



**DEPARTMENT OF ECONOMETRICS
AND BUSINESS STATISTICS**

**Reconstructing the Kalman Filter for Stationary and
Non Stationary Time series**

Ralph D. Snyder and Catherine S. Forbes

Reconstructing the Kalman Filter for Stationary and Non Stationary Time Series

***JOINT WORKING PAPER WITH THE DEPARTMENT OF ECONOMETRICS
& BUSINESS STATISTICS AND THE BUSINESS AND ECONOMIC
FORECASTING UNIT.***

October 2002

Ralph D. Snyder

Department of Econometrics and Business Statistics

Monash University

Clayton, Victoria 3185

Australia

(613) 9905 2366

ralph.snyder@buseco.monash.edu.au

Catherine S. Forbes

Department of Econometrics and Business Statistics

Monash University

Clayton, Victoria 3185

Australia

(613) 9905 2471

catherine.forbes@buseco.monash.edu.au

Reconstructing the Kalman Filter

Abstract

A Kalman filter, suitable for application to a stationary or a non-stationary time series, is proposed. It works on time series with missing values. It can be used on seasonal time series where the associated state space model may not satisfy the traditional observability condition.

A new concept called an ‘extended normal random vector’ is introduced and used throughout the paper to simplify the specification of the Kalman filter. It is an aggregate of means, variances, covariances and other information needed to define the state of a system at a given point in time. By working with this aggregate, the algorithm is specified without direct recourse to those relatively complex formulae for calculating associated means and variances, normally found in traditional expositions of the Kalman filter.

A computer implementation of the algorithm is also described where the extended normal random vector is treated as an object; the operations of addition, subtraction and multiplication are overloaded to work on instances of this object; and a form of statistical conditioning is implemented as an operator.

Keywords

Time series analysis, forecasting, Kalman filter, state space models, object-oriented programming.

JEL: C13, C22, C44

1. Introduction

Being the cornerstone of modern programming practice, objects and classes provide a useful basis for structuring computer code. They promote data aggregation based on concepts inherent in applications. They lead to simplified code that is usually easier to understand and to maintain.

Being a systematic way of thinking about problems and the contexts in which they occur, it is our contention that the object-oriented approach transcends computer programming in its usefulness. The purpose of this paper is to demonstrate that the approach can promote useful ways of thinking in statistical theory. We illustrate this point by treating normal random vectors as objects and applying them to the problem of filtering time series.

More specifically, data-aggregation possibilities associated with an object-oriented approach are exploited, to simplify the specification of the Kalman filter (Kalman, 1960; Kalman and Bucy, 1961). The mean and variance of a random vector are combined into a single structure. Operations of addition, multiplication and conditioning are defined on instances of this structure. Then the Kalman filter, for stationary time series, is defined in terms of the resulting object and its operations.

This normal random vector object is then extended to include a special matrix required in the case of non-stationary time series to carry additional information forward through time. This extended normal random vector object is used to specify a new augmented Kalman filter for non-stationary time series, one that has a more concise specification than existing approaches and one that avoids certain limitations, outlined later in the paper, of its closest antecedent: the approach of Harvey (1991).

The Kalman filter (Kalman, 1960, Kalman and Bucy, 1961) is essentially an algorithm for revising the moments of stochastic components of a linear time series model to reflect information about them contained in time series data. It is often used as a stepping-stone to deriving the moments of components at future points of time for forecasting purposes. It is useable in real time applications, because revisions to the moments are made recursively as each successive member of a time series is observed. It can also be used as part of the process required to find maximum likelihood estimates of unknown model parameters (Schweppe, 1965; Harvey, 1991). It is therefore generally perceived as having a central role in modern time series analysis.

The equations in the Kalman filter for calculating the required means and variances were originally derived using projection theory in linear spaces. Duncan and Horn (1972), who sought to simplify matters for those unfamiliar with this theory, showed that the equations could be derived using a stochastic coefficients regression framework. Meinhold and Singpurwalla (1983), with a similar quest, employed the theory of conditional probability. Both approaches simplified the *derivation* of the Kalman filter equations. They did not, however, simplify the *form* of these equations.

The plan of this paper is as follows. A condensed version of the linear state space model is introduced in Section 2. An outline of a general filter is presented in Section 3. The new object and associated operations are defined in Section 4. The reconstructed Kalman filter for stationary time series is presented in Section 5. An extension to the object to handle division by singular matrices is detailed in Section 6. The augmented Kalman filter for non-stationary time series is considered in Section 7. Finally, the role of the algebraic system in facilitating a logical structure for an object-oriented computer implementation of the Kalman filter is

outlined in Section 8. Throughout the paper, all random vectors are governed by a multivariate normal distribution. The term ‘normal random vector’ is abbreviated to NRV.

2. Linear Statistical Model: Dynamic Form

The key problem of time series analysis is to account for inter-temporal dependencies. A common practice is to rely on a mix of observable and unobservable quantities that are understood to summarise the state of a process under consideration at a given point of time and to specify how they change over time. Traditionally, the distinction between observable and unobservable quantities is preserved with the use of two symbols. In our quest to minimize notation we combine them into a single random vector, which at typical time t , is designated by $\underline{\tilde{x}}(t)$. The problem of distinguishing between both types of quantities will be considered later in the exposition. Note that, throughout the paper, a tilde is used to indicate that the associated quantity is random.

The inter-temporal dependencies are captured by the first-order recurrence relationship

$$\underline{\tilde{x}}(t) = A\underline{\tilde{x}}(t-1) + \underline{\tilde{u}}(t). \quad (2.1)$$

A is a fixed matrix. The random vector $\underline{\tilde{u}}(t)$ represents the effect of concurrent factors on the process. This random vector is assumed to have a zero mean and a constant variance matrix. The $\underline{\tilde{u}}(t)$ are serially uncorrelated. To reduce notational complexity, the scope of the paper is restricted to the invariant case of the so-called state space model. Invariant state space models are the most commonly used in practice. Moreover, what follows is easily generalised with the use of further time subscripts.

The recurrence relationship (2.1) must be provided with a seed. It is therefore also assumed that

$$\underline{x}(0) = \underline{x} \quad (2.2)$$

where \underline{x} is a random vector with a known probability distribution. It will be seen that for non-stationary series, the distribution of \underline{x} may not be completely known, in which case special measures must be taken to accommodate this more challenging case. Either way, it is further assumed that the $\underline{u}(t)$ are uncorrelated with \underline{x} .

Example 2.1

The local level model (Muth, 1960) is a common special case. It underlies simple exponential smoothing, a widely used method for forecasting inventory demand (Brown, 1959). The series values $\underline{z}(t)$ are randomly scattered about a path traced by a wandering level $\underline{\ell}(t)$.

More specifically

$$\begin{aligned} \underline{z}(t) &= \underline{\ell}(t-1) + \underline{u}_1(t) \\ \underline{\ell}(t) &= \underline{\ell}(t-1) + \underline{u}_2(t) \end{aligned}$$

This model can be rewritten in conformity with (2.1) as

$$\begin{bmatrix} \underline{z}(t) \\ \underline{\ell}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \underline{z}(t-1) \\ \underline{\ell}(t-1) \end{bmatrix} + \begin{bmatrix} \underline{u}_1(t) \\ \underline{u}_2(t) \end{bmatrix} .$$

Note that the component vector consists of the observable quantity $\underline{z}(t)$ and the unobservable quantity $\underline{\ell}(t)$.

3. Filtering Time Series

The state space model may be used, in conjunction with past data, to generate forecasts. A vector $x(t)$ without the tilde represents the observed value of the typical random components vector $\tilde{x}(t)$. In the case of vector auto-regressions, all the elements of the observation vector $x(t)$ are typically known (Lütkepohl, 1991). In models with unobserved components, however, the unobserved parts of $x(t)$ may be treated as missing values. Thus, in the local level example, an observation is represented by $x(t)' = [z(t), \#]$ where $\#$ designates a 'missing value'.

The forecasting problem is to derive the distributions of future components vectors using information from a sample of size n . It may be expressed more formally by introducing the notation $\tilde{x}(t, s)$ to represent the random vector $\tilde{x}(t)$ once its distribution has been revised with all the observations from periods 1 to s . The possibilities for $\tilde{x}(t, s)$ are depicted in Figure 1 for different values of s and t . In particular, the forecasting problem can be expressed as one of deriving the distributions of $\tilde{x}(n + j, n)$ where j designates the prediction lead-time index.

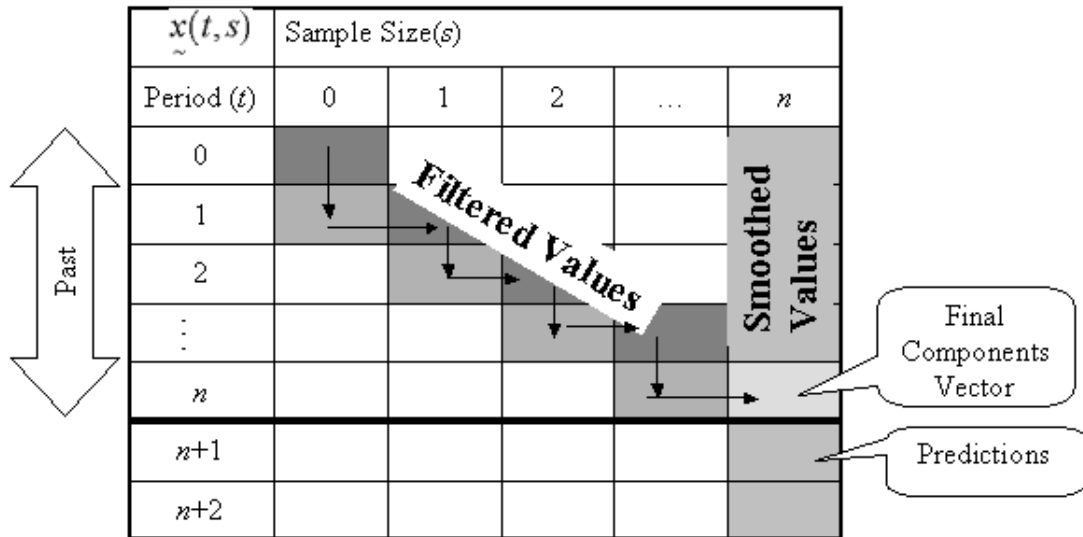


Figure 1. Difference between filtered and smoothed components vectors.

The required future prediction distributions are related to the distribution of the final components vector $\tilde{x}(n, n)$. Once the moments of the latter are known, the moments of the prediction distributions can be generated from appropriate recursive relationships implied by (2.1). Thus, the quest reduces to finding the distribution of the final components vector.

Two paths, depicted in Figure 1, may be followed in the derivation of the distribution of $\tilde{x}(n, n)$. The first, based on the approach of Duncan and Horn (1972), converts the dynamic linear model (2.1) into an equivalent random coefficients regression model. The unknown components are treated as the random coefficients. The known components are regarded as the observations of a dependent variable. The regression is fitted to the entire sample of size n to yield the moments of the random vectors depicted in the final column of Figure 1.

This regression approach is rarely used in practice. The associated data matrix is normally quite large and this is commonly perceived as a hindrance to efficient computation. Thus, a second approach may be used that effectively follows the diagonal path depicted by the

arrows in Figure 1. At the beginning of typical period t , the moments of $\underline{x}(t, t-1)$ are calculated from the moments of the previous $\underline{x}(t-1, t-1)$ using the moments equations implied by (2.1). At the end of typical period t , the random vector $\underline{x}(t, t-1)$ is revised with the observation $x(t)$ to give $\underline{x}(t, t)$. An algorithm used to implement this type of strategy is usually referred to as a filter. It may be summarized by the equations:

Initialisation

$$\underline{x}(0, 0) = \underline{x} \quad (3.1)$$

For successive periods $t = 1, 2, \dots, n$ apply the following steps:

Time Advance Step

$$\underline{x}(t, t-1) = A\underline{x}(t-1, t-1) + \underline{u}(t) \quad (3.2)$$

Revision Step

$$\underline{x}(t, t) = \underline{x}(t, t-1) | x(t). \quad (3.3)$$

Note that the equation (3.2) is obtained by conditioning (2.1) on $x(1), x(2), \dots, x(t-1)$. The equation (3.3) is obtained from a straightforward application of conditioning.

The equations (3.1)-(3.3) provide an overview of filtering. More details, however, must be elaborated, before these equations become operational. In particular, they imply relationships for calculating the moments of the associated random variables. It is one version of these relationships that form the so-called Kalman filter.

In a sense, the equations for the Kalman filter are redundant. The filtering equations (3.2) and (3.3) involve addition (+), multiplication by a fixed matrix and conditioning (|). If the impact of these operations on the moments of normally distributed random vectors is defined in general, then the equations (3.2) and (3.3) suffice to describe how the pertinent moments can be calculated without recourse to further equations specific to the filtering context. This strategy leads to an algebraic system for Gaussian random vectors.

4. An Algebraic System for Gaussian Random Vectors

In this paper, the notation $N(\mu_x, V_x)$ is used to represent an NRV \underline{x} . It enables us to highlight the dependence of the NRV on its mean μ_x and its variance V_x . Given this convention, it is legitimate to write $\underline{x} = N(\mu_x, V_x)$. Note that any constant vector x can be written as $N(x, 0)$.

This notation provides a means for succinctly defining operations on NRVs. More specifically, if \underline{x} and \underline{y} are uncorrelated NRVs, then

$$\underline{x} + \underline{y} = N(\mu_x + \mu_y, V_x + V_y) \quad (4.1)$$

$$\underline{x} - \underline{y} = N(\mu_x - \mu_y, V_x + V_y). \quad (4.2)$$

If A is a fixed matrix, then

$$A\underline{x} = N(A\mu_x, AV_x A'). \quad (4.3)$$

Furthermore, if A is a square, non-singular matrix, then

$$A \setminus \underline{x} = N\left(A^{-1}\mu_x, A^{-1}V_x A'^{-1}\right). \quad (4.4)$$

The notation, in conjunction with these definitions, admits the possibility of more compact representations of equations involving moments of NRVs. It is now possible to write

$\underline{z} = \underline{x} + \underline{y}$ instead of the two equations $\mu_z = \mu_x + \mu_y$ and $V_z = V_x + V_y$; $\underline{z} = \underline{x} - \underline{y}$ instead

of $\mu_z = \mu_x - \mu_y$ and $V_z = V_x + V_y$; $\underline{y} = A\underline{x}$ in place of the equations $\mu_y = A\mu_x$ and

$V_y = AV_x A'$. Note that the distributive law $(A + B)\underline{x} = A\underline{x} + B\underline{x}$ applies.

Conditioning is another important operation. It is established in the Appendix that $\underline{x} | x$ is a weighted average of \underline{x} and x . More specifically,

$$\underline{x} | x = (I - W)\underline{x} + Wx \quad (4.5)$$

where I is an identity matrix and W a ‘weight’ matrix. In the special case where x contains no missing values, we require that $\underline{x} | x = x$. This is achieved when $W = I$. At the other

extreme, when all the elements of x are missing values, no revision is needed. In other words $\underline{x} | x = \underline{x}$, a situation obtained when $W = 0$. In the most common case, where x is a partial

observation, the task of finding $\underline{x} | x$ is broken into a succession of smaller transformations of

the same form, one for each observed element in x . In the typical case, where the element x_p

of the k -vector x is a genuine observation, the following steps are applied. First, construct a

‘weight’ matrix W_p using the rule

$$W_p = V_x e_p e_p' / e_p' V_x e_p \quad (4.6)$$

where e_p is the unit coordinate vector with its p th element equal to 1. Second, obtain a revised value of \underline{x} using the generalised weighted average $(I - W_p)\underline{x} + W_p x$.

The definitions of the + and – operators in 4.1 and 4.2 have been restricted to uncorrelated NRVs. The framework can indirectly accommodate correlated NRVs by stacking them.

Thus, if \underline{x}_1 and \underline{x}_2 are two correlated random vectors, their sum can be found by stacking

them and then finding the product $\begin{bmatrix} I & I \end{bmatrix} \begin{bmatrix} \underline{x}_1 \\ \underline{x}_2 \end{bmatrix}$ where I is a commensurate identity matrix.

Since \underline{x}_1 and \underline{x}_2 only contain information on variances, covariance information would have to be incorporated separately into the stacking process. A stacking function like

$\underline{x} = \text{stack}(\underline{x}_1, \underline{x}_2, V_{12})$, where V_{12} is the covariance matrix, would be required in any computer implementation.

5. The Kalman Filter

Using the operations from Section 4, the equations (3.2) and (3.3) now fully define how the moments of the state vectors are revised over time. These equations are therefore a very compact representation of the Kalman filter. Compactness is achieved because:

The observed and unobserved components are combined into a single vector.

The state space model consists of only one equation instead of the more usual measurement and transition equations.

Means and variances are combined into a single entity so that the way they are calculated is described by a single equation.

It should be emphasised that all we have done so far is to repackage the Kalman filter. The reduction in notation and equations simplifies its specification. It does not, however, reduce computational loads. Depending on how it is implemented, computational loads stay about the same.

6. An Extension to the Algebraic System of NRVs

The division operation defined by (4.4) presumed the non-singularity of the matrix A . The focus of this section is on what happens when A is potentially singular. The theory of NRVs is extended to accommodate generalised inverses.

When A is singular, the Moore-Penrose generalised-inverse is represented by A^- and satisfies the condition $AA^-A = A$. The general solution of the equation $Ax = b$ is

$x = A^-b + (I - A^-A)\delta$ where δ is an arbitrary vector. When extended to the space of

NRVs, the equation $A\tilde{x} = \tilde{b}$ has a general solution $\tilde{x} = A^-\tilde{b} + (I - A^-A)\tilde{\delta}$ where $\tilde{\delta}$ is a Gaussian random vector with arbitrary variances.

To obtain a properly integrated algebraic system of NRVs that accommodates division by singular matrices, the typical NRV \tilde{x} is extended to a new object designated by \tilde{x}^* that is defined by $\tilde{x}^* = \tilde{x} + B_x\tilde{\delta}$ where B_x is a fixed matrix. It is inconvenient to carry the $\tilde{\delta}$ notation, so \tilde{x}^* is represented by $N(\mu_x, V_x, B_x)$, the $\tilde{\delta}$ now being implicit. We call \tilde{x}^* an extended normal random vector (ENRV). Note that an NRV is a special case of an ENRV where $B_x = 0$. A constant vector is a special case where, in addition, $V_x = 0$.

Suppose that $\underline{x}^* = \underline{x} + B_x \underline{\delta}$ and $\underline{y}^* = \underline{y} + B_y \underline{\delta}$ and that \underline{x} and \underline{y} are uncorrelated. Note that \underline{x}^* and \underline{y}^* cannot be uncorrelated because of their mutual dependence on $\underline{\delta}$. However, for any given value δ of $\underline{\delta}$, both \underline{x}^* and \underline{y}^* are then uncorrelated. We say that \underline{x}^* and \underline{y}^* are conditionally uncorrelated.

The random vector operators are extended as follows. First, if \underline{x}^* and \underline{y}^* are conditionally uncorrelated ENRVs, then the ‘plus’ and ‘minus’ operations are defined by

$$\underline{x}^* + \underline{y}^* = N(\mu_x + \mu_y, V_x + V_y, B_x + B_y) \quad (6.1)$$

$$\underline{x}^* - \underline{y}^* = N(\mu_x - \mu_y, V_x + V_y, B_x - B_y). \quad (6.2)$$

For a fixed matrix A and an ENRV \underline{x}^* , the definition of the ‘times’ operator is extended to:

$$A\underline{x}^* = N(A\mu_x, AV_x A', AB_x). \quad (6.3)$$

‘Division’ is defined in terms of generalised inverses as

$$A \setminus \underline{x}^* = N\left(A^- \mu_x, A^- V_x (A^-)'\right), (I - A^- A) + A^- B_x \quad (6.4)$$

The latter arises because $A \setminus \underline{x}^* = A^- \underline{x}^* + (I - A^- A) \underline{\delta}$. Substituting $\underline{x}^* = \underline{x} + B_x \underline{\delta}$ this becomes $A \setminus \underline{x}^* = A^- \underline{x} + A^- B_x \underline{\delta} + (I - A^- A) \underline{\delta}$.

Conditioning with extended NRVs involves added complexity because of the arbitrary component. This is illustrated by the following example.

Example 6.1

Suppose that $\underline{x}^* = N\left(\begin{bmatrix} 10 \\ 5 \end{bmatrix}, \begin{bmatrix} 8 & 2 \\ 2 & 30 \end{bmatrix}, \begin{bmatrix} 2 & 0 \\ 5 & 0 \end{bmatrix}\right)$ and $x = \begin{bmatrix} 20 \\ \# \end{bmatrix}$. Clearly, $\underline{x}_1^* = 10 + u_1 + 2\delta_1$

where u_1 is a Gaussian random variable with mean 0 and variance 8. Now let $x_1 = 20$ and solve for δ_1 to get $\delta_1 = 5 - 0.5u_1$. The second equation is $\underline{x}_2^* = 5 + u_2 + 5\delta_1$ where u_2 is a Gaussian random variable with mean 0 and variance 16. On the elimination of δ_1 , it becomes $\underline{x}_2^* = 30 - 2.5u_1 + u_2$. The dependence on the arbitrary vector δ_1 is then broken. Noting that u_1 and u_2 are correlated, it can be shown from these equations that

$\underline{x}^* | x = N\left(\begin{bmatrix} 20 \\ 30 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 60 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}\right)$. In effect, Gaussian elimination has been used to

reduce the first column of the B_x to a null vector, so destroying the dependence of \underline{x}^* on any arbitrary random vector.

More generally, if x is an observation of \underline{x}^* , possibly containing missing values, then

$$\underline{x}^* | x = N\left((I - W)\mu_x + Wx, (I - W)V_x(I - W)', (I - W)B_x\right) \quad (6.5)$$

where W is a ‘weight’ matrix chosen to annihilate columns of B_x associated with the arbitrary random vector and to annihilate missing values in x . The transformation (6.5) is broken down into a sequence of smaller transformations of a similar form applied for each successive observed value in x . In the typical case, where the element x_p of x is a genuine observation, the following steps are applied:

Case 1 (row p of B_x is null)

Apply the conventional conditioning update by constructing a ‘weight’ matrix W_p using the rule:

$$W_p = V_x e_p e_p' / e_p' V_x e_p . \quad (6.6)$$

Case 2 (the q th element in row p of B_x is non-zero)

Construct the weight matrix so that it can eliminate the arbitrary random variable δ_q :

$$W_p = B_x e_q e_q' / e_q' B_x e_q . \quad (6.7)$$

In both cases the revised \tilde{x}^* is obtained with $(I - W_p) \tilde{x}^* + W_p x$.

The rationale for (6.6) is given in the Appendix. The effect of the transformation (6.7) is to eliminate the q th column $B_x e_q$ of B_x because $(I - W_p) B_x e_q = 0$.

Example 6.2

Example 6.1 began with $\mu_x = \begin{bmatrix} 10 \\ 5 \end{bmatrix}$, $V_x = \begin{bmatrix} 8 & 2 \\ 2 & 20 \end{bmatrix}$ and $B_x = \begin{bmatrix} 2 & 0 \\ 5 & 0 \end{bmatrix}$. Thus, by (6.7)

$$W_1 = \begin{bmatrix} 2 & 0 \\ 5 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} / \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 5 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 2.5 & 0 \end{bmatrix} \text{ and } I - W_1 = \begin{bmatrix} 0 & 0 \\ -2.5 & 1 \end{bmatrix}. \text{ Using}$$

(6.5), the revised quantities are:

$$\mu_x^{\text{new}} = \begin{bmatrix} 0 & 0 \\ -2.5 & 1 \end{bmatrix} \begin{bmatrix} 10 \\ 5 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 2.5 & 0 \end{bmatrix} \begin{bmatrix} 20 \\ \# \end{bmatrix} = \begin{bmatrix} 20 \\ 30 \end{bmatrix}$$

$$V_x^{\text{new}} = \begin{bmatrix} 0 & 0 \\ -2.5 & 1 \end{bmatrix} \begin{bmatrix} 8 & 2 \\ 2 & 20 \end{bmatrix} \begin{bmatrix} 0 & -2.5 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 60 \end{bmatrix}$$

$$B_x^{\text{new}} = \begin{bmatrix} 0 & 0 \\ -2.5 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 5 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

These results are exactly the same as those obtained by the substitution method in Example 6.1. As before, the effect of the transformation has been to drive the remaining non-zero column of B_x to a null vector, ensuring that the new components vector does not depend on the undefined random vector ζ . The transformation has also annihilated the missing value, designated by #, in the observation vector.

7. Filtering Non-Stationary Time Series

Equation (3.1), which describes the initialisation of the Kalman filter, requires knowledge of the moments of $\underline{x}(0) = \underline{x}$. If the time series under consideration is stationary, then \underline{x} has a mean of zero and its variance satisfies the steady state equation

$$V_x = AV_xA' + V_u. \quad (7.1)$$

A method for solving this linear equation is described in Harvey (1991) for stationary time series. In the stationary case the mean and variance are both completely known so that

$$B_x = 0.$$

Most time series in business and economic applications are non-stationary, in which case the transition matrices possess unit eigenvalues. The steady state equation (7.1) then has no unique solution.

Example 7.1

The damped local trend model is defined by

$$\begin{aligned}
 \underline{z}(t) &= \underline{\ell}(t-1) + \underline{b}(t-1) + \underline{u}_1(t) \\
 \underline{\ell}(t) &= \underline{\ell}(t-1) + \underline{b}(t-1) + \underline{u}_2(t) \\
 \underline{b}(t) &= \phi \underline{b}(t-1) + \underline{u}_3(t) \\
 \underline{u}_j(t) &\sim \text{NID}(0, \sigma_j^2) \quad j = 1, 2, 3
 \end{aligned} \tag{7.2}$$

The notation is essentially the same as that used in Example 2.1. However, $\underline{b}(t)$ is a growth rate and ϕ is a damping parameter satisfying the condition $0 \leq \phi < 1$.

This model provides the framework for an approach to forecasting similar to damped exponential smoothing (Gardner, 1985). The growth rates $b(t)$ form a stationary series with a steady state distribution $N(0, \sigma_3^2 / (1 - \phi^2))$. The series formed by the levels $\underline{\ell}(t)$, however, is non-stationary. The variance of $\underline{\ell}(t)$ is arbitrarily large. The consequence is that the distribution of the seed vector $\underline{x} = (\underline{z}(0) \quad \underline{\ell}(0) \quad \underline{b}(0))'$ is only partially defined.

More generally, it is clear that the seed vector must be an ENRV rather than an NRV. Thus, for example, the NRVs shown in the state space model (2.1) then become ENRVs. The same thing happens to the NRVs in the Kalman filter specified in section 3. Instead of restating

these equations using asterisks to connote the ENRVs, we drop the asterisk on the understanding that quantities like $\underline{x}(t)$ now designate ENRVs rather than NRVs.

The fact that some elements of a seed vector \underline{x} possess arbitrary marginal distributions does not preclude the possibility that they have well-defined distributions after they are conditioned on successive values of $z(t)$. The extended algebraic system for ENRV's provides a systematic framework for implementing this conditioning process. Quite remarkably, no changes to the specification of the Kalman filter in (3.2) and (3.3) are required. All the required changes are made at the more basic level of the ENRV object.

The way this extended version of the Kalman filter is initialised depends on the type of model under consideration. In the stationary case, where the steady state moments of the components vector are completely defined, the seed is $\underline{x} = (0, V_x, 0)$. At the opposite extreme, in the non-stationary case where no components have a steady state distribution, the seed is $\underline{x} = (0, 0, I)$. In the partially defined case of the damped local trend example

$$\underline{x} = \left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_3^2/(1-\phi^2) + \sigma_1^2 & 0 & \sigma_3^2/(1-\phi^2) \\ 0 & 0 & 0 \\ \sigma_3^2/(1-\phi^2) & 0 & \sigma_3^2/(1-\phi^2) \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \right).$$

Another important insight into the operation of this modified version of the Kalman filter may be obtained by considering the following model.

Example 7.2

A model that contains a trend and seasonal effects takes the form

$$\begin{aligned}
\bar{z}(t) &= \bar{\ell}(t-1) + \bar{b}(t-1) + \bar{c}(t-m) + \underline{u}_1(t) \\
\bar{\ell}(t) &= \bar{\ell}(t-1) + \bar{b}(t-1) + \underline{u}_2(t) \\
\bar{b}(t) &= \bar{b}(t-1) + \underline{u}_3(t) \\
\bar{c}(t) &= \bar{c}(t-m) + \underline{u}_4(t)
\end{aligned}$$

where m designates the number of seasons per year and $\bar{c}(t)$ is the seasonal effect

associated with period t . This model has a lag $m > 1$. It can be converted to first-order form

to yield, in the case of a quarterly time series:

$$\begin{bmatrix} \bar{z}(t) \\ \bar{\ell}(t) \\ \bar{b}(t) \\ \bar{c}(t) \\ \bar{c}(t-1) \\ \bar{c}(t-2) \\ \bar{c}(t-3) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \bar{z}(t-1) \\ \bar{\ell}(t-1) \\ \bar{b}(t-1) \\ \bar{c}(t-1) \\ \bar{c}(t-2) \\ \bar{c}(t-3) \\ \bar{c}(t-4) \end{bmatrix} + \begin{bmatrix} \underline{u}_1(t) \\ \underline{u}_2(t) \\ \underline{u}_3(t) \\ \underline{u}_4(t) \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

This example underlies the additive Holt-Winters method of forecasting (Winters, 1960).

When the extended Kalman filter is applied to most time series, the arbitrary random variables are eliminated after a few iterations. However, in Example 7.2, a form of linear dependence exists between the level and the seasonal effects. The traditional observability condition is not satisfied. A consequence is that one of the arbitrary random variables is never eliminated as the algorithm progresses.

A closer study of this phenomenon indicates that the level and the seasonal effects, in any period, depend on this residual arbitrary random variable, but the growth rate does not. It transpires that the coefficients of this arbitrary random variable in the linear equations that

describe these relationships for the level and seasonal effects are always of equal magnitude, but have opposite sign. Using \square as a generic symbol for all those terms in a relationship that do *not* depend on a residual arbitrary random variable, an example of the relationships is

$$\underline{\ell}(t, t) = 0.1\delta_1 + \square \quad \text{and} \quad \underline{s}(t-3, t) = -0.1\delta_1 + \square .$$
 Then,

$$\underline{z}(t+1, t) = \underline{\ell}(t, t) + \underline{b}(t, t) + \underline{s}(t-3, t) = 0.1\delta_1 + \square - 0.1\delta_1 + \square = \square .$$
 The offending arbitrary

random variable disappears. This illustrates the point that the prediction distributions of the $z(t)$ are well defined, although the prediction distributions of the level and seasonal effects are never properly defined.

A common way of obtaining well-defined distributions of the level and seasonal effects is to impose the condition that the seed seasonal effects sum to zero. This provides the extra equation required to solve for the offending artificial variable and to eliminate it from subsequent expressions for the level and seasonal effects. The problem with this approach is that subsequent seasonal effects fail to satisfy the ‘sum to zero’ condition. In our view, a more sensible strategy is not to impose the condition on the seed seasonal effects. The fact that the linear dependency then persists at any point of time, suggests that it is legitimate to apply this condition to the output of the Kalman filter at any point of time. When this is done, the ‘adjusted’ seasonal effects always sum to zero. In order for this strategy to succeed, the adjusted seasonal effects and corresponding level should never be fed back into the Kalman filter itself.

To compare the proposed filter with its antecedents, the state space model (2.1) is partitioned into a measurement equation

$$\underline{x}_1(t) = A_{12}\underline{x}_2(t-1) + \underline{u}_1(t) \tag{7.3}$$

and transition equation

$$\underline{x}_2(t) = A_{22}\underline{x}_2(t-1) + u_2(t) \quad (7.4)$$

where $\underline{x}_1(t)$ consists of the observable quantities and $\underline{x}_2(t)$ consists of the unobservable quantities. Note that it has been assumed that $A_{11} = 0$ and $A_{21} = 0$ to get a model that more or less conforms to those used in earlier work. To expedite matters it is assumed that $\underline{x}_1(t)$ is scalar.

Harvey (1991, section 3.3.4) provides one possible solution to the initialisation problem. He assumes that

the state space model satisfies the observability conditions;

none of the observed values of $\underline{x}_1(1), \dots, \underline{x}_1(k)$ are missing.

Initially, he considers the case where all k elements of the seed vector $\underline{x}_2(0)$ have arbitrary distributions. By progressively applying (2.1), the observable series values $\underline{x}_1(1), \dots, \underline{x}_1(k)$ can be expressed as linear functions of the seed vector $\underline{x}_2(0)$. His approach is to substitute the observed values $x_1(1), \dots, x_1(k)$ of these quantities into these relationships to yield k linear equations in the k unknown seed variables. The observability condition ensures that these equations are non-singular. Thus, his method involves finding the solution to these equations and using it to seed the conventional Kalman filter. An adaptation of this approach to handle the case where the seed state vector is partially informative is then considered.

Harvey's method is predicated on assumptions (1) and (2) above. Since these assumptions can be violated in practice, his approach needs further adaptation before it can be used for a general-purpose implementation of the Kalman filter.

Another approach by de Jong (1991) can be restated in terms of ENRVs. It can then be seen to be quite similar to our filter, the primary difference being that the Gaussian elimination rule (6.7) is never used. Thus, in the run-in phase, the elements of the arbitrary random vector $\underline{\delta}$ do not progressively disappear from the equations. All pertinent quantities remain linear functions of $\underline{\delta}$. In particular, the observable quantities form a generalised linear regression, the elements of $\underline{\delta}$ being the regression coefficients. Additional equations are used to progressively calculate the ' $X'V^{-1}X$ ' matrix of the regression. On reaching that point where this weighted cross-product matrix becomes non-singular, the least-squares calculations are invoked, the results being used to eliminate $\underline{\delta}$. From this point onwards, his approach reverts to a conventional Kalman filter.

Ansley and Kohn (1985) provide yet another solution to the problem of filtering non-stationary time series. Koopman and Durbin (2000) recently proposed an algorithm in the same genre. These algorithms focus on the variance matrices of the state vectors, assuming that they can be written in the form $\kappa V_x^{(1)} + V_x^{(0)} + 0(1/\kappa)$ where κ is an arbitrarily large number and $0(1/\kappa)$ is a function that converges to zero as κ tends to infinity. It is possible to re-define an ENRV by $\underline{x} = (\underline{\mu}_x, V_x^{(0)}, V_x^{(1)})$. In effect the role of the B_x matrix is replaced by the $V_x^{(1)}$. Multiplication is now defined by $A\underline{x} = (A\underline{\mu}_x, AV_x^{(0)}A', AV_x^{(1)}A')$. The details of their algorithms are eschewed here. In the final analysis, however, we believe that their

approaches will involve higher computational loads. The redefined form of multiplication alone involves more operations than (6.3).

When the emphasis is on computational stability, square-root filters (Kaminsky, Bryson, and Schmidt, 1971) are preferred to the Kalman filter. A square-root filter in Snyder and Saligari (1996) is particularly pertinent in our context. It relies on fast Givens transformations to preserve the triangular structure of the square-root variance matrices. It was shown that the fast Givens transformations collapse to conventional Gaussian elimination during the run-in phase. It is this result that provides the rationale for our use of Gaussian elimination in section 6.

8. Computer Implementation: An Object Oriented Approach

The form of the algorithm, as outlined in this paper, was conceived while implementing the Kalman filter using the matrix-oriented computer programming language Matlab (www.mathworks.com). The facilities for object-oriented programming in this language provided an opportunity to explore data aggregation possibilities. The concept of an NRV object arose as a result.

A structured array is used to represent a NRV object in the implementation. It has three fields containing the mean, variance and the matrix of coefficients for the artificial components.

The notation used is kept as close as possible to the mathematical notation. Thus *xtilde* designates an NRV object. Its properties are designated by *xtilde.mu*, *xtilde.V* and *xtilde.B*.

The standard Matlab operators +, * and | are overloaded to work on moments-objects.

```

Function xtilde =KalmanFilter(x, A, utilde, xtilde0);
xtilde = xtilde0;
For t = 1:size(x,2);
    xtilde = A* xtilde + utilde;
    xtilde = xtilde | x(:,t);
End;

```

Figure 2. Kalman Filter

The code for the Kalman filter, based on the equations (3.1), (3.2) and(3.3), is shown in Figure 2. It is relies on inputs *utilde* and *xtilde0* for the disturbance and seed state vectors. These particular inputs are created outside the Kalman filter function using one of the variations of the polymorphic constructor function N shown in Table 1. The details of the code for this constructor function are eschewed in the interests of saving space. Data, to be filtered, is fed to the function through the $k \times n$ matrix *x*. Elements of *x* corresponding to unobserved components are treated as missing values. They are designated in Matlab using the not-a-number notation *NaN*. The output, *xtilde*, represents $\underline{x}(n, n)$.

<i>xtilde</i> = N(<i>mu</i> , <i>V</i>)	Creates a NRV object from a specified mean vector and variance matrix. The component $B = 0$.
<i>xtilde</i> = N (<i>mu</i> , <i>V</i> , <i>B</i>)	Creates an ENRV object.
<i>xtilde</i> = N (<i>x</i>)	Converts a vector to its equivalent ENRV object with a zero variance. Missing values are reflected in the <i>B</i> matrix.
<i>ytilde</i> = N (<i>xtilde</i>)	Creates an ENRV object <i>ytilde</i> equal to the ENRV object <i>xtilde</i> .

Table 1. Forms of the N Constructor Function

On encountering the + operator between two NRV objects in the code, Matlab automatically invokes the plus function shown in Figure 3. This function is also polymorphic because it allows the inputs to be NRVs or fixed vectors. Any fixed vector is converted to its random vector object counterpart in the first two lines of the body of the function.

Similarly, when * is encountered in a statement between a fixed matrix A and a random vector object x_{tilde} , a function called ‘times’, based on the definition (6.3), is invoked. The code for this operator is a relatively simple adaptation of the function shown in Figure 3.

```
function y = plus(x1, x2);  
x1 = N(x1);  
x2 = N(x2);  
y = N(x1.mu + x2.mu, x1.V + x2.V, x1.B + x2.B);
```

Figure 3. Code for the overloaded + operator.

Matters are more complex for the conditioning operator |. Matlab normally employs the symbol ‘|’ to represent the logical operator ‘or’. But this operator is overloaded and given a completely new interpretation by creating a function called ‘or’ based on the conditioning rules associated with (6.5). On encountering a statement such as $y_{tilde} = x_{tilde} | x$, Matlab inspects the term on the left of the | operator. If this term is a random vector object it calls the new function, and despite its name, applies the conditioning rules contained within it.

The functions ‘N’, ‘plus’, ‘times’ and ‘or’ are placed in files called N.m, plus.m, times.m and or.m respectively. These files are in turn placed in a folder called @N. The @ symbol is required by Matlab to designate folders that contain the functions defining objects. The string

following the @ symbol is the name of an object. The file for the associated constructor function has the same name. When Matlab encounters statements containing random vector objects, it automatically searches the @N folder for the files defining the specified operations.

The file containing the Kalman filter, as depicted in Figure 2, is not placed in the @N folder. It does not define a random vector object. Nor does it define associated operators. It is a ‘client’ of the random vector object and so it is placed in the folder hierarchy above the @N folder.

The version of the Kalman filter presented here, by relying on matrices like *wbar*, appears to involve higher computational loads than its more conventional counterpart. It is envisaged, however, that key matrices would be stored in sparse form and that sparse matrix arithmetic would be used to eliminate redundant calculations involving zeroes. An approach along these lines can be implemented in Matlab, using its sparse matrix facilities, with only slight modifications to the above code. On doing so, the computational loads become comparable with traditional implementations of the Kalman filter.

9. Conclusions

In this paper, a specification has been proposed for the state space model that consolidates the usual measurement and transition equations into a single equation (2.1). All components associated with a particular period, both observable and unobservable, are placed in a single random vector. Furthermore, all information pertaining to a multivariate normal distribution is consolidated into an entity called a normal random vector object. By defining appropriate addition, multiplication and conditioning operators on NRVs, a new algebraic system was obtained. When applied in the context of a dynamic linear statistical model, the algebraic

system led to an elegantly simple statement of the Kalman filter for stationary time series. A new augmented filter was derived for non-stationary time series.

It is possible to bypass the use of NRV objects and present the proposed filter in more conventional terms. However, we opted for NRVs because the resulting mathematics lends itself to an object-oriented implementation on computers. The main advantage is that the code then possesses a more transparent structure. The advantages, however, probably extend beyond computer implementations. Our conjecture is that the use of NRV objects reduces learning overheads, something that can only be tested in the classroom. It is also anticipated that NRVs and the associated operations will have useful applications beyond the scope of the Kalman filter.

The new filter has certain advantages over its nearest antecedent: Harvey's (1991) simultaneous equations method. It can be used when there are missing values. It can be used on models such as the one involving seasonal effects that violate the observability condition. It can also be used when the seed distribution is completely defined, partially defined, or completely undefined.

References

- Anderson, T.W. (1958). *An Introduction to Multivariate Statistical Analysis*, New York: John Wiley.
- Ansley, C.F. and R. Kohn, (1985). "Estimation, filtering and smoothing in state space models with incompletely specified initial conditions", *Ann. Stat.* 13, 1286-1316.
- Brown, R.G. (1959). *Statistical Forecasting for Inventory Control*, New York: Mc.Graw-Hill.
- De Jong, P. (1991), "The diffuse Kalman filter", *Ann. Stats.*, 19, 1073-1083.
- Duncan, D. B., and S. D. Horn (1972), "Linear dynamic recursive estimating from the viewpoint of regression analysis", *Journal of the American Statistical Association*, 67, 815-821.
- Gardner, E.S. and E. McKenzie (1985) "Forecasting trends in time series", *Management Science* 31, 1237-1246.
- Harvey, A. C. (1991), *Forecasting, Structural Time Series and the Kalman Filter*, Cambridge: Cambridge University Press.
- Kalman, R.E. (1960) "A new approach to linear filtering and prediction problems", *Journal of Basic Engineering, Transactions ASME, Series D*, 82, 35-45.
- Kalman, R.E. and R.S. Bucy. (1961). 'New results in linear filtering and prediction theory', *Journal of Basic Engineering, Transactions ASME, Series D*, 83, 95-108.
- Kaminsky, P.G., Bryson, A.E. and Schmidt, S.F. (1971) "Discrete square root filtering: a survey of current techniques", *IEEE Trans. Autom. Control AC-16*, 727-736.
- Koopman, S.J. and J.Durbin (2000) "Fast filtering and smoothing for multivariate state space models", *Journal of Time Series Analysis*, 21, 281-296.
- Lütkepohl, H. (1991) *Introduction to multiple time series analysis*, Springer-Verlag.
- Meinhold, R. J. and Singpurwalla, N. D. (1983) "Understanding the Kalman Filter", *The American Statistician*, 37, 123-127.
- Muth, J.F. (1960) "Optimal properties of exponentially weighted forecasts", *Journal of the American Statistical Association*, 55, 299-305.
- Schweppe, F. (1965), "Evaluation of likelihood functions for Gaussian signals," *IEEE Transactions on Information Theory*, 11, 61-70.
- Snyder, R. D., and Saligari, G. (1996), "Initialisation of the Kalman filter with partially diffuse initial conditions", *Journal of Time Series Analysis*, 17, 409-424.
- Winters, P.R. (1960), "Forecasting sales by exponentially weighted moving averages", *Management Science*, 6, 324-342.

Appendix

This Appendix is devoted to deriving the conditioning rules (4.6) and (6.6). All random quantities are assumed to be non-arbitrary, so the NRVs as defined in Section 4 are used. It is important to note, however, that we revert to using +, -, | and the implied multiplication operators in their traditional sense. The derivation is based on the following standard result from the theory of conditioning for Gaussian distributions (Anderson, 1958):

If \underline{y} and \underline{x} are correlated Gaussian random vectors then

$$E(\underline{x} | y) = \mu_x + V_{xy} V_{yy}^{-1} (y - \mu_y) \quad (\text{A1})$$

and

$$\text{Var}(\underline{x} | y) = V_{xx} - V_{xy} V_{yy}^{-1} V_{yx}. \quad (\text{A2})$$

Suppose \underline{y} is formed from some of the elements of \underline{x} using a 'selector' matrix S .

More specifically,

$$\underline{y} = S\underline{x}. \quad (\text{A3})$$

Then (A1) and (A2) become:

$$E(\underline{x} | Sx) = \mu_x + V_{xx} S' (S V_{xx} S')^{-1} S (x - \mu_x) \quad (\text{A4})$$

and

$$\text{Var}(\underline{x} | Sx) = V_{xx} - V_{xx} S' (S V_{xx} S')^{-1} S V_{xx}. \quad (\text{A5})$$

Letting

$$W = V_{xx} S' (S V_{xx} S')^{-1} S \quad (\text{A6})$$

it is possible rewrite (A4) as:

$$E(\underline{x} | Sx) = (I - W) \mu_x + Wx. \quad (\text{A7})$$

Hypothesise from this that

$$(\underline{x} | Sx) = (I - W) \underline{x} + Wx. \quad (\text{A8})$$

The corresponding variance matrix is then

$$\text{Var}(\underline{x} | Sx) = (I - W)V_{xx}(I - W)' \quad (\text{A9})$$

Expanding (A9) and using (A6) to simplify this expression, the variance reduces to (A5). This validates the hypothesis (A8).

In the case where Sx consists only of the p th element of x , then $S = e'_p$. Substituting this into (A6) gives (4.6) and (6.6) as required.

In conventional forms of the Kalman filter it is normal practice to use the analogue of (A5) to compute the variances. On finite precision computing machines, however, truncation errors can cause the variance matrix to lose its non-negative definiteness property. An alternative is to use (A9) to compute the variance. Because it only involves squaring operations, the computed variance matrix remains non-negative definite in the presence of truncation errors. This explains the slightly unorthodox way of computing the variance in the Equations (4.5) and (6.5).