Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER Journal of Economic Dynamics & Control 27 (2003) 1317–1333
JOURNAL OF Economic Dynamics & Control
www.elsevier.com/locate/econbase

Computing observation weights for signal extraction and filtering

Siem Jan Koopman[a,*], Andrew Harvey[b]

[a] *Department of Econometrics, Free University Amsterdam, De Boelelaan 1105, NL-1081 HV Amsterdam, Netherlands*
[b] *Faculty of Economics and Politics, University of Cambridge, Sidgwick Avenue, Cambridge CB3 9DD, UK*

Abstract

We present algorithms for computing the weights implicitly assigned to observations when estimating unobserved components, by filtering or smoothing, using a model in state space form. The algorithms are based on recursions derived from the Kalman filter and associated smoother. Since the method applies to any model with a linear state space form, it is not restricted to time-invariant systems and it can be used in a number of situations where other methods break down. Applications include the comparison of signal extraction weights with those used by nonparametric procedures and the computation of weights assigned to observations in making forecasts. © 2002 Elsevier Science B.V. All rights reserved.

JEL classification: C15; C22

Keywords: Kalman filter; Kernels; Nonparametric regression; State space; Vector autoregression

1. Introduction

Unobserved component (UC) time series models are used for prediction and signal extraction. For models in state space form these operations are carried out by the Kalman filter and the associated smoother as described in, amongst others, Anderson and Moore (1979), Harvey (1989), Kitagawa and Gersch (1996), Shumway and Stoffer (2000) and Durbin and Koopman (2001). In doing so, weights are implicitly assigned to the various observations in making predictions and forming smoothed estimates of

* Corresponding author. Tel.: +31-20-4446019; fax: +31-20-4446020.
E-mail addresses: s.j.koopman@feweb.vu.nl (S.J. Koopman), andrew.harvey@econ.cam.ac.uk (A. Harvey).

PII: S0165-1889(02)00061-1

the components at different points in time. The reason for wishing to compute these weights is that an examination of their patterns can be very informative. It helps the user to understand what a model actually does, and it enables comparisons to be made with other methods.

For time-invariant UC models, the Wiener–Kolmogorov approach provides general expressions, in terms of autocovariance generating functions, which can be used as the basis for finding filtering and smoothing weights. These expressions can be solved analytically in certain cases to give explicit formulae for the weights; see, for example, Whittle (1983, p. 69) and Harvey and Koopman (2000). More generally, procedures such as those developed by Cleveland and Tiao (1976) and Burman (1980) can be used to compute numerical values.

When signal extraction is carried out by a method which does not use weights directly, their implicit numerical values can often be computed simply by running the algorithm on a set of artificially constructed observations all of which are zero except for the one at the point of interest which is unity. We call this device the *zero–one method*. Unfortunately, it does not generally work when systems are time-varying, nor does it give accurate results for time-invariant models with correlated components when interest centres on the weights near the beginning or end of the sample. These points are elaborated upon in Section 2 of the paper.

The principal aim of this article is to present algorithms for computing the exact weights for smoothed and filtered estimators at any point in the sample. These algorithms consist of recursions derived from the Kalman filter and its associated smoother and so they apply to any model with a linear state space form. Unlike the algorithms derived from the Wiener–Kolmogorov approach, they are not restricted to time-invariant systems. Multivariate series may be handled straightforwardly and so the influence of any variable on the estimator of any linear combination of the state vector may be determined. The generality of the state space form means that the algorithm can deal with situations such as those involving heteroscedasticity and irregularly spaced observations.

The Kalman filter and its associated smoother give minimum mean square linear estimators (MMSLEs) of elements, or linear combinations, of the state vector at all points in time. The way in which they work is reviewed in Section 3. Section 4, which is the heart of the paper, presents the algorithms for computing the weights implicitly used by these filtering and smoothing operations.

Examples are presented in Section 5. The first concerns the basic random walk plus noise model and the graphs show the kind of patterns which are obtained when observations are missing. The weights may similarly be found for more complex UC models. One reason for doing this is that when nonparametric regressions are used to estimate trends, it is interesting to compare the weighting patterns of their associated kernels, or moving averages, with the weighting patterns implied by various UC models. The second example takes this point further by noting that the state space form allows a UC model formulated in continuous time to handle irregularly spaced observations. This enables the stochastic trend in a UC time series model to capture the relationship between two cross-sectional variables. Our algorithm allows the weights to be determined.

A somewhat different application of the weighting algorithm stems from the fact that the predictive filtering weights can be interpreted as the coefficients of an (vector) autoregressive (VAR) representation of a model. The final example illustrates how the algorithm gives autoregressive weights for the two variables in a bivariate random walk plus noise model. It then shows how the procedure may be extended to yield the vector error-correction model (VECM) representation of a UC model with common stochastic trends.

The algorithms were written using the software package *SsfPack*. Details of implementation and example programs are given in the appendix.

2. The zero–one method

This section discusses the idea behind the zero–one method for computing signal extraction weights and analyses the conditions under which it can be correctly applied.

Suppose we have a procedure for computing a series of signals, $m_{t|T}$, $t = 1,\ldots,T$, each one being based on the full set of observations, y_t, $t = 1,\ldots,T$. Assuming the procedure to be linear, we can write

$$m_{t|T} = \sum_{j=1}^{T} w_j(m_{t|T})y_j, \quad t = 1,\ldots,T, \tag{1}$$

where $w_j(m_{t|T})$ is the weight assigned to the jth observation in forming $m_{t|T}$. It is sometimes suggested that the weights can be obtained by running the algorithm on an artificial series consisting entirely of zeroes except for the one in the tth position; for, example, see Härdle (1990, p. 59). This is the *zero–one method*. To determine the conditions under which it is valid, we write expression (1) in matrix form as

$$m = Wy, \tag{2}$$

where $m = (m_{1|T},\ldots,m_{T|T})'$ and $y = (y_1,\ldots,y_T)'$. The zero–one method amounts to replacing the $T\times 1$ vector y by a $T\times 1$ selection vector ℓ_t, where ℓ_t is the tth column of the identity matrix I_T. However, if we postmultiply W by ℓ_t we obtain the tth column of W whereas what we want is its tth row. This will, of course, yield the required weights if W is symmetric. In other words there is a symmetry in the weighting patterns, characterised by the fact that $w_{T-t+1-i}(m_{T-t+1|T}) = w_{t+i}(m_{t|T})$, $i = -t+1,\ldots,T-t$.

When a time invariant model consists of two components driven by mutually uncorrelated disturbances, the smoothing algorithm is such that W is symmetric. If the components are correlated, W is not symmetric but it has a band structure which enables the zero–one method to produce reasonably accurate weights for points which are not near the beginning or end of a moderately large sample. The weights are actually produced in reverse order.

For models which are not time invariant, the zero–one method does not usually work. Of course, the correct weights, corresponding to all the elements of a specific row of W, could be obtained by running the procedure T times with a one in a different position on each occasion. However, to do this would mean computations of $O(T^2)$.

3. Kalman filter and fixed-interval smoothing

The linear state space model is given by

$$y_t = Z_t\alpha_t + G_t\varepsilon_t, \quad \varepsilon_t \sim \mathrm{WN}(0, I), \quad t = 1,\ldots,T,$$

$$\alpha_{t+1} = T_t\alpha_t + H_t\varepsilon_t, \quad \alpha_1 \sim \mathrm{WN}(a, P), \tag{3}$$

where y_t is the $N \times 1$ vector of observations, α_t is the $p \times 1$ state vector and ε_t is the $q \times 1$ vector of white noise (WN), that is serially uncorrelated, disturbances. The equation for y_t is called the measurement equation and the equation for α_{t+1} is the transition equation. The standardised disturbance vector ε_t appears in both equations but the disturbances in the two equations are mutually uncorrelated if $H_tG_t' = 0$. The initial state vector is to have mean a and variance matrix P. The system matrices Z_t, G_t, T_t and H_t, with appropriate dimensions, are nonstochastic in repeated realisations. The state space model (3) is time-invariant when the system matrices are constant over time t, that is $Z_t = Z$, $T_t = T$, $G_t = G$ and $H_t = H$, for $t = 1,\ldots,T$.

A predictive filtered estimator is an estimator of (a function of) the state vector at time $t+1$ based on observations up to and including time t. The Kalman filter computes the predictive filtered estimator $a_{t+1|t}$, the MMSLE of the state vector α_{t+1} conditional on the observations $Y_t = \{y_1,\ldots,y_t\}$, together with its MSE matrix, that is the covariance matrix of the estimation error, $P_{t+1|t}$. The Kalman filter is given by

$$v_t = y_t - Z_ta_{t|t-1}, \quad F_t = Z_tP_{t|t-1}Z_t' + G_tG_t', \quad M_t = T_tP_{t|t-1}Z_t' + H_tG_t', \tag{4}$$

$$a_{t+1|t} = T_ta_{t|t-1} + K_tv_t, \quad P_{t+1|t} = T_tP_{t|t-1}T_t' + H_tH_t' - K_tM_t',$$

for $t = 1,\ldots,T$, with Kalman gain matrix $K_t = M_tF_t^{-1}$ and initialisation $a_{1|0} = a$ and $P_{1|0} = P$. The one-step ahead prediction error is v_t with covariance matrix $\mathrm{Var}(v_t) = F_t$. The filtered estimator $a_{t|t}$, the MMSLE of the state vector α_t conditional on Y_t, and its MSE matrix $P_{t|t}$ can be computed by

$$a_{t|t} = a_{t|t-1} + P_{t|t-1}Z_t'F_t^{-1}v_t, \quad P_{t|t} = P_{t|t-1} - P_{t|t-1}Z_t'F_t^{-1}Z_tP_{t|t-1}, \tag{5}$$

with $t = 1,\ldots,T$. For a time-invariant state space model, the Kalman recursion for P_{t+1} may converge to a constant matrix $\bar{P}$ for a sufficiently large t. This so-called steady state is the solution to

$$\bar{P} = T\bar{P}T' + HH' - \bar{K}\bar{M}',$$

where $\bar{M} = T\bar{P}Z' + HG'$, $\bar{F} = Z\bar{P}Z' + GG'$ and $\bar{K} = \bar{M}\bar{F}^{-1}$.

Smoothed estimators of the state vector, or estimators of α_t based on all observations Y_T, are computed by first running the Kalman filter and, subsequently, running the backwards recursion

$$r_{t-1} = Z_t'F_t^{-1}v_t + L_t'r_t, \quad N_{t-1} = Z_t'F_t^{-1}Z_t + L_t'N_tL_t, \quad t = T,\ldots,1, \tag{6}$$

where $L_t = T_t - K_tZ_t$ and with initialisation $r_T = 0$ and $N_T = 0$. The Kalman filter output of v_t, F_t^{-1} and K_t must be stored for $t = 1,\ldots,T$. The smoothed state vector $a_{t|T}$ with

MSE matrix $P_{t|T}$ is then computed by

$$a_{t|T} = a_{t|t-1} + P_{t|t-1}r_{t-1}, \quad P_{t|T} = P_{t|t-1} - P_{t|t-1}N_{t-1}P_{t|t-1}, \quad t = T,\ldots,1. \tag{7}$$

Additional memory is required to store $a_{t|t-1}$ and $P_{t|t-1}$ for $t = 1,\ldots,T$.

4. Computation of the observation weights

In this section, we present general methods for computing the weights implicitly assigned to observations when estimating (linear combinations of) elements of the state vector. The elements of the state typically correspond to trends, seasonals, cycles and other components of interest. An algorithm for computing the weights used to form the filtered estimator $a_{t|t-1}$ is presented together with a minor modification for computing the weights of the contemporaneous filtered estimator $a_{t|t}$. Further, we present a procedure for computing vectors of weights to form the smoothed state vector $a_{t|T}$. The algorithm for computing weights for smoothed estimators, that is linear combinations of a_{tT}, has the same objective as the zero–one method of Section 2.

The methods of computing weights are related to leverage algorithms noted in Kohn and Ansley (1989) and de Jong (1989). However, the motivation of their papers is different and there is no mention of the fact that the algorithm can be used as a basis for extracting weights. The derivations of the weighting algorithms are presented in Appendix A. The algorithms are easy to implement and have been included in the latest update of *SsfPack* (see Koopman et al. (1999); examples of computer code are given in Appendix B to illustrate how the weights can be obtained in practice.

4.1. Computing weights for filtering

The filtered estimator of the state vector, that is the estimator of α_t based on information available at time $t-1$, can be written as

$$a_{t|t-1} = \sum_{j=1}^{t-1} w_j(a_{t|t-1})y_j. \tag{8}$$

After the Kalman filter has been applied up to time $t-1$ and the matrices K_j stored for $j = 1,\ldots,t-1$, the weight vectors can be computed by the backward recursion

$$w_j(a_{t|t-1}) = B_{t,j}K_j, \quad B_{t,j-1} = B_{t,j}T_j - w_j(a_{t|t-1})Z_j, \quad j = t-1, t-2,\ldots,1, \tag{9}$$

with $B_{t,t-1} = I$. When observations are univariate, $w_j(a_{t|t-1})$ is a $p \times 1$ vector.

The weights for $A_t a_{t|t-1}$, where A_t is a known matrix, are given by $A_t w_j(a_{t|t-1})$ but can be computed directly by the backwards recursion

$$w_j(A_t a_{t|t-1}) = b_{t,j}K_j, \quad b_{t,j-1} = b_{t,j}T_j - w_j(A_t a_{t|t-1})Z_j,$$
$$j = t-1, t-2,\ldots,1, \tag{10}$$

with $b_{t,t-1} = A_t$. Note that $b_{t,j} = A_t B_{t,j}$.

For the special case $A_t = Z_t$, we obtain the weights for $\hat{y}_{t|t-1} = Z_t a_{t|t-1}$, that is

$$\hat{y}_{t|t-1} = \sum_{j=1}^{t-1} w_j(\hat{y}_{t|t-1}) y_j. \tag{11}$$

Since $y_t = \hat{y}_{t|t-1} + v_t$, the weights from (11) give the (vector) autoregressive (VAR) representation for a time-invariant model when the filter is in a steady state. In more familiar notation,

$$y_t = \sum_{k=1}^{\infty} \Phi_k y_{t-k} + v_t, \quad \mathrm{Var}(v_t) = \bar{F}, \tag{12}$$

where $\Phi_k = w_{t-k}(\hat{y}_{t|t-1})$. The lag length will usually be infinite for a UC model, but the weights will tend to die away as k increases.

It is shown in Appendix A that the weight matrix for $a_{t|t}$ is given by $w_t(a_{t|t}) = P_{t|t-1}Z_t'F_t^{-1}$ while the weights $w_j(a_{t|t})$ for $j=t-1, t-2, \ldots, 1$ are the same as $w_j(a_{t|t-1})$ but premultiplied by the matrix $I - P_{t|t-1}Z_t'F_t^{-1}Z_t$. These weights can be computed using the backwards recursion

$$w_j(a_{t|t}) = B_{t,j}K_j, \quad B_{t,j-1} = B_{t,j}T_j - w_j(a_{t|t})Z_j, \quad j = t-1, t-2, \ldots, 1,$$

with the initialisation $B_{t,t-1} = I - P_{t|t-1}Z_t'F_t^{-1}Z_t$. This recursion is, apart from the initialisation, the same as (9).

4.2. Computing weights for smoothing

The smoothed state vector, $a_{t|T}$, can be represented as a weighted sum of all observations, that is

$$a_{t|T} = \sum_{j=1}^{T} w_j(a_{t|T}) y_j. \tag{13}$$

The weight matrices $w_j(a_{t|T})$ are computed after the Kalman filter has been applied for $t = 1, \ldots, T$ and the smoothing recursion (6) has been applied backwards up to time t. The Kalman filter stores K_j, for $j = 1, \ldots, T$, and the smoother stores the $p \times N$ matrix

$$C_j = Z_j'D_j - T_j'N_jK_j, \tag{14}$$

where $D_j = F_j^{-1} + K_j'N_jK_j$, for $j = T, T-1, \ldots, t$. The weight matrices $w_j(a_{t|T})$ are then calculated in the two opposite directions: (i) backwards in time ($j = t-1, \ldots, 1$); (ii) forwards in time ($j = t, t+1, \ldots, T$).

The weights $w_j(a_{t|T})$ for $j < t$ are given by $w_j(a_{t|T}) = (I - P_{t|t-1}N_{t-1})w_j(a_{t|t-1})$ and they can be computed by the backwards recursion (9) with initialisation $B_{t,t-1} = I - P_{t|t-1}N_{t-1}$. The weights $w_j(a_{t|T})$ for $j \geqslant t$ are computed by the forwards recursion

$$w_j(a_{t|T}) = B_{t,j}^* C_j, \quad B_{t,j+1}^* = B_{t,j}^* L_j', \quad j = t, t+1, \ldots, T, \tag{15}$$

with $B_{t,t}^* = P_{t|t-1}$.

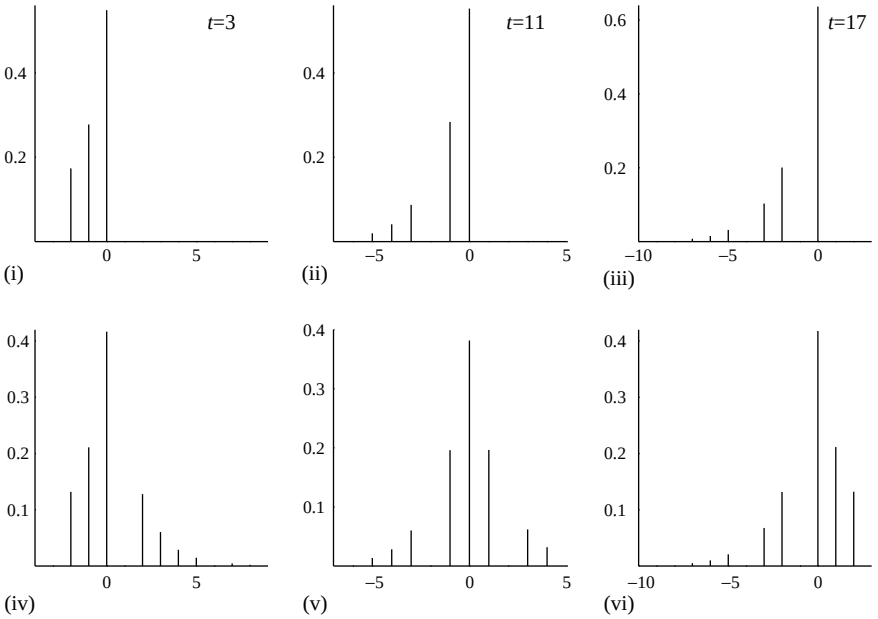

Fig. 1. Weight patterns for local level model with missing observations $t = 4, 9, 13, 16$. (i)–(iii) filtered weights associated with μ_3, μ_{11} and μ_{17}; (iv)–(vi) smoothing weights associated with μ_3, μ_{11} and μ_{17}.

5. Illustrations

The examples in this section illustrate some of the reasons for looking at weights and demonstrate the viability of the state space weighting algorithms.

5.1. Small samples and missing observations

Consider the local level model

$$y_t = \mu_t + \varepsilon_t, \quad \mu_{t+1} = \mu_t + \eta_t, \quad t = 1, \ldots, T, \tag{16}$$

where the disturbances ε_t and η_t are mutually uncorrelated white noise processes with variances σ_ε^2 and σ_η^2, respectively. To illustrate that weight patterns vary at different time points in the time series, we present the weights in Fig. 1 for filtered and smoothed estimates of μ_t at time points $t = 3$, $t = 11$ and $t = 17$ for model (16) with $\sigma_\varepsilon^2 = 1$ and $\sigma_\eta^2 = 0.6$ and for $T = 19$. The observations at $t = 4, 9, 13, 16$ are treated as missing.

The weights sum to one just as when there are no observations missing. It can be shown that the zero–one method also works if the value obtained for the weight associated with a missing observation is replaced by a zero. Harvey and Koopman (2000) present graphs of some of the asymmetric weighting patterns obtained for smoothed estimators when ε_t and η_t are correlated with each other in model (16). In such cases the

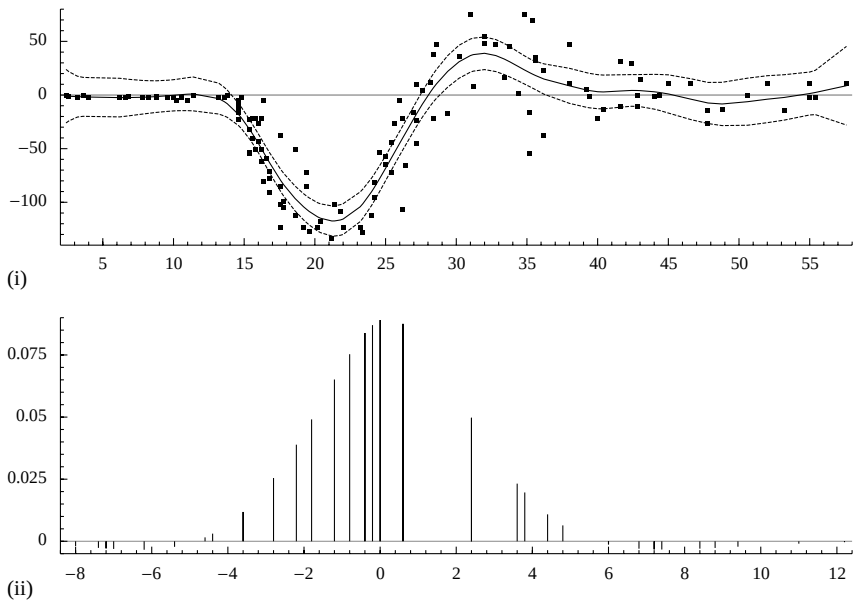

Fig. 2. Motorcycle acceleration data modelled by a cubic spline. (i) observations against time with spline and 95% confidence intervals and (ii) the weights for the spline at time $\tau_{105} = 35.6$ against time distance $\tau_{105} - \tau_j$.

zero–one method fails to give accurate weights except near the middle of a reasonably large sample.

5.2. Irregularly spaced observations

Fig. 2(i) shows 133 observations of acceleration against time (measured in milliseconds) for a simulated motorcycle accident; see Silverman (1985). The observations are a cross section rather than a time series; they are not equally spaced on the time axis and at certain time points multiple observations are recorded. These data are often used as an illustration for nonparametric regression using cubic splines. The connection between cubic splines and the state space framework has been known for many years; see Wecker and Ansley (1983). Given the data set $y_1, \ldots, y_T$ for which the tth observation is recorded at time τ_t, the time series model for a cubic spline is

$$\begin{aligned} y_t &= \mu_t + \varepsilon_t, \quad \varepsilon_t \sim \mathrm{WN}(0, \sigma_\varepsilon^2), \\ \mu_{t+1} &= \mu_t + \beta_t + \eta_t, \quad \eta_t \sim \mathrm{WN}(0, \sigma_\eta^2), \\ \beta_{t+1} &= \beta_t + \zeta_t, \quad \zeta_t \sim \mathrm{WN}(0, \sigma_\zeta^2) \end{aligned} \tag{17}$$

with

$$\varepsilon_t \sim \mathrm{WN}(0, \sigma_\varepsilon^2), \quad \eta_t \sim \mathrm{WN}(0, \sigma_\zeta^2 \delta_t^3/3), \quad \zeta_t \sim \mathrm{WN}(0, \sigma_\zeta^2 \delta_t), \quad \mathrm{E}(\eta_t \zeta_t) = \sigma_\zeta^2 \delta_t^2/2,$$

ε_t is uncorrelated with both η_t and ζ_t and $\delta_t = \tau_{t+1} - \tau_t$. The weighting pattern for the estimated signal μ_t can be interpreted as the equivalent kernel function and can be compared with kernel functions used in nonparametric regression.

The smoothing parameter, $q_\zeta = \sigma_\zeta^2/\sigma_\varepsilon^2$, can be estimated by maximum likelihood using the Kalman filter and is given by $\hat{q}_\zeta = 0.0275$. The state space smoothing algorithm is used to compute the estimates of μ_{τ_t} but it also computes the root mean square error (RMSE) of these estimates. The cubic spline (solid line) together with error bounds (dotted lines), which are based on the RMSEs, are given in Fig. 2(i).

Fig. 2(ii) shows the weights assigned to observations at adjacent time points for computing $\tilde{\mu}_{\tau_t}$, with $\tau_t = 35.6$ corresponding to the 105th observation. The gaps are caused by time periods for which no observations are recorded. The weighting pattern in Fig. 2(ii) is *not* symmetric. This is in contrast to the nonparametric approach where the weighting pattern is symmetric in that observations which are at the same 'distance' from the time point τ_t receive the same weight; see, for example, Green and Silverman (1994). The reason the patterns of optimal weights obtained from the model is different is that the number of data points observed around a particular observation is taken into account; if this number is large, the observation receives less weight.

5.3. Multivariate models and the VECM

The multivariate local level model can be written in state space form

$$\begin{aligned} y_t &= \mu_t + \varepsilon_t, \quad \varepsilon_t \sim \mathrm{WN}(0, \Sigma_\varepsilon), \quad t = 1, \ldots, T, \\ \mu_{t+1} &= \mu_t + \eta_t, \quad \eta_t \sim \mathrm{WN}(0, \Sigma_\eta), \end{aligned} \tag{18}$$

where WN now denotes multivariate white noise. If the rank of the covariance matrix Σ_η is $K < N$, an appropriate ordering of the series enables the model to be written in the common trends (or levels) form

$$\begin{aligned} y_{1t} &= \mu_t^\dagger + \varepsilon_{1t}, \\ y_{2t} &= \Pi \mu_t^\dagger + \bar{\mu} + \varepsilon_{2t}, \end{aligned} \tag{19}$$

where y_{1t} is a $K \times 1$ vector, y_{2t} is an $r \times 1$ vector with $r = N - K$, Π is an $r \times K$ matrix of coefficients and the $K \times 1$ vector $\mu_t^\dagger$ follows a multivariate random walk

$$\mu_t^\dagger = \mu_{t-1}^\dagger + \eta_t^\dagger, \quad \eta_t^\dagger \sim \mathrm{WN}(0, \Sigma_\eta^\dagger), \tag{20}$$

where $\eta_t^\dagger$ is a $K \times 1$ vector and $\Sigma_\eta^\dagger$ is a $K \times K$ positive definite matrix. The $r \times 1$ vector $\bar{\mu}$ is deterministic. In terms of (18), $\Sigma_\eta = \Pi^\dagger \Sigma_\eta^\dagger \Pi^{\dagger\prime}$, where $\Pi^\dagger = (I, \Pi')'$.

In the bivariate model, we can write

$$\Sigma_\eta = \begin{pmatrix} \sigma_{1\eta}^2 & \rho_\eta \sigma_{1\eta} \sigma_{2\eta} \\ \rho_\eta \sigma_{1\eta} \sigma_{2\eta} & \sigma_{2\eta}^2 \end{pmatrix}, \tag{21}$$

where ρ_η is the correlation. If $\rho_\eta = \pm 1$, there is a single common trend and so

$$\begin{aligned} y_{1t} &= \mu_{1t} + \varepsilon_{1t} = \mu_t^\dagger + \varepsilon_{1t}, \quad t = 1, \ldots, T, \\ y_{2t} &= \mu_{2t} + \varepsilon_{2t} = \pi \mu_t^\dagger + \bar{\mu} + \varepsilon_{2t}, \end{aligned} \tag{22}$$

where $\pi = \text{sgn}(\rho_\eta)\sigma_{2\eta}/\sigma_{1\eta}$. Some insight into weighting patterns can be achieved by transforming this model to

$$(y_{1t} + y_{2t}/\pi)/2 = \mu_t^\dagger + \bar{\mu}/2\pi + (\varepsilon_{1t} + \varepsilon_{2t}/\pi)/2,$$

$$-\pi y_{1t} + y_{2t} = \bar{\mu} + (-\pi\varepsilon_{1t} + \varepsilon_{2t}).$$

The Jacobian of the transformation is unity. The second equation does not contain the common trend while in the first equation $\bar{\mu}$ can be absorbed into the initial conditions for the trend. If we make the assumption that $\text{Var}(\varepsilon_{2t})/\text{Var}(\varepsilon_{1t}) = \pi^2 = \sigma_{2\eta}^2/\sigma_{1\eta}^2$ the irregular disturbances in the two transformed equations are uncorrelated with each other whatever the correlation, ρ_ε, between the original irregulars and so the weights for estimating the stochastic trend in the first equation will be as for a univariate local level with signal–noise ratio, q, equal to $(1 + \rho_\varepsilon)/2$ times the signal–noise ratio in the first untransformed equation. If $\bar{\mu}$ is known, this is the pattern which applies to $\mu_t^\dagger$, and hence to μ_{1t}, and so, unless $\rho_\varepsilon = 1$, the weights associated with $(y_{1t} + y_{2t}/\pi)/2$ will show a more rapid exponential decline than the weights for y_{1t} in the univariate model. The weights on y_{1t} sum to one-half while those on y_{2t} sum to $1/2\pi$. If $\bar{\mu}$ is not known, its estimator is obtained from the second transformed equation as $\bar{y}_2 - \pi\bar{y}_1$. The weights for the estimator of $\bar{\mu}/2\pi$ need to be subtracted from the weighting pattern for μ_{1t} obtained when $\bar{\mu}$ is known. This term contributes $1/2T$ to each observation in the first series and $-1/2T\pi$ to each observation in the second series. Thus, for extracting μ_{1t}, the weights on the first series now sum to one while those on the second sum to zero. The converse is true for μ_{2t}.

We now consider three different specifications, (a), (b) and (c). In all cases $\Sigma_\varepsilon = I_2$ while

$$\Sigma_\eta^{(a)} = \begin{pmatrix} 0.2 & 0 \\ 0 & 0.2 \end{pmatrix}, \quad \Sigma_\eta^{(b)} = \begin{pmatrix} 0.2 & -0.2 \\ -0.2 & 0.3 \end{pmatrix}, \quad \Sigma_\eta^{(c)} = \begin{pmatrix} 0.2 & 0.2 \\ 0.2 & 0.2 \end{pmatrix}.$$

Fig. 3 presents the weighting patterns used to construct smoothed estimators of the level in the first series in terms of its own values and those of the related series. The sample size is 100. The results for $\Sigma_\eta^{(a)}$ are in panels (i) and (ii), while for $\Sigma_\eta^{(b)}$ is in panels (iii) and (iv). Because the level disturbances in (b) are negatively correlated, the weights from the second series are negative. Case (c) has a common level as matrix $\Sigma_\eta^{(c)}$ is of rank one; π is equal to one. Weights when $\bar{\mu}$ in (22) is known are shown in panels (v) and (vi). The analysis of the previous paragraph is confirmed in that the weights on the target series decline faster than in the corresponding univariate model—these weights are as in panel (i). Panels (vii) and (viii) show that when $\bar{\mu}$ is estimated, $1/2T$ is added to the weights for the first series while the same quantity is subtracted from the weights for the second series.

The presence of common trends as in case (c) implies cointegration. In the local level model (19), there exist $r = N - K$ cointegrating vectors. Let A be an $r \times N$ matrix partitioned as $A = (A_1, A_2)$. Then pre-multiplying (19) by A gives

$$Ay_t = A_1 y_{1t} + A_2 y_{2t} = (A_1 + A_2\Pi)\mu_t^\dagger + A_2\bar{\mu} + A_1\varepsilon_{1t} + A_2\varepsilon_{2t}, \quad t = 1, \ldots, T. \quad (23)$$

The r series in Ay_t are stationary, and hence A consists of cointegrating vectors, if $A_1 + A_2\Pi = 0$.

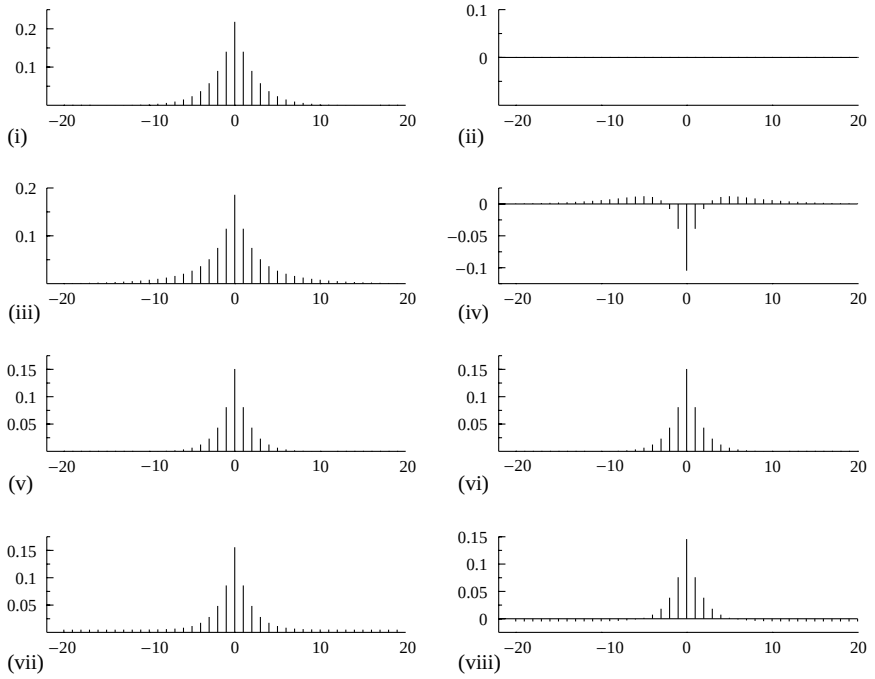

Fig. 3. Weighting patterns for estimated level of first (target) series. Multivariate local level model with $\mathrm{Var}(\varepsilon_t) = I_2$ and $T = 100$: (i) and (ii) weights for y_{1t} and y_{2t} in case (a), (iii) and (iv) weights for y_{1t} and y_{2t} in case (b), (v) and (vi) weights for y_{1t} and y_{2t} in case (c) with known constant, (vii) and (viii) weights for y_{1t} and y_{2t} in case (c) with unknown constant.

Cointegrated systems can be handled by VECM and it is interesting to compute the coefficients in the VECM representation of the common trends model. The VECM can be written as

$$\Delta y_t = \delta + \Phi^* y_{t-1} + \sum_{j=1}^{\infty} \Phi_j^* \Delta y_{t-j} + v_t, \tag{24}$$

where the relationship between the $N \times N$ parameter matrices and those in the VAR model of (12) is

$$\Phi^* = \sum_{k=1}^{\infty} \Phi_k - I, \quad \Phi_j^* = -\sum_{k=j+1}^{\infty} \Phi_k, \quad j = 1, 2, \ldots . \tag{25}$$

The rank of Φ^* is r. It is normally expressed as $\Phi^* = \Gamma A$, where A is the $r \times N$ matrix of cointegrating vectors and Γ is an $N \times r$ matrix of coefficients.

If we were to have a homogeneous model, (18) with $\Sigma_\eta = q\Sigma_\varepsilon$, then there is no cointegration, so $\Phi^* = 0$, and the model is just a VAR in first differences. The parameter matrices are $\Phi_j^* = -(-\theta)^j I$, $j = 1, 2, \ldots$ with

$\theta = \{-q - 2 + \sqrt{q^2 + 4q}\}/2$. If q is small, θ is close to -1 and there is a slow decline in the Φ_j^*'s.

Given a common trends model, (19), the predictive filtering algorithm is used to compute the coefficient matrices in the corresponding VAR. The vector $\bar{\mu}$ is set to zero. The VECM matrices, Φ^* and the Φ_j^*'s, are then obtained from (25). As is well known, the matrix A is not unique for $r > 1$ but it can be set to $(-\Pi, I)$. The Γ matrix is then computed so as to satisfy $\Phi^* = \Gamma A$, while $\delta = -\Phi^*(0, \bar{\mu}')'$. If Σ_η is dominated by Σ_ε, a slow decline in the Φ_j^*'s can be expected. This has implications for using a VECM to model a system for which a common trends model is appropriate. The *Ox* program `vecm.ox` in Appendix B can be used to compute the VAR and VECM coefficients.

Acknowledgements

During the earlier part of this research, the first author was a Research Fellow of the Royal Netherlands Academy of Arts and Sciences and its financial support is gratefully acknowledged. We would like to thank James Hamilton, Piet de Jong, Marius Ooms, Neil Shephard and two anonymous referees for helpful suggestions.

Appendix A. Derivations

A.1. Computing weights for filtering

The Kalman recursion for $a_{t+1|t}$ can be reformulated as

$$a_{t+1|t} = L_t a_{t|t-1} + K_t y_t,$$

where $L_t = T_t - K_t Z_t$. By simple substitution, it follows that

$$\begin{aligned} a_{t|t-1} &= K_{t-1} y_{t-1} + L_{t-1} K_{t-2} y_{t-2} + L_{t-1} L_{t-2} K_{t-3} y_{t-3} \\ &\quad + L_{t-1} L_{t-2} L_{t-3} K_{t-4} y_{t-4} + \cdots + L_{t-1} \ldots L_2 K_1 y_1 \\ &= \sum_{j=1}^{t-1} B_{t,j} K_j y_j, \end{aligned}$$

where

$$B_{t,t-1} = I, \quad B_{t,j} = L_{t-1} L_{t-2} \ldots L_{j+1}, \quad j = 1, \ldots, t-2. \tag{A.1}$$

Thus the weight vectors in (8) are given by $w_j(a_{t|t-1}) = B_{t,j} K_j$. They can be computed efficiently by the backwards recursion (9) since it follows from (A.1) that

$$B_{t,j-1} = B_{t,j} L_j, \quad j = t-1, t-2, \ldots, 1.$$

Minor manipulation, using the definition $L_j = T_j - K_j Z_j$, gives (9).

The estimator of the state vector at time t conditional on Y_t is given by

$$a_{t|t} = a_{t|t-1} + P_{t|t-1}Z_t'F_t^{-1}v_t$$

$$= (I - P_{t|t-1}Z_t'F_t^{-1}Z_t)a_{t|t-1} + P_{t|t-1}Z_t'F_t^{-1}y_t.$$

The weights for contemporaneous filtering are closely related to the weights for predictive filtering, $w_j(a_{t|t-1})$, since we have

$$w_t(a_{t|t}) = P_{t|t-1}Z_t'F_t^{-1},$$

$$w_j(a_{t|t}) = (I - P_{t|t-1}Z_t'F_t^{-1}Z_t)w_j(a_{t|t-1}), \quad j = t-1, t-2, \ldots, 1.$$

Therefore, recursion (9) can be used with $x_t = a_{t|t}$ and $B_{t,t-1} = I - P_{t|t-1}Z_t'F_t^{-1}Z_t$.

The weights for the filtered estimator $\hat{y}_{t|t}$ are given by $w_j(\hat{y}_{t|t}) = Z_t w_j(a_{t|t})$, for $j = t, t-1, \ldots, 1$. Since

$$Z_t P_{t|t-1}Z_t'F_t^{-1} = I - G_t G_t'F_t^{-1}, \quad Z_t(I - P_{t|t-1}Z_t'F_t^{-1}Z_t) = G_t G_t'F_t^{-1}Z_t.$$

A.2. Computing weights for smoothing

Here we develop (1) the weights for r_{t-1} and (2) the weights for $a_{t|T}$.

Step 1: Define

$$u_t = Z_t F_t^{-1}v_t, \quad U_t = Z_t F_t^{-1}Z_t'.$$

It follows that $U_t = \mathrm{Var}(u_t)$ and, since $v_t = y_t - Z_t a_{t|t-1}$,

$$w_t(u_t) = Z_t F_t^{-1} - U_t w_t(a_{t|t-1}), \quad w_j(u_t) = -U_t w_j(a_{t|t-1}), \tag{A.2}$$

for $j = t-1, \ldots, 1$. We can rewrite the smoothing recursions (6) as

$$r_{t-1} = u_t + L_t'r_t$$

$$= u_t + L_t'u_{t+1} + L_t'L_{t+1}'u_{t+2} + \cdots + L_t'\ldots L_{n-1}'u_n$$

$$= u_t + B_{t+1,t-1}'u_{t+1} + B_{t+2,t-1}'u_{t+2} + \cdots + B_{n,t-1}'u_n, \tag{A.3}$$

$$N_{t-1} = U_t + L_t'N_t L_t$$

$$= U_t + L_t'U_{t+1}L_t + L_t'L_{t+1}'U_{t+2}L_{t+1}L_{t+2} + \cdots + L_t'\ldots L_{T-1}'U_T L_{T-1}\ldots L_t$$

$$= U_t + B_{t+1,t-1}'U_{t+1}B_{t+1,t-1} + B_{t+2,t-1}'U_{t+2}B_{t+2,t-1}$$

$$+ \cdots + B_{T,t-1}'U_T B_{T,t-1}, \tag{A.4}$$

where matrix $B_{t,j}$ is defined in (A.1). Note some properties of matrix $B_{j,t-1}$:

$$B_{j,t-1} = L_{j-1}B_{j-1,t-1}, \quad j = t, \ldots, n,$$

$$B_{j,t-1} = B_{j,k}B_{k+1,t-1}, \quad j = t, \ldots, n, \ \ k = t-1, \ldots, j-1,$$

$$B_{j,t-1} = 0, \quad j = 1, \ldots, t-1. \tag{A.5}$$

The weights for r_{t-1} are obtained by

$$w_j(r_{t-1}) = w_j(u_t) + B'_{t+1,t-1}w_j(u_{t+1}) + B'_{t+2,t-1}w_j(u_{t+2}) + \cdots + B'_{T,t-1}w_j(u_T),$$

for $j = 1,\ldots,n$. By substitution of (A.2), we obtain

$$\begin{aligned} w_j(r_{t-1}) &= B'_{j,t-1}Z_jF_j^{-1} \\ &\quad - [U_tw_j(a_{t|t-1}) + B'_{t+1,t-1}U_{t+1}w_j(a_{t+1|t}) + \cdots + B'_{T,t-1}U_Tw_j(a_{T|T-1})]. \end{aligned}$$

From (9) we have $w_j(a_{t|t-1}) = B_{t,j}K_j$ and substitution leads to

$$w_j(r_{t-1}) = B'_{j,t-1}Z_jF_j^{-1} - [U_tB_{t,j} + B'_{t+1,t-1}U_{t+1}B_{t+1,j} + \cdots + B'_{T,t-1}U_TB_{T,j}]K_j.$$

Convenient expressions for $w_j(r_{t-1})$ can be obtained by using (29) for N_{t-1} and using the properties of $B_{j,t-1}$ in (A.5), we have

$$w_j(r_{t-1}) = \begin{cases} -N_{t-1}B_{t,j}K_j, & j = t-1, t-2, \ldots, 1, \\ Z_tF_t^{-1} - L'_tN_tK_t, & j = t, \\ B'_{j,t-1}Z_jF_j^{-1} - B'_{j+1,t-1}N_jK_j, & j = t+1, t+2, \ldots, T. \end{cases}$$

Using definition $D_j = F_j^{-1} + K'_jN_jK_j$ we further define

$$C_j = Z'_jF_j^{-1} - L'_jN_jK_j = Z'_jD_j - T'_jN_jK_j, \quad j = t, t+1, \ldots, T.$$

From the fact that $B_{j+1,t-1} = L_jB_{j,t-1}$, we obtain finally

$$w_j(r_{t-1}) = \begin{cases} -N_{t-1}B_{t,j}K_j, & j = t-1, t-2, \ldots, 1, \\ C_t, & j = t, \\ B'_{j,t-1}C_j, & j = t+1, t+2, \ldots, T. \end{cases}$$

Step 2: For the state smoothing recursions (6) and (7), it can be shown that $a_{t|T}$ is the weighted average

$$a_{t|T} = \sum_{j=1}^{T} w_j(a_{t|T})y_j.$$

The weights for $a_{t|T}$ depend on the position of j with relation to t. For example, for $j < t$ we have

$$\begin{aligned} w_j(a_{t|T}) &= w_j(a_{t|t-1}) + P_{t|t-1}w_j(r_{t-1}) \\ &= B_{t,j}K_j - P_{t|t-1}N_{t-1}B_{t,j}K_j \\ &= (I - P_{t|t-1}N_{t-1})B_{t,j}K_j, \quad j = t-1, t-2, \ldots, 1. \end{aligned}$$

It follows that the weights for the smoothed state vector $a_{t|T}$ are given by

$$w_j(a_{t|T}) = \begin{cases} (I - P_{t|t-1}N_{t-1})w_j(a_{t|t-1}), & j = t-1, t-2, \ldots, 1, \\ P_{t|t-1}C_t, & j = t, \\ P_{t|t-1}B'_{j,t-1}C_j, & j = t+1, t+2, \ldots, T. \end{cases}$$

Note that $w_j(a_{t|t-1}) = 0$ for $j \geqslant t$.

Evaluation of $B'_{j,t-1}$, for $j = t+1, t+2, \ldots, T$, can be done using the forwards recursion (15) since it follows that

$$B'_{j+1,t-1} = B'_{j,t-1}L'_j, \quad j = t+1, \ldots, T,$$

and $B'_{t+1,t-1} = L'_t$. The matrix $B^*_{t,j}$ in (15) for $x_t = a_{t|T}$ is defined as $B^*_{t,j} = P_{t|t-1}B'_{j,t-1}$, for $j = t+1, \ldots, T$.

The weights for the smoothed estimator of the signal are given by $w_j(Z_t a_{t|T}) = Z_t w_j(a_{t|T})$, for $j = 1, \ldots, T$. This leads to the following simplification:

$$Z_t a_{t|T} = \sum_{j=1}^{T} w_j(\hat{y}_{t|T})y_j,$$

where

$$w_j(Z_t a_{t|T}) = \begin{cases} G_t G'_t C'_t w_j(a_{t|t-1}), & j = t-1, t-2, \ldots, 1, \\ I - G_t G'_t D_t, & j = t, \\ G_t G'_t K'_t B'_{j,t} C_j, & j = t+1, t+2, \ldots, T. \end{cases}$$

Note that $Z_t(I - P_{t|t-1}N_{t-1}) = G_t G'_t C'_t$ and $Z_t P_{t|t-1}L'_t = G_t G'_t K_t$. Note also that $Z_t a_{t|T}$ is the estimator of y_t when it is missing.

Appendix B. Example programs

Listings of code written for the *Ox* programming language of Doornik (1998) are given below. The programs make use of the *SsfPack* package which can be downloaded from the Internet at www.ssfpack.com; for further details, see Koopman et al. (1999). The code can be downloaded from http://staff.feweb.vu.nl/koopman/weights/.

B.1. Standard code for computing weights

```
#include < oxstd.h>
#include < packages/ssfpack/ssfpack.h>
main()
{
    decl ct, loc, my, mphi, momega, q, r, mw, mw_zo;
```

```
    // data (zeroes with unity on position where weights are required)
    ct=11; loc=5;
    my=zeros(1,ct); my[0][loc]=1;

    // local level model
    mphi=< 1;1>; momega=unit(2);
    q=0.01; r=0.0;
    momega[0][0]=q; momega[1][0]=momega[0][1]=sqrt(q) * r;

    // exact method
    mw=SsfWeights(ST_SMO, my, mphi, momega);
    // zero one method
    SsfMomentEst(ST_SMO, & mw_zo, my, mphi, momega);

    // output
    decl index=range(-5, 5);
    println(index' ~ mw'~ reverser(mw_zo[:1][])');
    println(sumc(index' ~ mw'~ reverser(mw_zo[:1][])'));
}
```

Comments: (i) by replacing `SMO` with `FIL` the program computes the weights for filtering rather than for smoothing (note that in *Ox* indices for arrays start counting at 0, not 1); (ii) the functions `SsfMomentEst` and `SsfWeights` are from *SsfPack*.

B.2. Code for vector error-correction model weights

```
#include < oxstd.h>
#include < oxdraw.h>
#include < packages/ssfpack/ssfpack.h>

decl s_idraw=0, s_ct=400, s_loc=200;
decl s_mphi, s_momega, s_msigma, s_op=ST_PRED;

DrawVECM(const i, const j)
{
    decl my=zeros(2,s_ct); my[j][s_loc-1]=1;
    decl mw=SsfWeights(s_op, my, s_mphi, s_momega, s_msigma);
    mw=cumulate(mw')';

    decl d=mw[i][s_loc-1]; if (i==j) d-=1.0;
    println(" phi* ", i+1, " ", j+1, "=", d);

    // graph for mw[i][]
}
```

```
main()
{
    // bivariate LL (common) model
    s_mphi=< 1,0;0,1;1,0;0,1>;
    s_momega=unit(4);
    s_momega[:1][:1]=< .4,.2;.2,.1>;
    s_msigma=10^ 9 * s_momega[:1][:1]| 0;

    DrawVECM(0,0); DrawVECM(0,1); DrawVECM(1,0); DrawVECM(1,1);
    // show graphs
}
```

Comment: the *Ox* function `cumulate()` is used to compute the VECM matrices defined in (25).

References

Anderson, B.D.O., Moore, J.B., 1979. Optimal Filtering. Prentice-Hall, Englewood Cliffs, NJ.

Burman, J.P., 1980. Seasonal adjustment by signal extraction. Journal of the Royal Statistical Society, Series A 143, 321–337.

Cleveland, W.P., Tiao, G.C., 1976. Decomposition of seasonal time series: a model for the census X-11 program. Journal of the American Statistical Association 71, 581–587.

de Jong, P., 1989. Smoothing and interpolation with the state space model. Journal of the American Statistical Association 84, 1085–1088.

Doornik, J.A., 1998. Object-Oriented Matrix Programming using Ox 2.0. Timberlake Consultants Press, London.

Durbin, J., Koopman, S.J., 2001. Time Series Analysis by State Space Methods. Oxford University Press, Oxford.

Green, P.G., Silverman, B.W., 1994. Nonparametric Regression and Generalized Linear Models. Chapman & Hall, London.

Härdle, W., 1990. Applied Nonparameteric Regression. Cambridge University Press, Cambridge.

Harvey, A.C., 1989. Forecasting, Structural Time Series Models and the Kalman Filter. Cambridge University Press, Cambridge.

Harvey, A.C., Koopman, S.J., 2000. Signal extraction and the formulation of unobserved components models. Econometrics Journal 3, 84–107.

Kitagawa, G., Gersch, W., 1996. Smoothness Priors Analysis of Time Series. Springer, Berlin.

Kohn, R., Ansley, C.F., 1989. A fast algorithm for signal extraction, influence and cross-validation. Biometrika 76, 65–79.

Koopman, S.J., Shephard, N., Doornik, J.A., 1999. Statistical algorithms for models in state space using SsfPack 2.2. Econometrics Journal 2, 113–166.

Shumway, R.H., Stoffer, D.S., 2000. Time Series Analysis and its Application. Springer, New York.

Silverman, B.W., 1985. Some aspects of the spline smoothing approach to non-parametric regression curve fitting. Journal of the Royal Statistical Society, Series B 47, 1–52.

Wecker, W.E., Ansley, C.F., 1983. The signal extraction approach to nonlinear regression and spline smoothing. Journal of the American Statistical Association 78, 81–89.

Whittle, P., 1983. Prediction and Regulation, 2nd Edition. Blackwell, Oxford.