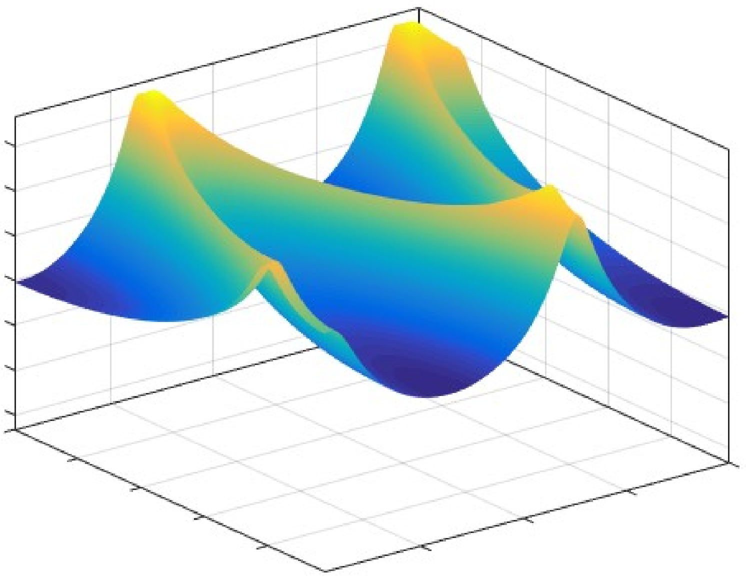
The CAPTAIN TOOLBOX for System Identification, Time Series Analysis, Forecasting and Control

<u>Guide to TVPMOD</u>

<u>Time Variable Parameter Models</u>



Handbook Part 2
3 July 2017
C J Taylor, D J Pedregal and P C Young

AUTHORSHIP

Current developers:

Dr. Wlodek Tych and Emeritus Prof. Peter C. Young

Lancaster Environment Centre (LEC), Lancaster University,
Lancaster, LA1 4YQ, United Kingdom.

Dr. C. James Taylor

Engineering Department, Lancaster University,
Lancaster, LA1 4YR, United Kingdom.

Co-author:

Prof. Diego J. Pedregal

Escuela Tenica Superior de Ingenieros, Industriales Edificio Politenica,
Campus Universitario s/n, 13071 Ciudad Real, Spain.

Additional contributors:

Dr. Paul G. McKenna and Dr. Renata Romanowicz

GLOSSARY

ACF	Autocorrelation Function
AR	Auto-Regression
ARMA	Auto-Regression Moving-Average
ARIMA	Auto-Regression Integrated Moving-Average
ARX	Auto-Regression with eXogenous variables
BSM	Basic Structural Model
DAR	Dynamic Auto-Regression
DARX	Dynamic Auto-Regression with eXogenous variables
DBM	Data-Based Mechanistic
DHR	Dynamic Harmonic Regression
DLR	Dynamic Linear Regression
DTF	Dynamic Transfer Function
FIS	Fixed Interval Smoothing
GRW	Generalised Random Walk
IRW	Integrated Random Walk
KF	Kalman Filter
LLT	Local Linear Trend
ML	Maximum Likelihood
NAN	Not-A-Number
NMSS	Non-Minimal State Space
NVR	Noise Variance Ratio
PACF	Partial Autocorrelation Function
PIP	Proportional-Integral-Plus
RIV	Refined Instrumental Variable
RIVSID	Refined Instrumental Variable System Identification (CAPTAIN folder)
RW	Random Walk
SDARX	State Dependent Auto-Regression with eXogenous variables
SDP	State Dependent Parameter
SDTF	State Dependent Transfer Function
SISO	Single Input, Single Output
SRIV	Simplified Refined Instrumental Variable
SRM	Smoothed Random Walk
SS	State Space
TDC	True Digital Control
TDCONT	True Digital CONTrol (CAPTAIN folder)
TF	Transfer Function
TVP	Time Variable Parameter
TVPMOD	Time Variable Parameter MODels (CAPTAIN folder)
UC	Unobserved Components

CONTENTS

CHAPTER 1
INTRODUCTION

The CAPTAIN Toolbox is a collection of MATLAB functions for non-stationary time series analysis, forecasting and control. The toolbox is useful for system identification, signal extraction, interpolation, forecasting, data-based mechanistic modelling and control of a wide range of linear and non-linear stochastic systems. The toolbox consists of three modules, organised into three folders (or directories) as follows:

- **TVPMOD**: **Time Variable Parameter** (TVP) **MODels**. For the identification of unobserved components models, with a particular focus on state-dependent and time-variable parameter models (includes the popular dynamic harmonic regression model).

- **RIVSID**: **Refined Instrumental Variable** (RIV) **System Identification** algorithms. For optimal RIV estimation of multiple-input, continuous- and discrete-time Transfer Function models.

- **TDCONT**: **True Digital CONTrol** (TDC). For multivariable, non-minimal state space control, including pole assignment and optimal design, and with backward shift and delta-operator options.

The present document is a guide to the TVPMOD module. The chapter headings in this guide follow a logical structured progress through the relevant methodology, using worked examples. TVP modelling is introduced in Chapter 2. Here, the filtering algorithm, smoothing algorithm, generalised random walk model and hyper-parameter optimisation routines are described. Chapter 2 presents the models in their most general state space form, while the following three chapters introduce various special cases, namely: unobserved component models in Chapter 3; dynamic regression models in Chapter 4; and state dependent parameter models in Chapter 5.

1.1 Use of the CAPTAIN Toolbox

1. Wherever the CAPTAIN Toolbox has been used to generate results which have then been used in any written work, please reference it using the text below:

Taylor, C.J., Pedregal, D.J., Young, P.C. and Tych, W. (2007) Environmental Time Series Analysis and Forecasting with the Captain Toolbox, *Environmental Modelling and Software*, **22**, pp. 797-814 (http://dx.doi.org/doi:10.1016/j.envsoft.2006.03.002).

2. The CAPTAIN Toolbox is supplied in good faith and whilst all efforts have been made to assure that it is free from errors and bugs, the authors and Lancaster University accept no liability for erroneous results obtained through the use of the Toolbox.
3. Installation of the CAPTAIN Toolbox is carried out at the user's own risk and neither the authors nor Lancaster University accept any responsibility in the unlikely event of installation resulting in detrimental effects to the user's computer hardware or software.
4. The user may not redistribute the Toolbox to any third party.
5. Ownership of the Toolbox is retained by the authors. The web download is an evaluation copy of the Toolbox. If you wish to continue using the Toolbox beyond the evaluation period, please contact C. James Taylor at the address below.

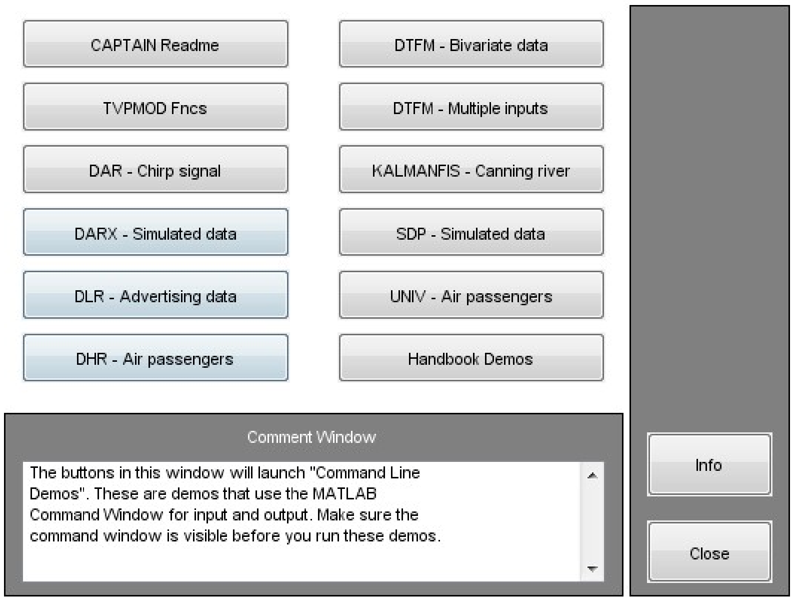
   E-mail: c.taylor@lancaster.ac.uk
   Web: http://www.lancs.ac.uk/staff/taylorcj/
6. Any part of the CAPTAIN Toolbox may be used for scientific or educational purposes. For commercial applications, permission is required from the authors.
7. CAPTAIN is provided without formal support, although questions and bug reports can be emailed to the authors.

1.2 Getting Started

For installation instructions and toolbox overview, please see the **Getting Started Guide**. Since CAPTAIN is largely a Command Line toolbox, it is also assumed that the reader is already familiar with basic MATLAB usage, such as loading data and plotting graphs. To obtain a full list of user functions, type **help tvpmod** in the Command Window, replacing 'tvpmod' if necessary with the actual name of the folder (directory) chosen when you first installed the toolbox. To illustrate, the first few lines of the function list are:

```
>> help tvpmod
  Captain Toolbox
  Time Variable Parameter Models

  Unobserved Components Models.
    dhr       - Dynamic Harmonic Regression analysis.
    dhropt    - DHR hyper-parameter estimation.
    irwsm     - Integrated Random Walk smoothing and decimation.
    irwsmopt  - IRWSM hyper-parameter estimation.
    univ      - Trend with Auto-Regression component.
    univopt   - Trend with AR hyper-parameter estimation.
```



The buttons in this window will launch "Command Line Demos". These are demos that use the MATLAB Command Window for input and output. Make sure the command window is visible before you run these demos.

Figure 1.1 Demos for the TVPMOD module obtained by typing **tvpdemo** at the MATLAB Command Line.

To open the Command Line demos for the TVPMOD module,

```
>> tvpdemo
```

A straightforward graphical user interface provides access to the demos (see Fig. 1.1). The latter utilise the Command Window for input and output, as well as generating graphs in a separate Figure Window, so make sure both of these are visible. Demo scripts can be opened in the MATLAB editor.

Each function includes help information obtained in the usual manner e.g. **help irwsm**. Note that default input arguments are shown in parenthesis while (*) implies no default (hence a user input argument is always required for this variable). Some functions include more information, readable using e.g. **type irwsm** or **edit irwsm** (see Chapter 3). Finally, entering the function name without any arguments summarises the calling syntax.

1.3 Caveat

CAPTAIN is presently developed for MATLAB R2017b onwards, although some backwards compatibility is maintained at the discretion of the authors. Note that, because of updates over time to MATLAB and CAPTAIN, you may obtain different numerical results when using the toolbox than shown by the examples below. Also, please check the help information within MATLAB for the latest calling syntax and default values of each toolbox function, since these may change over time.

CHAPTER 2
STATE SPACE MODELS

This chapter introduces one of the main methodological tools in CAPTAIN, namely the State Space (SS) model. Indeed, most of the models in the toolbox, though not all, are implemented in such a SS form. Originating from the state-variable method of describing differential equations, the SS approach has subsequently been developed for modelling by researchers in many different scientific disciplines and is, perhaps, the most natural and convenient approach for use with computers. It is a general and flexible tool that encompasses numerous time series models.

In fact, a number of models uniquely available in CAPTAIN are inherently based on such state space methods and, therefore, always require a SS formulation. This includes the entire category of Time Variable Parameter (TVP) models considered in Chapter 4. There are other practical situations in which it is particularly convenient, though not essential, to also use a SS model, e.g. when there are missing values in the time series or when interpolation, forecasting and backcasting operations are required. Finally, in certain cases, the SS form simply offers a solution that is identical to other, sometimes better known, conventional approaches.

For these reasons, whenever a SS form is necessary or convenient, CAPTAIN uses it, while in a few exceptional cases, it is avoided because other superior approaches are available. For example, the instrumental variable method for the estimation of fixed parameter Transfer Function (TF) models (in the SYSID module of CAPTAIN) replaces the state space- based Maximum Likelihood (ML) approach by default, unless the latter is specifically chosen. However, such issues are usually transparent to the user, since CAPTAIN chooses the optimal modelling strategy internally.

The present chapter provides an introduction to SS methods and may, therefore, be regarded as the broad theoretical basis for the special cases discussed subsequently. In this regard, a number of standard topics are briefly covered, typical of many publications and textbooks in this field. However, several aspects are novel and exclusive to CAPTAIN and these are highlighted in the text where appropriate.

Almost all of the modelling functions in CAPTAIN might be quoted in this chapter because they are particular cases of the general SS framework! However, to help the reader digest the general principles, we will restrict the present discussion to the simplest possible models, leaving the more advanced cases for subsequent chapters. In this regard, the key modelling functions covered below are **irwsm** and **irwsmopt**, while **fcast**, **reconst**, **acf** and **histon** are also introduced for data preparation and diagnostics. Together, these tools are useful for exploratory analysis or in situations where the *a priori* knowledge about the system is minimal, as will be seen in the later worked examples.

2.1 The State Space framework

A SS system is composed of two sets of equations, namely: (i) the so called *State Equations* (represented as a single equation in vector-matrix form below), that reflect all the dynamic behaviour of the system by relating the current value of the states to their past values, together with any deterministic and stochastic inputs; and (ii) the *Observation Equation* that defines how these state variables are related to the observed data. Although there are a number of different SS formulations possible, the one favoured in CAPTAIN is:

$$\begin{aligned} &\text{State Equations} &&: \quad \mathbf{x}_t = \mathbf{F}\mathbf{x}_{t-1} + \mathbf{G}\boldsymbol{\eta}_{t-1} \\ &\text{Observation Equation} &&: \quad y_t = \mathbf{H}_t\mathbf{x}_t + e_t \end{aligned} \tag{2.1}$$

where y_t is the stochastic observed variable; $\mathbf{x}_t$ is an n dimensional stochastic state vector; $\boldsymbol{\eta}_t$ is an k dimensional vector of system disturbances; and e_t is a vector of zero mean white noise variables (measurement noise). $\mathbf{F}, \mathbf{G}$ and $\mathbf{H}_t$ are, respectively, the *nxn*, *nxk*, and *1xn* non-stochastic system matrices.

The following assumptions apply to this system:

- $\boldsymbol{\eta}_t$~iid $N(\mathbf{0},\mathbf{Q})$; e_t~iid $N(0,\sigma^2)$; $\mathrm{cov}(\boldsymbol{\eta}_t, e_t) = \mathbf{0}$.
- The initial stochastic state $\mathbf{x}_0$ is independent of $\boldsymbol{\eta}_t$ and e_t for every *t*.
- The system matrices $\mathbf{F}, \mathbf{G}, \mathbf{H}_t, \mathbf{Q}$ and σ^2 are known (or have been previously estimated in some way) whereas the initial conditions for the states and their covariance matrix ($\mathbf{x}_0$ and $\mathbf{P}_0$ respectively) are unknown.

The main reason for this formulation is that, under these straightforward conditions, the associated recursive algorithms employed to estimate the state vector from measured data provide the optimal solution, in the sense that such estimators minimise the Mean Square

Error (see e.g. Young, 1984; or Harvey, 1989). In CAPTAIN, these recursive algorithms are the Kalman Filter (KF; Kalman, 1960) and Fixed Interval Smoothing (Bryson and Ho, 1969), as discussed in Section 2.2 below.

Numerous results may be found in the literature for various relaxations of the above conditions (see e.g. Durbin and Koopman, 2001). One particularly straightforward and useful case is when the gaussianity assumptions on the perturbations are dropped. In this case, the state estimates provided by the KF/FIS are still optimal in the sense that they minimise the mean square error within the class of all linear estimators.

The Generalised Random Walk model

The following Generalised Random Walk (GRW) model, which is one of the simplest SS models based on (2.1), is used extensively in CAPTAIN,

$$\begin{pmatrix} x_{1t} \\ x_{2t} \end{pmatrix} = \begin{pmatrix} \alpha & \beta \\ 0 & \gamma \end{pmatrix} \begin{pmatrix} x_{1t-1} \\ x_{2t-1} \end{pmatrix} + \begin{pmatrix} \eta_{1t} \\ \eta_{2t} \end{pmatrix} \qquad T_t = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_{1t} \\ x_{2t} \end{pmatrix} \tag{2.2}$$

Here, α, β and γ are constant parameters; T_t is a smoothed signal component consisting of the first state x_{1t}; and x_{2t} is a second state variable (generally known as the 'slope'); while η_{1t} and η_{2t} are zero mean, serially uncorrelated white noise variables with constant block diagonal covariance matrix $\mathbf{Q}$, as stated in the general formulation above.

This model subsumes a number of special cases that have received specific names in the literature. The main ones are the Random Walk (RW: $\alpha = 1$; $\beta = \gamma = 0$; $\eta_{2t} = 0$); Smoothed Random Walk ($0 < \alpha < 1$; $\beta = \gamma = 1$; $\eta_{1t} = 0$); the Integrated Random Walk (IRW: $\alpha = \beta = \gamma = 1$; $\eta_{1t} = 0$); the Local Linear Trend (LLT: $\alpha = \beta = \gamma = 1$); and the Damped Trend ($\alpha = \beta = 1$; $0 < \gamma < 1$).

When any one of these options is combined with additive noise in an observation equation, the resulting time series model might be called a 'GRW plus noise' model. One such case is the 'RW plus noise' model, ideal for the estimation of a time varying mean. Here, the SS representation takes the following form:

$$\begin{array}{ll} \text{State Equation} & : \; x_t = x_{t-1} + \eta_{t-1} \\ \text{Observation Equation} & : \; y_t = x_t + e_t \end{array} \tag{2.3}$$

The state equation may instead be written as $(1 - L)x_t = \eta_{t-1}$ where L is the backward-shift operator. Substituting in the observation equation, we obtain the so called *reduced form* of the model, i.e. a random walk model with added observational noise as shown below,

$$y_t = \frac{\eta_{t-1}}{(1-L)} + e_t \tag{2.4}$$

Similarly, the 'IRW plus noise' model, useful for extracting smooth trends to non-stationary time series, may be developed as follows,

$$\begin{aligned} &\text{State Equations} &&: \begin{pmatrix} x_{1t} \\ x_{2t} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_{1t-1} \\ x_{2t-1} \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \eta_{t-1} \\ &\text{Observation Equation} &&: y_t = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_{1t} \\ x_{2t} \end{pmatrix} + e_t \end{aligned} \tag{2.5}$$

Here, the two state equations that may instead be written as,

$$\begin{aligned} (1-B)x_{1t} &= x_{2t-1} \\ (1-B)x_{2t} &= \eta_{t-1} \end{aligned} \tag{2.6}$$

In this case, substituting the second state equation into the first, and then into the observation equation, we obtain the reduced form,

$$y_t = \frac{\eta_{t-2}}{(1-L)^2} + e_t \tag{2.7}$$

This IRW model with observational noise will be utilised in Example 2.1 below. First, however, an algorithm for the estimation of the states is required.

2.2 State Estimation

Given the model (2.1), in which all the system matrices are known, the estimation problem becomes one of finding the optimal distribution of the state vector, conditional to all the data in a sample. In the case of Gaussian disturbances, the distribution is completely characterised by the first and second order moments, i.e. the mean and variance, and most algorithms that perform this operation concentrate on the estimation of these two moments. Such tools include the Kalman Filter (KF, Kalman, 1960, Kalman and Bucy, 1961) and Fixed Interval Smoothing (FIS, e.g. Bryson and Ho, 1969) algorithms stated below.

1. Forward Pass Filtering Equations

Prediction:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}\hat{\mathbf{x}}_{t-1}$$
$$\hat{\mathbf{P}}_{t|t-1} = \mathbf{F}\hat{\mathbf{P}}_{t-1}\mathbf{F}^{T} + \mathbf{G}\mathbf{Q}_{r}\mathbf{G}^{T}$$

Correction: (2.8)

$$\hat{\mathbf{x}}_{t} = \hat{\mathbf{x}}_{t|t-1} + \hat{\mathbf{P}}_{t|t-1}\mathbf{H}_{t}^{T}\left[1 + \mathbf{H}_{t}\hat{\mathbf{P}}_{t|t-1}\mathbf{H}_{t}^{T}\right]^{-1}\left\{y_{t} - \mathbf{H}_{t}\hat{\mathbf{x}}_{t|t-1}\right\}$$
$$\hat{\mathbf{P}}_{t} = \hat{\mathbf{P}}_{t|t-1} - \hat{\mathbf{P}}_{t|t-1}\mathbf{H}_{t}^{T}\left[1 + \mathbf{H}_{t}\hat{\mathbf{P}}_{t|t-1}\mathbf{H}_{t}^{T}\right]^{-1}\mathbf{H}_{t}\hat{\mathbf{P}}_{t|t-1}$$

2. Backward Pass Smoothing Equations

$$\hat{\mathbf{x}}_{t|N} = \mathbf{F}^{-1}\left\lfloor\hat{\mathbf{x}}_{t+1|N} + \mathbf{G}\mathbf{Q}_{r}\mathbf{G}^{T}\mathbf{L}_{t}\right\rfloor$$
$$\mathbf{L}_{t} = \left[\mathbf{I} - \hat{\mathbf{P}}_{t+1}\mathbf{H}_{t+1}^{T}\mathbf{H}_{t+1}\right]^{T}\left[\mathbf{F}^{T}\mathbf{L}_{t+1} - \mathbf{H}_{t+1}^{T}\{y_{t+1} - \mathbf{H}_{t+1}\hat{\mathbf{x}}_{t+1}\}\right] \qquad (2.9)$$
$$\hat{\mathbf{P}}_{t|N} = \hat{\mathbf{P}}_{t} + \hat{\mathbf{P}}_{t}\mathbf{F}^{T}\hat{\mathbf{P}}_{t+1|t}^{-1}\left[\hat{\mathbf{P}}_{t+1|N} - \hat{\mathbf{P}}_{t+1|t}\right]\hat{\mathbf{P}}_{t+1|t}^{-1}\mathbf{F}\hat{\mathbf{P}}_{t}$$

with $\boldsymbol{L}_{N} = \mathbf{0}$. Note that the FIS algorithm is in the form of a backward recursion operating from the end of the sample set to the beginning. The main difference between the KF and FIS algorithms (2.8)-(2.9) utilised in CAPTAIN, and more conventional filtering/smoothing algorithms found in some other toolboxes, are the *nxn* Noise Variance Ratio (NVR) matrix $\mathbf{Q}_{r}$ and the *nxn* matrix $\hat{\mathbf{P}}_{t}$ defined below,

$$\mathbf{Q}_{r} = \frac{\mathbf{Q}}{\sigma^{2}}; \qquad \hat{\mathbf{P}}_{t} = \frac{\hat{\mathbf{P}}_{t}^{*}}{\sigma^{2}} \qquad (2.10)$$

Here, $\hat{\mathbf{P}}_{t}^{*}$ is the error covariance matrix associated with the state estimates $\hat{\mathbf{x}}_{t}$. In most of the models implemented in CAPTAIN, the NVR matrix $\mathbf{Q}_{r}$ is diagonal.

Forward Pass Filtering

For a data set of *T* samples, the KF algorithm (2.8) runs forward and returns a 'filtered' estimate of the state vector and its covariance matrix ($\hat{\mathbf{x}}_{t}$ and $\hat{\mathbf{P}}_{t}$, respectively) at every sample *t*, based on the time series data up to sample *t*. These estimates are computed in two steps. In the first instance, the one step ahead forecast for the mean value of the state vector and its covariance matrix ($\hat{\mathbf{x}}_{t|t-1}$ and $\hat{\mathbf{P}}_{t|t-1}$, respectively) are obtained from the prediction equations, using the model alone. Secondly, these estimates are updated by means of the correction equations, as each new data sample becomes available.

One interesting feature of the KF worth mentioning at this juncture, is that the state estimates (obtained using the first equation of each set of prediction and correction equations) depend on the previous state estimates, its covariance matrix and the data. By contrast, the current estimate of the covariance matrix itself, does not depend on either the state estimates or the data, but only on the model and the previous estimate of the covariance matrix. Another noteworthy point, is that significant saving in computational effort are obtained in CAPTAIN by the realisation that repeated operations are involved in the calculation of the required estimates, especially in the correction equations.

There are two intermediate variables calculated by the KF that are of particular importance (as will become clear in the following section 2.3), namely:

$$\begin{aligned} \hat{v}_t &= y_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-1} \\ \hat{f}_t &= 1 + \mathbf{H}_t \hat{\mathbf{P}}_{t|t-1} \mathbf{H}_t^T \end{aligned} \tag{2.11}$$

The first term above is the innovations sequence, or the one-step-ahead forecasting errors of the model, while the second is the variance of these innovations, where the latter are equal to the variance of the one-step-ahead forecasts. Both variables are scalar, so that the matrix inversions required in the KF correction equations are actually very simple in computational terms. The normalised innovations are used in many diagnostic tests, since, recalling the original formulation of the SS model (2.1), they should be iid $N(0, \sigma^2)$ in a correctly specified model.

When missing data are encountered within the data set, the correction equations are redundant and missing samples are effectively interpolated by using the filtered estimates. In the same manner, forecasts may be produced by artificially adding 'missing' data at the end of the series. In this case, the values of $\hat{v}_t$ and $\hat{f}_t$ outside the sample span are now the true multiple-steps-ahead forecasting errors and associated variance respectively, and may be used for computation of the confidence intervals.

Fixed interval smoothing

The FIS algorithm in (2.9) normally runs backwards after the filtering step and yields a 'smoothed' estimate of the state vector and its covariance matrix based, at every sample *t*, on all *T* samples of the data. This means that, as more information is used in the FIS algorithm with respect to the KF estimates, its Mean Square Error cannot be greater. When missing data are encountered, an interpolation is generated based on the data at both sides of the gap. Finally, if the missing observations are at the beginning of the sample, the FIS algorithm generates backcasts of the time series.

The FIS algorithm (2.9) is utilised by many software packages and is the default so called 'Q-algorithm' in CAPTAIN. However, an additional FIS algorithm is also available as an option in the toolbox. It should be used in special cases when numerical problems arise with certain models. Essentially, if a solution fails to converge, the appropriate input argument can be changed to select this alternative ‘P-algorithm’, obtained by replacing the first equation in (2.9) by,

$$\hat{\mathbf{x}}_{t|N} = \hat{\mathbf{x}}_{t+1|t} - \hat{\mathbf{P}}_t \mathbf{F}^T \mathbf{L}_{t+1} \tag{2.12}$$

Here, the new estimates of the state vector are based on the filtered ones, while in (2.9) it is calculated recursively from the previous estimate of the state vector. Furthermore, (2.12) does not require inversion of the **F** matrix. Both algorithms are discussed in detail by previous publications (see e.g. Young, 1984; Harvey, 1989; Ng and Young, 1990).

Example 2.1 Estimation of a trend for the air passenger series

The simple IRW plus noise model (2.7) has long been used for smoothing time series in the economics literature. It is equivalent to the well-known Hodrick-Prescott filter (HP; Hodrick and Prescott, 1997). This filter has been used by many researchers in the area of empirical business cycles, and the similarities between both filters was highlighted long ago (recently reviewed by Young and Pedregal, 1996).

Many researchers use fixed values of the smoothing constant depending on the frequency rate of the data. In CAPTAIN, however, the relationship between the smoothing constant (the NVR in this context) and the cut-off frequency properties of the low-pass filter is made explicit, and is easily generalisable to any type of model in SS form. The advantage is that the user may choose the NVR according to their particular needs and the properties of the data, it is not fixed by any prior conceptions. For example, given the RW family of models defined by (cf. (2.6) and (2.7)),

$$y_t = \frac{\eta_{t-j}}{(1-L)^j} + e_t \tag{2.13}$$

the cut-off frequency for 50% of spectral power is given by,

$$w = \arccos\left(1 - 0.5\ \mathrm{NVR}^{1/j}\right) \tag{2.14}$$

with $\mathrm{NVR} = \frac{\sigma_\eta^2}{\sigma^2}$ (Young and Pedregal, 1996).

In this case, $\mathbf{Q}_r$ in equation (2.10) is the scalar state noise variance σ_η^2. For an IRW filter (i.e. $j = 2$), Table 2.1 shows the relationship between the NVR and the associated cut-off period, first in samples, then with its equivalence in years depending on the frequency sampling of the signal.

NVR	Period (time samples)	Years (quarterly series)	Years (monthly series)
10	2.86	0.71	0.24
1	6.01	1.51	0.51
0.1	11.02	2.76	0.92
0.01	19.78	4.95	1.65
0.001	35.28	8.82	2.94
1/1600	39.69	9.92	3.31
0.0001	62.81	15.71	5.23

Table 2.1 Relationship between the NVR parameter and the associated bandwidth of the IRW filter, i.e. the minimum period of cycles included in the filtered series.

Table 2.1 suggests that for a NVR value of 0.001, the smoothed signal would contain all the information of the original signal from a period of 35.28 samples up to infinity. All periods below that value are filtered out.

Put another way, if a signal has to be estimated such that it contains all the information of the original series for approximately 5 years and above, a NVR of 0.01 would be the option for a quarterly series, while 0.0001 should be chosen for a monthly series.

To illustrate these points, consider again the well-known air passenger series introduced in Chapter 1 (Box and Jenkins, 1970, 1976). Figure 2.1 illustrates these data, together with two possible trends obtained from different NVR values (0.1 and 0.0001). This graph may be obtained in CAPTAIN by entering the following MATLAB® code,

```
>> load air.dat
>> t1 = irwsm(air, 1, 0.1);
>> t2 = irwsm(air, 1, 0.0001);
>> t = (1949 : 1/12 : 1961-1/12)';
>> plot(t, [air t1 t2]);
```

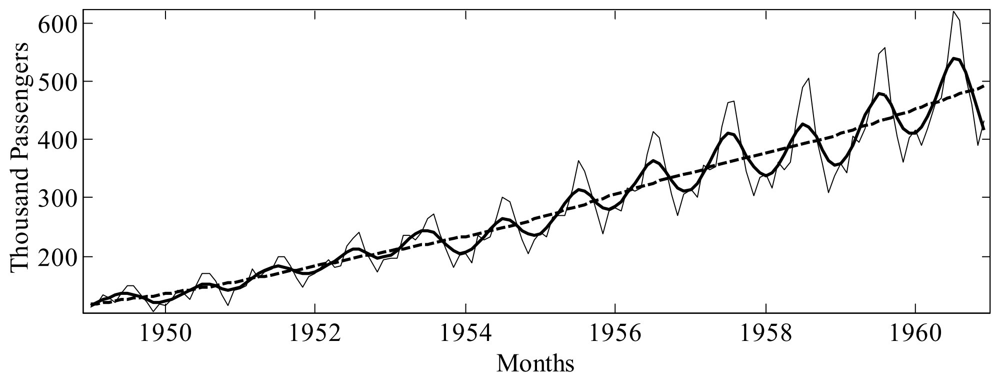


Figure 2.1 Air passenger data (thin trace) and IRW trends with NVR = 0.1 (thick solid) and NVR = 0.0001 (dashed).

Which one of these trends is the best? Although this is clearly a subjective matter, we may still say something about Figure 2.1, based on a general idea of what we mean by a trend. In particular, with the higher NVR of 0.1, the smoothed signal follows the data too closely to be regarded as a 'trend' in the normal sense, since it includes cyclic behaviour with a period of less than one year, i.e. it is really a combination of a trend and a seasonal component, something that in principle is undesirable. By contrast, with NVR = 0.0001, the smoothed signal does not appear to combine the trend and seasonal component in this manner, and so is more 'correct' in a signal decomposition sense.

In statistical terms, a more suitable approach would be to model the whole series with an Unobserved Components (UC) model, rather than attempt to extract the trend alone. Of course, this is also the main approach utilised in CAPTAIN and is described in Chapter 3. Nonetheless, CAPTAIN does offer the opportunity to just estimate a trend in this simple, exploratory context, using objective criteria for NVR optimisation, as discussed below.

2.3 (Hyper-) Parameter Estimation

The recursive KF and FIS algorithms above both require knowledge of all the system matrices $\mathbf{F}, \mathbf{G}, \mathbf{H}_t$ and $\mathbf{Q}_r$. In this regard, depending on the particular structure of the model chosen, there will be a number of elements either known prior to the analysis or fixed by the user. For example, if a RW model is specified, then clearly $\alpha = 1$ and $\beta = \gamma = 0$ in the SS model (2.2). However, in most cases, some unknown elements or *hyper-parameters* will remain unspecified and must be estimated separately. Typically, these include the NVR matrix $\mathbf{Q}_r$. In the following discussion, we will summarise all these hyper-parameters, whatever variables they may represent, in the vector Θ.

The hyper-parameter estimation problem is completely different to the state estimation problem and, in a certain sense, entirely independent of it. This fact is clearly seen in the range of methods available in the literature, some of which do not use the SS form at all! The most common methods include: Maximum Likelihood (ML) in the time domain (Schweppe 1965; Harvey, 1989); ML in the frequency domain based on a Fourier transform (Harvey, 1989; pages 191-204); alternative approaches in the frequency domain (Ng and Young, 1990; Young *et al.*, 1999); combinations of all the previous methods (Young and Pedregal, 1999); Bayesian approaches (West and Harrison, 1989); and estimation methods based on the *reduced* ARIMA form (Hillmer and Tiao, 1982; Hillmer *et al.*, 1983; Maravall and Gómez, 1998).

A selection of the most useful methods are implemented in CAPTAIN and listed below. Each method is associated with one of more different model-types in the toolbox. In some cases, several methods are available for the same type of model, with the one considered optimal as the default option (refer to the on-line help for each particular function).

- Time domain ML estimation for UC, TVP and State Dependent Parameter (SDP) models (Chapters 3, 4 and 5). This is the most widespread estimation method in the SS context, mainly because of its strong theoretical basis and because it is the most well known approach in many other areas of statistics. For this reason, it is discussed in detail below. See also Examples 2.2 and 2.4 below.

- Minimisation of the multiple-steps-ahead forecasting errors, also discussed below. This heuristic method is very useful when other methods do not provide a satisfactory answer to the problem, or when the objective of the research is strongly based on the forecasting performance of model. See also Examples 2.2 and 2.3 below.

- Frequency domain estimation, based on the spectral properties of the model (Young *et al.*, 1999). The parameters are estimated so that the logarithm of the model spectrum fits the logarithm of the empirical pseudo-spectrum (either an AR-spectrum or periodogram) in a least squares sense. A full description of this algorithm can be found in Chapter 3.

- Sequential Spectral Decomposition, reserved for a certain class of unobserved components models, i.e. Trend plus Auto-Regression (AR) also discussed in Chapter 3. This approach consists of decomposing the original series into quasi-orthogonal components, taking advantage of the exceptional spectral properties of the smoothing algorithms mentioned above. The overall non-linear problem is decomposed into several linear or quasi-linear steps, each solved in fully recursive terms. This yields a simple solution, with some loss of optimality from the ML viewpoint, but has proven to be very successful in practise. As a final step, filtering and smoothing are repeated using the whole SS formulation based on the analysis completed in the previous steps.

- Instrumental variable estimation of transfer function models in discrete and continuous time, as primarily implemented in the RIVSID module of CAPTAIN.

One interesting issue is that the complexity (or richness) of reality ensures that no single estimation method outperforms the rest in all possible situations - all of them have their own advantages and disadvantages. Therefore, the researcher's experience and knowledge is essential in selecting the best option for each application.

Because some of the methods mentioned above have been developed for specific models, their description is conveniently postponed to future chapters. In the present chapter, however, two general methods that are not intrinsically linked to particular models are reviewed, namely ML and the minimisation of the multiple-steps-ahead forecasting errors.

Maximum Likelihood

Assuming that all the disturbances in the SS form are normally distributed, the required Log-likelihood function can be computed using Kalman Filtering via 'prediction error decomposition' (Schweppe 1965; Harvey, 1989). The appropriate function for the general SS model in equation (2.1) is, therefore,

$$\log L(\Theta) = -\frac{T}{2}\log 2\pi - \frac{1}{2}\sum_{t=1}^{T}\log \hat{f}_t - \frac{1}{2}\sum_{t=1}^{T}\frac{\hat{v}_t^2}{\hat{f}_t} \tag{2.15}$$

where T is the number of observations, while $\hat{v}_t$ and $\hat{f}_t$ are the innovations and their variance respectively (see equation (2.11)), computed directly from the KF algorithm.

A number of issues must be taken into account when maximising (2.15); see e.g. Harvey, (1989) and Koopman *et al.* (2000). In the first place, the gradient and hessian necessary to find the optimum by numerical procedures can be evaluated either analytically or numerically, while the standard errors of the estimates may be found by means of the hessian, as is usual in the literature. Secondly, in the present context, it is usual to maximise the so called concentrated likelihood. This is because it is always possible to concentrate out one of the variances in σ^2 or $\mathbf{Q}$, reducing by one the number of parameters to estimate. In CAPTAIN, σ^2 is always the concentrated out variance, as shown by the definition of the diagonal NVR matrix $\mathbf{Q}_r$ introduced in the previous subsection.

Thirdly, in dynamic models, there always exists a problem of defining the initial conditions, i.e. the initial values of the state vector and its covariance matrix that are assumed unknown ($\hat{\mathbf{x}}_0$ and $\hat{\mathbf{P}}_0$, respectively). Once more, there are a number of solutions available. One is to define the *diffuse priors*, e.g. zero values for the initial state vector and big values for the diagonal elements of its covariance matrix, implying that there is little confidence in an arbitrary initialisation. In general, such an initialisation may affect the computation of the likelihood function, since the sum operators run over the whole sample.

An alternative solution is to start the summation in (2.15) from one observation past the length of the state vector (i.e. *n+1*), or from some other point where the KF algorithm has effectively converged. The effect of the initial conditions decreases as the length of the series increases, except for some very specific non-stationary models for which such effects never disappear (Casals *et al.*, 2000). Another, more theoretical, solution to the initial condition problem, is to incorporate the distribution of the initial conditions into the likelihood function itself, which yields the exact likelihood function, independent of such conditions and useful for short length time series (see e.g. De Jong, 1988, 1991; Casals *et al*., 2000). In CAPTAIN, the simplest diffuse priors option is chosen by default, although the user may always intervene and specify $\hat{\mathbf{x}}_0$ and $\hat{\mathbf{P}}_0$ directly.

Finally, a consideration common to all the estimation procedures in CAPTAIN, is that it is common to estimate the *scores* (certain transformations of the hyper-parameters) rather than the hyper-parameters themselves. Such parameters are then constrained to a certain domain in order to avoid nonsensical results. Two typical examples are that the NVR values should always be positive, while the α parameters in SRW models should always lie between zero and one inclusive. In this regard, the NVR and α scores used in CAPTAIN are listed in Table 2.2.

Score	Parameter	Score Range	Parameter Range
θ	$NVR = 10^{\theta}$	$(-\infty, \infty)$	$[0, \infty)$
ϑ	$\alpha = e^{\vartheta}/(1+e^{\vartheta})$	$(-\infty, \infty)$	$[0,1]$

Table 2.2 Hyper-parameters and scores in CAPTAIN.

In this way, the searching algorithm for ML looks for an unconstrained value of the scores from minus infinity to infinity, and it is transformed into a valid value of the corresponding hyper-parameters. The disadvantage when the scores are estimated rather than the parameter themselves is that the distribution of the scores is normal, according to theory, while the distribution of the parameters is, in general, not known. However, confidence intervals may always be reconstructed by applying the same transformation to the confidence interval of the scores.

Limitations of ML

There is no doubt that ML has theoretical and practical advantages. Its optimal properties are well-known and it is a widely applicable method for SS models. Indeed, the objective function (2.15) is applicable to any model written in SS form. The only requirement is that the user has to determine the appropriate form of the system matrices necessary to specify the model in SS terms. However, since CAPTAIN uses SS models as standard, this is not a problem in the present context.

Despite the advantages of ML on theoretical grounds, it does have some disadvantages, hence other methods are also implemented in CAPTAIN for specific types of model. In particular, in its normal form, ML is heavily dependent on the length of the series and the dimension of the model, since the recursive algorithms must be used to compute the Log-likelihood function at each iteration in the numerical optimisation. Furthermore, problems are sometimes encountered when the theoretical hypothesis on which the model is based does not hold in practice. An instructive example is the case of transfer function models when the input signal is a deterministic variable (such as steps, impulses, ramps, etc.). For this particular problem, however, the RIVSID module of CAPTAIN offers instrumental variable methods instead, which outperform ML and are never worse in other standard situations (Young, 1984).

Also, it is well-known (e.g. Young *et al*., 1999) that in certain UC models, the likelihood surface can be quite flat around its optimum, making the practical optimisation problem very inefficient at best and impossible in some cases. This is one important reason why these models usually have to be constrained when estimated by ML (we will come back to this issue in Chapter 3: see the examples therein). By contrast, for example, estimation methods in the frequency domain are free from these difficulties. In this regard, the approach implemented in CAPTAIN for UC models, optimises the hyper-parameters so that the logarithm of model spectrum fits the logarithm of the empirical spectrum in a least squares sense. This method has proven to allow for very much higher dimensional models than ML and computation times are greatly reduced, yielding solutions that are even better in likelihood terms than the constrained versions of the same model estimated by ML.

It should be clear from the discussion above that the solution achieved by the time and frequency domain methods are not necessarily equal, because each estimation method gives more emphasis to different aspects of the time series. Since the ML criterion is defined as an explicit, time domain function of the normalised one-step-ahead prediction errors (2.15), it would be expected that the hyper-parameters estimated in this way would provide good one-step-ahead forecasts. Indeed, provided all the theoretical assumptions that support ML methods are correct for a particular data set, it can be expected that the forecasts will be good for more than one-step-ahead as well. By contrast, estimation in the frequency domain is primarily concerned with ensuring that the spectral properties of the estimated components match the empirical spectrum of the data. In general, this tends to generate a solution which, in terms of the variance of the innovations, lies somewhere between that obtained by the ideal of unconstrained ML optimisation, which is difficult to consistently achieve in practice, and the constrained ML optimisation mentioned above.

Another situation where ML often does not always provide a sensible solution is when a smooth signal (or trend) is removed from a time series. In this case, ML tends to produce a signal that is too close to the original data, making the whole procedure invalid, as shown by Example 2.2 below. For such situations CAPTAIN, offers other estimation methods, like the minimisation of the multiple-steps-ahead forecasting errors discussed in the next subsection.

Given the pros and cons of each method, it is possible to improve overall model estimation by combining methods for different components of the model, i.e. to use the most appropriate method for each component. This is the approach suggested for SDP models for which, when necessary, frequency objective functions and ML are used together. SDP modelling is a very general procedure for the identification of non-linear relationships of many kinds. In this approach, an iterative procedure known as *back-fitting* is used. Here, parts of the model are estimated conditional on fixed values of the rest of parameters. In each iteration, the most appropriate estimation method can be used, either in the frequency domain or ML in the time domain.

Minimisation of the multiple-steps-ahead forecasting errors

The log-likelihood function (2.15) is dependent on the one-step-ahead forecasting errors, i.e. $\hat{v}_t = y_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-1}$, a natural outcome of the KF solution. However, the *multiple*-step-ahead forecasting errors can also be computed by simply repeating the KF prediction equations, say h times, without applying the correction equations. The associated forecasting errors are then as follows,

$$\hat{v}_t(h) = y_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-h} \tag{2.16}$$

This equation computes the forecasting errors at each t with all the information available up to t-h. Given these errors, one interesting option for hyper-parameter estimation is to minimise the sum of the squares of the h-step ahead errors, i.e.

$$\boldsymbol{J} = \sum_{t=n+h+1}^{\mathrm{T}} \hat{v}_t(h)^2 = \sum_{t=n+h+1}^{T} \left(y_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-h}\right)^2 \tag{2.17}$$

This approach is a natural heuristic extension to ML and may be applied when the latter does not provide a sensible solution to certain problems.

2.4 Worked examples

The following examples illustrate the basic functionality of hyper-parameter estimation in CAPTAIN, and show how the toolbox may be utilised for the preliminary analysis, smoothing and simple forecasting of non stationary time series.

Example 2.2 Hyper-parameter estimation for the air passenger series

Following directly from Example 2.1 above, consider again IRW smoothing of the air passenger series.

For some initial conditions, ML estimation does *not* yield useful results for this data set (Pedregal *et al*. 2007). This is because the theoretical properties assumed about the IRW model are not fulfilled by these particular data. In effect, the model (2.5) assumes that the perturbations about the trend are uncorrelated white noise, while it is clear here that the innovations are not at all white noise, indeed they are strongly periodic (see Figure 2.1). As a consequence, if the model is defined as an IRW model alone, then ML estimation can sometimes yield an unacceptable solution e.g. returning a very high NVR and a 'smoothed' signal that is almost identical to the original series.

However, **irwsmopt** uses an initial frequency domain estimation step, in order to obtain improved initial conditions for the subsequent ML estimation. For the present example, this approach yields satisfactory results, as indicated by the output shown below.

```
>> load air.dat
>> nvr = irwsmopt(air, 1, 'ml');

FREQUENCY DOMAIN INITIAL CONDITION ESTIMATION
METHOD: FREQUENCY DOMAIN. AR-SPECTRUM(24)
OPTIMISER: LSQNONLIN
0.016 seconds.
  PER.    RW        NVR         Score     S.E.     Alpha     Score     S.E.
   0.00   1.0    5.1196e-03   -2.2908   0.100    1.0000        -         -
Frequency Domain Objective Function: 2875.485

METHOD: MAXIMUM LIKELIHOOD
OPTIMISER: FMINSEARCH
Date: 24-Jun-2017 / Time: 21:19
0.219 seconds.
  PER.    RW        NVR        Score     S.E.     Alpha     Score     S.E.
   0.00   1.0    4.990e-06  -5.3019       -     1.0000        -         -
Likelihood: -750.466
```

Since **irwsmopt** is designed for straightforward smoothing applications, there are no user settings associated with the frequency domain initialisation step, and MATLAB® may sometimes return warnings about finding a local minimum. If required, **dhropt** may be used instead of **irwsmopt** to specify the initial conditions and/or adjust the model used for this initial spectral analysis (see later Chapter 3). However, for the present example, subsequent ML estimation to determine the final NVR proceeds without problem.

For information, Table 2.3 summarises the displayed tabular output from **irwsmopt**, while the header and footer above also confirm the optimisation method, the core MATLAB® function being utilised, the time taken to complete the optimisation and the Likelihood. Other hyper-parameter optimisation routines in CAPTAIN, including **dhropt**, **dlropt**, **daropt**, **darxopt**, **dtfopt** and **univopt** follow a similar convention. Note that the MATLAB® optimisation function (**fminsearch** in the example above) is automatically selected by CAPTAIN, depending on the optimisation method chosen and the presence other MATLAB® toolboxes installed on the users system, although this default can be overwritten (refer to the toolbox function help information).

Column Header	Meaning
PER	Period for periodic components (0 for trends).
RW	Type of Time Varying Parameter within the GRW model family (0-RW; 1-IRW).
NVR	Estimated NVR.
Score	Estimated score value from which the hyper-parameter is computed (only one NVR in the case above).
S.E.	Approximate standard error of the score.
Alpha	Estimated α parameter in SRW models.

Table 2.3 Tabular outputs from CAPTAIN hyper-parameter estimation functions.

Returning to the present example, an alternative to ML is to minimise the 12 steps ahead forecasting errors, as shown below. In general, the number of samples in a year, or the number of samples in any sort of cycle in the data, should be utilised in such optimisation.

```
>> nvr = irwsmopt(air, 1, 'f12');

METHOD: SUM OF SQUARES OF 12-STEPS-AHEAD FORECAST ERRORS
OPTIMISER: FMINSEARCH
0.672 seconds.
  PER.    RW       NVR        Score     S.E.     Alpha     Score     S.E.
   0.00   1.0    5.5777e-04  -3.2535      -     1.0000        -        -
Sum of Squares of 12-Steps-Ahead Forecast Errors: 268133.0759

>> tr = irwsm(air, 1, nvr);
>> t = (1949 : 1/12 : 1961-1/12)';
>> plot(t, [air tr])
```

For brevity, the frequency domain initialisation results are omitted from the output shown above (and in later examples). The NVR obtained in this case is 0.000558, with an associated cut-off period of 3.4 years (Table 2.1), returning the trend illustrated by Figure 2.2. Finally, it is straightforward to generate forecasts of the trend by adding **nan** values (MATLAB® Not-A-Number variables) to the end of the data, either by using standard commands or the CAPTAIN function **fcast**, as shown below.

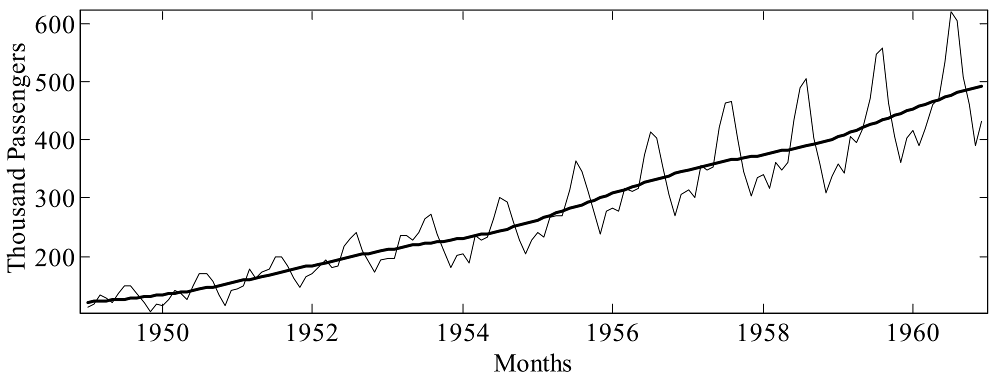


Figure 2.2 Air passenger data and trend chosen by minimising the 12-steps-ahead forecasting errors.

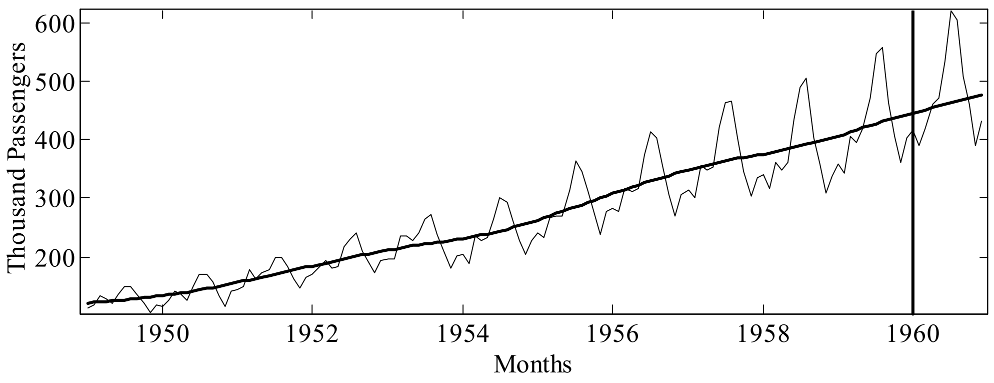


Figure 2.3 Air passenger data, trend and trend forecasts for the last year of data.

```
>> nvr = irwsmopt(air(1 : 132), 1, 'f12');
>> tr = irwsm(fcast(air, [133 144]), 1, nvr);
>> plot(t, [air tr]);
>> hold on
>> plot([t(132) t(132)], [0 650])
```

For illustrative purposes, the analysis above does not use the final year of the air passenger series, just the first 132 samples. Instead, the trend is forecasted for this year and compared with the original series, as illustrated in Figure 2.3, where the vertical line shows the forecasting horizon. It should be stressed that data to the right of the forecasting horizon are not used in the analysis, neither for estimating the NVR nor for smoothing the time series. Nonetheless, even with this simple IRW model, the forecasted trend is sensible, showing the long term behaviour of the series continuing in the correct direction.

Example 2.3 Interpolation and variance intervention for steel consumption in the UK

In order to illustrate the variance intervention and missing data handling capabilities of CAPTAIN, the quarterly steel consumption in the UK, measured in thousands of metric tons, is analysed from the last quarter of 1953 until the end of 1992. Variance intervention is useful when there are rapid or violent changes, or even discontinuities, in a series. Here, the estimate of $\hat{\mathbf{P}}_{t|t-1}$ in the prediction equation (2.8) is 'reset' to a large value, implying a lack of confidence in the estimates at that sample.

In order to plot the steel consumption with a correct time axis, as in Figure 2.4 below, enter the following MATLAB® commands:

```
>> load steel.dat
>> t = (1953.75 : 0.25 : 1992.75)';
>> plot(t, steel)
```

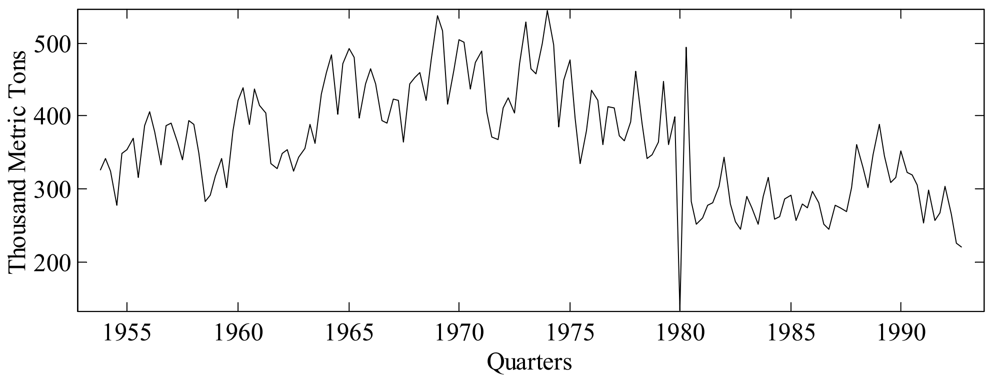


Figure 2.4 UK steel consumption from 1953Q4 to 1992Q4.

The anomalous large consumption figures in the first and second quarters of 1980 (referred to subsequently as 1980Q1 and 1980Q2) were due to strikes in the sector. Furthermore, two apparent falls in the mean level of steel consumption can be observed in 1975Q2 and 1980Q1. One way of handling these problems at the simple exploratory level is to smooth the series, assuming missing data (again, MATLAB® Not-A-Number values) for the strikes and setting variance intervention points at 1975Q2 and 1980Q1, as shown below.

```
>> y = fcast(steel, [106 107]);
>> nvr = irwsmopt(y, 1, 'f12', [87 106], [], 2);

METHOD: SUM OF SQUARES OF 12-STEPS-AHEAD FORECAST ERRORS
OPTIMISER: LSQNONLIN
0.516 seconds.
2 missing values
  PER.    RW       NVR       Score    S.E.    Alpha    Score    S.E.
   0.00   1.0    6.390e-05  -4.1945     -     1.0000      -       -
Sum of Squares of 12-Steps-Ahead Forecast Errors: 1756523.4294

>> tr =  irwsm(y, 1, nvr, [87 106]);
>> t = [1953.75:0.25:1992.75]';
>> subplot(211), plot(t, [y tr])
>> subplot(212), plot(t, y-tr)
```

Although not necessary for this example, the sixth input argument to **irwsmopt** illustrates how to change the MATLAB® optimisation function. In particular, the example above utilises **lsqnonlin**, which assumes that the MATLAB® Optimisation Toolbox is installed. The graphical output of this analysis is illustrated in Figure 2.5, where the jumps in the trend at the variance intervention points can be clearly seen and the perturbation about the trend (lower plot) appropriately shows that there are no remaining jumps in the series.

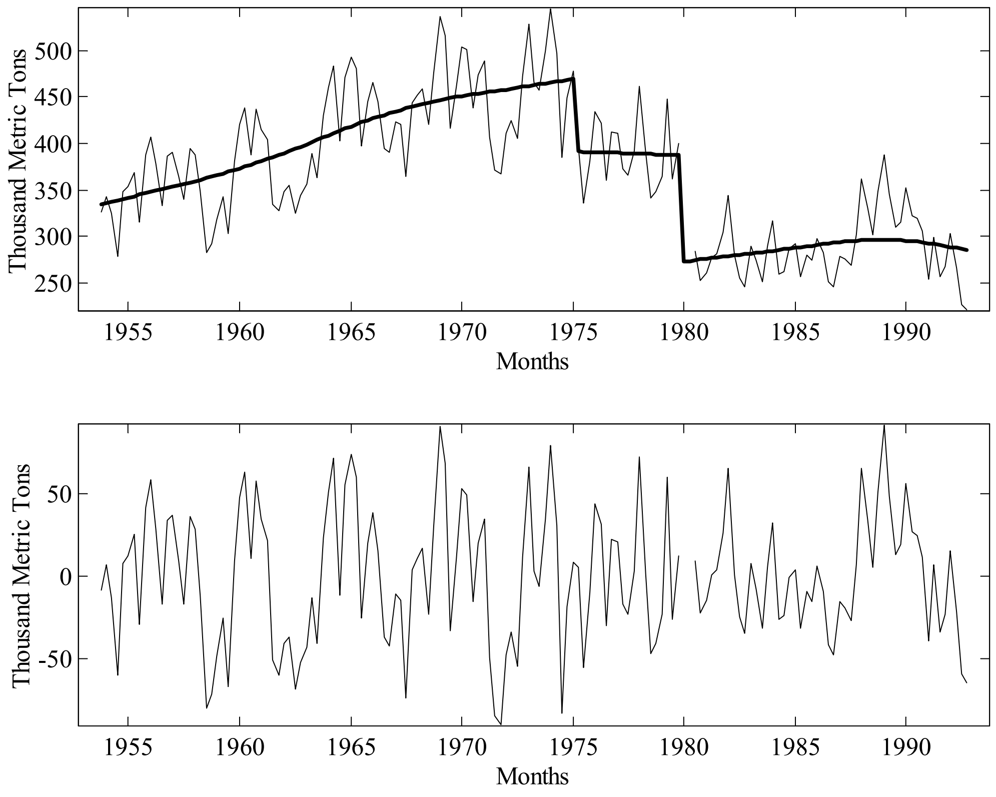


Figure 2.5 UK steel consumption, trend and perturbation about the trend.

Finally, it is sometimes interesting to reconstruct the data by assuming that the jumps in the trend did not occur, as shown below. Here, the CAPTAIN function **reconst** serves as a tool to remove the jumps, with the new estimate of the trend added to the perturbation signal. The value of such a calculation will be shown later in Chapter 3.

```
>> yr = y - tr + reconst(tr, [87 106]);
>> plot(t, yr)
```

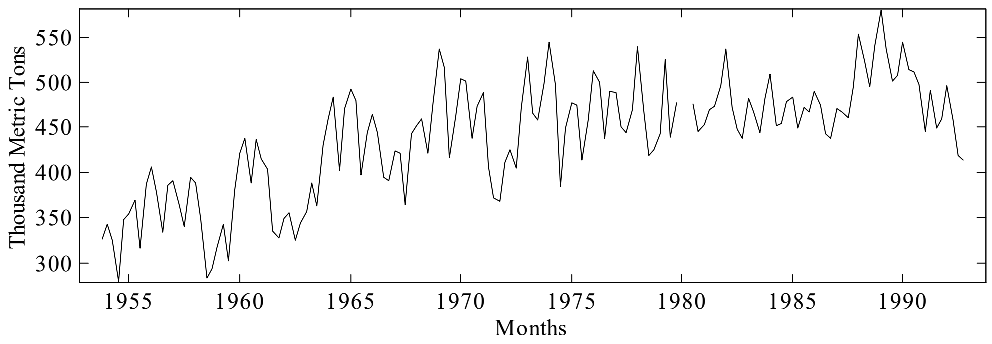


Figure 2.6 UK steel consumption with the jumps in the trend removed.

Before moving to the final example, it is useful to first introduce the Autocorrelation (ACF) and Partial Autocorrelation (PACF) Functions, together with the Ljung-Box test, since these will be utilised below and occasionally in later chapters.

Autocorrelation (ACF) and Partial Autocorrelation (PACF) functions

The ACF and PACF are two key identification tools, very useful for detecting time structure dependence in any time series. Popularised by Box and Jenkins (1970), they have been extensively used by time series analysers ever since. The aim is to determine the linear correlation coefficients between a time series and the lagged values of the same series. The representation of these coefficients against the lag, usually in the form of a bar diagram, is the ACF or Correlogram. More formally, the theoretical ACF of a stationary process y_t (i.e. constant mean and variance) is defined as,

$$\rho_k = \frac{\gamma_k}{\sigma_y^2} = \frac{\gamma_k}{\gamma_0} \qquad k = 0, 1, 2, \ldots \qquad (2.18)$$

where,

$$\gamma_k = E\left[(y_t - \mu_y)(y_{t+k} - \mu_y)\right] \qquad k = 0,1,2,\ldots \qquad (2.19)$$

Here, γ_k is the autocovariance function that measures the covariance between a time series and its past, while μ_y and σ_y^2 are the (constant) mean and variance of the process, respectively. The ACF is symmetrical around lag $k = 0$, so only positive values for the lag are considered.

Several estimators have been proposed in the literature, including the sample equivalents of the population counterparts, i.e.,

$$r_k = \frac{c_k}{s_y^2} = \frac{c_k}{c_0} \qquad k = 0, 1, 2, \ldots \qquad (2.20)$$

with,

$$c_k = \frac{1}{T-k} \sum_{t=1}^{T-k} (y_t - \bar{y})(y_{t+k} - \bar{y}) \quad k = 0,1,2,\ldots \tag{2.21}$$

where T is the number of samples and $\bar{y}$ is the sample mean. Note that, if the time series is white noise, an approximation of the variance of the autocorrelation estimators is,

$$Var[r_k] = \frac{1}{T} \qquad k = 0,1,2,\ldots \tag{2.22}$$

Equation (2.22) may be used to test the significance of any individual coefficient, i.e. any coefficients bigger than twice the standard error will be considered significantly different from zero.

The previous estimators are equivalent to the following set of linear regressions fitted to the data,

$$\left.\begin{aligned} y_t &= \alpha_1 + r_1 y_{t-1} + e_{1t} \\ y_t &= \alpha_2 + r_2 y_{t-2} + e_{2t} \\ &\vdots \\ y_t &= \alpha_k + r_k y_{t-k} + e_{kt} \end{aligned}\right\} \tag{2.23}$$

where α_i and e_{it} ($i = 1,2,\ldots,k$) are a set of constants and gaussian white noise terms, respectively. These equations are simply AR models of increasing orders, with all the intermediate parameters constrained to zero.

Apart from the individual test for each ACF parameter quoted above, a summary test of autocorrelation up to order m is the Ljung-Box test (Ljung and Box, 1978), given by the statistic,

$$Q = T(T+2) \sum_{k=1}^{m} \frac{r_k^2}{T-k} \tag{2.24}$$

This is distributed as a χ^2 with $m-1$ degrees of freedom under the null hypothesis of no autocorrelation.

The Partial Autocorrelation Function (PACF) extends the previous idea of the ACF using the standard statistical concept of partial correlation. This function measures the linear dependence between a time series and some lag of itself *discounting the effect of all the intermediate lags*. To understand this concept, and the difference with respect to the ACF, compare the following set of regressions with equation (2.23),

$$\left.\begin{array}{r} y_t = \alpha_1 + \phi_{11} y_{t-1} + e_{1t} \\ y_t = \alpha_2 + \phi_{21} y_{t-1} + \phi_{22} y_{t-2} + e_{2t} \\ \vdots \\ y_t = \alpha_k + \phi_{k1} y_{t-1} + \ldots + \phi_{kk} y_{t-k} + e_{kt} \end{array}\right\} \tag{2.25}$$

The PACF is a representation of the coefficients ϕ_{ii} $(i = 1,2,\ldots,k)$ against the lag. The variance of these coefficients are given by equation (2.22), or they may be computed as the standard error of the estimators in the previous set of regressions.

The ACF, PACF, their standard errors and the Q test are all computed by the CAPTAIN function **acf**. For example, Figure 2.7 is the output of the following code, where the ACF and PACF are estimated for a sequence of gaussian random numbers.

```
>> acf(randn(200, 1), 10);
  m      acf      desv       Q       Prob     m     pacf      desv
   1  -0.0199   0.0707   0.0808       -      1   -0.0199   0.0707
   2  -0.0306   0.0707   0.2713   0.6025    2   -0.0310   0.0707
   3  -0.0798   0.0708   1.5781   0.4543    3   -0.0812   0.0707
   4  -0.0251   0.0713   1.7076   0.6353    4   -0.0298   0.0707
   5  -0.0688   0.0713   2.6888   0.6112    5   -0.0759   0.0707
   6   0.0505   0.0716   3.2207   0.6660    6    0.0390   0.0707
   7   0.0481   0.0718   3.7060   0.7164    7    0.0415   0.0707
   8   0.0721   0.0720   4.7987   0.6845    8    0.0663   0.0707
   9  -0.0143   0.0723   4.8420   0.7743    9   -0.0041   0.0707
  10   0.0245   0.0723   4.9700   0.8369   10    0.0343   0.0707
```

The 2nd input argument to **acf** is the number of autocorrelation coefficients required by the user. As would be expected for white noise, Figure 2.7 shows that the series has no autocorrelation: the bars are all well within the standard errors (dotted trace). The same conclusion is strongly supported by the Q statistic and its probability value for any value of *m*, shown by the 4th and 5th columns above.

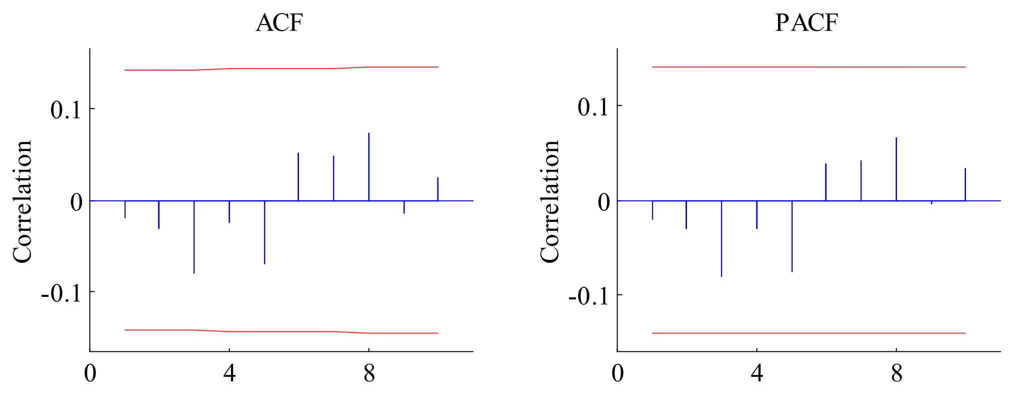


Figure 2.7 Autocorrelation and Partial Autocorrelation functions of a gaussian white noise signal.

Example 2.4 Time variable mean estimation for volume in the Nile river

Sometimes a debate arises among researchers about whether the mean level of a certain variable is changing over time or not. Numerous tests and procedures have been developed in order to investigate such a hypothesis. In this regard, if the stochastic behaviour of the signal about the time varying mean is approximately uncorrelated white noise, then there are some particularly simple and well formalised options. This is the case for the RW plus noise model introduced above. Here, the smoothed signal or trend is effectively a time varying mean variable that is assumed to evolve as a RW, while the rest of the signal is assumed white noise. Clearly, more complex options may be pursued by adding specific models to describe the perturbation about the trend if necessary, as discussed in Chapter 3.

Consider, for example, the Nile river annual volume measurements from 1871 to 1970 measured in 10^8 cubic meters, illustrated in Figure 2.8.

```
>> load nile.dat
>> t = (1871 : 1970)';
>> plot(t, nile)
```

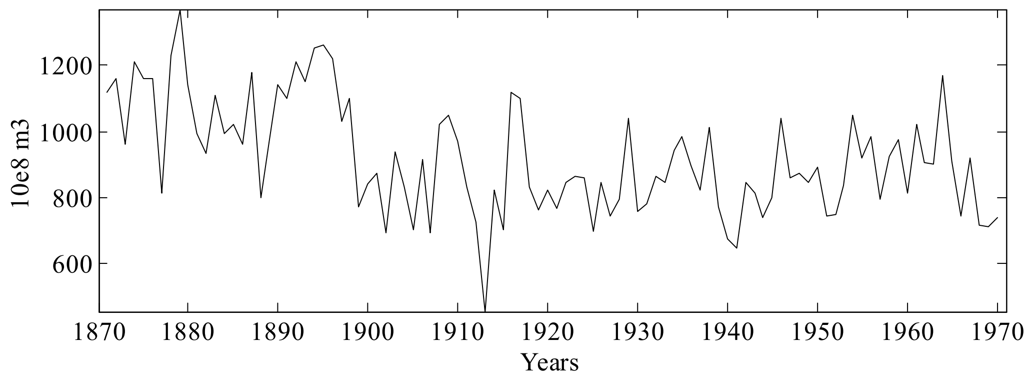


Figure 2.8 Annual volume of the Nile River in 10e8 cubic meters.

These data were analysed by Cobb (1978) and Balke (1993), among others. The key issue here is to determine whether there is a systematic decline in the level from 1899 onwards (sample 29), a feature that seems visually apparent from the figure. To investigate, the RW plus noise model is estimated as shown below, with the output illustrated in Figure 2.9. For brevity, detailed optimisation results from **irwsmopt** are omitted from the text below and the later examples (incidentally, the other hyper-parameter optimisation routines in the toolbox have an option to turn off tabular and graphical output if desired).

```
>> nvr = irwsmopt(nile, 0, 'ml')
nvr =
    0.0924
>> [tr, deriv, err] = irwsm(nile, 0, nvr);
>> plot(t, [nile tr tr+err tr-err])
```

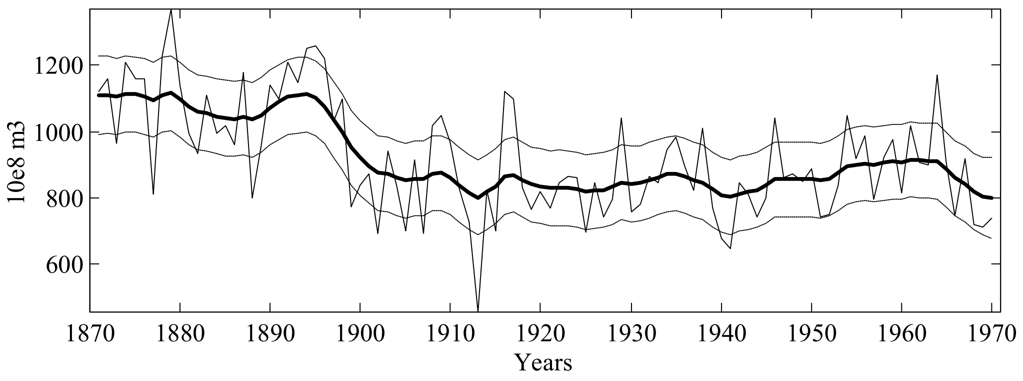


Figure 2.9 Annual volume of the Nile River, time variable mean and approximate 90% confidence intervals.

Note that the second input argument to both **irwsmopt** and **irwsm** is zero, in order to specify a RW trend, and that the NVR parameter is estimated by ML. In fact, this analysis takes advantage of the fact that, as it was seen in Example 2.2, ML will only provide a sensible solution if the theoretical assumptions about the model are fulfilled by the data. In this case, we can later check that the perturbations about the trend are indeed white noise.

It is clear from Figure 2.9 that the mean level has gone down since the beginning of the century. To test the adequacy of the model in a statistical sense, we examine the perturbations by means of the sample and partial autocorrelation functions (**acf**), together with a plot of the histogram superimposed over a Normal distribution (**histon**). The latter CAPTAIN function also returns a normality test, in the form of the Bera-Jarque statistic and associated probability value (Jarque and Bera, 1980).

```
>> acf(nile-tr, 20);
>> histon(nile-tr);
```

These graphs are illustrated in Figure 2.10. The Ljung-Box Q-test of autocorrelation for 20 lags is 17.7 indicating that there are no overall autocorrelation problems (Ljung and Box, 1978). Furthermore, the Bera-Jarque test indicates that the normality hypothesis cannot be rejected by a very wide margin. To sum up, both tests show that the theoretical assumptions about the model are fulfilled with no problem.

A second approach to the problem, which draws clearer light about the sharp decline in the level, is to use variance intervention again, directly specifying this 29th sample in the analysis, as shown below.

```
>> nvr = irwsmopt(nile, 0, 'ml', 29)
nvr =
  2.9035e-20
>> [tr, deriv, err] = irwsm(nile, 0, nvr, 29);
>> plot(t, [nile tr tr+err tr-err])
```

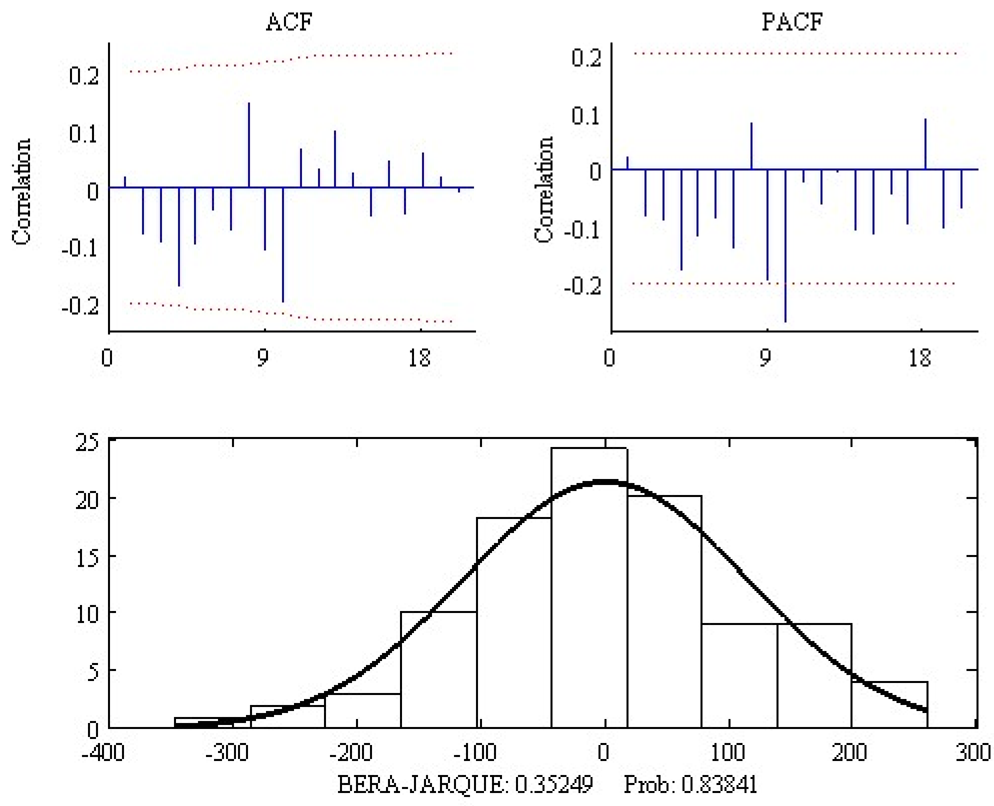


Figure 2.10 Analysis of the perturbations: Autocorrelation (ACF), Partial autocorrelation (PACF) and histogram of the residuals.

The estimated NVR is approximately zero, implying that the mean is constant apart from the break at the selected sample, as illustrated in Figure 2.11. Finally, although not shown here, **acf** and **histon** again indicate that the perturbations about the trend are white noise (the Q test for 20 lags is 14.35).

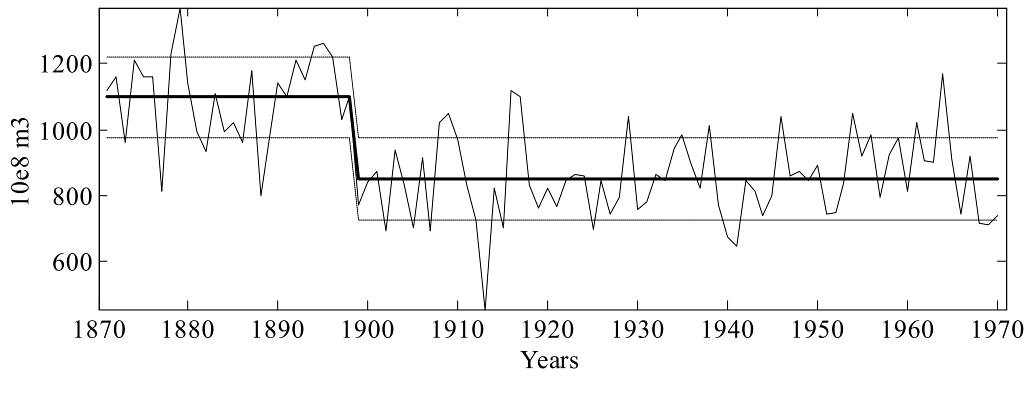


Figure 2.11 Annual volume of the Nile River, time variable mean and approximate 90% confidence intervals with variance intervention in 1899.

2.5 Conclusions

The present chapter has introduced the state space framework that is the basis for most of the models implemented in CAPTAIN and has formally described the associated filtering and smoothing algorithms at the heart of the toolbox. The chapter has also discussed a number of approaches for the estimation of any unknown elements or hyper-parameters in these models, concentrating on Maximum Likelihood and the minimisation of the multiple-steps-ahead forecasting errors.

However, to date, the models illustrated have been limited to the simplest Random Walk and Integrated Random Walk (IRW), plus measurement noise, cases. Whilst useful for basic smoothing operations, these models are largely concerned with the estimation of a simple trend. For additional components, such as seasonality and any other perturbations about the trend, we must turn to the more general, unobserved components model in the next chapter.

CHAPTER 3
UNOBSERVED COMPONENTS MODELS

Unobserved Components (UC) modelling is a general strategy for time series analysis and signal extraction, based on the assumption that the series is composed of an additive or multiplicative combination of different components that have defined statistical characteristics but which cannot be observed directly. In CAPTAIN, such components may include a trend, cyclical components, stochastic perturbations and so on. In the statistical literature, typical approaches to UC modelling include:

- *Ad-hoc* methods of seasonal adjustment in which smoothing procedures are used to extract trend and seasonal components from the time series. In this regard, one of the oldest and best known techniques for signal extraction is the Census X-11 method and its later extensions X-11 ARIMA and X-12 ARIMA. See e.g. Findley *et al*. (1996) and the references therein.
- The *ARIMA* or *Reduced Form* approach to UC model identification and estimation, based on the assumption that the series can be modelled as an Auto-Regressive-Integrated-Moving-Average (ARIMA) model. See e.g. Box *et al*. (1978); Maravall and Gómez (1998). Starting from this *reduced form* (ARIMA) model, the UC model (considered as a *structural form* following the Econometrics parallel) is obtained by the imposition of a number of (arbitrary) restrictions to ensure the existence and uniqueness of the decomposition.
- The *Optimal Regularisation* approach, based on direct optimal estimation of the components within a regularisation context. See e.g. Akaike (1980); Young and Pedregal (1996); Hodrick and Prescott (1997). In this case, constraints are imposed on the state estimates via a Lagrange Multiplier term within the cost function, in order to ensure that they possess the required characteristics.
- The *State Space* (SS) approach provides a rather more obvious formulation of UC concepts and, since this is the method implemented in CAPTAIN, is discussed in detail below. See also e.g. Ng and Young (1990); Young (1994); Young *et al*. (1999). Alternative SS approaches that have some points in common with CAPTAIN, as well as a few radically different aspects, are discussed by Harrison and Stevens (1976); Harvey (1989); and West and Harrison (1989).

It is clear from these examples that UC modelling may be regarded as a broad philosophy, an alternative to other more traditional ways of time series modelling, rather than as a particular model form and estimation method. However, the present authors believe that the SS approach is one of the most powerful and flexible frameworks for developing UC models. Indeed, the state estimation algorithms and associated methods for hyper-parameter optimisation, introduced in Chapter 2, provide a complete solution for the identification of UC models. All that remains is to characterise each component of the model in an appropriate SS form.

The previous chapter has already discussed one of the simplest cases, namely a trend component represented by an Integrated Random Walk (IRW) plus white noise model. Here, the CAPTAIN function **irwsmopt** is utilised to optimise the hyper-parameters, while **irwsm** provides the filtering, smoothing, forecasting and interpolation operations (see e.g. Example 2.4). Following a similar syntax, the **dhr**/**dhropt** and **univ**/**univopt** combinations in CAPTAIN provide for a more diverse range of UC models, as discussed below. Finally, the toolbox includes a number of functions to assist in the identification of these models, in both the time and frequency domains, namely: **aic**, **acf**, **arspec** and **period**.

3.1 General Form of the Unobserved Components Model

UC models in CAPTAIN can be synthesized by the following discrete-time equation,

$$y_t = T_t + C_t + S_t + f(\boldsymbol{u_t}) + N_t + e_t \qquad e_t \sim N(0, \sigma^2) \tag{3.1}$$

where y_t is the observed time series; T_t is a trend or low frequency component; C_t is a sustained cyclical or quasi-cyclical component (e.g. an economic cycle) with period different from that of any seasonality in the data; S_t is a seasonal component (e.g. annual seasonality); $f(\boldsymbol{u_t})$ captures the influence of a vector of exogenous variables $\boldsymbol{u}_t$, if necessary including stochastic, nonlinear static or dynamic relationships; N_t is a stochastic perturbation model, i.e. coloured noise modelled as an Auto-Regression (AR) process; and, as shown, e_t is an 'irregular' component, usually defined for analytical convenience as a normally distributed Gaussian sequence with zero mean value and variance σ^2 (i.e discrete-time white noise).

In the present context, equation (3.1) is regarded as the observation equation of a discrete time SS system, in which the dynamic behaviour of each of the UC's has to be defined via the state equations. In order to allow for nonstationarity in the time series y_t, the various components in (3.1), including the trend T_t, can all be characterised by stochastic, Time Variable Parameters (TVP's), with each TVP defined as a nonstationary stochastic variable, as discussed below.

One assumption that is maintained in every UC methodology is that all the components are orthogonal to the rest. In this context, it is noteworthy that the SS model representing equation (3.1) can be built by block-concatenation of all the matrices of each SS subsystem related to each of the components.

Despite the generality of equation (3.1), it should be stressed that in the majority of applications not all these components will be simultaneously necessary. Indeed, important identifiability problems may arise among the components if they are not defined appropriately. For example, a N_t AR component including seasonal roots will conflict severely with the seasonal component S_t if both are included in a single model. In a similar manner, unit roots in the N_t component would have problems with the trend component T_t. For these reasons, CAPTAIN normally restricts the user to formulations of the problem that are practically useful, so that such identification problems do not arise when using the toolbox (see examples below).

3.2 State Space form for UC Models

The SS model for each component in equation (3.1) is introduced below.

Trend models (T_t)

The trend models available in CAPTAIN are all particular cases of the General Random Walk (GRW) family of models represented by equation (2.2). These include, most commonly, the Random Walk (RW) and Integrated Random Walk (IRW) introduced in Chapter 2. A third option, the Local Linear Trend (LLT) model may be obtained by using RW and IRW models simultaneously, as shown below. The SS form of such a LLT model with observational noise is defined as follows,

$$\begin{pmatrix} x_{1t} \\ x_{2t} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_{1t-1} \\ x_{2t-1} \end{pmatrix} + \begin{pmatrix} \eta_{1t-1} \\ \eta_{2t-1} \end{pmatrix} \qquad T_t = \begin{pmatrix} 1 & 0 \end{pmatrix} \begin{pmatrix} x_{1t} \\ x_{2t} \end{pmatrix} + e_t \tag{3.2}$$

The state equations may be written using the backward-shift operator as,

$$\begin{aligned} (1-L)x_{1t} &= x_{2t-1} + \eta_{1t-1} \\ (1-L)x_{2t} &= \eta_{2t-1} \end{aligned} \tag{3.3}$$

Then, substituting the second equation in the first one we have,

$$(1-L)x_{1t} = \frac{\eta_{2t-2}}{(1-L)} + \eta_{1t-1} \tag{3.4}$$

Finally, substitution of this equation in the observation equation gives the reduced form,

$$y_t = \frac{\eta_{2t-2}}{(1-L)^2} + \frac{\eta_{1t-1}}{(1-L)} + e_t \tag{3.5}$$

i.e. the addition of a RW and IRW model if the state noises are independent of each other (an assumption that is usual in this context).

Two further trend models, available only in the CAPTAIN functions **univ** and **univopt** for AR + Trend analysis, are the so-called Integrated AR (IAR) or Double Integrated AR (DIAR) models. The motivation for these options arises from the observation that, for the simplest trend models, it is quite common to find correlation in the residuals. For example, when utilising an IRW model for the trend, it is assumed that its second difference will be white noise, which is not always the case in practice. IAR and DIAR models take advantage of the correlation, by building an addition model for these residuals, in order to improve the overall forecasting performance.

One caveat, however, is that it is possible for the FIS algorithm itself to induce this kind of correlation. Indeed, it can be shown (Young and Pedregal, 1996) that the correlation structure depends on the autocorrelation of the original time series itself. Nonetheless, if the second difference does show a predictable behaviour, especially if there is some physical meaning (e.g. related to the business cycle), then it would be worthwhile to try to forecast it. In this case, the DIAR model, which is defined below in a manner similar to the earlier examples, provides one particular approach,

$$\begin{aligned} T_t &= T_{t-1} + D_{t-1} \\ D_t &= D_{t-1} + x_{1t} \\ x_{1t} &= \frac{\eta_{3t}}{1+\phi_1 L + \phi_2 L^2 + \ldots + \phi_p L^p} \end{aligned} \tag{3.6}$$

In SS form, this model may be described by the following equations,

$$\left[\begin{array}{c} T \\ D \\ \hline x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_p \end{array}\right]_t = \left[\begin{array}{cc|cccccc} 1 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ \hline 0 & 0 & -\phi_1 & -\phi_2 & -\phi_3 & \cdots & -\phi_{p-1} & -\phi_p \\ 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \cdots & 1 & 0 \end{array}\right] \left[\begin{array}{c} T \\ D \\ \hline x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_p \end{array}\right]_{t-1} + \left[\begin{array}{c} 0 \\ 0 \\ \hline \eta_3 \\ 0 \\ 0 \\ \vdots \\ 0 \end{array}\right]_t \tag{3.7}$$

$$T_t = [1 \quad 0 \quad \cdots \quad 0]\boldsymbol{x}_t \quad \text{with} \quad \boldsymbol{x}_t = [T \quad D \mid x_{1t} \quad x_{2t} \quad \cdots \quad x_{pt}]^T$$

The model is then fully defined by the variance of η_{3t} and the coefficients of the AR polynomial. Although this is a rather more complex model than the GRW, it does have the capability of providing non-linear like forecasts of the trend, very useful in situations near turning points.

Cyclical and seasonal models (C_t and S_t)

Although these two types of components are given different names in equation (3.1), both can be treated in the same way from a modelling standpoint since both reflect a periodic kind of behaviour. The difference between them lies only on the period considered, with 'seasonal' usually reserved for an annual cycle. CAPTAIN provides two approaches for modelling any such periodic behaviour, as discussed below.

Dynamic Harmonic Regression (DHR; Ng and Young, 1990; Young *et al.*, 1999)

The DHR model is similar to a Fourier analysis, but with coefficients that evolve smoothly in time. The model is,

$$y_t = S_t + e_t = \sum_{j=0}^{\left[\frac{s}{2}\right]} \left\{ a_{jt} \cos(\omega_j t) + b_{jt} \sin(\omega_j t) \right\} + e_t \tag{3.8}$$

with,

$$\omega_j = \frac{2\pi j}{s} \qquad j = 1,\ 2,\ \ldots,\ \left[\frac{s}{2}\right] \tag{3.9}$$

and,

$$\begin{pmatrix} a_{jt} \\ a'_{j1t} \end{pmatrix} = \begin{pmatrix} \alpha_j & \beta_j \\ 0 & \gamma_j \end{pmatrix} \begin{pmatrix} a_{jt-1} \\ a'_{jt-1} \end{pmatrix} + \begin{pmatrix} \eta_{jt} \\ \eta'_{jt} \end{pmatrix} \qquad \text{NVR}(\eta_{jt}) = \text{NVR}(\eta'_{jt}) \tag{3.10}$$

Here, $[s/2] = s/2$ for even s and $[s/2] = (s-1)/2$ for odd s. Parameters a_{jt} and b_{jt} are represented as GRW models with the associated NVR values both equal for the same harmonic period. Note that setting $\omega_0 = 0$ reduces the correspondent term for $j = 0$ to a matrix of ones and zeros, implying that GRW trends are naturally accommodated within a DHR context.

As an illustration of the SS form of this model, consider the following IRW trend with a single harmonic modulated by RW parameters for a period of 12 samples, typical of monthly time series,

$$\begin{pmatrix} T_t \\ D_t \\ a_{1t} \\ b_{1t} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} T_{t-1} \\ D_{t-1} \\ a_{1t-1} \\ b_{1t-1} \end{pmatrix} + \begin{pmatrix} 0 \\ \eta_{Tt-1} \\ \eta_{a1t} \\ \eta_{b1t} \end{pmatrix}$$

$$y_t = \begin{bmatrix} 1 & 0 & \cos(2\pi t) & \sin(2\pi t) \end{bmatrix} \boldsymbol{x}_t + e_t$$

Compared with other methodologies for modelling periodic components, the framework above is one of the most flexible. In some other approaches (e.g. Harvey, 1989; West and Harrison, 1989) all the variances in the harmonics must be the same. Furthermore, in many standard approaches to the problem, all the harmonics of the seasonal component are introduced into every model. By contrast, when using CAPTAIN, the modeller is strongly recommended to look at the spectral properties of the series in order to identify the most appropriate model for the time series in question, i.e. to check whether all the harmonics are actually necessary. For this latter purpose, CAPTAIN includes the functions **period** and **arspec** to estimate the power spectrum and an AR-spectrum respectively. The examples considered in section 3.4 will demonstrate the importance of this identification stage.

The unknowns in the model (3.8)-(3.10), including the noise variances are the hyper-parameters. Given an estimation of these hyper-parameters, the KF and FIS algorithms yield estimates of each TVP and hence the trend and harmonic component, together with the total cyclical component, i.e. the sum of all the individual harmonics.

Finally, note that the DHR model may be regard as a particular example of Dynamic Linear Regression (see Chapter 4), in which the inputs are deterministic sinusoidal functions of time. Note that all the inputs in this model are orthogonal, a property that is highly desirable in regression methods and makes the TVP estimation problem particularly straightforward, even when a high number of parameters are involved.

Trigonometric Cycle or Seasonal (Harvey 1989; West and Harrison, 1989):

Here, the periodic components are introduced via the state equations of the SS model, rather than in the observation equation as for the DHR model. The full model is,

$$y_t = S_t + e_t = \sum_{j=1}^{\left[\frac{s}{2}\right]} S_{jt} + e_t \tag{3.11}$$

with each S_{jt} defined as follows,

$$\begin{pmatrix} S_{jt} \\ S'_{jt} \end{pmatrix} = \rho_j \begin{pmatrix} \cos\omega_j & \sin\omega_j \\ -\sin\omega_j & \cos\omega_j \end{pmatrix} \begin{pmatrix} S_{jt-1} \\ S'_{jt-1} \end{pmatrix} + \begin{pmatrix} \eta_{jt} \\ \eta'_{jt} \end{pmatrix} \qquad NVR(\eta_{jt}) = NVR(\eta'_{jt}) \quad (3.12)$$

The same noise definitions utilised in DHR models apply here. The ρ_j parameter is introduced to allow convergence in the cyclical components when it is constrained between 0 and 1. In seasonal models it is usually fixed at 1.

The SS form of this model is straightforward. For example, with a trend and a single harmonic component, the overall SS form would be,

$$\begin{pmatrix} T_t \\ D_t \\ S_{1t} \\ S'_{1t} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\omega_{12} & \sin\omega_{12} \\ 0 & 0 & -\sin\omega_{12} & \cos\omega_{12} \end{pmatrix} \begin{pmatrix} T_{t-1} \\ D_{t-1} \\ S_{1t-1} \\ S'_{1t-1} \end{pmatrix} + \begin{pmatrix} 0 \\ \eta_{Tt-1} \\ \eta_{1t} \\ \eta'_{1t} \end{pmatrix}$$

$$y_t = \begin{bmatrix} 1 & 0 & 1 & 0 \end{bmatrix} \boldsymbol{x}_t + e_t \qquad (3.13)$$

Such a seasonal component, together with a LLT model for the trend, is called a Basic Structural Model (BSM) by Harvey (1989), although recent publications by the same author refer instead to unobserved components rather than structural models.

Exogenous variables ($f(\boldsymbol{u}_t)$)

CAPTAIN provides numerous approaches for modelling the input-output relationship between variables. Such methods are discussed at length in future chapters, so will only be briefly listed here:

- Dynamic Linear Regression (DLR; Chapter 4), in which one output is related to several inputs in a linear regression form, but with TVP's.
- Dynamic Autorregression with eXogenous inputs (DARX; Chapter 4). This is an extension of the DLR model, in which past values of the output are also regressors of the system. Again, all the parameters affecting the exogenous variables and the past values of the output (endogenous variables) may be TVPs.
- Dynamic Transfer Function model (DTF, Chapter 4), effectively an extension of the DARX model but with a more widely applicable assumption for the noise.
- Standard Transfer Function (TF) models in discrete and continuous time with constant parameters. These models are estimated by recursive, Refined Instrumental Variable (RIV) algorithms within the RIVSID module of CAPTAIN.

- State Dependent Parameter analysis (SDP, Chapter 5), a general approach for the identification and estimation of non-linear, non-stationary dynamic relationships between variables.

Coloured noise components (N_t)

Several models implemented in CAPTAIN allow coloured noise components to handled as pure AR processes. In order to consider the SS form, it is convenient to assume that the sum of the coloured noise and the white noise component in equation (3.1) constitutes the AR process with the same white noise input (Young, 1984; Ng and Young, 1990), i.e.,

$$N_t + e_t = \frac{1}{\phi(L)} e_t \tag{3.14}$$

where $\phi(B) = 1 + \phi_1 L + \phi_2 L^2 + \ldots + \phi_p L^p$. One SS form of such a model is, therefore,

$$\begin{aligned} \begin{pmatrix} x_{1t} \\ x_{2t} \\ \vdots \\ x_{p-1t} \\ x_{pt} \end{pmatrix} &= \begin{pmatrix} -\phi_1 & 1 & 0 & \cdots & 0 \\ -\phi_2 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\phi_{p-1} & 0 & 0 & \cdots & 1 \\ -\phi_p & 0 & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} x_{1t-1} \\ x_{2t-1} \\ \vdots \\ x_{p-1t-1} \\ x_{pt-1} \end{pmatrix} + \begin{pmatrix} \eta_t \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \\ y_t &= \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix} \boldsymbol{x}_t \end{aligned} \tag{3.15}$$

Models with polynomials of different orders can be implemented by constraining the corresponding parameters to zero. In the same way, multiplicative polynomials typical of seasonal ARMA models, in the manner of Box-Jenkins, can be converted into this equivalent form by convoluting the polynomials and transforming the model into a higher order system.

A trend component straightforwardly attached to this model by simple block-concatenation of the appropriate SS matrices, in a similar manner to the earlier examples. This new model may be used as an alternative to DHR and BSM if the AR process order is high enough to allow for seasonal roots. It is also of interest in more general situations in which the signal has some none seasonal correlation about the trend.

3.3 (Hyper-) parameter estimation for UC models

The recursive state estimation algorithms require prior knowledge of all the system matrices in the state space models above. However, depending on the particular structure of the model chosen, some hyper-parameters will remain unspecified and must be estimated separately before state estimation can proceed. Typically, these include the NVR matrix. Section 2.3 of Chapter 2 introduced the hyper-parameter estimation problem for general SS models. Since most of the models considered in the present chapter are set up in such a SS form, all the issues previously raised apply here. In particular, Maximum Likelihood (ML) and the minimization of the multiple-steps-ahead forecasting errors are available options for all the UC models in CAPTAIN. Two additional approaches, developed specially for UC models, are considered below, namely Frequency Domain estimation and Sequential Spectral Decomposition.

Frequency domain estimation

This method has been developed for DHR and BSM models, implemented in the CAPTAIN functional pair **dhr/dhropt**. Frequency domain estimation methods are generally concerned with approximating the theoretical pseudo-spectrum of the model (a function of the hyper-parameters in question) to the empirical pseudo-spectrum obtained directly from the time series.

Given the DHR model (3.8) or the BSM (3.11), there are two logical steps when building the model spectrum: (a) derivation of the spectrum of the TVP models taken from a GRW process and (b) derivation of the spectrum of the sinusoidal components modulated by those TVP's. These are considered in turn below, followed by (c) the full algorithm.

(A) Pseudo-Spectra of GRW models

In order to derive the spectrum of GRW models, it is necessary to first obtain the reduced form (or transfer function) of its SS description. For example, in the case of a SRW model (i.e. equation (2.2) with $0 < \alpha < 1$; $\beta = \gamma = 1$; $\eta_{1t} = 0$) the reduced form is given by,

$$y_t = \frac{\eta_{2t-1}}{(1-L)(1-\alpha L)} + e_t \tag{3.16}$$

where $\nabla = (1-L)$ is the difference operator. The stationary version of the model is then,

$$\nabla y_t = \frac{\eta_{2t-1}}{(1-\alpha L)} + \nabla e_t \tag{3.17}$$

For this process, the spectrum can be calculated by recalling that the frequency response (e.g. Priestley, 1989) of a signal z_t,

$$z_t = \frac{\eta_t}{(1-\alpha L)} \tag{3.18}$$

is,

$$f_z(\omega)=\frac{1}{2\pi}\frac{\sigma_\eta^2}{\left|\left(1-\alpha e^{i\omega}\right)\right|}=\frac{1}{2\pi}\frac{\sigma_\eta^2}{\left(1-\alpha e^{i\omega}\right)\left(1-\alpha e^{-i\omega}\right)}=\frac{1}{2\pi}\frac{\sigma_\eta^2}{\left\{1+\alpha^2-2\cos(\omega)\right\}} \tag{3.19}$$

In this case, the spectrum for ∇y_t takes the following form,

$$f_{\nabla y}(\omega)=\frac{1}{2\pi}\left[\frac{\sigma_{\eta_2}^2}{\left\{1+\alpha^2-2\alpha\cos(\omega)\right\}}+\left\{2-2\cos(\omega)\right\}\sigma^2\right];\qquad \omega\in\left[0,\ \pi\right] \tag{3.20}$$

whilst for the non-stationary SRW process y_t, the pseudo-spectrum is,

$$f_y(\omega)=\frac{1}{2\pi}\left[\frac{\sigma_{\eta_2}^2}{\left\{1+\alpha^2-2\alpha\cos(\omega)\right\}\left\{2-2\cos(\omega)\right\}}+\sigma^2\right];\qquad \omega\in\left[0,\ \pi\right]. \tag{3.21}$$

This is a function in ω with a peak at ω_j. Spectra for particular cases can be found by just constraining the α parameter (i.e. 0 or 1 for RW or IRW models, respectively). However, for the other models considered so far, additional manipulations are necessary. For example, in the case of a Trigonometric Cycle, the reduced form is,

$$y_t=\frac{1+aL}{1+bL+cL^2}\eta_{jt}+e_t \tag{3.22}$$

with $a=\rho\left\{\sin\left(\omega_j\right)-\cos\left(\omega_j\right)\right\}$, $b=-2\rho\cos\left(\omega_j\right)$ and $c=\rho^2$. Therefore, its pseudo-spectrum is,

$$f_y(\omega)=\frac{1}{2\pi}\left\{\frac{1+a^2+2a\cos(\omega)}{1+b^2+c^2+2b(1+c)\cos(\omega)+2c\cos(2\omega)}\sigma_{\eta_j}^2+\sigma^2\right\} \tag{3.23}$$

This is also a function in ω with a peak at ω_j.

(B) <u>Pseudo-Spectra of DHR terms modulated by GRW parameters</u>

From the basic Fourier transform properties, the frequency response of amplitude modulated signals of the form $S_t=a_t\cos\left(\omega_j t\right)$ is known to be,

$$f_S(\omega)=\frac{1}{2}\left[f_A\left(\omega-\omega_j\right)+f_A\left(\omega+\omega_j\right)\right] \tag{3.24}$$

where $f_A(\omega)$ is the frequency response of a_t. Consider now the case of a single DHR component of the form $S_t=a_{jt}\cos\left(\omega_j t\right)+b_{jt}\sin\left(\omega_j t\right)$, in which the parameter variations are governed by SRW models. The pseudo-spectrum is given by,

$$f_S(\omega,\omega_j)=\frac{1}{2\pi}\left[\frac{\sigma_{\eta_j}^2}{\{1+\alpha^2-2\alpha\cos(\omega-\omega_j)\}\{2-2\cos(\omega-\omega_j)\}}+\right.$$
$$\left.+\frac{\sigma_{\eta_j}^2}{\{1+\alpha^2-2\alpha\cos(\omega+\omega_j)\}\{2-2\cos(\omega+\omega_j)\}}\right]$$

This is a function of ω with a maximum at ω_j in a way such that the height and width depend on the variances of the noises and the value of α. Once more, RW or IRW models may be found by constraining α to 0 or 1 respectively, while the LLT may be obtained by adding RW and IRW models together. Defining,

$$S(\omega,\omega_j)=\frac{1}{2\pi}\left[\frac{1}{\{1+\alpha^2-2\alpha\cos(\omega-\omega_j)\}\{2-2\cos(\omega-\omega_j)\}}+\right.$$
$$\left.+\frac{1}{\{1+\alpha^2-2\alpha\cos(\omega+\omega_j)\}\{2-2\cos(\omega+\omega_j)\}}\right]$$

then the pseudo-spectrum of the full DHR model (3.8) becomes,

$$f_y^*(\omega,\underline{\sigma}^2)=\sum_{j=0}^{\left[\frac{s}{2}\right]}\sigma_{\omega_j}^2 S(\omega,\omega_j)+\frac{\sigma^2}{2\pi} \qquad \underline{\sigma}^2=\begin{bmatrix}\sigma^2 & \sigma_{\omega_0}^2 & \sigma_{\omega_1}^2 & \dots & \sigma_{\omega\left[\frac{s}{2}\right]}^2\end{bmatrix}$$

This latter expression can also be described in terms of the hyper- parameters $\mathrm{NVR}_j=\sigma_{\omega_j}^2/\sigma^2$, i.e.,

$$f_y^*(\omega,\underline{\boldsymbol{NVR}})=\sigma^2\left\{\sum_{j=0}^{\left[\frac{s}{2}\right]}\mathrm{NVR}_j S(\omega,\omega_j)+1\right\} \tag{3.25}$$

Here, the full set of NVR parameters is represented by the vector $\underline{\boldsymbol{NVR}}$. Note also that (3.25) is linear in the NVR's and, as we shall see, this facilitates the initial estimation of these hyper-parameters. The extension of $S(\omega,\omega_j)$ to accommodate more complex combinations of RW, IRW and LLT defined trends and parameters is obvious.

(C) Full estimation algorithm in the frequency domain

The estimation problem is to find the set of parameters $\underline{\boldsymbol{NVR}}$ (and any other hyper-parameters present in the model, such as the α parameters in the SRW model) which yield the optimal least squares fit to the empirically estimated spectrum. Although a linear least-squares fit is the most obvious, Young *et al.* (1999) show that substantial advantages can be found when the following non-linear objective function is used instead,

$$\Im\left(f_y,\hat{f}_y^*\right)=\sum_{k=0}^{T-1}\left[\log\left\{f_y(\omega_k)\right\}-\log\left\{\hat{f}_y^*(\omega_k,\underline{\boldsymbol{NVR}})\right\}\right]^2 \tag{3.26}$$

Here, $f_y(\omega_k)$ is either the sample periodogram or the AR spectrum of the time series, with the AR order in the latter case identified by the Akaike Information Criterion (AIC; Akaike, 1974). Using the log transformed spectra yields much better defined NVR estimates since it concentrates attention on the most important shape of the 'shoulders' associated with the harmonic peaks in the AR Spectrum. The linear solution is used as an initialisation for this non-linear optimisation, which is computationally very fast.

The complete estimation algorithm in the frequency domain, thus consists of the following four steps:

- Estimate an AR(n) spectrum $f_y(\omega)$ of the observation process $y_t, t=1,2,\ldots,N$ and its associated residual variance $\hat{\sigma}^2$, with the AR order n normally identified by reference to the AIC. Note the $[s/2]$ significant peaks that characterise the spectrum (these will normally include a fundamental frequency and several or all of its associated harmonics).

- Find the Linear Least Squares estimate of the NVR parameter vector which minimises the following linear least squares objective function,

$$\Im^*\left(f_y,\hat{f}_y^*\right)=\sum_{k=0}^{T-1}\left[f_y(\omega_k)-\hat{f}_y^*(\omega_k,\underline{\boldsymbol{NVR}})\right]^2 \tag{3.27}$$

- Find the Nonlinear Least Squares estimate of the NVR parameter vector which minimises the following nonlinear least squares objective function,

$$\Im\left(f_y,\hat{f}_y^*\right)=\sum_{k=0}^{T-1}\left[\log\left\{f_y(\omega_k)\right\}-\log\left\{\hat{f}_y^*(\omega_k,\underline{\boldsymbol{NVR}})\right\}\right]^2 \tag{3.28}$$

  using the result from the second step to define the initial conditions.

- Use the NVR estimates from the third step to obtain the recursive forward pass (KF) and backward pass (FIS algorithm) smoothed estimates of the components in the DHR model: i.e. the trend; the total cyclical and seasonal components; the fundamental/harmonic components; and the residuals. This last step should allow for any interventions and outliers, interpolate over gaps and forecast as necessary in the normal manner.

Example 3.1 demonstrates the straightforward implementation of these four steps using the AR spectrum and DHR model estimation functions in CAPTAIN.

Example 3.1 Analysis of the Mauna Loa CO_2 data using DHR and related models

The measured CO_2 concentration at Mauna Loa, illustrated in Figure 3.1, clearly have a seasonal component, which is related to the global net uptake and release of CO_2 in the biosphere in the summer and winter.

```
>> load co2.dat
>> plot(co2)
```

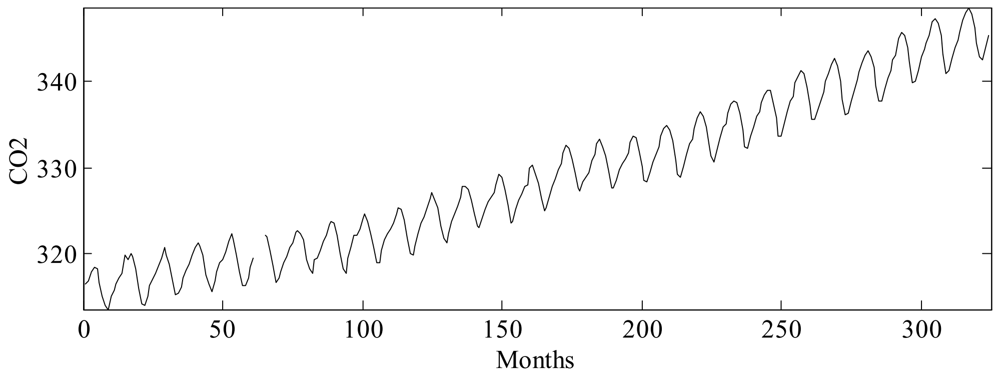


Figure 3.1 CO_2 concentration at Mauna Loa (parts-per-million).

The data are sampled monthly (i.e. $s=12$), hence the expected periodic components are 12, 6, 4, 3, 2.4 and 2 samples per cycle (i.e. $12/j$, $j=1,2,\ldots,6$). However, rather than rely on these theoretical harmonics, the first step in the analysis is normally to identify the most significant harmonics in the series by means of some spectral measurement. CAPTAIN offers two possibilities: the AR pseudo-spectrum (**arspec**) and the periodogram (**period**). For example, the following straightforward command yields Figure 3.2.

```
>> arspec(co2);
```

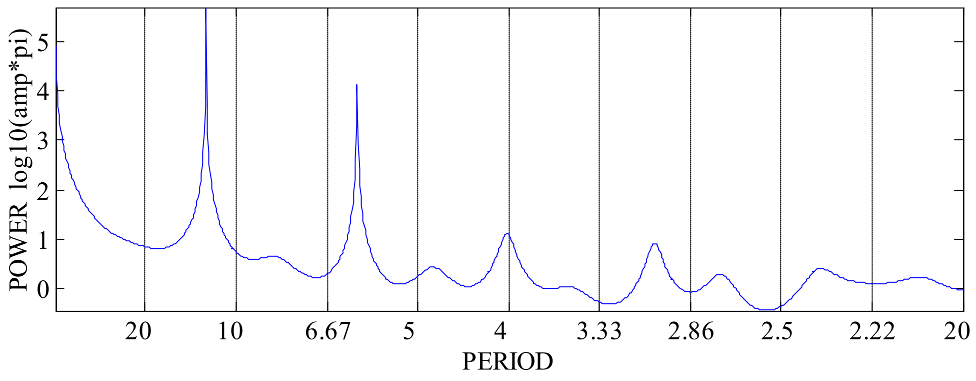


Figure 3.2 AR(27) pseudo-spectrum of the CO_2 data.

The AR pseudo-spectrum Figure 3.2 is determined on the basis of an AR(27) model, identified automatically by the default **arspec** call (see Example 1.2). Additional arguments are possible, allowing for the direct specification of the AR model order from any prior analysis (see help information for **arspec**). It is clear from the left hand side of Figure 3.2 that a trend is present while, most interestingly, the seasonal component is dominated by just the two first harmonics, i.e. the 12 and 6 samples per cycle (s/c) period components. In fact, since the harmonic corresponding to the period 2.4 s/c is very small and the 2 s/c harmonic is not present at all, these are ignored in the DHR analysis below.

The NVR hyper-parameters are first estimated in the (default) frequency domain, using an IRW model for the trend and RW models for the four dominant harmonics of the seasonal component (12, 6, 4, and 3 samples per cycle). Note that the leading zero in the **P** variable below represents the trend, while **TVP** specifies the model types. In this analysis, the final three years of data are removed from the series in order to later illustrate the forecasting performance of the model. The resulting fit in the frequency domain, a standard output of the **dhropt** function, is shown in Figure 3.3.

```
>> P = [0 12 6 4 3]
>> TVP = [1 0];
>> nvr = dhropt(co2(1:288), P, TVP);

METHOD: FREQUENCY DOMAIN. AR-SPECTRUM(24)
OPTIMISER: LEASTSQ
0.711 seconds.
3 missing values
   PER.     RW        NVR          Score     S.E.    Alpha     Score    S.E.
    0.00    1.0    3.4771e-003   -2.4588    0.042    1.0000        -       -
   12.00    0.0    7.1466e-002   -1.1459    0.030    1.0000        -       -
    6.00    0.0    2.0435e-002   -1.6896    0.040    1.0000        -       -
    4.00    0.0    8.0806e-004   -3.0926    0.088    1.0000        -       -
    3.00    0.0    1.7861e-004   -3.7481    0.131    1.0000        -       -
```

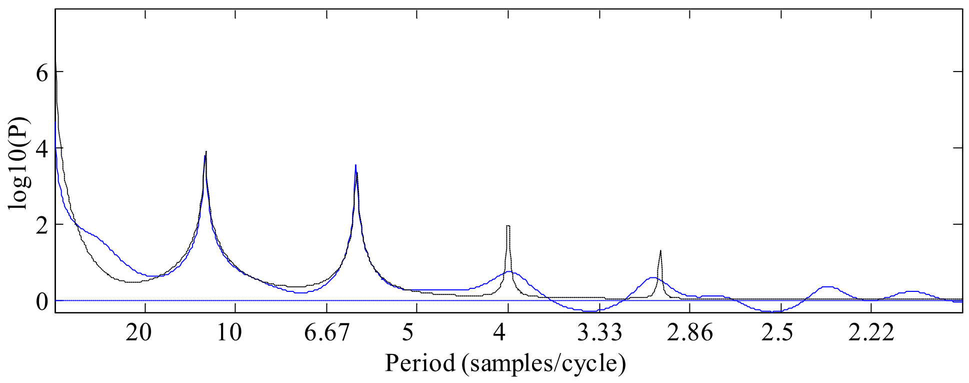


Figure 3.3 AR pseudo-spectrum of the CO$_2$ data (solid) and fit of the model (dotted).

Finally, the DHR model is estimated for the same data and settings using the NVR values listed above. Here, **fcast** is utilised to add **nan**'s to the series, representing the three years of artificially induced missing data; as discussed in Chapter 2, this is the approach taken in

CAPTAIN to generate forecasts. The trend estimate, forecasts of the series with 95% confidence intervals, seasonal component and its forecasts, together with the remaining irregular components are all illustrated in Figure 3.4, using the commands below. Here, the variable **tf** is introduced for convenience, simply to define the sample numbers over which the model is forecasted.

```
>> [fit, fitse, tr, trse, comp, e] = ...
>>              dhr(fcast(co2(1:288), [0 36]), P, TVP, nvr);
>> t = [1 : length(co2)]';
>> tf = (289 : 324)';
>> bands = [fit(tf)+2*fitse(tf) fit(tf)-2*fitse(tf)];  % Confidence bands
>> subplot(311); plot(t, [co2 fit tr(:, 1)], tf, bands, ':');
>> S = sum(comp')';  % Total seasonal component
>> subplot(312); plot(S)
>> subplot(313); plot([co2(1 : 288)-fit(1 : 288)])  % Irregular component
```

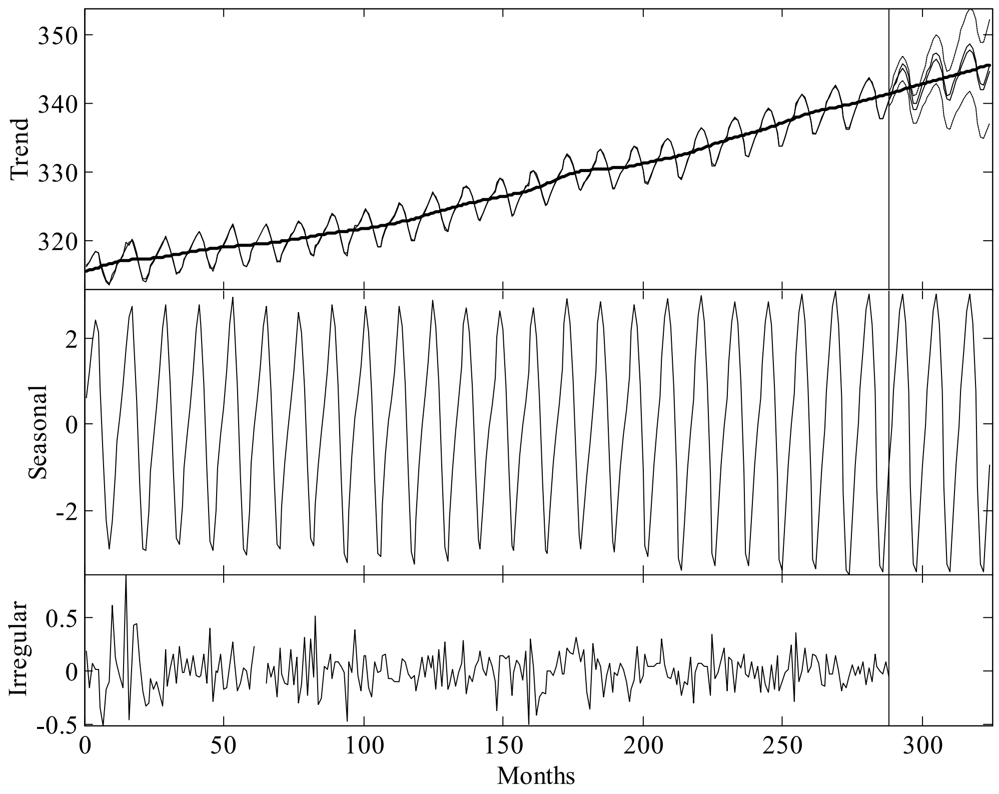


Figure 3.4 Trend, forecasts, seasonal and irregular estimates of the CO_2 series.

Many other model types can be implemented using the same **dhr**/**dhropt** pairing in CAPTAIN. For example, by choosing `TVP = [1 1]`, IRW models are selected for each of the TVP's modulating the harmonics while, if instead, `TVP = [1 2]`, then a trigonometric cycle is used. Alternatively, SRW models or damped seasonal/cyclical components may be specified as follows,

```
>> [nvr, alpha] = dhropt(co2(1 : 288), P, TVP, [], -2, -2);
```

Here, the -2 terms in **dhropt** indicate free (unconstrained) estimation of both the NVR hyper-parameter and the smoothing (α) parameter in equation (2.2). Finally, LLT or Damped trends may be estimated by specifying two zeros in the vector of periodic components, as shown below.

```
>> P = [0 0 12 6 4 3];  % periodic components for LLT or damped trend
>> TVP = [1 0];         % use RW and IRW trends simultaneously
>> nvr = dhropt(co2(1:288), P, TVP);  % LLT
>> [nvr, alpha] = dhropt(co2(1:288), P, TVP, [], -2, [1 -2 1]);  % Damped
```

For the present example, the damped trend yields numerical problems and a very poor model fit for the first part of the time series, when using the default ‘Q-algorithm’ for FIS. Changing to the ‘P-algorithm’ (see equation (2.12) in Chapter 2) by means of the ninth input argument to **dhr** solves the problem, as follows:

```
>> [fit, fitse, tr, trse, comp, e] = ...
     dhr(fcast(co2(1:288), [0 36]), P, TVP, nvr, alpha, [], [], [], 0);
```

All the examples above utilise the specially developed frequency domain optimisation routine for DHR model hyper-parameter estimation. However, as discussed in Chapter 2, ML is usually available in CAPTAIN as an alternative, even though it is sometimes unable to provide an appropriate solution unless it is constrained in some way. For example, the command,

```
>> nvr = dhropt(co2(1 : 288), P, TVP, -24);
```

utilises ML but takes a long time or is unable to find a solution before reaching the maximum number of iterations, or possibly converges to a local minimum, with associated MATLAB® warnings automatically generated. This is because the log-likelihood surface is very flat around the optimum in this case (see Young *et al.*, 1999). Note that the fourth input argument above specifies in an initial condition for ML obtained from a frequency optimisation with an AR(24) spectrum. To help address such problems, an appropriate constraint is introduced into the model. In this regard, one common solution is to impose the same NVR values to all the seasonal harmonics using the fifth input argument to **dhropt**. Of course, this solution might be very different to the earlier frequency domain results, because of the entirely different method and the artificial constraints imposed.

Sequential Spectral Decomposition

Sequential spectral decomposition is designed for the Trend + AR type of model implemented in the CAPTAIN functional pair **univ**/**univopt**, although in principle it may be applied to any UC model (see latter examples). The approach was developed to avoid certain identification problem that arise when using AR models. Such problems occur because the AR model, which is ideally reserved for the perturbational component about the trend, may in fact describe the whole series (i.e. the trend and the perturbation) if a joint estimation is attempted without constraints. In other words, there is nothing in the Trend + AR model that guarantees that the joint estimation is going to yield a perturbation and a trend orthogonal to each other, and with the frequency properties assumed in principle, i.e. the trend as a smooth line across the data and the perturbation as a component wandering about the zero line in a way such that the sum of both optimally fits the time series.

This does not mean that such a model would not be useful for forecasting purposes, simply that each component by itself is not meaningful. Typically, one may find that the trend does not follow the data and that the perturbational component does not oscillate around zero as required (i.e. it is not stationary). This identification problem does not appear in the BSM or DHR models because in these cases, each seasonal harmonic is by definition independent to the rest and to the trend, so that each one concentrates on a particular narrow frequency band. In the case of the Trend + AR models, the problem may be conveniently solved in either four or five steps, as follows.

- Estimate an initial trend component on the basis of the IRW model (see examples in Chapter 2). The NVR may be chosen using any *a priori* knowledge; on the basis of the frequency domain properties of this low-pass filter; or may be objectively estimated in some manner, such as by ML or the minimum of the multiple-steps-ahead forecasting errors. This initial selection of the NVR resembles Bayesian methods in an UC context, in the manner of West and Harrison (1989).

- If an IAR or a DIAR trend proves necessary, then the identification and estimation of the AR polynomial for the trend must be based on the first or second difference of the initial trend estimate, respectively.

- Obtain an initial estimate of the perturbational component as the difference between the data and the estimated trend in the previous step. Estimate an AR model or a subset AR model for this component.

- Re-estimate the NVR parameter for the trend. The initial NVR selected for the trend is, by definition, the ratio of the variance of the state noise to the variance of the observational noise in the initial model. However, this initial model does not account for the new perturbational AR component and the variance of the observational noise is generally much bigger than in the complete model, hence the NVR should be modified accordingly.

- Re-estimate the components based on the full model including the Trend and the AR model with the new NVR for the trend and the estimated AR model for the perturbational component.

All these steps are automatically handled by the **univopt** and **univ** functions. In this way, the overall non-linear problem is decomposed into several linear or quasi-linear steps, each solved in fully recursive terms. This simple solution, which has some loss of optimality from the ML viewpoint, has proven to be very successful in practice.

Example 3.2 Modelling the US GDP using Trend + AR models

Consider the quarterly US Gross Domestic Product (GDP) data from the first quarter of 1947 until the last quarter of 2002. This series, which has already been seasonally adjusted by the authorities, is illustrated in Figure 3.5 using the command below.

```
>> load usgdp.dat
>> t = (1947 : 1/4 : 2002.9)';
>> plot(t, usgdp);
>> y = log(usgdp(1 : 204));
```

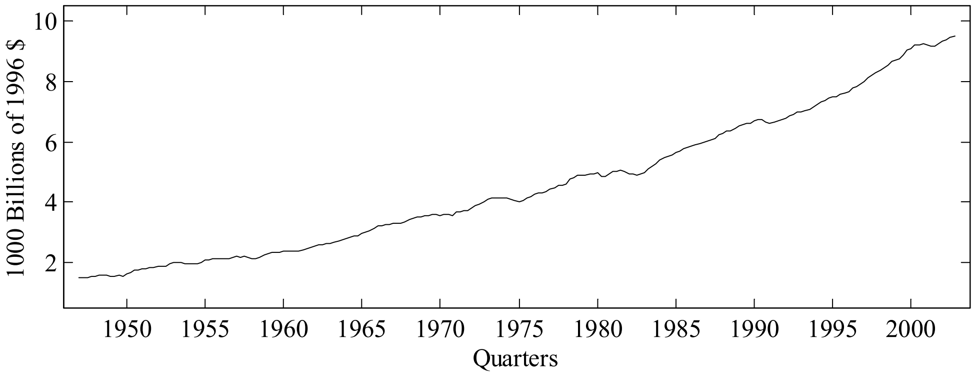


Figure 3.5 The seasonally adjusted US Gross Domestic Product between 1947Q1 and 2002Q4.

The last line of code above transforms the data to a logarithmic scale and reserves the final 20 observations for forecasting comparisons, in a similar manner to the earlier CO_2 example. The present data are particularly interesting since the perturbations about the trend cannot be modelled as a seasonal component because it is a seasonally adjusted time series. However, it is not believed that the remaining perturbations are white noise, because there is evidence of a business cycle in the series. In such cases, a DHR type model with some periodic cycle of appropriate length may be fitted (e.g. Koopman *et al.*, 2000), but the existence of such a cycle is rather dubious. Consider, for example, the inconclusive spectra estimates obtained from the following commands.

```
>> arspec(y);
>> period(y);
```

For these reasons, CAPTAIN provides the Trend + AR model. This approach may be regarded as a powerful extension to its simpler predecessors, i.e. the IRW or HP filter which are traditionally applied to these data (e.g. Hodrick and Prescott, 1997). Following the sequential spectral decomposition method outlined above, the first step is to select an NVR for the trend alone in an IRW + white noise model. For example, if the minimisation of the 4-steps-ahead forecasting errors is chosen (since 4 is the number of samples per year),

```
>> nvr0 = irwsmopt(y, 1, 'f4')
nvr0 =
    0.0013
>> tr = irwsm(y, 1, nvr0);
```

The NVR fitted in this way corresponds to a cut-off period of 8 years and one quarter, i.e. all the periods above that value are included in the trend (Table 2.1). The second step is to select the order of the AR model for the perturbations. It can be identified using either the **acf** or **aic** functions as follows.

```
>> aic(y-tr, [], 1);
>> acf(y-tr);
```

The optimal model order chosen by the AIC criterium is the AR(13) model. The estimation of the whole model, conditional on the NVR of the trend in the first step then follows,

```
>> [nvr, ARp] = univopt(y, [1:13], 1, nvr0);
ESTIMATION OF TREND+AR MODEL
AR Model for Perturbations:
==========================
  AR     ARp     S.E.       T
    1  -0.9709   0.0715  -13.5808
    2   0.1517   0.0948    1.6006
    3   0.1513   0.0936    1.6158
    4   0.0292   0.0941    0.3100
    ...
   10  -0.0208   0.0937   -0.2224
   11   0.0118   0.0930    0.1270
   12   0.2062   0.0927    2.2250
   13  -0.0605   0.0676   -0.8943
 Final trend NVR estimate: 5.9559e-03
 Integration order of trend: 2
```

The table shows the AR lag in the first column; the point estimate for each parameter in the second column; their standard error in the third column; and the typical T statistic (i.e. each parameter divided by its standard error) in the fourth column. At the bottom of the table the new estimate of the trend NVR is calculated, which is greater than the initial **nvr0**. The reason is clear, **nvr0** was the ratio between the trend variance and the perturbation variance, in a model in which the perturbation about the trend was assumed to white noise but which actually included the whole perturbational component; while the new **nvr** is the same ratio based on a much smaller observational noise variance.

It is clear that not all the lags are significant and necessary in the AR model based on the T-test, and a subset model would perform as well. This is the reason why a final estimation iteration is usually worthwhile, i.e.,

```
>> [nvr, ARp] = univopt(y, [1:3 12 13], 1, nvr0);
ESTIMATION OF TREND+AR MODEL
AR Model for Perturbations:
==========================
  AR     ARp     S.E.       T
    1  -1.0243   0.0686  -14.9365
    2   0.1265   0.0958    1.3204
    3   0.2418   0.0660    3.6648
   12   0.1849   0.0594    3.1138
   13  -0.0889   0.0593   -1.4987
 Final trend NVR estimate: 5.6884e-03
 Integration order of trend: 2
```

The associated smoothing, forecasting, signal extraction, etc. of the series, using the final NVR estimate above, is achieved by means of the **univ** function,

```
>> [fit, fitse, tr, trse, comp] = univ(fcast(y, [0 20]), ARp, 1, nvr);
>> subplot(311), plot(t(1 : 204), tr(1 : 204))
>> subplot(312), plot(t(1 : 204), comp(1 : 204))
>> subplot(313), plot(t(1 : 204), y-fit(1 : 204))
```

The components are shown in Figure 3.6, where a significant decrease in variance of both the cycle and the irregular component can be observed after the middle of 1982.

Finally, Figure 3.7 shows a detail of a forecasting exercise five full years ahead, starting on December 1997, where it can be seen that almost all the data lie within the standard error bounds of the forecast. This figure may be plotted as follows,

```
>> tfor = (205 : 224)';
>> confband = [fit(tfor) + 2*fitse(tfor) fit(tfor) - 2*fitse(tfor)];
>> plot(t, [log(usgdp) fit], '-', t(tfor), confband, ':')
```

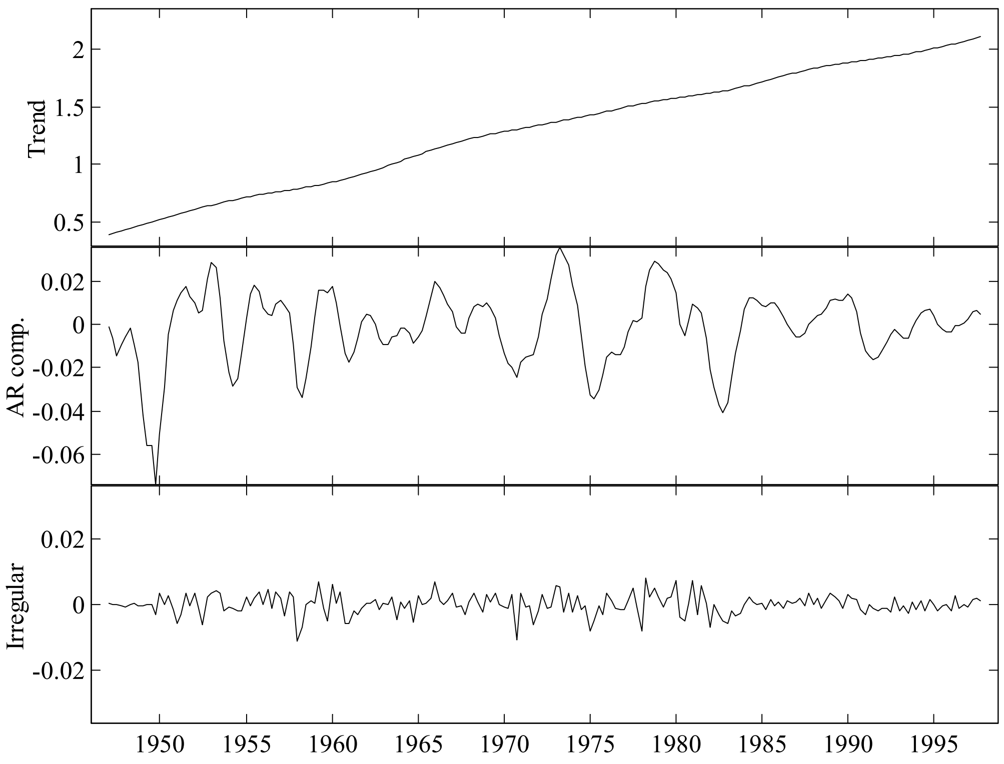


Figure 3.6 The components of the trend + AR model for the US GDP.

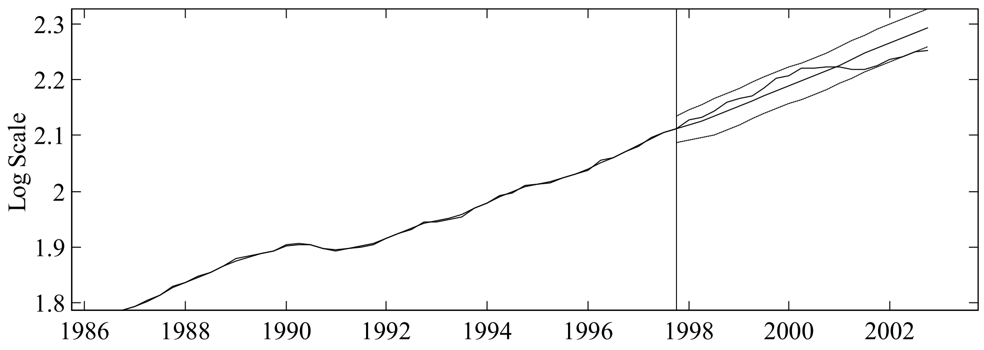


Figure 3.7 Example five year ahead forecasts for the US GDP.

3.4 Advanced Examples using Variance Intervention

The following two examples introduce an iterative approach for the simultaneous estimation of the trend NVR and perturbations when variance intervention is required.

Example 3.3 Steel consumption in the UK revisited

In Chapter 2, Example 2.3 considered the quarterly steel consumption data (Figure 2.4) in the context of a simple trend only model. However, as pointed out then, a better option would be to estimate a UC model in which all the components were fitted jointly, an approach pursued below.

Spectral identification is a very important step in this analysis, required in order to check for the existence of all or part of the seasonal harmonics. In this case, it is often preferable to calculate the AR-spectrum of the series when any dramatic jumps in the trend have already been removed using the **reconst** function, as discussed in Example 2.2 (see Figure 2.6). Such jumps in the trend could distort the spectrum estimate of the original series, especially the frequency band corresponding to the trend, as illustrated in Figure 3.8. In fact, it is clear from Figure 3.8, that while the distortion is minimal with respect to the seasonal spectral peaks (4 and 2 samples/cycle), it is quite noticeable in the low frequency band of the spectrum. Figure 3.8 also reveals the existence of an approximate four years cycle in the data and most of the associated harmonics (note that the seasonal peak is one of these harmonics).

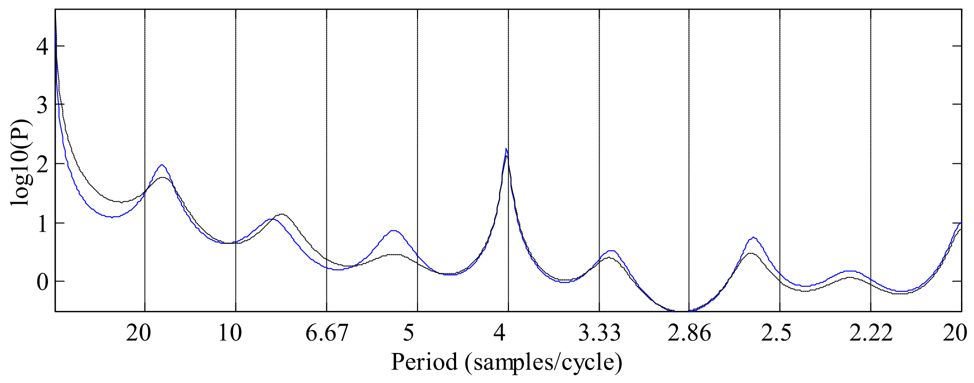


Figure 3.8 AR(17)-spectrum of raw data (dotted) and the reconstructed series from Figure 2.6 (solid).

Because of such potential distortions in the spectrum, frequency domain methods should always be applied with care, especially when estimating the trend (and cyclical) NVR hyper-parameters. Indeed, this is a typical case where ML in the time domain with simultaneous variance intervention estimation may yield a superior answer. In this regard, a DHR model may be estimated by entering the following commands.

```
>> load steel.dat
>> P = [0 16 8 16/3 4 2];
>> TVP = [1 0];
>> nvr = dhropt(steel, P, TVP, -17, [],[],[],[],[],[],[],[],[87 106]);
>> [fit, fitse, tr, trse, comp] = ...
>>            dhr(steel, P, TVP, nvr, [], [], [], [], [], [87 106]);
```

The fourth input argument to **dhropt** specifies ML optimization using a frequency domain estimation based on the AR(17) model as the initial condition. The final input argument provides the variance intervention points, using a similar syntax to the **irwsm/irwsmopt** pair introduced in Chapter 2.

A more satisfactory solution may be obtained by an iterative procedure which combines both frequency and time domain methods. Although more complex to implement, this approach arguably provides a more objective and complete UC analysis. Indeed, while the simple commands above may be regarded as the basic default option, the code below is a good illustration of the open architecture of CAPTAIN in the MATLAB® environment, by which each user may find different ways to exploit potential solutions to a particular time series problem.

The idea here is that NVR frequency estimation based on the raw data is contaminated by jumps in the trend. If these jumps were known, they could be removed and the contamination problem would disappear. Obviously, such information is not immediately available and has to be estimated simultaneously with the NVR hyper-parameters. In other words, trend jumps may be estimated conditional on given NVR parameters by variance intervention, while the NVR parameters themselves may be estimated in the frequency domain conditional on given estimates of the trend jumps. In this case, a simultaneous, unconditional estimate of the trend jumps and NVR parameters may be obtained by the following iterative algorithm (cf. Example 2.3).

```
>> Int=[87 106];  % interventions 1975Q1 and 1980Q1
>> nvr0 = irwsmopt(steel, 1, 'f12', Int);
>> tnew = irwsm(steel, 1, nvr0, Int);
>> ynew = steel - tnew + reconst(tnew, Int);
>> P = [0 16 8 16/3 4 2]; TVP = [1 0];
>> tol = 1e-4; iter= 0; obj= 1000*tol;
>> while obj > tol
>>    nvr = dhropt(ynew, P, TVP, 17);
>>    [fit, fitse, tnew, trse, comp] = ...
>>                   dhr(steel, P, TVP, nvr, [], [], [], [], [], Int);
>>    ynew = steel - tnew(:, 1) + reconst(tnew(:, 1), Int);
>>    if iter>0, obj = max(abs(nvrold-nvr)); end
>>    nvrold = nvr; iter = iter+1;
>>    [iter obj]
>> end
```

The first four lines yield an initial estimate of the constructed series without the trend jumps based on the IRW + noise model (see Figure 2.6). Subsequently, the iterations are built in such a way that the NVR parameters are estimated using continuously updated versions of the jump-free series (first line after the `while` sentence). However, in each case the smoothing is based on the DHR model for the raw data, the current estimate of the NVR parameters and the variance intervention points. The iterations end when the difference between two subsequent estimates of NVR values are less than the `tol` control value, here with NVR values $[0.0001, 0.0434, 0.0000, 0.0009, 0.0233, 0.0017]$. Finally, the trend estimates, together with the cyclical and seasonal components are all illustrated in Figure 3.9, obtained as follows.

```
>> subplot(311), plot([steel tnew])          % Trend and series
>> subplot(312), plot(sum(comp(:, 1:3)')')   % Cycle
>> subplot(313), plot(sum(comp(:, 4:5)')')   % Seasonal
```

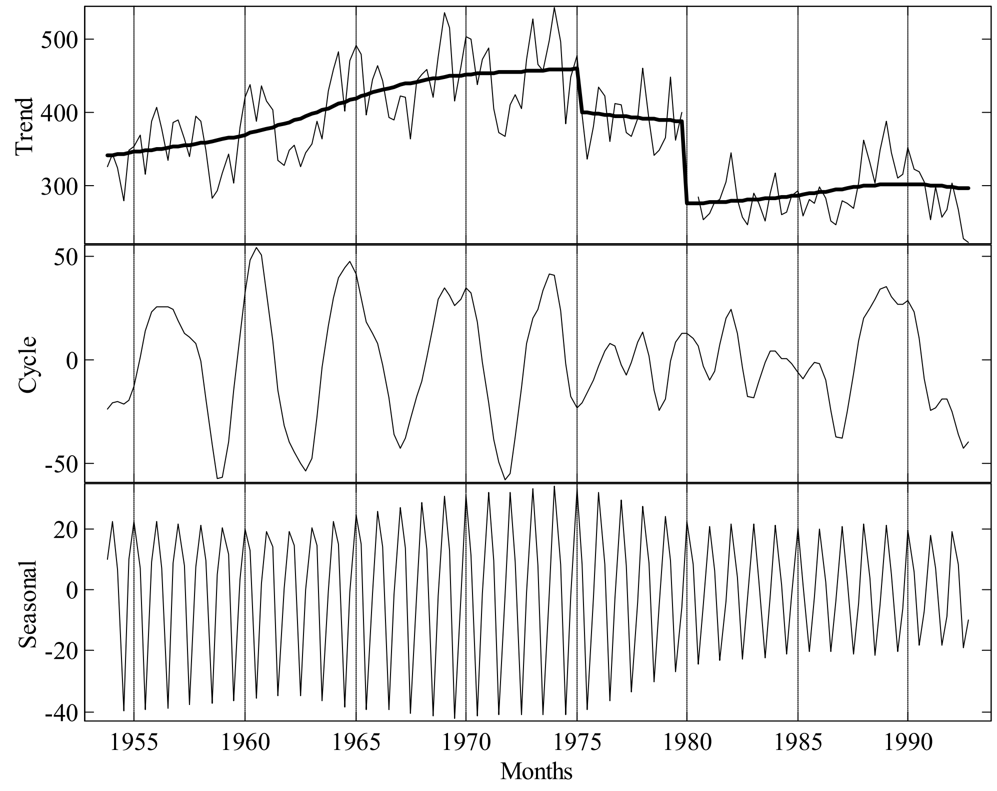


Figure 3.9 Estimated components of the steel consumption series, using the frequency domain method.

Example 3.4 Car drivers killed and seriously injured

In order to show that the iterative procedure developed in the previous example is applicable to other data sets, a similar analysis is performed for the monthly car drivers killed and seriously injured in Great Britain from January 1969 to December 1984. Figure 3.10 displays this series using the following commands.

```
>> load cars.dat
>> t = (1969 : 1/12 : 1984.99)';
>> plot(t, cars)
```

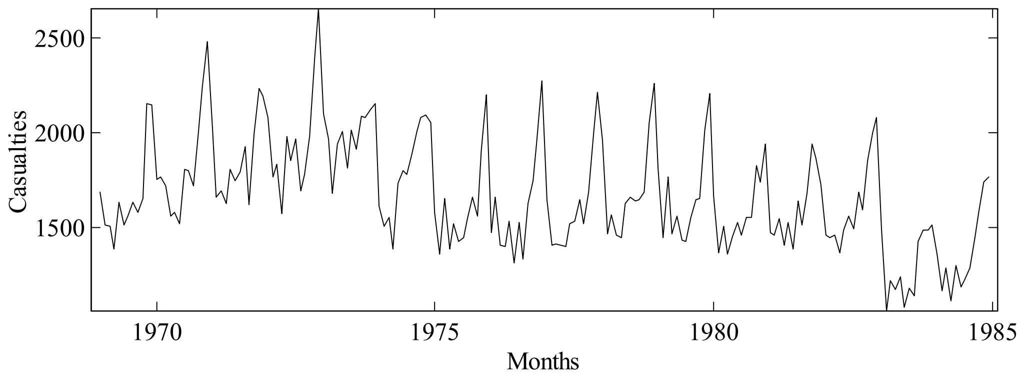


Figure 3.10 Car drivers killed and seriously injured in Great Britain from January 1969 to December 1984.

A mild trend and seasonal components are clearly visible in these data. However, more interesting, are the apparent jumps in the series related to seat-belt legislations in January 1974 and February 1983 (samples 61 and 179). As before, these may be accounted for with the trend signal using variance interventions. In this case, it is also clear that the seasonal pattern changes in 1974 and 1975 after the first intervention point, so for the purposes of the present example, a third intervention is set at sample 74 (February 1975).

An initial estimate of the trend is first obtained using **irwsm/irwsmopt** with variance interventions. The AR-spectrum of the raw data and the reconstructed intervened data are shown in Figure 3.11, where it is clear that the final harmonic is not present and is, therefore, not necessary in the analysis. Furthermore, the distortion due to the trend jumps are less significant here than in the steel consumption series. Nonetheless, for illustrative purposes, the full iterative procedure outlined in the previous subsection is still implemented and yields the components illustrated in Figure 3.12, together with the reconstructed series in Figure 3.13.

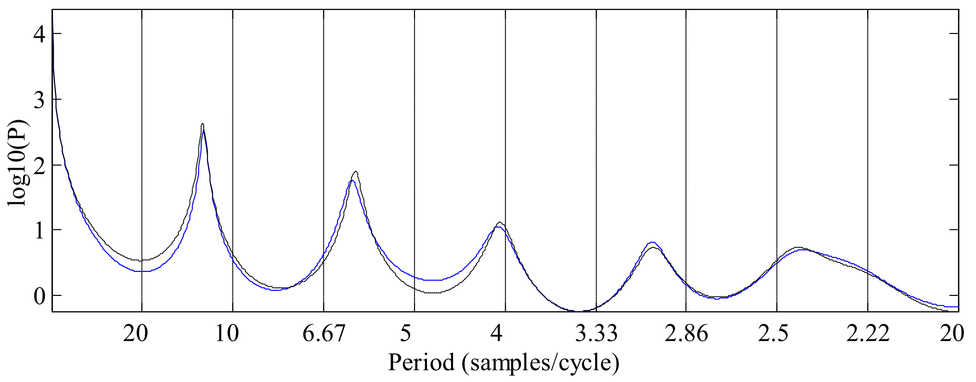


Figure 3.11 AR-spectrum of the driver casualties data (dotted) and constructed series (solid).

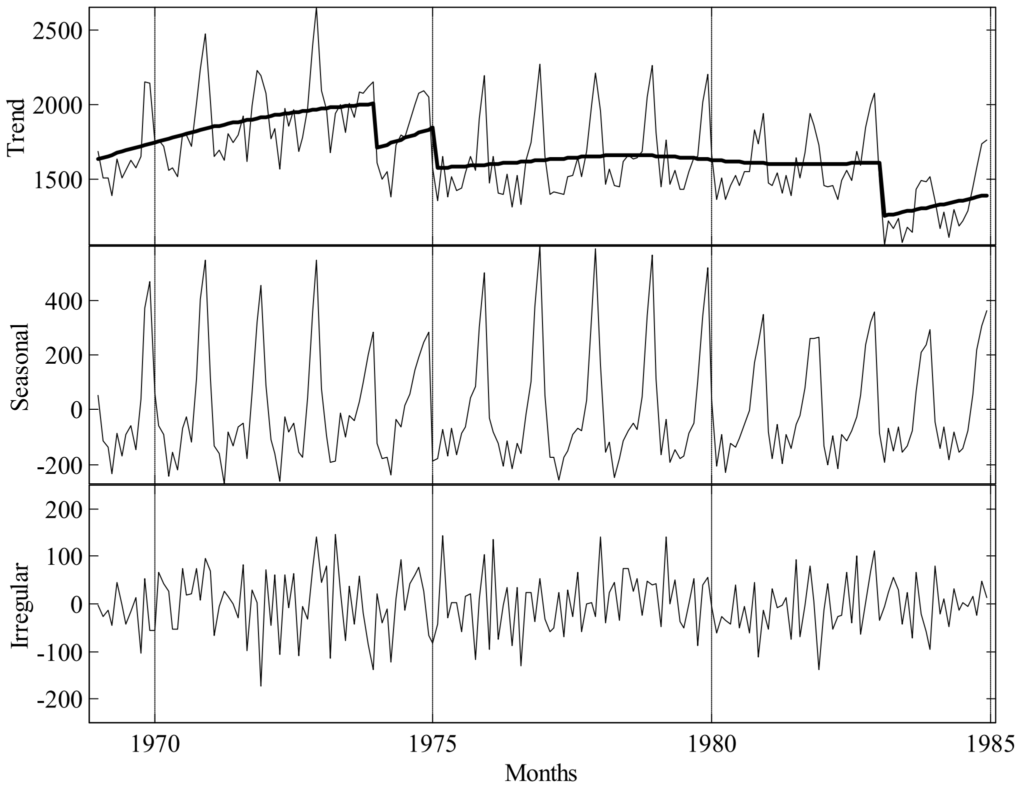


Figure 3.12 Estimated components of the driver casualties data.

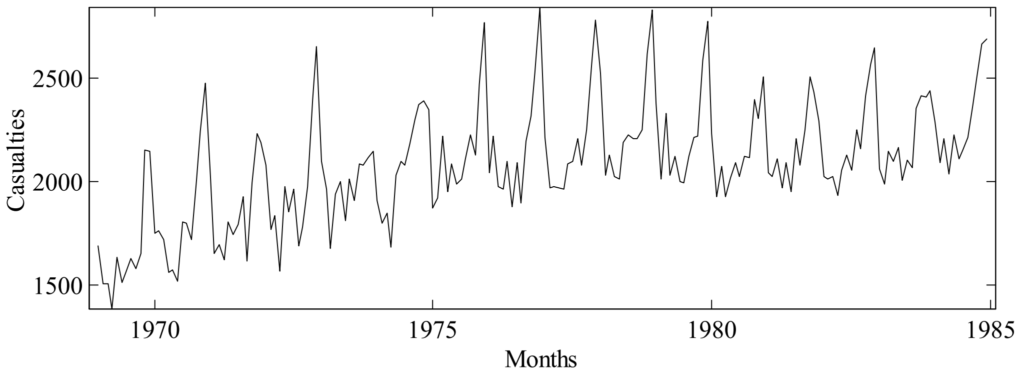


Figure 3.13 Driver casualties data with the variance interventions removed.

3.5 Conclusions

The present chapter has introduced the Unobserved Components (UC) modelling tools in CAPTAIN, and shown how the toolbox may be utilised for forecasting and signal extraction of periodic time series with widely varying characteristics.

One point worth stressing again, is that in every example included in the chapter, the seasonal components are such that not all of the theoretical harmonics are observed in the data. In this regard, the present authors believe that the identification stage utilised above, based on both spectral and time domain methods, is a particularly important part of the analysis, although it is often neglected in conventional UC modelling. In addition to providing evidence on whether all or only some of the harmonics are really necessary (especially important for frequency domain estimation methods), this also identifies the relative importance of each component (measured by the relative magnitude of their NVR). This latter information is useful if the model has to be constrained in some way at the estimation stage.

To illustrate the methodology, the models above have been limited to a trend and a cyclical or seasonal component. However, in fact, CAPTAIN allows for a much wider range of models than considered so far. The following chapters, therefore, introduce various additional components such as exogenous variables.

CHAPTER 4
TIME VARIABLE PARAMETER MODELS

Chapters 2 and 3 have developed an approach to nonstationary signal processing based on the identification and estimation of time variable parameter (TVP) stochastic models. The methodological tools that underpin this modelling philosophy are unified in terms of the discrete-time Unobserved Components (UC) model (3.1). Here, in order to allow for non stationarity in the time series y_t, it is assumed that the various components in the model, including the trend T_t, can be characterised by TVP's.

Most often, the nature of such parametric time variability will not be known prior to the analysis and so each TVP is defined as a non stationary stochastic variable. This adds a statistical degree of freedom to the estimation problem, so allowing for the estimation of any 'slow' parameter variations. By slow, we mean here variations that are slow in relation to the variations in the time series itself. Such variations may result from slow physical changes in the process or from some form of nonlinearity in the data. In this manner, the models obtained are all inherently self-adaptive: namely, they change their parameters automatically in an optimal manner to reflect changes in the nature of the time series. For this reason, they can be exploited in applications such as self-adaptive forecasting, operational control and management.

In practice, as mentioned in Chapter 3, not all the possible components in the UC model are necessary: indeed, the simultaneous presence of all these components can induce identifiability problems in which it is not possible to unambiguously estimate the model. For this reason, the models considered in Chapter 3 are limited to univariate time series characterised by a trend, together with a sustained cyclical and/or seasonal component. To complete the discussion, therefore, the present Chapter considers the other optimal recursive TVP models that may be estimated using CAPTAIN.

In particular, the Chapter describes the entire class of TVP, or 'dynamic', regression models, including Dynamic Linear Regression (DLR), Dynamic Harmonic Regression (DHR) and Dynamic Auto-Regression (DAR), as well as the closely related, TVP version of the Auto-Regressive eXogenous variables model (DARX). Finally, the Chapter considers an alternative Dynamic Transfer Function (DTF) model, estimated using an instrumental variable method of fixed interval smoothing, and shows how this is superior to the DARX model when measurement noise is present. The practical utility and self-adaptive functionality of the dynamic regression model in these various forms is illustrated by both simulated and practical examples.

In CAPTAIN, the required forward pass filtering and fixed interval smoothing algorithms are accessible via shells, namely the functions **dlr**, **dhr**, **dar/darsp**, **darx** and **dtfm**, while associated hyper-parameters are estimated using **dlropt**, **dhropt**, **daropt**, **darxopt** and **dtfmopt** respectively. These shells provide for ready estimation of the various special cases discussed below.

4.1 Dynamic Linear Regression (DLR)

As discussed in Chapter 2, the SS model (2.1) is particularly well suited to estimation based on optimal time variable parameter recursive estimation, in which the time variable parameters (acting as surrogate 'states') are estimated sequentially by the Kalman Filter (KF) whilst working through the data in temporal order. In the off-line situation, where all the time series data are available for analysis, this filtering operation may be accompanied by optimal Fixed Interval Smoothing (FIS).

In this regard, one of the simplest yet widely applicable SS models using time variable parameters, is a DLR model based on the exogenous input component $f(\mathbf{u}_t)$ of equation (3.1), interpreted in its most basic linear regression form, i.e.,

$$y_t = T_t + \sum_{i=1}^{i=m} b_{it} u_{it} + e_t \qquad e_t \sim N\{0,\sigma^2\} \qquad t = 1,2,...,N \tag{4.1}$$

where T_t is a trend or low frequency component; $b_{it}, i = 1,2,\ldots,m$ are either constant parameters (the normal regression model) or they may vary over the observation interval to reflect possible changes in the regression relationship; and $u_{it}, i = 1,2,\ldots,m$ are the regression (input or exogenous) variables that are assumed to affect the 'dependent' variable y_t. The presence of significant time variation can be due to various causes, dependent on the nature of the application, as discussed in the examples below. Finally, as shown, e_t is an 'irregular' component, normally defined for analytical convenience as a serially uncorrelated and normally distributed Gaussian sequence with zero mean value and variance σ^2 (i.e. discrete-time white noise).

Equation (4.1) is a generalisation of the two parameter DLR model introduced in the CAPTAIN Getting Started Guide: see equation (3.3). In this regard, note that T_t is effectively another TVP with an associated regression variable of unity and, if required, must be explicitly specified as such when using CAPTAIN.

Reflecting the statistical setting of the analysis, the stochastic evolution of each parameter is assumed to be described by the Generalised Random Walk (GRW) process introduced in Chapter 2, including RW, AR(1), IRW, SRW, LLT and damped trends as particular cases. As discussed previously, the AR(1), SRW and damped trend models all require the specification or optimisation of an additional 'hyper-parameter', α. An overall state space model (2.1) can then be constructed straightforwardly by the aggregation of the GRW subsystem matrices, in a similar manner to the examples given in Chapter 3 for dynamic harmonic regression.

Take, for example, a DLR model with an IRW trend and two inputs, where the latter are governed by SRW and RW parameters, respectively. The overall SS form of such a model is given by,

$$\begin{pmatrix} T_t \\ D_t \\ \hline b_{1t} \\ b'_{1t} \\ \hline b_{2t} \end{pmatrix} = \left(\begin{array}{cc|cc|c} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & \alpha & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} T_{t-1} \\ D_{t-1} \\ \hline b_{1t-1} \\ b'_{1t-1} \\ \hline b_{2t-1} \end{pmatrix} + \begin{pmatrix} 0 \\ \eta_{Tt-1} \\ \hline 0 \\ \eta_{1t-1} \\ \hline \eta_{2t-1} \end{pmatrix}$$

$$y_t = \begin{pmatrix} 1 & 0 & u_{1t} & 0 & u_{2t} \end{pmatrix} \boldsymbol{x}_t + e_t \tag{4.2}$$

Recall from equation (2.1) that the 'system disturbances' noise vector, denoted earlier by η_t, contains the white noise inputs to each of the TVP models, i.e. η_{Tt}, η_{1t} and η_{2t} here. These white noise inputs are assumed to be independent of the observation noise e_t and have a covariance matrix $\boldsymbol{Q}$ formed from the combination of the individual covariance matrices for each parameter. The associated NVR matrix $\boldsymbol{Q}_r$ is defined as follows,

$$\boldsymbol{Q}_r = \frac{\boldsymbol{Q}}{\sigma^2}$$

The NVR parameters that characterise $\boldsymbol{Q}_r$ are unknown prior to the analysis and clearly need to be estimated on the basis of the time series data y_t before the filtering and smoothing algorithms can be utilised. The optimization of both the NVR and α hyper-parameters in this DLR context, is accomplished either by Maximum Likelihood (ML) optimisation or by the minimisation of the multiple-steps-ahead forecasting errors, as discussed earlier.

Note that, in the case of the simplest random walk model for all the parameters involved, each parameter can be assumed to be time-invariant if the variance of the white noise input in the state equation is zero. Then the stochastic TVP setting reverts to the more normal, constant parameter regression situation. In other words, the recursive estimation algorithms described below for the general stochastic TVP case will provide constant parameter estimates identical to the normal *en-bloc* regression if RW models with zero variance white noise inputs are specified. Of course, there is some added value to the recursive solution even in this situation, since the user is provided with the recursive estimates over the whole interval.

Furthermore, forecasting, interpolation and backcasting are an inherent part of these filtering and smoothing algorithms. For example, if missing samples are encountered anywhere within the output series, then the KF and FIS algorithms provide an optimal interpolation (Chapter 2). If, on the contrary, missing observations are found immediately after the last sample or prior to the first one, optimal forecasts and backcasts are similarly produced. Of course, all these cases require knowledge or forecasts of the exogenous regression variables over the missing data period.

As illustrated in the example below, **dlr** and **dlropt** are the CAPTAIN functions for general DLR analysis and hyper-parameter optimisation, respectively.

Example 4.1 Initial Evaluation of the Relationship Between Sunlight and Dissolved Oxygen in the River Cam using DLR (Young, 1998b)

Although regression analysis is a particularly popular method of modelling economic, business and social data (see e.g. Example 1.1), DLR analysis can also prove useful in the initial data evaluation and the processing of environmental and other scientific data. For example, one interesting and successful practical example of the latter type is discussed by Young and Pedregal (1996) where this approach is utilised in the analysis of LIDAR (laser-radar) data. In the present example, however, we consider how DLR analysis can be applied to the data shown in Figure 4.1 (Beck and Young, 1975): namely 81 daily measurements of Dissolved Oxygen (DO) in the river Cam, near Cambridge; together with the associated measurements of sunlight hours.

As we shall see, DLR analysis is not an entirely appropriate method of modelling these data: indeed they have been selected here in order to stress the need for careful appraisal of the TVP estimation results before making any scientific inferences. In effect, the analysis does no more than provide an initial evaluation of the simplest possible relationship between the two time series and how it appears to change over time. However, the simplicity of the DLR analysis helps to expose more clearly how DLR modelling can function as a useful and easy to use exploratory tool in these initial stages of time series analysis.

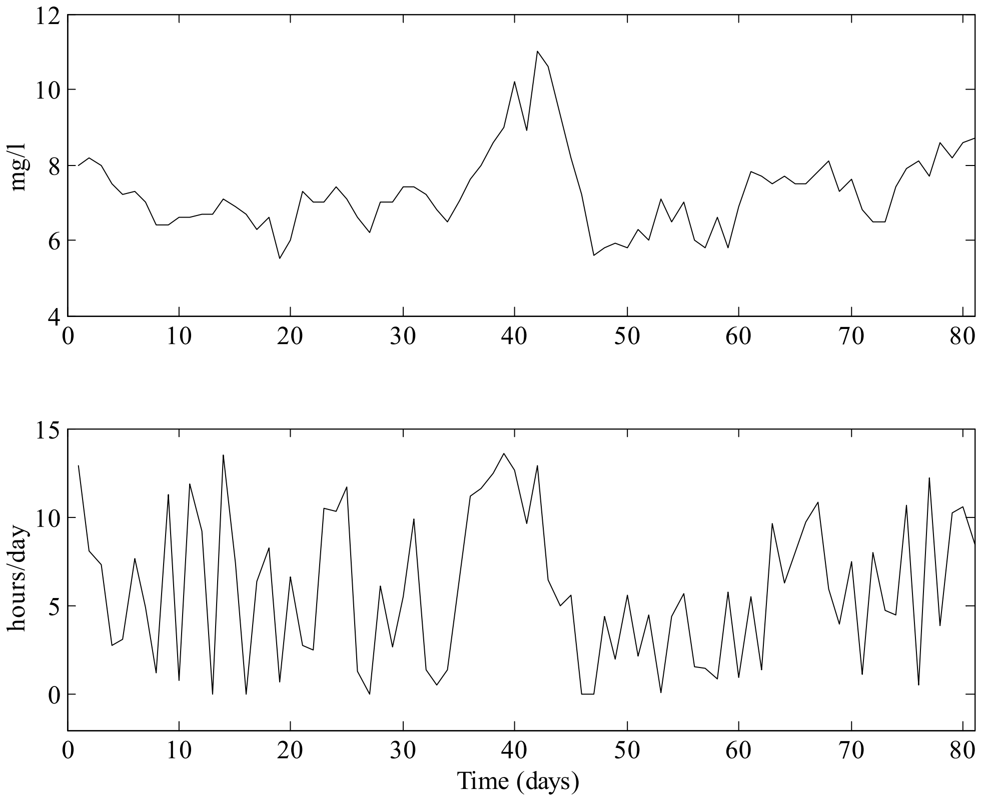


Figure 4.1 River Cam data: dissolved oxygen (top) and sunlight hours (bottom).

It is well known that sunlight can influence DO levels because of physical and biological factors and it is not surprising, therefore, that there is a visible relationship between the two variables in these plots. The maximum cross correlation coefficient between the series is 0.5542 when the sunlight series is lagged (delayed) by one sample.

Using CAPTAIN, this can be seen be entering the following commands,

```
>> load cam.dat
>> u = cam(:, 1);  % sunlight (hours/day)
>> y = cam(:, 2);  % DO (mg/l)
>> ccf(y, u);
```

Therefore, perhaps the most obvious *constant* parameter regression model takes the form,

$$y_t = T + b_1 u_{t-1} + e_t \qquad t = 1,2,...,81 \tag{4.3}$$

where y_t represents the DO measurements, while T and b_1 are constant parameters (again showing how the trend in the DLR model is a time variable equivalent of the 'intercept' parameter in the constant parameter regression model). This model is determined as follows,

```
>> z = [ones(size(u)) del(u, 1)];  % define regressors
>> [fit, fitse, par, parse] = dlr(y, z);
>> par(end, :)  % final parameter estimates
ans =
    6.4284    0.1423
>> parse(end, :)  % final standard errors
ans =
    0.1870    0.0256
>> r2 = 1-(cov(y)-cov(fit))/cov(y)  % coefficient of determination
r2 =
    0.3134
```

Here, the CAPTAIN function **del** provides the necessary lagged sunlight values. Only two input arguments to **dlr** are required, since constant parameters are assumed by default (i.e. RW model with NVR = 0 for both TVP's: see **dlr** help information). As shown above, this yields estimates of $T = 6.43 \pm 0.187$ and $\hat{b}_1 = 0.142 \pm 0.026$, together with a coefficient of determination $R^2 = 0.313$: i.e. the regression model with these constant parameter estimates explains only 31.3% of the DO series. The output of this model (dashed line) is compared with the DO data (circles) in Figure 4.2 and the poverty of the fit is obvious: it explains the intermediate values of DO between 6.5 and 8.5 mg/l to some extent, but fails completely to explain the larger deviations from the mean DO level (7.28 mg/l).

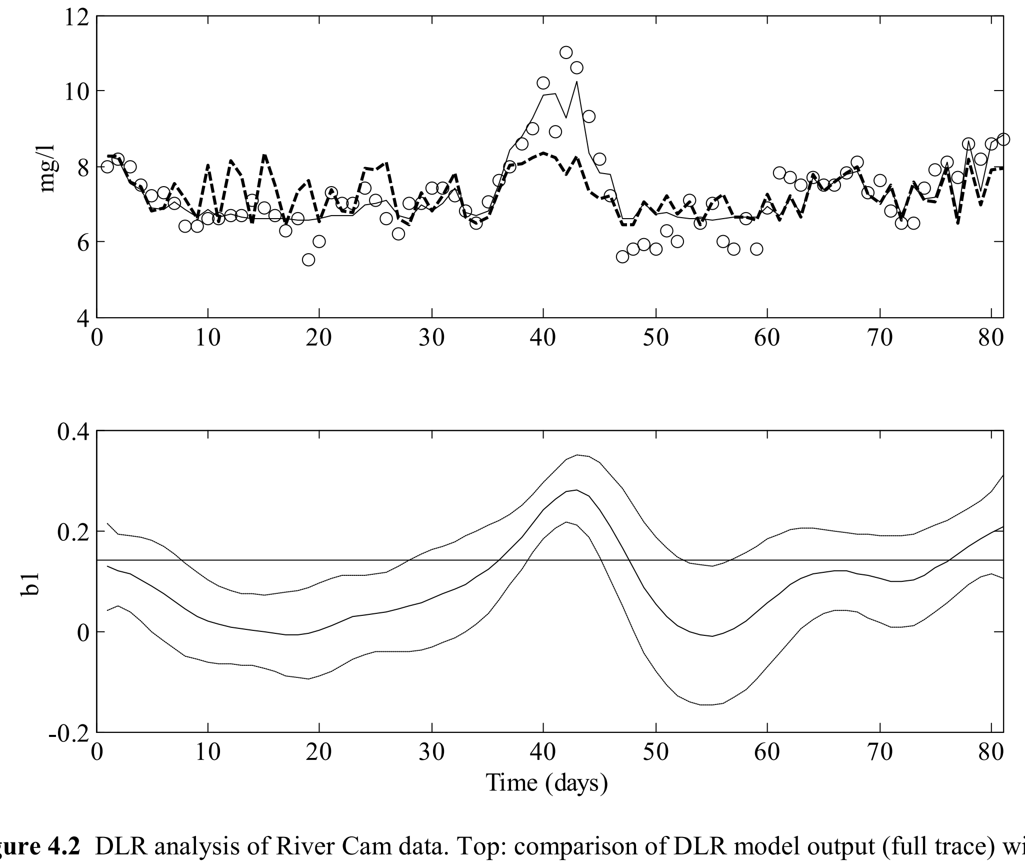


Figure 4.2 DLR analysis of River Cam data. Top: comparison of DLR model output (full trace) with DO data (circles), while the output of the constant parameter model is shown dashed. Bottom: time varying $\hat{b}_{1t}$ and standard errors, together with the constant parameter equivalent $\hat{b}_1 = 0.142$.

Therefore, it is useful to turn to the time variable form of the model. In fact, it makes some sense to constrain the $\hat{T}_t$ trend estimate to be constant in this case, in order to force all the estimated variation into the $\hat{b}_{1t}$ parameter, which controls the direct relationship between DO and sunlight. The first step is to optimise the NVR hyper-parameters as shown below,

```
>> nvr = dlropt(y, z, [0 1], [], [0 -2])
nvr =
  1.0e-003 *
         0
    0.4258
```

As before, the stochastic model for the variations in T_t are specified as a RW process with the associated NVR constrained to zero. Here, however, the $\hat{b}_{1t}$ parameter is defined as an IRW with a freely optimised NVR (see **dlropt** help information). It is clear that the default ML optimisation yields an NVR = 0.00043 for this model.

Some comment is required regarding the fifth input argument above, i.e. [0 -2], which specifies the constraints for each NVR, listed in the same order as the regressors. Here, any value greater than or equal to zero yields a fixed NVR of that value, while the -2 employed above implies free optimisation. Constrained optimisation is also possible: all NVRs associated with -1 will be optimised to the same value. For example, with 5 TVP's, specifying [-1 -1 -1 -2 0.1] implies that the first three NVRs will be optimised together (returning the same value), the fourth will be optimised independently, and the final NVR will take the defined fixed value (0.1).

Returning to the present example, the model fit is obtained in the usual manner,

```
>> [fit, fitse, par, parse] = dlr(y, z, [0 1], nvr);
>> par(end, :)
ans =
    6.6213    0.2081
>> parse(end, :)
ans =
    0.1501  0.0513
>> r2 = 1-(cov(y)-cov(fit))/cov(y)
r2_=
    0.6684
```

It is clear that the trend is now estimated as a constant value of $\hat{T} = 6.62 \pm 0.15$, while $\hat{b}_{1t}$ is shown as the time varying solid line in the lower panel of Figure 4.2 (it's final value $\hat{b}_{1N} = 0.2081$). As expected, this DLR model now has an improved value of $R^2 = 0.668$. Note that, by allowing both $\hat{T}_t$ and $\hat{b}_{1t}$ to vary over time as IRW processes, the fit may be improved further to $R^2 = 0.849$.

This DLR model seems reasonably satisfactory, but does it provide a meaningful representation of the data? In this regard, it is necessary first to consider whether the DLR normalised recursive residuals are satisfactory. Here, however, the Autocorrelation (ACF), Partial Autocorrelation (PACF) and Cross Correlation Functions (CCF) show some evidence of misspecification, albeit marginally, with some evidence of minor correlation in all cases. These statistical deficiencies of the model suggest simply that it is not an entirely appropriate representation of the relationship between sunlight and DO. This conclusion is not surprising since the real relationship is probably more complex and a static regression model, even with dynamically changing parameters, cannot hope to explain the data in an entirely satisfactory manner. For these reasons, we will return to this example later in the chapter, when discussing the more complicated DARX model.

4.2 Dynamic Harmonic Regression (DHR)

The DHR model contains the trend, cyclical, seasonal and white noise components of equation (3.1), i.e.,

$$y_t = T_t + S_t + C_t + e_t \qquad t = 1,2,\ldots,N \tag{4.4}$$

Although it is sometimes convenient to define the seasonal term S_t and the cyclical term C_t separately (e.g. Young, 1998), they are both modelled in the same manner and, in fact, no distinction is made in CAPTAIN. Both are defined by equation (3.8) and the CAPTAIN user simply specifies the periodic components required, i.e. the fundamental and harmonic frequencies associated with the seasonality, together with the frequencies associated with the (normally longer period) cyclical component.

In both cases, these frequency values are chosen by reference to the spectral properties of the time series, as discussed in Chapter 3. As for the DLR model above, the trend component T_t is also be considered as a stochastic, time variable 'intercept' parameter and so is incorporated, if so desired, into the cyclical or seasonal components as a zero frequency term. This DHR model can be considered as a straightforward extension of the classical, constant parameter, Harmonic Regression (or Fourier series) model, in which the gain and phase of the harmonic components can vary as a result of estimated temporal changes in the parameters.

In general, each of these TVP's, as well as the trend T_t, are modelled as GRW processes and the subsequent recursive estimation procedures are exactly the same as for the DLR model, except that the NVR values (and any other hyper-parameters) in the GRW models associated with the parameters of each ith component are usually constrained to be equal. However, as discussed in Chapter 3, the ML method used for hyper-parameter optimization in the DLR case does not work so well in this DHR context and so a novel frequency domain optimization algorithm has been developed for CAPTAIN.

It is worth noting that adaptive forecasting, interpolation and backcasting are much more straightforward than in the DLR case, because the regression variables in the DHR model are all known functions of time and can be specified easily outside the data sample when using the model for forecasting and backcasting purposes.

If desired, the DHR model can be estimated directly using the CAPTAIN function **dlr**, by manually specifying the regressors as appropriate harmonic components. However, special shells are included in the toolbox for this purpose, namely **dhr** and **dhropt**. These functions are useful for signal extraction and forecasting of periodic or quasi-periodic series, as shown by the examples in Chapter 3.

4.3 Dynamic Auto-Regression (DAR) and Time-Frequency Analysis

The basic DAR model is similar to the DLR model, except that the input variables are defined as past values of the output series. More formally, a DAR(p) model may be formulated as:

$$y_t = \frac{1}{A(L,t)} e_t \tag{4.5}$$

in which $A(L,t) = 1 + a_{1t}L + a_{2t}L^2 + \cdots + a_{pt}L^p$ is a time variable parameter polynomial in the backward shift operator L. On multiplying throughout by $A(L,t)$ so that it operates on y_t, we obtain the DAR(p) model in the discrete-time equation form:

$$y_t = -a_{1t}y_{t-1} - a_{2t}y_{t-2} - \cdots - a_{pt}y_{t-p} + e_t \tag{4.6}$$

In other words, y_t is dependent on past values of itself plus a random component in the form of the white noise e_t.

Noting that its constant parameter relative, the AR model, is used for spectral analysis in the form of the AR spectrum (see Chapter 3), the most obvious application of the DAR model is, therefore, in time-frequency analysis. Here, at the t-th time instant, the FIS estimated parameters $\hat{a}_{i,t|N}$, $i = 1,2,\ldots,p$ of the DAR model (4.5) and (4.6) are used to compute the instantaneous AR spectrum at that time from the well-known relationship (see e.g. Priestley, 1981),

$$\hat{h}(\omega)_t = \frac{\hat{\sigma}^2}{2\pi} \bullet \frac{1}{\left|1 + a_{1,t|N}\exp(-j\omega) + \cdots + a_{p,t|N}\exp(-j\omega)\right|^2} \quad ; \quad t = 1,2,\ldots,N \tag{4.7}$$

where $\hat{\sigma}^2$ is the estimated variance of the model residuals. The order p is selected either by the user on the basis of prior knowledge, or by use of the AIC (see Chapter 3). Then, for each user-selected value of ω over the range 0 (zero frequency) to 0.5 (the Nyquist frequency), $\hat{h}(\omega)_t$, or its logarithm, is evaluated with $\exp(-j\omega) = \cos(\omega) + j\sin(\omega)$. The set of all these instantaneous but smoothly changing spectra over the interval $t = 1, 2, \ldots, N$ then provide an indication of the changing spectral properties of the series y_t over this time interval. As we shall see in the example below, these time-frequency spectra can be presented in various ways.

The DAR model may be useful even when constant parameters are specified (by selecting zero NVR parameters), because unlike conventional AR-spectrum analysis, it still provides estimates when missing data are encountered. Indeed, when the CAPTAIN function **arspec** detects missing data, it automatically estimates the AR model recursively in this manner. Another important feature is that a constant parameter AR model estimated using the DAR function in CAPTAIN, provides recursive estimates of all the parameters and their standard errors, so that the assumed time invariance of these parameters may, in fact, be tested.

The DAR model can be estimated directly using the CAPTAIN function **dlr**, by manually specifying the regressors as appropriate past values of the output. However, this is not an optimal approach when the time series involves missing data, since such missing data in the output also generate missing values in the regressors (i.e. the lagged output variable). However, a solution is provided in CAPTAIN by replacing the missing values in the output and inputs by their expected values, according to the estimated model, as soon as they are detected. In other words, when a missing value is encountered at the p-th lagged output, the KF forecast replaces all subsequent occurrences of this datum in the analysis. Special shells are included in CAPTAIN for this purpose, namely **dar** and **daropt**, while the auxiliary function **darsp** allows for automatic graphing of the time-frequency spectra.

Example 4.2 Analysis of a signal with sawtooth changing frequency using DAR

This example considers a simulated signal with 'sawtooth' changing frequency. The data are loaded into the workspace, standardised and a time invariant AR model identified,

```
>> load tvp1.dat
>> y = stand(tvp1);
>> p = aic(y)
p =
    1.0000    0.0771   -0.1322
```

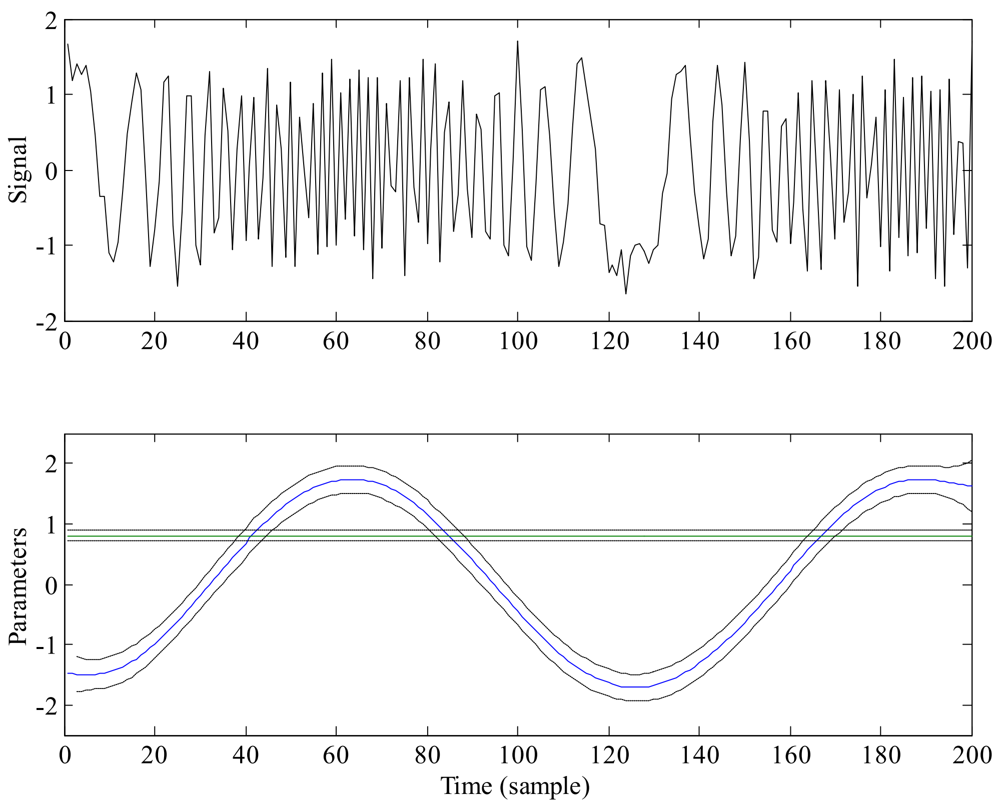


Figure 4.3 Simulated signal with sawtooth changing frequency (top); DAR parameters and their standard errors (bottom).

The standardised data are illustrated in Figure 4.3, while the AR(2) model is shown below,

$$y_t = -0.0771 y_{t-1} + 0.1322 y_{t-2} + e_t \tag{4.8}$$

In the straightforward command above, **aic** automatically selects the model order using the AIC criterion (Akaike, 1974). As an aside, note that the toolbox function **mar** can be used to estimate conventional AR models for any model structure. For example, the AR(2) model (4.8) is alternatively obtained using (see help information for **mar**),

```
>> th=mar(y, 2); p=getpar(th)
```

Returning to the DAR analysis, the NVR hyperparameters and DAR model are estimated, and the time-frequency spectra graphed, as shown below,

```
>> nvr = daropt(y, [1:2], [1 0])
nvr =
    0.0011
    0.0000
>> [fit, fitse, par, parse]= dar(y, [1:2], [1 0], nvr);
>> par(end, :)
ans =
    1.6164    0.8056
```

```
>> parse(end, :)
ans =
    0.2159    0.0456
>> darsp(par, 6, 1);
```

Note that the second input argument [1:2] in the calls to **daropt** and **dar** specify the structure of the DAR model, based on (4.8), and takes the same syntax as that used for **mar** and **univ** (see Chapter 3). For the purposes of this example, IRW and RW models are chosen for a_{1t} and a_{2t}, respectively, with the DAR model taking the following form,

$$y_t = -a_{1t}y_{t-1} - a_{2t}y_{t-2} + e_t \tag{4.9}$$

Here, it is interesting to note that the second TVP takes an almost constant value of $a_{2t} \approx 0.81 \pm 0.05$ (the associated ML optimised $\text{NVR} = 1.52\text{e-}017$), leaving the first parameter a_{1t} ($\text{NVR} = 0.0011$) to account for the time varying frequency of the original signal, as shown by the lower plot of Figure 4.3.

In this regard, of more interest is the time-frequency spectra of the series shown in Figure 4.4. Here, the second and third input arguments to **darsp** specify that a 3D spectrum plot with a resolution of 2^6 is required. The visual appearance of this plot depends on the computer platform and it is sometimes necessary to experiment to find the best resolution. Finally, note that contoured surfaces and 2D stacked plots may also be obtained by changing the 3rd input argument. It is clear from Figure 4.4 that this series has a single peak, the frequency of which gradually changes over time, not surprising since these simulated data were deliberately generated in such a form to illustrate the methodology.

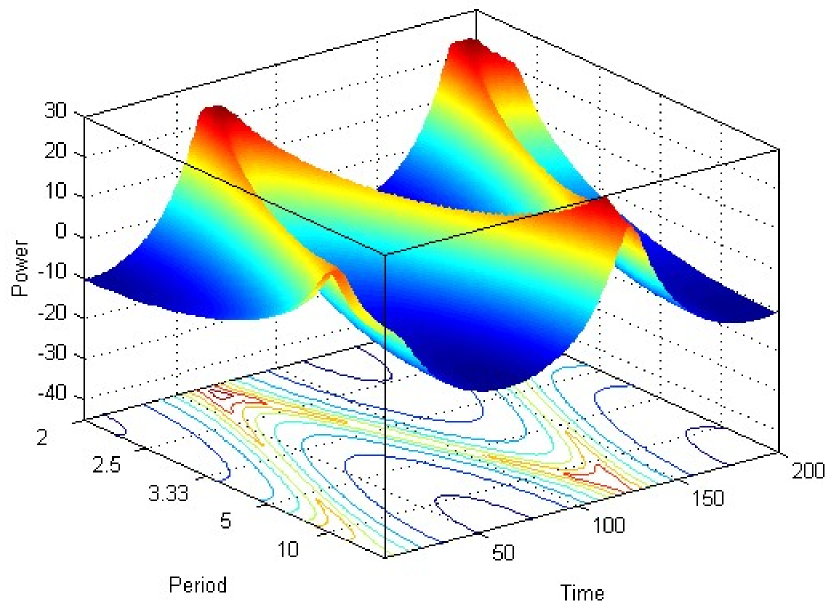


Figure 4.4 Time-frequency spectra.

Young (1998b) applies DAR analysis to the well known SPECtral MAPping series (SPECMAP: see Imbrie *et al*, 1992), which has been obtained from the analysis of oxygen isotope variations in deep ocean cores. Here, the spectral properties of the series are examined, revealing some localised variations in the estimated peak frequencies over time and a significant change at around 650-700 ka. Such analysis is useful in exposing time series data to greater scrutiny and 'focussing-in' on interesting aspects of the data which would not be nearly so apparent from the results of more conventional time series analysis.

4.4 Dynamic AutoRegressive eXogenous Variables (DARX)

The DARX model is simply the extension of the DAR model (4.6) to include measured exogenous or input time series that are thought to affect y_t in a truly dynamic, systems sense. In the case of a single input variable u_t, it takes the form,

$$y_t = -a_{1t}y_{t-1} - a_{2t}y_{t-2} - \cdots - a_{nt}y_{t-n} + b_{0t}u_{t-\delta} + b_{1t}u_{t-\delta-1} + \cdots + b_{mt}u_{t-\delta-m} + e_t \qquad (4.10)$$

where δ is a pure time delay, measured in sampling intervals, which is introduced to allow for any temporal delay that may occur between the incidence of a change in u_t and its first effect on y_t. Such 'transport delays' are, of course, a common feature of many environmental and engineering systems. This DARX model is, in fact, a special example of the discrete-time Transfer Function (TF) model, which is considered in more detail in the RIVSID module of CAPTAIN. This becomes apparent if it is written in the following L operator form,

$$y_t = \frac{B(L,t)}{A(L,t)}u_{t-\delta} + \frac{1}{A(L,t)}e_t \qquad (4.11)$$

where $B(L,t) = b_{0t} + b_{1t}L + b_{2t}L^2 + \cdots + b_{mt}L^m$. It is a special model because it assumes that the white noise input enters through the TF (or filter) $1/A(L,t)$, so avoiding certain difficult statistical problems that beset more general TF models (see RIVSID module and e.g. Chapter 8 of Taylor *et al.* (2013)).

The close relationship between the DAR and DARX models means that the hyper-parameter optimisation and recursive estimation of the TVP's is identical to the earlier examples in this chapter. Indeed, the DAR model may be estimated directly using the CAPTAIN function **dlr**, by manually specifying the regressors as appropriate past values of the input and output variables. This approach provides the greatest freedom for the user to also specify a trend and/or other components, as shown in Example 4.3 below. However, this solution may not be optimal; for example, if there are any missing data the same problems discussed in Section 4.3 above apply. Therefore, special shells are included for the estimation of a purely DAR model (4.11), namely **darx** and **darxopt**. Use of these functions is demonstrated later in Example 4.4.

Example 4.3 River Cam Data Revisited (Young, 1998b)

Given the DLR results for the Cam data (Example 4.1) and initial evaluation of different DARX model structures, the best identified DARX model is (Young, 1998b),

$$y_t = -a_{1t} y_{t-1} + b_{0t} u_t + T_t + e_t \qquad t = 1,2,...,81 \tag{4.12}$$

where u_{t-1} has been replaced by u_t because the analysis suggests strongly that the 'dynamic lag' effect introduced by the lagged term in y_{t-1} effectively removes the need for the pure time delay δ in this case. Under the assumption that the trend T_t and a_{1t} evolve as RW processes, while b_{1t} varies as an IRW processes, the associated NVR coefficients are optimised by ML, as shown below,

```
>> load cam.dat
>> u = cam(:, 1);  % sunlight (hours/day)
>> y = cam(:, 2);  % DO (mg/l)
>> z = [del(y, 1) ones(size(y)) u];  % define regressors
>> nvr = dlropt(y, z, [0 0 1]);
nvr =
  1.0e-006 *
    0.0000
    0.0012
    0.5094
>> [fit, fitse, par, parse] = dlr(y, z, [0 0 1], nvr);
>> par(end, :)
ans =
    0.7186    1.6557    0.0679
>> parse(end, :)
ans =
    0.0757    0.5275    0.0309
>> r2 = 1 - cov(y-fit)./cov(y)
r2 =
   0.7433
```

Initially, all three parameters were assumed to vary as IRW processes but the FIS estimation results then suggested strongly that T_t and a_{1t} were not varying significantly, so that more appropriate stochastic model in both cases was the RW process. In fact, as we see from the above optimised NVR values, which are insignificantly different from zero in the case of T_t and a_{1t}, the ML optimisation is indicating that these parameters are stationary. Indeed, the resulting estimates, using these optimised NVR's, are virtually constant in both cases, with $\hat{a}_1 = -0.719 \pm 0.076$ and $\hat{T} = 1.656 \pm 0.528$.

The estimate of b_{0t}, shown in Figure 4.5 is also not estimated as changing very much and it is difficult, at this point in the analysis, to say whether the variation is significant in comparison with the fully constant parameter ARX alternative, whose constant parameters are estimated as $\hat{a}_1 = -0.746 \pm 0.066$; $\hat{b}_0 = 0.064 \pm 0.017$; and $\hat{T} = 1.478 \pm 0.465$. These latter estimates are obtained by setting the third and fourth input arguments to **dlr** to the default zero, as shown below,

```
>> [fit, fitse, parc, parse] = dlr(y, z);
>> parc(end, :)
ans =
    0.7455    1.4784    0.0641
```

The output of the DLR model with time varying parameters (the output argument **fit**), as computed from the equation,

$$\hat{y}_{t|N} = -a_{1,t|N} y_{t-1} + b_{0,t|N} u_t + T_{t|N} \qquad t = 1,2,\ldots,81 \tag{4.13}$$

is compared with the DO data (circles) in the top graph of Figure 4.5.

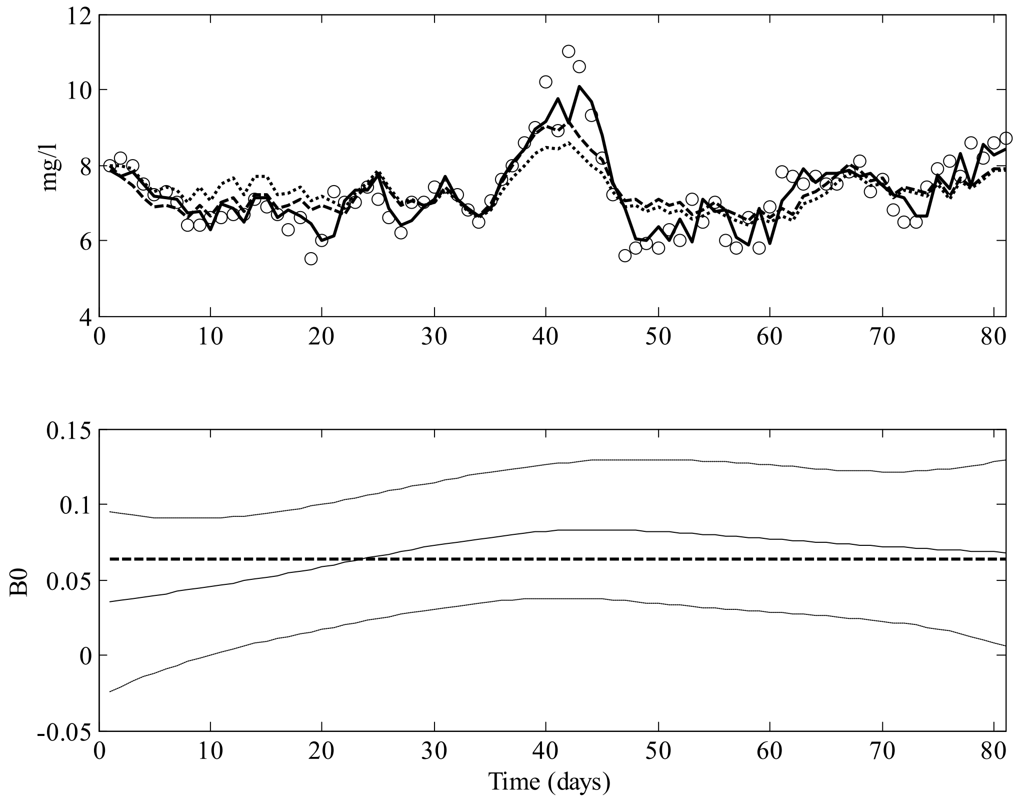


Figure 4.5 DLR analysis of River Cam data. Top: 1-step ahead predictions (full trace) with DO data (circles). The simulated model output (dashed) and constant parameter (dotted) model outputs are also shown for comparison. Bottom: time varying $\hat{b}_{0t}$ and standard errors, together with the constant parameter equivalent $\hat{b}_{0t} = 0.0641$ (dashed).

The DARX model $R^2 = 0.743$ can be compared with the initial DLR model (Example 4.1) $R^2 = 0.668$, and the constant parameter ARX model of $R^2 = 0.727$, which is only a little smaller. The statistical diagnostics are more satisfactory than the initial DLR model: the ACF and PACF of the normalised recursive residuals show no significant lag correlation and are consistent with the white noise assumption. However, the CCF between the residuals and the sunlight series shows some minor instantaneous correlation.

Since the coefficients of determination for the TVP and constant parameter models are so similar, it would appear at first sight that little is being gained by allowing for time variable parameters in this case. However, there is an important complicating factor that needs to be considered: the output $\hat{y}_{t|N}$ of the model in equation (4.13) represents only the *one-step-ahead predictions* of the DLR model, since y_{t-1} on the right had side of the equation is the last measured value of the DO. Consequently, the R^2 values relate to the one-step-ahead prediction errors, which is the normal definition of the coefficient of determination for regression-type models. Unfortunately, in the case of transfer function models of the DARX and ARX type, this measure can often provide a somewhat overly optimistic indication of the model's explanatory ability, which may be the reason why it is so often quoted in the modelling literature!

A much more discerning and critical measure of the model's ability to characterise the data is the coefficient of determination based on the simulation model residuals, R_T^2, in which the simulation model output $\hat{y}_{t|N}^s$ is generated from the equation,

$$\hat{y}_{t|N}^s = -a_{1,t|N}\hat{y}_{t-1|N}^s + b_{0,t|N}u_t + T_{t|N} \qquad t = 1,2,...,81 y_t \tag{4.14}$$

where now the lagged output term on the right hand side of the equation is the last value of the *simulated* output $\hat{y}_{t-1|N}^s$, rather than the measured y_{t-1}. In other words, $\hat{y}_{t|N}^s$ is generated solely from the sunlight series u_t and the trend term $T_{t|N}$ (here estimated as a constant), without any reference at all to the measured DO, y_t, as shown below.

```
>> tf=y(1);  % initial condition
>> for ff=2:length(y)
>>   tf(ff, 1) = par(ff, 1)*tf(ff-1) + par(ff, 2) + par(ff, 3)*u(ff);
>> end
>> r2 = 1 - cov(y-tf)./cov(y)
r2 =
   0.6189
```

As expected, the $R_T^2 = 0.619$ value based on this simulation model residuals, $\varepsilon_t = y_t - \hat{y}_{t|N}^s$, is quite a lot less than the equivalent $R^2 = 0.743$ obtained from the one-step-ahead prediction errors $\hat{e}_t = y_t - \hat{y}_{t|N}$, but it is much better than the $R_T^2 = 0.478$ based on the simulation model with all constant parameters. The plot of $\hat{y}_{t-1|N}^s$ for the DARX model is shown as the dashed line in Figure 9 and this is clearly superior to the equivalent graph of the constant parameter ARX model output, shown as the dotted line, despite the fact that the $\hat{b}_{1,t|N}$ TVP estimate is changing very smoothly and by quite small amount.

So what can we conclude from the results here and in the previous DLR modelling exercise? In terms of the R^2 values, the DLR and DARX models are very similar. Both have a single time variable parameter, the coefficient associated with the sunlight series, although the estimated variations of the DARX parameter are much less and much smoother than in the DLR case. But the DARX model has an additional constant parameter, the coefficient associated with its additional regression variable, the lagged dependent variable y_{t-1}. And, finally, the DARX model has superior, albeit not perfect, statistical diagnostics.

Taking all these factors in to consideration, the DARX model seems to be marginally superior. First, its single TVP varies less and much more smoothly that the equivalent TVP in the DLR model, which is clearly advantageous (if a constant parameter model can be identified and estimated it is always preferable to a TVP model). Second, it is a dynamic model, in the systems sense, which seems more acceptable from a physico-biological standpoint and better satisfies the *Data-Based Mechanistic* (DBM) modelling strategy developed at Lancaster over the past few decades (see e.g. Price *et al.* 1999 or Young *et al.* 2001a, and the references within). In addition to providing an efficiently parameterized model that explains the data well, this strategy requires that, if at all possible, the model should be interpretable in physically meaningful terms. In this regard, the above modelling provides a straightforward, quick and objective method of analysis, which reveals that the potentially changing nature of the relationship between lagged sunlight and DO can be represented by very simple models with only one smoothly changing parameter.

Furthermore, having completed this initial analysis, the estimated variations in the parameters can be investigated further to see if they are associated with other measured variables (states or inputs) of the system. For example, we might wish to investigate whether b_{0t} is a function of temperature, as it might well be from physico-biological considerations. The CCF between the b_{0t} parameters from both the DLR and DARX models and water temperature reveals a quite marked correlation with a maxima of about 0.7 at lag zero. Moreover, if the water temperature data is smoothed using **irwsm** with $\mathrm{NVR} = 0.00002$, then there is a remarkable maximum instantaneous correlation of 0.997, as shown below,

```
>> w = cam(:, 3);  % river water temperature (degrees Centigrade)
>> w = irwsm(w, 1, 0.00002);
>> ccf(w, par(:, 3));
```

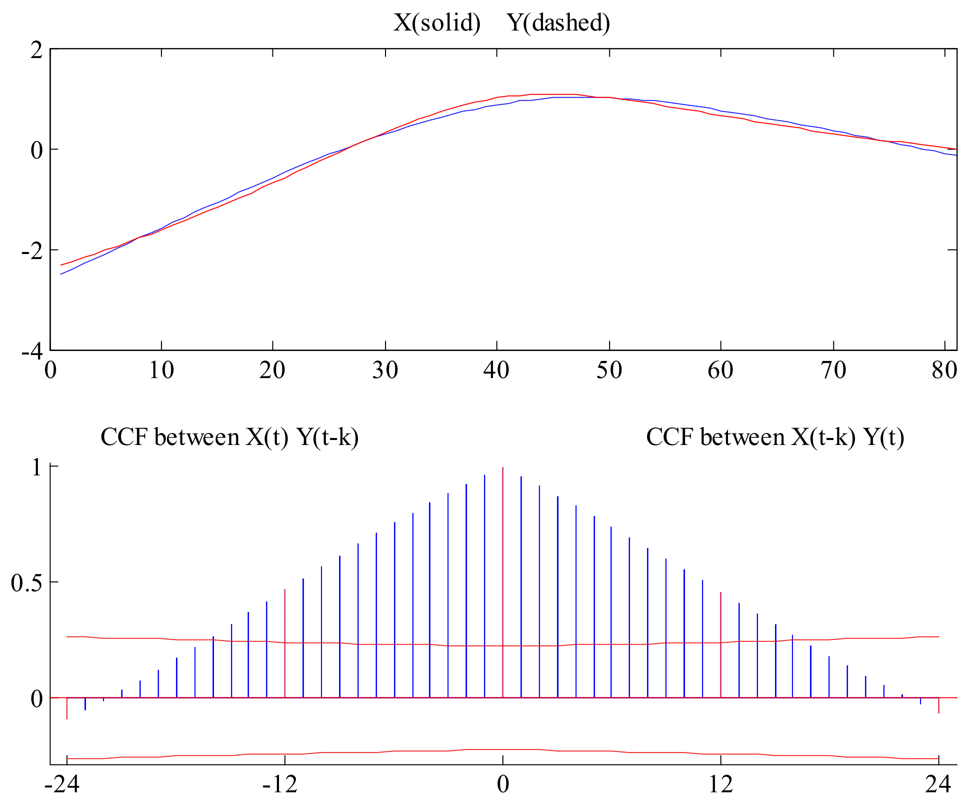


Figure 4.6 CCF between the DARX estimate $\hat{b}_{1t}$ (dashed) and smoothed water temperature (full).

While this certainly does not mean that there is a physical relationship between the TVP's and water temperature, it does show how the analysis can expose potential relationships and provide food for thought. In this case, it suggests that the b_{0t} parameter in both models *may* be a State Dependent Parameter (SDP), of the kind discussed in Chapter 5, and such state dependency is suggestive of a multiplicative nonlinearity of the bilinear kind.

Finally, by demonstrating the need for such a lagged, TVP or SDP relationship between the variables, it provides a useful prelude to further, more mechanistically oriented DBM modelling which considers the possibility of simple but nonlinear stochastic, dynamic relationships involving other relevant variables, such as: upstream measurements of DO; Biochemical Oxygen Demand (BOD) arising from pollution in the river, nutrient inputs and algal dynamics (see e.g. Beck and Young, 1975).

4.5 Dynamic Transfer Function (DTF)

Unfortunately, the DARX model (4.10) is limited in practical terms since it depends on the assumption of the, rather specific, signal topology, with the noise entering the model through a restricted AR process with a polynomial $A(L,t)$ equal to that of the denominator polynomial. A more general *Dynamic Transfer Function* (DTF) model, without the restrictions of the DARX, is the following,

$$y_t = \frac{B(L,t)}{A(L,t)} u_{t-\delta} + \xi_t \tag{4.15}$$

Here, $A(L,t)$ and $B(L,t)$ are time variable coefficient polynomials in L of the following form:

$$\begin{aligned} A(L,t) &= 1 + a_{1,t}L + a_{2,t}L^2 + \ldots + a_{n,t}L^n \\ B(L,t) &= b_{0,t} + b_{1,t}L + b_{2,t}L^2 + \ldots + b_{m,t}L^m \end{aligned} \tag{4.16}$$

and ξ_t represents uncertainty in the relationship arising from a combination of measurement noise, the effects of other unmeasured inputs and modelling error. Normally, ξ_t is assumed to be independent of u_t and is modelled as an AutoRegressive (AR) or AutoRegressive-Moving Average (ARMA) stochastic process (see e.g. Box and Jenkins, 1970; Young, 1984), although even this restriction can be avoided by the use of instrumental variable methods, as discussed below.

Equation (4.16) can be written in the following vector equation form,

$$y_t = \mathbf{z}_t^T \mathbf{p}_t + \eta_t \tag{4.17}$$

where,

$$\begin{aligned} \mathbf{z}_t^T &= \begin{bmatrix} -y_{t-1} & -y_{t-2} & . & . & . & -y_{t-n} & u_{t-\delta} & . & . & . & u_{t-\delta-m} \end{bmatrix} \\ \mathbf{p}_t &= \begin{bmatrix} a_{1,t} & a_{2,t} & . & . & . & a_{n,t} & b_{0,t} & . & . & . & b_{m,t} \end{bmatrix}^T \end{aligned} \tag{4.18}$$

and $\eta_t = A(L,t)\,\xi_t$. For convenience of notation, let $\mathbf{p}_t$ be defined as follows,

$$\mathbf{p}_t = \begin{bmatrix} p_{1,t} & p_{2,t} & . & . & . & p_{n+m+1,t} \end{bmatrix}^T \tag{4.19}$$

with $p_{i,t}, i = 1,2,\ldots,n+m+1$, relating to the TF model parameters $a_{i,t}$ and $b_{j,t}$ through (4.17). In order to estimate the assumed time variable model parameters in $\mathbf{p}_t$, it is necessary to make some assumptions about the nature of their temporal variability. As for the earlier examples in this chapter, the i^{th} parameter, $p_{i,t}, i = 1,2,\ldots,n+m+1$, in $\mathbf{p}_t$ is defined by a two dimensional stochastic state vector $\boldsymbol{x}_{i,t} = [l_{i,t} \quad d_{i,t}]^T$, where $l_{i,t}$ and $d_{i,t}$ are, respectively, the changing level and slope of the associated TVP. The stochastic evolution of each $\boldsymbol{x}_{i,t}$ (and, therefore, each of the n+m+1 parameters in $\mathbf{p}_t$) is assumed to be described by the GRW process defined in Chapter 2.

Having introduced the GRW models for the parameter variations, an overall SS model (2.1) can then be constructed straightforwardly by the aggregation of the subsystem matrices. As before, the white noise inputs $\eta_{i,t}$, which provide the stochastic stimulus for parametric change in the model, are assumed to be independent of the observation noise e_t and have a covariance matrix $\mathbf{Q}$ formed from the combination of the individual covariance matrices $\mathbf{Q}_{\eta,i}$. Finally, $\mathbf{H}_t$ is a $1 \times p$ vector of the following form,

$$\mathbf{H}_t = \left[-y_{t-1} \;\; 0 \;\; -y_{t-2} \;\; 0 \ldots . y_{t-n} \;\; 0 \;\; u_{t-\delta} \;\; 0 \ldots . u_{t-\delta-m} \;\; 0 \right]$$

that relates the scalar observation y_t to the state variables, so that it represents the DTF model (4.12) with each parameter defined as a GRW process. In the case of the scalar RW and AR(1) models, the alternate zeros are simply omitted.

In the previous sections of this chapter, a standard algorithmic approach to the problem has been utilized based on a forward-pass filtering algorithm, followed by fixed interval smoothing. In this regard, it should be noted that the recursive filtering algorithm is closely related to the Kalman Filter (KF: 1960) and is often referred to as such. The difference is that the $\mathbf{H}_t$ matrix in the present, recursive TVP estimation context for the TF model (4.12), is based on measured variables. In particular, the output variables $y_{t-i}, i = 1, 2, \ldots, n$, in $\mathbf{H}_t$ are affected by the noise ξ_t (the 'errors-in-variables' problem); whereas, strictly, in the KF, $\mathbf{H}_t$ has to be composed of exactly known (but, if necessary, time variable) deterministic coefficients.

This difference is important in the present TF context since it can be shown that the TVP estimates obtained from the standard recursive filtering/smoothing algorithm will be asymptotically biased away from their 'true' values. This bias may be unimportant if the model is to be used within a forecasting context since the forecasts produced by the model are not biased (although they may not be statistically efficient). However, the level of the bias is dependent on the magnitude of the measurement noise and it can be problematic in high noise situations, particularly if the parameters are physically meaningful (see e.g. Young, 1984 for a discussion of this problem in the constant parameter situation).

For this reason, it is necessary to modify the standard algorithm (2.5, 2.6) to avoid these biasing problems. The approach taken is similar to that used the RIVSID module of CAPTAIN for the identification of general transfer function models and requires the introduction of *instrumental variables*. In relation to the time series $y_t, t = 1, 2, \ldots, N$, the time variable parameter recursive Instrumental Variable (IV) filtering/smoothing algorithm has the following form:

1. Forward Pass Symmetric IV Equations (iterative)

Prediction:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}\hat{\mathbf{x}}_{t-1}$$
$$\hat{\mathbf{P}}_{t|t-1} = \mathbf{F}\hat{\mathbf{P}}_{t-1}\mathbf{F}^T + \mathbf{G}\mathbf{Q}_r\mathbf{G}^T \tag{4.21}$$

Correction:

$$\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_{t|t-1} + \hat{\mathbf{P}}_{t|t-1}\hat{\mathbf{H}}_t^T\left[1 + \hat{\mathbf{H}}_t\hat{\mathbf{P}}_{t|t-1}\hat{\mathbf{H}}_t^T\right]^{-1}\{y_t - \mathbf{H}_t\hat{\mathbf{x}}_{t|t-1}\}$$
$$\hat{\mathbf{P}}_t = \hat{\mathbf{P}}_{t|t-1} - \hat{\mathbf{P}}_{t|t-1}\hat{\mathbf{H}}_t^T\left[1 + \hat{\mathbf{H}}_t\hat{\mathbf{P}}_{t|t-1}\hat{\mathbf{H}}_t^T\right]^{-1}\hat{\mathbf{H}}_t\hat{\mathbf{P}}_{t|t-1} \tag{4.22}$$

where,

$$\hat{\mathbf{H}}_t = \left[-\hat{x}_{t-1} \;\; 0 \;\; -\hat{x}_{t-2} \;\; 0 \ldots . \hat{x}_{t-n} \;\; 0 \;\; u_{t-\delta} \;\; 0 \ldots . u_{t-\delta-m} \;\; 0\right] \tag{4.23}$$

$$\hat{x}_t = \frac{\hat{B}_{j-1}(L,t)}{\hat{A}_{j-1}(L,t)}u_{t-\delta} \tag{4.24}$$

As before, the FIS algorithm is in the form of a backward recursion operating from the end of the sample set to the beginning.

2. Backward Pass Fixed Interval Smoothing IV Equations (FISIV: single pass)

$$\hat{\mathbf{x}}_{t|N} = \mathbf{F}^{-1}\left[\hat{\mathbf{x}}_{t+1|N} + \mathbf{G}\mathbf{Q}_r\mathbf{G}^T\mathbf{L}_t\right]$$
$$\mathbf{L}_t = \left[\mathbf{I} - \hat{\mathbf{P}}_{t+1}\hat{\mathbf{H}}_{t+1}^T\hat{\mathbf{H}}_{t+1}\right]^T\left[\mathbf{F}^T\mathbf{L}_{t+1} - \hat{\mathbf{H}}_{t+1}^T\{y_{t+1} - \hat{\mathbf{H}}_{t+1}\hat{\mathbf{x}}_{t+1}\}\right] \tag{4.25}$$
$$\hat{\mathbf{P}}_{t|N} = \hat{\mathbf{P}}_t + \hat{\mathbf{P}}_t\mathbf{F}^T\hat{\mathbf{P}}_{t+1|t}^{-1}\left[\hat{\mathbf{P}}_{t+1|N} - \hat{\mathbf{P}}_{t+1|t}\right]\hat{\mathbf{P}}_{t+1|t}^{-1}\mathbf{F}\hat{\mathbf{P}}_t$$

with $\boldsymbol{L}_N = \boldsymbol{0}$.

The main difference between the above algorithm (4.23)-(4.25) and the standard filtering/smoothing algorithms is the introduction of 'hats' on the $\hat{\mathbf{H}}$ vector and the $\hat{\mathbf{P}}$ matrix. $\hat{\mathbf{H}}_t$ in (4.15) is the IV vector, which is used by the algorithm in the generation of all the $\hat{\mathbf{P}}_t$ terms and is the main vehicle in removing the bias from the TVP estimates. The subscript $j-1$ on $\hat{A}_{j-1}(L,t)$ and $\hat{B}_{j-1}(L,t)$ indicates that the estimated DTF polynomials in the auxiliary model, (4.15), which generates the instrumental variables $\hat{x}_t$ that appear in the definition of $\hat{\mathbf{H}}_t$, are updated in an iterative manner, starting with the least squares estimates of these polynomials.

Iteration is continued until the forward pass (filtered) IV estimates of the TVP's are no longer changing significantly: normally only 3 iterations are required.

This iterative approach is based on the IV algorithm for constant parameter TF models (e.g. Young, 1984), except that the symmetric gain version of the IV algorithm (Young, 1970; 1984, p.183) is used, rather than the more usual asymmetric version. This is necessary in order that the standard recursive FIS algorithm can be used to generate the smoothed estimates of the TVP's. Note also that, in these algorithms, the NVR matrix $\mathbf{Q}_r$ is defined by equation (2.10) as usual. The optimization of these hyper-parameters is achieved through either Maximum Likelihood estimation or the minimisation of the *n*-step ahead forecasting errors as discussed before.

These modified filtering and smoothing algorithms mean that the standard **dlr** function cannot be utilised to estimate the TF model (4.15). Instead, special shells are included in CAPTAIN, namely **dtfm** and **dtfmopt**, which require the user to specify only the structure of the model, as shown below.

Example 4.4 Comparison of DARX and DTFM for simulated data

As a straightforward example of DTF analysis, consider the estimation of the parameters in the following first order TVP model with gradually changing denominator parameter, a fixed numerator parameter and a time delay of two samples.

$$y_t = \frac{0.5}{1 + a_{1t}L} u_{t-2} + e_t \tag{4.25}$$

where y_t represents the output, u_t the input and e_t a zero mean white noise signal. For this example, we allow the denominator parameter to change slowly over time as a sine wave, as illustrated in Figure 4.7. The MATLAB® code for this example is shown below, where we initially assume IRW models for both parameters.

```
>> load tvp3.dat
>> y=tvp3(:, 1);  % output
>> u= tvp3 (:, 2);  % input
>> nvr=dtfmopt(y, u, [1 1 2], 1)
nvr =
  1.0e-007 *
    0.2951
    0.0000
```

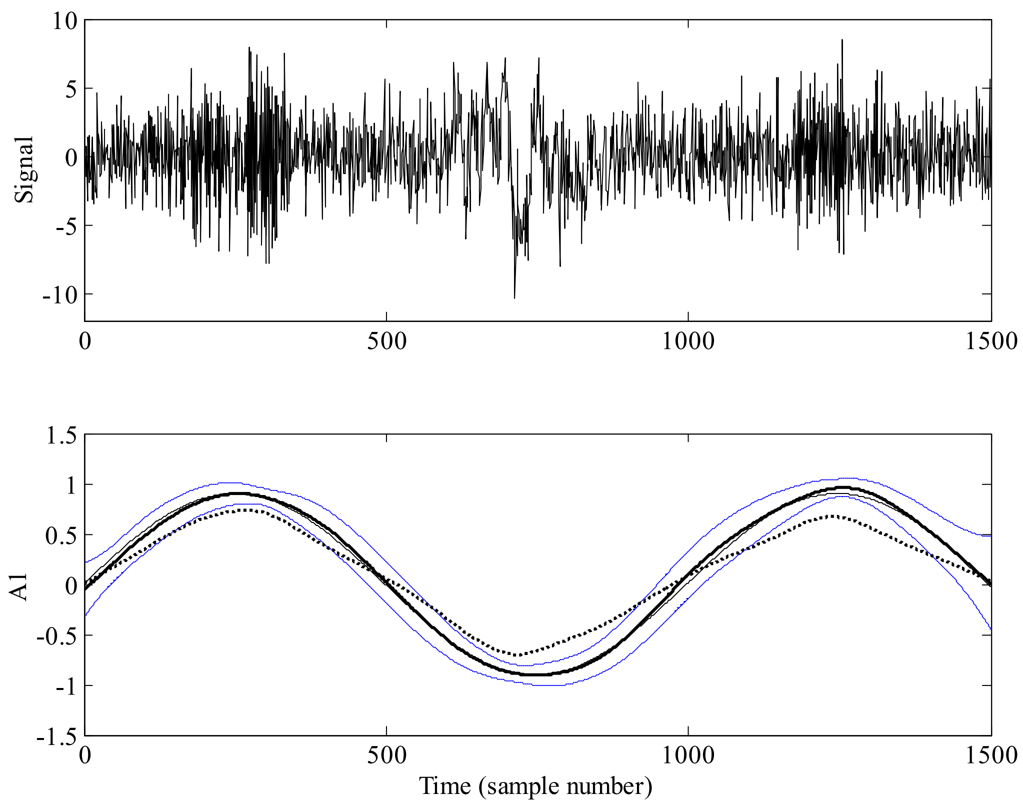


Figure 4.7 Top: output y_t; bottom: parameter (thin); DTF estimate (solid) and standard errors; for comparison, the DARX estimate is also shown (dotted).

ML optimization yields a NVR for the $b_{0,t}$ parameter that is insignificantly different from zero, indicating that the parameter is identified as being time invariant. This shows how, quite objectively, the ML optimization is able to identify the relative temporal variability of the model parameters from the input-output data. Continuing the analysis with time invariant $b_{0,t}$ now selected *a priori*,

```
>> nvr=dtfmopt(y, u, [1 1 2], [1 0], [], [-2 0]);
>> [tfs1, fit1, fitse1, par1, parse1]=dtfm(y, u, [1 1 2], [1 0], nvr);
>> [tfs2, fit2, fitse2, par2, parse2]=darx(y, u, [1 1 2], [1 0], nvr);
```

The final line above estimates the equivalent model using the DARX noise assumption, i.e. instrumental variables are not utilised in the filtering and smoothing algorithm. The third input argument in the calls to **dtfm**, **darx** and **dtfmopt** is the model structure (4.35), represented by the triad [n, m, δ]. Of course, this example assumes that the model structure is known prior to the analysis. In practice, the identification procedure could involve testing a range of potential model structures until the most appropriate one is found. Alternatively, the identification tools available in the RIVSID module of CAPTAIN may be utilised.

The dotted trace in the lower plot of Figure 4.7 shows the equivalent DARX estimates of a_{1t} using the same NVRs. The superiority of the DTF estimates is clear. The DTF model with these estimated parameters explains the data well: the coefficient of determination based (on the rather noisy) simulated model output compared with the noise free output, $\mathrm{R}_{\mathrm{T}}^{2} = 0.6075$; whilst for the DARX model, this is reduced to $\mathrm{R}_{\mathrm{T}}^{2} = 0.4705$. The model residuals (innovations) for the DTF model are also superior: they have an approximately normal amplitude distribution; and, as required, both the ACF and the CCF between the residuals and the input u_t, are insignificant at all lags. In contrast, the CCF for the DARX model residuals shows significant correlation with u_t at some lags.

4.6 Conclusions

The present chapter has briefly summarised the entire class of TVP or 'dynamic', regression models implemented in CAPTAIN, including Dynamic Linear Regression (DLR), Dynamic Harmonic Regression (DHR) and Dynamic Auto-Regression (DAR), as well as the closely related, TVP version of the Auto-Regressive eXogenous variables model (DARX) and an alternative Dynamic Transfer Function (DTF) model, estimated using an instrumental variable method of fixed interval smoothing.

As pointed out in the introduction to this chapter, such TVP models allow for the estimation of any slow parameter variations that may result from slow physical changes in the process or from some form of nonlinearity in the data. However, when the parameters vary at a rate commensurate with that of the system variables themselves then the model may behave in a heavily nonlinear or even chaotic manner. Nonetheless, if these TVP's are found to be functions of the state or input variables (i.e. they actually constitute stochastic state variables), then CAPTAIN provides for the estimation of State Dependent Parameter (SDP) models, as discussed in the next Chapter 5.

CHAPTER 5
STATE DEPENDENT PARAMETER MODELS

The idea of using *State-Dependent Parameter* (SDP) models to represent nonlinear dynamic systems goes back to Young (1978), who showed how the forced logistic growth equation could be represented, identified and estimated in SDP form. However, the practical development of these ideas is of a more recent origin (Young, 1993b, 1998a,b, 2000, 2001a,b; Young *et al.*, 2001). The associated **sdp** tool in CAPTAIN has been available since 2001 (Taylor *et al.* 2007).

5.1 The State-Dependent ARX (SDARX) Model

In order to introduce the ideas that underlie SDP models, consider first the *Dynamic ARX* (DARX) model introduced in Chapter 4 which, for the case of a single input variable, is written in the following form,

$$y_t = -a_{1t}y_{t-1} - a_{2t}y_{t-2} - \cdots - a_{nt}y_{t-n} + b_{0t}u_{t-\delta} + b_{1t}u_{t-\delta-1} + \cdots + b_{mt}u_{t-\delta-m} + e_t \qquad (5.1a)$$

or, in transfer function terms,

$$y_t = \frac{B(L,t)}{A(L,t)}u_{t-\delta} + \frac{1}{A(L,t)}e_t \qquad (5.1b)$$

Equation (5.1) is based on a nomenclature favoured by econometricians and used throughout most of the present book. However, earlier publications concerned with SDP models have utilised the following nomenclature for equation (5.1), as used by systems and control analysts,

$$y_k = \frac{B_k(z^{-1})}{A_k(z^{-1})}u_{k-\delta} + \frac{1}{A_k(z^{-1})}e_k \qquad (5.2a)$$

For consistency with these numerous earlier publications, the present chapter will utilise this alternative nomenclature. Here z^{-i}, rather than L^i is used as the backward shift operator and the subscript k is used to denote that the associated variable is sampled at a kth sampling instant: i.e. $z^{-i}y_k = y_{k-i}$; δ is still used to denote a pure time delay; and e_k is a zero mean, white noise signal. In this form, $A_k(z^{-1})$, $B_k(z^{-1})$ are the following TVP polynomials in the backward shift operator z^{-1} ,

$$\begin{aligned} A_k(z^{-1}) &= 1 + a_{1,k}z^{-1} + \ldots + a_{n,k}z^{-n} \\ B_k(z^{-1}) &= b_{1,k}z^{-1} + \ldots + b_{m,k}z^{-m} \end{aligned} \tag{5.2b}$$

The polynomial coefficients, $a_{i,k}, i = 1,2,\ldots,n$ and $b_{j,k}, j = 1,2,\ldots,m$ may vary between samples k to k+1, and the stochastic evolution of each parameter is assumed to be described by the Generalised Random Walk (GRW) process introduced in Chapter 2, including RW, AR(1), IRW, SRW, LLT and damped trends as particular cases. As discussed previously, the AR(1), SRW and damped trend models all require the specification or optimisation of an additional 'hyper-parameter', α , although this is rarely necessary in practice. Also, as before in Chapter 4, the model structure is defined by the triad [n, m, δ] (see Example 4.4).

The SDP version of the model (5.2a) is made explicit by writing the definitions of (5.2b) in the following SDP form:

$$\begin{aligned} A_k(z^{-1}) &= A(\chi_k, z^{-1}) = 1 + a_1(\chi_k)z^{-1} + \ldots + a_n(\chi_k)z^{-n} \\ B_k(z^{-1}) &= B(\chi_k, z^{-1}) = b_0(\chi_k) + b_1(\chi_k)z^{-1} + \ldots + b_m(\chi_k)z^{-m} \end{aligned} \tag{5.2c}$$

where the notation $a_i(\chi_k), i = 1,2,\ldots,n; b_j(\chi_k), j = 0,2,\ldots,m$ indicates that the parameters are nonlinear functions of the vector χ_k. In general, χ_k is defined in terms of any variables on which the parameters are identified to be dependent. In the present context, however, each SDP is assumed to be a nonlinear function of a *single* variable which might, for instance, be its associated past input or output variable, i.e.,

$$\begin{aligned} A(y_{k-i}, z^{-1}) &= 1 + a_1(y_{k-1})z^{-1} + \ldots + a_n(y_{k-n})z^{-n} \\ B(u_{k-j}, z^{-1}) &= b_1(u_{k-1})z^{-1} + \ldots + b_m(u_{k-m})z^{-m} \end{aligned} \tag{5.2d}$$

However, in general, the SDARX model can be written in the equation form of (5.1a) but with the time variable parameters defined explicitly as a function of user-specified variables $x_{i.k}, i = 1,2,\ldots,n + m + 1$, i.e.,

$$\begin{aligned} y_t = -a_1(x_{1,k})y_{t-1} - a_2(x_{2,k})y_{t-2} \cdots - a_n(x_{n,k})y_{k-n} \\ + b_0(x_{n+1,k})u_{t-\delta} + b_1(x_{n+2,k})u_{t-\delta-1} + \cdots + b_m(x_{n+m+1,k})u_{t-\delta-m} + e_t \end{aligned} \tag{5.2e}$$

Young (2000, 2001a) and Young *et al.* (2001) describe an identification and estimation strategy for SDP models of this general type. The details of this strategy are given in these references and it will suffice here to outline the main features of the approach.

Initial non-parametric estimation of the SDARX model

The identification and estimation of a model such as (5.2) follows a similar procedure to that discussed in previous chapters for TVP models: after all, a SDP is also a TVP. Indeed, if the variable $x_{i,k}$ is only slowly changing in relation to the changes in the input and output variables y_k and u_k, then it can be treated as a TVP model and estimated in the same manner as the DARX model considered in Chapter 4.

However, consider the situation where the input u_k and output y_k of the dynamic system are changing rapidly and the variable $x_{i,k}$ is changing at a rate that is commensurate with the changes in these variables (e.g. it could be a function of the input and output variables, as in the definition (5.2d) above). Under these conditions, normal TVP estimation will fail because the GRW models for the parameters will be unable to effectively track the very rapid variations in the SDPs. For instance, as we shall see, the SDP model can describe a chaotic process, in which case the SDPs could also be chaotic! In order to obviate these difficulties, it is necessary to perform the TVP estimation in a different manner, in which the data are 're-ordered' prior to estimation and the recursive FIS algorithm is applied using a special 'back-fitting' procedure.

The data re-ordering is a simple but very effective device for transforming the rapid TVP estimation into a much simpler and solvable, slow TVP estimation problem. It works on the basis that *if, at any sample time k in an off-line (non-real-time) situation, all the variables in an equation such as (5.2) are available for the purposes of estimation, then it is not necessary to consider each equation in the normal temporal order*, $k = 1,2,...N$. For instance, each equation and the variables appearing in this equation, can be re-ordered in some manner and the model parameters in the equation can then be recursively updated in this new, transformed data space. And if the re-ordering is chosen such that, in this transformed data space, the variables and associated parameters are changing quite slowly, then recursive FIS estimation, based on the GRW class of models for the parameter variations, will provide sensible estimates of the parameter variations *in the transformed data space*. Transformation of these estimated SDPs back into the original data space then reveals their true rapid variation in natural temporal terms.

In order to illustrate the nature of this re-ordering procedure, consider first a simple example in the form of the following *State Dependent AR* (SDAR) model:

$$y_k = a(y_{k-1}).y_{k-1} + e_k; \qquad a(y_{k-1}) = 4 - 4y_{k-1} \tag{5.3}$$

This is the chaotic version of the logistic growth equation and so the state dependency of the parameter induces rapid, chaotic changes in $a(y_{k-1})$ that are clearly not identifiable using standard TVP estimation. However, if the data are re-ordered in the ascending order (using the MATLAB® function **sort**) of the dependent state y_{k-1}, then the variations of $a(y_{k-1})$ in this re-ordered data space have the same degree of smoothness as the re-ordered y_{k-1}. This is shown in Figure 5.1, where the upper plot shows 200 samples of y_{k-1} in natural temporal order; while the lower plot shows y_{k-1} sorted in ascending order of magnitude. Table 1 compares the first ten samples of y_{k-1} (second column) with the first ten samples of y^o_{k-1}, the re-ordered y_{k-1} series, in the fourth column. The sampling index of y^o_{k-1} in the normal temporal order is shown in the third column.

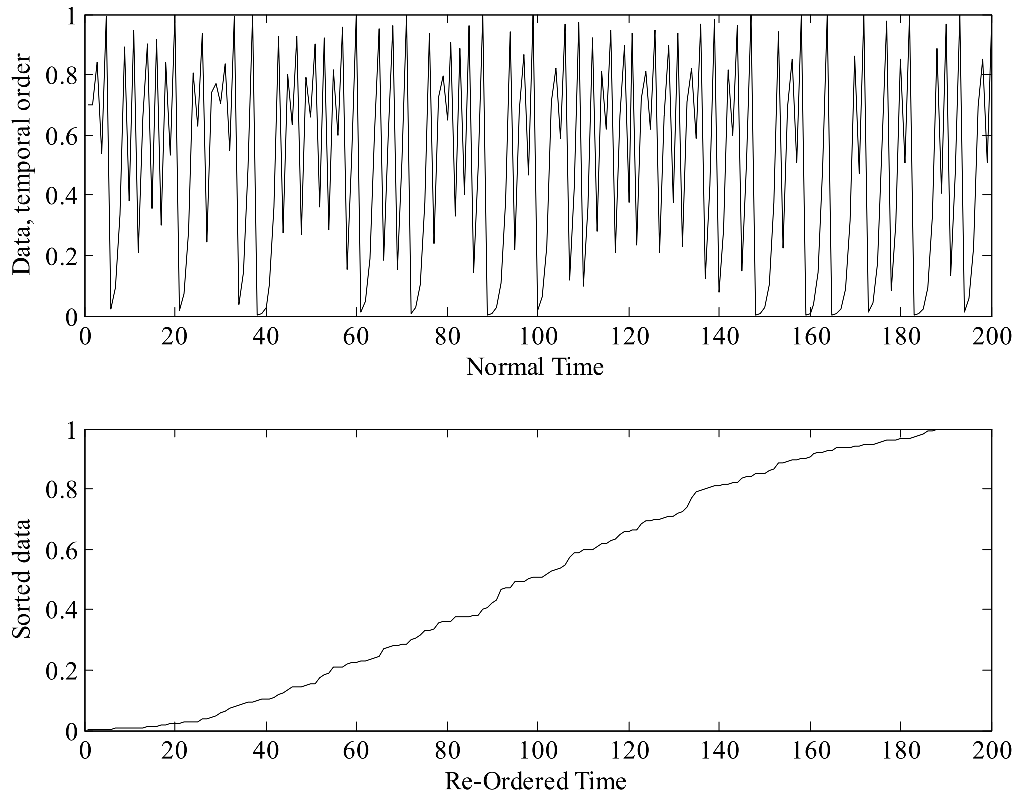


Figure 5.1 Chaotic example: model output in normal time (upper panel); model output re-ordered in ascending order of magnitude (lower panel).

k	y_{k-1}	k	y^o_{k-1}
1	0.7	47	0.0009
2	0.7	26	0.0019
3	0.8398	116	0.0027
4	0.5381	48	0.0036
5	0.9940	147	0.0038
6	0.0241	27	0.0077
7	0.0943	117	0.0106
8	0.3415	155	0.0117
9	0.8994	167	0.0120
10	0.3616	49	0.0144

Table 5.1 Example of sorted data.

In the case of this simple example, TVP estimation based on the re-ordered data provides information on the nature of the parametric state dependency. But what if there is more than one term on the right hand side of the model equation? How do we sort the data in this case if the associated parameters are dependent on different variables? This is where the back-fitting procedure comes into the analysis. To clarify this back-fitting procedure, consider the following, first order example of the SDARX model (5.2e),

$$y_k = -a_1(x_{1,k})y_{k-1} + b_0(x_{2,k})u_k + e_k \qquad k = 1,2,...,N \tag{5.4}$$

where the time delay δ has been removed for simplicity.

Backfitting Algorithm for the Model (5.4)

- Assume that, without any sorting, FIS estimation has yielded prior TVP estimates $\hat{a}^1_{1,k|N}$ and $\hat{b}^1_{0,k|N}$ of $a_1(x_{1,k})$ and $b_0(x_{2,k})$, respectively[1]. An SDP estimation equation for $a_{1,k} = a_1(x_{1,k})$ can then be formulated as,

$$[y_k - \hat{b}^1_{0,k|N}u_k]^{sy} = -a^{sy}_{1,k}.y_{k-1}{}^{sy} \tag{5.5}$$

  where the term on the left hand side can be considered as a *modified dependent variable* and the superscript *sy* denotes that all the variables are sorted in the ascending order of y_{k-1}. Application of the standard TVP algorithm to this single SDP sub-model then yields the FIS estimate $\hat{a}^{sx1}_{1,k|N}$ of $a^{sx1}_{1,k}$.

- $\hat{a}^{sy}_{1,k|N}$ is then 'unsorted' so that an SDP estimation equation for $b_{0,k} = b_0(x_{2,k})$ can be formulated as,

$$[y_k + \hat{a}_{1,k|N}y_{k-1}]^{su} = b^{su}_{0,k}.u^{su}_k \tag{5.6}$$

  with the superscript *su* denoting that all the variables are sorted in the ascending order of u_k. Application of the standard TVP algorithm to this single SDP sub-model then yields the FIS estimate $\hat{b}^{su}_{0,k|N}$ and the first iteration of the 'backfitting' algorithm is complete.

- This process is continued in an iterative manner (each time unsorting, forming the modified dependent variable, and sorting according to the current right hand side variable, prior to TVP estimation using the FIS algorithm), until the FIS estimates of the SDP's $\hat{a}_{1,k|N} = \hat{a}_1(x_{1,k})\{k\,|\,N\}$ and $\hat{b}_{0,k|N} = \hat{b}_0(x_{2,k})\{k\,|\,N\}$ (which are each time-series of length *N*) do not change significantly between iterations. Here, the

[1] The **sdp** tool in CAPTAIN uses the *constant* least squares parameter estimates, since the convergence of the backfitting procedure is not too sensitive to the prior estimates, provided they are reasonable.

nomenclature $\{k \mid N\}$ indicates the FIS estimate at sample k given the whole data set of N samples.

- The smoothing hyper-parameters required for FIS estimation at each iteration are optimized by Maximum Likelihood (ML), as discussed in Chapter 2. Such ML optimization can be carried out in various ways: after every complete iteration until convergence; only at the initial iteration, with the hyper-parameters maintained at these values for the rest of the backfitting; or just on the first two iterations. The latter seems most satisfactory in general practice, since very little change in the optimised NVR values or improvement in convergence occurs if optimization is continued after this stage. Normally, convergence is completed after only a few iterations. However, care must be taken to ensure that the convergence is completely satisfactory in each example, since a large number of iterations are sometimes required (see Young, 2001a).

Example 5.1 Analysis of a simulated SDARX model

This example utilizes data generated from the following 1st order SDARX relationship that is similar to equation (5.4) but with each parameter a function of its associated variable, i.e. with $x_{1,k} = y_{k-1}$ and $x_{2,k} = u_k$,

$$y_k = a_1(y_{k-1}).y_{k-1} + b_0(u_k).u_k + e_k \qquad e_k = N(0,0.000025) \qquad u_t = N(0,0.0064) \quad (5.7)$$

Here, the functions $a_1(y_{k-1})$ and $b_0(u_k)$ are defined as follows,

$$a_1(y_{k-1}) = 2.0(1 - y_{k-1})\,; \quad b_0(u_k) = 10u_k^{\,2} \quad (5.8)$$

When written in the nonlinear functional form,

$$y_k = f_1(y_{k-1}) + f_2(u_k) + e_k \quad (5.9)$$

or,

$$y_k = 2.0y_{k-1} - 2.0y_{k-1}^2 + 10u_k^3 + e_k \quad (5.10)$$

this is revealed as the SDP formulation of the non-chaotic logistic growth equation, with an input signal in the form of a normally distributed white noise sequence passed through a cubic law nonlinearity. A typical response of this system is shown in Figure 5.2, where the output y_k is in the top panel and the input u_k in the lower panel. Here, the percentage noise/signal, based on the standard deviations (i.e. $100\{\mathrm{sd}(e_k)/\mathrm{sd}(10u_k^3)\}$), is 28%. The MATLAB® code for this example, including both the simulation and SDP estimation is given below,

```
>> nn = 2000;                   % number of samples
>> e = 0.0053*randn(1, nn);     % measurement noise
>> u = 0.08*randn(nn, 1);       % input signal
>> y = zeros(nn, 1);
>> y(1) = 0.5;                  % initial condition
>> for i = 2:nn                 % simulation response with noise
>>   y(i) = 2.0*y(i-1)-2.0*y(i-1)*y(i-1)+10*u(i)*u(i)*u(i)+e(i);
>> end
>> yd = del(y, 1);              % output delayed by one sampling interval
>> z = [yd u];                  % states
>> x = [yd u];                  % regressors
>> nvrc = -2;
>> [fit, fitse, par, parse, zs, pars, parses, rsq, nvr] ...
                                      = sdp(y, z, x, [], nvrc);
```

The results of the SDARX analysis for a total sample size of $N = 2000$ are shown in the upper panels of Fig. 5.3. The hyper-parameter optimization is carried out at the first and second iterations (**nvr** specified as -2): thereafter, the NVR hyper-parameters are maintained constant, for the four iterations required to obtain good convergence in this case, at the following optimized values,

$$NVR\{a_1(y_{k-1})\} = 0.36 \quad ; \quad NVR\{b_0(u_k)\} = 1.0 \tag{5.11}$$

By default, **sdp** limits the maximum magnitude of each NVR to unity and, for the present example, this user adjustable constraint (see help information for **sdp**) has been reached for the second parameter. Clearly for a stochastic simulation (with noise) as here, the exact values obtained depend on the seed utilised in MATLAB® for the generation of the random signals. This caveat applies to all the numerical values given in the present section.

The upper panels of Figure 5.3 show the estimated SDPs obtained using these optimized NVR values, with $\hat{a}_1(y_{k-1})\{k \mid N\}$ in the left graph and $\hat{b}_0(u_k)\{k \mid N\}$ in the right. The thin traces are the actual SDP relationships, while the thick traces are the estimates. It is clear that the state dependency has been estimated well in both cases. The standard error (se) bounds are not plotted on these graphs but they are available as returned variables in the **parse** or **parses** matrices, corresponding to the unsorted (in normal temporal order) and sorted parameter estimates **par** and **pars** respectively. The other returned variables are **fit** and **fitse**, the output $\hat{y}_{k|k-1}$ of the SDARX model,

$$\hat{y}_{k|k-1} = \hat{a}_1(y_{k-1})\{k \mid N\}.y_{k-1} + \hat{b}_0(u_k)\{k \mid N\}.u_k \tag{5.12}$$

and its se bound, respectively; **zs** are the sorted variables on which the SDPs are dependent (in this case, the sorted y_{k-1} and u_k, respectively); **rsq** is the Coefficient Of Determination (COD), normally denoted by R^2, based on $\hat{y}_{k|k-1}$; and **nvre** are the optimised NVR values.

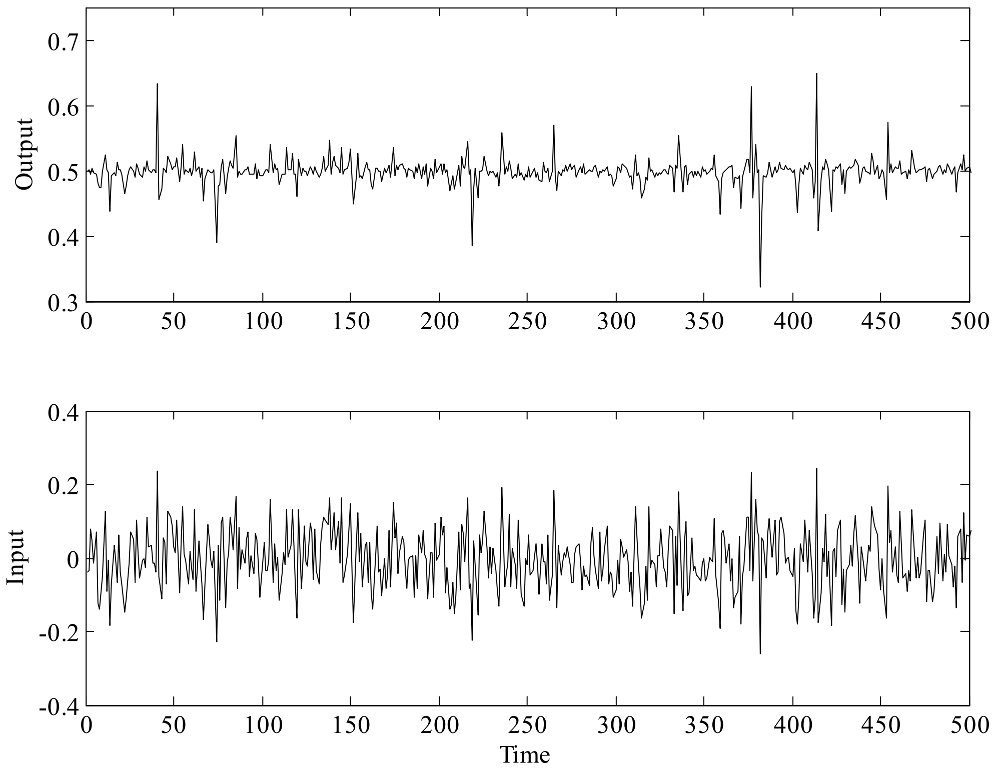


Figure 5.2 Input and output data for the SDARX simulation example.

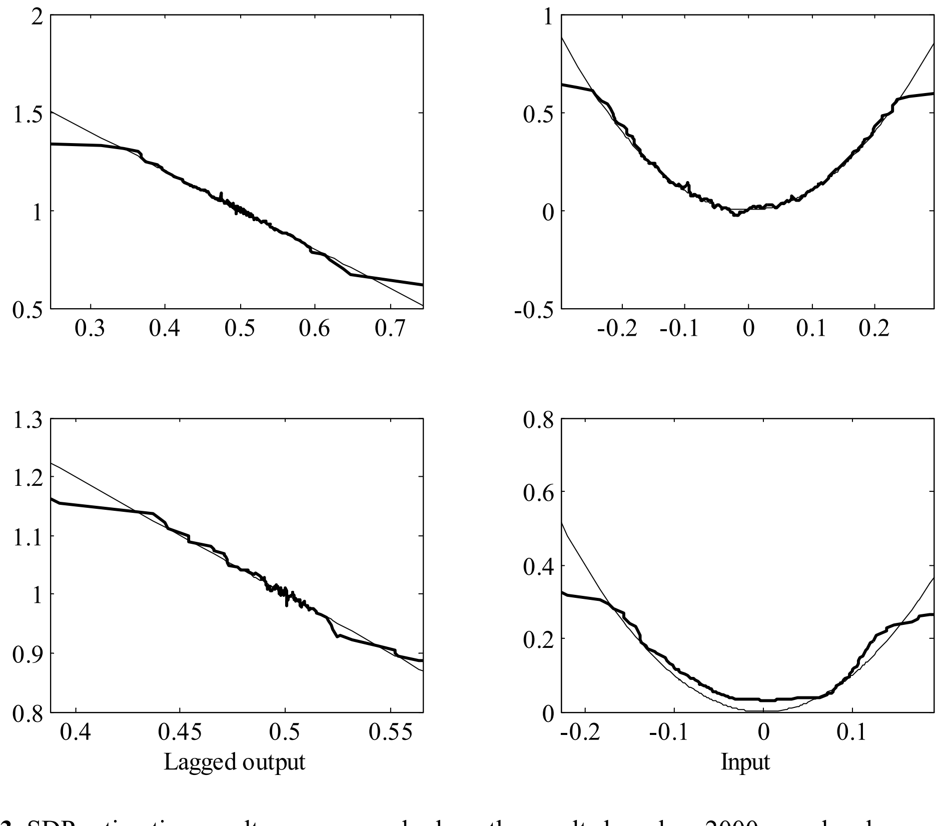


Figure 5.3 SDP estimation results: upper panels show the results based on 2000 samples; lower panels 200 samples. In both cases, the left hand plots show $a_1(y_{k-1})$ and the right hand plots $b_0(u_k)$, returned as the first and second columns in **pars** respectively, plotted against the associated state variable.

The response of the estimated SDARX model can be generated in two ways. First, directly from the equation (5.12), in the usual, SDARX regression-like manner, where it will be noted that the y_{k-1} on the right hand side of the equation is based on the actual measurements y_{k-1} and not the modelled value of this variable. This is returned as **rsq** (see above) and suggests that the model explains 95.8% of the output y_k; i.e. the regression-based COD is $R^2 = 0.958$ (again, this numerical value depends on the particular realisation of the stochastic signals). However, since the SDARX model is a truly dynamic nonlinear system, this is a little misleading. It is more sensible to base the COD on the *simulated* model output, as generated from the equation,

$$\hat{y}_k = \hat{a}_1(y_{k-1})\{k\,|N\}.\hat{y}_{k-1} + \hat{b}_0(u_k)\{k\,|N\}.u_k \qquad (5.13)$$

This is most easily generated by a SIMULINK® model using 'look-up' tables based on the SDP estimation results, as illustrated in Figure 5.4 below.

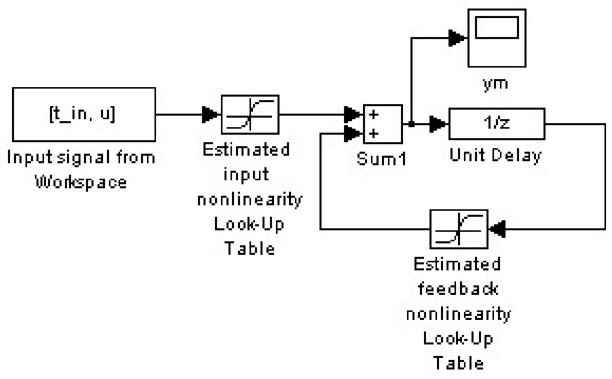

[t_in, u]
Input signal from Workspace
Estimated input nonlinearity Look-Up Table
Sum1
ym
1/z
Unit Delay
Estimated feedback nonlinearity Look-Up Table

Figure 5.4 SIMULINK® block diagram to determine the model response (5.13).

The COD obtained in relation to the actual output y_k, including the effects of the noise e_k, is $R_T^2 = 0.927$, where the subscript T is introduced to differentiate this simulation-based COD from the more normal standard, regression-based R^2. However, if this simulation model output is compared with the 'noise free' output (i.e. $e_k = 0\ \forall\ k$), then $R_T^2 = 0.987$ and it is clear that the SDARX model (5.13) provides an excellent representation of the nonlinear system (5.2). These results are obtained as follows,

```
>> x = zeros(nn, 1);
>> x(1) = 0.5;                  % initial condition
>> for i = 2:nn;                % noise-free simulation response
>>   x(i) = 2.0*x(i-1)-2.0*x(i-1)*x(i-1)+10*u(i).*u(i)*u(i);
>> end
>> t_in=[0:length(x)-1]';       % time vector for block diagram
>> sim('chapt5sim',1999);       % simulate block diagram (Figure 5.4)
>> RT2f = 1-cov(x-ym) / cov(x)  % RT2 based on noise-free system
RT2f = 0.9812
>> RT2n = 1-cov(y-ym) / cov(y)  % RT2 based on original noisy system
RT2n = 0.9161
```

Final parameteric estimation of the SDARX model

Each FIS estimated SDP in the SDARX model can be considered as a 'nonparametric' estimate because it has a different value at each sample in time and can only be viewed in complete form as a graph. However, as we see in the continued example below, it is possible to proceed to a final parametric identification and estimation stage, where the non-parametrically defined nonlinearities obtained initially by FIS estimation are parameterised in some manner in terms of their associated dependent variable. For example, this can be achieved by defining an appropriate parametric model in some convenient form, such as a polynomial or trigonometric function; a radial basis function; a more general neuro-fuzzy relationship; or a neural network. The parameters of this parameterised model can then be estimated directly from the input-output data using some method of dynamic model optimization: e.g. deterministic Nonlinear Least Squares (NLS) or a more statistically efficient stochastic method, such as maximum likelihood.

Example 5.2 Final parameter estimates for the model in Example 5.1

Even without our prior knowledge in this simulation example, it is fairly obvious from Figure 5.3 that the two SDPs are linear and quadratic functions of the associated variables respectively (i.e. the associated nonlinearities are quadratic and cubic functions, respectively). Thus, it is straightforward to obtain these parametric estimates, either by Least Squares (LS) or Weighted Least Squares (WLS) estimation based on the SDP estimation results (Young, 1993a; Young and Beven, 1994); or, preferably, by direct estimation from the data using NLS based on the identified model structure,

$$y_k = ay_{k-1} - by_{k-1}^2 + cu_k^3 + e_k \tag{5.14}$$

In this case, the two sets of estimation results are given as follows:

(i) LS from SDP estimates: $\hat{a} = 1.980\ (0.002);\ \hat{b} = 1.961\ (0.004);\ \hat{c} = 9.999\ (0.055)$.

(ii) Direct NLS from data: $\hat{a} = 1.995\ (0.006);\ \hat{b} = 1.990\ (0.012);\ \hat{c} = 10.00\ (0.056)$.

Here, the estimates (i) are obtained as follows,

```
>> z = [ones(size(zs(:, 1))) zs(:, 1)];
>> ab = inv(z'*z)*z'*pars(:, 1);  % parameters a and b
>> ee = pars(:, 1)-z*ab;
>> P = cov(ee)*inv(z'*z);
>> sd1 = sqrt(diag(P));           % standard errors
>> z = [zs(:, 2).*zs(:, 2)];
>> c = inv(z'*z)*z'*pars(:, 2);   % parameter c
>> ee = pars(:, 2)-z*c;
>> P = cov(ee)*inv(z'*z);
>> sd2 = sqrt(diag(P));
>> disp([ab' c'; sd1' sd2']);
   1.9803   -1.9613    9.9992
   0.0023    0.0042    0.0551
```

The estimates (ii) are obtained directly from the data using conventional MATLAB® optimisation tools, or functions such as **leastsq** available in specialist toolboxes. Although the standard errors on the initial SDP-based estimates (i) tend to be too optimistic, the parametric estimates themselves are close to the true values, showing the efficacy of the SDP estimation stage in identifying the nature of the nonlinearities. Indeed, the SDP estimates obtained for only $N = 200$ samples, as shown by the two graphs in the lower panel of Fig. 5.3, are quite good enough to identify the form of the nonlinear functions themselves.

5.2 Other SDP Models

Of course, SDP models are not restricted to the dynamic SDARX form: they can be of any chosen type as long as the model is in the form of a SDP regression model. The simplest and most obvious of these models is the SDP equivalent of the DLR model (4.1). In the case where the trend component is zero, this model has the following form,

$$y_k = \sum_{i=1}^{i=m} b(z_{i,k})u_{i,k} + e_k \qquad e_k \sim N\{0,\sigma^2\} \qquad k = 1,2,...,N \tag{5.15}$$

where the regression variables $u_{i,k}, k = 1,2,...,m$ are defined by the user. For instance, they could be simply other independent variables that are assumed to be related nonlinearly to the dependent variable y_k; they could be delayed versions of a single 'input' variable, so that the model is a nonlinear SDP version of the linear *Finite Impulse Response* (FIR) model; or they could be 'basis functions' defined in various ways: e.g. as orthogonal functions or principle components. Clearly, the possibilities are multi-various.

However, one model that is worthy of special mention is the *State Dependent Transfer Function* (SDTF) model, which takes the form:

$$y_k = \frac{B(x_{i,k},z^{-1})}{A(x_{i,k},z^{-1})}u_k + \xi_k \tag{5.16}$$

where ξ_k is, in general, coloured noise; and

$$\begin{aligned} A(x_{i,k},z^{-1}) &= 1 + a_1(x_{1,k})z^{-1} + \cdots + a_n(x_{n,k})z^{-n} \\ B(x_{j,k},z^{-1}) &= b_0(x_{n+1,k})z^{-1} + b_1(x_{n+2,k})z^{-2} + \cdots + b_m(x_{n+m-1,k})z^{-m} \end{aligned} \tag{5.17}$$

Once again, the $x_{i,k}, i = 1,2,...,n + m + 1$, are the variables on which the parameters are dependent. However, even in the case where ξ_k is zero mean value, white noise, the SDP estimates obtained from the **sdp** tool in CAPTAIN will show signs of bias caused by the noise. An example of these biasing effects is given in example 3 of Young (2000) and the user of CAPTAIN must take this into account when using the function.

To illustrate the wide ranging utility of the **sdp** tool, one further example is briefly considered below.

Example 5.3 Analysis of Squid Data (Young, 2001a)

A typical example of univariate SDP modelling is an analysis of the squid data shown in Figure 5.5 (Young, 2001a). These squid data were obtained by Kazu Aihara and Gen Matsumoto from experiments on the giant axon of a squid (see Mees *et al.*, 1992). A first order SDAR model of the following kind is identified from the data y_k,

$$y_k = a(y_{k-1}).y_{k-1} + e_k \tag{5.18}$$

The MATLAB® code to identify a SDP model is as follows,

```
>> load squid.dat
>> yd = del(squid, 1);
>> [fit, fitse, par, parse, zs, pars, parses, rsq, nvre] ...
                                = sdp(squid, yd, yd, [], -2);
```

Figure 5.6 is a plot of the SDP estimate against the delayed output y_{k-1}, with the estimated se bounds shown dashed. It is interesting to note that the parameter is roughly constant over the range $y_k < -120$ but that it has wide variations after this, leading to the observed chaotic behaviour. When the actual data y_k are compared with a simulated random realization of the SDP model, it is clear that the latter captures the major nonlinear dynamic characteristics of the electrical signal very well, as discussed by Young (2001a).

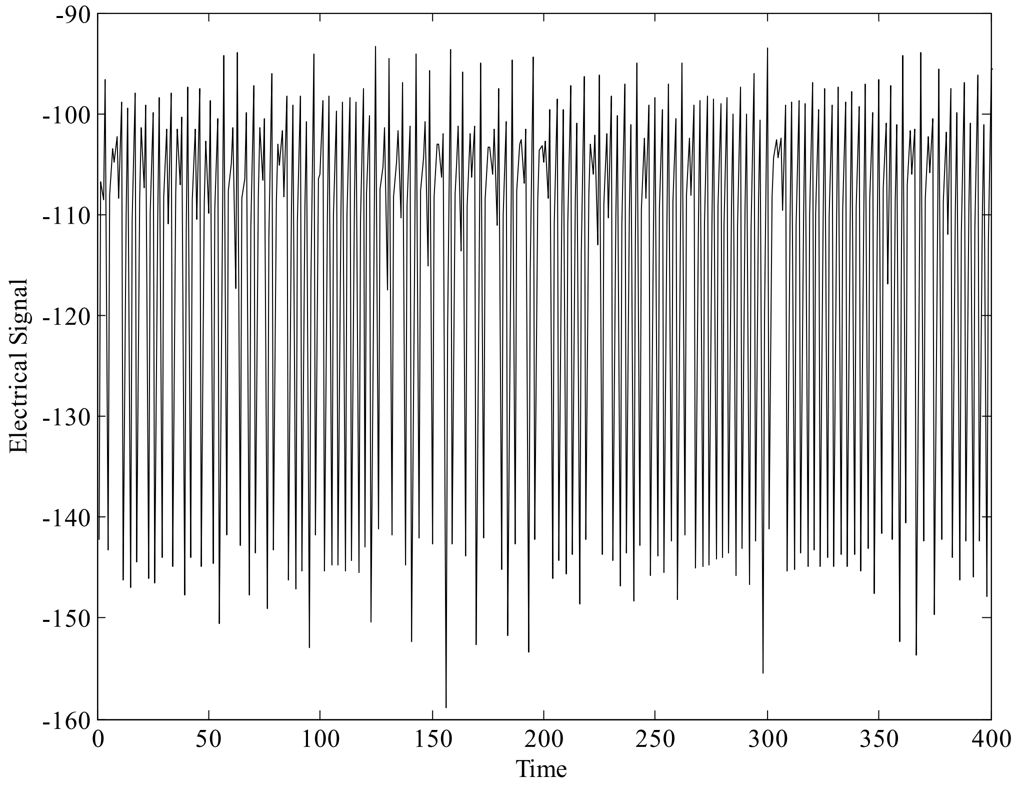


Figure 5.5 Electrical signal obtained from experiments on the giant axon of a squid.

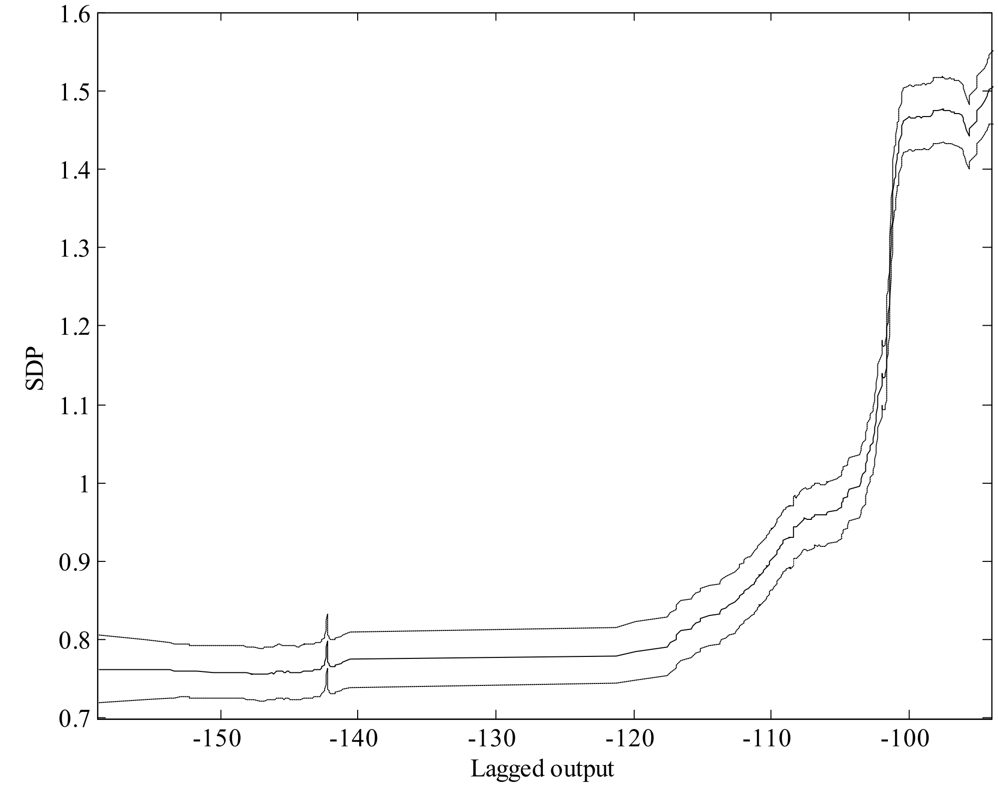


Figure 5.6 SDP estimation results for the squid data: plot of the SDP estimate of the parameter in a first order, SDARX model. The standard error bounds are shown as dashed lines.

5.4 Conclusions

This chapter has summarised the main features of the *State Dependent parameter* (SDP) class of nonlinear, stochastic models that can be identified and estimated using the **sdp** tool in CAPTAIN. This class of model has quite wide applicability, ranging from static SDP regression models to SDARX and SDTF stochastic, dynamic models. The simulation and real examples emphasise the practical utility of the **sdp** tool and relative ease of use. Indeed, it may have even wider application potential, as revealed by research that shows how it can not only result in a drastic reduction in the cost of the sensitivity analysis, but also allow for the estimation of the first order sensitivity terms of high dimensional model representations, at no additional cost (see Ratto *et al.*, 2004).

The examples in this guide are adapted from an earlier version of the CAPTAIN handbook (Pedregal *et al.* 2007). For more recent examples of SDP modelling and, indeed, for data-based mechanistic analysis more generally, as applied to a wide range of engineering, environmental, biological and socio-economic systems, please refer to more recent articles by the present authors, collaborators and other researchers e.g. those that have used and cited the toolbox.

REFERENCES

Akaike, H. (1974) A new look at the statistical model identification, *IEEE Transactions on Automatic Control*, AC-19, 716-723.

Akaike, H. (1980) Seasonal adjustment by a bayesian modeling, *Journal of Time Series Analysis*, **1**, 1, 1-13.

Balke, N.S. (1993) Detecting level shifts in time series, *Journal of Business and Economic Statistics*, **11**, 1, 81-92.

Beck, M.B. and Young, P.C. (1975) A Dynamic Model for DO-BOD Relationships in a Non-Tidal Stream, *Water Research*, **9**, 769-776

Box, G.E.P. and Jenkins, G.M. (1970), *Time Series Analysis: Forecasting and Control*, revised edn, San Francisco: Holden-Day, 1976.

Box, G.E.P., Hillmer, S.C. and Tiao, G.C. (1978) Analysis and modelling of seasonal time series, in A. Zellner (ed), *Seasonal Analysis of Economic time Series*, Washington D. C.: US Dept. of Commerce-Bureau of the Census, 309-334.

Bryson, A.E. and Ho, Y.C. (1969) *Applied Optimal Control, Optimization, Estimation and Control*, Waltham: Blaisdell Publishing Company.

Casals J., Jerez M. and Sotoca S. (2000) Exact Smoothing for stationary and non-stationary time series, *International Journal of Forecasting*, **16**, 59-69.

Cobb, G.W. (1978) The problem of the Nile: conditional solution to a change-point problem, *Biometrika*, **65**, 243-251.

De Jong, P. (1988) The Likelihood for a State Space Model, *Biometrika*, **75**, 1, 165-169.

De Jong, P. (1991) Stable algorithms for the State Space model, *Journal of Time Series Analysis*, **12**, 2, 143-157.

Durbin, J. and Koopman, S.J. (2001) *Time series analysis by state space methods*, Oxford: Oxford University Press.

Findley, D.F., Monsell, B.C., Bell, W.R., Otto, M.C. and Chen, B.C. (1996) New capabilities and methods of the X-12 ARIMA seasonal adjustment program, U.S. Bureau of the Census, mimeo.

Harrison, P.J. and Stevens, C.F. (1976) Bayesian Forecasting, *Journal Royal Statistical Society*, Series B., **38**, 205-247.

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

Hillmer, S.C. and Tiao, G.C. (1982) An ARIMA-Model based approach to seasonal adjustment, *Journal of the American Statistical Association*, **77**, 63-70.

Hillmer, S.C., Bell, W.R. and Tiao, G.C. (1983) Modelling Considerations in the Seasonal Adjustment of Economic Time Series, in A. Zellner (Ed.), *Applied Time Series Analysis* *of Economic Data*, Washington D. C.: US Dept. of Commerce-Bureau of the Census, 74-100.

Hodrick, T. and Prescott, E. (1997) Post-war US business cycles: an empirical investigation, *Journal of Money, Credit and Banking*, **29**, 1-16.

Hu, J., Kumamaru, K. and Hirasawa, K. (2001) A quasi-ARMAX approach to modelling nonlinear systems, *International Journal of Control*, **74**, 1754-1766.

Imbrie, J., Boyle, E.A., Clemens, S.C., Duffy, A., Howard, W.R.,Kukla, G., Kutzback, J., Martinson, D.G., McIntyre, A., Mix, A.C., Molfino, B., Morley, J.J., Peterson, L.C., Pisias, N.G., Prell, W.L., Raymo, M.E., Shackleton, N.J. and Toggweiler, J.R. (1992) On the structure and origin of major glaciation cycles: 1. Linear responses to Milankovitch forcing. *Paleoceanography*, **7**, 701-738.

Jakeman, A.J. and Young, P.C. (1981) Recursive filtering and the inversion of ill-posed causal problems, *Utilitas Math*, **25**, 351-376.

Jarque, C.M. and A.K. Bera (1980) Efficient tests for normality, homoskedasticity and serial independence of regression residuals, *Economic Letters*, **6**, 255-259.

Kalman, R.E. (1960), A new approach to linear filtering and prediction problems, *ASME Transactions Journal Basic Engineering*, **83**-D , 95-108.

Kalman, R.E. and Bucy, R.S. (1961) New results in linear filtering and prediction theory, *ASME Transactions*, *Journal of Basic Engineering*, **83**-D, 95.

Koopman, S.J., A.C. Harvey, J.A. Doornik and Shephard, N. (2000) *STAMP 6.0: Structural Time Series Analyser Modeller and Predictor*, Timberlake Consultants.

Ljung, G.M. and Box, G.E.P. (1978) On a measure of lack of fit in time series models, *Biometrika*, **65**, 297-303.

Maravall, A. and Gómez, V. (1998) *Programs TRAMO and SEATS, Instructions for the User (Beta Version: June 1998)*, Madrid: Bank of Spain.

Mees, A. I., Aihara, K., Adachi, M., Judd, K., Ikeguchi, T. and Matsumoto, G. (1992) Deterministic prediction and chaos in squid axon response, *Physics Letters A*, **169**, 41-45.

Ng, C.N. and Young, P.C. (1990) Recursive estimation and forecasting of nonstationary time series, *Journal of Forecasting*, **9**, 173-204.

Pedregal, D.J., Taylor, C.J. and Young, P.C. (2007) System Identification, Time Series Analysis and Forecasting: The Captain Toolbox, Lancaster University.

Price, L., Young, P., Berckmans, D., Janssens, K. and Taylor, J. (1999) Data-Based Mechanistic Modelling (DBM) and Control of Mass and Energy transfer in agricultural buildings, *Annual Reviews in Control*, **23**, 71-82.

Priestley, M.B. (1989) *Spectral Analysis and Time Series*, 2 vols. (6th printing), London and New York: Academic Press.

Ratto, M., Tarantola, S., Saltelli, A. and Young, P.C. (2004) Accelerated estimation of sensitivity indices using State Dependent Parameter models, *4th International Conference on Sensitivity Analysis of Model Output*, *SAMO 2004*, March 8-11, Santa Fe, NM, USA.

Schweppe, F. (1965) Evaluation of likelihood function for Gaussian signals, *IEEE Transaction Information Theory*, **11**, 61-70.

Taylor, C.J., Pedregal, D.J., Young, P.C. and Tych, W. (2007) Environmental Time Series Analysis and Forecasting with the Captain Toolbox, *Environmental Modelling and Software*, **22**, 797-814.

Taylor, C.J., Young, P.C. and Chotai, A. (2013) *True Digital Control: Statistical Modelling and Non–Minimal State Space Design*, John Wiley & Sons Ltd.

West, M. and Harrison, J. (1989) *Bayesian Forecasting and Dynamic Models*, New York: Springer-Verlag.

Young, P.C. (1970) An instrumental variable method for real-time identification of a noisy process, *Automatica*, **6**, 271-287.

Young, P.C. (1978) A general theory of modeling for badly defined dynamic systems, in G.C. Vansteenkiste (ed.), *Modeling, Identification and Control in Environmental Systems*, North Holland: Amsterdam, 103-135.

Young, P.C. (1984) *Recursive Estimation and Time-Series Analysis*, Berlin: Springer-Verlag.

Young, P.C. (1993a) *Concise Encyclopedia of Environmental Systems*, Oxford: Pergamon Press.

Young, P.C. (1993b) Time variable and state dependent modelling of nonstationary and nonlinear time series. In: Subba Rao. T. (Ed.), *Developments in time series analysis*, Chapman and Hall: London, 374-413.

Young, P.C. (1994) Time-variable parameter and trend estimation in non-stationary economic time series, *Journal of Forecasting*, **13**, 179-210.

Young, P.C. (1998a) Data-based mechanistic modelling of engineering systems, *Journal of Vibration and Control*, **4**, 5-28.

Young, P.C., (1998b) Data-based mechanistic modelling of environmental, ecological, economic and engineering systems, *Environmental Modelling and Software*, **13**, 105-122.

Young, P.C. (1999a) Nonstationary time series analysis and forecasting, *Progress in Environmental Science*, **1**, 3-48.

Young, P.C. (1999b) Data-based mechanistic modelling, generalised sensitivity and dominant mode analysis, *Computer Physics Communications*, **117**, 113-129.

Young, P.C. (2000) Stochastic, dynamic modelling and signal processing: time variable and state dependent parameter estimation. In: W. J. Fitzgerald *et al.* (Eds.) *Nonlinear and nonstationary signal processing*, Cambridge University Press: Cambridge, 74-114.

Young, P.C. (2001a) The identification and estimation of nonlinear stochastic systems. In: A.I. Mees (Ed.), *Nonlinear Dynamics and Statistics*, Birkhauser: Boston.

Young, P.C. (2001b) Comment on 'A quasi-ARMAX approach to the modelling of nonlinear systems' by J. Hu *et al.*, *International Journal of Control*, **74**, 1767-1771.

Young, P.C. and Beven, K.J. (1994) Data-based mechanistic modelling and the rainfall-flow nonlinearity', *Environmetrics*, **5**, 335-363.

Young, P.C. and Pedregal, D.J. (1996) Recursive fixed interval smoothing and the evaluation of LIDAR measurements: A comment on the paper by Holst *et al.* *Environmetrics*, **7**, 417-427.

Young, P.C. and Pedregal, D.J. (1999a) Macro-economic relativity: government spending, private investment and unemployment in the USA 1948-1998, *Structural Change and Economic Dynamics*, **10**, 359-380.

Young, P.C. and Pedregal, D.J. (1999b) Recursive and en bloc approaches to signal extraction. *Journal of Applied Statistics*, **26**, 103-128.

Young, P.C. Pedregal D.J. and Tych, W. (1999) Dynamic Harmonic Regression, *Journal of Forecasting*, **18**, 369-394.

Young, P.C., McKenna, P. and Bruun, J. (2001) Identification of non-linear stochastic systems by state dependent parameter estimation, *International Journal of Control*, **74**, 1837-1857.