equivalent unweighted criterion. This is done by introducing the *standardized* local past, \mathbf{s}_t^- , local future, \mathbf{s}_t^+ , and estimated local future, $\hat{\mathbf{s}}_t^+$. These quantities are defined using the left Cholesky factors of the sample covariance matrices of the local past and future:

$$\mathbf{s}_t^- := [C_{pp}]_L^{-1} \mathbf{z}_t^- \quad (\text{B.15})$$

$$\mathbf{s}_t^+ := [C_{ff}]_L^{-1} \mathbf{z}_t^+ \quad (\text{B.16})$$

$$\hat{\mathbf{s}}_t^+ := [C_{ff}]_L^{-1} \hat{\mathbf{z}}_t^+ \quad (\text{B.17})$$

The new vectors, \mathbf{s}_t^- and \mathbf{s}_t^+ , have identity sample covariance matrices, as the following calculation for the local past demonstrates.

$$S_{\mathcal{D}}(\mathbf{s}_t^-, \mathbf{s}_t^-) = S_{\mathcal{D}}([C_{pp}]_L^{-1} \mathbf{z}_t^-, [C_{pp}]_L^{-1} \mathbf{z}_t^-) \quad (\text{B.18})$$

$$= [C_{pp}]_L^{-1} S_{\mathcal{D}}(\mathbf{z}_t^-, \mathbf{z}_t^-) ([C_{pp}]_L^{-1})^T \quad (\text{B.19})$$

$$= [C_{pp}]_L^{-1} [C_{pp}]_L [C_{pp}]_L^T ([C_{pp}]_L^{-1})^T \quad (\text{B.20})$$

$$= I_{mp} \quad (\text{B.21})$$

Because the weighting matrix W in Eq. A.5 is defined by Eq. A.6, the performance objective can be rewritten as the equivalent unweighted performance objective

$$J_n = \sum_{t=p+1}^{N-f+1} (\hat{\mathbf{s}}_t^+ - \mathbf{s}_t^+)^T (\hat{\mathbf{s}}_t^+ - \mathbf{s}_t^+) \quad (\text{B.22})$$

where the estimate $\hat{\mathbf{s}}_t^+$ is a linear transformation of the past:

$$\hat{\mathbf{s}}_t^+ = P \mathbf{s}_t^- \quad (\text{B.23})$$

Choosing the matrix P in Eq. B.23 to minimize J_n in Eq. B.22 is equivalent to minimizing the weighted error J_n in Eq. A.5, provided that the matrix P is constrained to have a rank of n . From corollary 1 (and recalling that the sample covariance matrix of \mathbf{s}_t^- is an identity matrix), it is known that the optimal P with no rank constraint is

$$P = S_{\mathcal{D}}(\mathbf{s}_t^+, \mathbf{s}_t^-) S_{\mathcal{D}}(\mathbf{s}_t^-, \mathbf{s}_t^-)^{-1} \quad (\text{B.24})$$

$$P = S_{\mathcal{D}}(\mathbf{s}_t^+, \mathbf{s}_t^-) \quad (\text{B.25})$$

Substituting the definitions of the standardized past and future from Eqs. B.15 and B.16 into Eq. B.25 and using Eq. B.6, leads to

$$P = S_{\mathcal{D}}([C_{ff}]_L^{-1} \mathbf{z}_t^+, [C_{pp}]_L^{-1} \mathbf{z}_t^-) \quad (\text{B.26})$$

$$P = [C_{ff}]_L^{-1} [C_{fp}] ([C_{pp}]_L^{-1})^T \quad (\text{B.27})$$

To find the optimal P for the present problem, for which the rank of P is n , the P in Eq. B.27 is factored using the singular value decomposition:

$$P = U \Sigma V^T \quad (\text{B.28})$$

where U is an $mf \times mf$ orthogonal matrix, V is an $mp \times mp$ orthogonal matrix, and Σ is an $mf \times mp$ matrix. The diagonal elements, $\Sigma(k, k)$ for $k = 1$ to $\min(mf, mp)$, are the singular values of P arranged in order, from largest to smallest. The other elements of Σ are zero. From theorem 2, the best rank- n approximation to the unconstrained estimator is expressed in terms of submatrices:

$$P_n = U_n \Sigma_n V_n^T \quad (\text{B.29})$$

where Σ_n is the upper-left $n \times n$ submatrix of Σ . The $mp \times n$ submatrix V_n contains the first n columns of V , and the $mf \times n$ submatrix U_n contains the first n columns of U .

By replacing the P in Eq. B.23 with the P_n defined in Eq. B.29, replacing \mathbf{s}_t^- and $\hat{\mathbf{s}}_t^+$ by their definitions in Eqs. B.15 and B.17, and then solving Eq. B.23 for the predicted local future, the optimal rank- n predictor of \mathbf{z}_t^+ is found to satisfy the following equation:

$$\hat{\mathbf{z}}_t^+ = [C_{ff}]_L U_n \Sigma_n V_n^T [C_{pp}]_R^{-1} \mathbf{z}_t^- \quad (\text{B.30})$$

Equation B.30 gives the optimal estimator of the future, given the past, which was originally expressed (in Eqs. A.1 and A.2) as

$$\hat{\mathbf{z}}_t^+ = M^n L^n \mathbf{z}_t^- \quad (\text{B.31})$$

By comparing the Eq. B.31 with Eq. B.30, it can be verified that the optimal prediction matrices M^n and L^n are

$$L^n = V_n^T [C_{pp}]_R^{-1} \quad (\text{B.32})$$

$$M^n = [C_{ff}]_L U_n \Sigma_n \quad (\text{B.33})$$

Verifying that these are the correct expressions for L^n and M^n consists in checking that the estimated state vector, which is defined for $t = p + 1$ to $N + 1$ as

$$\hat{\mathbf{x}}_t^n = L^n \mathbf{z}_t^- \quad (\text{B.34})$$

has the following two properties: (1) its sample covariance matrix $S_{\mathcal{D}}(\hat{\mathbf{x}}_t^n, \hat{\mathbf{x}}_t^n)$ is the $n \times n$ identity matrix; and (2) it contains the states arranged with the most important state first, the second most important state second, etc.

In the parlance of canonical-variates theory, the estimated states defined by Eq. B.34 are canonical variates, and the singular values, $\Sigma(k, k)$ for $k = 1$ to n , are the first n canonical correlations between the local past and future. For Gaussian processes, the mutual information rate, $\text{MI}(n)$ (measured in bits per sample), between the local past and future conveyed by $\hat{\mathbf{x}}_t^n$ is given asymptotically, as the length of the time series increases, by the following formula [9]:

$$\text{MI}(n) = - \sum_{k=1}^n \log_2 \left(1 - \Sigma(k, k)^2 \right) / 2 \quad (\text{B.35})$$

The outputs of the CV analysis are the estimated canonical state vectors $\hat{\mathbf{x}}_t^n$ for times $t = p + 1$ to $N + 1$ and for state orders $n = 1$ to n_{\max} .

B.3.2 Least-Squares Parameter Estimation

The object of this processing is to use the estimated state vectors from the canonical analysis and the observed time series, \mathbf{y}_t for $t = 1$ to N , to estimate the parameters of a family of state-space models in innovations form. Each model corresponds to a different choice for n , the number of state variables. The model parameter matrices (Φ^n, G^n, H^n, C^n) are estimated by using least squares, as explained in the following.

Estimating H^n

The model output equation is

$$\mathbf{y}_t = H^n \hat{\mathbf{x}}_t^n + \mathbf{e}_t^n \quad (\text{B.36})$$

in which H^n is to be selected as the optimal estimator of \mathbf{y}_t , given $\hat{\mathbf{x}}_t^n$, and \mathbf{e}_t is the estimation error. The MSE to be minimized is given by $J_n^{(1)}$ in Eq. A.8. From corollary 1, it follows that

$$H^n = S_{\mathcal{D}}(\mathbf{y}_t, \hat{\mathbf{x}}_t^n) S_{\mathcal{D}}(\hat{\mathbf{x}}_t^n, \hat{\mathbf{x}}_t^n)^{-1} \quad (\text{B.37})$$

But the sample covariance of $\hat{\mathbf{x}}_t^n$ is an identity matrix. Therefore

$$H^n = S_{\mathcal{D}}(\mathbf{y}_t, \hat{\mathbf{x}}_t^n) \quad (\text{B.38})$$

Having determined H^n in Eq. B.38, and given \mathbf{y}_t and $\hat{\mathbf{x}}_t^n$, the innovations sequence \mathbf{e}_t is computed using Eq. B.36 for $t = p + 1$ to N :

$$\mathbf{e}_t = \mathbf{y}_t - H^n \hat{\mathbf{x}}_t^n \quad (\text{B.39})$$

Estimating Φ^n and G^n

The next step is to determine Φ^n and G^n , which occur in the state propagation equation of the model. This equation is

$$\hat{\mathbf{x}}_{t+1}^n = \Phi^n \hat{\mathbf{x}}_t^n + G^n \mathbf{e}_t^n + \mathbf{r}_t^n \quad (\text{B.40})$$

where the residual term \mathbf{r}_t^n is the error in optimally estimating $\hat{\mathbf{x}}_{t+1}^n$ from $\hat{\mathbf{x}}_t^n$ and \mathbf{e}_t^n . The MSE to be minimized is given as $J_n^{(2)}$ in Eq. A.10. It is useful to note that $\hat{\mathbf{x}}_t^n$ and \mathbf{e}_t^n are orthogonal to each other ($S_{\mathcal{D}}(\hat{\mathbf{x}}_t^n, \mathbf{e}_t^n) = 0$), because \mathbf{e}_t^n is the error of an optimal estimator (H^n in Eq. B.36) that estimates \mathbf{y}_t from $\hat{\mathbf{x}}_t^n$. Therefore, Φ^n and G^n , viewed as optimal estimators operating on orthogonal data sets, can be computed independently of each other. From corollary 1, it follows that

$$\Phi^n = C_{x_1x}^n \quad (\text{B.41})$$

$$G^n = C_{x_1e}^n C_{ee}^{-1} \quad (\text{B.42})$$

$$C_{x_1x}^n := S_{\mathcal{D}}(\hat{\mathbf{x}}_{t+1}^n, \hat{\mathbf{x}}_t^n) \quad (\text{B.43})$$

$$C_{x_1e}^n := S_{\mathcal{D}}(\hat{\mathbf{x}}_{t+1}^n, \mathbf{e}_t^n) \quad (\text{B.44})$$

$$C_{ee}^n := S_{\mathcal{D}}(\mathbf{e}_t^n, \mathbf{e}_t^n) \quad (\text{B.45})$$

Estimating C^n

Although a reasonable way of defining the innovations covariance of the n -state model is to set $C^n = C_{ee}^n$, a more accurate algorithm is used in the baseline CV algorithm. This more accurate algorithm takes into account the fact that the innovations process \mathbf{e}_t appearing in Eqs. 1.3 to 1.4 is not quite the same as the process \mathbf{e}_t^n defined by Eq. B.39. What is needed is the best estimate of the covariance of the noise in Eqs. 1.3 to 1.4 for different numbers of state variables. To this end, these equations are rearranged so that \mathbf{y}_t is the *input* process and \mathbf{e}_t^n is the *output*. This is done as follows. Solve Eq. 1.4 for \mathbf{e}_t :

$$\mathbf{e}_t = \mathbf{y}_t - H\mathbf{x}_t \quad (\text{B.46})$$

Substitute this expression for \mathbf{e}_t in Eq. 1.3 and collect terms:

$$\mathbf{x}_{t+1} = \Phi\mathbf{x}_t + G(\mathbf{y}_t - H\mathbf{x}_t) \quad (\text{B.47})$$

$$\mathbf{x}_{t+1} = (\Phi - GH)\mathbf{x}_t + G\mathbf{y}_t \quad (\text{B.48})$$

Finally, to emphasize that Eqs. B.46 and B.48 will be used repeatedly, with different numbers of state variables, substitute \mathbf{e}_t^n for \mathbf{e}_t and use subscripts on the coefficient matrices. The resulting equations for computing \mathbf{e}_t^n from the time series \mathbf{y}_t are then

$$\mathbf{x}_{t+1}^n = (\Phi^n - G^n H^n)\mathbf{x}_t^n + G^n \mathbf{y}_t \quad (\text{B.49})$$

$$\mathbf{e}_t^n = \mathbf{y}_t - H^n \mathbf{x}_t^n \quad (\text{B.50})$$

Equations B.49 and B.50 are used, with initial condition

$$\mathbf{x}_{p+1}^n = \hat{\mathbf{x}}_{p+1}^n \quad (\text{B.51})$$

to compute \mathbf{e}_t^n for $t = p + 1$ to N . Then the innovations covariance matrix for the n -state model is computed as the sample covariance

$$C^n = S_{\mathcal{D}}(\mathbf{e}_t^n, \mathbf{e}_t^n) \quad (\text{B.52})$$