A First Course in Time Series Analysis

Young K. Truong
Department of Biostatistics
The University of North Carolina at Chapel Hill

August, 2000

Contents

1 Introduction **1**

1.1 What is a time series? . . . 1
1.2 Types of time series . . . 1
1.3 Examples . . . 2
1.4 Aims of time series analysis . . . 9
1.5 Pseudo random variates . . . 11
1.6 Data sets and software . . . 12
1.7 Examples for using S functions . . . 13

2 Elements of time series **18**

2.1 Finite dimensional distributions . . . 18
2.2 Stationary time series . . . 19
2.2.1 Examples of stationary time series . . . 20
2.2.2 Examples of non-stationary time series . . . 20
2.3 Stationary time series models . . . 21
2.3.1 White noise . . . 21
2.3.2 Moving averages (MA) . . . 22
2.3.3 Autoregressive time series (AR) . . . 24
2.3.4 Gaussian time series . . . 25
2.4 Bivariate time series . . . 27
2.5 Harmonic processes . . . 28
2.6 Convention . . . 28
2.7 More examples . . . 29
2.8 Partial correlations . . . 30
2.9 Methods of time series analysis . . . 30
2.10 Writing S functions . . . 30

3 Spectral analysis **38**

3.1 Preliminaries . . . 39
3.1.1 Harmonic function and frequency . . . 39
3.1.2 Sampling and aliasing . . . 39
3.1.3 Complex numbers . . . 40

3.1.4 Euler's formula — $e^{i\pi} + 1 = 0$ . . . 40
3.2 Spectral representation . . . 41
3.3 Power spectra: Some heuristics . . . 41
3.4 Definition of power spectrum . . . 42
3.5 Linear filters . . . 42
3.6 Examples of power spectra . . . 44
3.6.1 White noise . . . 44
3.6.2 $MA(3)$ . . . 45
3.6.3 $AR(1)$ . . . 45
3.6.4 $AR(2)$ . . . 47
3.6.5 Power spectra of harmonic processes . . . 47
3.7 Spectral representation of real stationary process . . . 48
3.8 S Examples . . . 49
3.9 Appendix . . . 51
3.9.1 Trigonometric identities . . . 51
3.9.2 Orthogonality of sinusoids . . . 52

4 Descriptive Statistics 54
4.1 Sample autocovariance function . . . 54
4.2 Applications . . . 55
4.3 Estimation of power spectrum . . . 57
4.3.1 Choice of $\hat{c}(u)$ . . . 57
4.3.2 Why Fourier Transformation? . . . 59
4.3.3 Discrete Fourier Transformation – DFT . . . 60
4.3.4 Distribution of the DFT . . . 60
4.3.5 The periodogram . . . 63
4.3.6 Confidence interval for $f_x(\lambda)$ . . . 66
4.4 Examples of harmonic analysis . . . 66
4.4.1 Simple harmonic process . . . 66
4.4.2 A step function . . . 68
4.4.3 Periodogram analysis of a simple harmonic series . . . 68
4.4.4 Complex demodulation . . . 70
4.5 Smoothed periodograms . . . 73
4.6 Asymptotic distributions . . . 78
4.7 Confidence intervals . . . 78
4.8 $\log_{10} \hat{f}(\cdot)$ — *decibels* (db) . . . 78
4.9 Harmonics . . . 79
4.10 Testing for periodogram ordinates . . . 80
4.11 The cumulative periodogram . . . 81
4.12 Appendix 4A. Discrete Fourier Transform . . . 84
4.12.1 Preliminary . . . 84
4.12.2 Orthonormal bases . . . 84

4.12.3 Discrete Fourier Transformation . . . 85
4.12.4 Parseval identity . . . 85
4.12.5 Periodogram . . . 86
4.13 Appendix 4B. Complex normal distribution . . . 86
4.14 Appendix 4C. Normal approximation to $\log(\chi^2_\nu/\nu)$ . . . 87
4.14.1 Gamma distributions . . . 87
4.14.2 χ^2 distributions . . . 87
4.14.3 F distributions . . . 87
4.15 Appendix 4D. Density estimation . . . 88

5 ARMA Processes 93
5.1 The B Operator . . . 93
5.2 Autoregressive processes . . . 93
5.2.1 Causality . . . 94
5.2.2 Yule-Walker equations . . . 94
5.2.3 Computing the autocovariance function . . . 94
5.2.4 Partial autocorrelation (ACF) . . . 95
5.2.5 Estimation . . . 95
5.2.6 Computing the estimates . . . 96
5.2.7 The Durbin-Levinson Algorithm . . . 99
5.2.8 Burg's algorithm for AR(p) . . . 99
5.2.9 AR spectral estimates . . . 100
5.3 Moving-average processes . . . 101
5.3.1 Invertibility . . . 102
5.3.2 Estimation . . . 102
5.3.3 Conditional likelihood . . . 104
5.4 Autoregressive-moving average (ARMA) process . . . 105
5.4.1 Causality . . . 105
5.4.2 Invertibility . . . 106
5.4.3 Infinite moving average process . . . 106
5.4.4 Preliminary estimation . . . 107
5.4.5 Conditional likelihood . . . 107
5.4.6 Exact likelihood . . . 108
5.4.7 Least squares estimators (LSE) . . . 109
5.4.8 Asymptotic distribution of the LSE . . . 110
5.4.9 Computing the estimates . . . 110
5.4.10 Examples . . . 111
5.5 Model identification . . . 114
5.6 $ARIMA\ (p,d,q) \times (P,D,Q)_s$ . . . 116
5.7 Remarks . . . 122
5.8 Examples . . . 123
5.8.1 Blood pressure data . . . 127

5.8.2 Lutenizing hormone data 129
5.8.3 Mortality data 133
5.8.4 Monthly accidental death data 141
5.8.5 More on simulation 147
5.9 Forecasting 149
5.10 Diagnostics and portmanteau test 150
5.11 SAS Examples 150
5.12 Appendix A. Transformation formulae 180
5.13 Appendix B. Covariance generating function 181

6 Bivariate Time Series 184
6.1 Cross-correlations 184
6.2 Test for independence 185
6.3 Cross-spectra 192
6.4 Co-spectra and quad-spectra 193
6.5 Amplitude (gain) and phase spectra 193
6.6 An example 194
6.7 Linear relationships 194
6.8 Estimation of cross-spectra 196
6.9 Testing for zero coherence 197
6.9.1 S functions for computing estimates of spectra 198
6.9.2 S functions for computing coherence and phase 202
6.9.3 S functions for testing zero coherence 205
6.10 Estimation of the transfer function 211
6.11 Alignment 215
6.12 Regression analysis 220
6.13 SAS Examples 231
6.14 SAS Examples 243

7 State Space Models and the Kalman Filter 251
7.1 Measurement equation 251
7.2 Transition equation 251
7.3 ARMA via State Space 251
7.4 Minimum Mean Square Estimation 252
7.4.1 Combining Prior and Sample Information 252
7.5 Updating and Prediction Equations 252
7.5.1 Prediction 253
7.5.2 Updating 253
7.6 Examples 254
7.6.1 Example 1 254
7.6.2 Example 2 255
7.7 Computing the likelihood function 255

7.8 Examples 256

8 Advanced Topics 285
8.1 A closer look at the inconsistency of $I_n(\lambda)$ 285
8.2 The LOGSPLINE methodology 287
8.3 The LSPEC methodology 291
8.4 Trend estimation 297
8.5 Functional modeling in regression 297
8.6 Functional modeling via ELM 302

9 S Functions for Time Series Analysis 303
9.1 Standard S functions 303
9.2 Additional S functions 322

10 Appendix: Data sets and S functions 332

Preface

Time series analysis concerns the analysis of a sequence of data that are measured or observed over time. It is an important branch of statistics whose practical importance has increased with the advance of computer technology. Now there is a good collection of routines for time series analysis in many standard statistical computing packages such as SAS and SPLUS. Such developments provided an ideal environment for applied time series analysis.

The book is an introductory text, written for teaching time series analysis to students in biostatistics or other apply areas with a particular interest in biological applications. Data analysis drawn from many different fields are discussed throughout the book. All discussions are kept at an intuitive level. Difficult mathematical concepts are illustrated with computer experiments or simulations. Data and software written in S functions are stored on a diskette, and they are essential for understanding the methodological developments.

In the past, I have used the book by P. Diggle and R. Shumway. The former is closer to what I had in mind for biostatistics, but the text lacks a lot of software and exercises. The latter was designed for business or geophysical signal applications. In preparing the book, I wrote a library of S functions as a supplement to SPLUS for time series analysis based on TIMESLAB (Newton). For example, the estimate of the spectral density is better than the one in SPLUS, and the software for bivariate modeling based on transfer function method has a better implementation. Fisher's test is also provided. For the data, I have drawn them from neurobiology to environmental studies. Thus they should provide a wide spectrum of time series analysis.

In teaching time series, it is difficult to avoid asymptotic theory because there is hardly any exact theory. I have developed a sequence of experiments to make that theory easy to understand without going through the mathematical arguments. I have found that to be very effective in teaching time series.

The book assumes the basic knowledge from an introductory course on simple linear regression, a little bit of probability with expectation and covariance. Some exposure to multivariate normal distribution and matrix theory would be helpful but is not essential. There is an appendix for handling this part.

Chapter 1

Introduction

1.1 What is a time series?

A time series is a sequence of measurements of some variable through 'time'. We usually denote the time series by $\{y_t : t = 1, \ldots, n\}$. In ordinary statistical approaches, the sequence $y_1, y_2, \ldots, y_n$ are taken to be *mutually independent*, while in time series analysis, the particular interest would be to study the nature of *dependence* among the members of the sequence.

1.2 Types of time series

1. A time series is called continuous or discrete depends on whether the measurement y_t is continuous or discrete.

2. In practice, the time t is always discrete and might not be necessarily equally spaced. This often happens when we work with discretized data of a continuous (or analog) time series such as ECG data. A more general notation is $\{y_{t_i} : i = 1, \ldots, n\}$.

3. It often happens that at a particular time t, there will be more than one measurement recorded such as the temperatures at two different cities. Such a time series is called a *vector-valued time series.*

4. When we speak of time series, we often think of t as 'time'. In the statistical study of spatial patterns and processes, t might be a location on a surface, such as the pixel location of a medical image. Thus, t is the index consisting of a pair of numbers: the horizontal and the vertical coordinates. In this case, we have a *spatial process* rather than a time series.

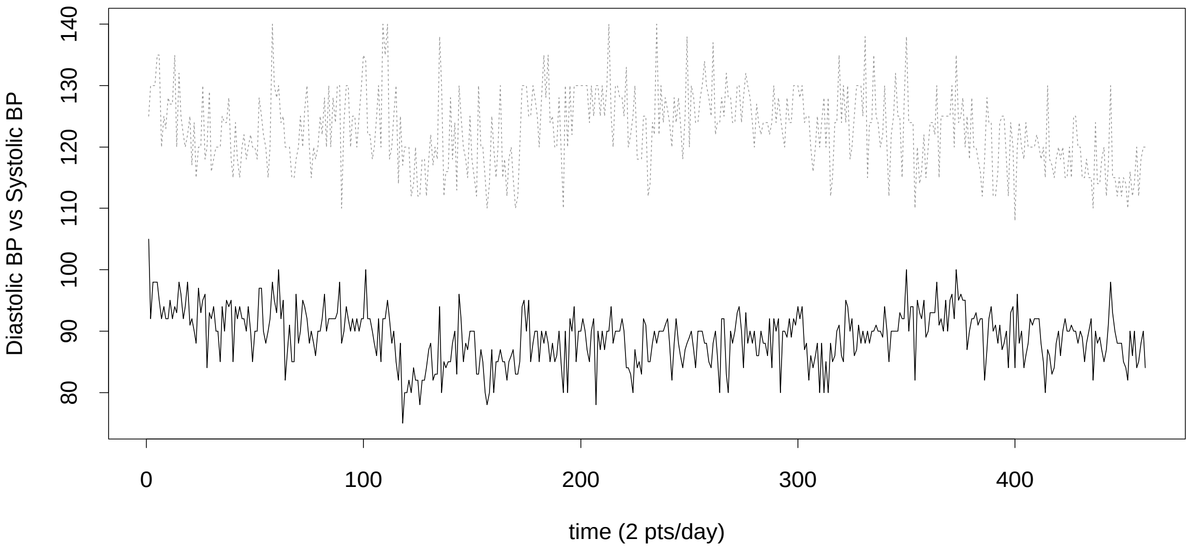


Fig 1.1. *460 blood pressure measurements (twice/day) (Shumway, 1988).*

1.3 Examples

1. Figure 1.1 plots the simultaneous recordings of systolic and diastolic blood pressure measurements (see the file `bp`) of an individual, sampled twice a day for 230 days (Shumway, 1988). We note these values oscillate rather rapidly and we say that they represent high frequency observations (or noisy data).

2. In a neurophysiological experiment consisting of electrical potential (EP) measurements, it started with a 24 Hz (cycles/sec), $500\mu m$ peak to peak sinusoidal stimulus applied to the right fingertip. The responses are the EP measurements at the scalp and wrist. Each EP measurement lasted for 6 seconds, with the stimulus coming on at 2 seconds and staying on for the remainder of the trial. The channels were sampled at 256 times/sec, giving a total of 1536 sampling points per channel. Figure 1.2 shows the averages of 30 EP responses (see the file `ep`) from the scalp and wrist. Note the top series is smoother than the bottom series.

3. Figure 1.3 shows four time series of lutenizing hormone (LH) in blood samples (see the file `lh`) taken from a woman at ten-minute intervals over an eight-hour period. The cyclic patterns are important features for studying the human reproductive system (Murdoch et al, 1985). LH plays an important role in the reproductive system. The fundamental questions posed by these data include the inverstigation of possible cyclic patterns within each series due to pulsatile release of LH, characterization of differences between the properties of series

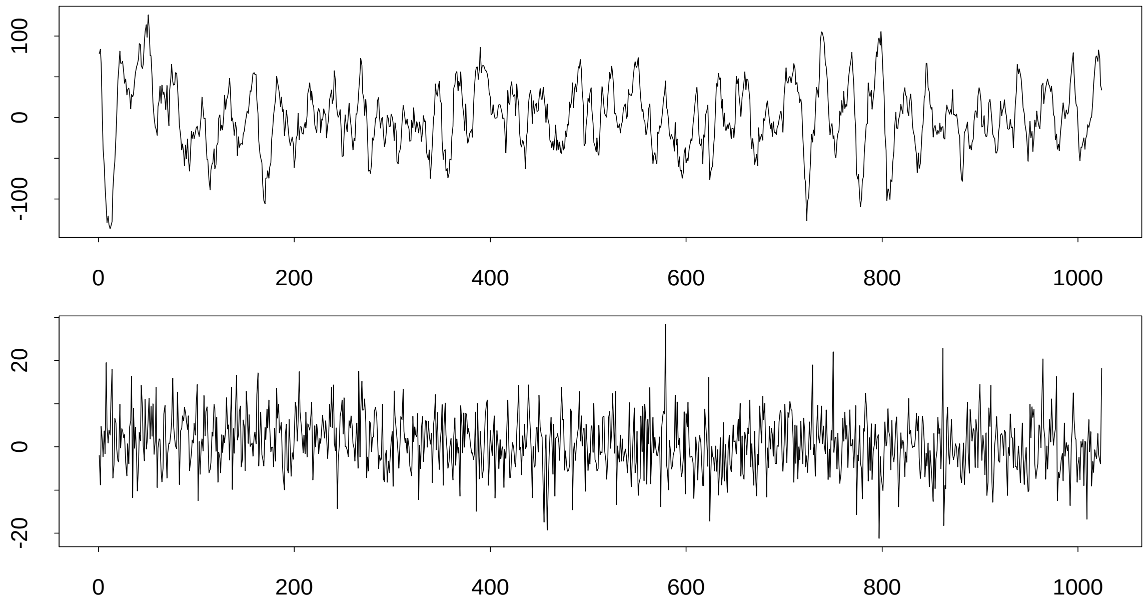


Fig 1.2. *Averages of 30 series of electrical potential (EP) measurements from the scalp (top) and wrist (bottom).*

taken from different phases of the menstrual cycle, and assessment of whether or not series taken from the same phase of successive cycles are comparable.

4. Mortality and environmental series (cf. `pm`): Figure 1.4 shows the daily deaths of residents of the Birmingham, Alabama, metropolitan area from 1985–1988. In addition we have daily mean, max, min temperatures (C), specific humidity (g/kg), wind speed (m/s), air pressure (mb) and PM readings averaged over three days [Schwartz (1993).] The objective was to determine which, if any, of the air pollution or weather factors might be most strongly associated with increased mortality.

5. Speech analysis: Figure 1.5 shows a time series (cf. `ahh`) of recorded speech of the syllable ahh recorded over a tenth of a second.

6. Figure 1.6 shows the 373 monthly registered live births (in thousands) in USA from January 1948 through January 1979. Data (cf. `births`) were taken from the Monthly Vital Statistics Reporter, National Center for Health Statistics, US Public Health Service (current values may have been adjusted). Observe that oscillating values contains a strong periodic pattern that completes one full cycle every year. The births tend to be highest in August or September and lowest during February. The overall smoother trend (mean) may be of greatest interest.

7. Magnetic field exposure level (cf. `p103`): Figure 1.7 shows the measurement

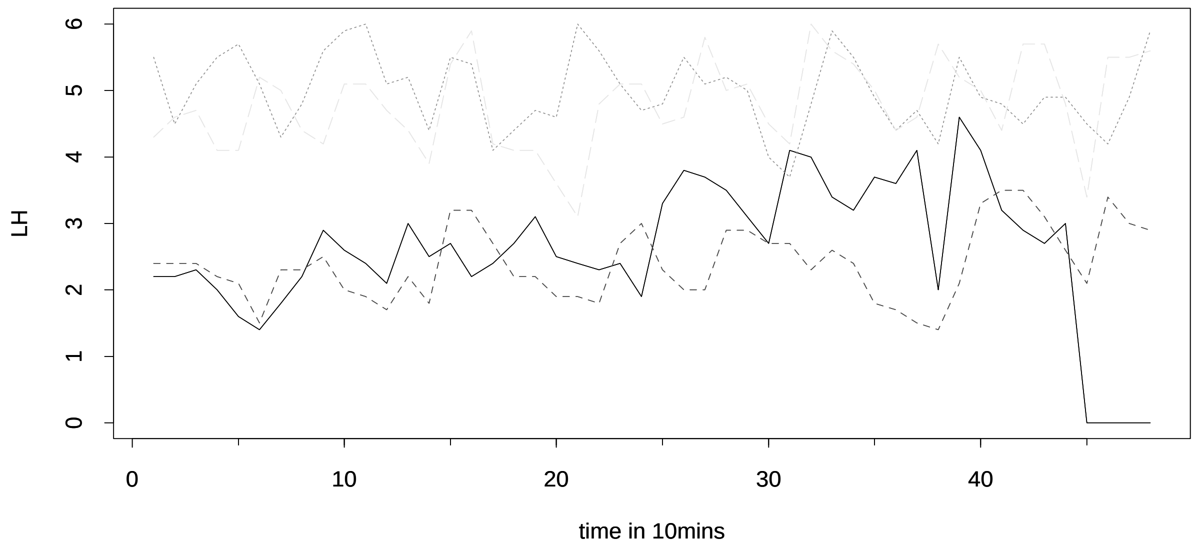


Fig 1.3. *Luteinizing hormone in blood.*

of magnetic field exposure level of a lineman. These were obtained from the EMDEX-C meter, a portable battery operated unit with a microprocessor controller and memory storage capability to sample and store electric or magnetic field measurements at programmed intervals of 10 seconds with storage capacity up to 4.4 days. There are 107 of such time series, and the object is to study the exposure patterns of linemen in the field.

8. Hydrology: river and lake level. Figure 1.8 is a time series from the Lake Erie data (cf. `eriel`). One of the interesting features of this data set is the seasonal component, which will be discussed later in the course.

9. Economics: Figure 1.9 shows 372 monthly Federal Reserve Board Production Index and the unemployment figures (cf. `unemp`) over a 30-year period (1948-1978). The FRBPI has a steadily increasing trend which will be modeled and studied using time series methods.

10. Astronomy: 459 monthly sunspots (1749-1978) are plotted in Figure 1.10. The cycles are approximately 11-year. This data set, along with the Canadian Lynx data set, are examples of non-Gaussian time series.

11. Mortality from lung disease and environmental series (cf. `mort`): Figure 1.11 shows the mortality, sulfur dioxide, British smoke (estimated from the residue left on filter paper), daily temperature and relative humidity collected daily for a total of 105 days during the winter of 1958 in London. The objective was

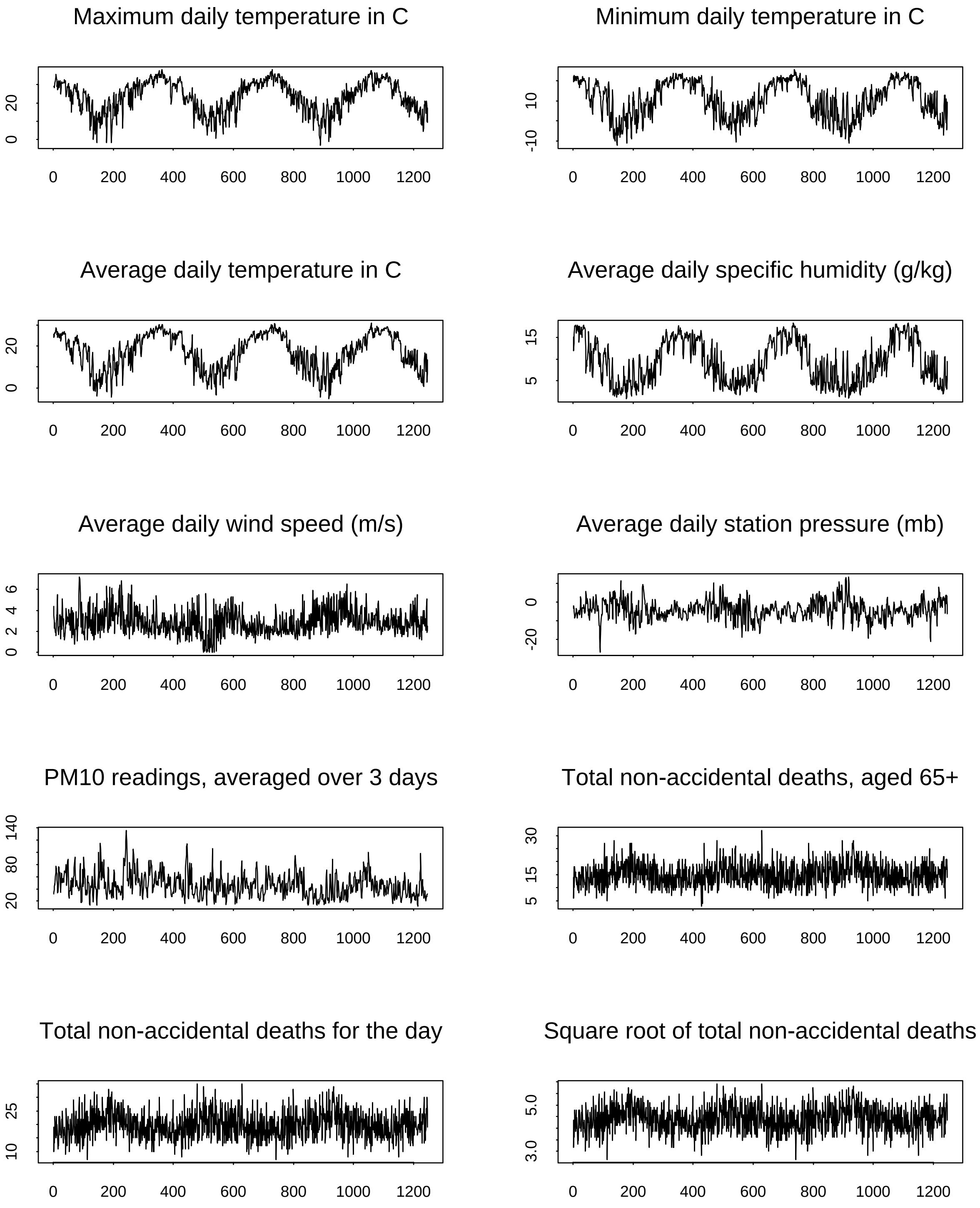


Fig 1.4. *Air pollution and daily mortality in Birmingham, Alabama.*

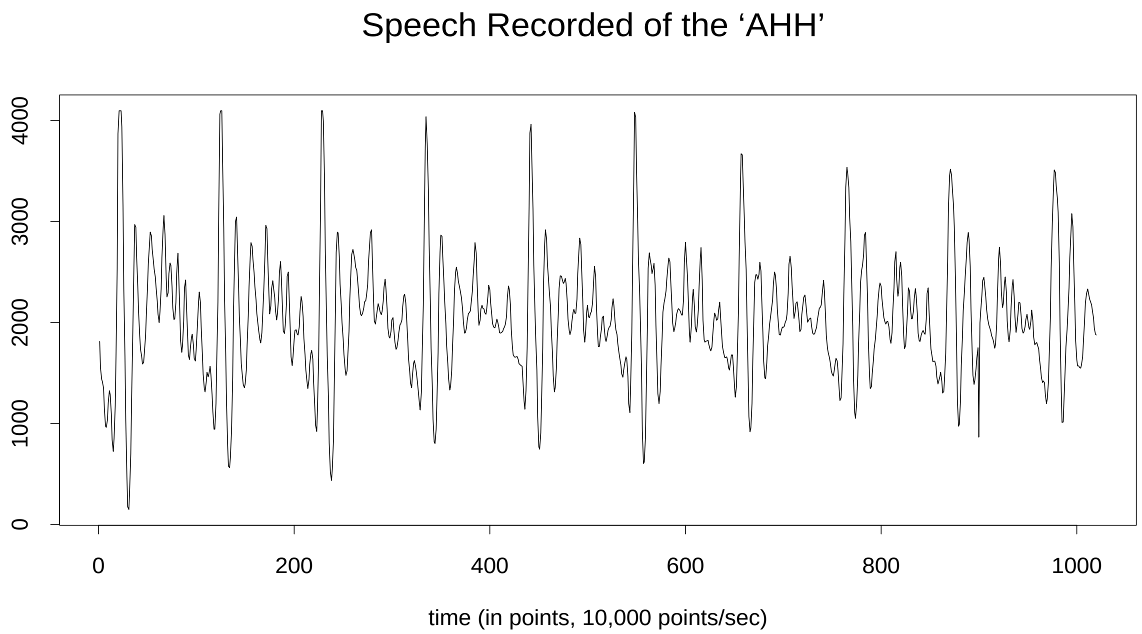


Fig 1.5. *Speech Recorded of the 'AHH'.*

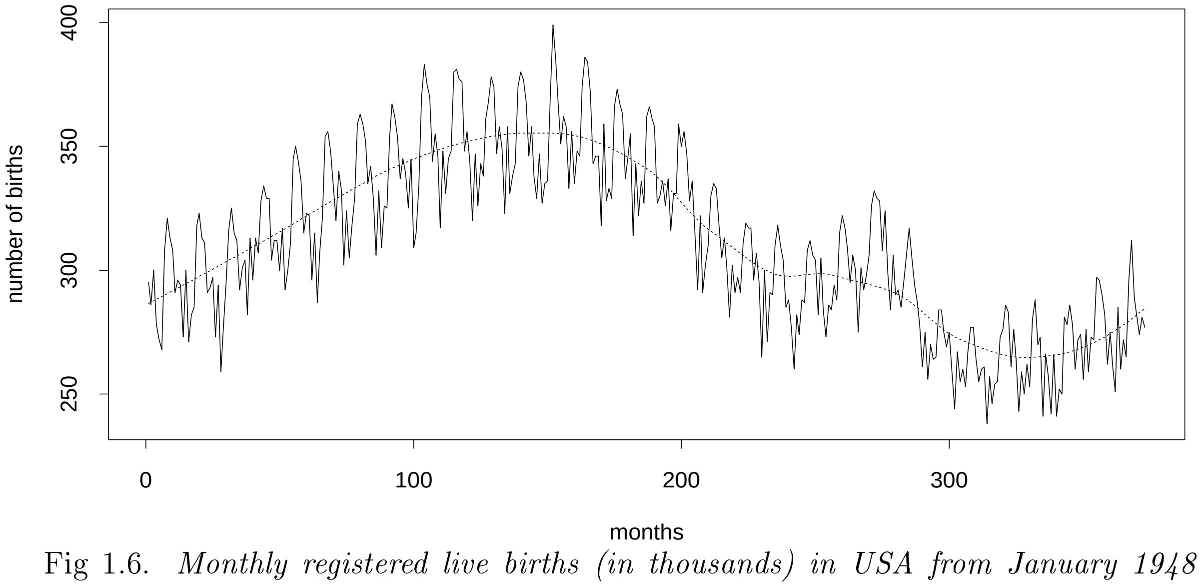


Fig 1.6. *Monthly registered live births (in thousands) in USA from January 1948 through January 1979 (T=373).*

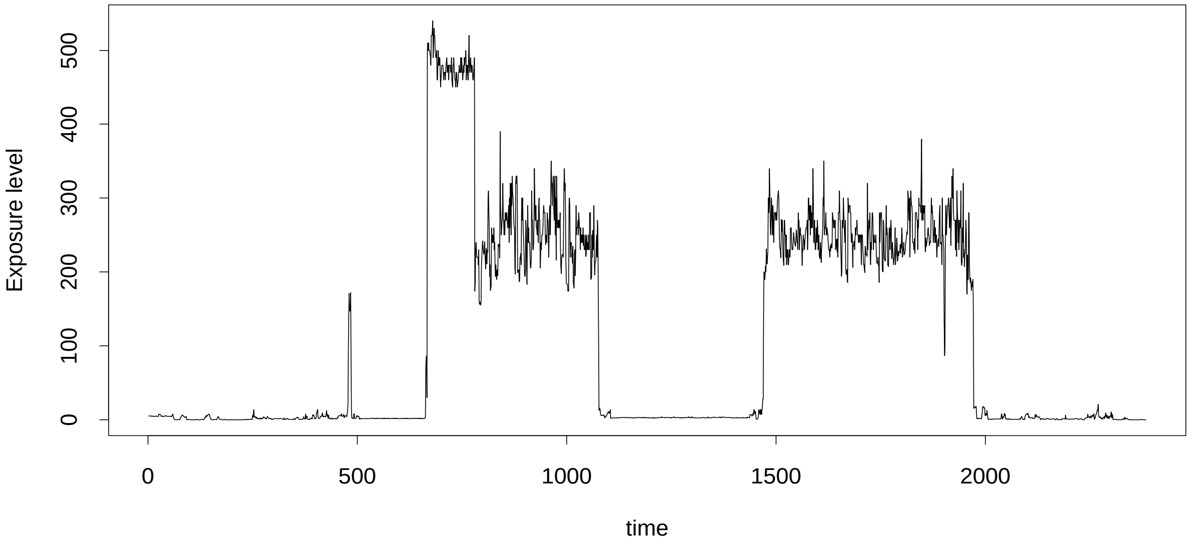


Fig 1.7. *Magnetic exposure level.*

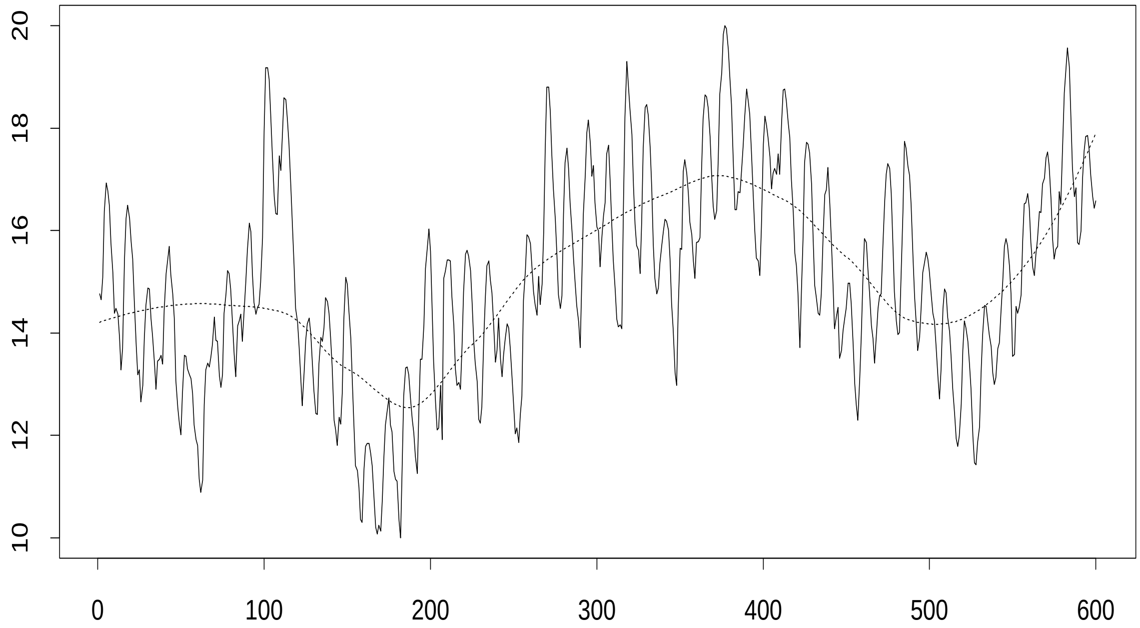


Fig 1.8. *Lake Erie Levels 1921-1970.*

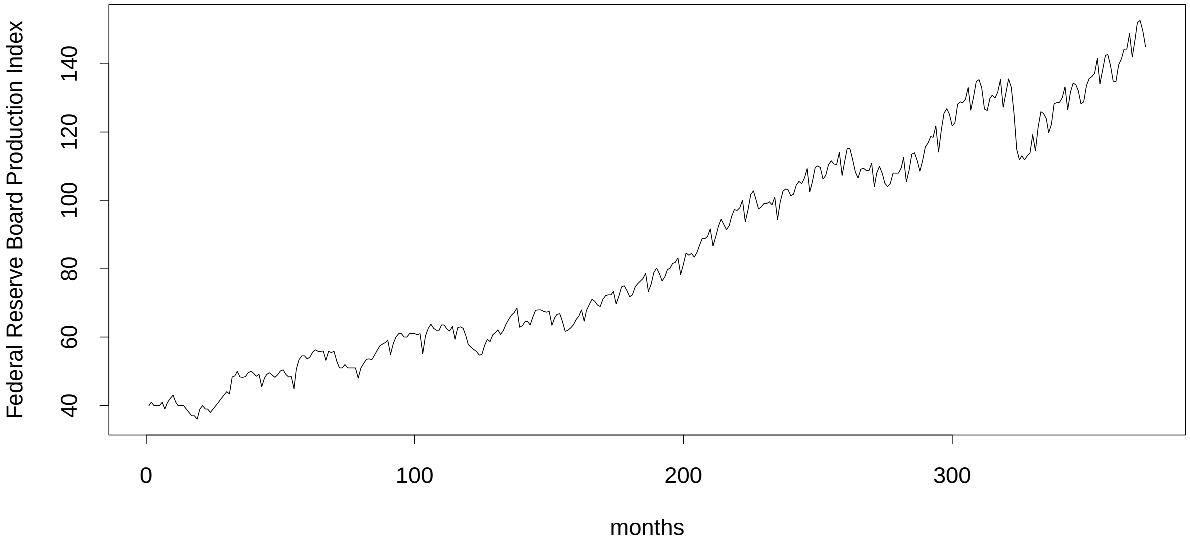


Fig 1.9a. *Federal Reserve Board Production Index, 1948–1979 (T=372).*

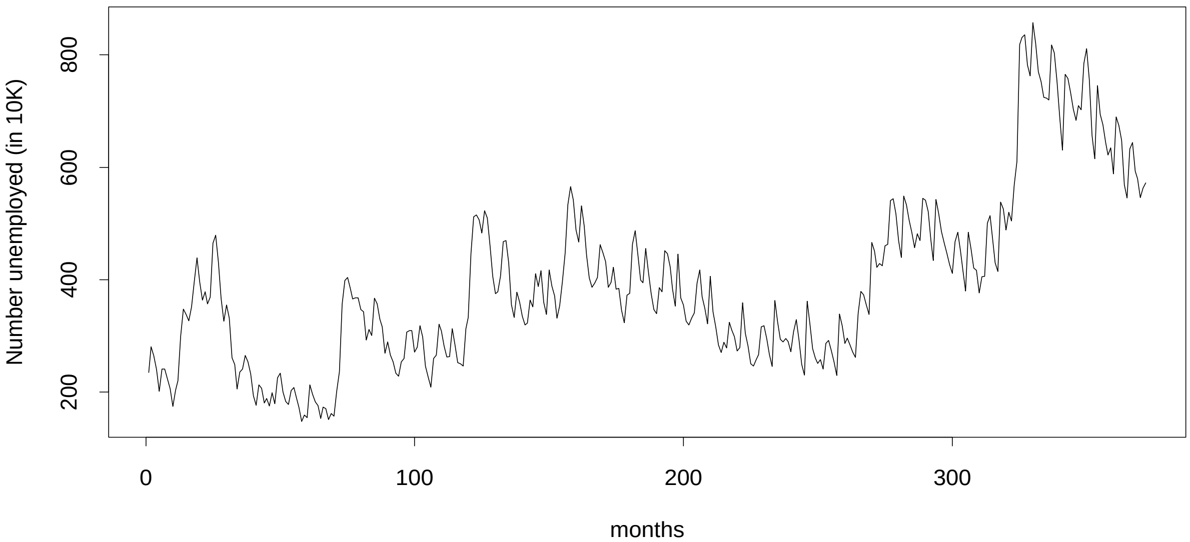


Fig 1.9b. *Unemployment figures, 1948–1979 (T=372).*

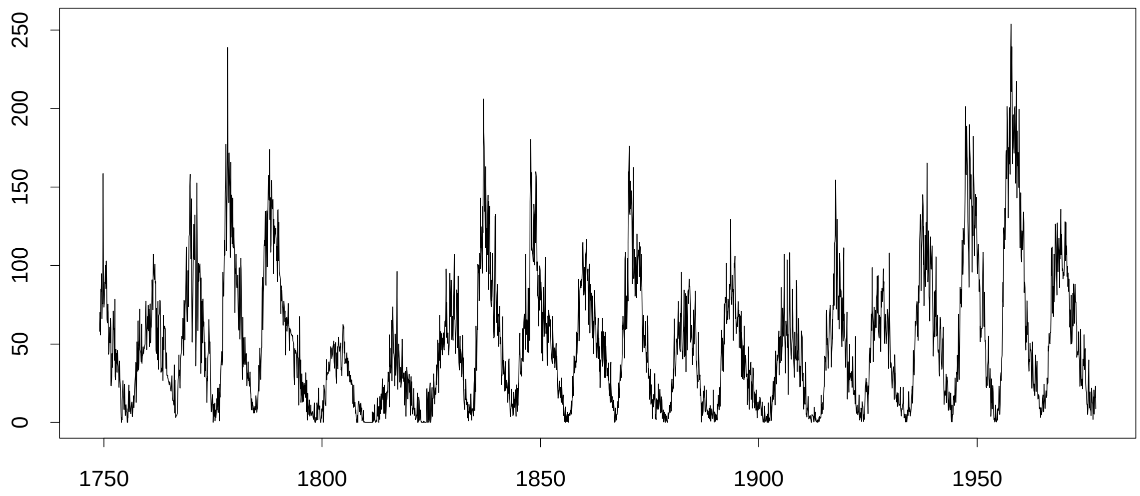


Fig 1.10. *Monthly Mean Relative Sunspots, 1750-1975.*

to determine which, if any, of the pollution or weather factors might be more strongly associated with mortality.

12. Gas furnace data `gasfurn` (Box and Jenkins, 1971). This is a well known data set for studying regression problem in time series. The input series is the gas rate (`rate`) and the output is the carbon dioxide (`CO2`). We will use this data for discussing issues of modeling the transfer function.

Note. Continuous time variables are viewed as discrete time measurements.

1.4 Aims of time series analysis

1. Descriptive: Look for special patterns. For example,
   - *cyclical patterns* exhibited in the LH data (Figure 1.3), sunspots (Figure 1.10)
   - *seasonal variation* in the US births series (Figure 1.6)
   - *trends* exhibit in the FRBPI series (Figure 1.9a)
   - *extraordinary observations* such the outlier in the LH series.
2. Inferential: Tesing serial correlations, periodic components, ...
3. Predictive: Forcasting mortality, births and hospital register during extreme weather conditions.

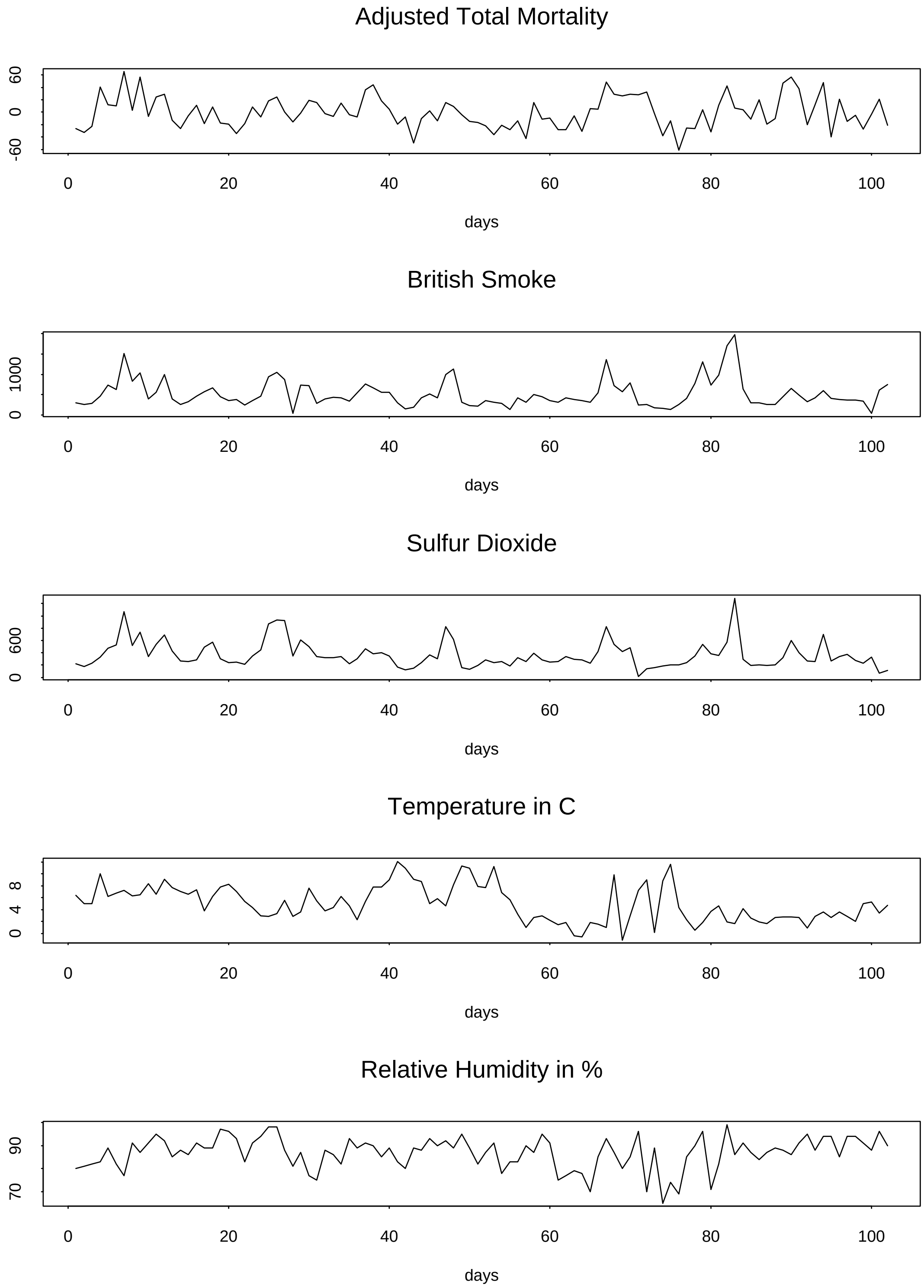


Fig 1.11. *Mortality in lung disease.*

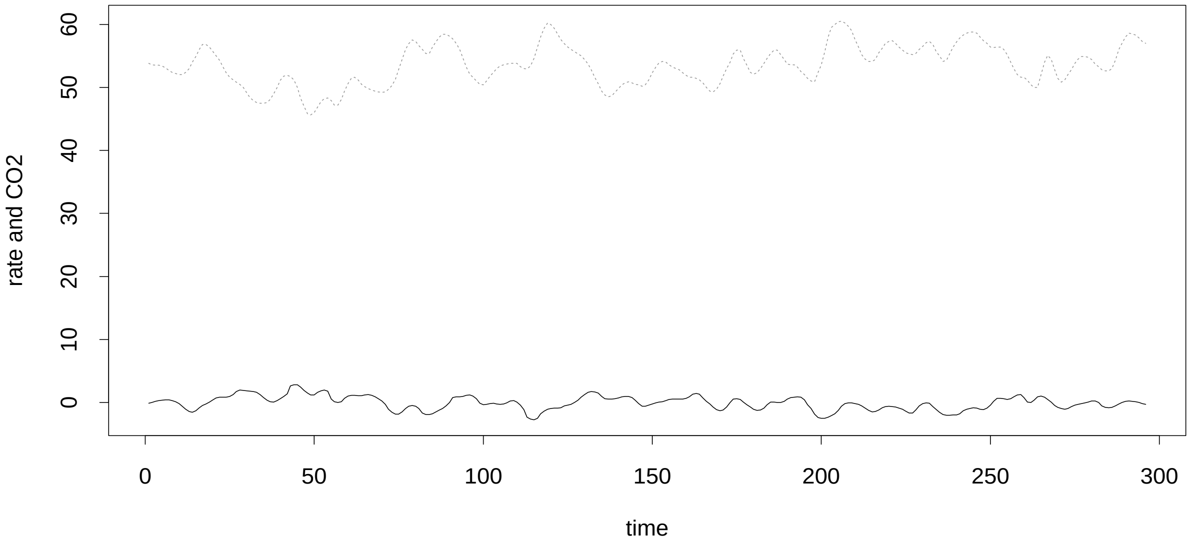


Fig 1.12. *The Gas-Furnace data.*

4. Control: Make sure the future values are in line.

1.5 Pseudo random variates

In this section, we discuss briefly the principle of generating random numbers on the computer by means of exercises.

1. *Inverse Transform Method* Let $U \sim \text{unif}(0,1)$ and let $F(\cdot)$ be a continuous cdf so that the inverse $F^{-1}(\cdot)$ exists. Show that the random variable $F^{-1}(U)$ has a distribution F.

2. *Exponential Random Variates* Describe how you would generate a random variate from the exponential distribution $\exp(\beta)$, i.e. $F(t) = 1 - \exp(-t/\beta)$ $(t > 0)$. Given that α is a positive integer, discuss whether it is possible to generate observations from a gamma distribution $\gamma(\alpha, \lambda)$. Recall the density of gamma random variable is given by

$$f(x) = \frac{x^{\alpha-1}e^{-x/\beta}}{\beta^\alpha \Gamma(\alpha)} \cdot 1_{\{x>0\}},$$

   where $\alpha, \beta > 0$.

3. *Normal Random Variates* The Box-Muller formula:

$$\begin{aligned} Z_1 &= (-2\log U_1)^{1/2}\cos 2\pi U_2 \\ Z_2 &= (-2\log U_1)^{1/2}\sin 2\pi U_2 \end{aligned}$$

yield two independent standard normal vairates Z_1 and Z_2, where U_1 and U_2 are two independent unif(0,1). Equivalently, this can be written as

$$Z_1 = (2V)^{1/2}\cos 2\pi U \quad \text{and} \quad Z_2 = (2V)^{1/2}\sin 2\pi U,$$

where V and U are independent exp(1) and unif(0,1), respectively. [Hint. Apply Jacobian formula to $Z_1^2 + Z_2^2 = 2V$ and $(Z_2/Z_1) = \tan 2\pi U$.]

4. *Acceptance-Rejection Method* Let $f(\cdot)$ denote a density function and suppose that

$$f(x) = Cg(x)h(x) \tag{1.1}$$

where $C \geq 1$, $0 \leq g(x) \leq 1$ and $h(x)$ is also a density function. Let $U \sim$ unif$(0,1)$ and $Y \sim h(x)$, respectively. Verify

$$\mathsf{P}(Y = x | U \leq g(Y)) = f(x) \tag{1.2}$$

[Hint. Apply Bayes' formula to $\mathsf{P}(Y = x|U \leq g(Y))$.]

The implication of this result is as follows: To generate observations from $f(\cdot)$, we first find a pair of functions $g(\cdot)$ and $h(\cdot)$ such that (1.1) holds. Then generate a pair of random variates U and Y from unif(0,1) and h, respectively. If $g(Y) \geq U$, then accept Y which, according to (1.2), has the desired distribution $f(\cdot)$.

5. Discuss how you would generate normal variates from the following density:

$$p(x) = \left(\frac{2}{\pi}\right)^{1/2} \exp(-x^2/2), \qquad x \geq 0$$

by applying acception-rejection method with

$$h(x) = e^{-x}, \quad C = (2e/\pi)^{1/2} \quad \text{and } g(x) = \exp\left(-(x-1)^2/2\right).$$

1.6 Data sets and software

All the data sets and S functions for time series analysis are available in S-PLUS. Description of the datasets can be found in the directory `/distrib/bios259`. For example, if the name of a dataset is given by (say) `bp`, then the details of the dataset is described in `bp.d`. These data sets can be obtained by entering S and type

```
library(tslib)
```

S-PLUS will be our primary software for analyzing data. Here a useful reference is Venables and Ripley (1994) and it will be referred to as MASS.

A tutorial for using S-PLUS is given the next section.

1.7 Examples for using S functions

Before using S-PLUS in Unix, use the command `mkdir` to create a working directory. For example,

```
% mkdir b259
```

will create a new directory called `b259`. Move into this new directory by entering

```
% cd b259
```

Then enter

```
% mkdir .Data
```

so that all your data created during the S-PLUS session will be stored in the directory `.Data`. Now we are ready to use S-PLUS. To do this, simply type

```
% Splus
```

Once in S-PLUS, you will see something similar to this

```
S-PLUS : Copyright 1988, 1995 MathSoft, Inc.
S : Copyright AT&T.
Version 3.x  ...
Working data will be in ...
>
```

Note the symbol `>` is the prompt for S-PLUS. Type `?help` to see a list of instructions on how to get help.

```
> ?help     # Get help
...
> sink("output")    # save all subsequent printing to ‘output’
> a <- .31          # a = .31
> a
[1] .31
> sink()            # stop saving to ‘output’
> library(tslib)    # Load tslib for datasets and functions
```

(In S, the sequence `a <-` is read as "a gets".) To quit S-PLUS, type `q()`.

```
> q()   # adieu
```

Now start cooking!

Examples

1. The most important thing of all is to learn how to read in the data. The function `rts` should do this job. Type `?rts` to read more details about this function. To create a time series `x` of length 15, enter

   ```
   > x <- rts(scan(n=15))
   ```

   S-PLUS responds with

   ```
   1:
   ```

   Then enter the time series next to `1:` such as

   ```
   1: 2.4 2.4 2.4 2.2 2.1 1.5 2.3 2.3 2.5 2.0 1.9 1.7 2.2 1.8 3.2
   ```

   [What happens if you entered more than 15 numbers?] To see what is in `x`, simply type `x` as in

   ```
   > x
    1: 2.4 2.4 2.4 2.2 2.1 1.5 2.3 2.3 2.5 2.0 1.9 1.7 2.2 1.8 3.2
    start deltat frequency
        1      1         1
   ```

2. Before plotting figures, you need to type

   ```
   > X11() # or motif()
   ```

   Now plot the time series `x` by issuing

   ```
   ts.plot(x)      # tsplot(x)
   ```

   Note that in S-PLUS, the special symbol `#` is used for making comments or remarks, and anything after that symbol will not be executed. Now try `tsplot(x)` and see what happens.

3. The following S functions were used to produce Figure 1.1.

   ```
   library(tslib)
   attach(bp)
   tsplot(dbp,sbp,xlab="time (2 pts/day)",
            ylab="Diastolic BP vs Systolic BP")
   plot(dbp,sbp)     # scatterplot
   ```

4. S functions for Figure 1.4.

```
attach(pm)

par(mfrow=c(5,2))
avet <- (tmax+tmin)/2
y <- sqrt(tmort)

tsplot(tmax)
title("Maximum daily temperature in C")
tsplot(tmin)
title("Minimum daily temperature in C")
tsplot(avet)
title("Average daily temperature in C")
tsplot(mnsh)
title("Average daily specific humidity (g/kg)")
tsplot(awnd)
title("Average daily wind speed (m/s)")

tsplot(pres)
title("Average daily station pressure (mb)")
tsplot(pm10)
title("PM10 readings, averaged over 3 days")
tsplot(mort65)
title("Total non-accidental deaths, aged 65+")
tsplot(tmort)
title("Total non-accidental deaths for the day")
tsplot(y)
title("Square root of total non-accidental deaths")
```

5. S functions for Figure 1.5.

```
attach(ahh)
tsplot(ahh,xlab="time (in points, 10,000 points/sec)")
title("Speech Recorded of the ‘AHH’ ")
```

6. S functions for Figure 1.6.

```
attach(births)
tsplot(births,xlab="months",ylab="number of births")
title("US Births")
```

7. S functions for Figure 1.9.

```
attach(umemp)
tsplot(index,xlab="months",ylab="Federal Reserve Board Production Index")
tsplot(noune,xlab="months",ylab="Number unemployed (in 10K)")
# To put two plots on the same page, try this:
par(mfrow = c(2, 1))
tsplot(index,xlab="months",ylab="Federal Reserve Board Production Index")
tsplot(noune,xlab="months",ylab="Number unemployed (in 10K)")
```

8. S functions for Figure 1.10.

```
tsplot(sunspots)
title("Monthly Mean Relative Sunspots, 1750-1975")
tsplot(lynx)        # lynx trapped data (not in the figure)
title("Canadian Lynx Trappings, 1821-1934")
```

9. S functions for Figure 1.11.

```
attach(lung)
par(mfrow = c(5, 1))
tsplot(atm,xlab="days")
title("Adjusted Total Mortality")
tsplot(smk,xlab="days")
title("British Smoke")
tsplot(so2,xlab="days")
title("Sulfur Dioxide")
tsplot(tmp,xlab="days")
title("Temperature in C")
tsplot(rhu,xlab="days")
title("Relative Humidity in %")
```

10. S functions for Figure 1.12.

```
attach(gasfurn)
tsplot(gasfurn,xlab="time")
title("Gas Furnace data")
```

11. Simulate time series using S functions.

```
tsplot(rnorm(200))      # Gaussian
title("200 N(0,1) white noise")
tsplot(rgamma(200,5))   # Gamma series
title("200 Gamma(5) white noise")
```

12. S-PLUS has many interesting time series. To see just a few, try

```
tsplot(rain.nyc1)
title("NY City precipitation, 1869-1957")
tsplot(rain.nyc2)
title("NY City precipitation, 1869-1957")
```

Problems

1. Compute the mean and variance of each of the series in Figure 1.1.
2. Compute the variance of each of the series (`ep`) in Figure 1.2.
3. Write a set of S functions for generating Figure 1.3.
4. Study the functions `rts` and `smooth` in S. Write a simple S function to smooth the `births` (Fig 1.6) and `sunspots` (Fig 1.10) series. [You may want to look at the definition of `as.rts` too.]
5. The data set for Figure 1.8 is called `eriel`, which can be obtained by typing `attach(eriel)`. (See also `eriel.d` in `/distrib/bios259/eriel.d` for the description of the data.) Plot the data and comment on the variation.
6. Plot the logarithm and square root of the `sunspots` data. Comment briefly on the peaks and the valleys in the transformed data.
7. We will discuss the problem of estimating trends and seasonal variation in time series in later chapters. For the moment, think about how to model the trend in Fig 1.9a by identifying the predictor and the response variables in a simple linear regression setting.
8. This exercise is based on the Birmingham mortality dataset `pm`.
   (a) Produce a scatterplot of the daily maximum temperature `tmax` vs specific humidity `mnsh`.
   (b) Produce a scatterplot of the daily maximum temperature `tmax2` vs specific humidity `tmax`.
   (c) Produce a scatterplot of the daily maximum temperature `tmax` vs specific humidity `pm10`.
   (d) Produce a scatterplot of the daily maximum temperature `tmax2` vs specific humidity `pm10`.

Chapter 2

Elements of time series

In this chapter, we describe the distribution of a time series. Since a time series is a collection of random variables indexed by time t, its distributional properties can be described in a manner that is closely related to that for stochastic processes. In order to do this, we need to define the *finite dimensional distribution* of a random process. This includes the marginal and bivariate distributions as special cases.

2.1 Finite dimensional distributions

Marginal distribution $F_t(\cdot)$

This is the distribution of the random variable $x(t)$ at time t. [To emphasize the time index, we sometimes denote a time series $\{x_t\}$ by $\{x(t)\}$.] That is, $F_t(a) = \mathsf{P}\{x(t) \leq a\}$. Thus the mean at time t, $\mu_t = \mu(t)$, is defined by $\mu_t = E(x_t) = \int x \, dF_t(x)$, and the variance at time t, $\sigma_t^2 = \sigma^2(t) = \text{var}(x_t) = E(x_t - \mu_t)^2 = \int (x - \mu_t)^2 \, dF(x)$. If the distribution has a probability density function $f_t(\cdot)$, then the mean and the variance are given by

$$\mu_t = \int x f_t(x) \, dx \qquad \text{and} \qquad \sigma_t^2 = \int (x - \mu_t)^2 f_t(x) \, dx.$$

Note that the mean and the variance of $x(t)$ depend on t. If t is continuous, we have infinite many means and variances. If t ranges from $1, \ldots, n$, then we have n means and variances.

Bivariate distribution $F_{t_1,t_2}(\cdot, \cdot)$

This is defined as the joint distribution of random variables $x(t_1)$ and $x(t_2)$. That is,

$$F_{t_1,t_2}(a_1, a_2) = \mathsf{P}\{x(t_1) \leq a_1, x(t_2) \leq a_2\}.$$

The covariance $C(t_1, t_2)$ of $x(t_1)$ and $x(t_2)$, defined by

$$C(t_1, t_2) = \mathrm{cov}(x(t_1), x(t_2)) = E[x(t_1) - \mu(t_1)][x(t_2) - \mu(t_2)],$$

is a very important quantity in time series analysis. In particular, $C(t, t) = \sigma^2(t)$. We also define the correlation coefficient between $x(t_1)$ and $x(t_2)$ by

$$\rho(t_1, t_2) = \frac{C(t_1, t_2)}{\sqrt{C(t_1, t_1)C(t_2, t_2)}}.$$

Question: How many covariance functions are there for continuous time observations? Discrete time observations?

Finite dimensional distributions $F_{t_1,\ldots,t_k}(a_1, \ldots, a_k)$

Denote the random variables at times $t_1, \ldots, t_k$ by $x(t_1), \ldots, x(t_k)$ and put

$$F_{t_1,\ldots,t_k}(a_1, \ldots, a_k) := \mathrm{prob}\{x(t_1) \leq a_1, \ldots, x(t_k) \leq a_k\}.$$

This is the joint distribution of $x(t_1), \ldots, x(t_k)$. The collection

$$\{F_{t_1,\ldots,t_k} : \quad t_i = 1, 2, \ldots\}$$

is called the finite dimensional distributions of the time series $x(\cdot)$.

According to Kolmogorov's theorem, the distribution of a random process $\{x_t\}$ is completely specified through its finite dimensional distributions. In particular, the study of the distribution of a time series in continuous time can be carried out in terms of the finite dimensional distributions. For the discrete time case, it would seem theoretically possible to study the whole distribution of $\{x_1, \ldots, x_n\}$. For example, one may assume that the random vector $\mathbf{x} = (x_1, \ldots, x_n)$ has a Gaussian (normal) probability density function given by

$$f(\mathbf{x}) = (2\pi)^{-n/2}|\mathbf{C}|^{-1/2} \exp\left\{-\tfrac{1}{2}(\mathbf{x} - \boldsymbol{\mu})'\mathbf{C}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right\},$$

where $\boldsymbol{\mu}$ and $\mathbf{C}$ are the mean and covariance of $\mathbf{x}$. But this is still far from being practical since there will be too many parameters to fit. Further reduction is needed. This leads us to consider a more parsimonious approach, based on notions such as stationarity and Markovian properties.

2.2 Stationary time series

There are two types of stationarities.

- *Strict sense:* Distributions that are not affected by shifting the time through a constant u, i.e.

$$F_{t_1,\ldots,t_k} = F_{t_1+u,\ldots,t_k+u} \qquad t_1, \ldots, t_k, u = 1, 2, \ldots.$$

- *Wide sense:* This is classified into stationary in mean or *first order stationary* in which
$$\mu(t) = \mu, \qquad t = 1, 2, \ldots.$$
Stationary in the covariance structure or *second order stationary*:
$$C(s,t) = C(s+u, t+u), \qquad s, t, u = 1, 2, 3, \ldots.$$
Although higher order stationarity can be defined similarly, it is sufficient (in this course) to examine the first two types of stationarity.

Setting $u = -s$, we have $C(s,t) = C(0, t-s)$. Thus for a second order stationary time series, the covariance structure can be specified by

$$C(u) := C(0,u) = \operatorname{cov}(x_t, x_{t+u}), \qquad u = 0, \pm 1, \pm 2, \ldots$$

or,

$$\rho(u) = \rho(0,u) = \operatorname{corr}(x_t, x_{t+u}), \qquad u = 0, \pm 1, \pm 2, \ldots.$$

Before we look at the examples, note that

- strict stationarity implies wide sense stationarity,
- stationarity is essential for estimating the distributions (at least for the marginal one) and the autocovariance function,
- strict stationarity is important for studying theoretical properties of a time series, while wide sense stationarity is useful for applications and methodological developments.

2.2.1 Examples of stationary time series

The blood pressure data (Figure 1.1), the EP data (Figure 1.2), the mortality data (Figures 1.4) and the speech data (Figure 1.5) appear to be stationary in mean and variance. What about the sunspots data (Figure 1.10)?

2.2.2 Examples of non-stationary time series

The hormone data (Figure 1.3), the US birth data (Figure 1.6), the magnetic exposure data (Figure 1.7), the Lake Erie data (Figure 1.8) and the unemployment data (Figure 1.9) are not stationary as the mean or the variance or both seem to vary with time. In particular, the births data is not stationary in mean but appears to be stationary in the variance.

From the above examples, we observe that non-stationarity is usually detected by examining the mean, or the variance, or both. In practice, one approaches this type of non-stationarity by using (a) regression methods to estimate and remove the

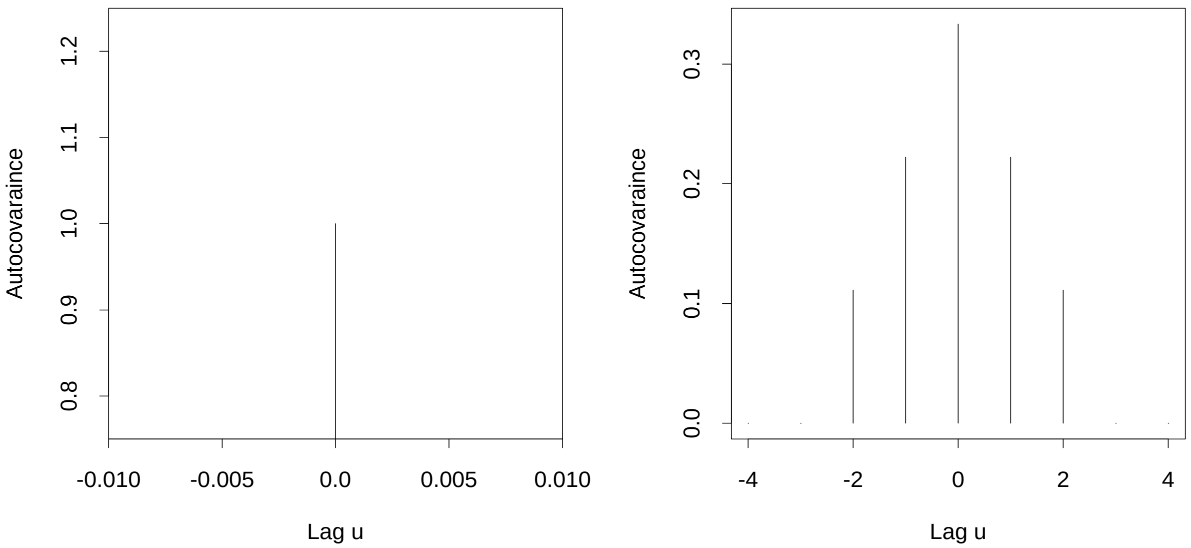


Fig 2.1. (a) *The autocovariance function of white noise.* (b) *The autocovariance function of MA(3).*

mean; (b) transformations to stabilize the variance. If the demeaned or transformed series appears to be stationary, then one can analyze the resulting data by using methods developed for stationary time series. Thus stationarity will be considered as our leading case in data analyses.

2.3 Stationary time series models

2.3.1 White noise

This is a sequence of independent and identically distributed (iid) random variables. It is usually abbreviated by

$$\{e_t : t = 1, 2, \ldots\}.$$

Note that this is a strict stationary time series. If these random variables have the common mean μ and variance σ^2, then $\mu(t) = \mu$ and

$$C(u) = \begin{cases} \sigma^2, & \text{if } u = 0; \\ 0, & \text{otherwise.} \end{cases}$$

The autocovariance function is plotted in Figure 2.1(a).

It is easy to simulate white noise using S-PLUS. For example,

```
tsplot(rnorm(400))       # Gaussian
tsplot(rgamma(400,1))    # Gamma series
```

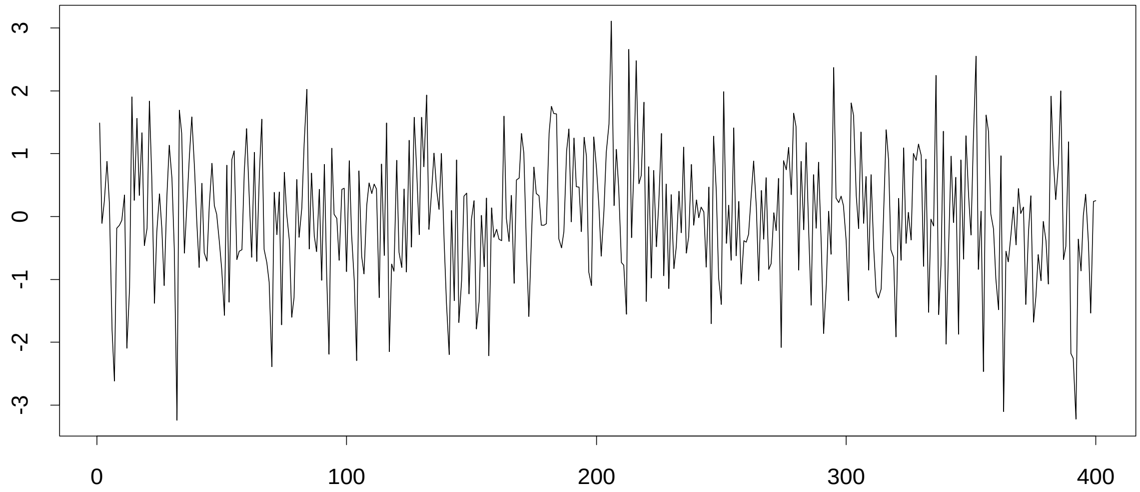


Fig 2.2a. *400 observations of Gaussian white noise.*

will simulate a Gaussian and an exponential series, respectively. See Figures 2.1a and 2.1b, respectively.

2.3.2 Moving averages (MA)

Strict stationary time series can be easily obtained from the white noise. A simple one is the moving average time series which is obtained by locally averaging the white noise series:

$$x(i) = [e_{i-1} + e_i + e_{i+1}]/3.$$

The mean of this series is μ and the autocovariance function is

$$C(u) = \begin{cases} \sigma^2/3, & \text{if } u = 0; \\ 2\sigma^2/9, & \text{if } |u| = 1; \\ \sigma^2/9, & \text{if } |u| = 2; \\ 0, & \text{if } |u| > 2. \end{cases}$$

This function is plotted in Figure 2.1(b).

The following S function will simulate the above moving average process with Gaussian noise ($n = 200$):

```
x <- arima.sim(200, model=list(ma=c(-1/3,-1/3,-1/3)))
tsplot(x)
```

The series is plotted in Figure 2.3.

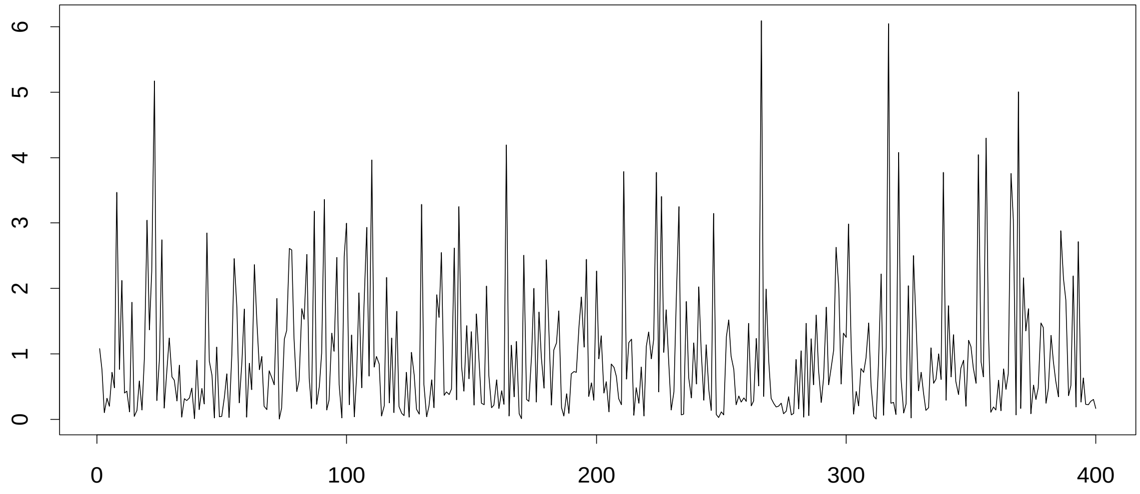


Fig 2.2b. *400 observations of exponential (1) white noise.*

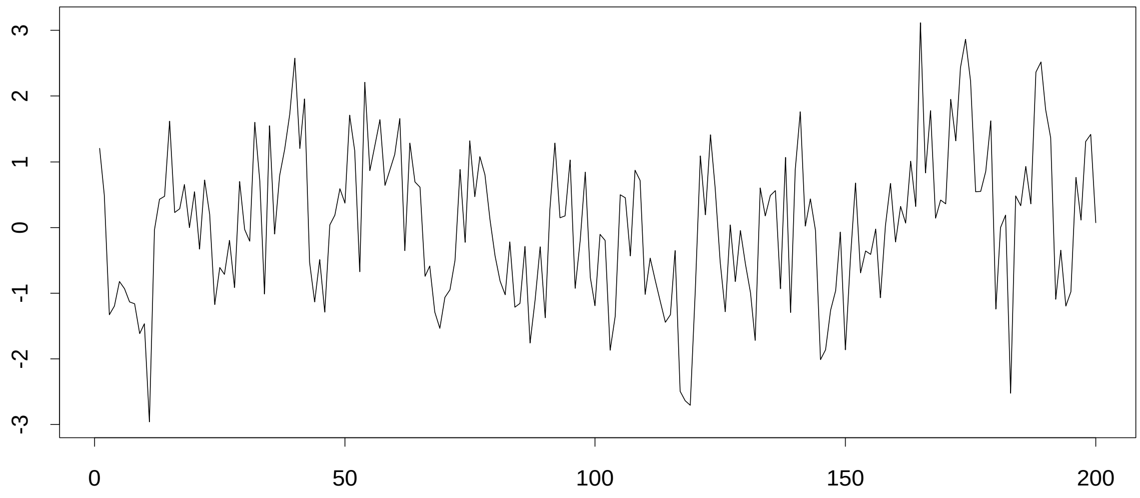


Fig 2.3. *200 MA (3) observations using Gaussian white noise.*

2.3.3 Autoregressive time series (AR)

This is a Markov process given by

$$x(t) = ax(t-1) + e(t).$$

Unlike the moving average time series, this AR process is stationary provided that $|a| < 1$ and that the process can be extended backward. Given $x_1, \ldots, x_n$, the observation at time t can be written as

$$\begin{aligned} x_t &= ax_{t-1} + e_t \\ &= a(ax_{t-2} + e_{t-1}) + e_t \\ &= \cdots \\ &= e_t + ae_{t-1} + a^2 e_{t-2} + \cdots + a^{t-1} e_1. \end{aligned}$$

Hence,

$$E(x_t) = \mu(1 + a + a^2 + \cdots + a^{t-1}) = \begin{cases} \mu(1-a^t)/(1-a), & \text{if } a \neq 1; \\ \mu t, & \text{if } a = 1. \end{cases}$$

Since the mean depends on t, the series is not stationary. However, if the white noise series has mean zero, then the AR series also has mean zero and is therefore first order stationary. From now on assume that $\mu = 0$.

Suppose $u \geq 0$. Then

$$\begin{aligned} \operatorname{cov}(x_t, x_{t+u}) &= E[(e_{t+u} + ae_{t+u-1} + \cdots + a^{t+u-1}e_1)(e_t + ae_{t-1} + \cdots + a^{t-1}e_1)] \\ &= \sigma^2(a^u + \cdots + a^{u+2(t-1)}) \\ &= \begin{cases} \sigma^2 a^u (1-a^{2t})/(1-a^2), & \text{if } |a| \neq 1; \\ \sigma^2 t, & \text{if } a = 1; \\ \sigma^2 t(-1)^u, & \text{if } a = -1. \end{cases} \end{aligned}$$

Again, the series is not second order stationary, but is asymptotically so. For if $|a| < 1$, then

$$\operatorname{cov}(x_t, x_{t+u}) \approx \sigma^2 a^u/(1-a^2).$$

Also,

$$\operatorname{var}(x_t) \approx \sigma^2/(1-a^2).$$

Thus

$$\rho(u) \approx a^u, \qquad u = 0, \pm 1, \pm 2 \ldots.$$

Note that these relationships can also be derived from the stationarity assumption. Namely, it follows from

$$x_t x_{t-u} = ax_{t-1}x_{t-u} + e_t x_{t-u}$$

that

$$C(u) = aC(u-1) = a^2 C(u-2) = \cdots = a^u C(0).$$

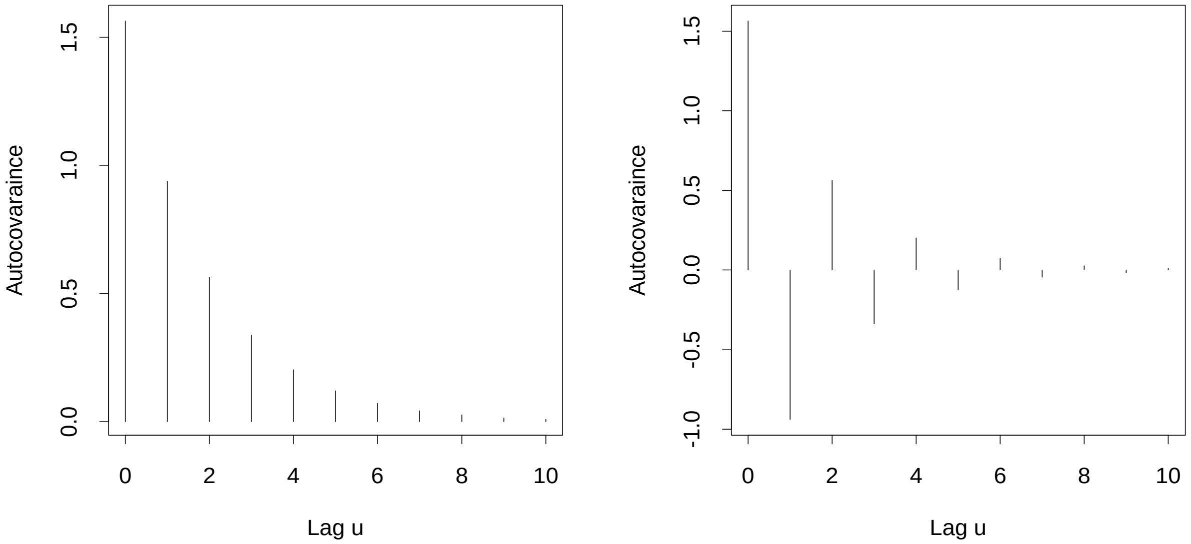


Fig 2.4. *The autocovariance function of AR(1):* (a) $a = 0.6$; (b) $a = -0.6$.

But $C(0) = \text{var}(x_t) = a^2\text{var}(x_{t-1}) + \sigma^2$ or, $C(0) = \sigma^2/(1 - a^2)$. Hence,

$$C(u) = a^u \sigma^2/(1 - a^2)$$

as has to be shown. See Figures 2.4 (a) and (b) for graphs of the function.

It is easy to simulate autoregressive processes using S-PLUS. For example,

```
x <- arima.sim(200, model=list(ar=c(-0.6)))
x <- arima.sim(200, model=list(ar=c(0.6)))
```

will simulate 200 AR(1) observations with $a = -0.6$ and $a = 0.6$, respectively. The series are plotted in Figures 2.5a and 2.5b.

Note that the consecutive observations in AR(1) with $a = -0.6$ are negatively correlated, while those for AR(1) with $a = 0.6$ are positively correlated; hence the series in Figure 2.5a is noisier than the one in Figure 2.5b. See also Figures 2.4 (a) and (b).

2.3.4 Gaussian time series

This is a very special time series, since its distribution can be completely specified through its mean and auto-covariance function $C(s, t)$. The function $C(s, t)$ can be further modeled by using moving-average and autoregressive processes, which provides a rich collection of models for the covariance structure. We will see later why this is called the *leading case* in time series.

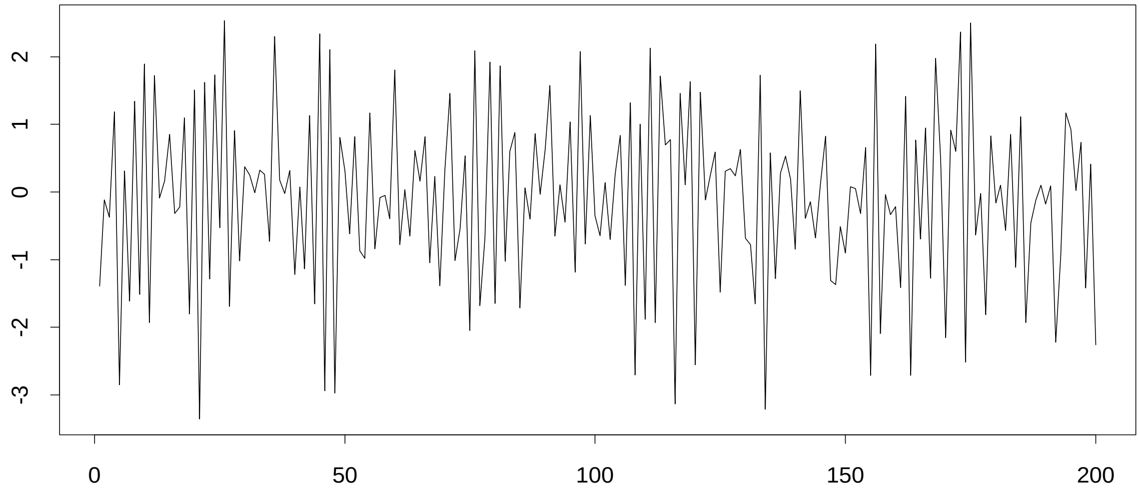


Fig 2.5a. *200 observations from AR(1):* $x_t = -0.6x_{t-1} + e_t$.

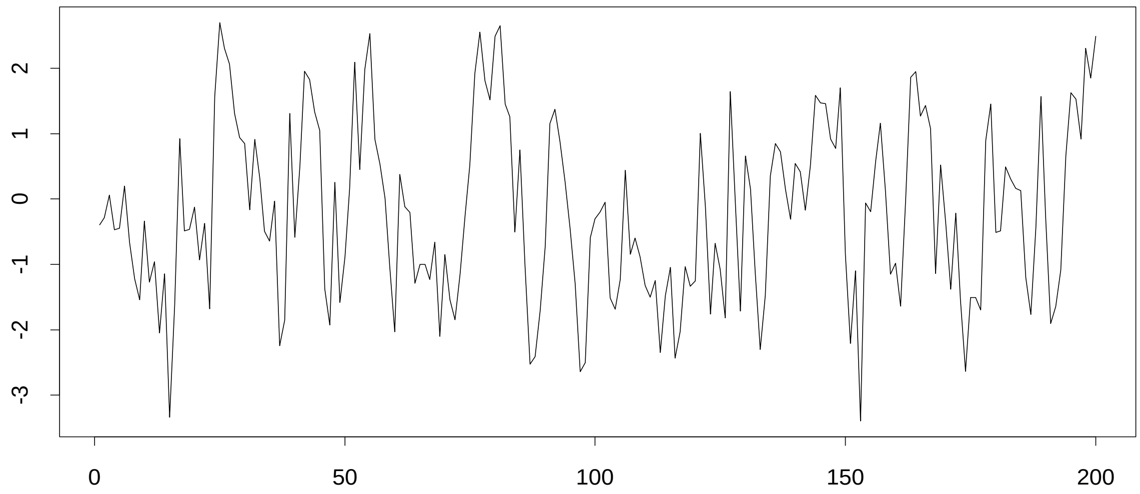


Fig 2.5b. *200 observations from AR(1):* $x_t = 0.6x_{t-1} + e_t$.

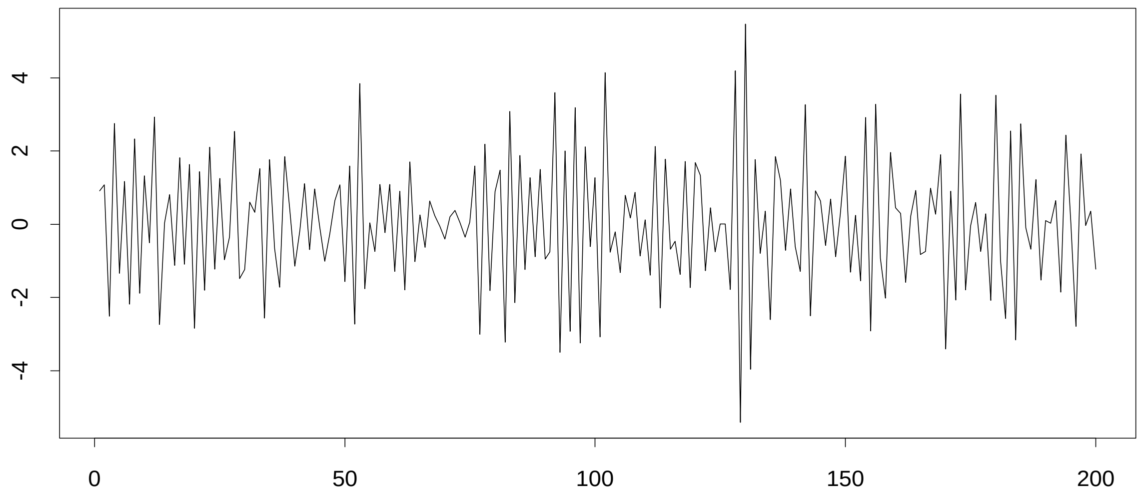


Fig 2.6. *200 observations from ARMA(1,1):* $x_t = -0.6x_{t-1} + e_t - 0.7e_{t-1}$.

There are many ways to simulate a Gaussian process with a specified covariance structure. For example, one can specify the autocovariance function using an autoregressive and moving average process (ARMA) as in

$$x_t + 0.6x_{t-1} = e_t - 0.7e_{t-1}.$$

This is easily done in S-PLUS via

```
x <- arima.sim(200, model=list(ar=c(-0.6),ma=c(.7)))
```

Figure 2.6 depicts 200 observations simulated from the above ARMA(1,1) model.

2.4 Bivariate time series

Let $(x_0, y_0), \ldots, (x_n, y_n)$ denote a bivariate time series. The cross covariance function is defined by

$$C(s,t) = E\{[x_s - E(x_s)][y_t - E(y_t)]\}.$$

Under the stationarity condition, we can examine the cross-covariance function

$$C_{xy}(u) = E\{(x_{t+u} - \mu_x)(y_t - \mu_y)\},$$

where $\mu_x = E(x_t)$ and $\mu_y = E(y_t)$. Note that the cross-covariance function is not symmetric but it satisfies the following relationship

$$C_{xy}(-u) = C_{yx}(u).$$

Similarly, the cross-correlation function is defined by

$$\rho_{xy}(u) = \frac{C_{xy}(u)}{\sqrt{C_x(0)C_y(0)}}.$$

2.5 Harmonic processes

Let Φ be a uniform random variable on $(-\pi, \pi)$. Let A and ν denote constants. Then the harmonic process is defined by

$$x_t = A\cos(2\pi\nu t + \Phi).$$

It can be shown that this is strictly stationary with mean

$$\mu(t) = \frac{A}{2\pi}\int_{-\pi}^{\pi}\cos(2\pi\nu t + \phi)\, d\phi = \frac{A}{2\pi}\sin(2\pi\nu t + \phi)\Big|_{-\pi}^{\pi} = 0, \qquad t = 1, 2, \ldots$$

and autocovariance function

$$\begin{aligned} C(u) &= \frac{A^2}{2\pi}\int_{-\pi}^{\pi}\cos(2\pi\nu t + \phi)\cos(2\pi\nu(t+u) + \phi)\, d\phi \\ &= \frac{A^2}{4\pi}\int_{-\pi}^{\pi}[\cos(2\pi\nu u) + \cos(\lambda(2t+u) + 2\phi)]\, d\phi \\ &= \frac{A^2}{2}\cos(2\pi\nu u), \qquad u = 0, \pm 1, \ldots. \end{aligned}$$

More generally, we can consider a sum of harmonic series

$$x_t = \sum_{k=1}^{p} A_k \cos(2\pi\nu_k t + \Phi_k),$$

where A_k are constants and Φ_k are independent uniform random variables on $[-\pi, \pi]$ for $k = 1, \ldots, p$. We have

$$C(u) = \frac{1}{2}\sum_{k=1}^{p} A_k^2 \cos(2\pi\nu_k u).$$

Figure 2.7 shows a realization of a sum of four simple harmonic functions. Note that the nice smooth periodic component of a simple harmonic process is replaced by a rather noisy one. This provides an example for using spectral method to study noisy data.

2.6 Convention

From now on, stationarity implies the series has a constant mean, i.e. $E(x_t) = \mu$, and the covariance function satisfies

$$C(s,t) = C(s+u, t+u), \qquad s, t, u = 0, \pm 1, \pm 2, \ldots.$$

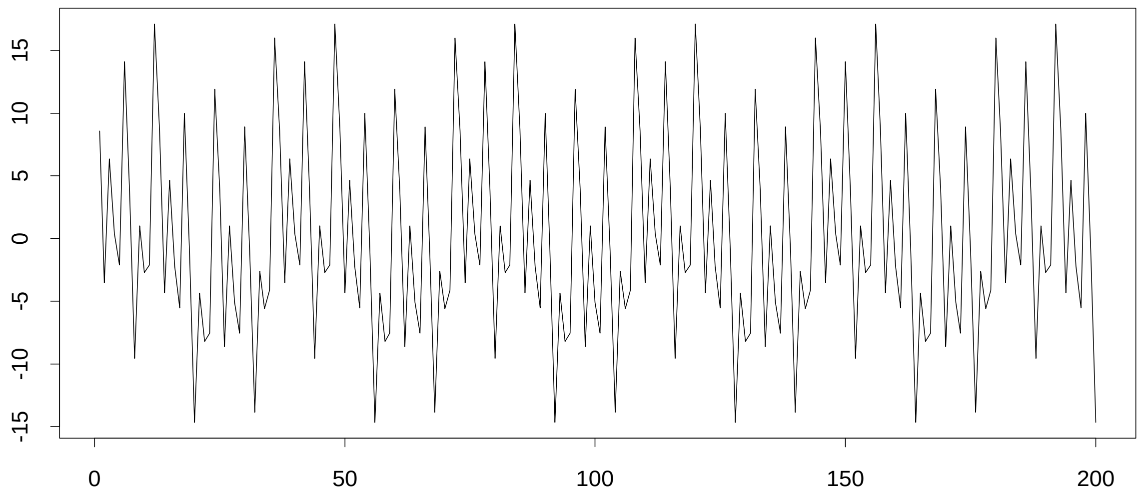


Figure 2.7. *Sum of 4 simple harmonic functions.*

2.7 More examples

1. Let

$$\begin{aligned} x_1, x_3, x_5, \ldots &\sim_{\text{iid}} \exp(1); \\ x_2, x_4, x_6, \ldots &\sim_{\text{iid}} N(1,1). \end{aligned}$$

   Then $C(0) = 1$ and $C(u) = 0$ for $|u| \geq 1$. This shows wide sense stationary is not necessarily stationary in the strict sense.

2. Let y_t be a stationary time series with mean 0 and autocovariance function $C(u)$. Define

$$x_t = \begin{cases} y_t, & \text{if } t \text{ is even,} \\ y_t + 1, & \text{otherwise.} \end{cases}$$

   Then $\text{cov}(x_{t+u}, x_t) = C(u)$. Comment briefly on the stationarity in terms of mean and covariance of the series x_t.

3. Consider the bivariate series

$$\begin{aligned} x_t &= ax_{t-1} + e_t, \qquad |a| < 1, \\ y_t &= bx_{t-1} + \epsilon_t, \end{aligned}$$

   where e_t and ϵ_t are independent $N(0, \sigma^2)$ and $N(0, \tau^2)$, respectively.

   - Find $\text{cov}(x_t, x_{t-2})$.

- Verify $\text{cov}(y_t, x_{t-1}) = b\sigma^2/(1-a^2)$.
- Verify $\text{cov}(x_t, y_{t-1}) = ba^2\sigma^2/(1-a^2)$.

2.8 Partial correlations

We have shown previously that the autocorrelation $\rho(u)$ of the MA(3) dies off when $|u| > 2$, while the AR(1) has no such clear cut-off point. But since x_t and x_{t+2} both depend on x_{t+1} linearly, the correlation between x_t and x_{t+2} adjusted for x_{t+1} is zero. In fact,

$$\text{cov}(x_t - ax_{t+1}, x_{t+2} - ax_{t+1}) = \text{cov}(x_t, e_{t+2}) - a\text{cov}(x_{t+1}, e_{t+2}) = 0.$$

The correlation coefficient between the adjusted variables is called the *partial correlation coefficient.* For the AR(1) model, the *partial autocorrelation function* is defined by

$$PACF(u) = \begin{cases} \rho(1)(=a), & \text{if u=1;} \\ 0, & \text{otherwise.} \end{cases}$$

2.9 Methods of time series analysis

What makes time series analysis interesting is that different series may require different methods and modeling strategies. This is caused by the difficulty arises in what we try to do statistical inferences based on only a stretch of series, which may be considered as consisting of some deterministic as well as random components. In any case, methods of analyses can be roughly grouped into two domains: The frequency and the time domains.

1. Frequency domain: View the whole series as the sum of many little sub-series called frequency components or harmonic series. See Figure 2.6. This is supported by the spectral representation of a stationary random process. High frequency variation is rapid vairation over short time intervals. Low frequency variation is slow variation over long time intervals.

2. Time domain: View the future observation as the sum of a linear combination of past values. This is supported by the Wold's theory (1938).

2.10 Writing S functions

This section illustrate how to write simple S functions. The syntax for a S function is

```
fun.name <- function(...)
{
...
}
```

where `fun.name` is the function name to be used by the user, `function` is one of S functions, (...) to be used for setting parameters. Below is a simple function for averaging every three consecutive observations of a gaussian series.

```
# Create a moving average process using gaussian white noise
# Lowpass filter
# L=length of the series

sim.ma <- function(a=c(1/3,1/3,1/3),L=200)
{
 x <- as.rts(rnorm(L))
 y <- filter(x,a)

 ts.plot(x,y)
}
```

Suppose this is stored in a file `simma`. We can use this function to generate and plot a moving average process as shown in the following.

```
% Splus
  ...
> X11()
> source("simma")
> sim.ma()    # Figure 2.8
```

The following will average every four consecutive observations of a time series of length 300.

```
> a <- rep(1/4,4)   # a = (0.25, 0.25, 0.25, 0.25)
> sim.ma(a,L=300)   # new a and L
```

The following performs a weighted average on every 21 consecutive observations.

```
> a  <- exp(-((-10:10)^2)/5)  # gaussian lowpass filter
> sim.ma(a,L=300)   # Figure 2.9
```

The file `expsmo` is a function for performing the exponential smoothing.

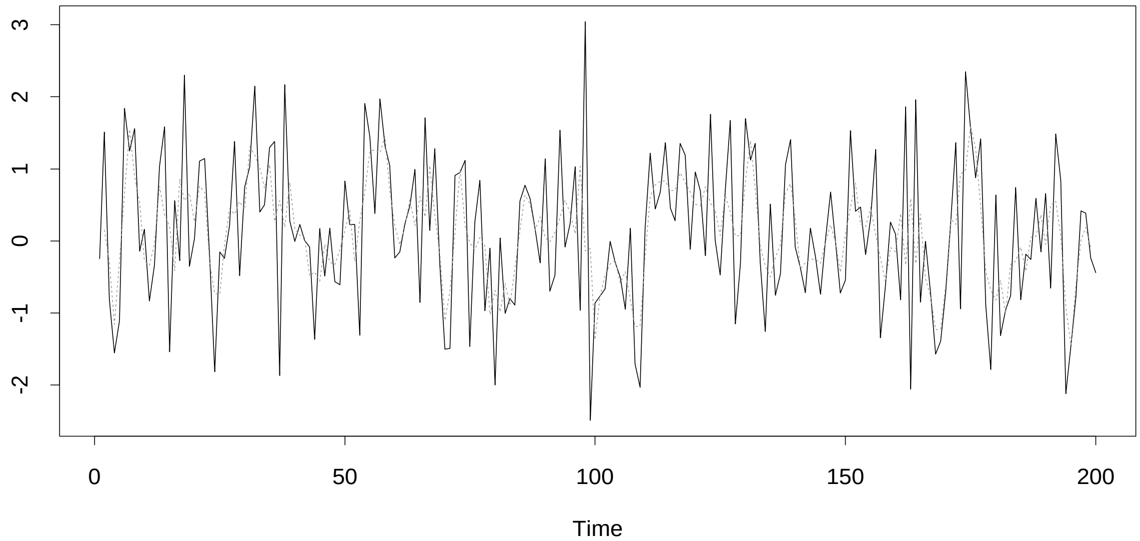


Fig 2.8. *An example of a lowpass filter on 200 gaussian white noise.*

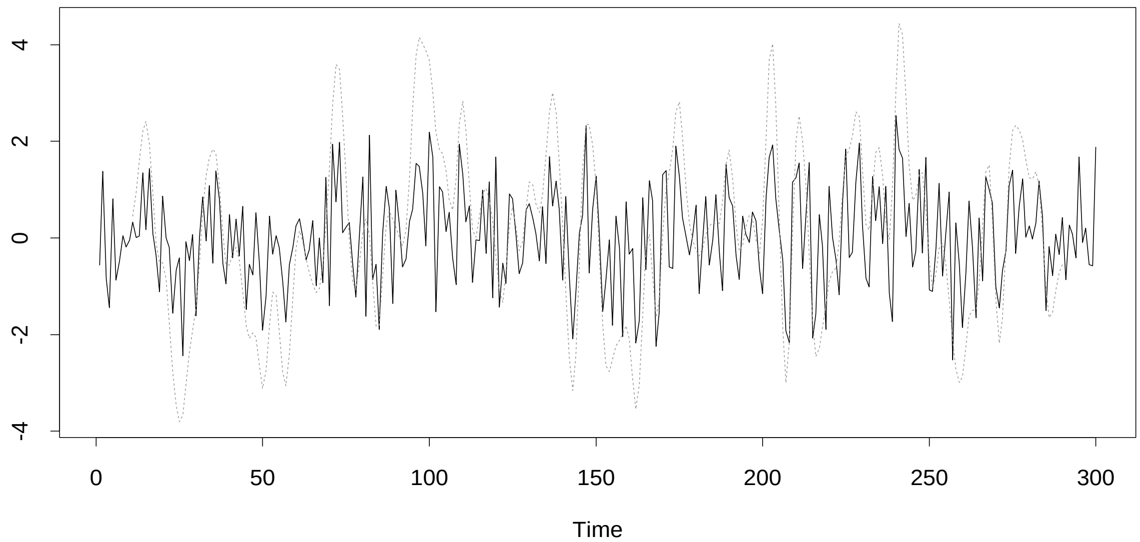


Fig 2.9. *An example of a gaussian lowpass filter.*

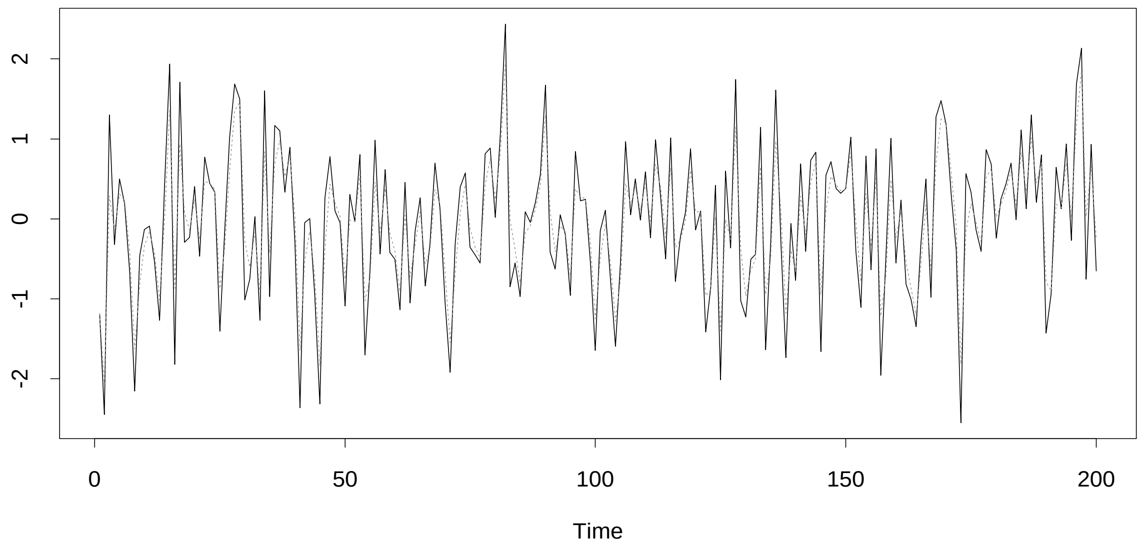


Fig 2.10. *An example of exponential smoothing.*

```
exp.smooth <- function(x, a)
{
 if(length(a) > 1)
        stop("a must be a single number")
 if(a > 1 || a <= 0)
        stop("a must be between zero and one")
 xa <- x * a
 xa[1] <- x[1]
 filter(xa, filter = 1 - a, method = "rec")
}
```

The following performs the exponential smoothing ($a = 0.7$) on a gaussian series.

```
> source("expsmo")
> x  <- rnorm(200)
> tsplot(x,exp.smooth(x,a=.7)) # Figure 2.10
```

The file `wn` is a function for simulating while noise time series with eight different distributions.

```
wn <- function(seed,n,dist=1)
#
#   Generate a white noise time series of length n having
#   distribution specified by dist. seed is a nonnegative
#   integer seed for the random number generator (if seed=0,
```

```
#   then the output seed from last call is used).
#
#   dist = 1: normal (0,1)
#   dist = 2: unif (0,1)
#   dist = 3: exponential (1)
#   dist = 4: logistic (mean 0 and var pi^2/3)
#   dist = 5: Cauchy
#   dist = 6: Extreme value (log(exp(1))
#   dist = 7: lognormal
#   dist = 8: double exponential
#
{
   if(seed!=0) set.seed(seed)

   if(dist==1) return(rnorm(n))

   u <- runif(n)

   if(dist==2) return(u)

   if(dist==3) return(-log(1-u))

   if(dist==4) return(log(u/(1-u)))

   if(dist==5) return(tan(pi*(u-.5)))

   if(dist==6) return(log(-log(1-u)))

   if(dist==7) return(exp(rnorm(n)))

   if(dist==8) { xx <- (1:n)[u>.5]; u[xx] <- 1-u[xx]; u <- log(2*u);
                 u[xx] <- -u[xx]; return(u);}
}
```

The series are plotted in Figure 2.11.

Problems

1. We can use `hist`, `boxplot` and `density` to examine the marginal distribution of a time series.

   (a) Simulate a Gaussian white noise series with mean 1 and variance 2, then use the above functions to study its distribution. Comment briefly.

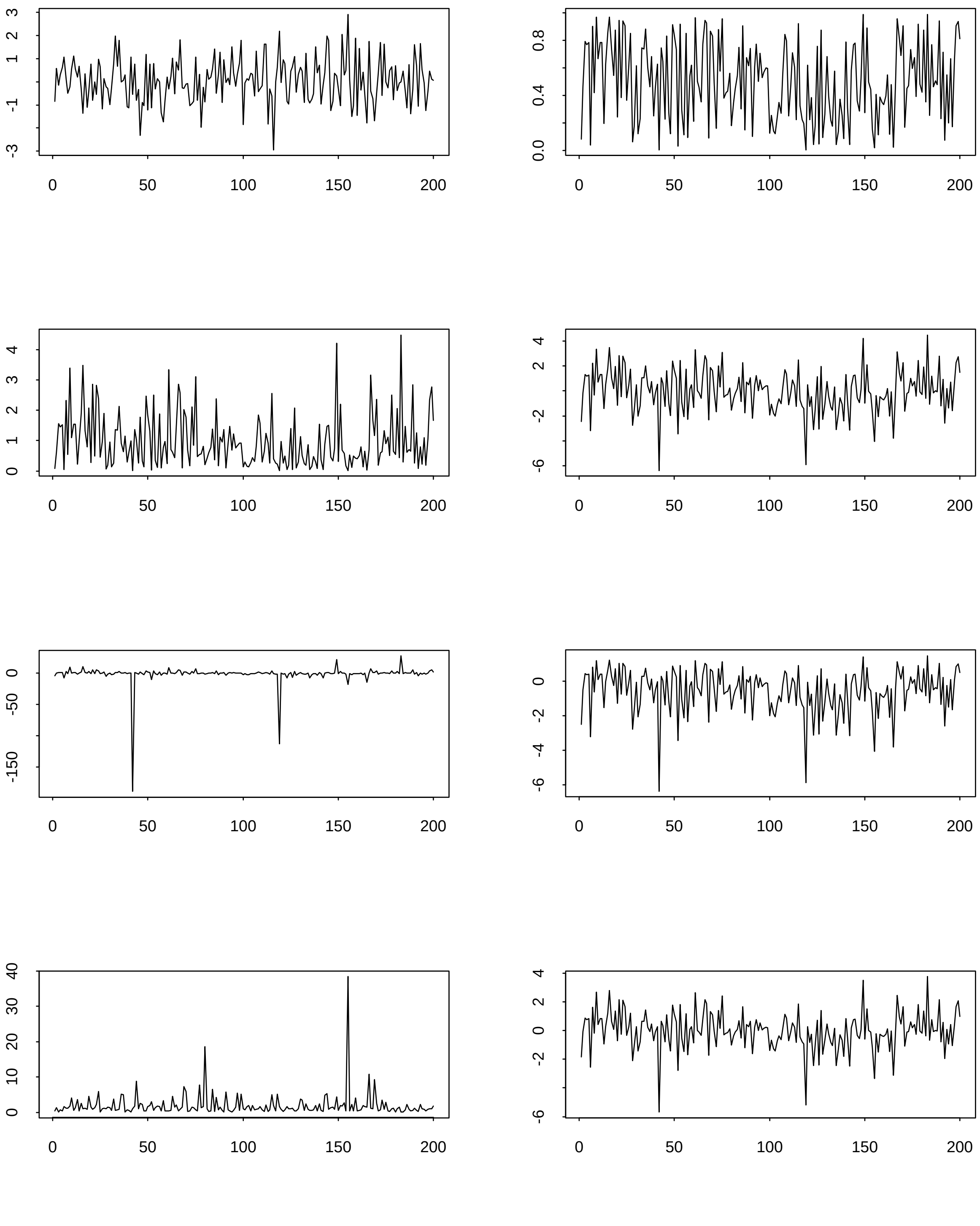


Fig 2.11. *Examples of white noise series.*

(b) Repeat the above question using a gamma series with mean 1.

(c) Repeat the above question using a cauchy series.

2. Use `hist`, `boxplot` and `density` to examine the marginal distribution of hormone data `lh`. Comment briefly. [Note that there are four series in that data set.] Hint. Check out the function `bandwidth.nrd` on p.138 of Venables and Ripley (1994).

3. Use `qqnorm` to check the normality of the blood pressure data `bp`. You may want to use `qqline` to draw a line for better visual effect.

4. Can the functions `hist`, `boxplot` and `density` be applied to examine the marginal distribution of the US Births data `births`? Why and why not?

5. Use `hist`, `boxplot` and `density` to examine the marginal distribution of the sunspot data. Comment briefly. [The sunspot data is available in S-PLUS.]

6. Use `qqnorm` and `qqline` to examine the normality of the sunspot data.

7. Is it possible to have a series that exhibits stationary variance, but nonstationary in the mean? Give an example to support your answer.

8. For the MA process of the form

$$x_t = e_{t-2} + 4e_{t-1} + 6e_t + 4e_{t+1} + e_{t+2},$$

where $\{e_t\}$ are uncorrelated with zero mean and variance 1. Determine the autocorrelation function and plot.

9. Let $\{e_t\}$ be independent Bernoulli random variables taking values $+1$ and -1 with probability $1/2$ each. Set

$$x_t = \mu + e_t - \theta e_{t-1},$$

where μ and θ are real-valued parameters.

(a) Show that $E(x_t) = \mu$.

(b) Verify that

$$c(u) = \begin{cases} 1+\theta^2, & u = 0, \\ -\theta, & u = \pm 1, \\ 0, & |u| \geq 2. \end{cases}$$

10. Let

$$\mathbf{A} = \begin{pmatrix} \sigma^2 & \eta \\ \eta & \sigma^2 \end{pmatrix}.$$

(a) State a simple condition in terms of η and σ^2 under which $\mathbf{A}$ is a covariance matrix of a normal random vector in $\mathbb{R}^2$.

(b) Under the condition you found in (a), describe how to generate observations from $N_2(0, \mathbf{A})$.

11. *Geometric series.* Verify that

$$s_n = 1 + a + a^2 + \cdots + a^n = \frac{1 - a^{n+1}}{1 - a}.$$

If $|a| < 1$, then

$$\lim_{n \to \infty} s_n = \frac{1}{1 - a}.$$

This is usually written as

$$1 + a + a^2 + \cdots = \frac{1}{1 - a}.$$

Chapter 3

Spectral analysis

Let Φ be a uniform random variable on $[-\pi, \pi]$ and ν be a constant. The simple stationary (random phase) harmonic process

$$x_t = A\cos(2\pi\nu t + \Phi)$$

was described and discussed in Chapter 2. Suppose A is also random. (This is sometimes referred to as the random amplitude harmonic process.) Using $\cos(u+v) = \cos(u)\cos(v) - \sin(u)\sin(v)$, the above process can be written as

$$x_t = a\cos(2\pi\nu t) + b\sin(2\pi\nu t),$$

where $a = A\cos(\Phi)$ and $b = -A\sin(\Phi)$. Note that $a^2 + b^2 = A^2$. If A and Φ are independent, then

$$E(a) = E(b) = E(ab) = 0, \qquad E(a^2) = E(b^2) = \frac{1}{2} \cdot E(A^2). \tag{3.1}$$

Thus x_t is stationary with mean zero and covariance function

$$C(u) = \frac{E(A^2)}{2}\cos(2\pi\nu u), \qquad u = 0, \pm 1, \pm 2, \ldots. \tag{3.2}$$

Since the sum of independent stationary processes is stationary, the simplest stationary model for a series containing more than a single frequency component is the sum of an ensemble of independent harmonic processes. In this chapter, we will study stationary processes using the spectral representation of the form

$$x_t = \sum_{k=1}^{p} A_k\cos(2\pi\nu_k t + \Phi_k),$$

where Φ_k are independent uniform random variables on $[-\pi, \pi]$, A_k and Φ_k are independent.

3.1 Preliminaries

3.1.1 Harmonic function and frequency

To describe oscillation, we usually start with the simple harmonic function

$$x(t) = a\sin(2\pi\nu t), \quad t = 0, \Delta, 2\Delta, \ldots,$$

where, a is called the *amplitude*; ν is called *frequency*, measured in cycles per unit time or *Hertz* (Hz) and Δ is the sampling interval (will be defined shortly). For examples, if $\nu = 3$ and the unit of the time t is second ($\Delta = 1$ second), then $x(\cdot)$ has three cycles per second.

The *period*, T, of an oscillating function is defined to be the length of time required for one full cycle. Since a complete cycle of $\sin\theta$ requires time 2π, the period T of the above harmonic function $x(\cdot)$ satisfies the equation

$$2\pi\,\nu\,T = 2\pi \qquad \text{or} \qquad T = 1/\nu.$$

So the period of an oscillating function is the reciprocal of the frequency. For example, the period of $x(t) = \sin(2\pi\,3t)$ is equal to $T = 1/3$. That is, it takes one third of a second to complete one full cycle.

Occasionally, frequency is described by the *angular frequency*,

$$\lambda = 2\pi\,\nu.$$

In this case, the unit of λ is *radian per unit time.*

3.1.2 Sampling and aliasing

Suppose the measurements are made at equally spaced points in time. Let Δ be the time interval. Then

$$\Delta = \frac{\text{time units}}{\text{sample}}$$

and the sampling rate is

$$r = \Delta^{-1} = \frac{\text{samples}}{\text{time units}}.$$

In sampling a time series in continuous time through digitization at time intervals $t = 0, \Delta, 2\Delta, \ldots$, one should observe the fact that the functions $\cos(2\pi\nu t)$ and $\cos 2\pi(\nu + s/\Delta)t$ are not distinguisable for any integer s. Thus the frequency ν can not be distinguished from the frequencies $\nu \pm s/\Delta$ for $s = 1, 2, \ldots$. This lack of identifiability is called *aliasing*. The frquencies $\nu \pm s/\Delta$ for $s = 1, 2, \ldots$ are aliases of one another.

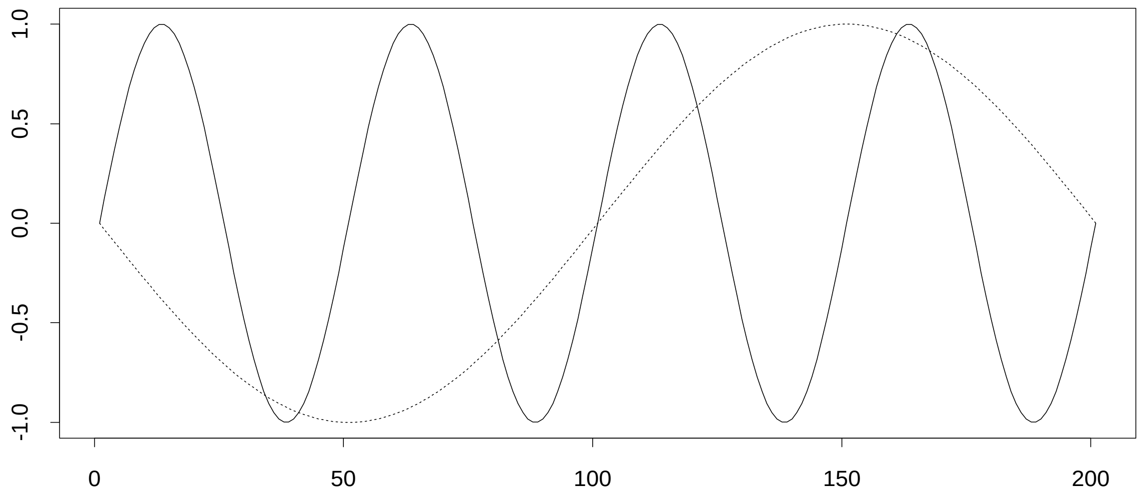


Figure 3.1. *Aliasing effect in sampling a time series in continuous time.*

3.1.3 Complex numbers

A complex number z is denoted by $z = x + yi$, where $i = \sqrt{-1}$ is the *imaginary unit*, x and y are the *real part* and *imaginary part* of z. The conjugate of z is defined by $\bar{z} = x - yi$. The *norm* (or modulus, or length) of $z = x + yi$ is defined by

$$|z| = \sqrt{z\bar{z}} = \sqrt{x^2 + y^2}.$$

Note that $i^2 = -1$. S-PLUS provides a nice environment for working with complex numbers. For example, a complex number can be assigned to a variable `z` using

```
z <- 3 + 2i
```

Then use `Re(z)` and `Im(z)` to get the real and imaginary parts of `z`. Try `sqrt(-1)` to see what happens. Then try again with `sqrt(as.complex(-1))`. Learn more about how to work with complex numbers by using `?Mod`, `?Re` and `?Im`.

3.1.4 Euler's formula — $e^{i\pi} + 1 = 0$

We have

$$\exp(i\alpha) = \cos\,\alpha + i \sin\,\alpha, \quad \exp(-i\alpha) = \cos\,\alpha - i \sin\,\alpha.$$

The use of complex exponential is extremely convenient in discussing the *fast Fourier transform*. Most of the computations involving the sine and cosine functions can be easily done using `fft` in S-PLUS.

3.2 Spectral representation

Figure 2.6 shows that noisy data can be constructed using a sum of harmonic processes. In fact, under some reasonable conditions, a stationary time series can be represented by

$$x_t = \int_{-1/2}^{1/2} e^{2\pi i\lambda t} Z(d\lambda),$$

which may be interpreted as a (infinite) sum of harmonic series

$$x_t \approx \sum_j e^{2\pi i\lambda_j t} Z(d\lambda_j)$$

with random amplitude $Z(d\lambda_j)$. The term $e^{2\pi i\lambda_j t} Z(d\lambda_j)$ is called the jth harmonic component.

3.3 Power spectra: Some heuristics

To see the importance of $Z(d\lambda_i)$, we will use the spectral representation to compute the autocovariance function:

$$c(u) = E(x_{t+u}\overline{x_t}) = \iint e^{2\pi i\lambda t} e^{-2\pi i\lambda'(t+u)} E(\overline{dZ(\lambda)} dZ(\lambda')).$$

Since, by stationrity, the autocovariance function depends only on u, we must have

$$E(\overline{dZ(\lambda)} dZ(\lambda')) = 0, \qquad \lambda' \neq \lambda.$$

Therefore

$$c(u) = \int e^{2\pi i\lambda u} E[|dZ(\lambda)|^2]. \tag{1}$$

In particular,

$$\mathrm{var}(x_t) = c(0) = \int E[|dZ(\lambda)|^2].$$

This basically says that the variance of the process x_t may be interpreted as the sum of the variances of the uncorrelated $dZ(\lambda_i)$. This is most conveniently abbreviated by

$$\mathrm{var}(x_t) = \int_{-1/2}^{1/2} f(\lambda)\, d\lambda.$$

The function $f(\lambda)$ is called the *power spectrum* or *spectral density* of the stationary process x_t and $f(\lambda)\, d\lambda$ is the variance of $Z(d\lambda)$.

According to (1), $c(u) = \int e^{2\pi iu\lambda} f(\lambda)\, d\lambda$ and hence the spectral density $f(\lambda)$ can be computed from the autocovariance function via

$$f(\lambda) = \sum c(u) e^{-2\pi iu\lambda}.$$

This is sometimes used as the definition of power spectra, as in the next section.

3.4 Definition of power spectrum

Let x_t denote a stationary time series with $\sum |c(u)| < \infty$. The *power spectrum* is defined by

$$f(\lambda) = \sum_{-\infty}^{\infty} c(u)e^{-2\pi i\lambda u}, \qquad \lambda \in (-1/2, 1/2).$$

Remarks

1. The power spectrum is symmetric about 0 and periodic with period 1: $f(\lambda+1) = f(\lambda)$. Therefore, in practice, it is sufficient to plot the function over the interval $[0, .5]$.

2. The λ in the above definition is called the *frequency* and is measured in cycles per unit time. Alternatively, the power spectrum can be defined by

$$f(\nu) = \frac{1}{2\pi}\sum_{-\infty}^{\infty} c(u)e^{-i\nu u}, \qquad \nu \in (-\pi, \pi),$$

   where ν is the angular frequency measured in radians per unit time.

3. The above definition shows that the spectral density function is determined by the autocovariance function. On the other hand, the autocovariance can be recovered from the spectral density function via

$$\begin{aligned} \int f(\lambda)e^{2\pi i\lambda v}d\lambda &= \int \sum_u c(u)e^{-2\pi i\lambda(u-v)}d\lambda \\ &= \sum_u c(u)\int e^{-2\pi i\lambda(u-v)}d\lambda \\ &= c(v). \end{aligned}$$

3.5 Linear filters

Suppose the input series x_t is related to the output series y_t via

$$y(t) = \sum_u a(t-u)x(u) = \sum_u a(u)x(t-u),$$

where $\{a(u)\}$ is called the *filter* of the system and $a(u)$ is called the *impulse response*. The importance of the spectral analysis can be seen by using the harmonic series as the input series $x_t = e^{2\pi i\lambda t}$. Then $x(t-u) = e^{2\pi i\lambda(t-u)}$ and

$$y(t) = e^{2\pi i\lambda t}\sum_u a(u)e^{-2\pi i\lambda u} = e^{2\pi i\lambda t}A(\lambda), \text{say}.$$

$A(\lambda)$ is called the *transfer function* of the impulse response function $a(u)$. Note that

$$A(\lambda) = \sum_u a(u) e^{-2pii\lambda u}, \quad a(u) = \int_{-1/2}^{1/2} A(\lambda) e^{2\pi i \lambda u}\, d\lambda.$$

Given an input series x_t with specral density $f_x(\cdot)$, we will show that the spectral density of the filtered series y_t can be obtained through the transfer function $A(\lambda)$. Write

$$\begin{aligned} y(t) &= \sum a(u) \int e^{2\pi i \lambda(t-u)} dZ_x(\lambda) \\ &= \int e^{2\pi i \lambda t} \left[\sum a(u) e^{-2\pi i \lambda u}\right] dZ_x(\lambda) \\ &= \int e^{2\pi i \lambda t} A(\lambda) dZ_x(\lambda). \end{aligned}$$

Since the spectral representation of the series $y(t)$ is

$$y(t) = \int e^{2\pi i \lambda t} dZ_y(\lambda).$$

We conclude that

$$dZ_y(\lambda) = A(\lambda) dZ_x(\lambda).$$

Therefore,

$$E|dZ_y(\lambda)|^2 = |A(\lambda)|^2 E|dZ_x(\lambda)|^2.$$

Or, equivalently,

$$f_y(\lambda) = |A(\lambda)|^2 f_x(\lambda).$$

Theorem 1 *Let f_x denote the spectral density of $\{x_t\}$. Suppose*

$$y(t) = \sum_u a(t-u) x(u) = \sum_u a(u) x(t-u).$$

Then the spectral density function f_y of $\{y_t\}$ is given by

$$f_y(\lambda) = |A(\lambda)|^2 f_x(\lambda),$$

where $A(\lambda) = \sum_u a(u) e^{-2\pi i \lambda u}$.

An important implication

We know that the value of $f(\lambda)$ has interpretation as the variance of the periodic component with frequency λ. Suppose the series y_t results by passing x_t through the linear filter with transfer function $A(\lambda)$. Then

$$f_y(\lambda) = |A(\lambda)|^2 f_x(\lambda)$$

and

$$\text{var}(y_t) = \int f_y(\lambda) d\lambda = \int |A(\lambda)|^2 f_x(\lambda) d\lambda.$$

Now if the filter is the band pass filter defined by

$$A(\lambda) = \begin{cases} (4\delta)^{-1/2}, & \text{if } |\omega \pm \lambda| \leq \delta, \\ 0, & \text{otherwise,} \end{cases}$$

then

$$\text{var}(y_t) \approx f_x(\omega).$$

This implies $f(\omega) \geq 0$ for $-\pi \leq \omega \leq \pi$.

3.6 Examples of power spectra

3.6.1 White noise

The autocovariance function of a white noise series is

$$c(u) = \begin{cases} \sigma^2, & \text{if } u = 0, \\ 0, & \text{otherwise.} \end{cases}$$

See Figure 3.2.

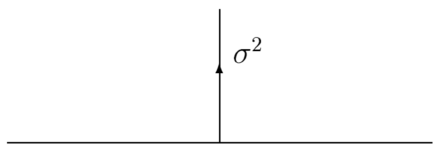


Figure 3.2. *The autocovariance function of a white noise.*

The power spectrum of a white noise series is

$$f(\lambda) = \sum c(u) e^{-2\pi i \lambda u} = \sigma^2.$$

Thus the spectrum of a white noise is a constant function. See Figure 3.3.

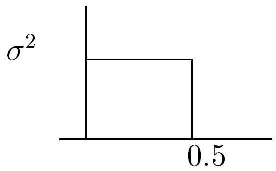


Figure 3.3. *The power spectrum of a white noise.*

3.6.2 $MA(3)$

For the moving average process

$$x_t = (e_{t-1} + e_t + e_{t+1})/3,$$

the spectral density can be obtained from the transfer function. Namely,

$$f_x(\lambda) = |A(\lambda)|^2 f_e(\lambda),$$

where $A(\lambda)$ is the transfer function of the filter

$$a(-1) = 1/3, a(0) = 1/3, a(1) = 1/3, \quad a(u) = 0, \quad \text{for } |u| > 1,$$

which is given by

$$A(\lambda) = (e^{2\pi i\lambda} + 1 + e^{-2\pi i\lambda})/3.$$

Consequently,

$$\begin{aligned} |A(\lambda)|^2 = A(-\lambda)A(\lambda) &= \frac{1}{9}(e^{2\pi i\lambda} + 1 + e^{-2\pi i\lambda})(e^{-2\pi i\lambda} + 1 + e^{2\pi i\lambda}) \\ &= [3 + 4\cos(2\pi\lambda) + 2\cos(4\pi\lambda)]/9. \end{aligned}$$

Therefore the spectral density of the series x_t is

$$f_x(\lambda) = \frac{\sigma^2}{9} \cdot [3 + 4\cos(2\pi\lambda) + 2\cos(4\pi\lambda)].$$

3.6.3 $AR(1)$

Consider the $AR(1)$ process

$$x_t - \phi x_{t-1} = e_t, \qquad e_t \sim_{\text{iid}} (0, \sigma^2).$$

The spectral densities of x_t and e_t are related through

$$f_x(\lambda)|A(\lambda)|^2 = f_e(\lambda),$$

where the transfer function $A(\lambda) = 1 - \phi e^{-2\pi i\lambda}$. Thus

$$f_x(\lambda) = \frac{\sigma^2}{1 + \phi^2 - 2\phi\cos(2\pi\lambda)}.$$

Figure 3.5 plots the power spectra of the two AR series.

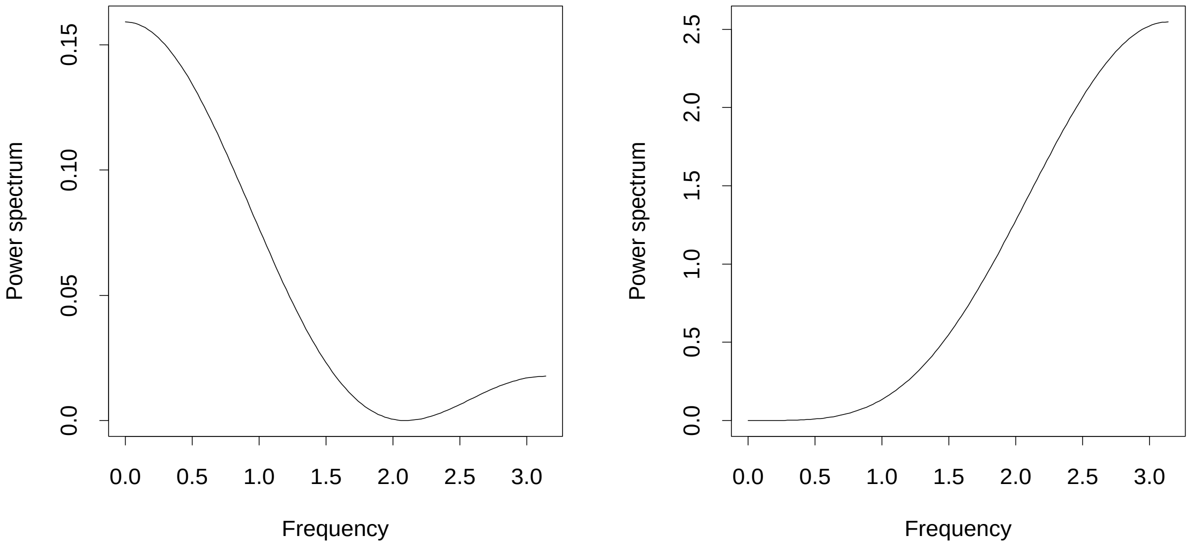


Figure 3.4. *Power spectra of MA(3).* (a) $\beta = (1/3, 1/3, 1/3)$ *lopass filter*, (b) $\beta = (1, -2, 1)$ *hipass filter.*

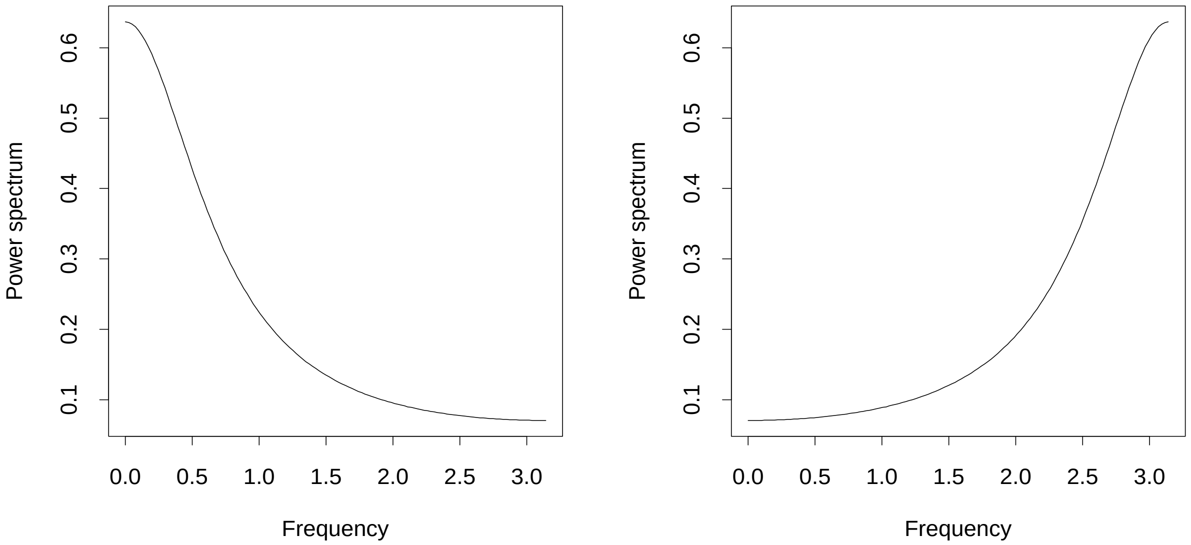


Figure 3.5. *Power spectra of AR(1).* (a) $\phi = 0.5$ *lopass filter*, (b) $\phi = -0.5$ *hipass filter.*

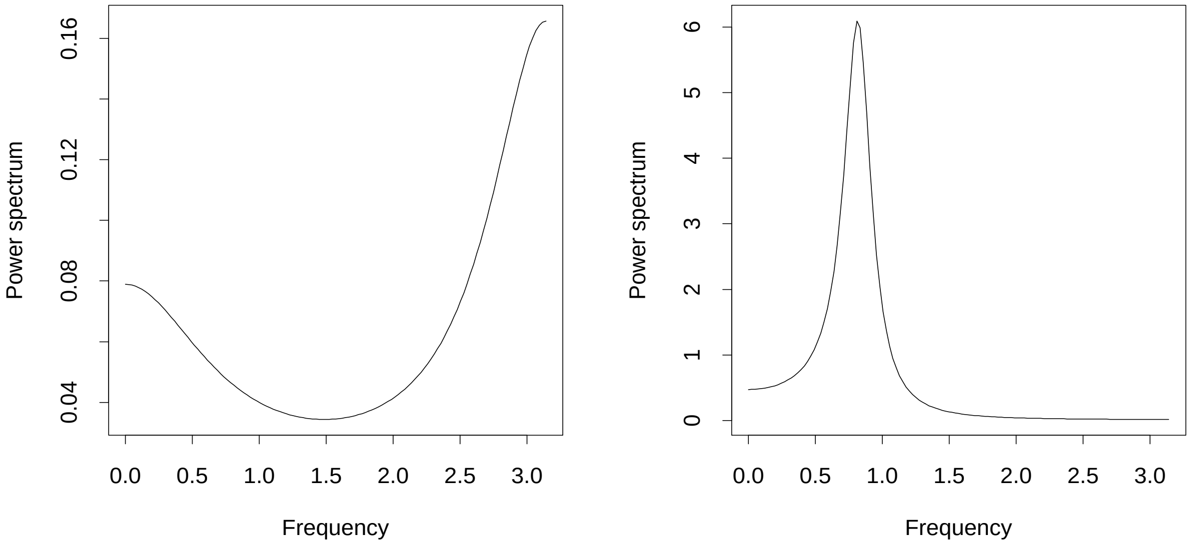


Figure 3.6. *Power spectrum of AR(2).* (a) $\phi = (-1.2, 0.78)$, (b) $\phi = (1.2, -0.78)$.

3.6.4 $AR(2)$

For the $AR(2)$ process:

$$x_t - \phi_1 x_{t-1} - \phi_2 x_{t-2} = e_t,$$

we have

$$A(\lambda) = (1 - \phi_1 e^{-2\pi i\lambda} - \phi_2 e^{-4\pi i\lambda}).$$

Thus

$$\begin{aligned} |A(\lambda)|^2 &= (1 - \phi_1 e^{-i\lambda} - \phi_2 e^{-i2\lambda})(1 - \phi_1 e^{i\lambda} - \phi_2 e^{i2\lambda}), \\ &= 1 + \phi_1^2 + \phi_2^2 - 2\phi_1(1-\phi_2)\cos(\lambda) - 2\phi_2\cos(2\lambda). \end{aligned}$$

The power spectrum is therefore given by

$$f_x(\lambda) = \frac{\sigma^2}{2\pi} \cdot \frac{1}{1 + \phi_1^2 + \phi_2^2 - 2\phi_1(1-\phi_2)\cos(\lambda) - 2\phi_2\cos(2\lambda)}.$$

Figure 3.6 shows the spectra of two different AR(2) processes.

3.6.5 Power spectra of harmonic processes

Recall that the simple harmonic process (monochromatic series) is given by

$$x_t = A\cos(2\pi\nu t + \Phi) = a\cos(2\pi\nu t) + b\sin(2\pi\nu t),$$

where ν is a constant, $a = A\cos(\Phi)$, $b = -A\sin(\Phi)$, Φ is a uniform random variable on $(-\pi, \pi)$, A and Φ are independent. The autocovariance function is

$$\begin{aligned} C(u) &= \frac{E(A^2)}{2}\cos(2\pi\nu u) \\ &= \frac{E(a^2+b^2)}{2}\cos(2\pi\nu u) \\ &= E(a^2)\cos(2\pi\nu u) \\ &= E(b^2)\cos(2\pi\nu u) \qquad u = 0, \pm 1, \pm 2, \ldots. \end{aligned}$$

It follows from $C(u) = \int \exp(it\lambda) f(\lambda)\, d\lambda$ that

$$f(\lambda) = \begin{cases} E(a^2), & \lambda = \pm\nu \\ 0, & \text{otherwise.} \end{cases}$$

Similarly, the harmonic process

$$x_t = \sum_1^p A_k \cos(2\pi\nu_k t + \Phi_k)$$

has a power spectrum

$$f_x(\lambda) = \begin{cases} \frac{E(A_k^2)}{2}, & \text{if } \lambda = \pm\nu_k, \quad k = 1, \ldots, p, \\ 0, & \text{otherwise.} \end{cases}$$

Figure 3.7 plots the power spectrum of the series presented in Figure 2.7. Note that the high frequency component has a dominated ordinate, this explains why the data are so noisy.

3.7 Spectral representation of real stationary process

Put $dZ(\lambda) = [a(\lambda) + ib(\lambda)]d\lambda$ and write

$$x_t = \int e^{i\lambda t}[a(\lambda) + ib(\lambda)]d\lambda.$$

Then

$$\overline{x_t} = \int e^{-i\lambda t}[a(\lambda) - ib(\lambda)]d\lambda = \int e^{i\lambda' t}[a(-\lambda') - ib(-\lambda')]d\lambda'.$$

x_t is real-valued, $x_t = \overline{x_t}$ implies

$$a(\lambda) = a(-\lambda), \quad b(\lambda) = -b(-\lambda).$$

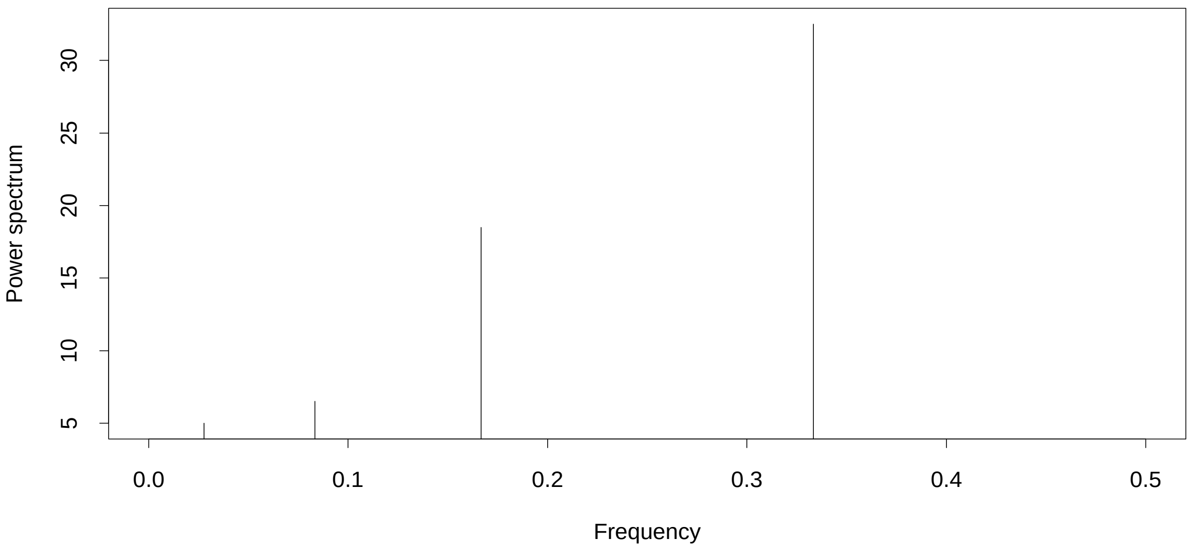


Figure 3.7. *Power spectrum of a harmonic process.*

Consequently,

$$\begin{aligned} x(t) &= \int [\cos(\lambda t) + i \sin(\lambda t)][a(\lambda) + ib(\lambda)] d\lambda \\ &= \int [a(\lambda)\cos(\lambda t) - b(\lambda)\sin(\lambda t)] d\lambda + i \int [a(\lambda)\sin(\lambda t) + b(\lambda)\cos(\lambda t)] d\lambda \\ &= \int [a(\lambda)\cos(\lambda t) - b(\lambda)\sin(\lambda t)] d\lambda, \end{aligned}$$

as $\int [a(\lambda)\sin(\lambda t) d\lambda = \int b(\lambda)\cos(\lambda t)] d\lambda = 0$ by virtue of (*).

3.8 S Examples

1. The function `cos.sim` simulates a harmonic process involving a few frequencies.

```
# Simulate a series consisting of 4 cosine functions
# a: cosine coefficients
# b: sine coefficients
# f: frequencies

cos.sim <- function(a=c(2,1,6,7),
               b=c(3,3,1,4),f=c(1/12,1/36,1/6,1/3),L=200)
{
```

```
 x <- seq(1,L)       # x = (1,2,...,L)
 y <- rep(0,L)       # an array of L zeros.

 for(j in 1:length(a))
   {
    y <- y+a[j]*cos(2*pi*f[j]*x)+b[j]*sin(2*pi*f[j]*x)
   }
 y
}
```

In S, a[j] denotes the jth entry of the array a. The following illustrates how to use this function.

```
> library(tslib)
> y <- cos.sim()     # generate the series
> X11()
> tsplot(y)          # plot it
```

Try your own examples by changing the parameters a, b and f.

2. The function armasp takes the parameters of an autoregressive and moving average process and plots the spetral density. The figures in this chapter were obtained using this function.

```
#
#   Find spectral density for
#    x(t) - phi(1)X(t-1) - ... - phi(p)x(t-p) =
#          + e(t) - beta(1)e(t-1) - ... - beta(q)e(t-q).
#
armasp <- function(phi=c(0), beta=c(0), sigma=1, Q=256)
{
 f1 <- f2 <- rep(1,Q)

 p <- length(phi)
 q <- length(beta)

 if (p!=0) f2 <- Mod(fft(c(1,-phi,rep(0,Q-p-1))))^2
 if (q!=0) f1 <- Mod(fft(c(1,-beta,rep(0,Q-q-1))))^2

 f <- (sigma * f1) / (f2 *2*pi)

 f[1:((Q / 2) + 1)]
}
```

The function `fft` will be discussed in the next chapter. Figure 3.6 was generated via the followings

```
par(mfrow=c(1,2))
x <- pi*(0:128)/128
f <- armasp(alpha=c(1.2,-.78))
plot(x,f,xlab="Frequency",ylab="Power spectrum",type="l")
f <- armasp(alpha=c(-1.2,.78))
plot(x,f,xlab="Frequency",ylab="Power spectrum",type="l")
```

Problems

1. Write a S function that simulates the process

$$y_t = a \cdot \cos(2\pi\nu t) + e_t, \qquad e_t \sim N(0, \sigma^2).$$

Design this function so that a, ν, σ and the length of the series are the parameters. How would you generate a series of length 200 that exhibits 7 cosine cycles?

2. Use `arima.sim` to simulate the process $x_t - \phi x_{t-1} = e_t$ using $\phi = .9, -.9$. Plot the power spectrum of this series using `armasp`. Discuss briefly.

3. Use `arima.sim` to simulate the process $x_t + .3x_{t-1} = e_t + .9e_{t-1}$. Plot the power spectrum of this series and discuss briefly.

4. Verify (3.1) and (3.2).

5. Plot the power spectrum of the MA process

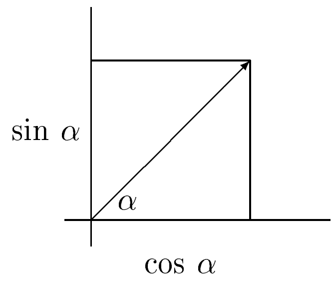
$$x_t = e_{t-2} + 4e_{t-1} + 6e_t + 4e_{t+1} + e_{t+2}.$$

Compare with the autocorrelation function and discuss briefly.

3.9 Appendix

3.9.1 Trigonometric identities

Let (x, y) denote the coordinates of a point on a unit circle centered at the origin. Draw a vector from the origin to the point (x, y). Denote the angle between this vector and the x-axis by α. Then $x = \cos\,\alpha$ and $y = \sin\,\alpha$.



The following are useful identities:

$$\sin(k \cdot \pi) = 0, \qquad k = 0, \pm 1, \pm 2, \ldots,$$

$$\sin((4k+1) \cdot \pi/2) = 1, \qquad k = 0, \pm 1, \ldots,$$
$$\sin((4k-1) \cdot \pi/2) = -1, \qquad k = 0, \pm 1, \ldots,$$
$$\cos((2k+1) \cdot \pi/2) = 0, \qquad k = 0, \pm 1, \ldots,$$
$$\cos(2k \cdot \pi) = 1, \qquad k = 0, \pm 1, \ldots,$$
$$\cos((2k+1) \cdot \pi) = -1, \qquad k = 0, \pm 1, \ldots,$$
$$\sin(\alpha + 2k \cdot \pi) = \sin(\alpha),$$
$$\cos(\alpha + 2k \cdot \pi) = \cos(\alpha),$$
$$\cos^2(\alpha) + \sin^2(\alpha) = 1,$$
$$\sin(\alpha \pm \beta) = \sin(\alpha)\cos(\beta) \pm \sin(\beta)\cos(\alpha),$$
$$\cos(\alpha \pm \beta) = \cos(\alpha)\cos(\beta) \mp \sin(\alpha)\sin(\beta),$$
$$2\cos(\alpha)\cos(\beta) = \cos(\alpha+\beta) + \cos(\alpha-\beta),$$
$$2\sin(\alpha)\sin(\beta) = \cos(\alpha-\beta) - \cos(\alpha+\beta).$$

3.9.2 Orthogonality of sinusoids

Let $\nu_j = (j-1)/n$ for $j = 1, \ldots, [n/2]+1$. Then

$$\sum_{t=1}^{n} \cos\, 2\pi(t-1)\nu_j\, \cos\, 2\pi(t-1)\nu_k = \begin{cases} 0, & \nu_j \neq \nu_k, \\ n, & \nu_j = \nu_k = 0, 0.5, \\ n/2, & \nu_j = \nu_k \neq 0, 0.5. \end{cases} \tag{3.3}$$

$$\sum_{t=1}^{n} \sin\, 2\pi(t-1)\nu_j\, \sin\, 2\pi(t-1)\nu_k = \begin{cases} n/2, & \nu_j = \nu_k \neq 0, 0.5, \\ 0, & \text{otherwise.} \end{cases} \tag{3.4}$$

$$\sum_{t=1}^{n} \cos\, 2\pi(t-1)\nu_j\, \sin\, 2\pi(t-1)\nu_k = 0, \qquad \nu_j, \nu_k \quad j,k = 1, \ldots, [n/2]+1. \tag{3.5}$$

$$\sum_{t=1}^{n} \cos 2\pi(t-1)\nu_j = \begin{cases} n, & \nu_j = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{3.6}$$

$$\sum_{t=1}^{n} \sin 2\pi(t-1)\nu_j = 0, \qquad \nu_j, \quad j = 1, \ldots, [n/2]+1. \tag{3.7}$$

Chapter 4

Descriptive Statistics

4.1 Sample autocovariance function

Given a time series $x_1, x_2, \ldots, x_n$. We define the sample autocovariance function as

$$\begin{aligned} \hat{c}(u) \equiv \hat{c}^n(u) &= \frac{1}{n}\sum_{t=1}^{n-u}(x_{t+u} - \bar{x})(x_t - \bar{x}) \\ \hat{c}(-u) &= \hat{c}(u), \end{aligned}$$

where

$$\bar{x} = \frac{1}{n}\sum_{t=1}^{n} x_t.$$

Similarly, the sample autocorrelation function is defined by

$$\hat{\rho}(u) = \frac{\hat{c}(u)}{\hat{c}(0)}.$$

In practice, the sample autocorrelation function is widely used than the autocovariance function because of the following result.

Theorem 2 *If $\{x_t\}$ is the stationary process*

$$x_t - \mu = \sum a_j e_{t-j}, \qquad e_j \sim_{\text{iid}} (0, \sigma^2),$$

with $\sum |a_j| < \infty$ and $\sum |j| a_j^2 < \infty$, then

$$(\hat{\rho}(1), \ldots, \hat{\rho}(u))' \sim_a N((\rho(1), \ldots, \rho(u))', W/n), \quad u = 1, 2, \ldots,$$

where the $u \times u$ matrix $W = (w_{ij})$ is given by Bartlett's formula

$$w_{ij} = \sum_{k=1}^{\infty}[\rho(k+i) + \rho(k-i) - 2\rho(i)\rho(k)][\rho(k+j) + \rho(k-j) - 2\rho(j)\rho(k)].$$

4.2 Applications

The following examples illustrate some applications of Theorem 1.

1. For the white noise series: $x_t \sim_{\text{iid}} (0, \sigma^2)$, we have

$$w_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

   For large n, $\hat\rho(1), \ldots, \hat\rho(u)$ are approximately independent $N(0, n^{-1})$ random variables. Thus, a 95% confidence interval is given by $\pm 1.95/\sqrt{n}$.

2. For the moving average process MA(q):

$$x_t = e_t + b_1 e_{t-1} + \cdots + b_q e_{t-q}, \qquad e_t \sim_{\text{iid}} (0, \sigma^2),$$

   we have

$$w_{ii} = 1 + 2\rho^2(1) + 2\rho^2(2) + \cdots + 2\rho^2(q), \quad i > q,$$

   as the asymptotic variance of $\sqrt{n}\hat\rho(i)$.

3. The asymptotic variance of $\hat\rho(i)$ for the $AR(p)$ is quite complicated. However for the AR(1) process $x_t - a x_{t-1} = e_t$, it is given by

$$w_{ii} \approx (1 + a^2)/(1 - a^2), \qquad i \gg 1.$$

   When dealing with the AR process, the PACF should be used instead of ACF. We will discuss the distribution of PACF in the estimation of parametric models.

4. In SPLUS, the function `acf` estimates and displays autocovariance, autocorrelation or partial autocorrelation functions. For example,

```
> ?acf          # Find out more about acf
> acf(lh)       # lh: 4 x 4 ACF plots
```

   displays 4 by 4 plots of autocovariance and cross-covariance functions of the series in the time series `lh`. The autocovariance functions of each series are on the main diagonal line. There are four arguments in `acf`. Namely,

```
acf(x, lag.max, type="correlation", plot=T)
```

   Here `x` is a univariate or multivariate time series. `lag.max` is the maximum lag to be used in computing the covariance functions. Set the character string in `type` to `"covariance"` for computing the covariance estimates, `"correlation"` for the autocorrelation function, or `"partial"`, if the partial autocorrelation function is desired. `plot=T` or `plot=F` set the plot on or off. If it is `T` (true), the autocovariance or autocorrelation function between pairs of univariate series will be plotted in an array of at most 5 by 5 plots per page. If type is "correlation" or "partial" approximate 95% confidence limits are drawn on the plots.

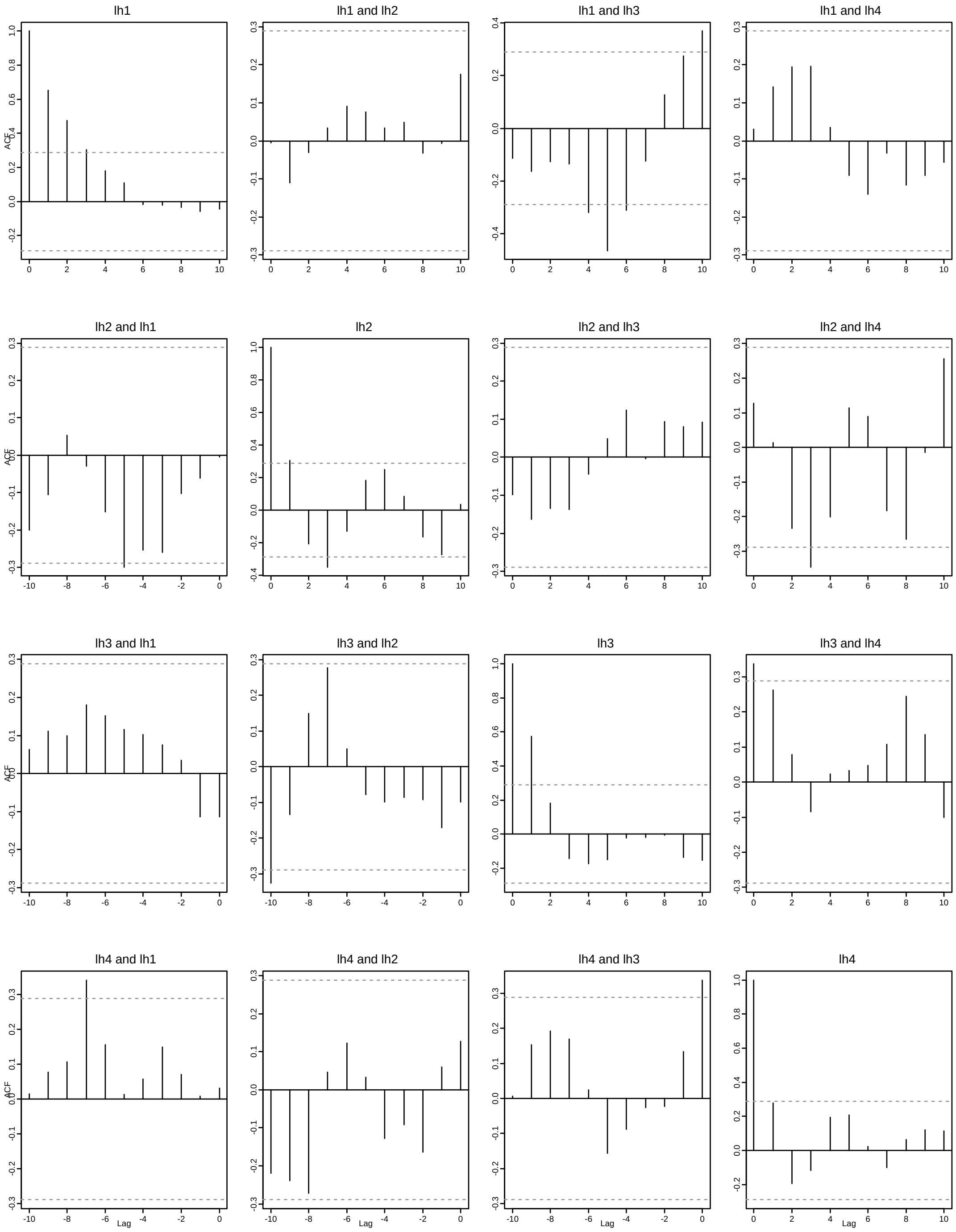


Fig 4.1 *Autocovariance functions for* `lh`*.*

5. You may find the function `lag.plot` useful when you are examining the autocovariance function.

```
> attach(lh)
> lag.plot(lh2,lags=4,layout=c(2,2))  # Fig 4-2
```

yields four scatterplots using lag =1,2,3,4.

4.3 Estimation of power spectrum

The estimate is given by the sample version of the power spectrum:

$$\hat{f}(\lambda) = \sum_{|u|<n} \hat{c}(u)e^{-2\pi i u\lambda}, \qquad \lambda \in [0,1].$$

Since the power spectrum is a nonnegative function, the estimate should have the same property. Thus the estimate $\hat{c}(u)$ should be chosen to reflect this.

4.3.1 Choice of $\hat{c}(u)$

To simplify our discussion, suppose the mean $\mu = 0$ and $x_1, \ldots, x_n$ denotes a realisation. Then we have two estimates of $c(u)$

$$\hat{c}^*(u) = \frac{1}{n-|u|} \sum_{t=1}^{n-|u|} (x_t - \bar{x})(x_{t+|u|} - \bar{x}) \tag{4.1}$$

and

$$\hat{c}(u) = \frac{1}{n} \sum_{t=1}^{n-|u|} (x_t - \bar{x})(x_{t+|u|} - \bar{x}) \tag{4.2}$$

Note that

$$E(\hat{c}^*(u)) = c(u), \qquad E(\hat{c}(u)) = (1 - \frac{|u|}{n})c(u).$$

Hence (4.1) is unbiased while (4.2) is not. When the stationary process has a smooth spectrum, the autocovariance function satisfies

$$\hat{c}(u) \to 0 \quad \text{as } |u| \to \infty.$$

So the bias of $\hat{c}(u)$ is negligible. Moreover, $\mathrm{var}(\hat{c}(u)) = O(1/n)$ while $\mathrm{var}(\hat{c}^*(u)) = O(1/(n-|u|))$. For large value of $|u|$, the unbiased estimate would have a large variance, hence resulting in a larger mean squared error compared to the biased estimate $\hat{c}(u)$.

It can be shown that the estimate of the power spectrum constructed using the biased estimate $\hat{c}(u)$ is a non-negative definite function. In contrast, the unbiased one would not have such a property. So a natural choice of the estimate of $R(u)$ is $\hat{c}(u)$ in estimating the spectral density function.

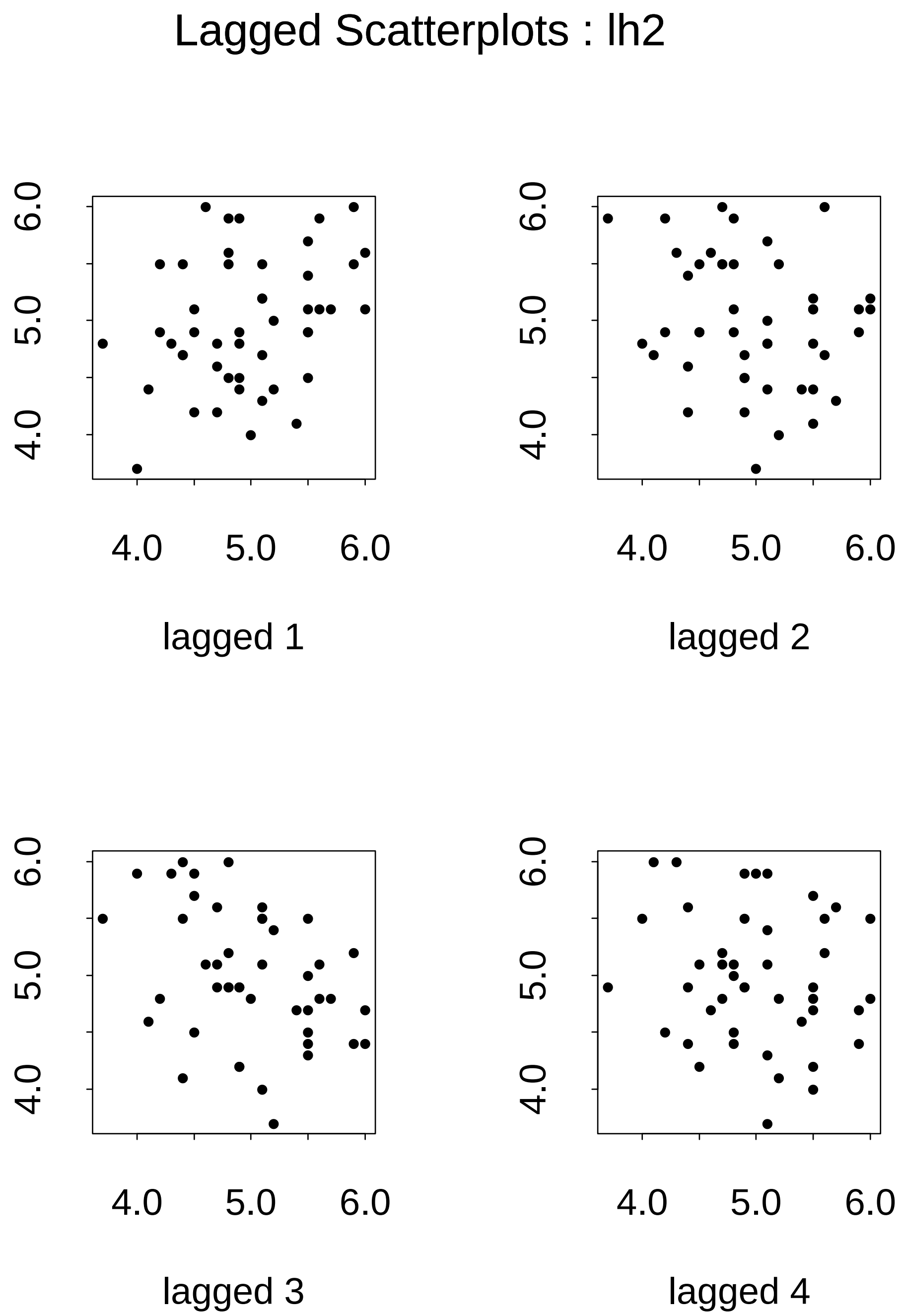


Fig 4.2 *Lag scatterplots.*

4.3.2 Why Fourier Transformation?

This can be explained via the following result.

$$\hat{f}(\lambda) = \left(n^{-1/2} \sum_{t=1}^{n} (x_t - \bar{x}) e^{2\pi i (t-1)\lambda} \right) \left(n^{-1/2} \sum_{s=1}^{n} (x_s - \bar{x}) e^{-2\pi i (s-1)\lambda} \right).$$

In fact, according to the definition of the spectral density estimates:

$$\hat{f}(\lambda) = \sum \hat{c}(u) e^{-2\pi i u\lambda} = \sum_{|u|<n} \left(\frac{1}{n} \sum_{t=1}^{n-|u|} (x_t - \bar{x})(x_{t+|u|} - \bar{x}) \right) e^{-2\pi i u\lambda}. \tag{4.3}$$

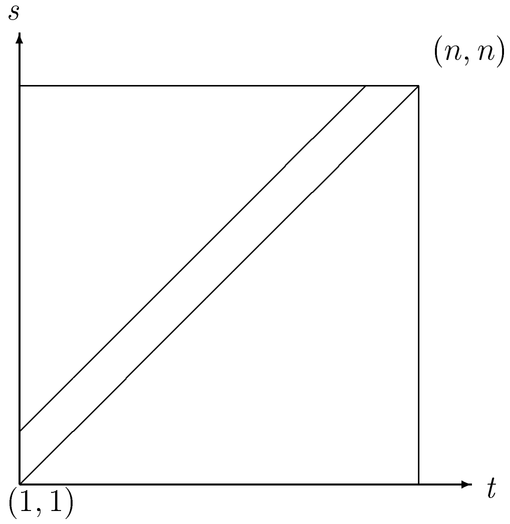


Fig 4. Sum over the diagonals.

Observe that the above double sum can be viewed as the sum over the diagonal as shown in Figure 4. Now rearrange the sum by summing over columns and then adding the column sums as follows.

$$\sum_{0<u<n} \sum_{t=1}^{n-u} (x_t - \bar{x})(x_{t+u} - \bar{x}) e^{2\pi i u\lambda} = \sum_{t=1}^{n-1} \sum_{s=t+1}^{n} (x_t - \bar{x})(x_s - \bar{x}) e^{2\pi i (s-t)\lambda}, \tag{4.4}$$

and

$$\sum_{-n<u\leq 0} \sum_{t=1}^{n+u} (x_t - \bar{x})(x_{t+u} - \bar{x}) e^{2\pi i u\lambda} = \sum_{t=1}^{n} \sum_{s=1}^{t} (x_t - \bar{x})(x_s - \bar{x}) e^{2\pi i (s-t)\lambda}. \tag{4.5}$$

By (4.3)–(4.5),

$$
\begin{aligned}
\hat{f}(\lambda) &= \frac{1}{n}\sum_{t=1}^{n}\sum_{s=1}^{n}(x_t-\bar{x})(x_s-\bar{x})e^{2\pi i(s-t)\lambda} \\
&= \left(n^{-1/2}\sum_{t=1}^{n}(x_t-\bar{x})e^{2\pi i t\lambda}\right)\left(n^{-1/2}\sum_{s=1}^{n}(x_s-\bar{x})e^{-2\pi i s\lambda}\right)
\end{aligned}
$$

as claim.

4.3.3 Discrete Fourier Transformation – DFT

The Fourier transformation of a stationary process x_t is defined as

$$
J_x(\lambda) = J_x^n(\lambda) = \sum_{t=1}^{n} x_t e^{-2\pi i\lambda(t-1)} = \sum_{t=1}^{n}\Big(x_t\cos 2\pi\lambda(t-1) + i x_t \sin 2\pi\lambda(t-1)\Big), \qquad 0 \le \lambda \le 1.
$$

The discrete Fourier transformation (DFT) of x_t is defined as

$$
J_x(\lambda_k), \qquad k = 0, \ldots, n,
$$

where $\lambda_k = k/n$ are the **Fourier frequencies**. Other commonly used notations for FT and DFT are:

$$
d(\lambda) \equiv J(\lambda), \qquad \text{and} \qquad d_j \equiv d(\lambda_j) = J(\lambda_j).
$$

In SPLUS, FFT is easily computed using the function `fft`.

4.3.4 Distribution of the DFT

Let x_t denote a stationary process with mean $\mu = Ex_t$ and power spectrum $f_x(\lambda)$. The following result provides approximated distributions of the DFT:

$$
J_x^n(\lambda_j) = n^{-1/2}\sum_{t=1}^{n} x_t e^{-2\pi i\lambda_j(t-1)}, \quad \lambda_j = j/n, \quad j = 0, \pm 1, \ldots.
$$

Theorem 3 *(Brillinger, 1981) The DFT has the following distributions*

$$
J_x^T(\lambda_j) \overset{a}{\sim} \begin{cases} N^C(0, 2\pi f_x(\lambda_j)), & \textit{if } j \not\equiv 0 \pmod{T/2}, \\ N(\mu, 2\pi f_x(0)), & \textit{if } j = 0, \pm 2T/2, \ldots, \\ N(0, 2\pi f_x(\pi)), & \textit{if } j = \pm T/2, \pm 3T/2, \ldots. \end{cases}
$$

And for $j \not\equiv k \pmod{T/2}$, $J_x^T(\lambda_j)$ and $J_x^T(\lambda_k)$ are asymptotically independent.

We can prove this theorem using the `qqnorm` and `fft` in SPLUS.

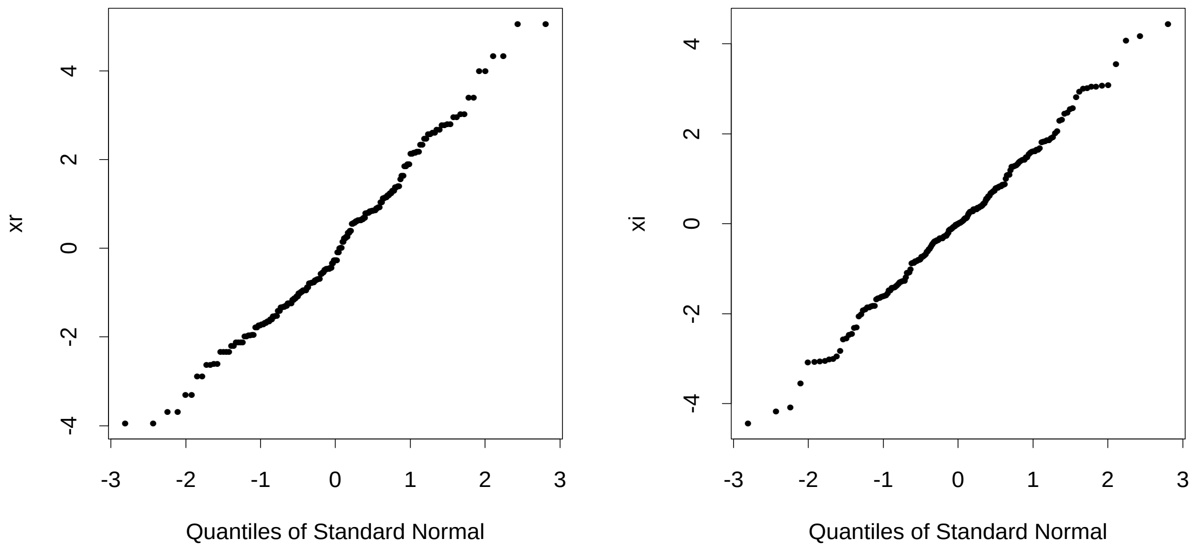


Fig 4.3 *qqnorm of the Re and Im of the FFT.*

```
> x <- rgamma(200,5)     # 200 gamma variates
> x <- x - mean(x)       # remove the mean
> xr <- Re(fft(x))/sqrt(200)
> xi <- Im(fft(x))/sqrt(200)
> qqnorm(xr)  # Fig 4.3 (left)
> qqnorm(xi)  # Fig 4.3 (right)
> y <- matrix(c(xr,xi),200,2)
> acf(y)        # Fig 4.4
```

Interpretation

The above result can be interpreted in terms of real and imaginary parts of the DFT as follows. To simplify our discussion, let $\lambda_j \neq 0$. Then

$$\operatorname{Re} J_x^n(\lambda_j) = n^{-1/2} \sum_{t=1}^{n} x_t \cos 2\pi\lambda_j(t-1) \overset{a}{\sim} N(0, \pi f_x(\lambda_j)),$$

$$\operatorname{Im} J_x^n(\lambda_j) = n^{-1/2} \sum_{t=1}^{n} x_t \sin 2\pi\lambda_j(t-1) \overset{a}{\sim} N(0, \pi f_x(\lambda_j)),$$

and more importantly, $\operatorname{Re} J_x^n(\lambda_j)$ and $\operatorname{Im} J_x^n(\lambda_j)$ are *independent*.

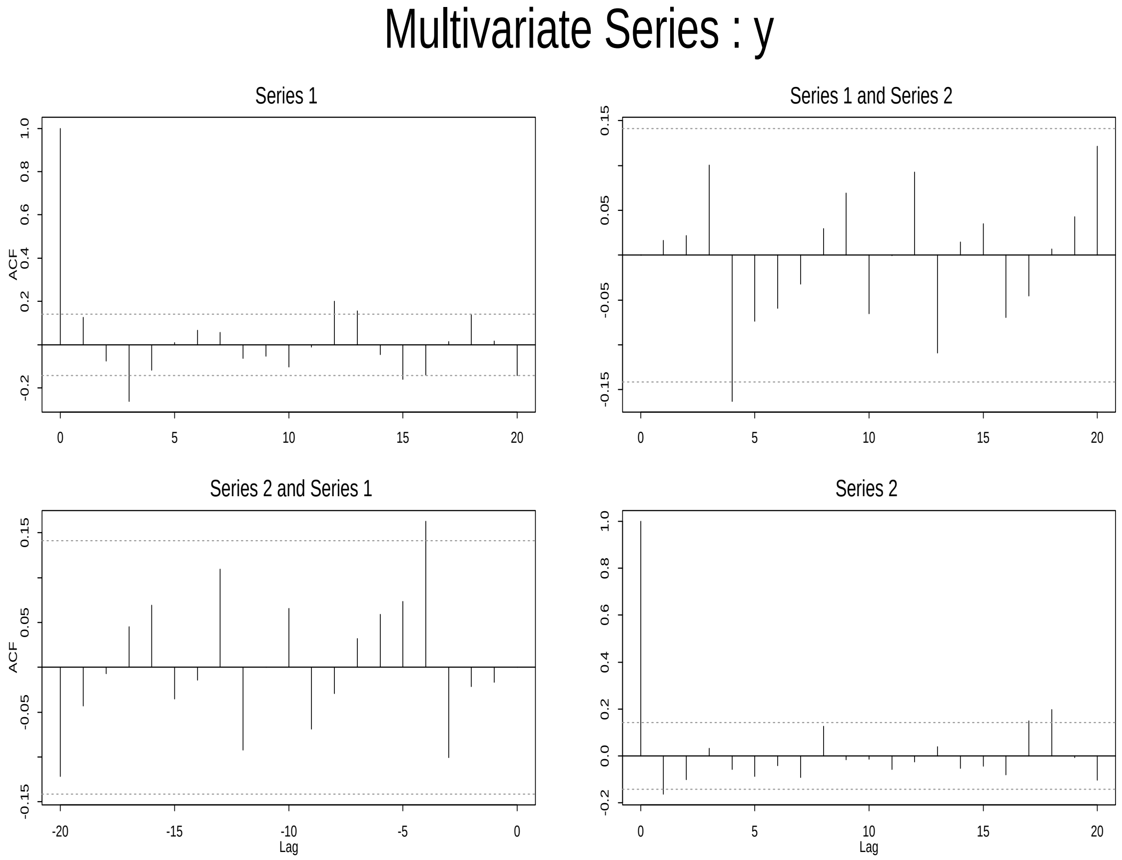


Fig 4.4 *Covariance functions of the Re and Im of the FFT.*

Implications

1. Since $N(0,1)^2 = \chi_1^2$, it follows from the above result that

$$\frac{[\mathrm{Re}\, J_x^n(\lambda_j)]^2}{\pi f(\lambda_j)} + \frac{[\mathrm{Im}\, J_x^n(\lambda_j)]^2}{\pi f(\lambda_j)} \overset{a}{\sim} \chi_2^2$$

   or,

$$\frac{1}{2\pi}|J_x^n(\lambda_j)|^2 \overset{a}{\sim} \frac{1}{2} f_x(\lambda_j)\chi_2^2. \quad (1)$$

2. Since $\chi_2^2/2$ is an exponential distribution with mean 1, we have

$$E\left\{\frac{1}{2\pi}|J_x^n(\lambda_j)|^2\right\} \approx f(\lambda_j), \quad (2)$$

$$\mathrm{var}\left\{\frac{1}{2\pi}|J_x^n(\lambda_j)|^2\right\} \approx [f(\lambda_j)]^2. \quad (3)$$

4.3.5 The periodogram

The periodogram of the observed values $x_1, x_2, \ldots, x_n$ is defined to be

$$I_x(\lambda_j) = I_x^n(\lambda_j) = |J_x^n(\lambda_j)|^2 = n^{-1}|\sum_{t=1}^{n} x_t e^{-i\lambda_j(t-1)}|^2.$$

The periodogram is an estimate of the spectral density as suggested by the above distributional result. Also, note that we have previously obtained the same estimate by using the estimated autocovariance function $\hat{R}(u)$.

Periodogram is computed easily using `spectrum`.

```
> spectrum(rgamma(1000,5))  # compute the periodogram for 1000 gamma

> x <- rgamma(1000,5)
> y <- Mod(fft(x))^2/length(x)
> y[1] <- 0
> x <- c(0:999)/2000
> plot(x,y,type="l") # Fig 4.6
> plot(density(y),type="l") #Fig 4.6
```

We can see that the distributions of periodogram ordinates are (approx) independent and have an exponential distribution.

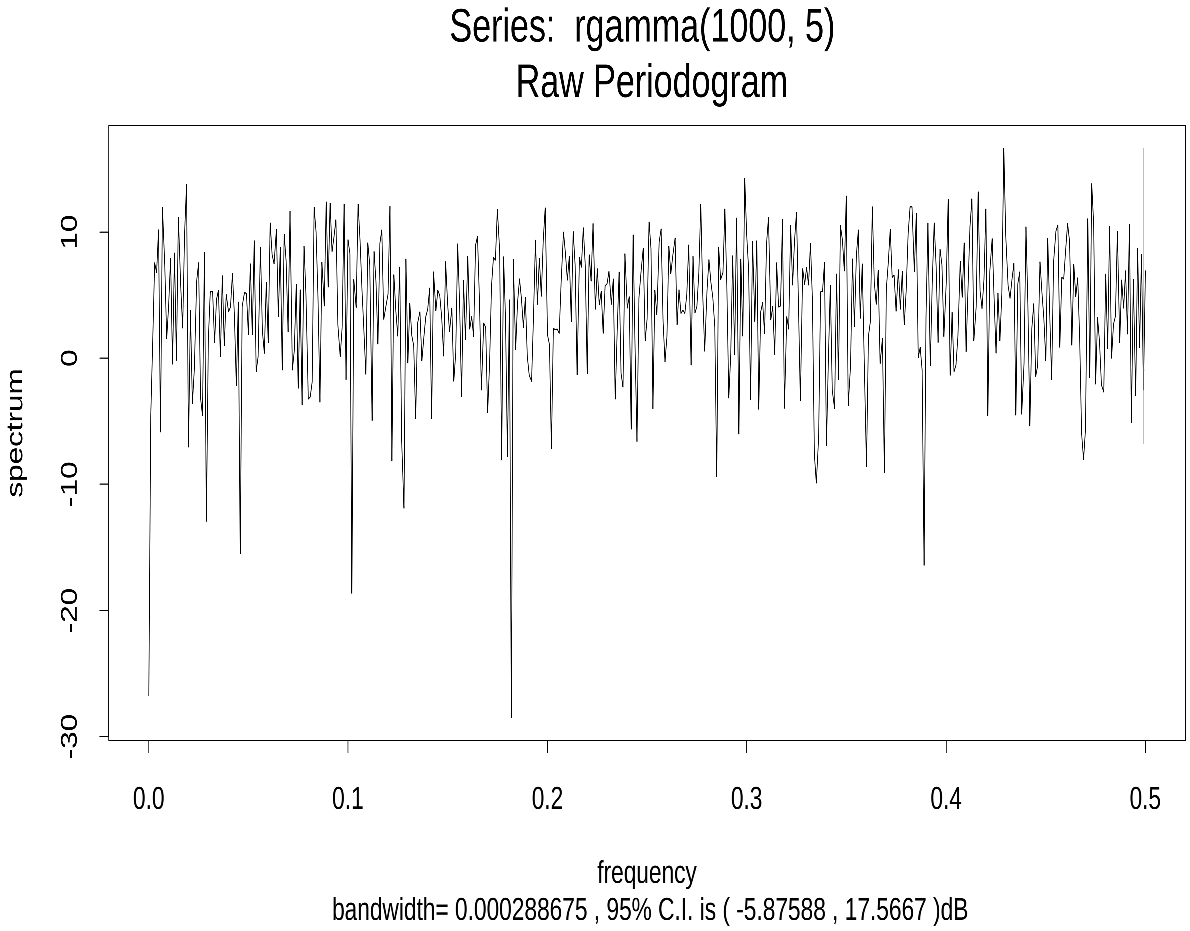


Fig 4.5 *Periodogram of 1000 gamma (5): log 10.*

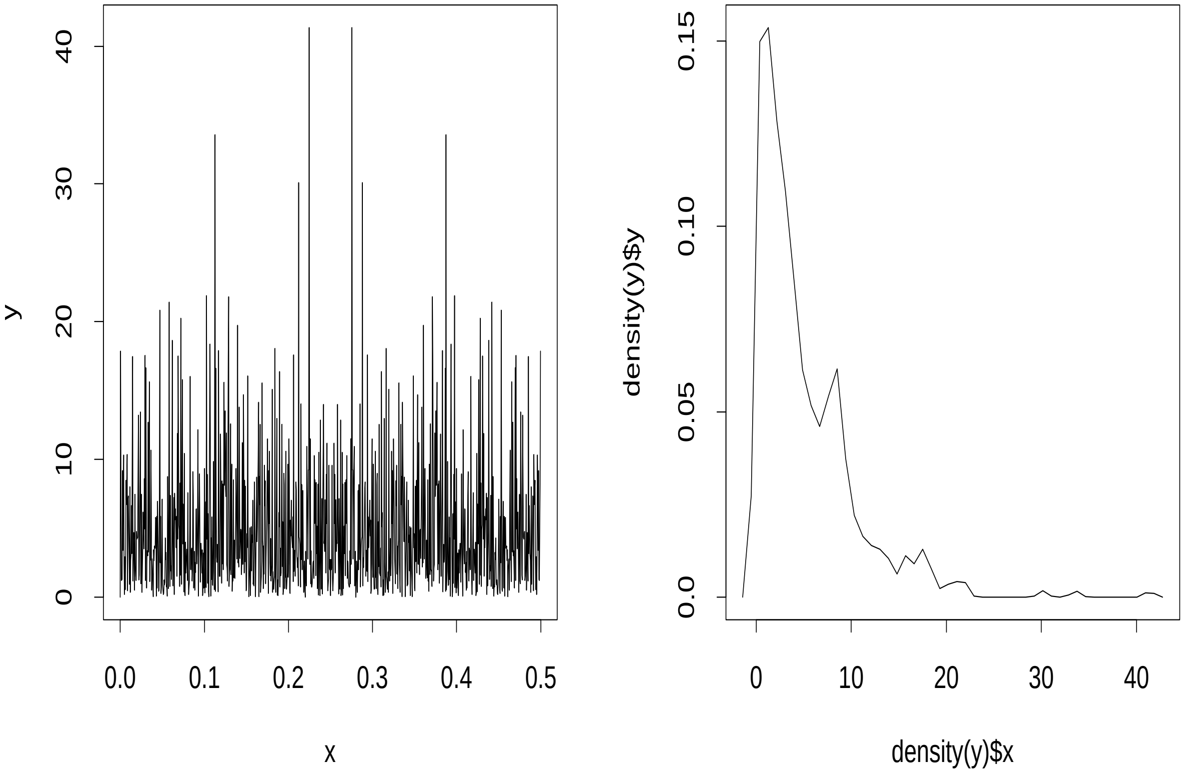


Fig 4.6 *Left: Periodogram of 1000 gamma (5). Right: Density of the periodogram ordinates.*

Remarks

1. (1) suggests that the *values* of the periodogram $I_x(\lambda_j)$ (for different λ_j) are approximately independent and they have an exponential distribution with mean $f_x(\lambda_j)$.
2. (2) indicates that the estimate of the spectral density function is basically obtained by replacing the mean with the only observation. It also says that the estimator is unbiased.
3. (3) implies that the estimator is not *consistent*. Recall that
$$\text{Mean squared error} = \text{bias}^2 + \text{variance}.$$

4.3.6 Confidence interval for $f_x(\lambda)$

Since $I_x^n(\lambda_j) \sim^a f_x(\lambda_j)\chi_2^2/2$ and $\chi_2^2/2$ has an exponential distribution with mean 1, a $100(1-\alpha)\%$ confidence interval for $f_x(\lambda_j)$ is given by

$$\frac{I_x(\lambda_j)}{-\log(\alpha/2)} \le f_x(\lambda_j) \le \frac{I_x(\lambda_j)}{-\log(1-\alpha/2)}.$$

4.4 Examples of harmonic analysis

4.4.1 Simple harmonic process

Consider the time series $x_t = A\cos(\lambda t + \phi)$, $t = 1, \ldots, n$. Since

$$\cos(\lambda t + \phi) = \frac{1}{2}[e^{i(\lambda t+\phi)} + e^{-i(\lambda t+\phi)}],$$

it is enough to consider the DFT of $e^{i\lambda t}$. Using the geometric series,

$$J_j \equiv J(\lambda_j) = \sum_{t=1}^{n} \exp i(\lambda - \lambda_j)t = \exp[i(\lambda-\lambda_j)(n+1)/2] \cdot \frac{\sin(\lambda-\lambda_j)n/2}{\sin(\lambda-\lambda_j)/2},$$

where $\lambda_j = 2\pi j/n$ $j = 0, \ldots$ are the Fourier frequencies. In fact, set $z = \exp i(\lambda - \lambda_j)$. Then according to the geometric series,

$$J_j = \sum_{t=1}^{n} z^t = \frac{z - z^{n+1}}{1-z} = \frac{z z^{n/2}(z^{-n/2} - z^{n/2})}{z^{1/2}(z^{-1/2} - z^{1/2})}.$$

The desired result follows from $z^{-n/2} - z^{n/2} = -2i\sin(\lambda-\lambda_j)n/2$ and $z^{-1/2} - z^{1/2} = -2i\sin(\lambda-\lambda_j)/2$.

Note that if $\lambda = \lambda_k$, then $J_j = 0$ for $j \neq k$ and $J_k = n$. In general, if the frequency of the harmonic series λ is not a Fourier frequency, then the transform is nonzero. This is called *leakage*.

When λ is small and n is large, the graph of $\sin(n\lambda/2)/n\sin(\lambda/2)$ is approximately equal to the graph of $\sin(n\lambda/2)/(n\lambda/2)$ which is illustrated in Figures 4.8a,b.

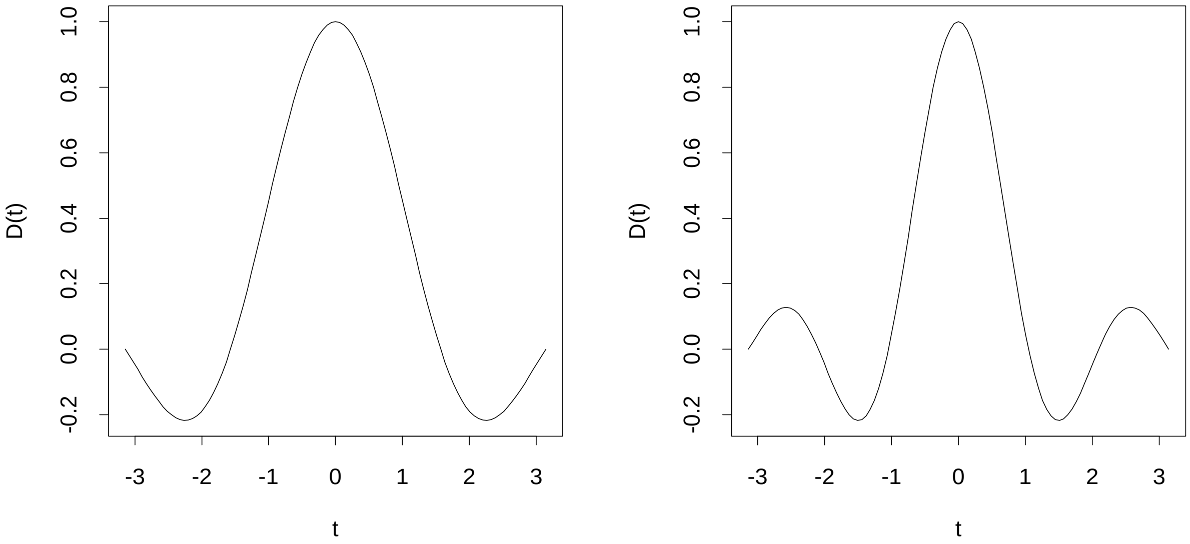


Fig 4.8a *Dirichlet kernel for* $n = 4, 6, 10, 14$.

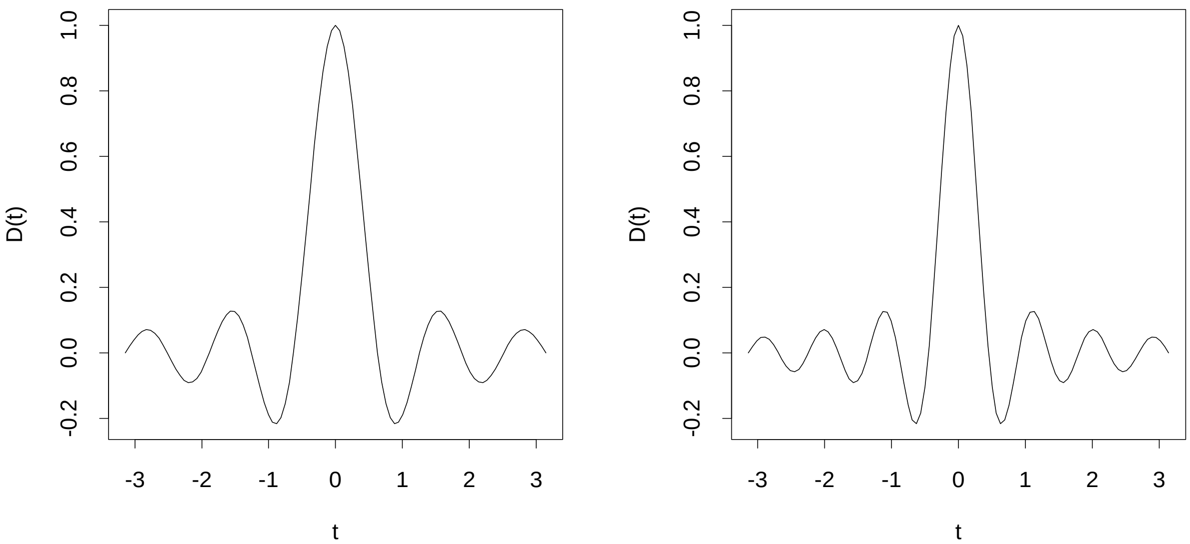


Fig 4.8b *Dirichlet kernel for* $n = 4, 6, 10, 14$.

4.4.2 A step function

Consider the following time series:

$$x_t = \begin{cases} 1, & \text{if } 1 \leq t \leq m, \\ 0, & \text{if } m < t \leq n. \end{cases}$$

Then

$$J_j = \sum_{t=1}^{m} \exp(-i\lambda_j t) = \exp[-i\lambda_j(m+1)/2] \cdot \frac{\sin \lambda_j m/2}{\sin \lambda_j/2}.$$

4.4.3 Periodogram analysis of a simple harmonic series

```
> y <- cos.sim(a=c(.01),b=c(.01),f=8/40,L=40)
> y.pgram <- pgram(y,rm.mean=F)
> pgram.plot(y.pgram)   # Figure 4.8c (left)

> y.pgram
 [1] 2.708339e-36 1.282977e-35 2.226395e-35 1.421197e-35 1.391722e-35
 [6] 4.755005e-35 6.722448e-35 2.151841e-34 2.000000e-03 1.738081e-34
[11] 2.845637e-35 1.258007e-35 4.814825e-36 4.881771e-36 7.678044e-37
[16] 1.162437e-36 5.943928e-36 2.758271e-37 4.009012e-36 2.195724e-36
[21] 1.474540e-35
```

```
> y <- cos.sim(a=c(.01),b=c(.01),f=8.5/40,L=40)
> y.pgram <- pgram(y)
> pgram.plot(y.pgram)   # Figure 4.8c (right)

> y.pgram
 [1] 1.802225e-07 3.166344e-07 7.786611e-07 1.764693e-06 3.795402e-06
 [6] 8.342913e-06 2.063747e-05 6.985654e-05 7.480014e-04 8.754177e-04
[11] 1.125751e-04 4.656473e-05 2.717976e-05 1.877427e-05 1.435127e-05
[16] 1.175416e-05 1.013373e-05 9.100755e-06 8.459437e-06 8.107623e-06
[21] 7.995368e-06
```

Example. Periodogram analysis of a boxcar.

```
y <- c(rep(1,a),rep(0,a*2))
tsplot(y) # Figure 4.8d
per <- pgram(y)
pgram.plot(y.pgram)  # Figure 4.8d
```

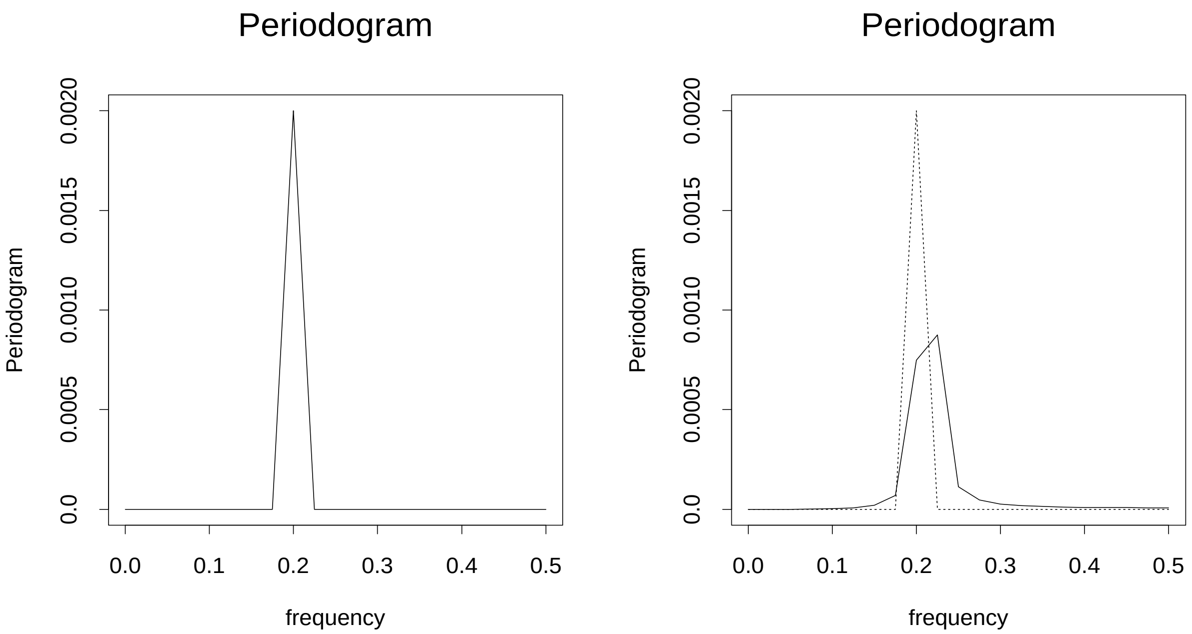


Fig 4.8c *Periodograms of a simple harmonic process.*

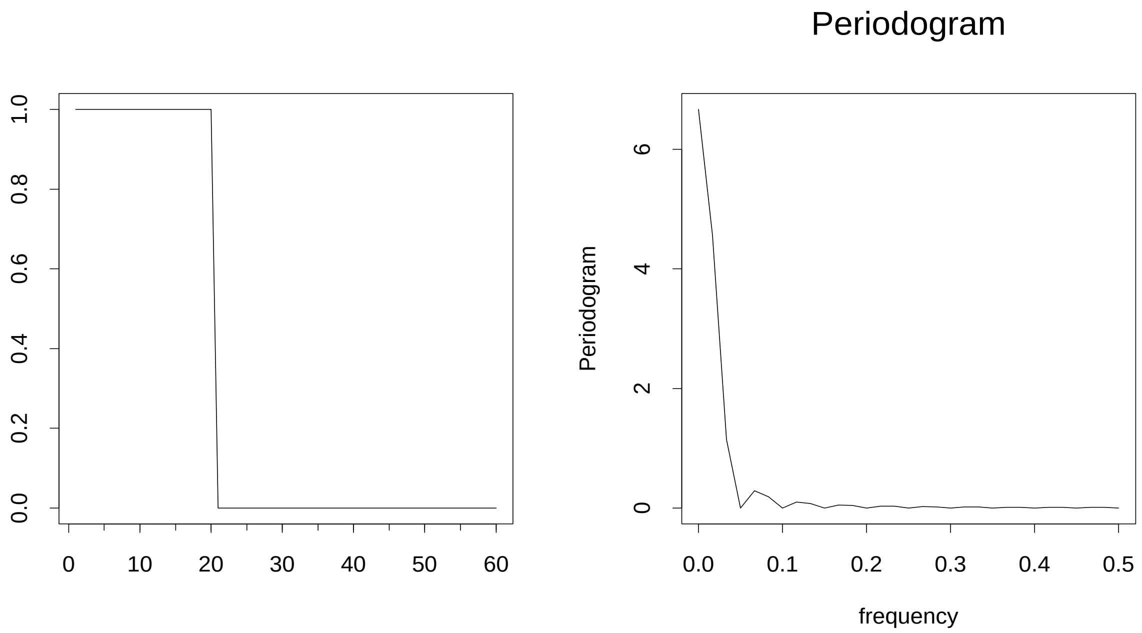


Fig 4.8d *Periodograms of a boxcar.*

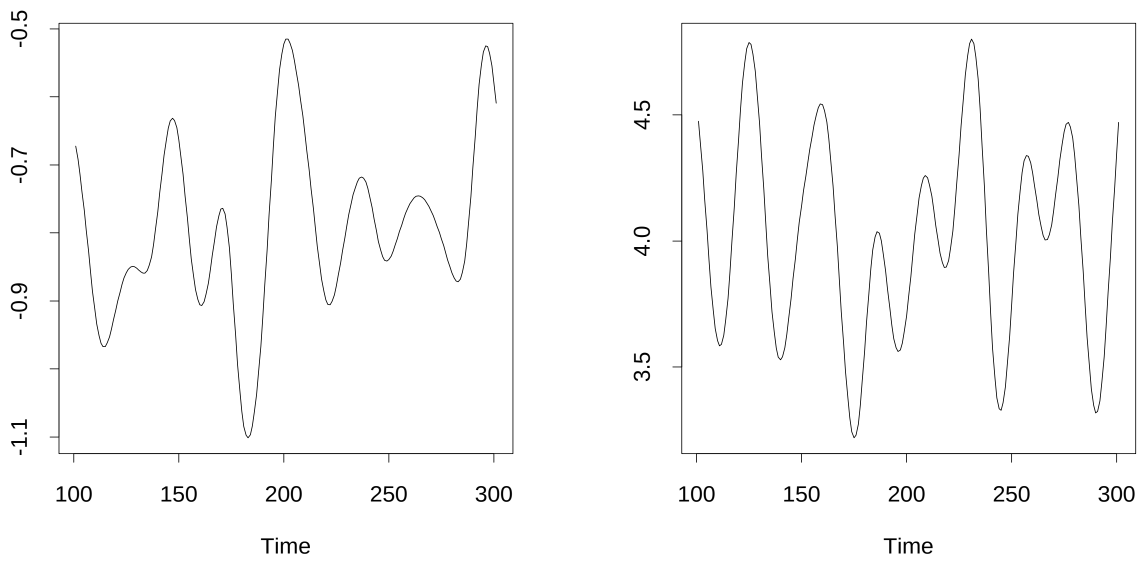


Fig 4.8e *Examples of* `demod`.

4.4.4 Complex demodulation

Complex demodulation is a technique for analyzing nonstationary time series by estimating the instantaneous amplitude and phase of a given harmonic component. We use a simple model to illustrate this. Suppose we have a simple harmonic process

$$x_t = A\cos(\lambda t + \phi) + w_t.$$

Rewrite this process as

$$x_t = \frac{A}{2}[e^{i(\lambda t+\phi)} + e^{-i(\lambda t+\phi)}] + w_t.$$

Multiply this series by $e^{-i\lambda t}$, yielding

$$x_t e^{-i\lambda t} = \frac{A}{2}e^{i\phi} + \frac{A}{2}e^{-i(2\lambda t+\phi)} + w_t e^{-i\lambda t}.$$

The first term is a low frequency while the last two terms are high frequency components, which can be eliminated by running mean or low pass filter. This is implemented in the function `demod` in S-PLUS. Note that the effectiveness of this function depends on a reasonable estimate of the frequency λ, which is usually the frequency that yields the maximum periodogram. The following example illustrates this point.

```
# -------------------------------
```

```
set.seed(22)
> n <- 400
> x <- seq(0,n)
> noise <- rnorm(n+1)
> y <- 4 * cos(2*pi/20*x-pi/4) + noise
> yp <- pgram(y)
> z <- minmax0(yp)
> z
$xmax:
[1] 1501.646

$imax:
[1] 21

ydem <- demod(y,(z$imax-1)/n, .045,.055)
ts.plot(ydem$phase) # Fig 4.8e
ts.plot(ydem$amp)   # Fig 4.8e

> A <- (4*z$xmax/n)^.5
> A
[1] 3.875108

> f <- (z$imax-1)/n
> f
[1] 0.05

> p <- mean(ydem$phase[101:200])
> p
[1] -0.7797612

# Nonlinear least square method
#
> z <-  nls(y ~ A * cos(2*pi*f * x + p),
        data = data.frame(y = y, x = x),
        start = list(A = A, f = f, p = p))
> z
Residual sum of squares : 425.1559
parameters:
        A          f          p
 4.025627 0.04998233 -0.7530855
formula: y ~ A * cos(2 * pi * f * x + p)
401 observations
```

We note that the relationship between the maximum of the periodogram and the amplitude A is given by

$$\text{max periodogram} = nA^2/4.$$

The corresponding S function is given by

```
> yp <- pgram(y)
> z <- minmax0(yp)
> A <- (4*z$xmax/n)^.5
```

This can be verified as a simple exercise.

The above method also illustrates a way to estimate the parameters of the model

$$x_t = a\cos(\lambda t) + b\sin(\lambda t) + w_t.$$

In fact, the complex demodulation and the use of nonlinear least squares method `nls(...)` yield estimates $\widehat{A}$ and $\widehat{\phi}$. Thus natural estimates of a and b are given by

$$\widehat{a} = \widehat{A}\cos\widehat{\phi}, \qquad \widehat{b} = \widehat{A}\sin\widehat{\phi}.$$

```
# --------------------------------

> a <- 4.025627 * cos(-0.7530855)
> b <- 4.025627 * sin(-0.7530855)
> z <- nls(y ~ a * cos(2 * pi * f * x)+ b * sin(2 * pi * f * x),
          data = data.frame(y = y, x = x),
          start = list(a = b, b = b, f = f) )
> z
Residual sum of squares : 425.1559
parameters:
       a        b          f
 2.93705 2.753072 0.04998232
formula: y ~ a * cos(2 * pi * f * x) + b * sin(2 * pi * f * x)
401 observations
```

More examples on `nls()`.

```
# --------------------------------

> set.seed(20)
> noise <- rnorm(100,sd=.5)
> x <- seq(0,2*pi,length=100)
> my <- 5*sin(x - pi) + noise
> plot(x,my)
```

```
> rm(y)
> nls(y ~ amp * sin(x - phi),
+   data = data.frame(y = my, x = x),
+   start = list(amp = 1, phi = 0))

Residual sum of squares : 20.25668
parameters:
       amp         phi
 -5.112182 0.008521062
formula: y ~ amp * sin(x - phi)
100 observations
```

4.5 Smoothed periodograms

We know that the periodogram ordinates $I(\lambda_j)$ are rather erratic due to the nonzero variance [see (3)]. But since they are approximately independent $f(\lambda_j)\exp(1)$ and one may expect the variance be reduced by averaging the nearby periodogram ordinates. This leads to the following estimate:

$$\hat{f}(\lambda_k) = L^{-1} \sum_{|j-k|\le (L-1)/2} I(\lambda_j).$$

Note that if the spectral density $f(\lambda_k)$ is continuous (or smooth) at λ_k and L is not too large, then $f(\lambda_j) \approx f(\lambda_k)$ with $|j-k| \le (L-1)/2$. Therefore $\hat{f}(\lambda_k)$ has approximately a chi-squared distribution with L degrees of freedom. Consequently,

$$E[\hat{f}(\lambda_k)] \approx f(\lambda_k)$$

and

$$\mathrm{var}(\hat{f}(\lambda_k)) \approx \frac{f(\lambda_k)^2}{L}.$$

Bandwidth

Define

$$b_k := \{\lambda_j : |j-k| \le (L-1)/2\}.$$

This plays an important role in spectral density estimation. The *resolution* or *bias* of the estimate is controlled by the width $b = 2\pi L/n$: large value of L results in large bias in the estimate. On the other hand, the *stability* or *variance* of the estimator is proportional to $1/L$.

How do we choose L? In practice, we would plot several smoothed estimates using different values of L and use common sense to select one that matches the interpretations of data analysis.

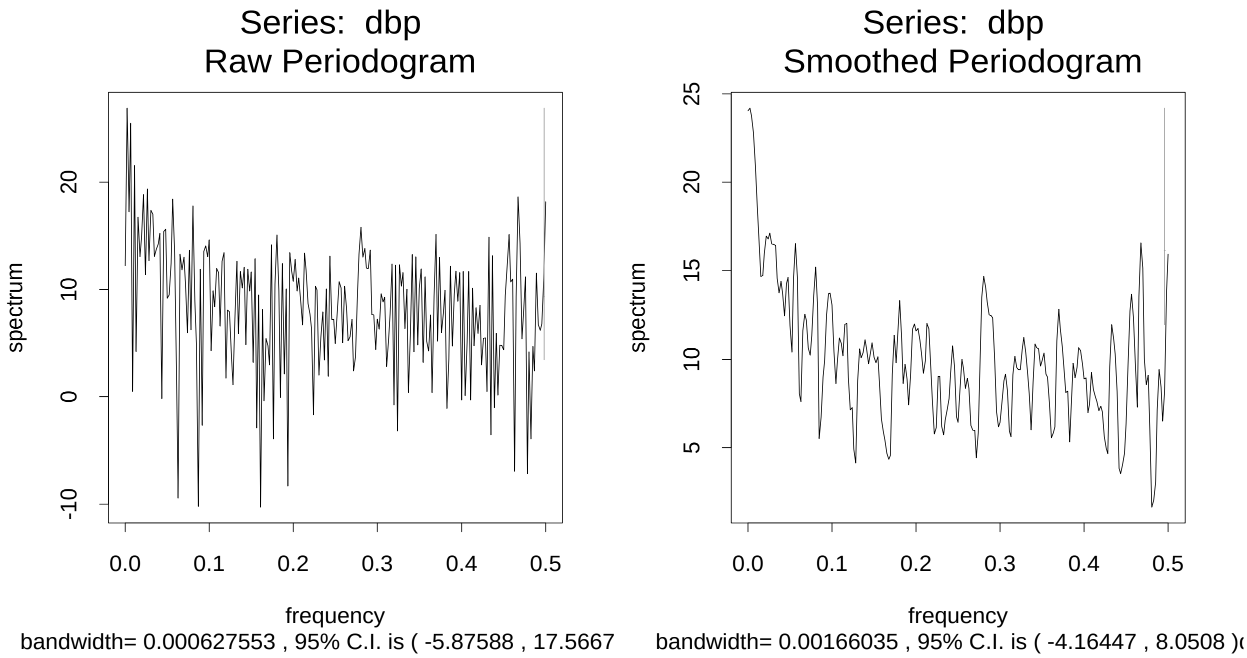


Fig 4.9 *Power spectrum of DBP series.*

Estimate Spectrum of Time Series in S-PLUS

```
       spectrum(x, method="pgram", plot=T, ...)

REQUIRED ARGUMENTS:
x:     a univariate or multivariate time series, or a vector,  or
       a  matrix  with  a  univariate series per column.  Missing
       values are allowed only at the ends.  If method="ar", then
       this can also be a list as returned by the ar function.

...

EXAMPLES:
       spectrum(log(lynx), pad=1, spans=3)  # Compute and plot Raw Periodogram.
       p2 <- spectrum(log(lynx), method="ar", plot=F)
       spec.plot(p2,add=T)       # Superimpose an autoregressive spectrum
```

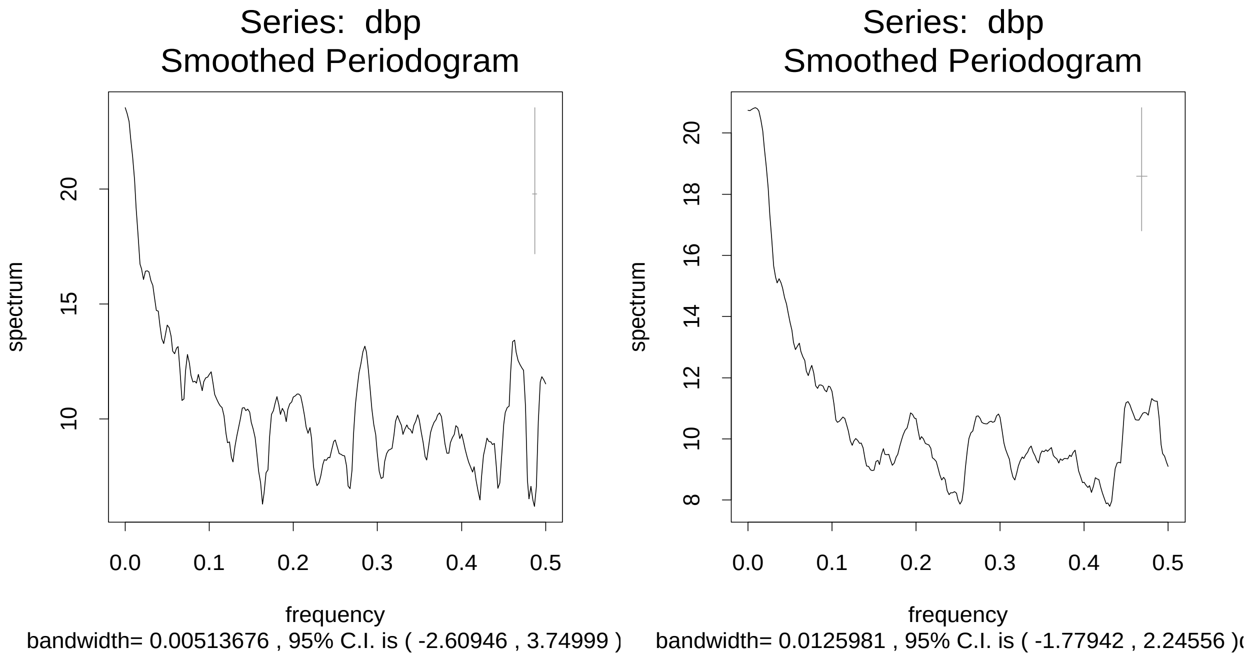


Fig 4.9 *Power spectrum of DBP series.*

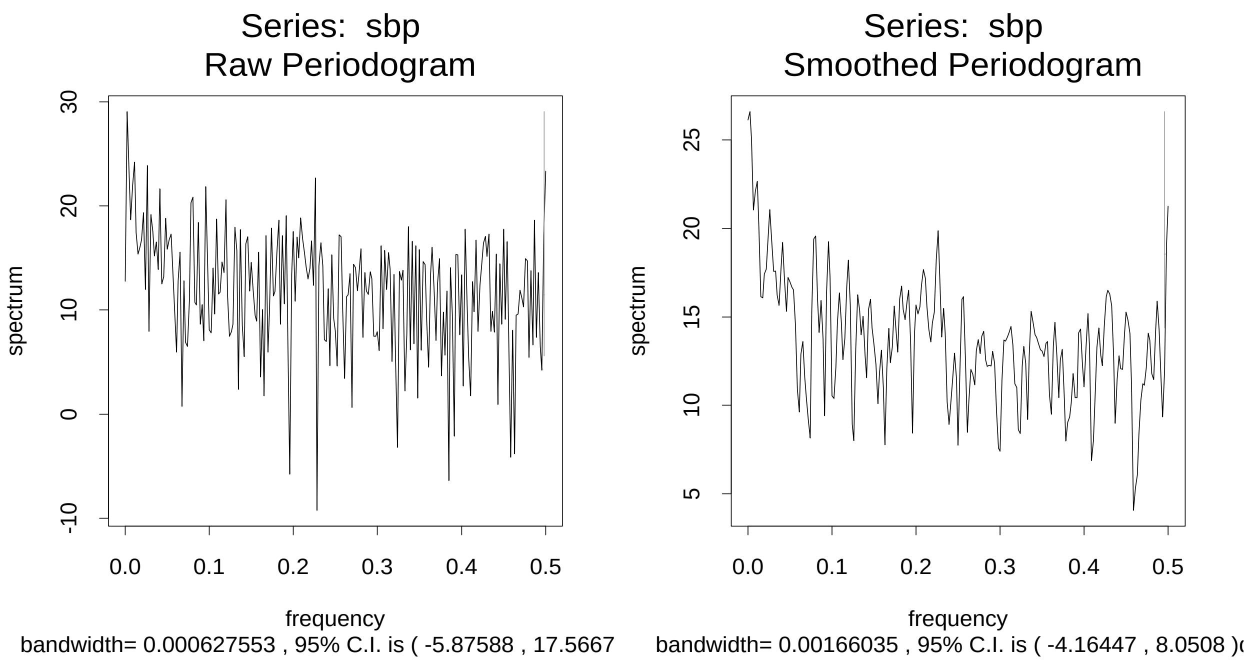


Fig 4.10 *Power spectrum of SBP series.*

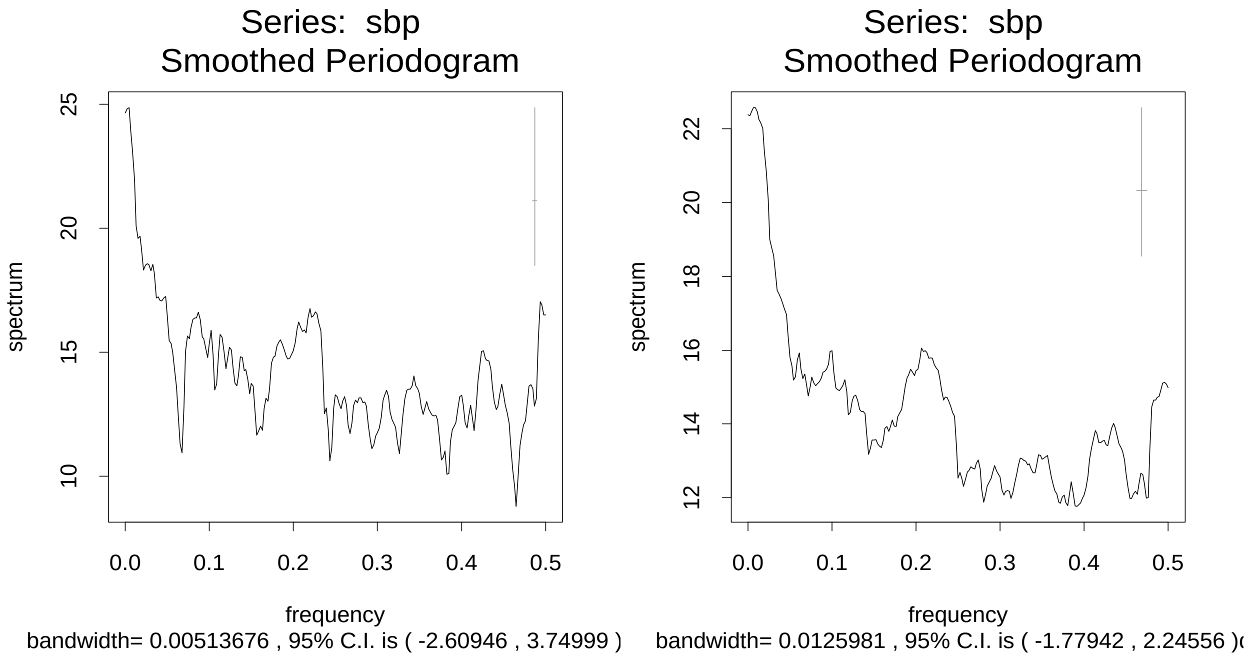


Fig 4.10 *Power spectrum of SBP series.*

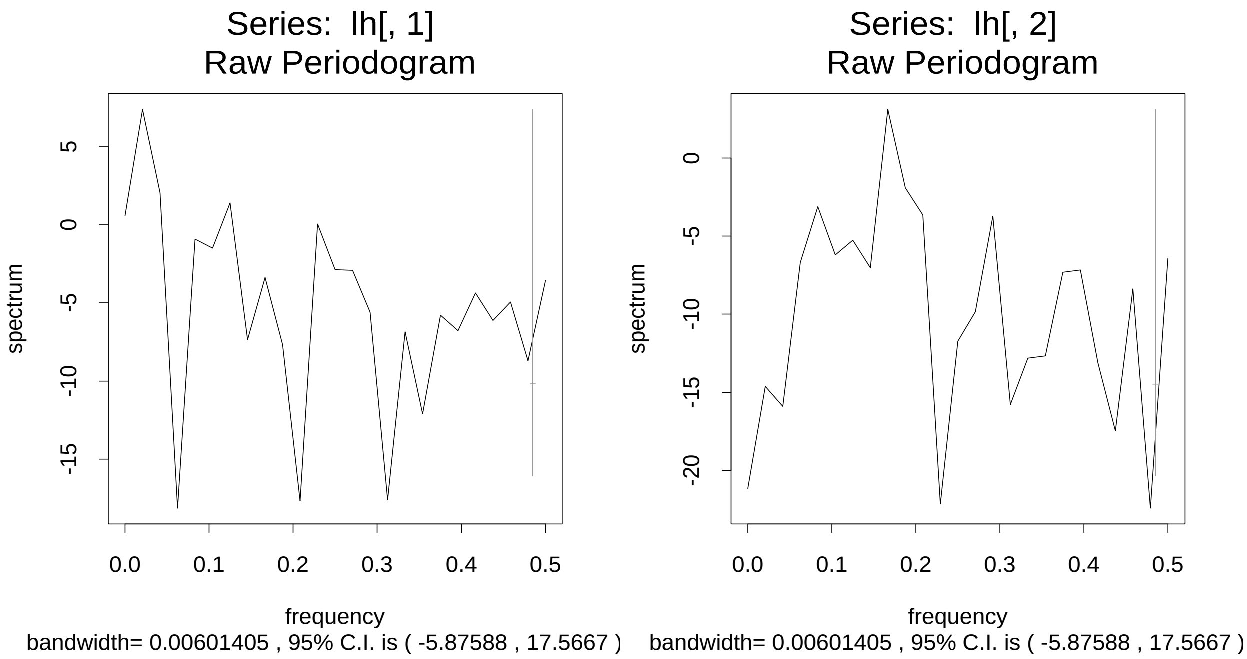


Fig 4.11 *Periodograms of the hormone series.*

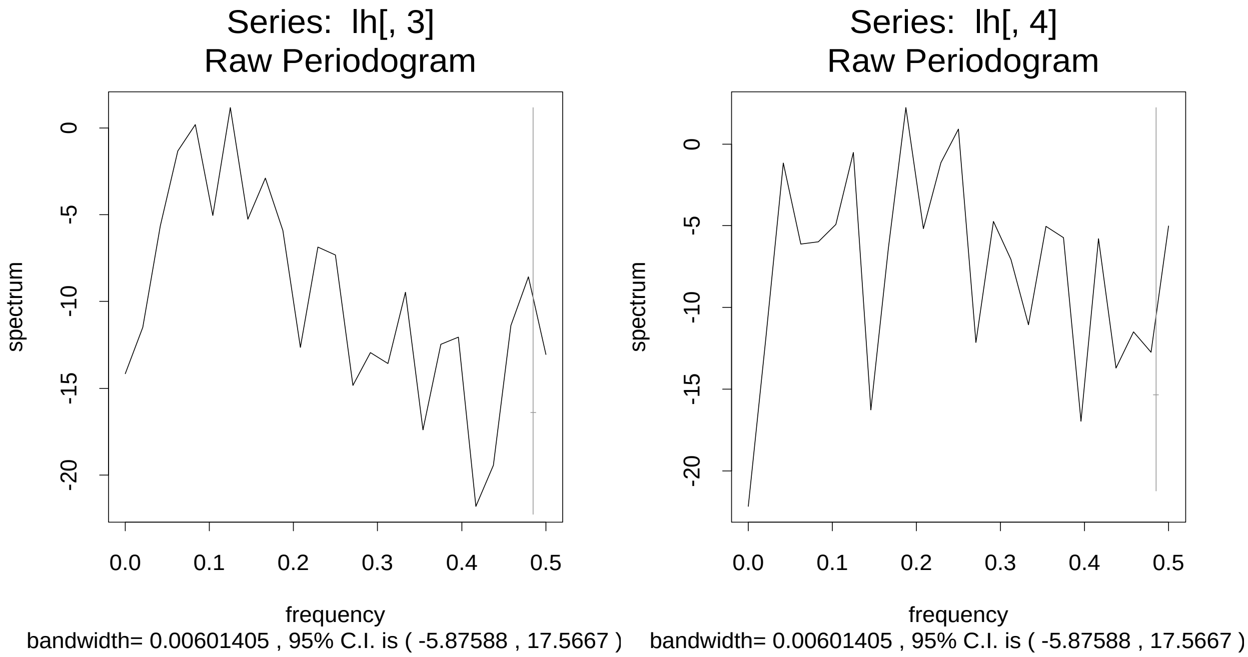


Fig 4.11 *Periodograms of the hormone series.*

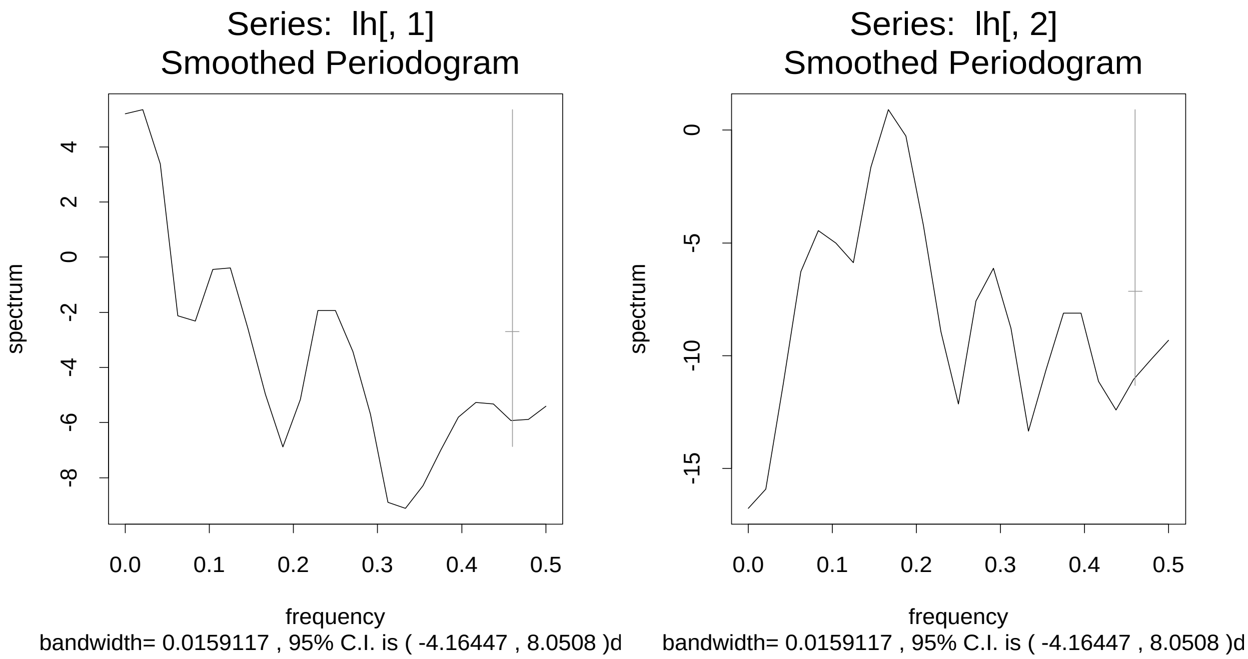


Fig 4.12 *Power spectrum of the hormone series.*

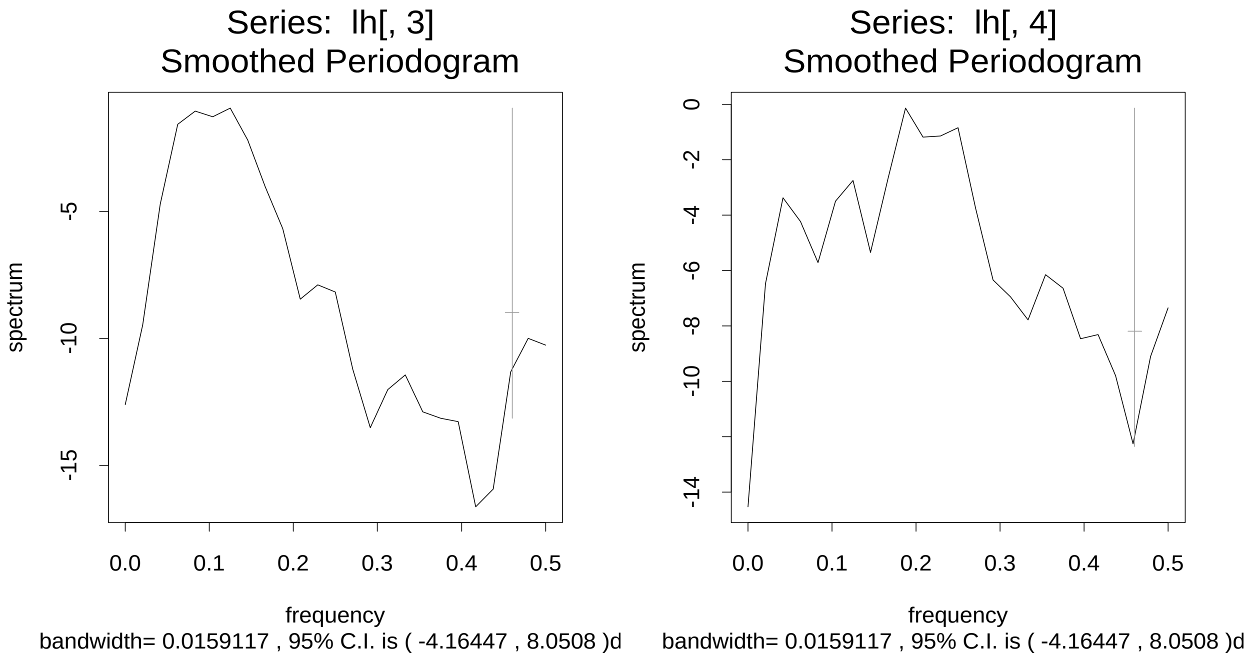


Fig 4.12 *Power spectrum of the hormone series.*

4.6 Asymptotic distributions

The smoothed periodogram estimators, $\hat{f}(\lambda_j)$, evaluated at the Fourier frequencies are asymptotically independent with

$$\hat{f}(\lambda_j) \sim_a \begin{cases} f(\lambda_j)\chi^2_{2L}/2L, & \text{if } j \neq 0, \\ f(0)\chi^2_{L-1}/(L-1), & \text{if } j = 0. \end{cases}$$

4.7 Confidence intervals

Let $\chi^2_\nu(\alpha)$ denote the upper α percentile of the chi-squared distribution with ν df. Then a $100(1-\alpha)$ confidence interval for $f(\lambda_k)$ is given by

$$\frac{2L\hat{f}(\lambda_k)}{\chi^2_{2L}(\alpha/2)} \leq f(\lambda_k) \leq \frac{2L\hat{f}(\lambda_k)}{\chi^2_{2L}(1-\alpha/2)}.$$

4.8 $\log_{10}\hat{f}(\cdot)$ — *decibels* (db)

Another way to set confidence limits is to take the logarithm of $f(\cdot)$ which is interpreted in terms of decibels (db) in engineering. Here $\log \equiv \log_{10}$:

$$\log(\hat{f}(\lambda_k)) - \log(\chi^2_{2L}(\alpha/2)/2L) \leq \log f(\lambda_k) \leq \log(\hat{f}(\lambda_k)) - \log(\chi^2_{2L}(1-\alpha/2)/2L).$$

An advantage is that the length of the CI does not depend on the frequencies.

Note that when ν is large, $\log(\chi^2_\nu/\nu)$ is approximately $N(0, 2(.4343)^2/\nu)$. Hence

$$\log(\hat{f}(\lambda_k)) - z(\alpha/2)(0.4343)\sqrt{1/L} \leq \log f(\lambda_k) \leq \log(\hat{f}(\lambda_k)) - z(1-\alpha/2)(0.4343)\sqrt{1/L}.$$

4.9 Harmonics

Suppose that the series $x_1, \ldots, x_n$ is periodic with h, $n = kh$. Let

$$K_l = \sum_{t=1}^{h} x_t e^{-it2\pi l/h}.$$

Then

$$x_t = \sum_{l=1}^{h} K_l e^{it2\pi l/h}.$$

Note that the above representation is periodic and

$$2\pi l/h = 2\pi kl/n = \lambda_{kl}.$$

From $x_t = \sum_{j=1}^{n} J_j e^{it2\pi\lambda_j}$ we conclude that

$$J_j = \begin{cases} K_l, & \text{if } j = kl; \\ 0, & \text{otherwise.} \end{cases}$$

That is, the DFT of x_t is nonzero at the frequency $2\pi/h$ and is also nonzero at its multiples $2\pi l/h, (l = 2, \ldots, h-1)$. The latter are called the *harmonics*.

Some Remarks.

1. For $j, k = 1, \ldots, n$ the following relationship indicates that the function $\{e^{i(\lambda_j - \lambda_k)t}\}$ has an orthogonal structure:

$$\sum_{t=1}^{n} e^{i(\lambda_j - \lambda_k)t} = \begin{cases} n, & \text{if } j = k; \\ 0, & \text{otherwise.} \end{cases}$$

2. The DFT $d^n(\lambda)$ has a growth rate of the order $\sqrt{n \log n}$:

$$P\left(\limsup \sup_{\lambda} \frac{|d^n(\lambda)|}{\sqrt{n \log n}} \leq C\right) = 1,$$

where C is a function of $\sup f(\lambda)$.

4.10 Testing for periodogram ordinates

When the periodogram contains a numbers of peaks, it is important to test if these are caused by the harmonic components. Note that the periodogram ordinates are independent and have a common exponential distribution, and hence there are many spikes. Suppose that x_t, $t = 1, \ldots, n$ is a Gaussian white noise, then the spectral density is equal to σ^2 and the periodogram ordinates $I_j \equiv I(\lambda_j)$ $j = 0, 1, \ldots, [n/2]$ are independent. Moreover,

$$I_j/\sigma^2 \sim \chi_2^2/2 \qquad j = 1, \ldots, [(n-1)/2].$$

That is

$$P(I_j/\sigma^2 \leq x) = 1 - \exp(-x).$$

Set $m = [(n-1)/2]$. Then

$$P\left(\max\{I_j\}/\theta \leq x\right) = P\left(I_j/\theta \leq x, j = 1, \ldots, m\right) = (1 - \exp(-x))^m.$$

Suppose σ^2 is known, then the P-value can be computed from

$$P\left(\max\{I_j\}/\theta > x\right) = 1 - (1 - \exp(-x))^m.$$

The test statistic $\max\{I_j\}/\theta$ was used by Schuster (1898) to test for hidden periodicity.

In practice, σ^2 is not known and has to be estimated. Recall that

$$\frac{1}{m} E\Big(\sum_1^m I_j\Big) = \hat{\sigma}^2 \to \sigma^2.$$

Hence, it is natural to consider the following test statistic:

$$g_m := \frac{\max I_j}{\sum_1^m I_j/m} \sim_a \frac{\max I_j}{\sigma^2}.$$

That is,

$$P\left(\frac{\max_j I_j}{\sum_1^m I_j/m} > x\right) \doteq 1 - (1 - e^{-x})^m.$$

This can be shown to be a first order approximation to *Fisher's exact test.*

Fisher's exact test. Fisher (1929) used the test statistic g_m to test the null hypothesis that x_t is a Gaussian white noise against the alternative hypothesis that x_t contains a harmonic component of unspecified frequency. The exact P-value (derived by Fisher using an ingenious geometric argument) is given by

$$P(g_m \geq z) = 1 - \sum_{j=0}^m (-1)^j \binom{m}{j} (1 - jz/m)_+^{m-1}, \tag{4.6}$$

where $x_+ = \max(x, 0)$. Thus, using the first term from the exact P-value, we have

$$P(g_m \geq z) \doteq m(1 - z/m)^{m-1} \doteq m\exp(-z) \doteq 1 - (1 - \exp(-z))^m. \tag{4.7}$$

(The last approximation is obtained by using the first term in the expansion of $1 - (1 - \exp(-z))^m$.)

For example, suppose we have a time series with $n = 200$ and the test statistic $g_m = 9.2$. Then the exact and the approximated P-values are 0.007 and 0.010.

References

Priestley, M. B. (1980). *Spectral Analysis and Time Series.* Academic Press, N.Y.

Schuster, A. (1898). On the investigation of hidden periodicities with application to a supposed 26-day period of meteorological phenomena. *Terr. Mag. Atmos. Elect.* **3** 13–41.

4.11 The cumulative periodogram

The cumulative periodogram is defined by

$$U(\lambda) = \frac{\sum_{j:\lambda_j \leq \lambda} I_j}{\sum_1^m I_j}.$$

Under the null hypothesis that the time series is a Gaussian white noise, then plotting the graph of $U(\lambda)$ vs λ should yield a straight line. Moreover the maximum deviation from that straight line has a distribution given by the Kolmogorov–Smirnov statistic, with a 95% confidence limits given by $\pm 1.358/[\sqrt{m} + 0.12 + 0.11/\sqrt{m}]$.

This test is also useful as a test of whether a particular spectral density is appropriate. If f is the spectral density under the null hypothesis, then the test can be carried out using the above cumulative periodogram $U(\lambda)$ with I_j replaced by $I_j/f(\lambda_j)$.

The following S function computes the U statistic and plots the confidence limits based on the Kolmogorov–Smirnov statistic.

```
# From Venebles and Ripley (1994, p.360)
#
cpgrm <- function(ts, taper = 0.1, main = paste("Series: ",
                    deparse(substitute(ts))))
{
x <- as.vector(ts)
x <- x[!is.na(x)]
x <- spec.taper(scale(x, T, F), p = taper)
y <- Mod(fft(x))^2/length(x)
```

```
y[1] <- 0
n <- length(x)
x <- ((0:(n/2)) * frequency(ts))/n
if(length(x) %% 2 == 0) {
n <- length(x) - 1
y <- y[1:n]
x <- x[1:n]
}
else y <- y[1:length(x)]
xm <- frequency(ts)/2
mp <- length(x) - 1
crit <- 1.358/(sqrt(mp) + 0.12 + 0.11/sqrt(mp))
oldpty <- par()$pty
par(pty = "s")
plot(x, cumsum(y)/sum(y), type = "s", xlim = c(0, xm),
ylim = c(0, 1), xaxs = "i", yaxs = "i", xlab =
"frequency", ylab = "", pty = "s")
lines(c(0, xm * (1 - crit)), c(crit, 1))
lines(c(xm * crit, xm), c(0, 1 - crit))
title(main = main)
par(pty = oldpty)
invisible()
}
```

Examples. The cumulative periodograms of the blood pressure and hormone data are depicted in Figures 4.13 and 4.14.

```
> cpgrm(dbp)
> cpgrm(sbp) # Figure 4.13

> cpgrm(lh[,1])
> cpgrm(lh[,2])
> cpgrm(lh[,3])
> cpgrm(lh[,4]) # Figure 4.14
```

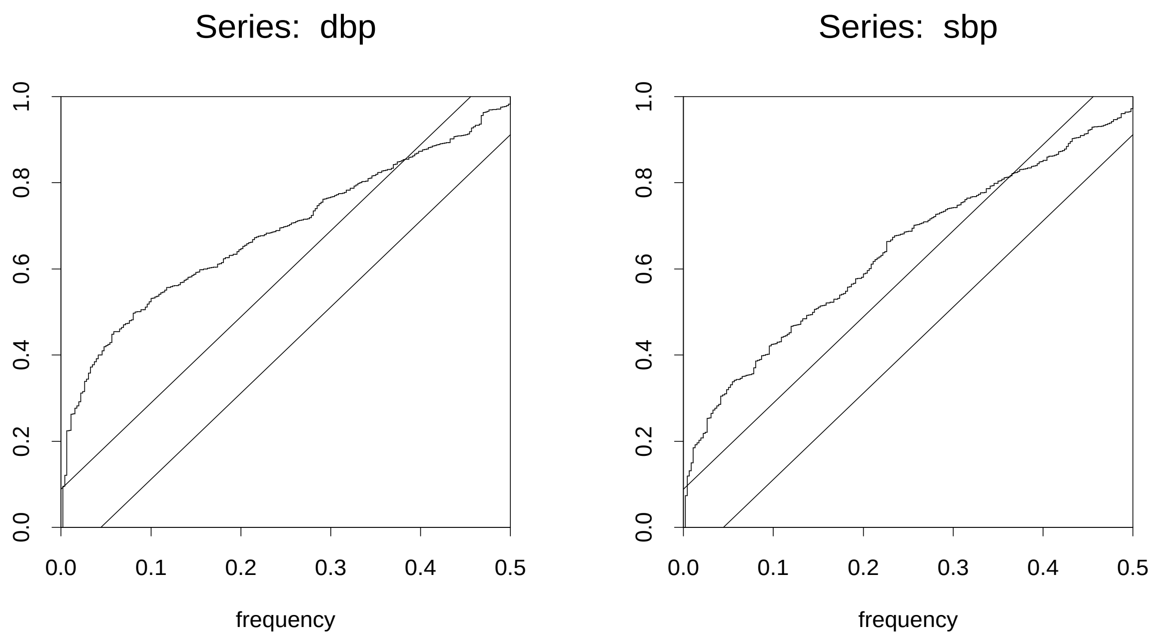
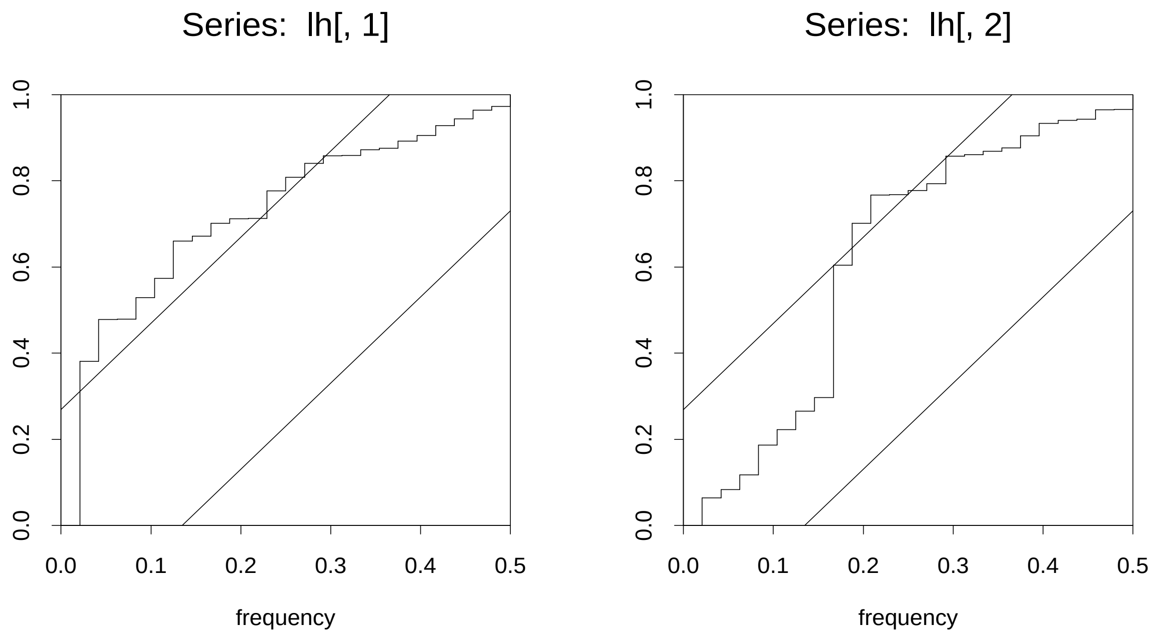


Fig 4.13 *Cumulative periodograms of the BP series.*



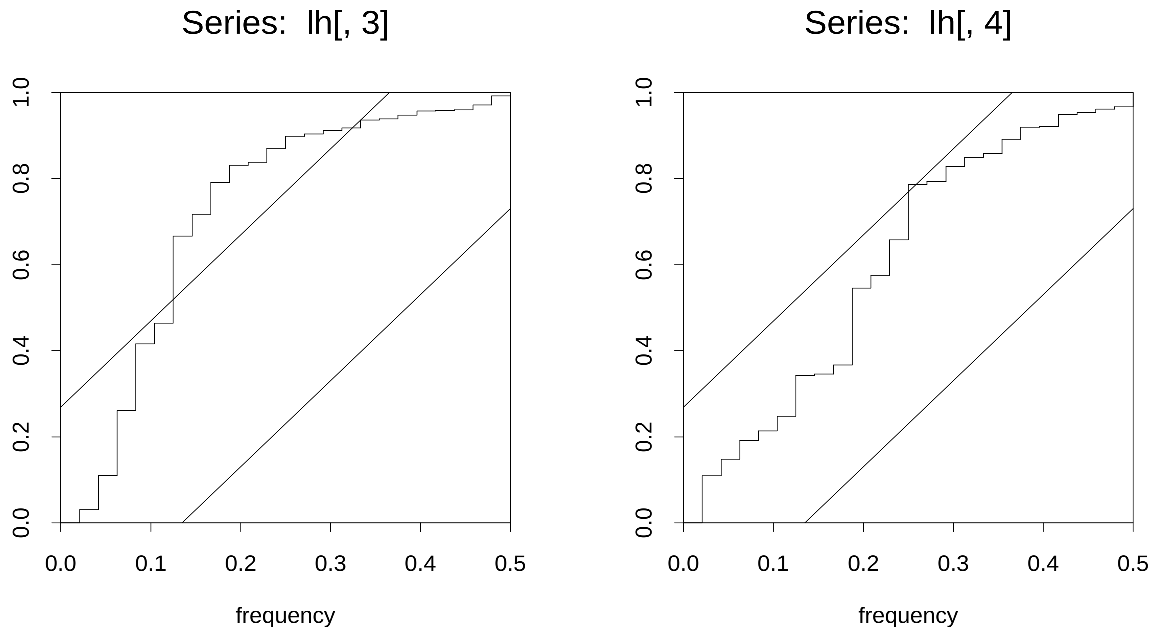


Fig 4.14 *Cumulative periodograms of the hormone series.*

4.12 Appendix 4A. Discrete Fourier Transform

4.12.1 Preliminary

Let $\mathbf{C}$ denote the set of complex numbers and let $\mathbf{C}^n$ be the set of n-tuples with complex components. That is,

$$\mathbf{C}^n \equiv \{\mathbf{x} : \mathbf{x} = (x_1, \ldots, x_n)^\top, x_i \in \mathbf{C}, i = 1, \ldots, n\}.$$

Recall that the set $\mathbf{C}^n$ has a natural inner product and metric (distance function) defined by

$$\langle \mathbf{x}, \mathbf{y} \rangle \equiv \sum_1^n x_j \overline{y_j}, \qquad \|\mathbf{x}\| \equiv \langle \mathbf{x}, \mathbf{x} \rangle^{1/2}.$$

4.12.2 Orthonormal bases

$\{\mathbf{u}_k, k = 1, 2, \ldots\}$ is said to be a set of *orthonormal* bases if

$$\langle \mathbf{u}_j, \mathbf{u}_k \rangle = \begin{cases} 1, & \text{if } j = k; \\ 0, & \text{otherwise.} \end{cases}$$

It can be shown that every orthonormal basis in $\mathbf{C}^n$ has exactly n elements. Given an orthonormal basis $\{\mathbf{u}_k : k = 1, \ldots, n\}$, all the members of $\mathbf{C}^n$ can be written as

$$\mathbf{x} = a_1 \mathbf{u}_1 + \cdots + a_n \mathbf{u}_n,$$

and this representation is unique since $a_k = \langle \mathbf{x}, \mathbf{u}_k \rangle$. A useful orthonormal basis for $\mathbf{C}^n$ is the set

$$\{\mathbf{u}_k \equiv n^{-1/2}(1, e^{i\lambda_{k-1}}, \ldots, e^{i\lambda_{k-1}(n-1)})', \quad k = 1, \ldots, n\},$$

where $\lambda_j = 2\pi j/n$, $j = 0, 1, \ldots, n-1$ are the Fourier frequencies. Given $\mathbf{x} \in \mathbf{C}^n$. Then

$$\mathbf{x} = \sum_1^n \check{x}_k \mathbf{u}_k,$$

where

$$\check{x}_k = \langle \mathbf{x}, \mathbf{u}_k \rangle = \frac{1}{\sqrt{n}} \sum_{t=1}^n x_t e^{-i\lambda_{k-1}(t-1)}.$$

4.12.3 Discrete Fourier Transformation

$J_k = \sum_1^n x_t e^{-i\lambda_{k-1}(t-1)}$ is called the k-th Fourier transform of $\mathbf{x}$. We have $J_k = \sqrt{n}\check{x}_k$. Note that

$$x_t = \frac{1}{\sqrt{n}} \sum_1^n \check{x}_k e^{i\lambda_{k-1}(t-1)}, \qquad t = 1, \ldots, n.$$

Thus data can be recovered from its Fourier transformation. This is called the inverse Fourier transformation.

Given a time series x, the function `fft(x)` will perform a fast Fourier transformation (FFT) on x. The inverse FFT is achieved via `fft(x, inverse=T)`.

4.12.4 Parseval identity

Given $\mathbf{x} \in \mathbf{C}^n$. Then by the orthogonality of $e^{-i\lambda_j \cdot}$,

$$\|\mathbf{x}\|^2 = \sum_1^n |\check{x}_k|^2 = n\overline{x} + \sum_2^n |\check{x}_k|^2.$$

Hence

$$\frac{1}{n} \sum_2^n |\check{x}_k|^2 = \frac{1}{n} \sum_1^n x_t^2 - \overline{x} = \hat{\sigma}^2.$$

That is, the total sum of squares of $\mathbf{x}$ is expressed as the total sum of squares of orthogonal components. This is called the spectral ANOVA and can also be viewed as a discrete approximation to

$$\int f(\lambda) d\lambda = \sigma^2.$$

4.12.5 Periodogram

A graph of $|\check{x}_k|^2$ vs k is called the periodogram and $|\check{x}_k|^2$ is called the periodogram ordinate.

4.13 Appendix 4B. Complex normal distribution

Let X and Y be real-valued random variables. Define a complex-valued random variable Z by $Z = X + iY$. The expectation of Z is defined as

$$E(Z) = E(X) + iE(Y).$$

If $\mathbf{Z}$ is a random matrix $\mathbf{Z} = [Z_{ij}]$, define its expectation by

$$E(\mathbf{Z}) = [E(Z_{ij})].$$

Suppose $\mathbf{Z}_1$ and $\mathbf{Z}_2$ are two $n \times 1$ complex random vectors. Define

$$\operatorname{cov}(\mathbf{Z}_1, \mathbf{Z}_2) = E(\mathbf{Z}_1 - E(\mathbf{Z}_1))\overline{(\mathbf{Z}_2 - E(\mathbf{Z}_2))}'$$

Now if $\mathbf{Z} = \mathbf{X} + i\mathbf{Y}$, then it can be shown that

$$\operatorname{cov}(\mathbf{Z}) = \operatorname{cov}(\mathbf{X}) + \operatorname{cov}(\mathbf{Y}) + i(\operatorname{cov}(\mathbf{Y}, \mathbf{X}) - \operatorname{cov}(\mathbf{X}, \mathbf{Y})$$

Thus if we want to write $\operatorname{cov}(\mathbf{Z}) = \mathbf{C} + i\mathbf{Q}$, then

$$\operatorname{cov}(\mathbf{X}) + \operatorname{cov}(\mathbf{Y}) = \mathbf{C}$$

$$\operatorname{cov}(\mathbf{Y}, \mathbf{X}) - \operatorname{cov}(\mathbf{X}, \mathbf{Y}) = \mathbf{Q}.$$

Hence,

$$\operatorname{cov}(\mathbf{X}) = \operatorname{cov}(\mathbf{Y}) = \mathbf{C}/2$$
$$\operatorname{cov}(\mathbf{X}, \mathbf{Y}) = \operatorname{cov}(\mathbf{Y}, \mathbf{X})' = -\mathbf{Q}/2.$$

The random variable $\mathbf{Z} = \mathbf{X} + i\mathbf{Y}$ is siad to have a complex normal distribution iff

$$\begin{pmatrix} \mathbf{X} \\ \mathbf{Y} \end{pmatrix} \sim N\left(\begin{pmatrix} E\mathbf{X} \\ E\mathbf{Y} \end{pmatrix}, \frac{1}{2} \begin{pmatrix} \mathbf{C} & -\mathbf{Q} \\ \mathbf{Q} & \mathbf{C} \end{pmatrix} \right).$$

We abbreviate this by $\mathbf{Z} \sim N^C(\mu_{\mathbf{Z}}, \Sigma_{\mathbf{Z}})$.

4.14 Appendix 4C. Normal approximation to $\log(\chi^2_\nu/\nu)$

Let $\bar{X}$ denote the sample mean with $E(\bar{X}) = \mu$ and $\mathrm{var}(\bar{X}) = \sigma^2/n$. If f is a function with bounded second derivative, then by Taylor approximation

$$f(\bar{X}) \approx f(\mu) + (\bar{X} - \mu)f'(\mu) \sim_a N(f(\mu), \sigma^2[f'(\mu)]^2/n).$$

Now put $\bar{X} = \chi^2_\nu/\nu$ and $f(\cdot) = \log(\cdot) = \log_{10}(\cdot)$. Note that $f'(\mu) = (\mu \log_e 10)^{-1}$. Hence

$$\mu = E(\chi^2_\nu/\nu) = 1, \qquad \mathrm{var}(\chi^2_\nu/\nu) = 2/\nu$$

and

$$\log(\chi^2_\nu/\nu) \sim_a N(0, 2(.4343)^2/\nu).$$

4.14.1 Gamma distributions

$X \sim \Gamma(p, \lambda)$ iff

$$f_X(x) = g_{p,\lambda}(x) = \frac{\lambda^p x^{p-1} e^{-\lambda x}}{\Gamma(\lambda)}, \quad x > 0, \qquad \Gamma(\lambda) = \int_0^\infty t^{p-1} e^{-t} dt.$$

Note that $\Gamma(p+1) = p\Gamma(p)$ and $\Gamma(1) = 1$. Thus $\Gamma(n+1) = n!$.

4.14.2 χ^2 distributions

If $X \sim \Gamma(\frac{m}{2}, \frac{1}{2})$, then we say X has a chi-squared distribution with m df. That is, $\chi^2_m = \Gamma(\frac{m}{2}, \frac{1}{2})$. We also have

$$E(\chi^2_m) = m, \qquad \mathrm{var}(\chi^2_m) = 2m.$$

Also, note that $\exp(1) = \chi^2_2/2$.

4.14.3 F distributions

Let X and Y be independent rv with $X \sim \chi^2_m$ and $Y \sim \chi^2_n$. Then

$$F = \frac{X/m}{Y/n}$$

has a F-distribution with m, n df.

4.15 Appendix 4D. Density estimation

Kernel density estimate. Let X and Z be independent random variables such that X has a distribution F and Z has a density function g. Let σ denote a positive constant. Then the random variable $X + \sigma Z$ has a density function given by

$$f_{X+\sigma Z}(y) = \frac{1}{\sigma}\int g\left(\frac{y-x}{\sigma}\right)\, dF(x) = \frac{1}{\sigma}E\left\{g\left(\frac{y-X}{\sigma}\right)\right\}.$$

Note that we do not assume that X has a density function, but we can still study the shape of the distribution of X by using this *perturbation* idea. If X has a continuous density function $f = F'$, then the density function of $X + \sigma Z$ tends to the density function of X as $\sigma \to 0$. That is,

$$f_{X+\sigma Z}(y) \to f_X(y) = f(y) \qquad \text{as } \sigma \to 0.$$

Or, equivalently,

$$\frac{1}{\sigma}\int g\left(\frac{y-x}{\sigma}\right)\, dF(x) = \frac{1}{\sigma}E\left\{g\left(\frac{y-X}{\sigma}\right)\right\} \to f(y) \qquad \text{as } \sigma \to 0.$$

In fact, this can be verified directly using dominated convergence theorem.

Let $X_1, \ldots, X_n$ be a random sample from f. Then by the method of moments,

$$\frac{1}{n}\sum_{i=1}^{n} g\left(\frac{y-X_i}{\sigma}\right)$$

is a reasonable estimate of $E\{g((y-X)/\sigma)\}$. Thus

$$\hat{f}(y) = \frac{1}{n\sigma}\sum_{i=1}^{n} g\left(\frac{y-X_i}{\sigma}\right)$$

is a reasonable estimate of $f(y)$ provided σ is small.

Note that we can construct the estimate of f using any distribution to perturb the data. The parameter σ is called the bandwidth. This distribution is referred to as the kernel function in the density estimation literature and is usually denoted by $K(\cdot)$. Below is a list of commonly used kernel functions:

Kernel	$K(t)$
Epanechnikov	$3(1-t^2/5)/4\sqrt{5} \times 1(\lvert t\rvert < \sqrt{5})$
Biweight	$15(1-t^2)^2/16 \times 1(\lvert t\rvert < 1)$
Triangular	$(1-\lvert t\rvert) \times 1(\lvert t\rvert < 1)$
Gaussian	$\exp(-t^2/2)/\sqrt{2\pi}$
Uniform	$1(\lvert t\rvert < 1/2)$

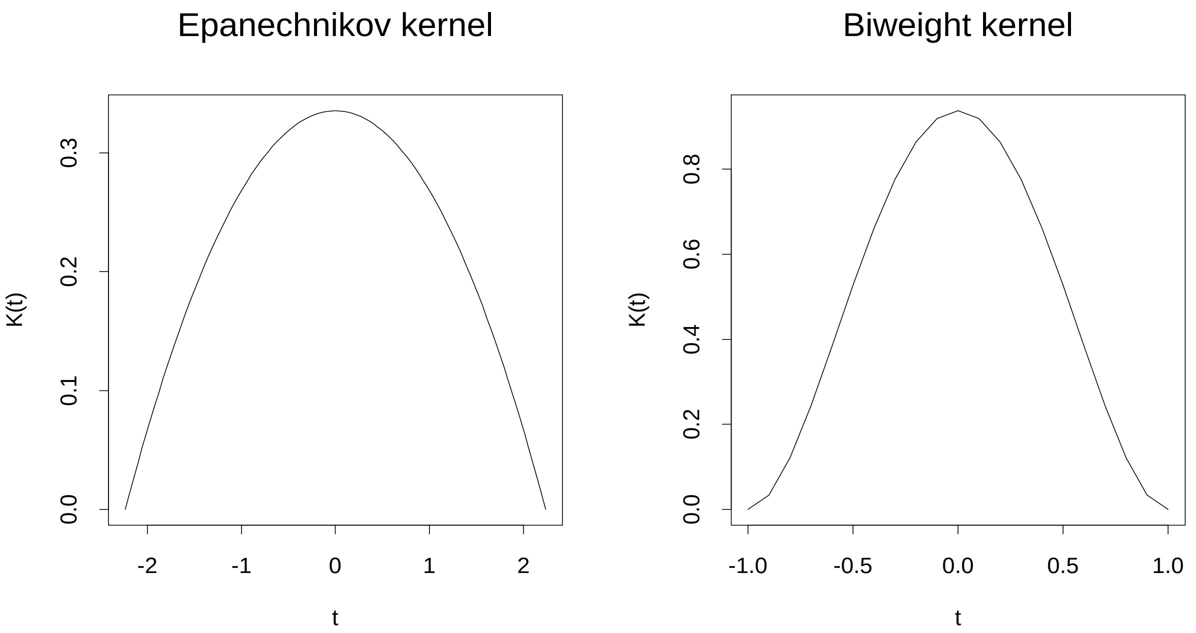


Fig 4.7a *Kernel functions.*

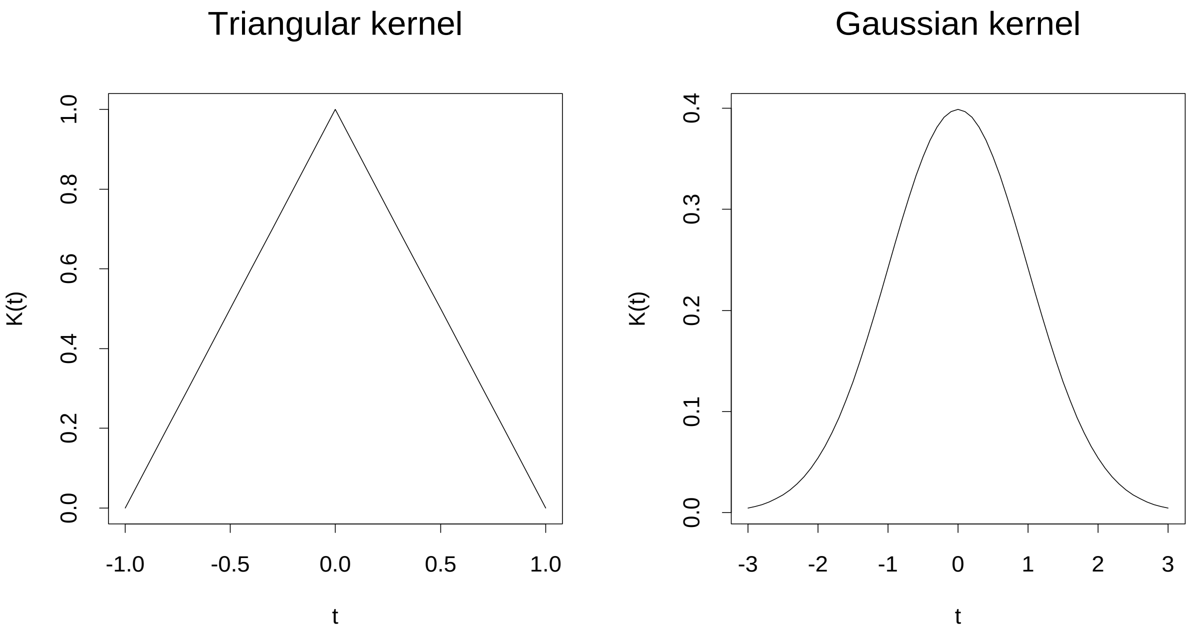


Fig 4.7b *Kernel functions.*

Sampling properties. Many basic statistical properties can be derived by taking Z as a uniform random variable on $[-1/2, 1/2]$. That is, $K(z) = 1$ for $|z| \le 1/2$ and 0 otherwise. Then $\sum_{i=1}^{n} K\left(\frac{y-X_i}{\sigma}\right)$ is a binomial random variable with parameters (n, p), where $p = F(y + \frac{1}{2}\sigma) - F(y - \frac{1}{2}\sigma)$.

1. It follows from the mean of a binomial rv and the definition of derivative,

$$E(\hat{f}(y)) = \frac{F(y + \frac{1}{2}\sigma) - F(y - \frac{1}{2}\sigma)}{\sigma} \to f(y) \qquad \text{as } \sigma \to 0.$$

   That is, the estimate $\hat{f}(y)$ is *asymptotically unbiased* when the bandwidth $\sigma \to 0$.

2. Recall the variance of a binomial rv is $np(1-p)$. The variance of $\hat{f}(y)$ is

$$\text{var}(\hat{f}(y)) = \frac{1}{n\sigma} \frac{F(y + \frac{1}{2}\sigma) - F(y - \frac{1}{2}\sigma)}{\sigma} \left\{1 - F(y + \tfrac{1}{2}\sigma) + F(y - \tfrac{1}{2}\sigma)\right\}.$$

   Thus the variance tends to zero if $n\sigma \to \infty$.

3. Consistency is achieved when bias and variance both tend to zero. It is sufficient to choose σ such that $\sigma \to 0$ and $n\sigma \to \infty$. That is, we should let σ tend to zero, but should not let it go too fast.

Density estimation in S-PLUS In SPLUS, the function

```
density(x, n=50, window="g", na.rm=F, width, from, to, cut)
```

computes the kernel estimate based on your data `x`. Here `n` is the number of equally spaced points at which to estimate the density, `window` is the kernel function $K(\cdot)$ and is one of the `cosine, gaussian, rectangular, triangular`, `width` is the bandwidth ... [cf. pp 136–138 of Venables and Ripley (1994)].

Problems

The first four exercises will help you build a little library for spectral analysis.

1. Write a S function for computing the periodogram using `fft`. The inputs are a time series and the option for the mean removal. The return value is a vector of periodogram ordinates. Note that you only need to keep half of these values due to the basic properties of the periodogram.

2. Write a S function for plotting the periodogram ordinates. Design you graph so that the vertical axis is the periodogram and the horizontal axis is the frequency ranging from $[0, .5]$. The input to this function is the periodogram computed using the S function developed in the last problem.

3. Write a S function for finding the maximum periodogram ordinate and its corresponding frequency. The input is the periodogram and the return values are the maximum periodogram and its frequency. [Hint: You may find the function `order` useful for this purpose. Also, you can ignore the tied situation.]

4. Fisher's exact and approximated tests. This problem has two parts.

   (a) Write a S function for computing the Fisher's test. The input is a time series and the return value is the test.

   (b) Write a S function that computes the P-value of Fisher's test. This function has two inputs: the time series and the option for computing the P-value exactly or approximately. The return value is the P-value.

   Once these have been tested and debugged, combine them into one function that takes the time series and the option for exact or approximate test, and returns the P-value.

5. Apply your S functions to test the periodogram of all the series listed in our lecture notes. In particular, carry out the test on the sunspots and lynx series. Comment on your findings briefly.

6. Verify that

$$\sum_{t=1}^{n} \cos(\lambda t) = \frac{\sin(\lambda n/2)}{\sin \lambda/2} \cos \lambda(n+1)/2$$

   and

$$\sum_{t=1}^{n} \sin(\lambda t) = \frac{\sin(\lambda n/2)}{\sin \lambda/2} \sin \lambda(n+1)/2.$$

7. This exercise establishes the relationship between the maximum periodogram ordinate and the amplitude and frequency of the simple harmonic process $x_t = A\cos(\lambda t + \phi)$, $t = 1, \ldots, n$.

   (a) Verify that the maximum periodogram ordinate is given by

$$\text{max periodogram} = nA^2/4.$$

   Hence the amplitude of the underlying signal is estimated by

$$\widehat{A} = \sqrt{4 \times \text{max periodogram}/n}.$$

   (b) Show that the simple harmonic process $x_t = a\cos(\lambda t) + b\sin(\lambda t)$ can be written as $x_t = A\cos(\lambda t + \phi)$, where $A = a/\cos(\phi)$ with $\phi = \arctan(-b/a)$. Thus the maximum periodogram is given by

$$\text{max periodogram} = n(a^2 + b^2)/4.$$

(c) Simulating a simple harmonic process with a given amplitude and requency, and then verify your above results by estimating the amplitude using the maximum periodogram ordinate.

(d) Generalized the above results to K harmonic components.

8. Suppose the frequency λ of the process $x_t = a\cos(\lambda t) + b\sin(\lambda t)$ is available (from the periodogram, say). Describe a regression method for estimating the initial values of a and b.

9. Simulate a simple harmonic series with a Fourier and a non-Fourier frequency. You may do this using `sim.cos` by adding a white noise to it.

   (a) Compute the periodogram for these series. Find the maximum ordinate and its corresponding frequency.

   (b) Test the periodogram of these series using the tests you had developed. Comment briefly.

   (c) Describe the relationship of the maximum ordinate and the amplitude of the simulated harmonic component. Use this relationship to construct an estimate of the underlying amplitude.

   (d) Remove the harmonic component using the estimated frequency and amplitude. Then carry out Fisher's test on the residual series. Compare your result with those obtained by using `cpgrm`.

   (e) Carry out an experiment by testing the significance of the maximum periodogram while gradually increasing the variance of your noise series. What kind of results you might expect to get?

10. Compute the `fft` of the series discussed in our notes. Then use `cpgrm` (provided in `cpgram.s`) to test the independence of the real and the imaginary parts of the `fft`. Compare your results with those obtained via the function `acf` and comment briefly.

11. Compute the power spectrum estimates of all the series in our notes by using several bandwidths. Construct the confidence intervals based on your estimates. Moreover, study the effect of bandwidth on the estimates.

Chapter 5

ARMA Processes

5.1 The B Operator

This is a very useful symbolic operator and is defined by

$$B^k x_t = x_{t-k}, \qquad k = 0, 1, 2, \ldots.$$

It is also called the backshift operator because $Bx_t = B^1 x_t = x_{t-1}$.

For the above AR(1) model, we have

$$x_t = aBx_t + e_t,$$

or

$$(1 - aB)x_t = e_t.$$

If $|aB| < 1$, then by the geometric series

$$\begin{aligned} x_t &= (1 - aB)^{-1} e_t = (1 + aB + a^2 B^2 + \cdots + a^u B^u + \cdots) e_t \\ &= e_t + a e_{t-1} + a^2 e_{t-2} + \cdots + a^u e_{t-u} + \cdots. \end{aligned}$$

Based on this representation, we see that the AR(1) series is strictly stationary.

5.2 Autoregressive processes

Autoregressive process of order 2 [AR(2)] is a special case of the following:

$$x_t - \phi_1 x_{t-1} - \phi_2 x_{t-2} - \cdots - \phi_p x_{t-p} = w_t, \qquad w_t \sim_{\text{iid}} (0, \sigma^2), \tag{5.1}$$

where w_t is independent of x_{t-j} $(j = 1, 2, \ldots)$ and $\phi_1, \ldots, \phi_p$ are called the parameters or coefficient of the process, which is refered to as $AR(p)$. Note that this process can also be abbreviated as

$$\phi(B)x_t = w_t$$

where $\phi(z) = 1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p$ is the autoregressive polynomial and B is the backshift operator.

5.2.1 Causality

An AR(p) process is called *causal* if the autoregressive polynomial $\phi(z)$ has no roots in the unit circle of the complex plane. That is, $\phi(\cdot)$ satisfies

$$\phi(z) \neq 0, \qquad z \in \mathbf{C}, |z| \leq 1.$$

Causality is a necessary and sufficient condition for an AR(p) process to be stationary. See for example, Brockwell and Davis (1991, p.85).

The roots of the polynomial $a_k z^k + \cdots + a_1 z + a_0 = 0$ can be found by using the SPLUS function `polyroot`. For example, to find the roots of $z^2 + 5z + 6 = 0$:

```
> polyroot(c(6,5,1))
[1] -2+5.169879e-026i -3-5.169879e-026i
> Mod(polyroot(c(6,5,1)))
[1] 2 3
```

5.2.2 Yule-Walker equations

Multiplying each side of (1) by x_{t-u} $(u = 0, 1, \ldots, p)$ and taking expectations, we obtain

$$c(u) - \phi_1 c(u-1) - \phi_2 c(u-2) - \cdots - \phi_p c(u-p) = 0, \qquad u = 1, \ldots, p, \tag{5.2}$$

and (using $c(-u) = c(u)$)

$$c(0) - \phi_1 c(1) - \phi_2 c(2) - \cdots - \phi_p c(p) = \sigma^2. \tag{5.3}$$

Eq. (5.2) is usually given in the following form

$$\begin{pmatrix} c(0) & c(1) & c(2) & \cdots & c(p-1) \\ c(1) & c(0) & c(1) & \cdots & c(p-2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c(p-1) & c(p-2) & c(p-3) & \cdots & c(0) \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_p \end{pmatrix} = \begin{pmatrix} c(1) \\ c(2) \\ \vdots \\ c(p) \end{pmatrix}.$$

Let Γ_p denote the $p \times p$ matrix $[c(i-j)]_{i,j=1}^{p}$ and $\phi' = (\phi_1, \ldots, \phi_p)$. Put $\mathbf{c}' = (c(1), \ldots, c(p))$. Then eqs. (5.2) and (5.3) can be written as

$$\Gamma_p \phi = \mathbf{c}, \quad \sigma^2 = c(0) - \phi' \mathbf{c}.$$

5.2.3 Computing the autocovariance function

The Yule-Walker equation (YWE) is particularly useful in computing the autocovariance function of an AR(p) process. Given the parameters $\phi_1, \ldots, \phi_p$, eqs. (5.2) and (5.3) can be combined to compute the $c(\cdots)$.

Example. Let $x_t - ax_{t-1} = w_t$ be an AR(1) process. Causality implies that $|a| < 1$. Now set up the YWE as follows

$$\begin{aligned} c(0) - ac(1) &= \sigma^2, \\ c(1) - ac(0) &= 0. \end{aligned}$$

Or equivalently, solving the following for $c(0)$ and $c(1)$:

$$\begin{pmatrix} 1 & -a \\ -a & 1 \end{pmatrix} \begin{pmatrix} c(0) \\ c(1) \end{pmatrix} = \begin{pmatrix} \sigma^2 \\ 0 \end{pmatrix}.$$

The other $c(u)$ can be computed recursively via

$$c(u) - ac(u-1) = 0, \qquad u = 2, 3, \ldots.$$

Besides computing the autocovariance function of an AR(p) process, the YWE can be used to compute the coefficients of the process. See eqs. (5.2) and (5.3). A useful application is to obtain the partial correlation of the an AR(p) process from its autocovariance function.

5.2.4 Partial autocorrelation (ACF)

The partial correlation function $\alpha(\cdot)$ is defined by

$$\alpha(1) = \mathrm{corr}(x_2, x_1),$$

$$\alpha(k) = \mathrm{corr}(x_{k+1} - E(x_{k+1}|x_k, \ldots, x_2), x_1 - E(x_1|x_k, \ldots, x_2)), \qquad k \geq 2.$$

Thus $\alpha(1) = \phi_1, \ldots, \alpha(p) = \phi_p$, and $\alpha(k) = 0$ for $k > p$.

5.2.5 Estimation

Given a realization $x_1, \ldots, x_T$ of an AR(p) process. We are interested in estimating the coefficients $\phi_1, \ldots, \phi_p$. Because of the YWE, this will provide the complete information about the structure of the process. Thus one can obtain the estimates by the method of moments. Namely, replace the autocovariance function by its estimate $\hat{c}(\cdot)$ and solve the YWE (5.2) and (5.3) for $\hat{\sigma}^2, \hat{\phi}_1, \ldots, \hat{\phi}_p$. We have

Theorem 4 *For an AR(p) process,*

$$\sqrt{n}(\hat{\phi} - \phi) \to N(\mathbf{0}, \Gamma_p^{-1}), \qquad \hat{\sigma}^2 \to \sigma^2,$$

where $\Gamma_p = [c(i-j)]$.

One can use this result to obtain an $100(1-\alpha)\%$ confidence interval for ϕ:

$$\{\phi \in \mathbb{R}^p : (\phi - \hat{\phi})' \hat{\Gamma}_p (\phi - \hat{\phi}) \leq n^{-1} \hat{\sigma}^2 \chi_p^2(1-\alpha)\},$$

where $\chi_p^2(1-\alpha)$ denotes the upper α percentile of a chi-squared distribution with p df.

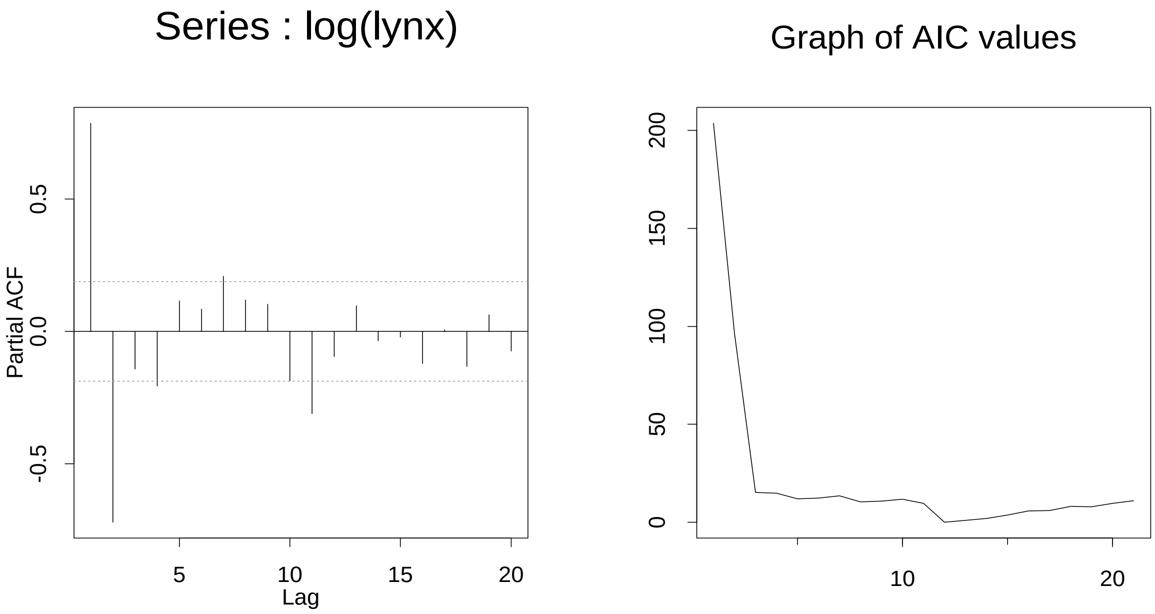


Fig 5.1 *The partial autocorrelation (ACF) and Akaike Information Criterion (AIC).*

5.2.6 Computing the estimates

The general SPLUS function for AR estimation is `ar`.

```
ar(x, aic=T, order.max, method="yule-walker")
# method = "y" (yule-walker) or "b" (burg)

EXAMPLES

> a <- ar(log(lynx))
> a$order # AR order
[1] 11
> a$ar    # coefficients
, , 1
             [,1]
 [1,]  1.13871050
 [2,] -0.50803661
 [3,]  0.21265374
 [4,] -0.27017918
 [5,]  0.11269173
 [6,] -0.12398130
 [7,]  0.06772482
 [8,] -0.04004321
 [9,]  0.13370071
```

```
[10,]  0.18527256
[11,] -0.31095836
> a$var.pred      # predicted variance
          [,1]
[1,] 0.2529533
> a$aic
 [1] 203.6860046  96.4504395  15.1466675  14.7889709  11.8372498  12.3137207
 [7]  13.4956665  10.4664917  10.8578491  11.6462402   9.5932312   0.0000000
[13]   0.9552612   1.8796692   3.7330933   5.6779480   6.0010681   7.9973755
[19]   7.9676208   9.5235291  10.8982544
> a$order.max
[1] 20
> a$partialacf    # Partial acf

, , 1
              [,1]
 [1,]  0.785124123
 [2,] -0.720031440
 [3,] -0.143070951
 [4,] -0.206170604
 [5,]  0.115216792
 [6,]  0.084558606
 [7,]  0.207742065
 [8,]  0.118370444
 [9,]  0.102817982
[10,] -0.186890379
[11,] -0.310958356
[12,] -0.095509544
[13,]  0.096902721
[14,] -0.035859577
[15,] -0.021977440
[16,] -0.120840460
[17,]  0.005669787
[18,] -0.132842585
[19,]  0.062357444
[20,] -0.073957071
> a$resid # not shown

> acf.plot(a, conf=T)     # Figure 5-1
tsplot(a$aic)
# Fit an AR(11) to this time series
lynx.fit <- ar(log(lynx), aic=F, order=11, method="b")
```

```
# function to predict using an ar model:
# ahead gives the number of predictions to make

pred.ar <- function(series, ar.est, ahead = 1)
{
        order <- ar.est$order
        series <- as.matrix(series)
        pred.out <- array(NA, dim = c(order + ahead, ncol(series)), dimnames =
                list(NULL, dimnames(series)[[2]]))
        mean.ser <- apply(series, 2, mean)

        ser.cent <- sweep(series, 2, mean.ser)
        pred.out[seq(order),  ] <- ser.cent[rev(nrow(series) - seq(order) +
                1),  ]
        for(i in (order + 1):nrow(pred.out)) {
                pred.out[i,  ] <- apply(aperm(ar.est$ar, c(1, 3, 2)) *
                        as.vector(pred.out[i - seq(order),  ]), 3, sum)
        }
        sweep(pred.out[ - seq(order),  , drop = F], 2, mean.ser, "+")
}


sun.ar <- ar(sunspots)
tsplot(sun.ar$resid, main="Residuals from AR(27) Fit of Sunspots")

# the structure in the residuals means the model is inadequate
```

If method="yule-walker", the autocovariance matrices of the time series are estimated and Whittle's recursion [a multivariate extension of the Levinson-Durbin algorithm (cf. Section 5.2.7)] is used to estimate the autoregressive coefficients. The coefficients can also be estimated by using Burg's algorithm, based on the principle of maximum entropy. The output may be used in spec.ar to estimate the spectrum of the process, or acf.plot to produce a plot of the partial autocorrelation function.

See also `ar.yw` and `ar.burg`.

```
ar.yw(x, aic=T, order.max)      # the YW method
ar.burg(x, aic=T, order.max)    # the Burg method
```

Partial correlation can be estimated via the SPLUS function `acf`:

```
acf(x, lag.max, type="partial", ...)
```

Alternatively, one can use `ar$partialacf` to yield a vector of the partial autocorrelation coefficients (this is computed only if method="yule-walker").

```
a <- ar.yw(log(lynx))
acf.plot(a, conf=T)     # look at the partial correlations
```

5.2.7 The Durbin-Levinson Algorithm

Solving the YWE requries matrix inversion and this can be quite time consuming when large order of AR process has to be fitted, especially in identifying the order the model. However, there is a better way to do so and this brings us to the following algorithm.

This algorithm provides a sequence of estimates for all AR(m). Put

$$x_t - \hat{\phi}_{m1}x_{t-1} - \hat{\phi}_{m2}x_{t-2} - \cdots - \hat{\phi}_{mm}x_{t-m} = w_t, \qquad w_t \sim_{\text{iid}} (0, \hat{\sigma}_m^2).$$

Then $\hat{\phi}_{m1}, \hat{\phi}_{m2}, \ldots - \hat{\phi}_{mm}$ are given by

$$\begin{aligned}
\hat{\phi}_{11} &= \hat{c}(1)/\hat{c}(0), \quad \hat{\sigma}_1^2 = \hat{c}(0)[1 - \hat{\phi}_{11}^2], \\
\hat{\phi}_{mm} &= \left[\hat{c}(m) - \sum_{j=1}^{m-1} \hat{\phi}_{m-1,j}\hat{c}(m-j)\right] / \hat{\sigma}_{m-1}^2, \\
\begin{pmatrix} \hat{\phi}_{m1} \\ \vdots \\ \hat{\phi}_{m,m-1} \end{pmatrix} &= \begin{pmatrix} \hat{\phi}_{m-1,1} \\ \vdots \\ \hat{\phi}_{m-1,m-1} \end{pmatrix} - \hat{\phi}_{mm} \begin{pmatrix} \hat{\phi}_{m-1,m-1} \\ \vdots \\ \hat{\phi}_{m-1,1} \end{pmatrix}, \\
\hat{\sigma}_m^2 &= \hat{\sigma}_{m-1}^2(1 - \hat{\phi}_{mm}^2).
\end{aligned}$$

The usefulness of this algorithm is its recursive way of getting the estimate. We can also obtain estimate the partial autocovariance function of any order from this algorithm. Namely,

$$\hat{\alpha}(k) = \hat{\phi}_{kk}.$$

We know that for an AR(p), the partial correlation of order great than p is zero. Thus this provides a diagnostic tool for model identification. Since the partial correlations are estimated from the data, we can use the following result to get confidence intervals for $\alpha(k)$ when $k > p$:

Theorem 5 $\sqrt{n}\hat{\phi}_{kk} \to N(0,1)$ *for* $k > p$.

Thus if the time series is an AR(p) process, then $|\hat{\alpha}(k)| < 1.96/\sqrt{n}$ with a probability close to 0.95 for $k > p$.

5.2.8 Burg's algorithm for AR(p)

Observe that the coefficient ϕ_{mm} in the Durbin-Levinson algorithm is the partial correlation at lag m, $\alpha(m)$. Thus a new class of AR estimates can be obtained by providing a different estimate of $\alpha(m)$. Burg (1967) suggests that

$$\hat{\alpha}(m) = \hat{\phi}_{m,m} = \frac{-2\tilde{\mathbf{f}}_{m-1}^T \tilde{\mathbf{e}}_{m-1}}{\tilde{\mathbf{f}}_{m-1}^T \tilde{\mathbf{f}}_{m-1} + \tilde{\mathbf{e}}_{m-1}^T \tilde{\mathbf{e}}_{m-1}},$$

where $\tilde{\mathbf{f}}_{m-1}^T = (f_{m-1}(m), \ldots, f_{m-1}(n))$ is the vector of forward prediction errors defined by

$$f_{m-1}(k) = x(k) - \sum_{i=1}^{p} \phi_i x(k-i), \quad k = m, \ldots, n.$$

Similarly, $\tilde{\mathbf{e}}_{m-1}^T = (e_{m-1}(m), \ldots, e_{m-1}(n))$ is the vector of backward prediction errors defined by

$$e_{m-1}(k) = x(k-m+1) - \sum_{i=1}^{p} \phi_i x(k-m+1+i), \quad k = m, \ldots, n.$$

Burg also observes that these forward and backward errors satisfy the following recursive relationships

$$f_m(k) = f_{m-1}(k) + \hat{\phi}_{mm} f_{m-1}(k-1), \quad f_0(k) = x(k),$$

$$e_m(k) = e_{m-1}(k-1) + \hat{\phi}_{mm} e_{m-1}(k), \quad e_0(k) = x(k).$$

Conditional likelihood

Assume $w_t \sim_{\text{iid}} N(0, \sigma^2)$. Then the likelihood function of $w_{p+1}, \ldots, w_n$ is

$$\ell(\phi_1, \ldots, \phi_p, \sigma^2) = \log L(\phi_1, \ldots, \phi_p, \sigma^2) = -\frac{1}{2}(n-p)\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{j=p+1}^{n} w_j^2.$$

To express this in terms of the original observations, note that the transformation from $w_{p+1}, \ldots, w_n$ to $x_{p+1}, \ldots, x_n$ subject to $x_1, \ldots, x_p$ fixed has a Jacobian $= 1$. Thus

$$\ell(\phi_1, \ldots, \phi_p, \sigma^2) = -\frac{1}{2}(n-p)\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{j=p+1}^{n}\left(x_j - \sum_{k=1}^{p}\phi_{j-k}x_{j-k}\right)^2.$$

The MLE of $\phi_1, \ldots, \phi_p$ can be seen to be the solution of the YWE based on $x_{p+1}, \ldots, x_n$. For details, see Shumway (1988, p.147).

5.2.9 AR spectral estimates

To end the discussion on AR process, we mention that AR provides an useful way of estimating the spectral density of an univariate series. This is sometimes referred to as prewhitten-spectral analysis. To obtain an AR spectral estimate, use the function `spec.ar` in SPLUS.

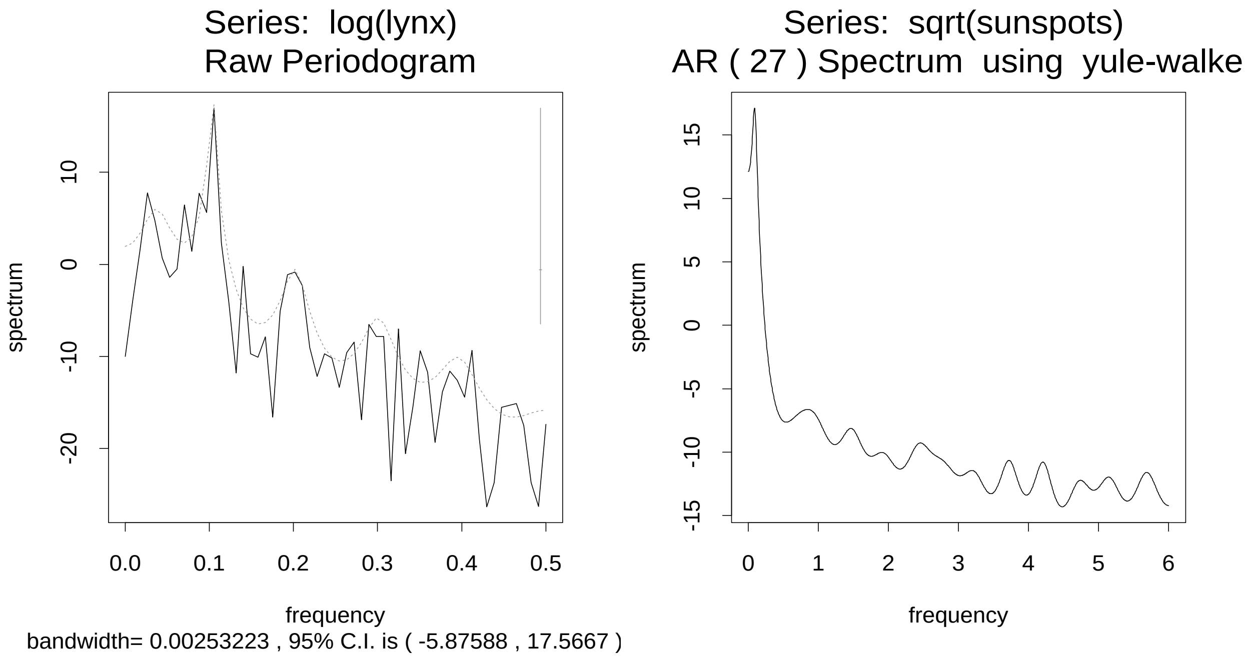


Fig 5.2 *AR spectral estimates of the log-lynx data and the square root of sunspots data.*

```
ll.ar <- ar(log(lynx))
spec <- spec.pgram(log(lynx),plot=T)
ll.spec.ar <- spec.ar(ll.ar, plot=F)
spec.plot(ll.spec.ar, add=T) # Figure 5-2

ss.ar <- ar(sqrt(sunspots))
ss.spec.ar <- spec.ar(ss.ar, plot=T)  # Figure 5-2
```

5.3 Moving-average processes

A time series x_t is called a moving average of order q, abbreviated as $MA(q)$, iff

$$x_t = w_t + \theta_1 w_{t-1} + \cdots + \theta_q w_{t-q}, \quad w_t \sim_{\text{iid}} (0, \sigma^2).$$

Note that this is a stationary time series and its autocovariance function is given by

$$c(u) = \begin{cases} \sigma^2 \sum_{k=0}^{q-|u|} \theta_k \theta_{k+|u|}, & |u| \leq q \\ 0, & |u| > q. \end{cases}$$

Thus the AR has an autocovariance function which "dies out gradually", the autocovariance function of a moving average process "cuts off" after the point q.

5.3.1 Invertibility

There are $2^q - 1$ sets of σ^2 and θ's that arise from a given value of autocovariances $c(0), \dots, c(q)$. For example, given $c(0) = 5$ and $c(1) = 2$. Then $(\sigma^2 = 1, \theta_1 = 2)$ and $(\sigma^2 = 4, \theta_1 = .5)$ give the same autocovariance structure of a MA(1). To achieve the uniqueness, we require the following condition:

$$\theta(z) := 1 + \theta_1 z + \cdots + \theta_q z^q \neq 0, \quad |z| \leq 1,$$

where $\theta(z)$ is called the characteristic polynomial. That is, the characteristic polynomial does not have roots lying inside the unit disc of the complex plane. This is also called the invertibility condition, since under this condition the process can be *inverted* to have an AR representation.

5.3.2 Estimation

Given a time series, we may want to model it by a MA(q). This implies that the parameters will have to be estimated. We mention two methods here, a third method will be introduced when we discuss the likelihood function approach.

Method of moments

We illustrate the idea by an example. Consider

$$x_t = w_t + \theta w_{t-1}, \quad w_t \sim_{\text{iid}} (0, \sigma^2).$$

Then

$$\hat{c}(0) = \hat{\sigma}^2 + \hat{\theta}\hat{\sigma}^2, \qquad \hat{c}(1) = \hat{\theta}\hat{\sigma}^2.$$

Thus

$$\hat{\theta}^2 - a\hat{\theta} + 1 = 0, \qquad a = \hat{c}(0)/\hat{c}(1).$$

This equation has real roots iff $|a| \geq 2$ or $|\hat{r}(1)| = |\hat{c}(1)/\hat{c}(0)| \leq 1/2$. If $|\hat{r}(1)| > 1/2$, we define $\hat{\theta} = \text{sgn}(\hat{r}(1))$. Thus

$$\hat{\theta} = g(r(1)),$$

where

$$g(x) = \begin{cases} -1, & \text{if } x < -1/2, \\ (1 - (1 - 4x^2)^{1/2})/2x, & \text{if } |x| \leq 1/2, \\ 1, & \text{if } x > 1/2. \end{cases}$$

Further, it can be shown that

$$\hat{\theta} \sim_a N(g(\rho(1)), \sigma_1^2/n),$$

where

$$\sigma_1^2 = (1 + \theta^2 + 4\theta^4 + \theta^6 + \theta^8)/(1 - \theta^2)^2.$$

Wilson (1979) has developed an algorithm for calculating these estimates for general MA(q).

Method of innovations

The idea is to find $\hat{\theta}_{m1}, \ldots, \hat{\theta}_{mm}$ so that

$$\hat{x}_t = w_t + \hat{\theta}_{m1} w_{t-1} + \cdots + \hat{\theta}_{mm} w_{t-m}, \quad w_j \sim_{\text{iid}} (0, v_m^2).$$

The algorithm is:

$$\begin{aligned} v_0 &= \hat{c}(0), \\ \hat{\theta}_{m,m-k} &= v_k \left[\hat{c}(m-k) - \sum_{j=0}^{k-1} \hat{\theta}_{m,m-j} \hat{\theta}_{k,k-j} v_j \right], \quad k = 0, \ldots, m-1, \\ v_m &= \hat{c}(0) - \sum_{j=0}^{m-1} \hat{\theta}^2_{m,m-j} v_j. \end{aligned}$$

For $m = o(n^{1/3})$, it can be shown that

$$n^{1/2}(\hat{\theta}_{m,j} - \theta_j) \to N\left(0, \sum_{k=1}^{j} \theta^2_{j-k}\right), \qquad j = 1, \ldots, q.$$

Moreover,

$$v_m \to \sigma^2.$$

The asymptotic variance can be further estimated by $\hat{\sigma}_j^2 = \sum_{k=0}^{j-1} \hat{\theta}^2_{m,k}$. This result suggests that if $\hat{\theta}_{m,j}$ lies outside the bounds $\pm 1.96 \hat{\sigma}_j$, θ_j is probably not zero.

Remark. These estimates are refered to as the "preliminary esitmates" in Brockwell and Davis (1987).

In practice, we will have to estimate the order q of the process as well. For the moment, note that the autocorrelation $r(u) = c(u)/c(0) = 0$, for $|u| > q$. Also, the estimated correlation

$$\hat{r}(u) \sim_a N(0, n^{-1}[1 + 2r^2(1) + \cdots + 2r^2(q)]) \quad |u| > q.$$

Thus the graph of ACF — correlograms — is used to test against white noise by checking to see if the $r(\hat{u})$ lies outside the bounds $1.96/\sqrt{n}$. More stringent bounds can be obtained by the above asymptotic result. Namely,

$$\hat{r}(u) \pm 1.96 \sqrt{[1 + 2r^2(1) + \cdots + 2r^2(q)]/n}$$

is a 95% confidence interval for $r(u)$ ($|u| > q$).

5.3.3 Conditional likelihood

Given a realization $x_1, \dots, x_n$ from a MA process, we would like to obtain the likelihood function by assuming

$$w_t \sim_{\text{iid}} N(0, \sigma^2).$$

Suppose $w_t = 0, t \leq 0$. Then

$$\begin{aligned}
w_1 &= x_1, \\
w_2 &= x_2 - \theta_1 w_1, \\
w_3 &= x_3 - \theta_1 w_1 - \theta_2 w_2, \\
&\vdots \\
w_n &= x_n - \theta_1 w_{n-1} - \theta_2 w_{n-2} - \cdots - \theta_q w_{n-q}.
\end{aligned}$$

This implies that w_t is a function of $\theta_1, \dots, \theta_q$ and we denote this by $w_t(\theta)$. Now the likelihood function is given by

$$\ell(\theta, \sigma^2) = \log L(\theta, \sigma^2) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{t=1}^{n} w_t^2(\theta).$$

For a given value of $\theta_1, \dots, \theta_q$, the likelihood function can be evaluated. Thus the MLEs can be obtained by maximising the likelihood function. That is,

$$(\hat{\theta}, \hat{\sigma}^2) = \max{}^{-1}\ell(\theta, \sigma^2).$$

Note that

$$\hat{\sigma}^2 = n^{-1}\sum w_t^2(\hat{\theta}_1, \dots, \hat{\theta}_q)$$

and $\hat{\theta}_1, \dots, \hat{\theta}_q$ are obtained via Newton-Gauss method by minimising $\sum_t w_t^2(\theta)$.

By Taylor expansion,

$$w_t(\theta) = w_t(\theta_0) + [Dw_t(\theta_0)](\theta - \theta_0),$$

where θ_0 is an initial solution and

$$[Dw_t(\theta_0)] = \begin{pmatrix} \partial w_t(\theta)/\partial\theta_1 \\ \partial w_t(\theta)/\partial\theta_2 \\ \vdots \\ \partial w_t(\theta)/\partial\theta_q \end{pmatrix}.$$

Now finding θ to minimize $\sum w_t^2(\theta)$ is equivalent to solving the regression problem

$$w_t(\theta_0) = -[Dw_t(\theta_0)](\theta - \theta_0) + w_t(\theta).$$

Hence

$$\theta - \theta_0 = \left(\sum [Dw_t(\theta_0)][Dw_t(\theta_0)]^T\right)^{-1} \sum [-Dw_t(\theta_0)]w_t(\theta_0).$$

Iterate this until $w_t(\theta)$ can not be reduced anymore. (In practice, this is controlled by user-specified accuracy and the maximum number of iterations.)

Note that the values of $w_t(\theta)$ and $D^k w_t(\theta)$ can be computed recursively, ($D^k = \partial/\partial\theta_k$)

$$w_t(\theta) = x_t + \sum_{j=1}^{q} \theta_j w_{t-j}(\theta),$$

$$D^k w_t(\theta) = D^k w_{t-k}(\theta) + \sum_{j=1}^{q} \theta_j (D^k w_{t-j}(\theta)), \qquad k = 1, \dots, q.$$

To maximize the likelihood function $\ell(\cdot)$ in general, it is required to solve

$$\ell'(\psi) = 0.$$

This usually involves iterative procedures. Given an initial solution ψ_0, the next (improved) solution is (so called Newton-Raphson method)

$$\psi_1 = \psi_0 - \ell''(\psi_0)^{-1}\ell'(\psi_0),$$

or (called the method of scoring)

$$\psi_1 = \psi_0 + E(-\ell''(\psi_0))^{-1}\ell'(\psi_0),$$

where $I(\psi_0) = E(-\ell''(\psi_0))$ is the information matrix.

5.4 Autoregressive-moving average (ARMA) process

Given nonnegative integers p and q, the ARMA process is defined by

$$x_t - \sum_{j=1}^{p} \phi_j x_{t-j} = w_t + \sum_{k=1}^{q} \theta_k w_{t-k}, \qquad w_k \sim_{\text{iid}} (0, \sigma^2).$$

That is, the process has both an autoregressive and moving average components.

5.4.1 Causality

This comes from the AR process. That is, the AR characteristic polynomial $\phi(z)$ has no roots inside the unit circle of the complex plane. That is, $\phi(\cdot)$ satisfies

$$\phi(z) \neq 0, \quad z \in \mathbf{C}, \quad |z| \leq 1.$$

5.4.2 Invertibility

This comes from the MA component, for which the characteristic polynomial satisfies

$$\theta(z) := 1 + \theta_1 z + \cdots + \theta_q z^q \neq 0, \quad |z| \leq 1.$$

5.4.3 Infinite moving average process

The ARMA process can be abbreviated as

$$\phi(B)x_t = \theta(B)w_t, \qquad \text{or} \qquad x_t = \phi^{-1}(B)\theta(B)w_t.$$

Suppose that the polynomials $\theta(z)$ and $\phi(z)$ do not have common roots. Then causality implies that there exists a sequence of constants $\{\psi_j\}$ such that $\sum |\psi_j| < \infty$ and

$$x_t = \sum_{j=0}^{\infty} \psi_j w_{t-j}.$$

Moreover, the sequence $\{\psi_j\}$ is determined by

$$\psi(z) = \sum \psi_j z^j = \theta(z)/\psi(z).$$

For a proof of this fact, see Brockwell and Davis (1987). This result is very useful in computing the covariance of ARMA processes. For, based on the above representation, the covariance function is (homework)

$$c(u) = \sum_{k=0}^{\infty} \psi_k \psi_{k+|u|}.$$

Note that this can easily be computed from

$$\psi(z)\phi(z) = \theta(z),$$

viz,

$$\begin{aligned} \psi_j - \sum_{0<k\leq j} \phi_k \psi_{j-k} &= \theta_j, \qquad 0 \leq j < \max\{p, q+1\}, \\ \psi_j - \sum_{0<k\leq p} \phi_k \psi_{j-k} &= 0, \qquad j \geq \max\{p, q+1\}. \end{aligned}$$

That is,

$$\begin{aligned} \psi_0 &= \theta_0 = 1, \\ \psi_1 &= \theta_1 + \psi_0\phi_1 = \theta_1 + \phi_1, \\ \psi_2 &= \theta_2 + \psi_0\phi_2 + \psi_1\phi_1, \\ &\cdots\cdots \end{aligned}$$

For ARMA processes, we would need to compute the first few values of $\psi_0, \ldots, \psi_q$, and then obtain the remaining ones via the following recursive relationships.

$$c(k) - \phi_1 c(k-1) - \cdots - \phi_p c(k-p) = \sigma^2 \sum_{k \le j \le q} \theta_j \psi_{j-k}, \quad 0 \le k < \max\{p, q+1\},$$

$$c(k) - \phi_1 c(k-1) - \cdots - \phi_p c(k-p) = 0, \quad k \ge \max\{p, q+1\}.$$

For example, compute the covariance function of

$$x_t - .5x_{t-1} = w_t - .5w_{t-1}.$$

5.4.4 Preliminary estimation

This is based on the causality that x_t can be expressed as a moving average process via the innovation algorithm:

$$x_t = \sum_0^\infty \psi_j w_{t-j}.$$

The coefficients ψ_j are related to the parameters of the process by

$$\psi_0 = 1, \qquad \psi_j = \theta_j + \sum_{i=1}^{\min(j,p)} \phi_i \theta_{j-i}, \quad (j = 1, 2, \ldots, p+q),$$

where $\phi_j = 0$ for $j > p$ and $\theta_j = 0$ for $j > q$. Note that $\psi_1, \ldots, \psi_{p+q}$ can be estimated by the innovation estimates $\hat{\theta}_{m1}, \ldots, \hat{\theta}_{m,p+q}$. Thus $\phi_1, \ldots, \phi_p$ and $\theta_1, \ldots, \theta_q$ can be estimated by solving the above equations.

5.4.5 Conditional likelihood

Suppose $w_t \sim_{\text{iid}} N(0, \sigma^2)$. Then the likelihood function is given by

$$\ell(\phi, \theta, \sigma^2) = \log L(\phi, \theta, \sigma^2) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{t=p+1}^{n} w_t^2(\phi, \theta),$$

where $w_t(\phi, \theta) = 0$ for $t \le p$. According to Box and Jenkins (1970), the likelihood can be evaluated as follows:

$$\begin{aligned}
w_{p+1} &= \phi(B)x_{p+1}, \\
w_{p+2} &= \phi(B)x_{p+2} - \theta_1 w_{p+1}, \\
w_{p+3} &= \phi(B)x_{p+3} - \theta_1 w_{p+2} - \theta_2 w_{p+1}, \\
&\vdots \\
w_n &= \phi(B)x_n - \theta_1 w_{n-1} - \cdots - \theta_q w_{n-q}.
\end{aligned}$$

The MLE of (ϕ, θ) are obtained by minimising $\sum_{t=p+1}^{n} w_t^2(\phi, \theta)$ numerically via the Newton-Gauss procedure. Moreover,

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{t=p+1}^{n} w_t^2(\phi, \theta).$$

To calculate the derivatives, note that

$$\frac{\partial w_t(\phi, \theta)}{\partial \theta_k} = \frac{\partial w_{t-k}(\phi, \theta)}{\partial \theta_k} + \sum_{j=1}^{q} \theta_j \frac{\partial w_{t-j}(\phi, \theta)}{\partial \theta_k}, \qquad k = 1, \ldots, q,$$

and

$$\frac{\partial w_t(\phi, \theta)}{\partial \phi_i} = u_{t-i},$$

where the u's are obtained from

$$w_t = \phi(B) u_t, \quad \theta(B) u_t = x_t, \quad u_t = 0, \quad t \leq p.$$

5.4.6 Exact likelihood

This is based on the observation that if $w_t \sim_{\text{iid}} N(0, \sigma^2)$, then

$$(x_1, \ldots, x_n)' \sim N(\mathbf{0}, \Sigma),$$

where $\Sigma = \Sigma_n$ is the $n \times n$ covariance matrix of $X = (x_1, \ldots, x_n)'$. That is,

$$\ell(X) = \ell(x_1, \ldots, x_n) = -\frac{n}{2} \log(2\pi) - \frac{1}{2} \log(|\Sigma|) - \frac{1}{2} X' \Sigma^{-1} X.$$

This may be written as

$$\ell(x_1, \ldots, x_n) = \ell(x_1, \ldots, x_{n-1}) + \ell(x_n | x_{n-1}, \ldots, x_1).$$

Note that

$$x_n | x_{n-1}, \ldots, x_1 \sim N(\mu_{n|n-1}, \tau_n^2),$$

where $\mu_{n|n-1} = E(x_n | x_{n-1}, \ldots, x_1)$ and $\tau_n = \text{var}(x_n | x_{n-1}, \ldots, x_1)$ are the (conditional) mean and variance of the conditional distribution of x_n given $x_{n-1}, \ldots, x_1$. Thus

$$\ell(x_n | x_{n-1}, \ldots, x_1) = -\frac{1}{2} \log(2\pi\tau_n^2) - \frac{1}{2} \tau_n^{-2} (x_n - \mu_{n|n-1})^2,$$

which may further be interpreted as the error distribution of $x_n - \mu_{n|n-1}$.

Repeat this process, we may write

$$\begin{aligned}\ell(x_1,\ldots,x_n) &= \sum_{j=2}^{n}\ell(x_j|x_{j-1},\ldots,x_1)+\ell(x_1)\\ &= -\frac{1}{2}\sum_{j=1}^{n}\log(2\pi\tau_j^2)-\frac{1}{2}\sum_{1}^{n}\frac{v_j^2}{\tau_j^2},\end{aligned}$$

where $v_j = x_j - E(x_j|x_{j-1},\ldots,x_1)$, $j \geq 2$, are the predicted errors and $v_1 = x_1$, $\tau_1 = \mathrm{var}(x_1)$.

Thus the likelihood function can be evaluated by calculating the conditional mean $\mu_{j|j-1}$ and variance τ_j from the autocovariance function of the ARMA process. (It may be useful to recall the conditional mean and variance of the Gaussian distribution.) Since the autocovariance function has a nice recursive formula, the likelihood can be quickly computed. The MLE of the parameters are defined as

$$(\hat{\phi},\hat{\theta},\hat{\sigma}^2) = \max{}^{-1}\ell(\phi,\theta,\sigma^2),$$

which may be obtained by directly maximising the likelihood function via iterative numerical procedures.

Example. Consider the following AR(1) process:

$$x_t = \phi x_{t-1} + w_t, \quad |\phi| < 1, \quad w_t \sim_{\mathrm{iid}} N(0,\sigma^2).$$

Then $\mu_{j|j-1} = \phi x_{j-1}$ and $\tau_j^2 = \sigma^2$ for $j \geq 2$. Also,

$$x_1 \sim N(0, \sigma^2/(1-\phi^2)).$$

Thus

$$\ell(X) = -\frac{n}{2}\log(2\pi\sigma^2)+\frac{1}{2}\log(1-\phi^2)-\frac{1}{2\sigma^2}(1-\phi^2)x_1^2-\frac{1}{2\sigma^2}\sum_{2}^{n}(x_j-\phi x_{j-1})^2.$$

5.4.7 Least squares estimators (LSE)

These are denoted by $(\hat{\phi},\hat{\theta})$ and they satisfy

$$(\hat{\phi},\hat{\theta}) = \min{}^{-1}\sum_{1}^{n}\frac{(x_j-\mu_{j|j-1})^2}{\tau_j^2}.$$

5.4.8 Asymptotic distribution of the LSE

Under some reasonable conditions, the least squares estimators are approximately normal. Namely,

$$\begin{pmatrix} \hat{\phi} \\ \hat{\theta} \end{pmatrix} \xrightarrow{d} N\left(\begin{pmatrix} \phi \\ \theta \end{pmatrix}, \sigma^2 \begin{pmatrix} E\mathbf{U}\mathbf{U}' & E\mathbf{U}\mathbf{V}' \\ E\mathbf{V}\mathbf{U}' & E\mathbf{V}\mathbf{V}' \end{pmatrix}^{-1} \right),$$

where $\mathbf{U} = (U_p, \ldots, U_1)'$ and $\mathbf{V} = (V_q, \ldots, V_1)'$ are respectively AR and MA processes

$$\phi(B)U_t = w_t, \qquad V_t = \theta(B)w_t.$$

Example. For the following ARMA(1,1) process:

$$x_t - \phi x_{t-1} = w_t + \theta w_{t-1}, \quad |\phi| < 1, \quad |\theta| < 1,$$

the asymptotic variance matrix is given by

$$V(\phi, \theta) = \begin{pmatrix} (1-\phi^2)^{-1} & (1+\phi\theta)^{-1} \\ (1+\phi\theta)^{-1} & (1-\theta^2)^{-1} \end{pmatrix}^{-1}.$$

5.4.9 Computing the estimates

The S-PLUS function `arima.mle` fits ARIMA models via the conditional Gaussian maximum likelihood.

```
> ?arima.mle
> ...
  arima.mle(x, model, n.cond=<<see below>>, xreg=NULL,  ...)


REQUIRED ARGUMENTS

x       a univariate time series or a vector. Missing values (NAs) are allowed.
model   a list specifying an ARIMA model. If a simple ARIMA model is specified
        (i.e., only one component), then model should be a list with component
        names from: order, period, ar, ma, ndiff, ar.opt, ma.trans, and ma.opt.
        See the DETAILS section for further details.

OPTIONAL ARGUMENTS

...
```

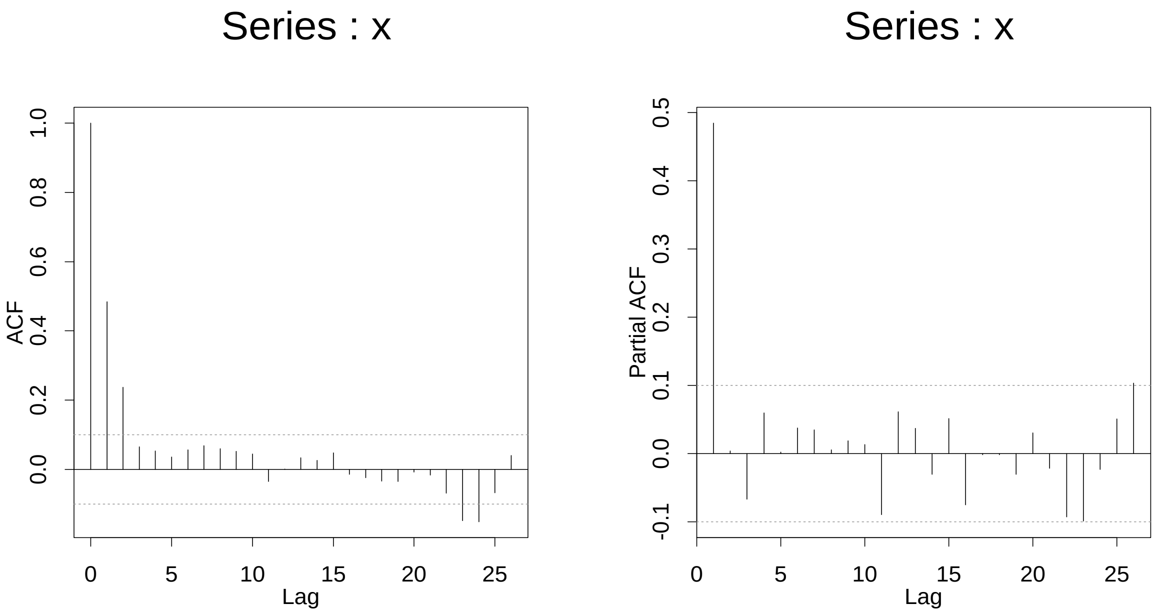


Fig 5.3 *The ACF and the partial ACF of a* `ma2`.

5.4.10 Examples

```
# Simulate an MA(2) series and fit the series using Gaussian Maximum Likelihood

# x <- arima.sim(list(order=c(0,0,2), ma=c(-.5, -.25)),n=400)
> x <- scan("ma2")
> acf(x) # Figure 5.3
> acf(x,type="p") # Figure 5.3
> x.arima <- arima.mle(x,list(order=c(0,0,2)),max.iter=50)
> x.arima
$model:
$model$order:
[1] 0 0 2

$model$ndiff:
[1] 0

$model$ma:
[1] -0.4876061 -0.2527129


$var.coef:
             ma(1)        ma(2)
ma(1) 0.0023403407 0.0009109544
```

```
ma(2) 0.0009109544 0.0023403407

$method:
[1] "Maximum Likelihood"

$series:
[1] "x"

$aic:
[1] 1134.118

$loglik:
[1] 1130.118

$sigma2:
[1] 0.986766

$n.used:
[1] 400

$n.cond:
[1] 0

$converged:
[1] T

$conv.type:
[1] "relative function convergence"

> x.arima$loglik + 2*2 # Check the definition of AIC
[1] 1134.118


# Now try this with initial values
> x.arima <- arima.mle(x,list(order=c(0,0,2), ma=c(-.5, -.25)),max.iter=50)
> x.arima
$model:
$model$ma:
[1] -0.4876064 -0.2527120

$model$order:
[1] 0 0 2
```

```
$var.coef:
             ma(1)        ma(2)
ma(1) 0.0023403419 0.0009109561
ma(2) 0.0009109561 0.0023403419

$aic:
[1] 1134.118

$loglik:
[1] 1130.118

$sigma2:
[1] 0.986766

$n.used:
[1] 400

$n.cond:
[1] 0

$converged:
[1] T

$conv.type:
[1] "relative function convergence"


# Fit a larger MA model
> x.arima <- arima.mle(x, model=list(order=c(0,0,4)))
> x.arima
$model:
$model$order:
[1] 0 0 4

$model$ma:
[1] -0.49426184 -0.28001477 -0.05165939 -0.04621831


$var.coef:
             ma(1)        ma(2)        ma(3)        ma(4)
ma(1) 2.494660e-03 0.0012296855 0.0006676824 7.203861e-05
```

```
ma(2) 1.229686e-03 0.0030987248 0.0015395236 6.676824e-04
ma(3) 6.676824e-04 0.0015395236 0.0030987248 1.229686e-03
ma(4) 7.203861e-05 0.0006676824 0.0012296855 2.494660e-03

$aic:
[1] 1136.814

$loglik:
[1] 1128.814

$sigma2:
[1] 0.9835296

$converged:
[1] T

> sqrt(diag(x.arima$var.coef))
[1] 0.04994657 0.05566619 0.05566619 0.04994657
# Check that MA_3 and MA_4 are not significant.

# Look at the AIC
> arima.mle(x, model=list(order=c(0,0,3)))$aic
[1] 1135.693

> arima.mle(x, model=list(order=c(0,0,1)))$aic
[1] 1159.127
```

5.5 Model identification

Graphical methods

These are based on the ACF, PACF and the power spectra which we have discussed in some detail.

Final Prediction Error (FPE)

Given an AR process, denote the maximum likelihood estimate of σ^2 by

$$\hat{\sigma}^2 = n^{-1} \sum_1^n \frac{(x_j - \mu_{j|j-1})}{\tau_j^2}.$$

Then the final prediction error is defined to be

$$\text{FPE} = \hat{\sigma}^2 \frac{n+p}{n-p}.$$

The order p is chosen to minimize FPE.

Akaike Information Criterion (AIC)

For an ARMA process, the AIC (Akaike, 1969) is defined as

$$\text{AIC} = -2\ell(\hat{\phi}, \hat{\theta}, \hat{\sigma}^2) + 2(p+q).$$

Note that under the invertibility condition, this may be expressed by

$$\text{AIC} = n \log(\hat{\sigma}^2) + 2(p+q).$$

The parameters p and q are then chosen so that AIC is minimum. That is,

$$(\hat{p}, \hat{q}) = \min^{-1} AIC(p,q).$$

Bayesian Information Criterion (BIC)

The BIC for an ARMA process is defined to be

$$\text{BIC} = (n-p-q)\log[n\hat{\sigma}^2/(n-p-q)] + (p+q)\log[(n\hat{c}(0) - n\hat{\sigma}^2)/(p+q)].$$

For an AR model, a modification of this leads to the following criterion:

$$\text{BIC} = n\log(\hat{\sigma}^2) + p\log(n)$$

minimization of which, according to Lütkepohl (1985), often leads to a correct estimate of the order.

AICC

It is known that AIC has the tendency to overfit autoregressive models [see Jones (1975) and Shibata (1976)]. To correct for this bias, Hurvich and Tsai (1989) introduced the following modified AIC:

$$\text{AICC} = -2\ell(\hat{\phi}, \hat{\theta}, \hat{\sigma}^2) + 2(p+q+1)n/(n-p-q-2).$$

Comparisons

It has shown that the BIC leads to a consistent estimator of the order (p, q). That is, for a $ARMA(p, q)$:

$$\hat{p}_{\text{BIC}} \to p \qquad , \hat{q}_{\text{BIC}} \to q \qquad \text{in probability.}$$

This property is not shared by the AIC, FPE and AICC. However, the latter criteria are asymptotically efficient. That is, if $H(p)$ denote the mean squared prediction error when an AR(p) is fitted to the data $X_1, \ldots, X_n$, and if $\hat{p}$ is obtained by AIC (say), then

$$\frac{\min_{1 \le p \le k_n} H(p)}{H(\hat{p})} \to 1 \qquad \text{in probability.}$$

See Shibata (1980) and Hurvich and Tsai (1989).

Criterion autoregressive transfer function (CAT)

Define

$$\tilde{\sigma}_j^2 = \frac{n}{n-j}\hat{\sigma}_j^2, \qquad j = 1, \ldots, p_{\max}.$$

Then

$$CAT(k) = \begin{cases} (n^{-1}\sum_1^k \tilde{\sigma}_j^{-2}) - \tilde{\sigma}_j^{-2}, & \text{if } k = 1, \ldots, p_{\max}; \\ (n+1)\hat{c}(0)/n, & k = 0. \end{cases}$$

See Parzen (1977).

5.6 $ARIMA\ (p, d, q) \times (P, D, Q)_s$

The following model is usually used to fit seasonal data:

$$\Phi(B^s)\phi(B)\nabla_s^D \nabla^d X_t = \Theta(B^s)\theta(B)e_t,$$

where

$$\nabla^d = (1-B)^d, \qquad \nabla_s^D = (1-B^s)^D,$$

$$\phi(z) = 1 - \phi_1 z - \cdots - \phi_p z^p,$$

$$\theta(z) = 1 + \theta_1 z + \cdots + \theta_q z^q$$

$$\Phi(z) = 1 - \Phi_1 z - \cdots - \Phi_P z^P,$$

$$\Theta(z) = 1 + \Theta_1 z + \cdots + \Theta_Q z^Q.$$

In S-PLUS, this model is specified by

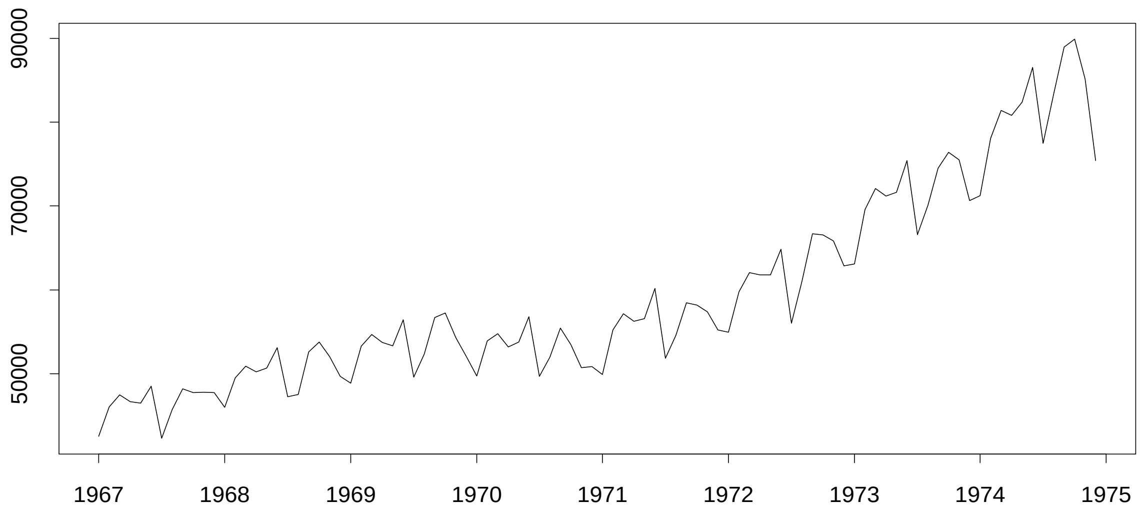


Fig 5.4 *The* `ship` *data.*

```
model=list( list(order=c(p,d,q)), list(order=c(P,D,Q), period=s) )

# Fit a Box-Jenkins (0,1,1)x(0,1,1)12 Airline model to the ship data
# Use zeros as the starting values for the optimizer
> tsplot(ship)  # Figure 5.4

> dship <- diff(ship,12)
> tsplot(dship)
> ddship <- diff(dship)
> tsplot(ddship)         # Figure 5.5

> acf(ddship)
> acf(ddship,type="p")   # Figure 5.6

> spectrum(ddship,span=c(3,5))
> cpgarm(ddship)                    # Figure 5.7

> mod1 <- list(list(order=c(0,1,1)), list(order=c(0,1,1), period=12))
> ship.1 <- arima.mle(ship, model=mod1)

> ship.1
$model:
$model[[1]]:
```

```
$model[[1]]$order:
[1] 0 1 1

$model[[1]]$ndiff:
[1] 1

$model[[1]]$ma:
[1] -0.2795918


$model[[2]]:
$model[[2]]$order:
[1] 0 1 1

$model[[2]]$period:
[1] 12

$model[[2]]$ndiff:
[1] 1

$model[[2]]$ma:
[1] 0.1907319

$var.coef:
              ma(1)       ma(12)
 ma(1) 1.110637e-02 8.292478e-09
ma(12) 8.292478e-09 1.160990e-02

$aic:
[1] 1430.2

$loglik:
[1] 1426.2

$sigma2:
[1] 1687777

$n.used:
[1] 83

$n.cond:
[1] 13
```

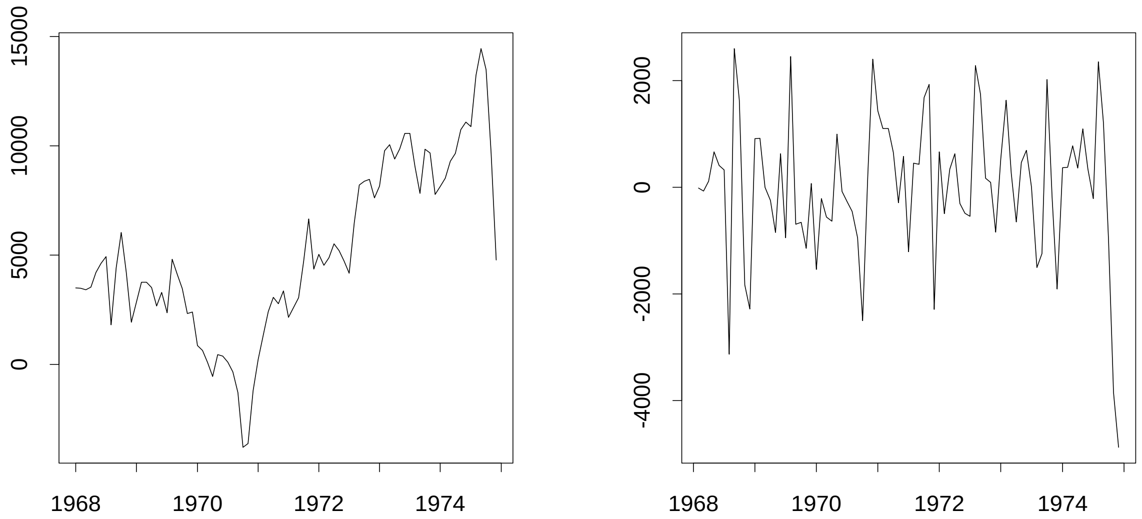


Fig 5.5 *Time series plot of* ∇_{12}`ship` *and* $\nabla\nabla_{12}$`ship`.

```
$converged:
[1] T

$conv.type:
[1] "relative function convergence"

> arima.diag(ship.1)              # Figure 5.7
```

Note. The bottom panel of the `arima.diag()` dipicts the P-value for the Box-Peirce (1970) *portmanteau test* (cf. Section 5.10)

$$Q_K = n \sum_{k=1}^{K} C_k^2$$

applied to the residuals. The maximum K is set by the parameter `gof.lag` (which defaults to 10) plus $p+q$. Note that AIC of the AR(3) is smaller, although a formal test of the difference in deviances, 4.52, using a χ_2^2 distribution is not significant.

The function `arima.diag` can be used to get residuals from the fit by `arima.mle`, by setting `plot=F, acf.resid=F, gof.lag=0` and `resid=T` or `std.resid=T`.

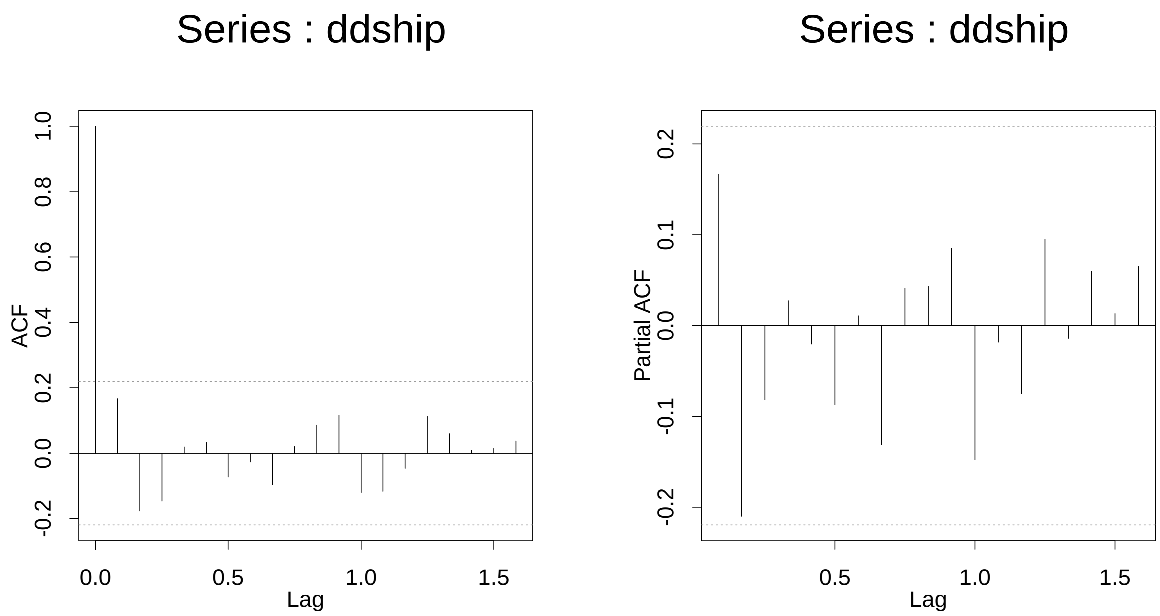


Fig 5.6 *Time analysis of* $\nabla\nabla_{12}$`ship`.

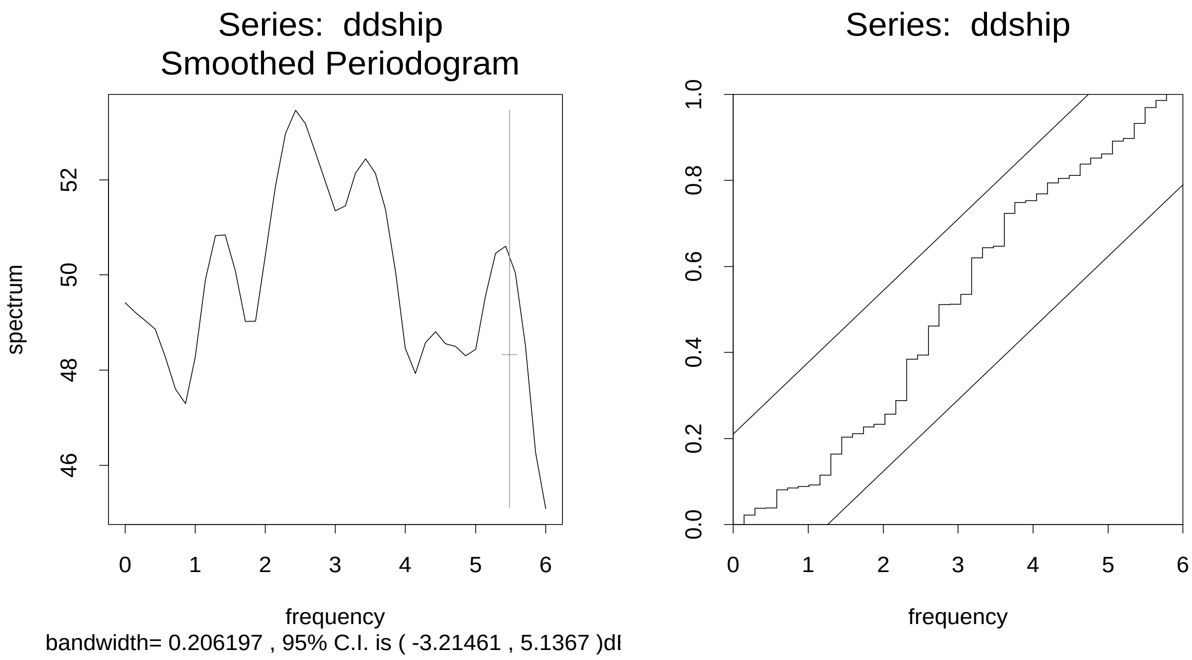


Fig 5.7 *Spectral analysis of* $\nabla\nabla_{12}$`ship`.

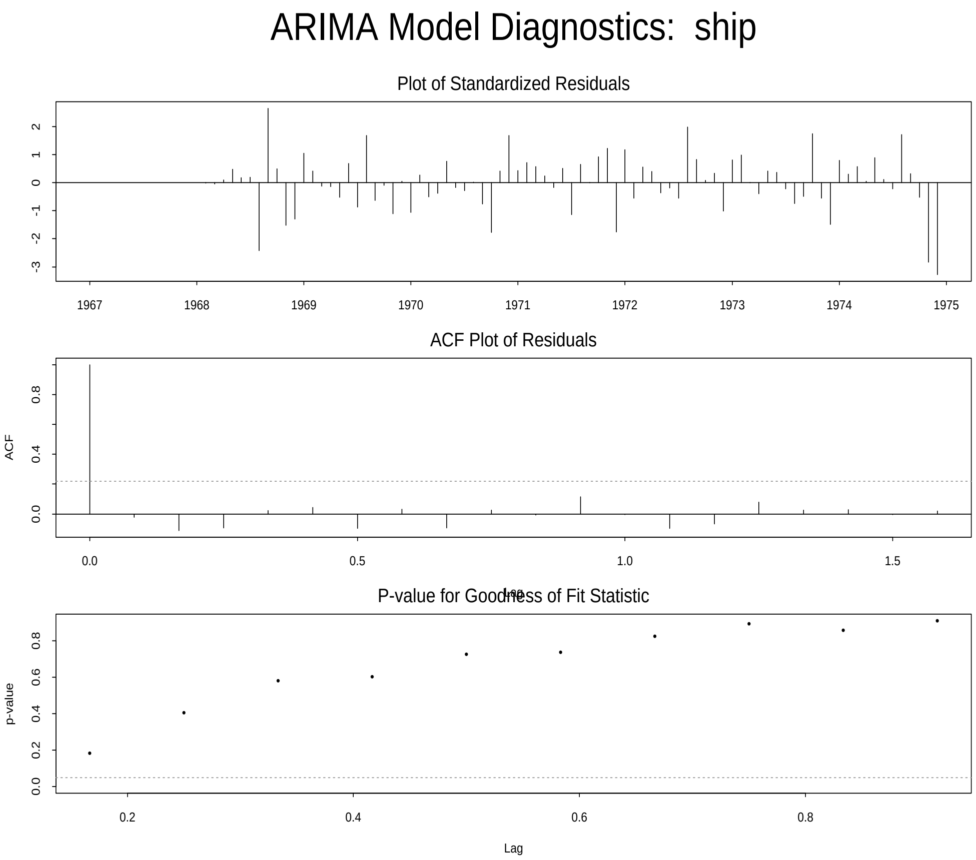


Fig 5.8 *Diagnostics of fit given by ARIMA* $(0, 1, 1) \times (0, 1, 1)_{12}$.

5.7 Remarks

The likelihood is conditioned on n.cond observations. If no missing values are present, the likelihood is computed by forming the Choleski decomposition of the covariance matrix of the process (see Ansley, 1979). If missing values are present, then the likelihood is computed using the Kalman filter applied to a state space representation of the likelihood. The state space representation is the same as that used by Kohn and Ansley (1986). The method of initializing the Kalman filter recursions is that given by Bell and Hillmer (1987). The regression parameters are concentrated out of the likelihood, as in Kohn and Ansley (1985). The autoregressive and moving average parameters are transformed to ensure stationarity and invertibility as in Jones (1980).

5.8 Examples

```
# ----------------------------------------------------
#
# Simulate a AR(1) using ar=c(.7)
# then fit a model via arima.mle.
#
# ----------------------------------------------------

> x <- arima.sim(list(order=c(1,0,0),ar=c(.7)),n=400)
> tsplot(x)
> acf(x,type="p")
> x.ar <- arima.mle(x,model=list(order=c(1,0,0)),max.iter=200)
> x.ar$converged
[1] T
> x.ar$aic
[1] 1109.344
> x.ar$model
$order:
[1] 1 0 0

$ar:
[1] 0.7506356

$ndiff:
[1] 0

> x.ar$var.coef
            ar(1)
ar(1) 0.001094101

# ----------------------------------------------------
#
# Simulate a MA(1) using ma=c(.7)
# then fit a model via arima.mle.
#
# ----------------------------------------------------
> x <- arima.sim(list(order=c(0,0,1),ma=c(.7)),n=400)
> x.ma <- arima.mle(x,model=list(order=c(0,0,1)),max.iter=200)
> x.ma$converged
[1] T
> x.ma$aic
```

```
[1] 1162.043
> x.ma$model
$order:
[1] 0 0 1

$ndiff:
[1] 0

$ma:
[1] 0.7227775

> x.ma <- arima.mle(x,model=list(order=c(0,0,3)),max.iter=200)
> x.ma$converged
[1] T
> x.ma$aic
[1] 1162.089
> x.ma$model
$order:
[1] 0 0 3

$ndiff:
[1] 0

$ma:
[1]  0.74718482 -0.09986601  0.09227798

> x.ma$var.coef
               ma(1)        ma(2)          ma(3)
ma(1)  0.00247871152 -0.001844923  0.00007729313
ma(2) -0.00184492302  0.003849491 -0.00184492302
ma(3)  0.00007729313 -0.001844923  0.00247871152

# Note. The second and third coefficients are not significant.

# ------------------------------------------------------
#
# Simulate a MA(2) using
# theta <- c(1.4, -0.6)
# then estimate theta via arima.mle.
#
# ------------------------------------------------------
> theta <- c(1.4, -0.6)
```

```
> x <- arima.sim(list(order=c(0,0,2),ma=theta),n=400)
> x.ma <- arima.mle(x,model=list(order=c(0,0,2)),max.iter=200)
> acf(x)
> tsplot(x)
> x.ma$converged
[1] T
> x.ma$model
$order:
[1] 0 0 2

$ndiff:
[1] 0

$ma:
[1]  1.445378 -0.641508
> theta
[1]  1.4 -0.6

> x.ma$aic
[1] 1139.233
> x.ma$var.coef
             ma(1)        ma(2)
ma(1)  0.001471169 -0.001295391
ma(2) -0.001295391  0.001471169

# ----------------------------------------------------
#
# Simulate a MA(3) using
# alpha <- c(-0.4364617, -0.2958094, 0.4239308)
# then estimate alpha via arima.mle.
#
# ----------------------------------------------------
> alpha <- c(-0.4364617, -0.2958094, 0.4239308)
> x <- arima.sim(list(order=c(0,0,3),ma=alpha),n=400)
> tsplot(x)
> acf(x)
> x.ma <- arima.mle(x,model=list(order=c(0,0,3)),max.iter=200)
> x.ma$converged
[1] T
> x.ma$model
$order:
[1] 0 0 3
```

```
$ndiff:
[1] 0

$ma:
[1] -0.4475751 -0.2965091  0.4881926
> alpha
[1] -0.4364617 -0.2958094  0.4239308
# ----------------------------------------------------
#
# Simulate a ARMA(1,0,1) using
# ar=c(.5), ma=c(.7)
# then estimate parameter via arima.mle.
#
# ----------------------------------------------------
> x <- arima.sim(list(order=c(1,0,1),ar=c(.5),ma=c(.7)),n=400)
> tsplot(x)
> acf(x)
> acf(x)
> acf(x,type="p")
> x.arima <- arima.mle(x,model=list(order=c(1,0,1)),max.iter=200)
> x.arima$model
$order:
[1] 1 0 1

$ar:
[1] 0.4412684

$ndiff:
[1] 0

$ma:
[1] 0.6255287

> x.arima$converged
[1] T

> x.arima$aic
[1] 1105.994
> x.arima <- arima.mle(x,model=list(order=c(2,0,1)),max.iter=200)
> x.arima$converged
[1] T
```

```
> x.arima$aic
[1] 1106.142
> x.arima <- arima.mle(x,model=list(order=c(1,0,2)),max.iter=200)
> x.ma$converged
[1] T
> x.arima$aic
[1] 1107.962
> x.arima <- arima.mle(x,model=list(order=c(2,0,2)),max.iter=200)
> x.arima$converged
[1] T
> x.arima$aic
[1] 1107.78
> x.arima <- arima.mle(x,model=list(order=c(4,0,2)),max.iter=200)
> x.arima$converged
[1] F
> x.arima$aic
[1] 1100.55
```

5.8.1 Blood pressure data

```
# Blood pressure example

# bp <- matrix(scan("bp"),byrow=T,ncol=2)
> bpd <- bp[,1]
> bps <- bp[,2]
> bp.ar <- ar(bp,order.max=20)
> bp.ar$ar

, , 1
             [,1]        [,2]
[1,]  0.180830121  0.01281316
[2,]  0.206193164  0.07547262
[3,]  0.104753308  0.03944677
[4,]  0.116934128 -0.01264059
[5,] -0.009576633  0.20214292
[6,]  0.052957755 -0.05286565
[7,]  0.056119278 -0.11910456
[8,]  0.039997157 -0.07353139
[9,]  0.029086722 -0.14943744

, , 2
             [,1]        [,2]
```

```
[1,]  0.067128830 0.225042820
[2,]  0.007398657 0.101212256
[3,] -0.021054149 0.006975340
[4,] -0.029704213 0.128258005
[5,]  0.017009316 0.009406844
[6,]  0.004282581 0.034202822
[7,] -0.092260480 0.029752219
[8,] -0.049331952 0.033908091
[9,]  0.031743463 0.172150806
> bp.ar$aic
 [1] 244.483398  96.099609  42.797852  33.146484  14.995117
 [6]  16.110352  16.270508   4.820312   5.945312   0.000000
[11]   1.416992   3.359375   7.512695   6.806641   7.285156
[16]   8.752930  14.444336  22.272461  28.332031  34.538086
[21]  40.589844

> bpd.ar <- ar(bpd,order.max=20)
> bpd.ar$ar
, , 1
          [,1]
[1,] 0.2418572
[2,] 0.2130693
[3,] 0.1087744
[4,] 0.1337108
> bpd.ar$aic
 [1] 142.231201  52.142578  13.894531   6.298584   0.000000
 [6]   1.712402   1.642578   3.614258   5.463379   6.558350
[11]   5.728027   3.974854   2.951660   4.941406   6.052734
[16]   6.564941   7.584961   9.578369   9.968750  11.875488
[21]  13.079590

> bps.ar <- ar(bps,order.max=20)
> bps.ar$ar
, , 1
           [,1]
[1,] 0.24859726
[2,] 0.13664407
[3,] 0.03276047
[4,] 0.13752967
[5,] 0.09207170
> bps.ar$aic
 [1] 94.4624023 29.7319336 13.9204102 12.1176758  1.9160156
```

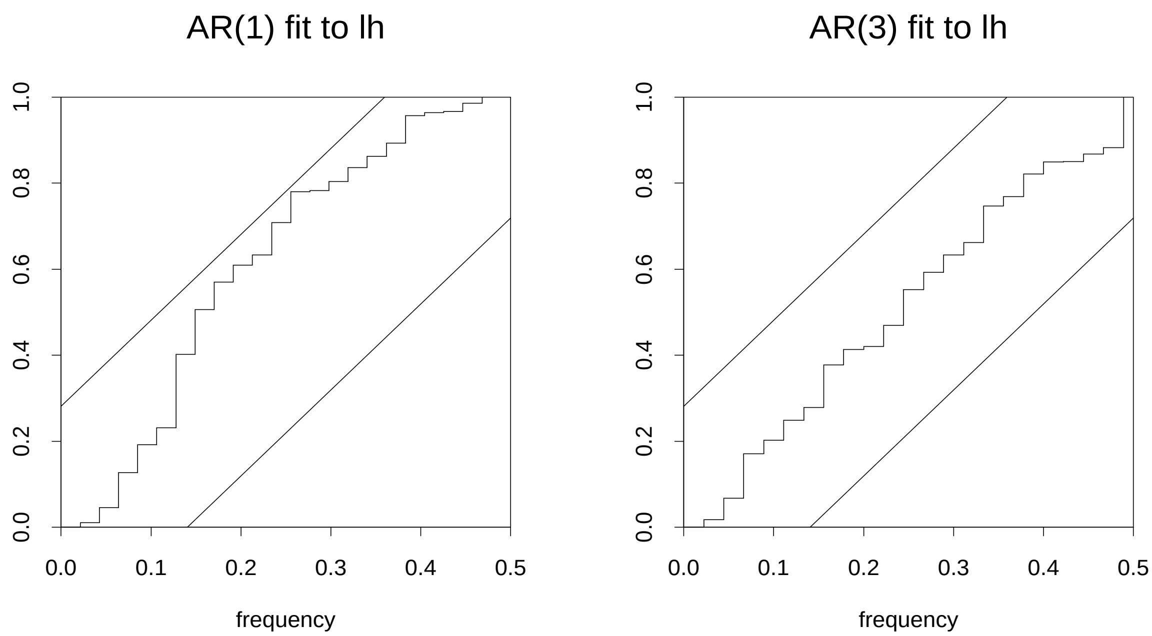


Fig 5.9 *The spectral analysis of the residuals based on various AR fits to* `lh[,3]` *data.*

```
 [6]  0.0000000  1.3554688  3.1328125  4.5346680  0.8105469
[11]  1.0209961  1.9980469  3.4750977  4.0234375  3.6210938
[16]  4.5239258  6.0234375  7.9477539  9.5781250 11.3369141
[21] 13.0551758
```

5.8.2 Lutenizing hormone data

```
# Fit AR models to lh
> lh1 <- lh[,3]
> lh.ar1 <- ar(lh1,F,1)  # Fit AR(1) to lh
> lh.ar1$ar # alpha(1)
, , 1
          [,1]
[1,] 0.5755244
> lh.ar1$var.pred # sigma^2
          [,1]
[1,] 0.2079007

> cpgarm(lh.ar1$resid, main="AR(1) fit to lh") # Figure 5.9

> lh.ar <- ar(lh1,order.max=9)
> lh.ar$order
[1] 3
```

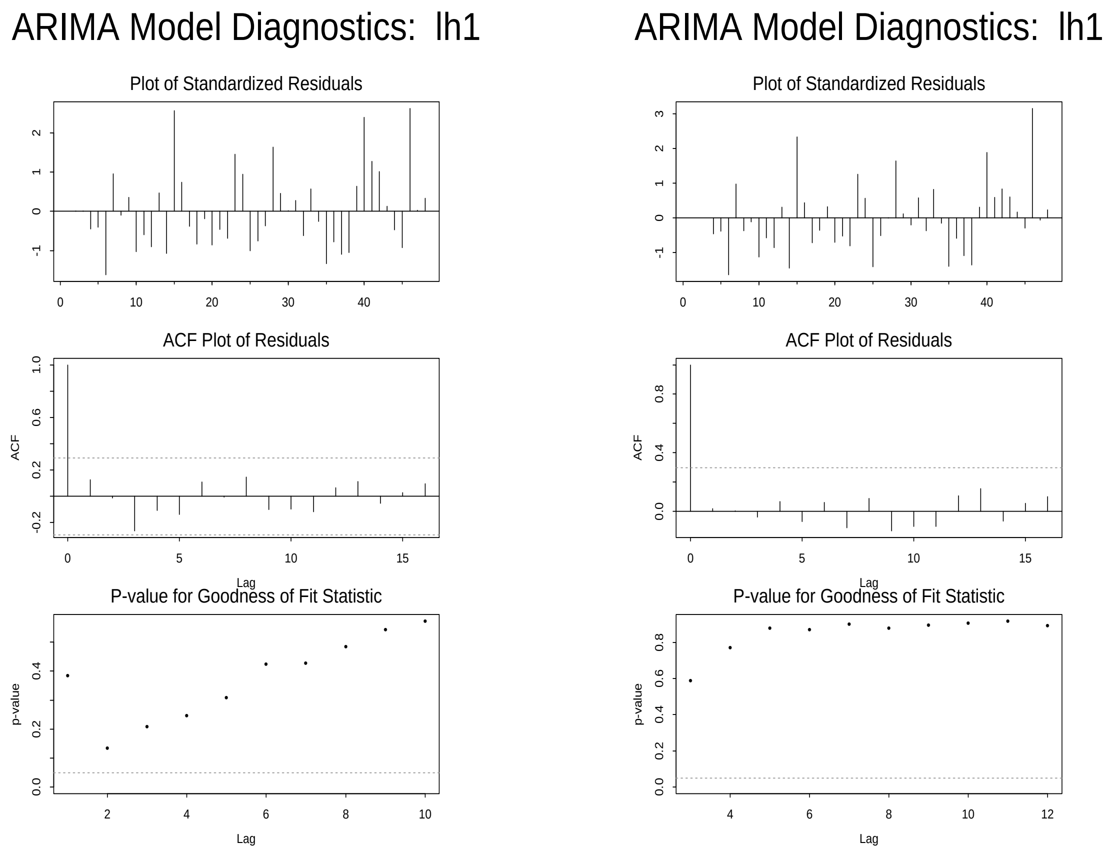


Fig 5.10 *The diagnostics of AR(1) and AR(3) fits to* `lh[,3]` *data.*

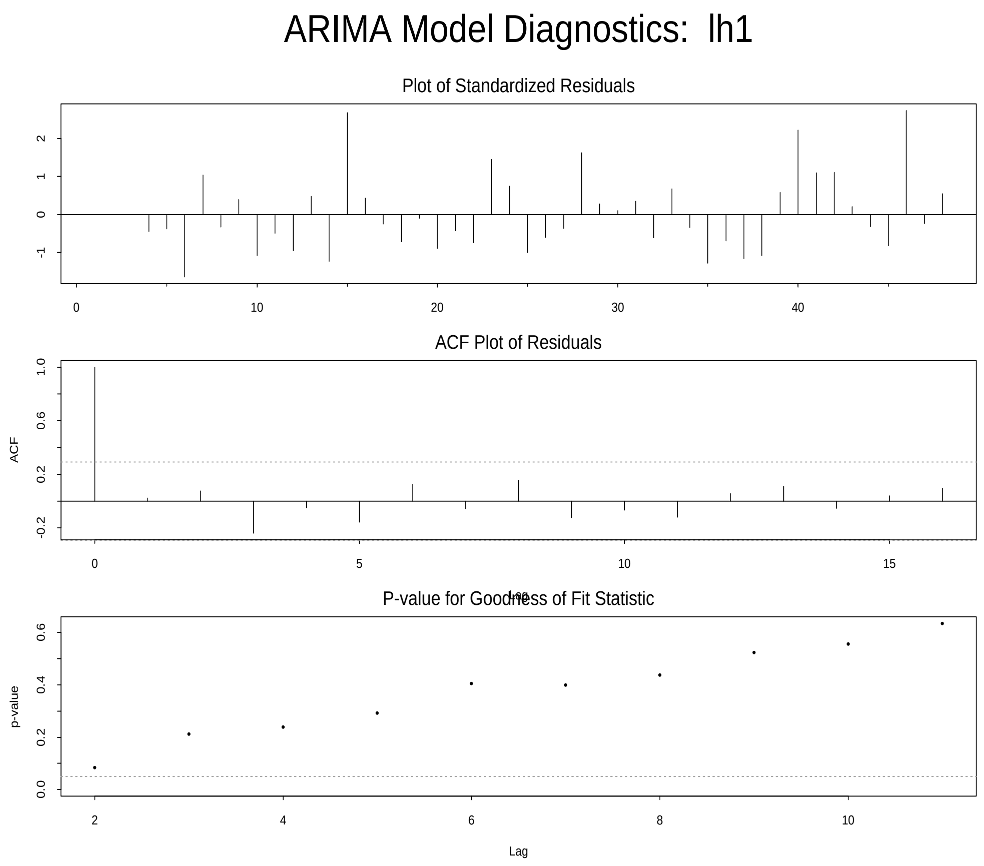


Fig 5.11 *The diagnostics of the ARMA (1,1) fit to* `lh[,3]` *data.*

```
> lh.ar$aic
 [1] 18.3066635  0.9956589  0.5380249  0.0000000  1.4903641
 [6]  3.2127914  4.9932175  6.4694901  8.4625702  8.7411880

> lh.ar$ar
, , 1
            [,1]
[1,]  0.65340161
[2,] -0.06362088
[3,] -0.22694018
> lh.ar$var.pred
          [,1]
[1,] 0.1958671

> cpgarm(lh.ar$resid,main="AR(3) fit to lh") # Figure 5.9

> lh1 <- lh1 - mean(lh1)
> lh.arima1 <- arima.mle(lh1,model=list(order=c(1,0,0)),n.cond=3)
> lh.arima1$model
$order:
[1] 1 0 0

$ar:
[1] 0.5857651

$ndiff:
[1] 0

> lh.arima1$var.coef
           ar(1)
ar(1) 0.01459732

> lh.arima1$loglik
[1] 57.61393
> lh.arima1$aic
[1] 59.61393
> lh.arima1$sigma2
[1] 0.2106478
> arima.diag(lh.arima1) # Figure 5.10

> lh.arima3 <- arima.mle(lh1,model=list(order=c(3,0,0)),n.cond=3)
```

```
> lh.arima3$loglik
[1] 53.08904
> lh.arima3$aic
[1] 59.08904
> lh.arima3$sigma2
[1] 0.1904967
> arima.diag(lh.arima3) # Figure 5.10
> lh.arima1$loglik - lh.arima3$loglik
[1] 4.524888

> lh.arima11 <- arima.mle(lh1,model=list(order=c(1,0,1)),n.cond=3)
> lh.arima11$loglik
[1] 56.41655
> lh.arima11$aic
[1] 60.41655
> lh.arima11$sigma2
[1] 0.2051164
> arima.diag(lh.arima11) # Figure 5.11
```

5.8.3 Mortality data

You can load the functions described in *Modern Applied Statistics with S-PLUS* (MASS) into your local session by using

```
> library(MASS) # Load MASS
```

```
> library(MASS)
> deaths.stl <- stl(deaths, "periodic")
> ts.plot(deaths, deaths.stl$sea, deaths.stl$rem) # Figure 5.12
> dsd <- deaths.stl$rem
> ts.plot(dsd)
> acf(dsd)  # Figure 5.13
> acf(dsd, type="partial") # Figure 5.13
> spectrum(dsd, span=c(3,3)) # Figure 5.13
> cpgram(dsd) # Figure 5.14
> dsd.ar <- ar(dsd)
> dsd.ar$order
[1] 1
> dsd.ar$aic
 [1]  3.6483154  0.0000000  1.2263184  0.4084473  1.7557373
 [6]  3.4693604  4.4677734  6.2944336  8.0922852  9.1273193
[11]  4.9443359  6.5844727  5.2198486  5.8538818  7.3585205
```

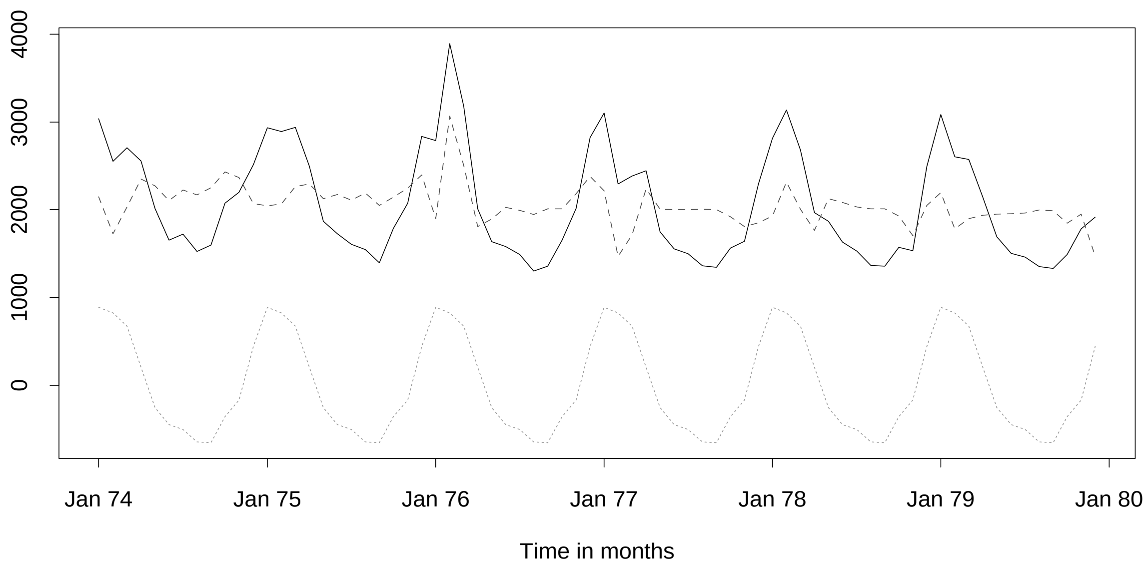


Fig 5.12 `stl` *decomposition for the* `deaths` *data (solid line). The dotted series is the seasonal component, the dashed series the remainder.*

```
[16]  9.2658691 10.5830078 12.5444336 14.5443115
> dsd.ar$ar

, , 1
          [,1]
[1,] 0.2746854
> cpgram(dsd.ar$resid, main="AR(1) residuals") # Figure 5.14
> dsd.rar <- ar.gm(dsd)
> dsd.rar$ar
[1] 0.4149288


# ARIMA model fitting via deaths

> acf(deaths.diff, 30) # Figure 5.15
> acf(deaths.diff, 30, type="partial") # Figure 5.15
> ar(deaths.diff)
$order:
[1] 12

$ar:

, , 1
```

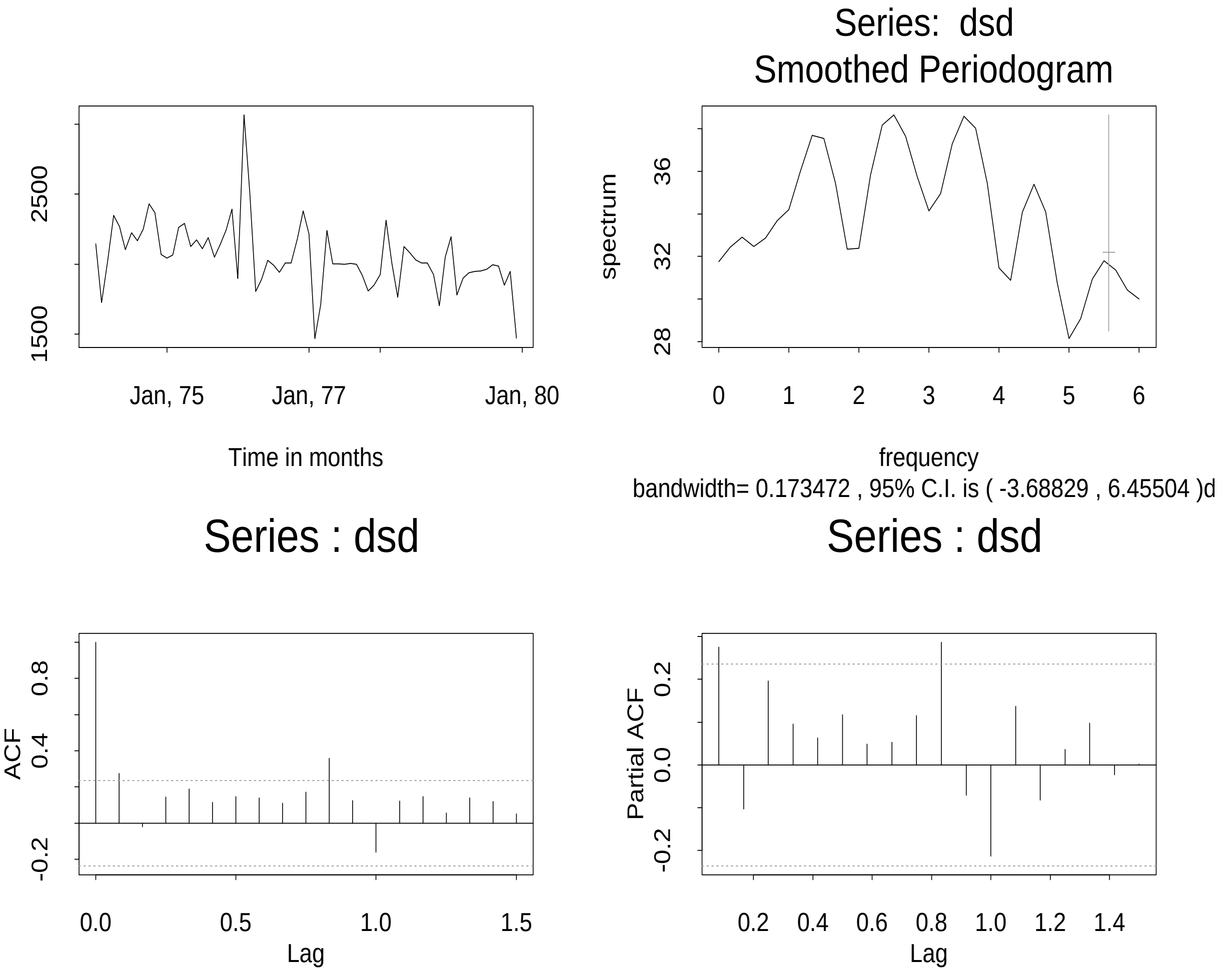


Fig 5.13 *Residual analysis after using* `stl`.

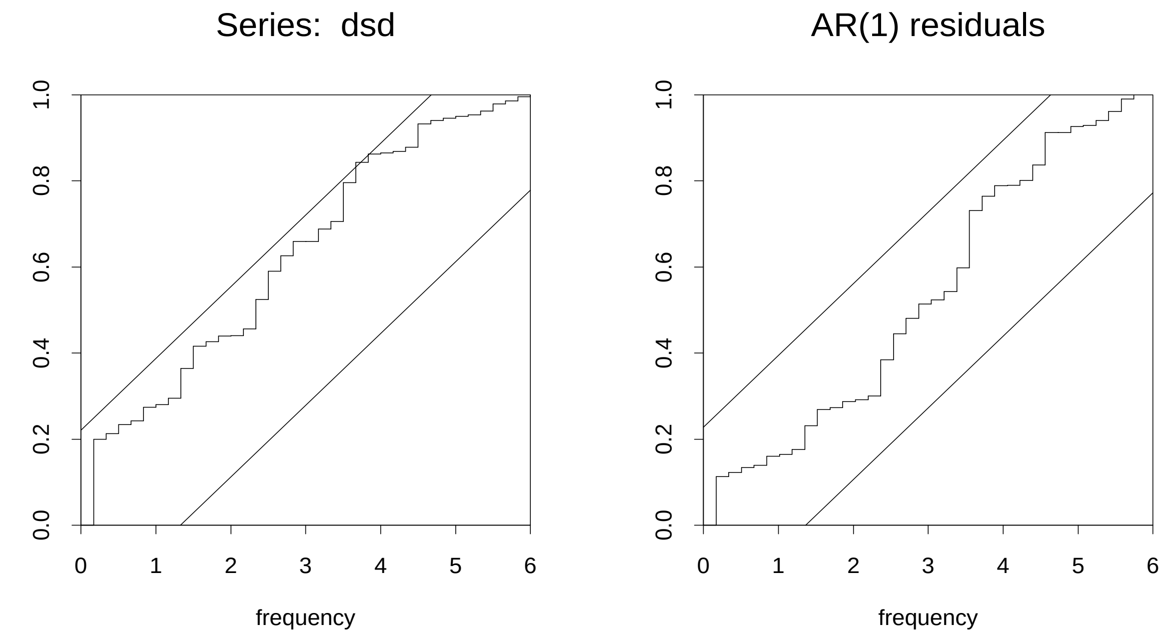


Fig 5.14 *Diagnostic for residual analysis after using* `stl`*.*

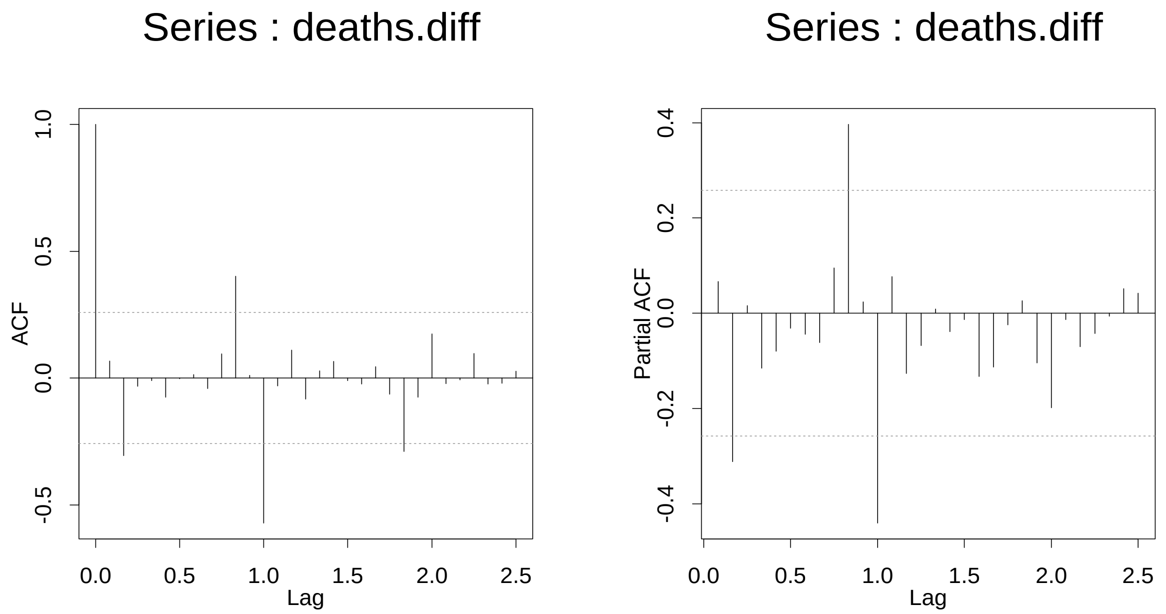


Fig 5.15 *ACF and PACF plots for the seasonally differenced* `deaths` *series.*

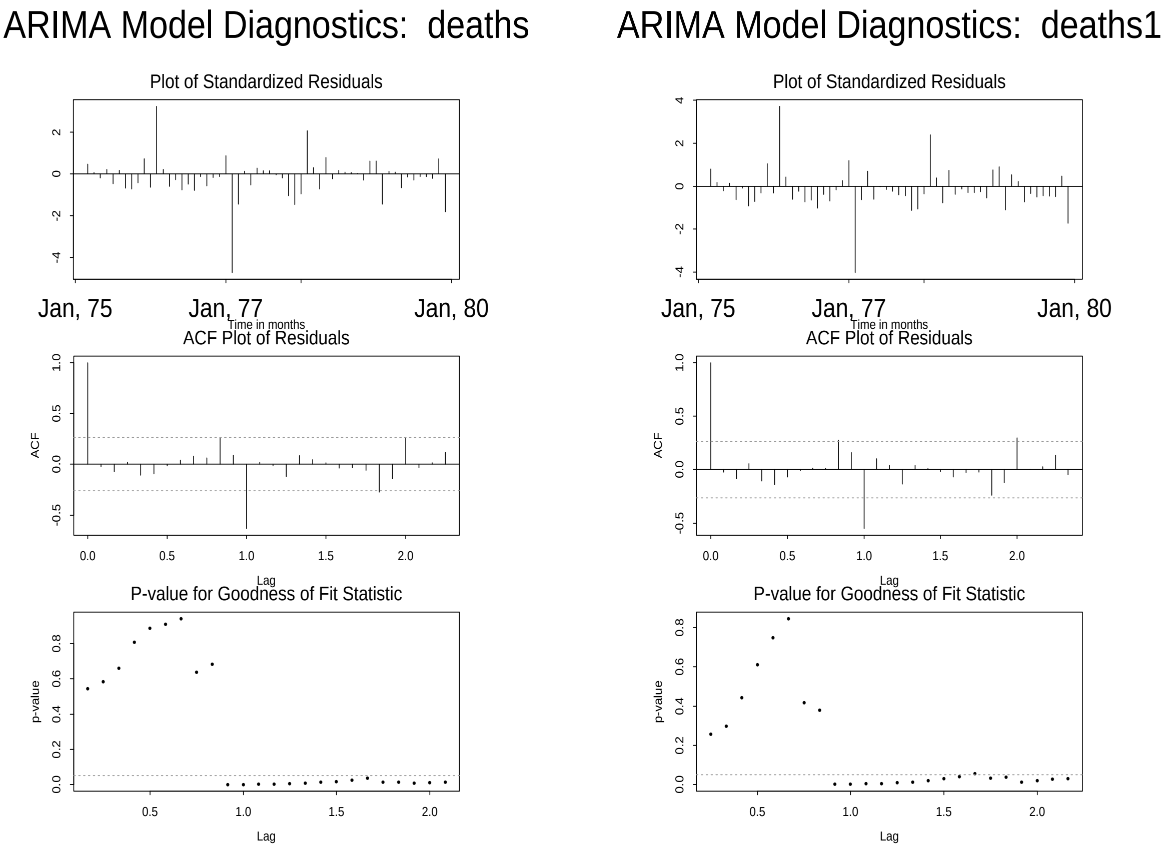


Fig 5.16 *Diagnostics of* `deaths.arima1` *and* `deaths.arima2`.

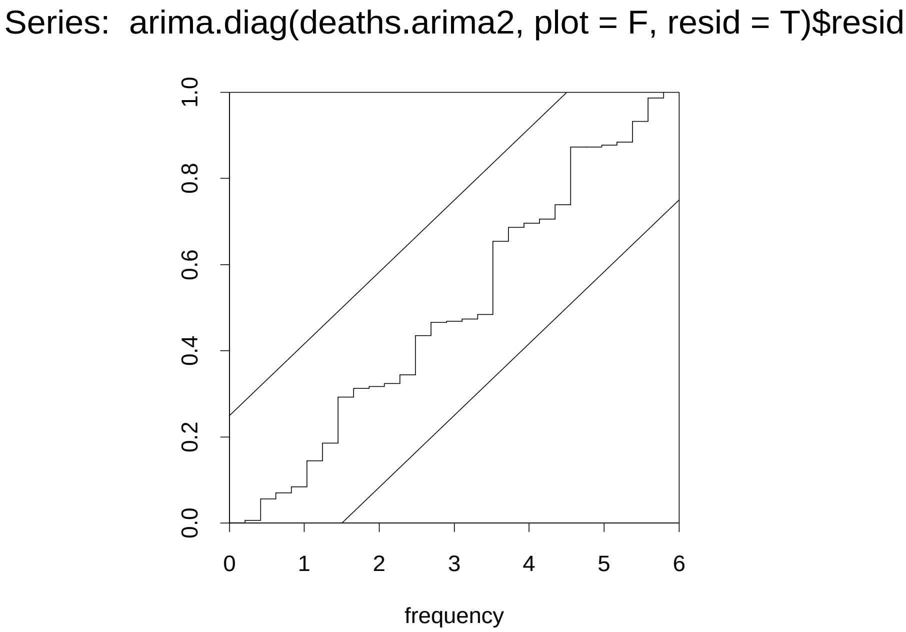


Fig 5.17 `cpgram` *of the residuals from* `deaths.arima2`*.*

```
              [,1]
 [1,]  0.0476588160
 [2,] -0.1520935595
 [3,]  0.0234203804
 [4,] -0.0671844482
 [5,] -0.0492192060
 [6,] -0.0007684373
 [7,] -0.0219185911
 [8,]  0.0274347812
 [9,]  0.0661595315
[10,]  0.2515296340
[11,]  0.0404291004
[12,] -0.4406891465

$var.pred:
        [,1]
[1,] 96728.2

$aic:
 [1]  7.814331  9.547180  5.408264  7.392944  8.583984 10.197998
 [7] 12.138855 14.020203 15.792664 17.250488  8.990601 10.955750
[13]  0.000000  1.647278  2.684631  4.409790  6.404724  8.315247
```

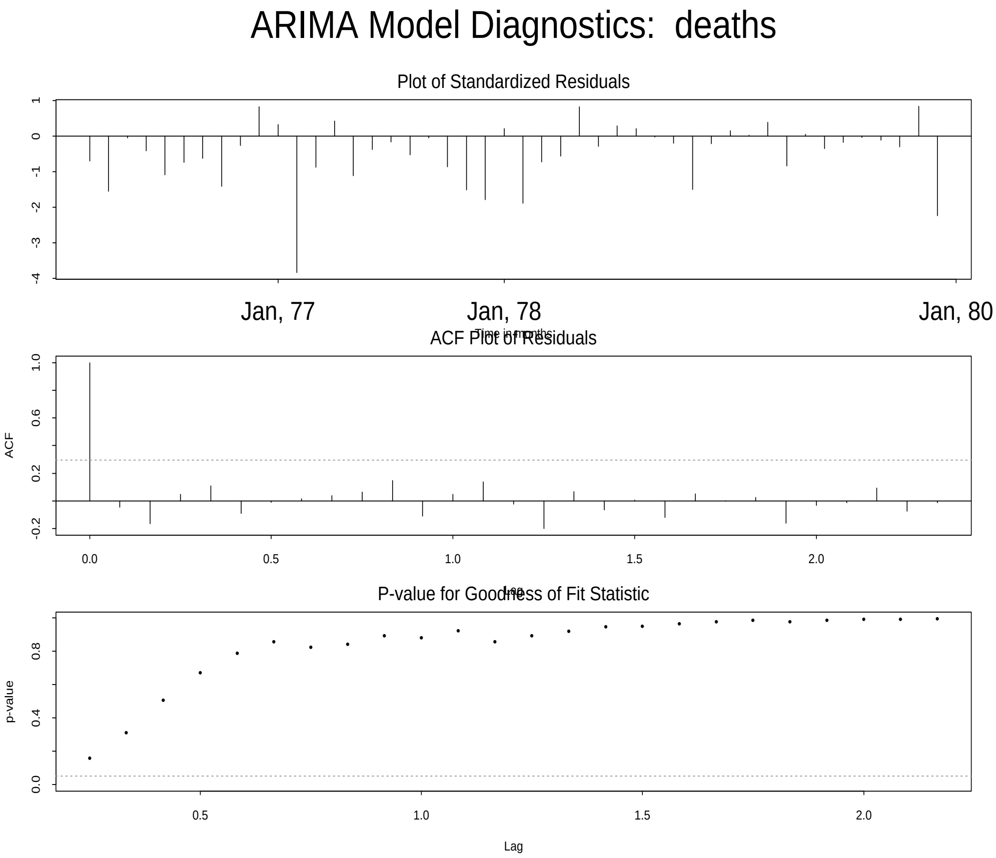


Fig 5.18 *Diagnostics of* `deaths.arima3`.

```
...

# this suggests the seasonal effect is still present.

> deaths.arima1 <- arima.mle(deaths, model=list(
+      list(order=c(2,0,0)), list(order=c(0,1,0), period=12)) )
> deaths.arima1$aic
[1] 847.4084
> deaths.arima1$model[[1]]$ar  # the non-seasonal part
[1]  0.1230075 -0.3052218
> sqrt(diag(deaths.arima1$var.coef))
[1] 0.1250407 0.1250407
> arima.diag(deaths.arima1, gof.lag=24) # Figure 5.?
# need a seasonal AR term

> deaths1 <- deaths - mean(deaths)
> deaths.arima2 <- arima.mle(deaths1, model=list(
+      list(order=c(2,0,0)),  list(order=c(1,0,0), period=12)) )
> deaths.arima2$aic
[1] 845.3751
> deaths.arima2$model[[1]]$ar # non-seasonal part
[1]  0.2160105 -0.2535556
> deaths.arima2$model[[2]]$ar # seasonal part
[1] 0.829428
> sqrt(diag(deaths.arima2$var.coef))
[1] 0.12701544 0.12701544 0.07334958
> arima.diag(deaths.arima2, gof.lag=24) # Figure 5.?
> cpgram(arima.diag(deaths.arima2, plot=F, resid=T)$resid)

> deaths.arima3 <- arima.mle(deaths, model=list(
+      list(order=c(2,0,0)), list(order=c(1,1,0), period=12)) )
> deaths.arima3$aic  # not comparable to those above
[1] 638.207
> deaths.arima3$model[[1]]$ar
[1]  0.4121201 -0.2693848
> deaths.arima3$model[[2]]$ar
[1] -0.7269017
> sqrt(diag(deaths.arima3$var.coef))
[1] 0.1419914 0.1419914 0.1012545
> arima.diag(deaths.arima3, gof.lag=24) # Figure 5.?

# Comparing AIC
```

```
> arima.mle(deaths1, model=list(list(order=c(2,0,0)),
+      list(order=c(1,0,0), period=12)), n.cond=26 )$aic
[1] 664.1366

> deaths.arima4 <- arima.mle(deaths1, model=list(
+      list(order=c(2,0,0)), list(order=c(2,0,0), period=12)) )
> deaths.arima4$aic
[1] 634.0667
> deaths.arima4$model[[1]]$ar
[1]  0.4782125 -0.2787332
> deaths.arima4$model[[2]]$ar
[1] 0.1734640 0.6347404
> sqrt(diag(deaths.arima4$var.coef))
[1] 0.1415986 0.1415986 0.1139322 0.1139322
> arima.diag(deaths.arima4, gof.lag=24)
```

The diagnostics of the first AR fit (`deaths.arima1`) indicates that seasonal effects are still present. Dropping the differencing in favour of a seasonal AR model gives an AIC = 845.38 which favours the second fit. However, the diagnostics based on `deaths.arima2` suggests that there is still seasonal effect in the residuals, so we include differencing and a seasonal AR term, yielding an ARIMA $(2,0,0) \times (1,1,0)_{12}$ model

$$(1 + 0.727B^{12})(1 - B^{12})Y = (1 - 0.273B^{12} - 0,727B^{24})Y = \epsilon,$$

where $Y_t = X_t - 0.412X_{t-1} + 0.270X_{t-2}$. Note that Y corresponds to the non-seasonal part of the series. The AIC of the current fit is 634.0667, which is smaller than the AIC (= 664.14) given by the previous fit based on the ARIMA $(2,0,0) \times (1,0,0)_{12}$ model. Since the current fit is an ARIMA $(2,0,0) \times (2,0,0)_{12}$ model, we fit the data again using this model via `deaths.arima4`. The resulting AIC is 634.07.

5.8.4 Monthly accidental death data

The data is named `accdeaths`. The plot `tsplot(accdeaths)` indicates there is a seasonal effect. Deseas the series using `diff`, yielding `dacc`. Plotting this suggests that there is a linear trend. Simple use of `diff` to detrend it, resuling a series called `ddacc`. The correlogram plot of `ddacc` suggests an ARIMA $(0,0,1) \times (0,0,12)$ model. We fit this model to the demeaned series (`mean=28.83`), yielding

$$\nabla\nabla_{12}X = 28.83 + (1 - 0.488B)(1 - 0.585B^{12})\epsilon,$$

which is a MA(13) model with many zero parameters. To show the flexibility of S-PLUS, we can fit a constrained MA(13) model by forcing some of the parameters to be zero. Fit number 2 does it, and is stored in `ddacc.2`. Note the use of

```
model=list(order=c(0,0,13),
ma.opt=c(T,F,F,F,F,T,F,F,F,F,F,T,T))
```

in `arima.mle()`. The fitted model is

$$\nabla\nabla_{12}X = 28.83 + (1 - 0.608B - 0.411B^6 - 0.677B^{12} + 0.473B^{13})\epsilon.$$

Note that the AIC (from `ddacc.2$aic`) is calculated with 13 parameters and is apparently wrong. The corrected AIC should be 851.8527. Also, standard errors are returned for zero-valued parameters, suggesting SE are wrong too! However, the flexibility of S-PLUS allows us to calculate the constrained covariance matrix.

```
# The monthly accidental deaths in the USA 1973-78,
# from Brockwell and Davis (1991).

> par(mfrow=c(2,3))
> dacc <- diff(accdeaths, 12) # remove the seasonal effect
> ts.plot(dacc)
> acf(dacc, 30)
> acf(dacc, 30, "partial") # Fig 5.19
> ddacc <- diff(dacc)
> ts.plot(ddacc)
> acf(ddacc, 30)
> acf(ddacc, 30, "partial") # Fig 5.19
> mean(ddacc)
[1] 28.83051
> ddacc.1 <- arima.mle(ddacc-mean(ddacc),
+      model=list(list(order=c(0,0,1)),
+      list(order=c(0,0,1), period=12)))
> ddacc.1
$model:
$model[[1]]:
$model[[1]]$order:
[1] 0 0 1

$model[[1]]$ndiff:
[1] 0

$model[[1]]$ma:
[1] 0.4883391


$model[[2]]:
```

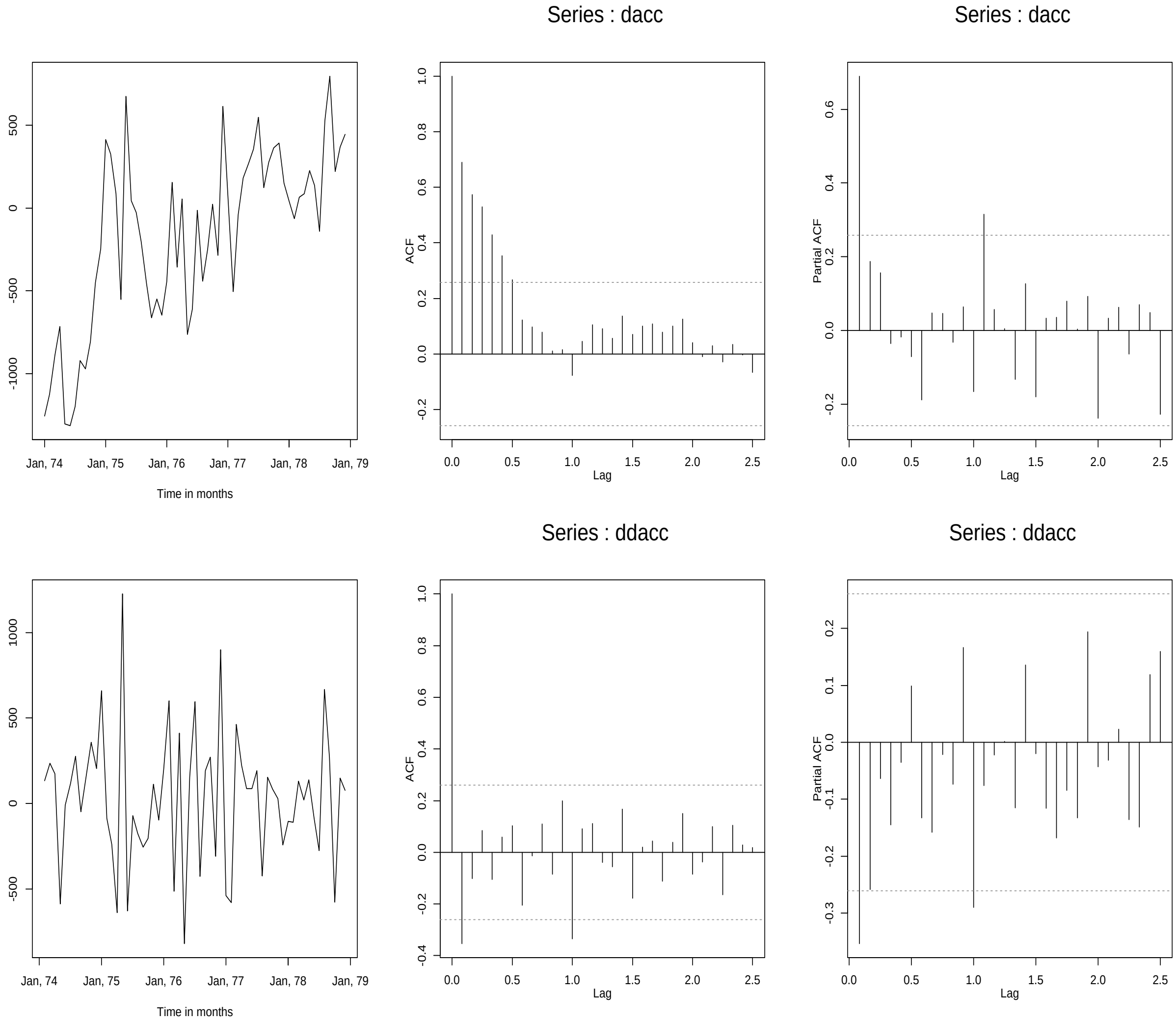


Fig 5.19 *Seasonally differenced (top) and then differenced (bottom) series of* `accdeaths` *with ACF and PACF plots.*

```
$model[[2]]$order:
[1] 0 0 1

$model[[2]]$period:
[1] 12

$model[[2]]$ndiff:
[1] 0

$model[[2]]$ma:
[1] 0.5853442

$var.coef:
                ma(1)         ma(12)
 ma(1)  1.290720e-002 -4.474952e-006
ma(12) -4.474952e-006  1.114190e-002

$method:
[1] "Maximum Likelihood"

$series:
[1] "ddacc - mean(ddacc)"

$aic:
[1] 852.7246

$loglik:
[1] 848.7246

$sigma2:
[1] 94629.48

$n.used:
[1] 59

$n.cond:
[1] 0

$converged:
[1] T

$conv.type:
```

```
[1] "relative function convergence"

> sqrt(diag(ddacc.1$var.coef))
[1] 0.1136099 0.1055552

> sqrt(diag(ddacc.1$var.coef))
[1] 0.1136099 0.1055552

# Fit number 2

> ddacc.2 <- arima.mle(ddacc-mean(ddacc),
+       model=list(order=c(0,0,13),
+       ma.opt=c(T,F,F,F,F,T,F,F,F,F,F,T,T)),
+       max.iter=50, max.fcal=100)
> ddacc.2
$model:
$model$ma.opt:
 [1] T F F F F T F F F F F T T

$model$order:
[1]  0  0 13

$model$ma.opt:
 [1] T F F F F T F F F F F T T

$model$ndiff:
[1] 0

$model$ma:
 [1]  0.6078374  0.0000000  0.0000000  0.0000000  0.0000000
 [6]  0.4111893  0.0000000  0.0000000  0.0000000  0.0000000
[11]  0.0000000  0.6769307 -0.4726044

$aic:
[1] 869.8527

$loglik:
[1] 843.8527

$sigma2:
[1] 70539.74
```

```
# The above AIC is wrong!
> ddacc.2$loglik+2*4 # Compute the AIC for the constrained model
[1] 851.8527

> sqrt(diag(ddacc.2$var.coef))
 [1] 0.1147321 0.1079764 0.1079764 0.1079764 0.1079764 0.1079764
 [7] 0.1205180 0.1079764 0.1079764 0.1079764 0.1079764 0.1079764
[13] 0.1147321

# Standard errors are returned for zero-valued parameters,
# suggesting SE are wrong!

> dd.VI <- solve(ddacc.2$var.coef)
> sqrt(diag(
+      solve(dd.VI[ddacc.2$model$ma.opt,ddacc.2$model$ma.opt])
+      ))
[1] 0.09668969 0.08577824 0.09478055 0.09596390

# Another fit

> ddacc.3 <- arima.mle(ddacc-mean(ddacc),
+       model=list(order=c(0,0,13),
+       ma.opt=c(T,F,F,F,F,F,F,F,F,F,F,T,T)),
+       max.iter=50, max.fcal=100)
> ddacc.3$model$ma:
 [1]  0.4634952  0.0000000  0.0000000  0.0000000  0.0000000
 [6]  0.0000000  0.0000000  0.0000000  0.0000000  0.0000000
[11]  0.0000000  0.6296492 -0.2174101
> ddacc.3$aic
[1] 874.4742
> ddacc.3$loglik
[1] 848.4742
> ddacc.3$loglik + 2*3 # corrected AIC
[1] 854.4742 # larger than the 2nd fit
> dd.VI <- solve(ddacc.3$var.coef)
> sqrt(diag(
+      solve(dd.VI[ddacc.3$model$ma.opt,ddacc.3$model$ma.opt])
+      ))
[1] 0.1143939 0.1000374 0.1270747
```

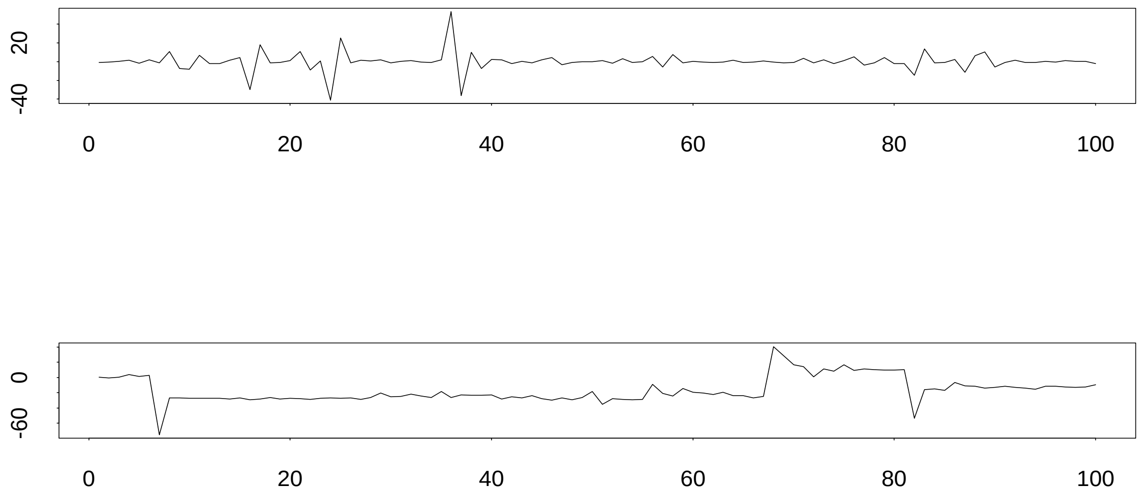


Fig 5.20 *Simulated Cauchy series.*

5.8.5 More on simulation

In simulating ARIMA series, the default distribution is Gaussian. This can be modified as follows

```
wdist <- function(n) rcauchy(n)

x.wild <- arima.sim(100,model=list(ndiff=0,ma=.6),
            n.start=100, rand.gen=wdist)
tsplot(x.wild) # Fig 5.20

x.wild <- arima.sim(100,model=list(ndiff=1,ma=.6),
            n.start=100, rand.gen=wdist)
tsplot(x.wild) # Fig 5.20
```

Simulating stationary ARMA process

To simulate stationary ARMA series, you need a set of parameters ϕ or θ that are roots of the characteristic polynomials lie outside the unit circle. The following are parameters for generating such stationary AR and MA processes. They are also available in `/distrib/bios259/phi`. (Note. When you used these coefficients in S-PLUS, make sure to multiply by -1 to each row, otherwise you may get nonstationary AR series.)

```
 -0.0258833  0.55038285
```

```
0.40824 0.9097945
 0.0650565 -0.82507026
0.8255498 0.74502
 0.5826802 -0.39998
-0.03420475 -0.6378452
-0.5858838  0.0254229
 0.286268 -0.260578
-0.2489758  0.4875937
 0.2403249 -0.624766

-0.4925788  0.48252952 -0.04722543
-0.556842  0.8653622 -0.3342438
 0.037252 -0.64470333  0.29490352
-0.535747 -0.6458060  0.7806200
 .2542 -0.4805222 -0.6964065
0.3780024 0.735689 0.497827
0.7937500 0.766980 0.853042
 0.2836493 -0.520933  0.03000862
 .092493 -0.05897534 -0.5808883
-0.8534272  0.305837  0.5068055

 .0049238 -0.536203 -.285347 -0.373043
 .34749  0.69750 -0.64564 -0.3628274
 0.245740 -0.8055574  0.4852738  0.824059
 0.24056880 -0.3057995 -0.27768603 -0.05667499
 0.379206  0.356342  0.2004730 -0.2676037
 0.470400 -0.33684322  0.08344959  0.240435
-0.936648  0.80799 -0.380722 -0.3262988
 0.345449 -0.0537638  0.785982  0.8339500
-0.3763560 -0.95872676  0.39426330 -0.037394
 0.344236 -0.093407  0.259947  0.8950052

-0.4656022 -.493964  .358049  0.565832 -0.924747
 0.7650478 -0.405492 -.37399 -0.56885  0.348423
-0.2062545  0.24268994 -0.56366372  0.06389096  0.30277497
 0.42467800 -0.6207593 -0.6462444  0.0674026 -0.09072670
 0.27085474  .2530200  0.0993764  0.7357483 -0.03954048
 0.07356268 -0.07069807 -0.429863 -0.63724232  0.5082549
 0.56678289 -0.066470  0.33477348 -0.4902828 -0.63447398
-.627524  0.503763  .777533 -.3320750  0.4439326
 .2444654  0.707486  .059205  0.9607 -0.476303
 0.246576 -0.640930 -0.3252  0.084660 -0.25056
```

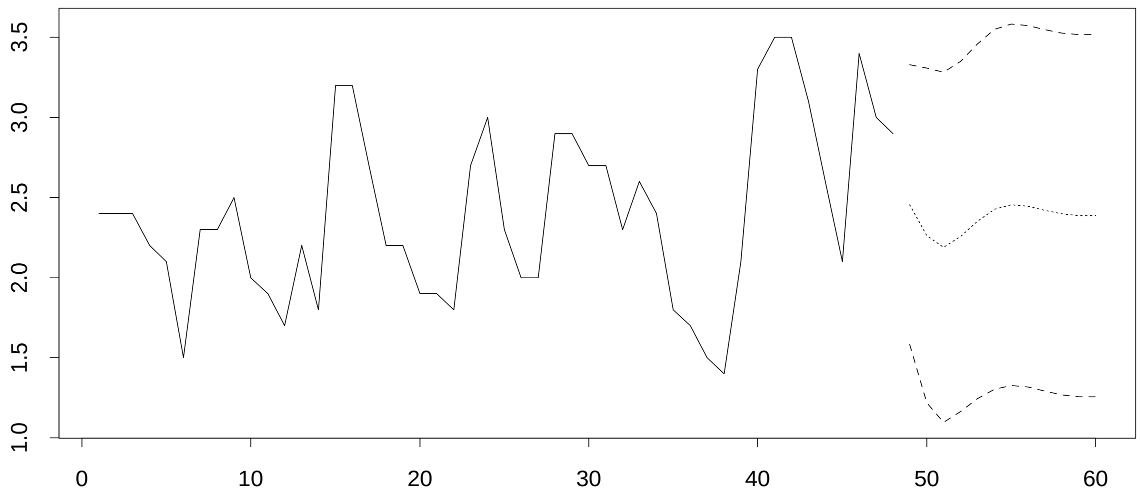


Fig 5.21 *Predictions for 12 step ahead for* `lh[,3]` *with approximate 95% confidence intervals.*

5.9 Forecasting

After fitting an ARIMA model to the data, we can use the function `arima.forecast` to predict future values.

```
x <- lh[,3] - mean(lh[,3])
xmle <- arima.mle(x, model=list(order=c(3,0,0)), n.cond=3)
xf <- arima.forecast(x, n=12, model=xmle$model)
xf$mean <- xf$mean + mean(lh[,3])

plot(seq(1,60),c(lh[,3], xf$mean[1:10],
   max(xf$mean+2*xf$std.err),min(xf$mean-2*xf$std.er)),type="n",xlab=" ")
lines(seq(1,48),lh[,3])
lines(seq(49,60),xf$mean,lty=2)
lines(seq(49,60),xf$mean+2*xf$std.err,lty=3)
lines(seq(49,60),xf$mean-2*xf$std.err,lty=3) # Fig 5.21
```

Remarks

1. Note that the mean of the series was removed before using `arima.mle`, so the forecast values will have the mean added as indicated in the following:

```
xf$mean <- xf$mean + mean(lh[,3])
```

2. The standard error of the prediction does not account for the effect of the estimates of the mean and the parameters of ARIMA models.

5.10 Diagnostics and portmanteau test

The function `arima.diag` yields three graphs. The first is the plot of the standardized residuals, the second plots the auto-correlations and the last plots the P-values of the portmanteau test [Box-Peirce (1970)]

$$Q_K = n \sum_{k=1}^{K} C_k^2,$$

where C_k is the sample auto-correlation at lag k. The parameter K is set by the parameter `gof.lag` (which defaults to 10) plus $p+q$. If the ARIMA is approximately correct, and the noise is Gaussian, then the test statistic Q has an approximate χ^2 distribution with $K-(p+q)$ degree of freedom.

5.11 SAS Examples

```
/*
    Illustrate the use of PROC AUTOREG
    via a simulated AR process.
*/

options ls=70 ps=100 nodate pageno=1;

title 'Simulate a Time Series with Roots: (X-1.25)(X**4-1.25)';
data ar;
   x=0;
   do i=1 to 2500;
      x=x+.05*ranuni(387213);
      e=rannor(387317);
      n=sum( e, .8*n1, .8*n4, -.64*n5);  * the ar process;
      y=20+x+n;                          * the systematic model;
      n5=n4; n4=n3; n3=n2; n2=n1; n1=n;  * set lags;
      if i>500 then output;
      end;
proc autoreg partial data=ar;
     model y=x /method=ml nlag=(1 4 5);
```

```
run;
```

```
        Simulate a Time Series with Roots: (X-1.25)(X**4-1.25)       1

                          Autoreg Procedure

Dependent Variable = Y


                    Ordinary Least Squares Estimates

            SSE           28976.48    DFE             1998
            MSE           14.50274    Root MSE    3.808247
            SBC           11037.63    AIC         11026.43
            Reg Rsq         0.9343    Total Rsq     0.9343
            Durbin-Watson   0.1825


   Variable     DF        B Value   Std Error   t Ratio Approx Prob

   Intercept     1      20.348917      0.2382    85.430      0.0001
   X             1       0.980104     0.00582   168.534      0.0001


                     Estimates of Autocorrelations

                      Lag  Covariance  Correlation

                        0    14.48824     1.000000
                        1    13.15059     0.907674
                        2    12.38431     0.854783
                        3    12.50354     0.863013
                        4    13.12971     0.906232
                        5    11.73285     0.809819


                        Partial Autocorrelations
```

1	0.907674
4	0.481559
5	-0.626105

Preliminary MSE = 1.030145

Estimates of the Autoregressive Parameters

Lag	Coefficient	Std Error	t Ratio
1	-0.79358849	0.014503	-54.721
4	-0.78965427	0.014610	-54.050
5	0.62610501	0.017457	35.865

Expected Autocorrelations

Lag	Autocorr
0	1.0000
1	0.9059
2	0.8599
3	0.8594
4	0.9044
5	0.8070

Maximum Likelihood Estimates

SSE	1975.488	DFE	1995
MSE	0.99022	Root MSE	0.995098
SBC	5694.896	AIC	5666.892
Reg Rsq	0.2651	Total Rsq	0.9955
Durbin-Watson	2.0214		

Variable	DF	B Value	Std Error	t Ratio	Approx Prob
Intercept	1	20.809245	1.4831	14.031	0.0001
X	1	0.968823	0.0361	26.823	0.0001
A(1)	1	-0.794492	0.0144	-55.060	0.0001
A(4)	1	-0.796611	0.0144	-55.474	0.0001
A(5)	1	0.631925	0.0173	36.446	0.0001

```
                 Expected Autocorrelations

                        Lag Autocorr
                          0   1.0000
                          1   0.9083
                          2   0.8637
                          3   0.8639
                          4   0.9090
                          5   0.8138


          Autoregressive parameters assumed given.


   Variable     DF        B Value   Std Error   t Ratio Approx Prob

   Intercept     1      20.809245      1.4830    14.032      0.0001
   X             1       0.968823      0.0361    26.827      0.0001
```

Seasonal modeling

```
/*
    ARIMA EXAMPLE

    Kohn, Robert, and Ansley, Craig F. (1986).
    Estimation, prediction, and interpolation
    for ARIMA Models with missing data.
    JASA, 81, 751-761.
*/

options ls=70 ps=100 nodate pageno=1;

/*
   Seasonal Model for the Airline Series
*/

title 'International Airline Passengers';
data seriesg;
   infile 'c:/z2/b259/sas/airline';
   input x @@;
```

```
   xlog=log(x);
   year=int((_n_-1)/12)+1949;
   month=mod(_n_-1,12)+1;
   date=mdy(month,1,year);
   drop year month;
   format date monyy.;

proc arima data=seriesg;
   identify var=xlog center noprint;  * Check for differencing;
   estimate p=(1)(12) noconstant method=uls;
   estimate p=(1)(12) q=(1)(12) noconstant method=uls converge=0.1;

   identify var=xlog(1,12);   * Fit the usual model;
   estimate q=(1)(12) noconstant method=uls;
   forecast out=b back=14 lead=24 id=date interval=month;
quit;

proc plot data=b(firstobs=115);
   plot xlog*date='*' forecast*date='F' l95*date='L' u95*date='U'
      /overlay haxis='1JAN58'd '1JAN59'd '1JAN60'd '1JAN61'd
         '1JAN62'd;
data c;
   set b;
   x=exp(xlog);
   forecast=exp(forecast+std*std/2);
   l95=exp(l95);
   u95=exp(u95);
run;

proc plot data=c(firstobs=115);
   plot x*date='*' forecast*date='F' l95*date='L' u95*date='U'
      /overlay haxis='1JAN58'd '1JAN59'd '1JAN60'd '1JAN61'd
         '1JAN62'd;
quit;
```

```
                   International Airline Passengers                    1

                           ARIMA Procedure


                Unconditional Least Squares Estimation
```

```
                            Approx.
Parameter   Estimate     Std Error    T Ratio  Lag
AR1,1        0.94391       0.02792      33.80    1
AR2,1        0.99948     0.0040711     245.50   12

Variance  Estimate = 0.00187139
Std Error Estimate = 0.04325953
AIC                = -407.87453
SBC                =  -401.9349
Number of Residuals=        144

          Correlations of the Estimates


       Parameter       AR1,1       AR2,1

       AR1,1            1.000      -0.024
       AR2,1           -0.024       1.000


       Autocorrelation Check of Residuals

 To   Chi                     Autocorrelations
Lag  Square DF   Prob
  6   22.12  4  0.000 -0.302  0.130 -0.171  0.045  0.082  0.058
 12   49.04 10  0.000 -0.029  0.023  0.191 -0.053  0.089 -0.349
 18   61.60 16  0.000  0.166 -0.037  0.165 -0.113  0.086  0.031
 24   73.77 22  0.000  0.009 -0.095  0.061 -0.067  0.230 -0.007



Model for variable XLOG

Data have been centered by subtracting the value 5.5421759585.
No mean term in this model.

Autoregressive Factors
Factor 1: 1 - 0.94391 B**(1)
Factor 2: 1 - 0.99948 B**(12)

                International Airline Passengers                2
```

ARIMA Procedure

Unconditional Least Squares Estimation

Parameter	Estimate	Approx. Std Error	T Ratio	Lag
MA1,1	0.36660	0.08105	4.52	1
MA2,1	0.61362	0.07709	7.96	12
AR1,1	0.99979	0.0045936	217.65	1
AR2,1	1.00000	0.0001061	9426.31	12

```
Variance  Estimate = 0.00125734
Std Error Estimate = 0.03545895
AIC                = -396.40812
SBC                = -384.52887
Number of Residuals=        144
```

Correlations of the Estimates

Parameter	MA1,1	MA2,1	AR1,1	AR2,1
MA1,1	1.000	-0.067	0.121	-0.002
MA2,1	-0.067	1.000	0.024	0.016
AR1,1	0.121	0.024	1.000	-0.002
AR2,1	-0.002	0.016	-0.002	1.000

Autocorrelation Check of Residuals

To Lag	Chi Square	DF	Prob	Autocorrelations					
6	6.18	2	0.046	-0.005	0.018	-0.126	-0.129	0.061	0.068
12	9.55	8	0.298	-0.066	-0.043	0.104	-0.063	0.025	0.008
18	14.85	14	0.388	0.020	0.039	0.048	-0.162	0.040	0.006
24	27.19	20	0.130	-0.102	-0.101	-0.033	-0.030	0.219	0.034

Model for variable XLOG

```
Data have been centered by subtracting the value 5.5421759585.
No mean term in this model.

Autoregressive Factors
Factor 1: 1 - 0.99979 B**(1)
Factor 2: 1 -  1 B**(12)


Moving Average Factors
Factor 1: 1 - 0.3666 B**(1)
Factor 2: 1 - 0.61362 B**(12)

               International Airline Passengers                  3

                       ARIMA Procedure


              Name of variable = XLOG.

              Period(s) of Differencing = 1,12.
              Mean of working series = 0.000291
              Standard deviation     = 0.045673
              Number of observations =      131
              NOTE: The first 13 observations were eliminated by
                    differencing.

                       Autocorrelations

Lag Covariance Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
  0  0.0020860    1.00000  |                    |********************|
  1 -0.0007116   -0.34112  |              *******|  .                 |
  2 0.00021913    0.10505  |                .   |** .                |
  3 -0.0004217   -0.20214  |               ****|    .                |
  4 0.00004456    0.02136  |                .   |    .                |
  5  0.0001161    0.05565  |                .   |*   .                |
  6 0.00006426    0.03080  |                .   |*   .                |
  7 -0.0001159   -0.05558  |                .  *|    .                |
  8 -1.5867E-6   -0.00076  |                .   |    .                |
  9 0.00036791    0.17637  |                .   |****                 |
 10 -0.0001593   -0.07636  |                . **|    .                |
 11 0.00013431    0.06438  |                .   |*   .                |
 12 -0.0008065   -0.38661  |             ********|    .                |
```

```
13 0.00031624    0.15160  |               .    |*** .               |
14 -0.0001202   -0.05761  |               .   *|    .               |
15   0.000312    0.14957  |               .    |*** .               |
16 -0.0002898   -0.13894  |               . ***|    .               |
17 0.00014703    0.07048  |               .    |*   .               |
18 0.00003261    0.01563  |               .    |    .               |
19 -0.0000221   -0.01061  |               .    |    .               |
20 -0.0002435   -0.11673  |               .  **|    .               |
21 0.00008042    0.03855  |               .    |*   .               |
22 -0.0001906   -0.09136  |               .  **|    .               |
23 0.00046574    0.22327  |               .    |****.               |
24 -0.0000384   -0.01842  |               .    |    .               |
                          "." marks two standard errors


                      Inverse Autocorrelations

          Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
            1    0.32632  |                 .  |*******             |
            2    0.09594  |                 .  |**.                 |
            3    0.09992  |                 .  |**.                 |
            4    0.10889  |                 .  |**.                 |
            5   -0.12127  |                 .**|  .                 |
            6   -0.16601  |                 ***|  .                 |
            7   -0.05979  |                 . *|  .                 |
            8    0.02949  |                 .  |* .                 |
            9   -0.08480  |                 .**|  .                 |
           10    0.01413  |                 .  |  .                 |
           11    0.10508  |                 .  |**.                 |
           12    0.37985  |                 .  |********            |
           13    0.12446  |                 .  |**.                 |
           14    0.05655  |                 .  |* .                 |
           15    0.05144  |                 .  |* .                 |
           16    0.09499  |                 .  |**.                 |
           17   -0.04675  |                 . *|  .                 |
           18   -0.05409  |                 . *|  .                 |
           19    0.04468  |                 .  |* .                 |
           20    0.03780  |                 .  |* .                 |
           21   -0.06184  |                 . *|  .                 |
           22   -0.00864  |                 .  |  .                 |
           23   -0.06173  |                 . *|  .                 |
           24    0.04423  |                 .  |* .                 |
```

```
                 International Airline Passengers                  4

                          ARIMA Procedure


                     Partial Autocorrelations

Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
  1   -0.34112  |             *******|  .                 |
  2   -0.01281  |                 .  |  .                 |
  3   -0.19266  |                ****|  .                 |
  4   -0.12503  |                 ***|  .                 |
  5    0.03309  |                 .  |* .                 |
  6    0.03468  |                 .  |* .                 |
  7   -0.06019  |                 . *|  .                 |
  8   -0.02022  |                 .  |  .                 |
  9    0.22558  |                 .  |*****               |
 10    0.04307  |                 .  |* .                 |
 11    0.04659  |                 .  |* .                 |
 12   -0.33869  |             *******|  .                 |
 13   -0.10918  |                 .**|  .                 |
 14   -0.07684  |                 .**|  .                 |
 15   -0.02175  |                 .  |  .                 |
 16   -0.13955  |                 ***|  .                 |
 17    0.02589  |                 .  |* .                 |
 18    0.11482  |                 .  |**.                 |
 19   -0.01316  |                 .  |  .                 |
 20   -0.16743  |                 ***|  .                 |
 21    0.13240  |                 .  |***                 |
 22   -0.07204  |                 . *|  .                 |
 23    0.14285  |                 .  |***                 |
 24   -0.06733  |                 . *|  .                 |
```

Autocorrelation Check for White Noise

To Lag	Chi Square	DF	Prob	Autocorrelations					
6	23.27	6	0.001	-0.341	0.105	-0.202	0.021	0.056	0.031
12	51.47	12	0.000	-0.056	-0.001	0.176	-0.076	0.064	-0.387
18	62.44	18	0.000	0.152	-0.058	0.150	-0.139	0.070	0.016
24	74.27	24	0.000	-0.011	-0.117	0.039	-0.091	0.223	-0.018

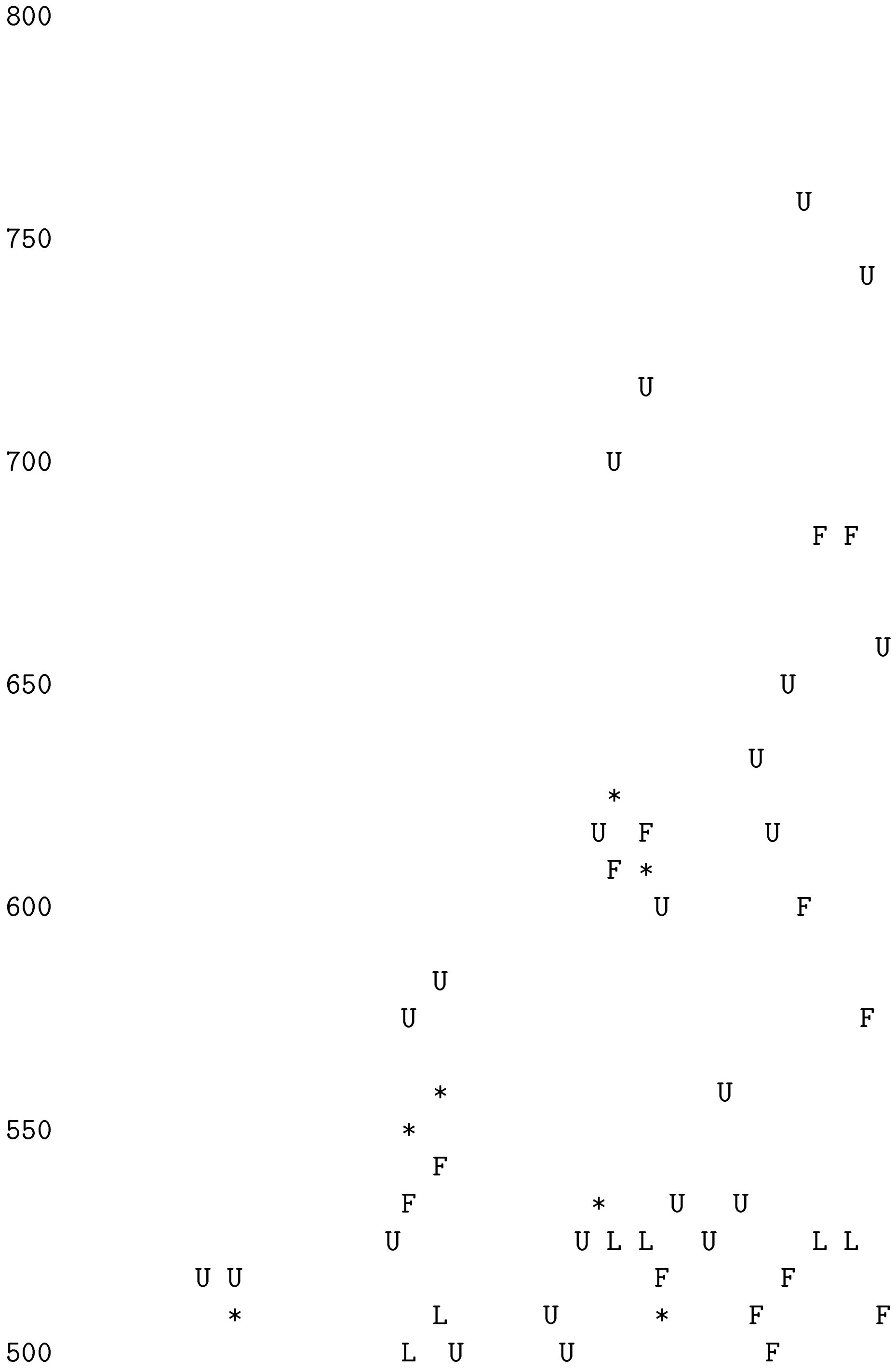
```
               International Airline Passengers                    5

                       ARIMA Procedure


            Unconditional Least Squares Estimation

                              Approx.
     Parameter   Estimate    Std Error   T Ratio  Lag
     MA1,1        0.39594      0.08149      4.86   1
     MA2,1        0.61331      0.07961      7.70  12

     Variance  Estimate = 0.00136313
     Std Error Estimate = 0.03692065
     AIC                = -484.75494
     SBC                = -479.00455
     Number of Residuals=        131

                 Correlations of the Estimates


            Parameter       MA1,1       MA2,1

            MA1,1           1.000      -0.055
            MA2,1          -0.055       1.000


              Autocorrelation Check of Residuals

 To   Chi                     Autocorrelations
Lag  Square DF   Prob
  6    5.56  4  0.235  0.022  0.024 -0.125 -0.129  0.057  0.065
 12    8.49 10  0.582 -0.065 -0.042  0.102 -0.060  0.023  0.007
 18   13.23 16  0.656  0.022  0.039  0.045 -0.162  0.035  0.001
 24   24.99 22  0.298 -0.106 -0.104 -0.037 -0.027  0.219  0.040



            Model for variable XLOG

            No mean term in this model.
            Period(s) of Differencing = 1,12.
```

```
Moving Average Factors
Factor 1: 1 - 0.39594 B**(1)
Factor 2: 1 - 0.61331 B**(12)

 International Airline Passengers                  6

                ARIMA Procedure


Forecasts for variable XLOG

 Obs     Forecast Std Error   Lower 95%   Upper 95%
           Actual  Residual
 131       5.8733    0.0369      5.8010      5.9457
           5.8916    0.0183
 132       5.9782    0.0431      5.8936      6.0627
           6.0039    0.0257
 133       6.0169    0.0486      5.9218      6.1121
           6.0331    0.0162
 134       5.9693    0.0534      5.8646      6.0740
           5.9687   -0.0006
 135       6.1252    0.0579      6.0117      6.2387
           6.0379   -0.0873
 136       6.0990    0.0620      5.9774      6.2206
           6.1334    0.0344
 137       6.1366    0.0659      6.0074      6.2658
           6.1570    0.0204
 138       6.2821    0.0696      6.1457      6.4186
           6.2823    0.0001
 139       6.4086    0.0731      6.2653      6.5518
           6.4329    0.0243
 140       6.4195    0.0764      6.2697      6.5692
           6.4069   -0.0126
 141       6.2446    0.0796      6.0886      6.4006
           6.2305   -0.0141
 142       6.1138    0.0827      5.9517      6.2758
           6.1334    0.0196
 143       5.9779    0.0904      5.8007      6.1550
           5.9661   -0.0117
 144       6.0827    0.0955      5.8954      6.2699
           6.0684   -0.0142
```

```
          145      6.1214    0.1004      5.9246      6.3183
          146      6.0738    0.1051      5.8678      6.2797
          147      6.2297    0.1095      6.0150      6.4444
          148      6.2035    0.1138      5.9804      6.4266
          149      6.2411    0.1179      6.0099      6.4723
          150      6.3867    0.1219      6.1477      6.6256
          151      6.5131    0.1258      6.2666      6.7597
          152      6.5240    0.1295      6.2701      6.7779
          153      6.3491    0.1332      6.0881      6.6101
          154      6.2183    0.1367      5.9503      6.4862
                  International Airline Passengers                  7

            Plot of XLOG*DATE.      Symbol used is '*'.
            Plot of FORECAST*DATE.  Symbol used is 'F'.
            Plot of L95*DATE.       Symbol used is 'L'.
            Plot of U95*DATE.       Symbol used is 'U'.

     XLOG
      6.8

                                                        U
                                                       U



      6.7




                                                      U
                                                         U
      6.6
                                            U
                                           U

                                                        F
                                                       F
      6.5
                                                          U
                                                     U
```

```
                                            U
                                    *        U
                                   UFF
6.4                                  *U
                                               F
                         U
                        U
                                                  F
                         *
                                          U
6.3                     *F
                        F          *       U
                                  U LL U U      LL
                       U
            UU                  U     F       F
             *           LU           *     F
                        L        U                 F
6.2         *                                F
                       F
            FF
                       *  F       *     U
              U           *       FL           L
                                F*     *  F
                       L      U        F
6.1         LL                   F
                           U          L  F        L
              F       U   L    U         * F
                    U        U
                     U*         *
                              *
                           *  F L L         L L
6.0           *     * F      *
                    F*                       L
                     F       F * L      *
               U                       L           L
                           LU
                 UU   L       L           L

5.9                ULL       L           L
               *  *         *
                            F  L           L
```

```
                       FF

                      U  *
                     L *
      5.8                                 L           L
                       LL

                      F  L
                      *


      5.7
                      L






      5.6


           JAN58       JAN59       JAN60       JAN61       JAN62

                                   DATE

NOTE: 10 obs had missing values.  7 obs hidden.
                   International Airline Passengers                  8

             Plot of X*DATE.         Symbol used is '*'.
             Plot of FORECAST*DATE.  Symbol used is 'F'.
             Plot of L95*DATE.       Symbol used is 'L'.
             Plot of U95*DATE.       Symbol used is 'U'.

      X

    900


                                                         U

                                                       U
    850
```


800
750
700
650
600
550
500

```
                *
                  F         F
                F           *           *      U
                   U            *       FL            L
                                      F*      *  F
    450           L         L       U  F      F
                L                U           L  F         L
                   F       U    L  U U          * F
                         U
                          U*        * *
                   L     *       * *F L L          L L
    400            *      *F                   F
                    U    FF        F * L       *    L
                                 LU           L            L
                      UU   L        L            L
                        ULL        L            L
                    *  *          *
    350               FF             L            L
                     U  *
                    L *           L            L
                      LL
                     F  L
                     *
    300
                     L




    250

         JAN58        JAN59        JAN60        JAN61        JAN62

                                   DATE

NOTE: 10 obs had missing values.  7 obs hidden.
```

Intervention Analysis

```
/*
    An Intervention Model for Ozone Data
```

```
    Box, G.E.P., and Tiao, G.C. (1975). Intervention analysis
    with applications to economic and environmental problems.
    J. Amer. Statist. Assoc. 70: 70-92.
*/

options ls=70 ps=100 nodate pageno=1;

title1 'Intervention Data for Ozone Concentration';
title2 'Box and Tiao, JASA 1975 P.70';
data air;
   infile 'c:/z2/b259/sas/ozone';
   input ozone @@;
   label ozone='Ozone Concentration'
         x1='Intervention'
         summer='Summer Months Intervention'
         winter='Winter Months Intervention';
   format date date7.;
   year=55+floor((_n_-1)/12);
   month=mod(_n_-1,12)+1;
   date=mdy(month,1,year);
   x1=year>=60;
   summer=(5<month<11)*(year>65);
   winter=(year>65)-summer;

proc arima;  * Fit the Intervention Model;

/*  Seasonally Difference Ozone Series  */
   identify var=ozone(12) crosscor=(x1(12) summer winter) noprint;

/*
  Estimate a straight regression with a seasonal MA
  model using unconditional least squares
  conditional least squares and maximum likelihood.
*/
   estimate q=(1)(12) input=(x1 summer winter) printall maxit=20
            noconstant method=uls;
   estimate q=(1)(12) input=(x1 summer winter) printall maxit=20
            noconstant method=cls;
   estimate q=(1)(12) input=(x1 summer winter) printall maxit=20
            noconstant method=ml;
```

```
   forecast lead=14 back=2;  * Forecast Some Values;
quit;
```

```
              Intervention Data for Ozone Concentration             1
                     Box and Tiao, JASA 1975 P.70

                           ARIMA Procedure


                        Preliminary Estimation


                   Initial Moving Average Estimates

                     1       -0.29241

                   Initial Moving Average Estimates

                    12        0.40740

                 White Noise Variance Est = 0.944969

              Intervention Data for Ozone Concentration             2
                     Box and Tiao, JASA 1975 P.70

                           ARIMA Procedure


                 Conditional Least Squares Estimation

Iteration   SSE     MA1,1    MA2,1     NUM1     NUM2     NUM3   Lambda
                R Crit
  0   154.53372  -0.2924   0.4074  -1.1349  -0.1173  0.05581  0.00001
                      1
  1   146.19630  -0.2926   0.5984  -1.2029  -0.2978  -0.1157     1E-6
                0.230552
  2   145.87909  -0.3007   0.5924  -1.2617  -0.2625 -0.08247     1E-7
                0.046601
  3   145.87883  -0.2998   0.5924  -1.2625  -0.2615 -0.08197     1E-8
                0.001345
  4   145.87883  -0.2998   0.5923  -1.2624  -0.2615 -0.08196     1E-9
```

```
                0.000125


                Unconditional Least Squares Estimation

Iteration   SSE     MA1,1    MA2,1     NUM1     NUM2     NUM3   Lambda
                R Crit
  0    133.79418  -0.2998   0.5923  -1.2624  -0.2615 -0.08196  0.00001
                        1
  1    125.08193  -0.2683   0.8195  -1.3717  -0.2336 -0.07355     1E-6
                0.222194
  2    124.20610  -0.2573   0.8691  -1.3642  -0.2178 -0.08124     1E-7
                0.063641
  3    123.22633  -0.2561   0.9139  -1.3657  -0.2115 -0.08283     1E-8
                0.055774
  4    122.09956  -0.2544   0.9422  -1.3694  -0.2072 -0.08356     1E-8
                0.118515
  5    120.84221  -0.2535   0.9627  -1.3718  -0.2049 -0.08389     1E-8
                0.263332
  6    119.80833  -0.2531   0.9755  -1.3729  -0.2039 -0.08401     1E-8
                0.500249
  7    117.52192  -0.2526   0.9981  -1.3742  -0.2027 -0.08415     1E-8
                0.730024
  8    117.46479  -0.2526   0.9986  -1.3742  -0.2027 -0.08415     1E-8
                0.999124
  9    117.40764  -0.2526   0.9990  -1.3742  -0.2027 -0.08415     1E-8
                0.999627
 10    117.35022  -0.2526   0.9995  -1.3742  -0.2027 -0.08415     1E-8
                0.999625
 11    117.33589  -0.2526   0.9997  -1.3742  -0.2027 -0.08415     1E-8
                0.999915
 12    117.32156  -0.2526   0.9998  -1.3742  -0.2027 -0.08415     1E-8
                0.999955
 13    117.30723  -0.2526   0.9999  -1.3742  -0.2027 -0.08415     1E-8
                0.999982
 14    117.30365  -0.2526   0.9999  -1.3742  -0.2027 -0.08415     1E-8
                0.999997
 15    117.30007  -0.2526   1.0000  -1.3742  -0.2027 -0.08415     1E-8
                0.999999
 16    117.29649  -0.2526   1.0000  -1.3742  -0.2027 -0.08415     1E-8
                        1
 17    117.29649  -0.2526   1.0000  -1.3742  -0.2027 -0.08415     1E-8
```

```
                    1

WARNING: The model defined by the new estimates is unstable. The
         iteration process has been terminated.
WARNING: Estimates may not have converged.

                ARIMA Estimation Optimization Summary

   Estimation Method:                    Unconditional Least Squares
   Parameters Estimated:                                           5
   Termination Criteria:       Maximum Relative Change in Estimates
   Iteration Stopping Value:                                   0.001
   Criteria Value:                                        2.00001211
   Maximum Absolute Value of Gradient:                    58.6479236
   R-Square (Relative Change in Regression SSE) from Last Iteration
      Step:                                                        1
   Objective Function:                      Sum of Squared Residuals
   Objective Function Value:                              117.296489
   Marquardt's Lambda Coefficient:                              1E-8
   Numerical Derivative Perturbation Delta:                    0.001
   Iterations:                                                    17
   Warning Message:               Estimates may not have converged.


                Unconditional Least Squares Estimation

                            Approx.
   Parameter   Estimate    Std Error   T Ratio  Lag  Variable Shift
   MA1,1       -0.25264     0.06882     -3.67    1   OZONE        0
   MA2,1        1.00000     0.14179      7.05   12   OZONE        0
   NUM1        -1.37418     0.15993     -8.59    0   X1           0
   NUM2        -0.20272     0.04372     -4.64    0   SUMMER       0
   NUM3        -0.08415     0.03786     -2.22    0   WINTER       0

              Intervention Data for Ozone Concentration            3
                     Box and Tiao, JASA 1975 P.70

                           ARIMA Procedure


   Variance  Estimate = 0.58942959
   Std Error Estimate = 0.76774318
```

```
AIC                 =  510.78033
SBC                 =  527.37093
Number of Residuals=         204
```

Correlations of the Estimates

Variable	Parameter	OZONE MA1,1	OZONE MA2,1	X1 NUM1	SUMMER NUM2	WINTER NUM3
OZONE	MA1,1	1.000	0.003	-0.011	0.034	-0.028
OZONE	MA2,1	0.003	1.000	-0.008	0.009	0.003
X1	NUM1	-0.011	-0.008	1.000	-0.296	-0.339
SUMMER	NUM2	0.034	0.009	-0.296	1.000	0.174
WINTER	NUM3	-0.028	0.003	-0.339	0.174	1.000

Autocorrelation Check of Residuals

To Lag	Chi Square	DF	Prob	Autocorrelations					
6	7.06	4	0.133	0.023	0.069	0.021	0.068	-0.043	0.146
12	19.42	10	0.035	-0.044	-0.072	-0.060	0.037	0.017	0.210
18	24.43	16	0.080	0.040	0.023	-0.108	0.011	-0.061	0.069
24	37.11	22	0.023	-0.018	-0.086	-0.082	0.106	-0.049	0.163
30	45.51	28	0.020	-0.075	-0.023	-0.001	-0.040	-0.120	0.114
36	58.37	34	0.006	0.002	-0.111	-0.027	0.101	-0.064	0.157

```
Model for variable OZONE

No mean term in this model.
Period(s) of Differencing = 12.

Moving Average Factors
Factor 1: 1 + 0.25264 B**(1)
Factor 2: 1 -  1 B**(12)


Input Number 1 is X1.
Period(s) of Differencing = 12.
```

```
                Overall Regression Factor  = -1.37418

                Input Number 2 is SUMMER.
                Overall Regression Factor  = -0.20272

                Input Number 3 is WINTER.
                Overall Regression Factor  = -0.08415

              Intervention Data for Ozone Concentration               4
                     Box and Tiao, JASA 1975 P.70

                           ARIMA Procedure


                        Preliminary Estimation


                   Initial Moving Average Estimates

                     1       -0.29241

                   Initial Moving Average Estimates

                    12        0.40740

                 White Noise Variance Est = 0.944969

              Intervention Data for Ozone Concentration               5
                     Box and Tiao, JASA 1975 P.70

                           ARIMA Procedure


                 Conditional Least Squares Estimation

Iteration    SSE     MA1,1    MA2,1      NUM1      NUM2      NUM3   Lambda
                R Crit
  0    154.53372  -0.2924   0.4074  -1.1349  -0.1173  0.05581  0.00001
                        1
  1    146.19630  -0.2926   0.5984  -1.2029  -0.2978  -0.1157     1E-6
                0.230552
  2    145.87909  -0.3007   0.5924  -1.2617  -0.2625 -0.08247     1E-7
```

```
               0.046601
 3   145.87883  -0.2998   0.5924  -1.2625  -0.2615 -0.08197     1E-8
               0.001345
 4   145.87883  -0.2998   0.5923  -1.2624  -0.2615 -0.08196     1E-9
               0.000125

                ARIMA Estimation Optimization Summary

 Estimation Method:                     Conditional Least Squares
 Parameters Estimated:                                          5
 Termination Criteria:        Maximum Relative Change in Estimates
 Iteration Stopping Value:                                  0.001
 Criteria Value:                                       0.00022638
 Alternate Criteria:         Relative Change in Objective Function
 Alternate Criteria Value:                             2.91843E-9
 Maximum Absolute Value of Gradient:                   0.01748851
 R-Square (Relative Change in Regression SSE) from Last Iteration
    Step:                                              0.00012455
 Objective Function:                        Sum of Squared Residuals
 Objective Function Value:                             145.878826
 Marquardt's Lambda Coefficient:                             1E-9
 Numerical Derivative Perturbation Delta:                   0.001
 Iterations:                                                    4


                 Conditional Least Squares Estimation

                          Approx.
 Parameter   Estimate    Std Error   T Ratio  Lag  Variable Shift
 MA1,1       -0.29983     0.06777     -4.42    1   OZONE        0
 MA2,1        0.59234     0.05828     10.16   12   OZONE        0
 NUM1        -1.26243     0.25861     -4.88    0   X1           0
 NUM2        -0.26154     0.08864     -2.95    0   SUMMER       0
 NUM3        -0.08196     0.07570     -1.08    0   WINTER       0

 Variance  Estimate = 0.73305943
 Std Error Estimate =  0.8561889
 AIC                = 520.516811*
 SBC                = 537.107411*
 Number of Residuals=        204
 * Does not include log determinant.
```

Correlations of the Estimates

Variable	Parameter	OZONE MA1,1	OZONE MA2,1	X1 NUM1	SUMMER NUM2	WINTER NUM3
OZONE	MA1,1	1.000	-0.000	-0.028	0.036	-0.033
OZONE	MA2,1	-0.000	1.000	-0.138	0.077	0.031
X1	NUM1	-0.028	-0.138	1.000	-0.027	-0.021
SUMMER	NUM2	0.036	0.077	-0.027	1.000	0.101
WINTER	NUM3	-0.033	0.031	-0.021	0.101	1.000

Autocorrelation Check of Residuals

To Lag	Chi Square	DF	Prob	Autocorrelations					
6	7.12	4	0.130	0.005	0.011	0.013	0.083	0.004	0.163
12	10.61	10	0.389	-0.000	-0.085	-0.070	-0.029	0.057	-0.005
18	16.36	16	0.428	0.073	-0.036	-0.131	0.016	-0.013	0.040
24	21.50	22	0.490	0.024	-0.086	-0.090	0.077	-0.017	-0.013
30	26.97	28	0.520	-0.059	-0.058	0.026	-0.036	-0.081	0.087
36	30.72	34	0.629	0.033	-0.100	-0.007	0.056	-0.027	0.015

```
Model for variable OZONE

No mean term in this model.
Period(s) of Differencing = 12.

Moving Average Factors
Factor 1: 1 + 0.29983 B**(1)
Factor 2: 1 - 0.59234 B**(12)


Input Number 1 is X1.
Period(s) of Differencing = 12.
Overall Regression Factor  = -1.26243

Input Number 2 is SUMMER.
```

```
              Intervention Data for Ozone Concentration               6
                     Box and Tiao, JASA 1975 P.70

                           ARIMA Procedure


                Overall Regression Factor  = -0.26154

                Input Number 3 is WINTER.
                Overall Regression Factor  = -0.08196

              Intervention Data for Ozone Concentration               7
                     Box and Tiao, JASA 1975 P.70

                           ARIMA Procedure


                        Preliminary Estimation


                   Initial Moving Average Estimates

                     1       -0.29241

                   Initial Moving Average Estimates

                    12        0.40740

                 White Noise Variance Est = 0.944969

              Intervention Data for Ozone Concentration               8
                     Box and Tiao, JASA 1975 P.70

                           ARIMA Procedure


                 Conditional Least Squares Estimation

Iteration   SSE     MA1,1    MA2,1     NUM1     NUM2     NUM3   Lambda
                R Crit
  0   154.53372  -0.2924   0.4074  -1.1349  -0.1173  0.05581  0.00001
                       1
```

```
  1   146.19630  -0.2926   0.5984  -1.2029  -0.2978  -0.1157     1E-6
                 0.230552
  2   145.87909  -0.3007   0.5924  -1.2617  -0.2625 -0.08247     1E-7
                 0.046601
  3   145.87883  -0.2998   0.5924  -1.2625  -0.2615 -0.08197     1E-8
                 0.001345
  4   145.87883  -0.2998   0.5923  -1.2624  -0.2615 -0.08196     1E-9
                 0.000125


                    Maximum Likelihood Estimation

Iter    Loglike     MA1,1    MA2,1     NUM1     NUM2     NUM3   Lambda
                 R Crit
  0  -249.07778  -0.2998   0.5923  -1.2624  -0.2615 -0.08196  0.00001
                        1
  1  -245.89135  -0.2683   0.7663  -1.3449  -0.2398 -0.07578     1E-6
                 0.169445
  2  -245.88484  -0.2665   0.7662  -1.3305  -0.2394 -0.08025     1E-7
                 0.008044
  3  -245.88482  -0.2669   0.7666  -1.3307  -0.2394 -0.08020     1E-8
                 0.000603
  4  -245.88481  -0.2668   0.7666  -1.3306  -0.2394 -0.08021     1E-9
                 0.000073

                 ARIMA Estimation Optimization Summary

   Estimation Method:                            Maximum Likelihood
   Parameters Estimated:                                          5
   Termination Criteria:       Maximum Relative Change in Estimates
   Iteration Stopping Value:                                  0.001
   Criteria Value:                                       0.00019544
   Alternate Criteria:        Relative Change in Objective Function
   Alternate Criteria Value:                             1.24707E-8
   Maximum Absolute Value of Gradient:                   0.00712034
   R-Square (Relative Change in Regression SSE) from Last Iteration
      Step:                                              0.00007323
   Objective Function:                      Log Gaussian Likelihood
   Objective Function Value:                             -245.88481
   Marquardt's Lambda Coefficient:                             1E-9
   Numerical Derivative Perturbation Delta:                   0.001
   Iterations:                                                    4
```

Maximum Likelihood Estimation

Parameter	Estimate	Approx. Std Error	T Ratio	Lag	Variable	Shift
MA1,1	-0.26684	0.06710	-3.98	1	OZONE	0
MA2,1	0.76665	0.05973	12.83	12	OZONE	0
NUM1	-1.33062	0.19236	-6.92	0	X1	0
NUM2	-0.23936	0.05952	-4.02	0	SUMMER	0
NUM3	-0.08021	0.04978	-1.61	0	WINTER	0

```
Variance  Estimate = 0.63450562
Std Error Estimate = 0.79655861
AIC                = 501.769629
SBC                = 518.360229
Number of Residuals=        204
```

Correlations of the Estimates

Variable	Parameter	OZONE MA1,1	OZONE MA2,1	X1 NUM1	SUMMER NUM2	WINTER NUM3
OZONE	MA1,1	1.000	0.090	-0.039	0.062	-0.034
OZONE	MA2,1	0.090	1.000	-0.169	0.211	0.022
X1	NUM1	-0.039	-0.169	1.000	-0.124	-0.107
SUMMER	NUM2	0.062	0.211	-0.124	1.000	0.097
WINTER	NUM3	-0.034	0.022	-0.107	0.097	1.000

Autocorrelation Check of Residuals

To Lag	Chi Square	DF	Prob	Autocorrelations					
6	7.47	4	0.113	0.017	0.054	0.043	0.101	-0.022	0.140
12	10.21	10	0.422	-0.024	-0.059	-0.047	0.014	0.032	0.072
18	14.53	16	0.559	0.054	0.006	-0.110	0.028	-0.042	0.043
24	19.99	22	0.583	0.003	-0.074	-0.074	0.098	-0.038	0.043
30	27.00	28	0.518	-0.072	-0.035	0.023	-0.028	-0.107	0.100
36	32.65	34	0.534	0.022	-0.099	-0.006	0.087	-0.046	0.053

```
Intervention Data for Ozone Concentration                 9
      Box and Tiao, JASA 1975 P.70

            ARIMA Procedure


  Model for variable OZONE

  No mean term in this model.
  Period(s) of Differencing = 12.

  Moving Average Factors
  Factor 1: 1 + 0.26684 B**(1)
  Factor 2: 1 - 0.76665 B**(12)


  Input Number 1 is X1.
  Period(s) of Differencing = 12.
  Overall Regression Factor  = -1.33062

  Input Number 2 is SUMMER.
  Overall Regression Factor  = -0.23936

  Input Number 3 is WINTER.
  Overall Regression Factor  = -0.08021

Intervention Data for Ozone Concentration                10
      Box and Tiao, JASA 1975 P.70

            ARIMA Procedure


Forecasts for variable OZONE

 Obs    Forecast Std Error   Lower 95%   Upper 95%
          Actual  Residual
 215      1.8760    0.7966      0.3147      3.4372
          1.6000   -0.2760
 216      1.3914    0.8244     -0.2245      3.0072
          1.3000   -0.0914
 217      1.4252    0.8244     -0.1906      3.0411
 218      1.8446    0.8244      0.2287      3.4604
```

219	2.4567	0.8244	0.8408	4.0725
220	2.8590	0.8244	1.2431	4.4748
221	3.1501	0.8244	1.5342	4.7659
222	2.7211	0.8244	1.1053	4.3370
223	3.3147	0.8244	1.6989	4.9306
224	3.4787	0.8244	1.8629	5.0946
225	2.9405	0.8244	1.3247	4.5564
226	2.3587	0.8244	0.7429	3.9746
227	1.9232	0.8451	0.2668	3.5796
228	1.3111	0.8466	-0.3481	2.9704

Problems

1. Fit an AR model to each of the components of `lh`. Carry the diagnostic analysis by examining the residuals of the fit.

2. Fit an AR model to each of the components of `bp`. Carry the diagnostic analysis by examining the residuals of the fit.

3. Pick a distribution from `wn.q`, say. Then simulate a simple ARMA process such as AR(1) or MA(1) or ARMA(1,1). Study the robustness property of the function `arima.mle`. Describe what method would you use to evaluate this property.

4. Fit an ARIMA model to each component in `lung`. Describe the diagnostics of your fits.

5. Remove the mean from `births` using `loess`. Then fit a seasonal ARIMA model to the remainder. Also check the residuals of fit.

6. Simulate a simple harmonic series $x_t = 3\cos(2\pi\omega) + w_t$ of length 400 with your own frequency ω. Then fit an AR(p) model to your simulated series. Discuss briefly. (You may want to try this several times.)

7. Show that the autocovariance function of MA (∞) process $X_t = \sum \psi_j w_{t-j}$ is given by

$$\begin{aligned} c(u) &= \operatorname{cov}\left(\sum \psi_j w_{t+u-j}, \sum \psi_k w_{t-k}\right) \\ &= \sum_j \sum_k \psi_j \psi_k \operatorname{cov}(w_{t+u-j}, w_{t-k}) \\ &= \sum_j \sum_k \psi_j \psi_k c_w(u-j+k) \\ &= \sum_{k=0}^{\infty} \psi_k \psi_{k+|u|}. \end{aligned}$$

5.12 Appendix A. Transformation formulae

Let $\mathbf{X}, \mathbf{Y}$ denote random vectors in $\mathbb{R}^n$ with densities $f_{\mathbf{X}}(\cdot)$ and $f_{\mathbf{Y}}(\cdot)$, respectively. Suppose $\mathbf{Y} = \mathbf{g}(\mathbf{X})$, where $\mathbf{g}(\cdot) = (g_1(\cdot), \ldots, g_n(\cdot))^T \in \mathcal{C}^1$ and $g^{-1}(\cdot)$ exists. Then

$$f_{\mathbf{Y}}(\mathbf{y}) = \frac{f_{\mathbf{X}}(g^{-1}(\mathbf{y}))}{|J_g(g^{-1}(\mathbf{y}))|},$$

where $J_g(\cdot)$ is the Jacobian defined by

$$J_g(\cdot) = [D^j g_i(\cdot)] = \begin{pmatrix} \frac{\partial g_1(\cdot)}{\partial x_1} & \cdots & \frac{\partial g_1(\cdot)}{\partial x_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial g_n(\cdot)}{\partial x_1} & \cdots & \frac{\partial g_n(\cdot)}{\partial x_n} \end{pmatrix}.$$

Example. We now apply this transformation formula to get the likelihood function of the AR process:

$$x_t - \phi_1 x_{t-1} - \phi_2 x_{t-2} - \cdots - \phi_p x_{t-p} = w_t, \qquad t = 1, \ldots, n.$$

Write

$$\begin{pmatrix} w_{p+1} \\ w_{p+2} \\ \vdots \\ w_{2p} \\ w_{2p+1} \\ \vdots \\ w_n \end{pmatrix} = A \cdot \begin{pmatrix} x_{p+1} \\ x_{p+2} \\ \vdots \\ x_{2p} \\ x_{2p+1} \\ \vdots \\ x_n \end{pmatrix} - \begin{pmatrix} -\phi_1 x_p \cdots - \phi_p x_1 \\ -\phi_2 x_p \cdots - \phi_p x_2 \\ \vdots \\ -\phi_p x_p \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

where

$$A = \begin{pmatrix} 1 & & & & & & & \\ -\phi_1 & 1 & & & & & & \\ -\phi_2 & -\phi_1 & 1 & & & & & \\ \vdots & \vdots & & \ddots & & & & \\ -\phi_p & -\phi_{p-1} & \cdots & -\phi_1 & 1 & & & \\ 0 & -\phi_p & -\phi_{p-1} & \cdots & -\phi_1 & 1 & & \\ \vdots & & & & \ddots & & \ddots & \\ 0 & \cdots & 0 & -\phi_p & \cdots & -\phi_2 & -\phi_1 & 1 \end{pmatrix}.$$

Suppose $x_1, \ldots, x_p$ are fixed and note that $|A| = 1$. Then given the above transformation formula, the density function of $(x_{p+1}, \ldots, x_n)$ can be obtained from the denisty of $(w_{p+1}, \ldots, w_n)$, which is the product of $(n-p)$ $N(0, \sigma^2)$ densities.

5.13 Appendix B. Covariance generating function

Let $c(u)$ denote the covariance function of a stationary time series $\{x_t\}$. Define the *covariance generating function* by

$$G(z) = \sum_{u=-\infty}^{\infty} c(u) z^u, \qquad z \in \mathbf{C}.$$

To justify the term "generating function", note that $c(u)$ is uniquely determined by the spectral density $f(\cdot)$, viz

$$c(u) = \int f(\lambda)e^{-iu\lambda}d\lambda,$$

and since the spectral density function is defined by

$$f(\lambda) = \frac{1}{2\pi}\sum c(u)\exp(-iu\lambda),$$

or

$$f(\lambda) = \frac{1}{2\pi}G(e^{-i\lambda}).$$

The covariance generating function can be easily computed for some parametric time processes. For example, the generating function for

$$\phi(B)x_t = w_t, \qquad w_t \sim_{\text{iid}} (0, \sigma^2)$$

can be obtained by first compute the spectral density function via

$$f(\lambda) = \frac{\sigma^2}{2\pi|\phi(e^{-i\lambda})|^2} = \frac{\sigma^2}{2\pi}\frac{1}{\phi(e^{i\lambda})\phi(e^{-i\lambda})}.$$

Hence,

$$G(z) = \frac{\sigma^2}{\phi(z)\phi(z^{-1})}.$$

Another example of covariance generating function, based on the moving average process $x_t = \theta(B)w_t$, is given by

$$G(z) = \sigma^2\theta(z)\theta(z^{-1}).$$

The covariance generating function is used to study various parametric time series models, especially the MA(q) ones.

To understand the invertibility condition in greater details, we recall the covariance generating function of $x_t = \theta(B)w_t$ is given by

$$G(z) = \sigma^2\theta(z)\theta(z^{-1}).$$

Suppose that the q-th degree polynomial

$$\theta(z) = 1 + \theta_1 z + \cdots + \theta_q z^q$$

has zeros $\lambda_1, \ldots, \lambda_q$ outside the unit disc so that

$$\theta(z) = \left(1 - \frac{z}{\lambda_1}\right)\cdots\left(1 - \frac{z}{\lambda_q}\right),$$

and

$$G(z) = \sigma^2 \prod \left(1 - \frac{z}{\lambda_j}\right) \prod \left(1 - \frac{1}{z\lambda_j}\right).$$

That is, $G(z)$ is determined by the zeros of $\theta(z)$. In fact, it is determined by

$$\lambda_1^{\pm 1}, \ldots, \lambda_q^{\pm 1}.$$

Since from

$$\left(1 - \frac{z}{\lambda_j}\right) \left(1 - \frac{1}{z\lambda_j}\right) = \frac{1}{\lambda_j^2} \left(1 - \lambda_j z\right) \left(1 - \frac{\lambda_j}{z}\right),$$

we conclude that the form of $G(z)$ is unchange if one of zeros λ_j is replaced by its reciprocals, with variance σ^2 properly replaced by σ^2/λ_j^2.

A practical implication of this observation is that there are 2^q sets of $\lambda_1^{\pm 1}, \ldots, \lambda_q^{\pm 1}$ giving rise the same covariance structure of $x_t = \theta(B)w_t$. But there is one set with values lie outside the unit disc.

Chapter 6

Bivariate Time Series

For univariate time series, we use auto-correlation and its Fourier transformed, the spectral density function, to examine the correlated stuctures. For bivariate series, in addition to the autocorrelation within each series, we will also study the cross-correlations, and model the relationship between them.

6.1 Cross-correlations

Consider a stationary bivariate time series (x_t, y_t). Recall the auto-correlations are defined by

$$\rho_{11}(u) = \text{corr}(x_t, x_{t-u}) \qquad \text{and} \qquad \rho_{22}(u) = \text{corr}(y_t, y_{t-u}), \qquad u = 0, \pm 1, \pm 2, \ldots.$$

The cross-correlation of x and y at lag u is defined by

$$\rho_{12}(u) = \text{corr}(x_t, y_{t-u}), \qquad u = 0, \pm 1, \pm 2, \ldots.$$

Note that $\rho_{12}(u) \neq \rho_{21}(u)$. However, $\rho_{12}(u) = \text{corr}(x_t, y_{t-u}) = \text{corr}(y_{t-u}, x_t) = \rho_{21}(-u)$. Cross-correlations are computed in S-PLUS via `acf`.

```
> x <- matrix(scan("bp"),byrow=T,ncol=2)
> acf(x)$acf:

, , 1
          [,1]         [,2]
 [1,] 1.0000000  0.3928337991
 [2,] 0.4259430  0.1378178746
 [3,] 0.4183573  0.1305052191
 [4,] 0.3576657  0.1115939021
 [5,] 0.3556857  0.1108404547
...
```

```
, , 2
             [,1]       [,2]
 [1,]  0.392833799 1.00000000
 [2,]  0.193048775 0.36747178
 [3,]  0.150568888 0.30360535
 [4,]  0.093718901 0.23694050
 [5,]  0.078654386 0.28379062
...

# acf[i,j,k] is the covariance (or correlation) between the j-th series
# at time t and the k-th series at time t+1-i.
```

6.2 Test for independence

Consider two independent AR(1) processes with coefficients α and β, respectively. Then $\rho_{11}(u) = \alpha^u$, $\rho_{22}(u) = \beta^u$ and

$$\mathrm{var}(\hat{\rho}_{12}) \doteq \frac{1}{n}\frac{1}{1-\alpha\beta}.$$

Thus the 95% CI for $\rho_{12}(u)$ is given by

$$\hat{\rho}_{12}(u) \pm \frac{2}{\sqrt{n(1-\alpha\beta)}}.$$

For $n = 100$, $\alpha = .8$ and $\beta = .9$, this interval is $\hat{\rho}_{12} \pm 0.38$, which means that it is possible to observe a sample cross-correlation as large as 0.38.

The above problem can be remedied by (1) fit AR processes to the univariate series, (2) calculate the cross-correlations of the residuals, and (3) test whether these cross-correlations are significantly different from zero. This is carried out using the function `indtest`, which depends on the function `prewhite` to do (1). Note that the cross-correlations are invariant under linear transformations (prewhitenning).

```
> x1 <- arima.sim(list(order(1,0,0),ar=c(.5)),n=200)
> x2 <- arima.sim(list(order(2,0,0),ar=c(.5,.25)),n=200)
> indtest(x1,x2)        # Figure 6.1


> x12 <- prewhite(x1,x2)
> acf(matrix(c(x12$x,x12$y),ncol=2))
> acf(matrix(c(x1,x2),200,2))        # Figure 6.2
```

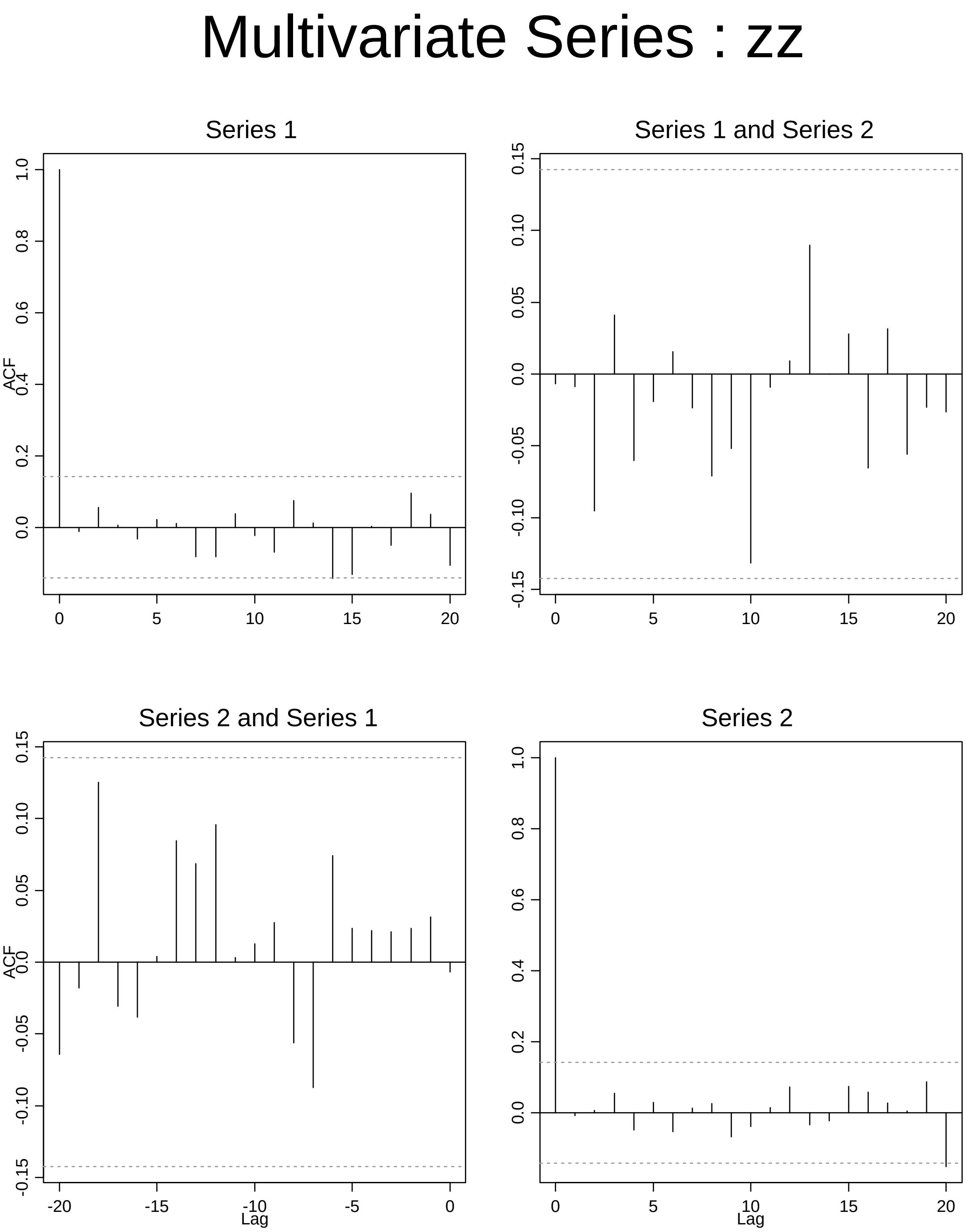


Fig 6.1 *Testing the independence of two series using* `indtest`.

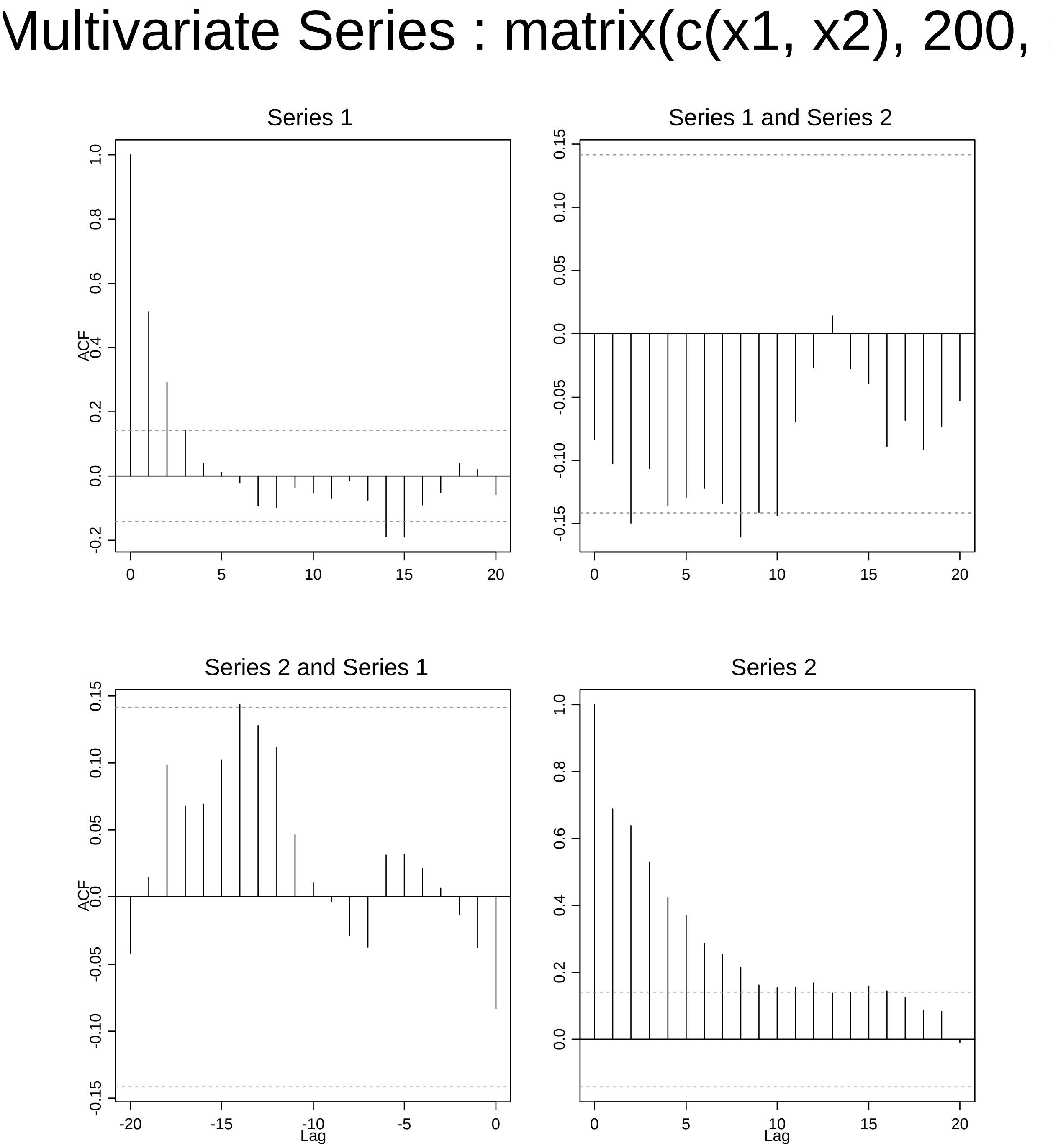


Fig 6.2 *Checking independence of two series using* `acf`.

If we test each of the $2M + 1$ cross-correlations (lags $-M$ to M), we should use a significance level of α^{2M+1} for each lag so that the overall significance level will be α. The function `indtest` returns the value α^{2M+1} for construnctiong confidence bands.

```
prewhite <- function(x,y)
# -----------------------------------------------------
#
# Use AR filters to prewhiten and align two time series
#
# -----------------------------------------------------
{
  n <- length(x)
  x <- x - mean(x)
  y <- y - mean(y)

  x.ar <- ar(x)
  y.ar <- ar(y)
  px <- x.ar$order
  py <- y.ar$order

  if (px == 0) xx <- x
  if (px > 0) {
    xx <- x.ar$resid
    xx <- xx[(px+1):n]
  }

  if (py == 0) yy <- y
  if (py > 0) {
    yy <- y.ar$resid
    yy <- yy[(py+1):n]
  }

  nx <- n - px
  ny <- n - py
  nn <- nx

  if (nx > ny) {
    nn <- ny
    n1 <- nx-ny
    n1 <- n1 + 1
    xx <- xx[n1:nx]
  }
```

```
  if (nx < ny) {
    nn <- nx
    n1 <- ny-nx
    n1 <- n1 + 1
    yy <- yy[n1:ny]
  }

  list(n=nn, x=xx, y=yy, ax=x.ar$ar, ay=y.ar$ar)
}

indtest <- function(x1, x2, nlag=20, alpha=.05)
# ------------------------------------------------
#
#  To test the independence of two time series
#  using cross-correlations
#
#  Argument: x1, x2, nlag, alpha
#
#  Value:    upper confidence limit
#
# ------------------------------------------------
{
  z <- prewhite(x1,x2)
  zz <- matrix(c(z$x,z$y),ncol=2)
  zacf <- acf(zz, lag.max=nlag)     # plot acf

  n <- length(z$x)      # for CI
  qnorm( (1+(1-alpha)^(1/(2*nlag+1)))/2) / (n^.5)   # upper CI
}
```

We apply the function `indtest` to test the independence of blood pressure data `bp` and the gas furnace data `gasfurn` [Box-Jenkins (1970)].

```
# Blood pressure

> attach(bp)
> indtest(dbp,sbp)        # Figure 6.3
[1] 0.1512914
> indtest(dbp,sbp,nlag=25)
[1] 0.1541933

# Gas furnace data
```

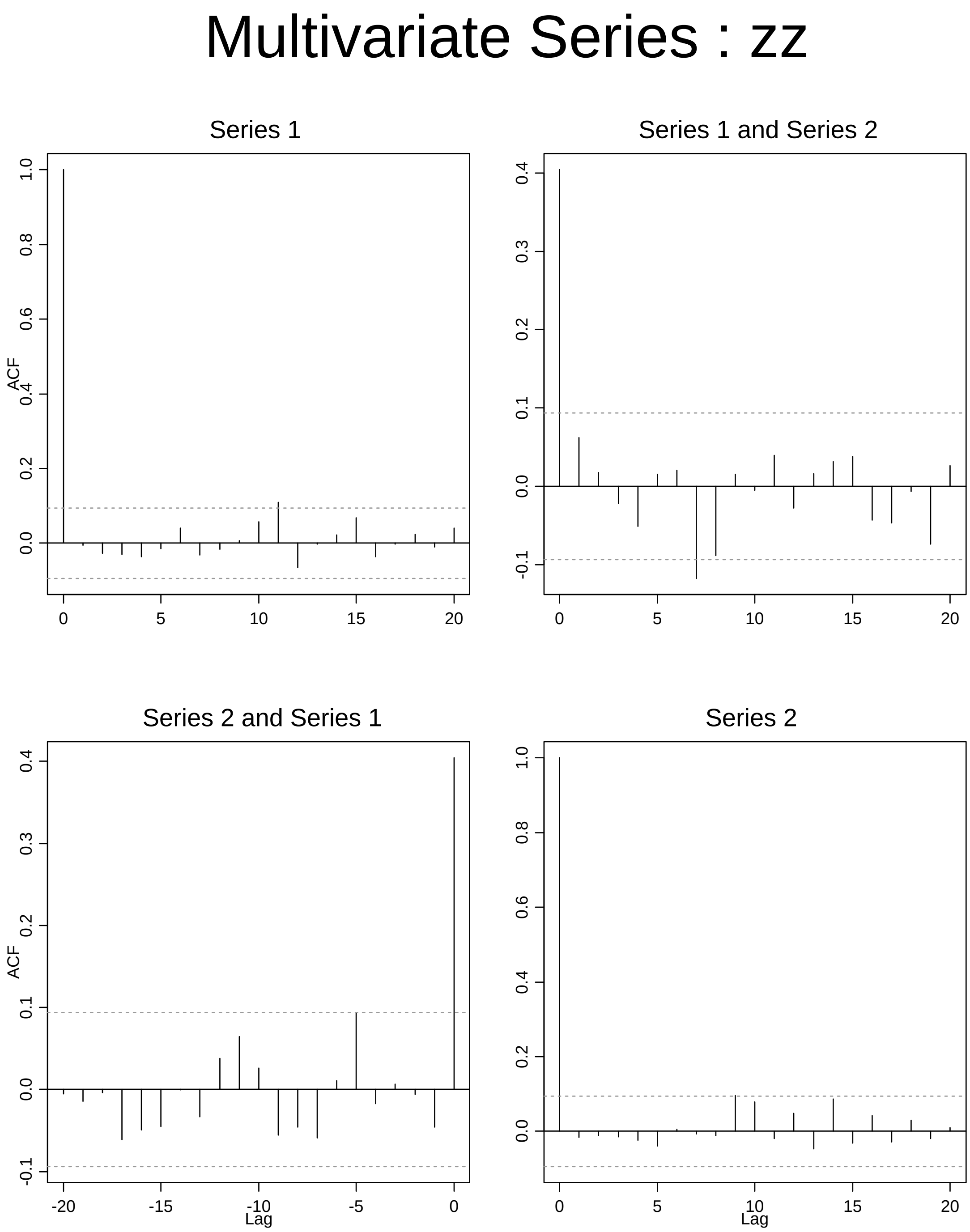


Fig 6.3 *Testing indepdendence of the blood pressure data via* `indtest`.

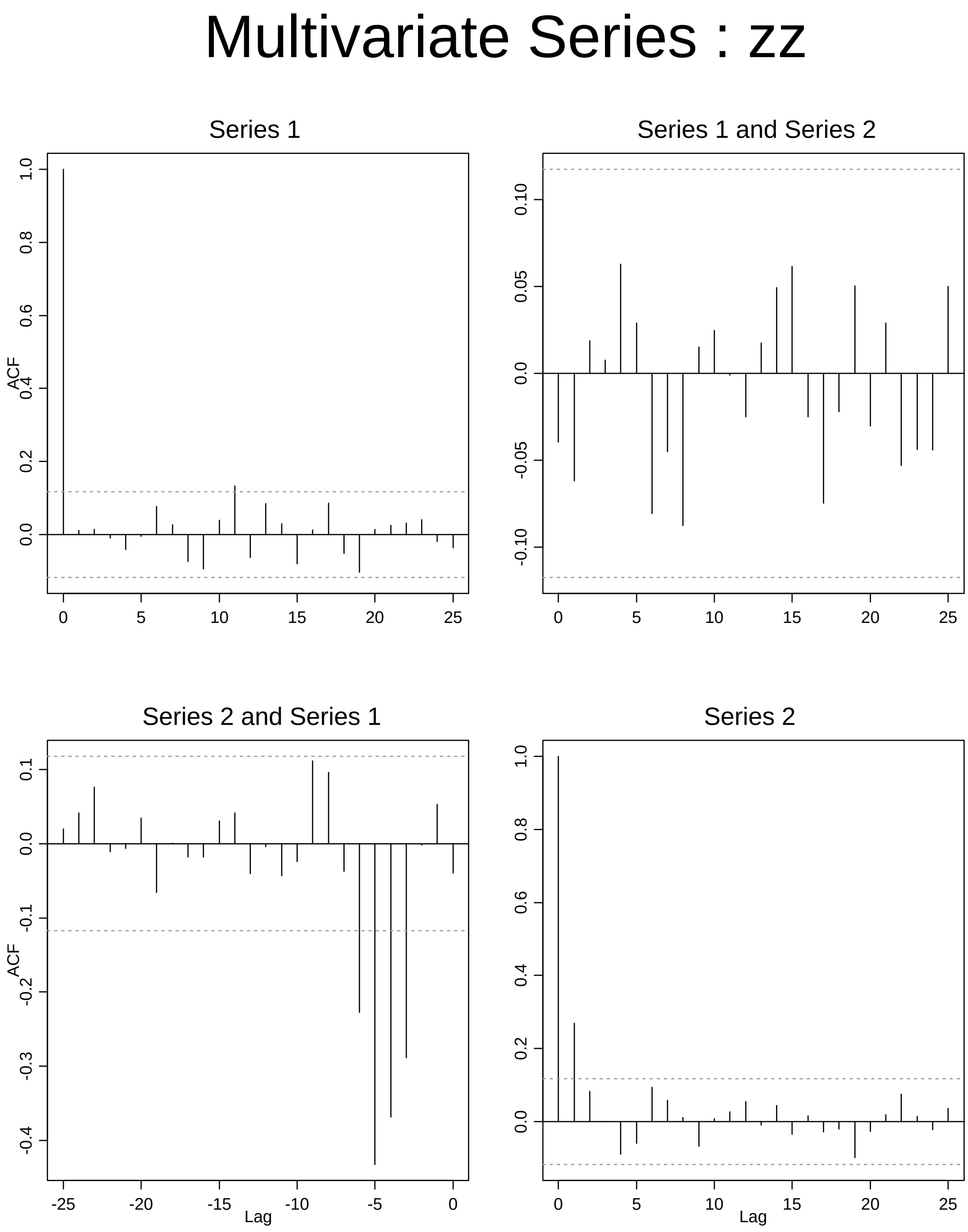


Fig 6.4 *Testing indepdendence of the gas furnace data via* `indtest`.

```
> attach(gasfurn)
> indtest(rate,co2,nlag=25)         # Figure 6.4
[1] 0.1541933

> ar(rate)$ar

, , 1
           [,1]
[1,]  1.9207350
[2,] -1.1852245
[3,]  0.1706730
[4,] -0.1188190
[5,]  0.2724795
[6,] -0.1116236

> ar(co2)$ar

, , 1
           [,1]
[1,]  1.8535718
[2,] -0.8389512
[3,] -0.3062926
[4,]  0.2599877
```

6.3 Cross-spectra

Let (x_t, y_t) denote a stationary bivariate time series with $E(x_t) = E(y_t) = 0$. From the spectral representations

$$x_{t+u} = \int e^{i(t+u)\lambda}\, dZ_x(\lambda), \quad y_t = \int e^{it\omega}\, dZ_y(\omega)$$

we write

$$\rho_{12}(u) = \operatorname{cov}(x_{t+u}, y_t) = E(x_{t+u}\overline{y_t}) = E \int\int e^{i(t+u)\lambda} e^{-it\omega}\, dZ_x(\lambda)\overline{dZ_y(\omega)}.$$

Now note that the left hand side does not depend on t. Thus

$$E\left(dZ_x(\lambda)\overline{dZ_y(\omega)}\right) = 0, \qquad \lambda \neq \omega.$$

That is, stationarity implies that different frequency components $dZ_x(\lambda)$ and $dZ_y(\omega)$ are uncorrelated. Consequently,

$$\rho_{12}(u) = \int e^{iu\lambda}\, E[dZ_x(\lambda)\overline{dZ_y(\lambda)}].$$

We denote $E[dZ_x(\lambda)\overline{dZ_y(\lambda)}]$ by $f_{12}(\lambda)\,d\lambda$ and call this the cross spectral or spectrum of x and y. Note that the cross spectra are complex-valued functions.

Since $\rho_{12}(u) = \int e^{iu\lambda} f_{12}(\lambda)\,d\lambda$, by inverser Fourier transformation,

$$f_{12}(\lambda) = \frac{1}{2\pi}\sum \rho_{12}(u)e^{-iu\lambda}.$$

Similarly,

$$f_{21}(\lambda) = \frac{1}{2\pi}\sum \rho_{21}(u)e^{-iu\lambda}.$$

It follows from $\rho_{12}(u) = \rho_{21}(-u)$ that $f_{12}(\lambda) = \overline{f_{21}(\lambda)}$. This can also be seen from

$$E[dZ_x(\lambda)\overline{dZ_y(\lambda)}] = \overline{E[\overline{dZ_x(\lambda)}dZ_y(\lambda)]}.$$

This indicates the matrix

$$\mathbf{f} = \begin{pmatrix} f_{11}(\lambda) & f_{12}(\lambda) \\ f_{21}(\lambda) & f_{22}(\lambda) \end{pmatrix}$$

is Hermitian; that is, $\mathbf{f}^* = \mathbf{f}$ where the (i,j)th entry of $\mathbf{f}^*$ is equal to the complex conjugate of the (j,i)th entry of $\mathbf{f}$. Moreover, $\mathbf{f}(\lambda) = \mathbf{f}^*(1-\lambda)$ for $\lambda \in [-\pi, \pi]$. Thus, the real part of f_{12} is symmetric about 0.5, while the imaginary part is antisymmetric about 0.5; that is, the values at frequencies λ and $1-\lambda$ are the same except they have opposite signs.

6.4 Co-spectra and quad-spectra

Write

$$f_{12}(\lambda) = c_{12}(\lambda) - iq_{12}(\lambda).$$

Then $c_{12}(\lambda)$ and $q_{12}(\lambda)$ are called the *co-spetrum* and *quad-spetrum*, respectively.

From $f_{12}(\lambda) = \bar{f}_{21}(\lambda)$, we have

$$c_{12}(\lambda) = c_{21}(\lambda) \qquad \text{and} \qquad q_{12}(\lambda) = -q_{21}(\lambda).$$

The cospectrum at frequency λ measures the linear relationship between the coefficients of the sinusoids, while the quadrature spectrum is for opposite trigonometric functions.

6.5 Amplitude (gain) and phase spectra

For a given frequency response function $A(\lambda)$, we write

$$f_{12}(\lambda) = A_{12}(\lambda)e^{i\phi_{12}(\lambda)},$$

where the amplitude and phase spectra are given by

$$A_{12}(\lambda) = |f_{12}(\lambda)| \qquad \text{and} \qquad \phi_{12}(\lambda) = \arctan\left(\frac{-q_{12}(\lambda)}{c_{12}(\lambda)}\right).$$

The phase sepctrum measures how out of phase the frequency components of the univariate series tend to be. The phase is a number between $-\pi$ and π, with $-\pi$ or π indicating that the components are perfectly out of phase; that is, they are like cosine and sine pair — one goes up the other goes down. For example, $\phi_{12}(1/12) = \pi$ (a half cycle) means that frequency components of period 12 tend to be perfectly out of phase — if the data are monthly, then the 12-month components are shifted 6 months from each other, with one leading (or lagging) the other by 6 months. On the other hand, $\phi_{12}(1/12) = \pi/2$ indicating that one series leads (or lags) the other by 3 months (a quarter of a cycle). Note the phase does not indicate which series leads the other.

6.6 An example

Consider the bivariate time series $y_t = ax_{t-\tau}$. Then $f_{12}(\lambda) = ae^{i\lambda\tau} f_{11}(\lambda)$. Hence

$$A_{12}(\lambda) = af_{11} \qquad \text{and} \qquad \phi(\lambda) = \lambda\tau.$$

Thus for this simple lag model, the gain of the frequency response function is a constant (= the coefficient of the filter) and the phase is a linear function with the slope being equal to the *lag* τ.

6.7 Linear relationships

Let $\{a_u\}$ denote a filter. Let y_t, x_t and e_t denote stationary time series so that

$$y_t = \sum a_{t-u} x_u + e_t.$$

Write

$$\begin{aligned} f_{11}(\lambda) &= E|Z_x(\lambda)|^2, & f_{22}(\lambda) &= E|Z_y(\lambda)|^2, \\ f_{ee}(\lambda) &= E|Z_e(\lambda)|^2, & A(\lambda) &= \sum a_u e^{-iu\lambda}. \end{aligned}$$

Then

$$\begin{aligned} \int e^{it\lambda}\, dZ_y(\lambda) &= \sum a_{t-u} \int e^{iu\lambda}\, dZ_x(\lambda) + \int e^{it\lambda}\, dZ_e(\lambda) \\ &= \int \left\{ \sum a_{t-u} e^{-i(t-u)\lambda} \right\} e^{it\lambda}\, dZ_x(\lambda) + \int e^{it\lambda}\, dZ_e(\lambda). \end{aligned}$$

Hence

$$dZ_y(\lambda) = A(\lambda)dZ_x(\lambda) + dZ_e(\lambda).$$

Consequently,

$$f_{21}(\lambda) = A(\lambda)f_{11}(\lambda)$$

and

$$\begin{aligned} f_{ee}(\lambda) &= f_{22}(\lambda) - |A(\lambda)|^2 f_{11}(\lambda) \\ &= f_{22}(\lambda)\left[1 - \frac{|f_{21}(\lambda)|^2}{f_{11}(\lambda)f_{22}(\lambda)}\right]. \end{aligned}$$

Define the *coherence of x_t with y_t at frequency λ* by

$$w_{21}(\lambda) = \frac{|f_{21}(\lambda)|}{\sqrt{f_{11}(\lambda)f_{22}(\lambda)}}$$

and

$$w_{12}(\lambda) = \frac{|f_{12}(\lambda)|}{f_{11}(\lambda)f_{22}(\lambda)}.$$

Note that $|w_{12}(\lambda)|^2 = |w_{21}(\lambda)|^2$ and coherence is a very important quantity for assessing the input-output linear relationship because of the following

$$\sigma_e^2 = \int_{-\pi}^{\pi} f_{ee}(\lambda)\,d\lambda = \int_{-\pi}^{\pi} f_{22}(\lambda)[1 - |w_{21}(\lambda)|^2]\,d\lambda.$$

Observe that if $|w_{21}(\lambda)|^2 \approx 1$ for $\lambda \in [-\pi, \pi]$, then $\sigma_e^2 \approx 0$. That is, the two series are linearly related.

Recall that if X and Y are mean zero rvs and if $EX^2 < \infty$, $EY^2 < \infty$. Suppose we are trying to approximate Y by aX, where a is a constant, by minimizing $E(Y - aX)^2$. Since

$$\begin{aligned} E(Y - aX)^2 &= E(Y^2) - 2aE(XY) + a^2E(X^2) \\ &= E(X^2)\left(a - \frac{E(XY)}{E(X^2)}\right)^2 + E(Y^2) - \frac{[E(XY)]^2}{E(X^2)}. \end{aligned}$$

We have $a = \hat{a} = E(XY)/E(X^2)$. Also,

$$E(Y - \hat{a}X)^2 = E(Y^2)\left(1 - \frac{[E(XY)]^2}{E(X^2)E(Y^2)}\right) = E(Y^2)[1 - \rho^2],$$

where $\rho = E(XY)/\sqrt{E(X^2)E(Y^2)}$ is the correlation coefficient of X and Y.

6.8 Estimation of cross-spectra

Recall the followings

$$I_{11}(\lambda) = \frac{1}{2\pi T}\left|\sum x_t e^{-it\lambda}\right|^2 \quad \text{and} \quad I_{22}(\lambda) = \frac{1}{2\pi T}\left|\sum y_t e^{-it\lambda}\right|^2.$$

Define the cross periodogram by

$$I_{12}(\lambda) = \frac{1}{2\pi T}\left(\sum x_t e^{-it\lambda}\right)\left(\sum y_t e^{it\lambda}\right).$$

Thus

$$I_{12}(\lambda) = I_{11}(\lambda) I_{22}(\lambda). \tag{6.1}$$

Now consider the smoothed version of the corresponding periodograms. Let

$$\widehat{\mathbf{f}}(\omega) = \sum_{v=-\infty}^{\infty} \lambda\left(\frac{v}{M}\right)\widehat{\mathbf{R}}(v)\exp(-2\pi i v\omega), \qquad \widehat{\mathbf{R}}(v) = \begin{pmatrix} R_{11}(v) & R_{12}(v) \\ R_{21}(v) & R_{22}(v) \end{pmatrix}.$$

The smoothed estimates can be obtainted via `spe` and `xspe`. It is sometimes helpful to write these estimates informally via

$$\hat{f}_{11}(\lambda) = \text{smooth}\{I_{11}(\lambda)\}, \qquad \hat{f}_{22}(\lambda) = \text{smooth}\{I_{22}(\lambda)\}$$

and

$$\hat{f}_{12}(\lambda) = \text{smooth}\{I_{12}(\lambda)\}.$$

The sample squared coherence is defined by

$$|\hat{w}_{12}(\lambda)|^2 = \frac{|\hat{f}_{12}(\lambda)|^2}{\hat{f}_{11}(\lambda)\hat{f}_{22}(\lambda)}.$$

Suppose $M \to \infty$ and $M/n \to 0$ as $n \to \infty$. Under appropriate conditions, $\widehat{\mathbf{f}}(\omega_j)$ and $\widehat{\mathbf{f}}(\omega_k)$ are asymptotically uncorrelated for $k \neq j$. Set

$$\widehat{\mathbf{Z}}(\omega) = (\widehat{f}_{11}(\omega), \widehat{f}_{22}(\omega), \widehat{c}_{12}(\omega), \widehat{q}_{12}(\omega))$$

and

$$\mathbf{Z}(\omega) = (f_{11}(\omega), f_{22}(\omega), c_{12}(\omega), q_{12}(\omega)).$$

Then $\sqrt{n/M}(\widehat{\mathbf{Z}}(\omega) - \mathbf{Z}(\omega))$ is asymptotically normal with mean zero and variance (as a function of ω) given by

$$\gamma \times \begin{pmatrix} f_{11}^2 & |f_{12}|^2 & f_{11}c_{12} & f_{11}q_{12} \\ |f_{12}|^2 & f_{22} & f_{22}c_{12} & f_{22}q_{12} \\ f_{11}c_{12} & f_{22}c_{12} & (f_{11}f_{22} + c_{12}^2 - q_{12}^2)/2 & c_{12}q_{12} \\ f_{11}q_{12} & f_{22}q_{12} & c_{12}q_{12} & (f_{11}f_{22} - c_{12}^2 + q_{12}^2)/2 \end{pmatrix},$$

where

$$\gamma = \int \lambda^2(u)\, du.$$

The above result can be utilized to derive the asymptotic distributions of other spectral quantities. In fact, let $\widehat{A}_{12}$ and $\widehat{\phi}$ denote the corresponding smoothed estimates of the amplitude and phase spectra. Then

1. $\sqrt{n/M}(\widehat{A}_{12}(\omega) - A_{12}(\omega))$ is asymptotically normal with mean zero and variance

$$\frac{1}{2}\gamma A_{12}^2(\omega)\Big(1 + \frac{1}{|w_{12}(\omega)|^2}\Big).$$

2. $\sqrt{n/M}(\widehat{\phi}_{12}(\omega) - \phi_{12}(\omega))$ is asymptotically normal with mean zero and variance

$$\frac{1}{2}\gamma A_{12}^2(\omega)\Big(\frac{1}{|w_{12}(\omega)|^2} - 1\Big).$$

3. $\sqrt{n/M}(\widehat{w}_{12}(\omega) - w_{12}(\omega))$ is asymptotically normal with mean zero and variance

$$\frac{\gamma}{2}(1 - |w_{12}(\omega)|^2).$$

The functions `crossp` and `coher` are implemented using these results.

6.9 Testing for zero coherence

Under the null $H_0 : w_{12}(\lambda) = 0$, we have

$$\frac{\hat{w}_{12}^2(\lambda)}{1 - \hat{w}_{12}^2(\lambda)} \times \frac{2(L-1)}{2} \sim F_{2,2(L-1)}.$$

Thus the strength of the linear relationship can be assessed by comparing $|\hat{w}_{12}(\lambda)|^2$ with the constant

$$K_\alpha = \frac{C_\alpha}{1 + C_\alpha}, \qquad C_\alpha = \frac{2F_{2,2(L-1)}(\alpha)}{2(L-1)}.$$

The function `cohtest` is implemented using these results. Note that it is necessary to smooth the spectra in estimating coherence, since the unsmoothed version is equal to 1 [(6.1)].

6.9.1 S functions for computing estimates of spectra

The function xspe computes the cross-spectrum of a bivariate series. It returns the real and imaginary part of the cross-spectrum, and a critical value for constructing confidence intervals.

```
xspe <- function(x1, x2, ioptw, M, alpha=.05)
#------------------------------------------------------------
#
#   Function to find  cross-spectra using scale parameter M,
#   window number ioptw:
#
#   1   Truncated
#   2   Bartlett
#   3   Tukey
#   4   Parzen
#   5   Bohman
#   6   Daniell
#   7   Bartlett-Priestley
#   8   Parzen-Cogburn-Davis
#
#   The spectral estimator and the constant c used in confidence
#   intervals are returned in a list.
#
#   TIMESLAB (p.335)
#
#------------------------------------------------------------
{
   if (length(x1) != length(x2))
     stop("Series with unequal length!")
   n <- length(x1)

   if(M >= n || ioptw < 1 || ioptw > 8
      || alpha <= 0. || alpha >= 1.)
   stop("Illegal Input to xspe()")

 x <- matrix(c(x1,x2),n,2)
   x.acf <- acf(x,lag.max=n-1,type="cov",plot=F)
 c12 <- x.acf$acf[,1,2]
 c21 <- x.acf$acf[,2,1]

   z1 <- rep(0,n)
```

```
if(ioptw == 1) z1[1:(M+1)] <- rep(1, M+1)

else if(ioptw == 2)  z1[1:(M+1)] <- 1 - (c(0:M) / M)

else if(ioptw == 3)  z1[1:(M+1)] <- .54+.46*cos(pi*c(0:M)/M)

else if(ioptw == 4) {
   u  <- c(1:M)/M
   u1 <- u[u<=.5]
   u2 <- u[u>.5]
   z1[1:(M+1)]  <- c(1,1-6*u1^2+6*u1^3,2*(1-u2)^3)
}

else if(ioptw == 5) {
   u   <- c(0:M) / M
   piu <- pi * u
   z1[1:(M+1)]  <- (1 - u) * cos(piu) + sin(piu) / pi
}

else if(ioptw == 6) {
   piu <- pi * (c(1:(n-1)) / M)
   z1[1:n] <- c(1,sin(piu)/piu)
}

else if(ioptw == 7) {
   piu <- pi * (c(1:(n-1)) / M)
   z1[1:n] <- c(1,3*((sin(piu)/piu) - cos(piu)) / (piu*piu))
}

else if(ioptw == 8) {
   u <- c(0:(n-1))/M
   z1[1:n] <- 1 /(1+u^4)
}

z <- fft(z1[1:n]*c12[1:n])+Conj(fft(c(0,z1[2:n]*c21[2:n])))
z <- z[1:((n / 2) + 1)]

ilam2 <- c(2,2./3.,.795,.539,.586,1,1.2,1.66)
fac <- qnorm(1-(alpha/2)) * sqrt(M*ilam2[ioptw]/n)

return(list(re=Re(z), im=Im(z), c=fac))
```

```
}
```

This functions spe and xspe are used in crossp for computing the co- and quad-spectra and confidence intervals.

```
crossp <- function(x1,x2,ioptw,M, ci=T)
#-------------------------------------------------------
#
#  Function to estimate, plot the co and quad spectra
#  of bivariate series x1 and x2 using window ioptw and M
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#
#  Value: None
#
# TIMESLAB (p.338)
#-------------------------------------------------------
{
  f11 <- spe(x1,ioptw,M)$f
  f22 <- spe(x2,ioptw,M)$f
  f12 <- xspe(x1,x2,ioptw,M)
  crv <- f12$c
  c12 <- f12$re
  q12 <- -f12$im

# Compute the avar

  v1 <- f11*f22
  v2 <- c12^2-q12^2
  sd <- matrix(c(sqrt((v1+v2)/2),sqrt((v1-v2)/2)),ncol=2)

  f <- matrix(c(c12,q12),ncol=2)
  n <- length(f[,1])

  par(mfrow=c(1,2))
  for (j in (1:2))
  {
      s <- f[,j]
      se <- sd[,j]*crv
      cil <- s - se
```

```
      ciu <- s + se
      if (j==1)
        {
          y.lab <- "Co-spectrum"
        }
      else
        {
          y.lab <- "Quad-spectrum"
        }
      if (ci) ci.plot(s, cil, ciu, y.lab = y.lab)
        else sp.plot(s,y.lab=y.lab)
  }
  par(mfrow=c(1,1))
  invisible()
}

sp.plot <- function(f, y.lab="Spectrum", p.type="l")
# -----------------------------------------------------
#
#   Function for plotting spectral quantities
#
# -----------------------------------------------------
{
   q <- length(f)
   plot(c(0:(q-1))/(2*(q-1)),f,xlab="frequency",
        ylab=y.lab,type=p.type)
}

ci.plot <- function(f,fl,fu,y.lab="Spectrum")
# -----------------------------------------------------
#
#   Function for plotting spectral quantities and CI
#
# -----------------------------------------------------
{
  n <- length(f)
  plot(c(0,.5), c(min(fl), max(fu)), xlab = "frequency",
      ylab = y.lab, type = "n")
  lines(c(0:(n - 1))/(2 * (n - 1)), f)
  lines(c(0:(n - 1))/(2 * (n - 1)), fl, lty=2)
  lines(c(0:(n - 1))/(2 * (n - 1)), fu, lty=2)
}
```

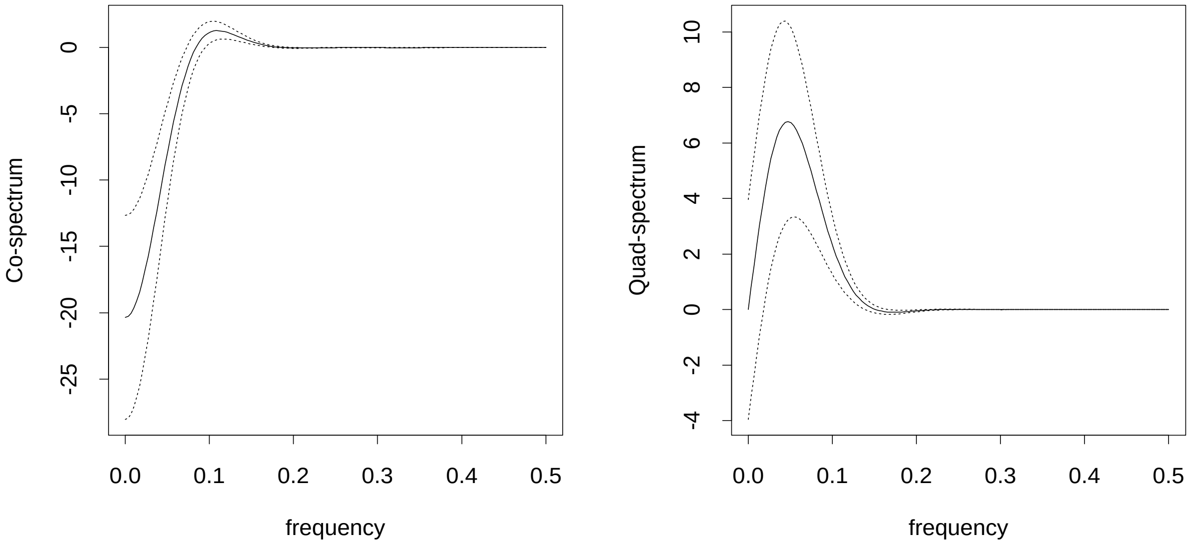


Fig 6.5 `crossp` *of the gas furnace data.*

Example

```
x <- matrix(scan("gasfurn"),ncol=2,byrow=T)
x1 <- x[,1]
x2 <- x[,2]
par(mfrow=c(4,4))
for (i in 1:8) crossp(x1,x2,ioptw=i,M=10)        # not shown
for (i in 1:8) crossp(x1,x2,ioptw=4,M=i*5)       # not shown

crossp(x1,x2,ioptw=4,M=15)      # Fig 6.5
crossp(x1,x2,ioptw=6,M=15)      # Fig 6.6
crossp(x1,x2,ioptw=4,M=30)      # Fig 6.7
```

6.9.2 S functions for computing coherence and phase

The function `coher` estimate and plot the cross-spectra, the coherence and the phase spectra. Confidence intervals are constructed based on the theory discussed previously.

```
coher <- function(x1,x2,ioptw,M)
#-----------------------------------------------------------
#
#  Function to estimate the spectra, coherence and phase
```

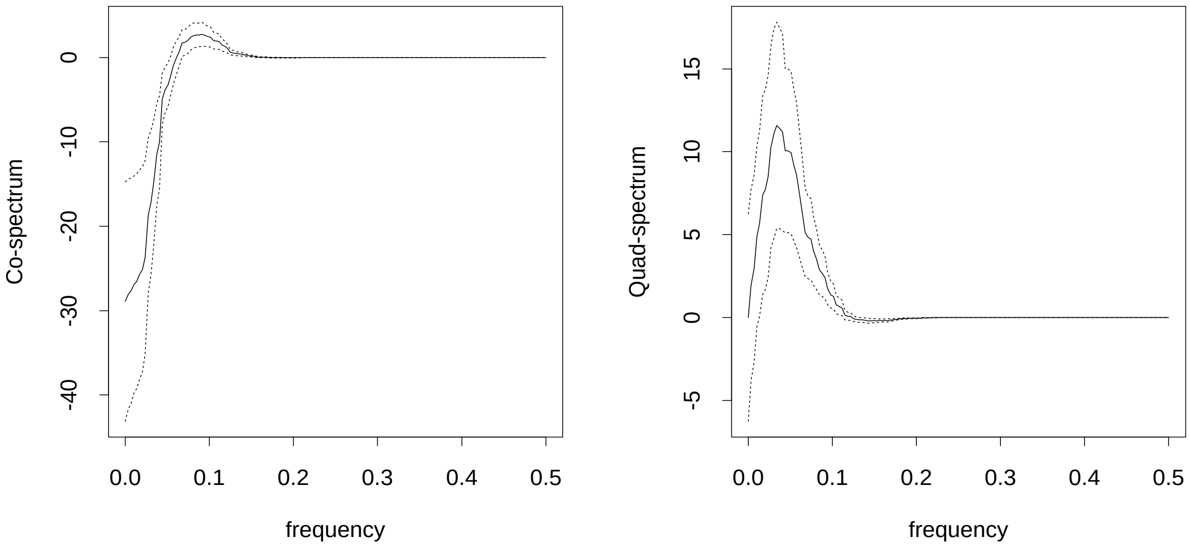


Fig 6.6 `crossp` *of the gas furnace data with a different* `ioptw`*.*

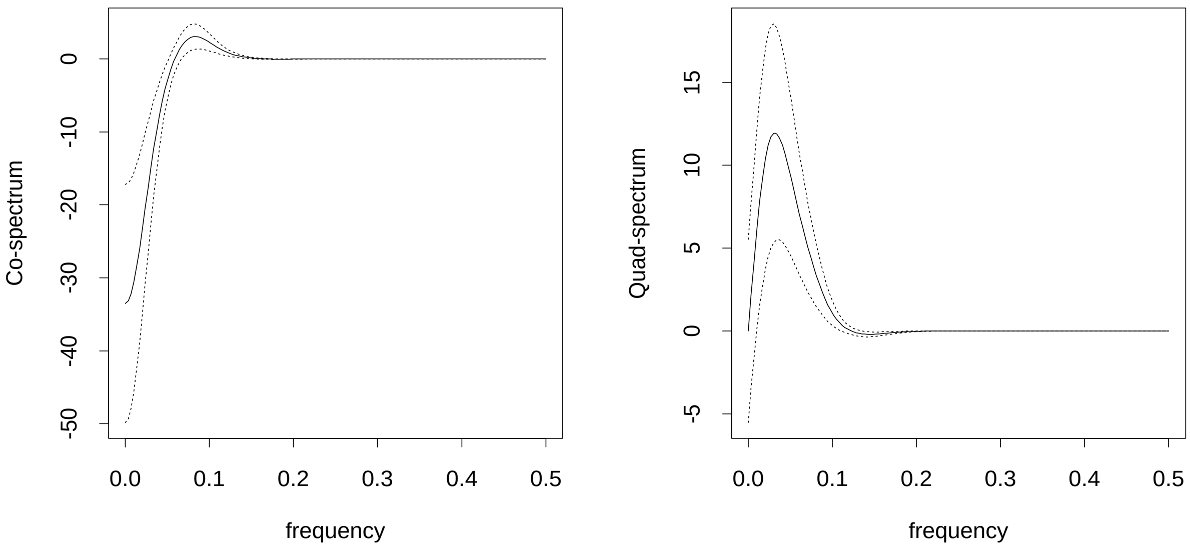


Fig 6.7 `crossp` *of the gas furnace data, larger* `M`*.*

```
#  of bivariate series x1 and x2 using window ioptw and M
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#
#  Value: None
#
#-----------------------------------------------------------
{
 f11 <- spe(x1,ioptw,M)$f
 f22 <- spe(x2,ioptw,M)$f
 f12 <- xspe(x1,x2,ioptw,M)
 c12 <- f12$re
 q12 <- f12$im

 amp <- polar(c12,q12)$amp
 phi <- polar(c12,q12)$phase
 coh <- amp^2/(f11*f22)

 par(mfrow=c(2,2))

 sp.plot(f11, y.lab = "Spectrum f11")
 sp.plot(f22, y.lab = "Spectrum f22")

 ilam2 <- c(2,2./3.,.795,.539,.586,1,1.2,1.66)

 w12 <- sqrt(coh)
 se <- sqrt(abs(1-coh)*M*ilam2[ioptw]/(2*length(coh)))
 lo <- w12-2*se
 up <- w12+2*se
 ci.plot(w12, lo, up, y.lab="Coherence")

 se <- sqrt((amp^2*abs(1-coh)/coh)*M*ilam2[ioptw]/(2*length(coh)))
 lo <- phi-2*se
 up <- phi+2*se
 ci.plot(phi, lo, up, y.lab="Phase")

 par(mfrow=c(1,1))
 invisible()
}
```

```
polar <- function(zr,zi)
#---------------------------------------------------------
#
# Function for converting to polar representation
#
#---------------------------------------------------------
{
 return(list(amp=sqrt(zr^2+zi^2), phase=atan(zi,zr)))
}
```

Example

```
# Gas furnace data
attach(gasfurn)
coher(rate,co2,ioptw=4,M=15)       # Fig 6.8
coher(rate,co2,ioptw=6,M=15)       # Fig 6.9
coher(rate,co2,ioptw=4,M=30)       # Fig 6.10

# Blood pressure

attach(bp)
crossp(dbp,sbp,ioptw=4,M=15)      # Fig 6.11
coher(dbp,sbp,ioptw=4,M=15)       # Fig 6.12
```

6.9.3 S functions for testing zero coherence

The following function performs the test for zero coherence.

```
cohtest <- function(x1,x2,ioptw,M)
#---------------------------------------------------------
#
#  Function to estimate the spectra, coherence and phase
#  of bivariate series x1 and x2 using window ioptw and M
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#
#  Value: None
#
```

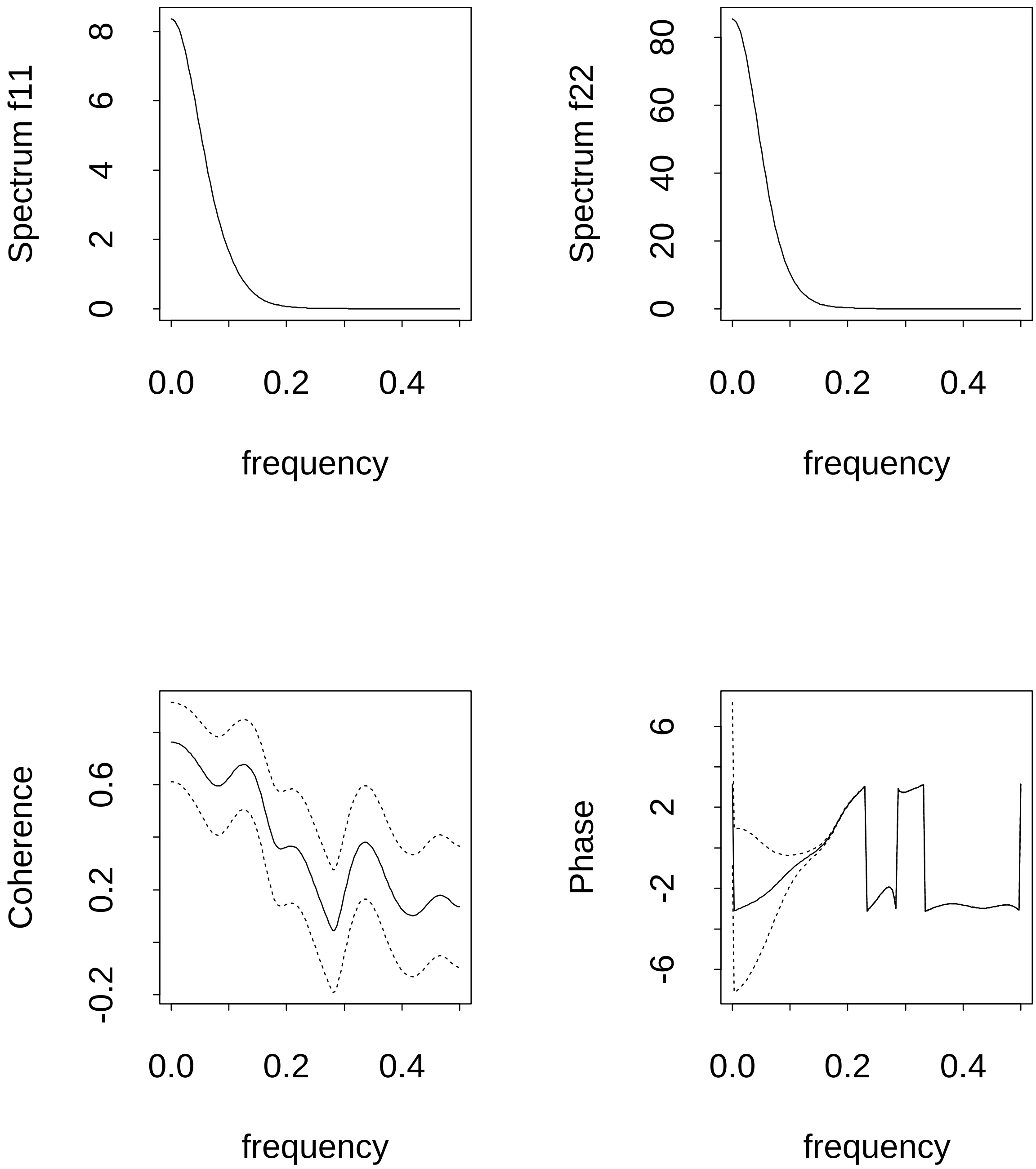


Fig 6.8 *Estimating the coherence of the gas furnace data via* `coher`*.*

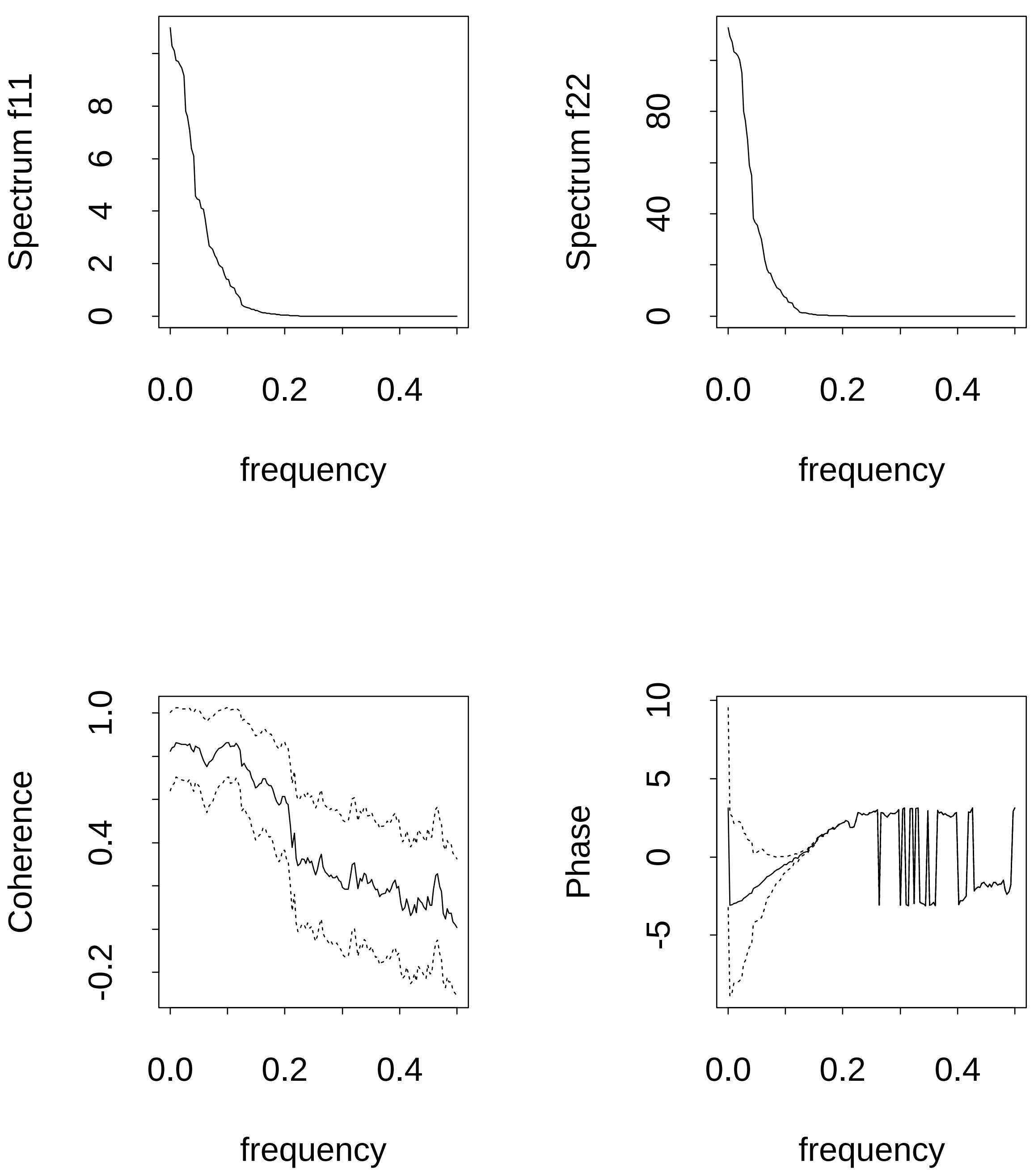


Fig 6.9 *Continue* `coher` *of the gas furnace data using ioptw=6.*

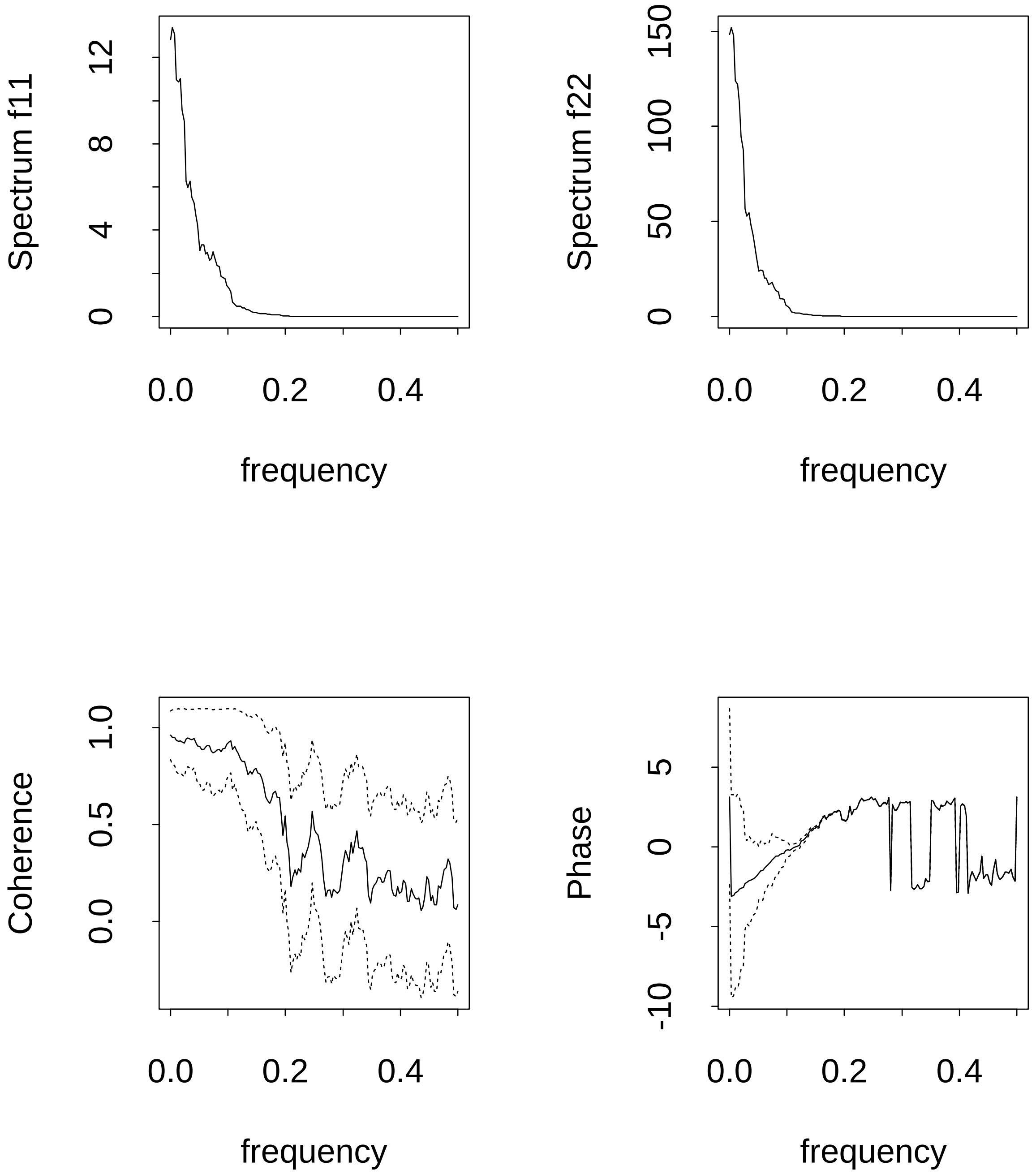


Fig 6.10 `coher` *of* `gasfurn` *with ioptw=6 and M=30.*

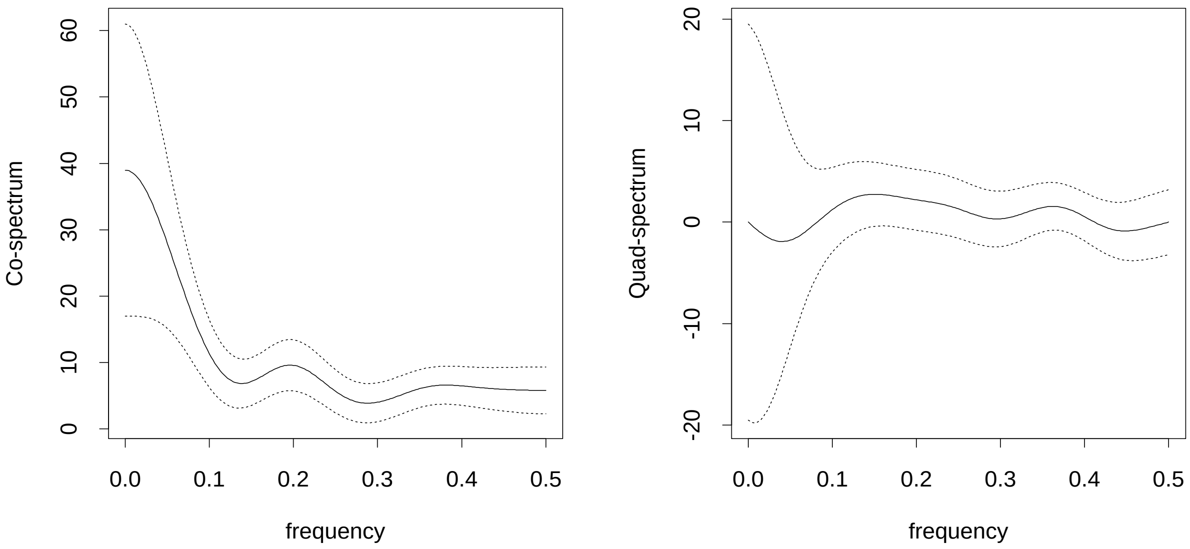


Fig 6.11 *Apply* `crossp` *to the blood pressure data.*

```
#-----------------------------------------------------------
{
  f11 <- spe(x1,ioptw,M)$f
  f22 <- spe(x2,ioptw,M)$f
  f12 <- xspe(x1,x2,ioptw,M)
  c12 <- f12$re
  q12 <- f12$im

  amp <- polar(c12,q12)$amp
  phi <- polar(c12,q12)$phase
  coh <- amp^2/(f11*f22)

  n <- length(f11)
  par(mfrow=c(2,2))

  sp.plot(f11, y.lab = "Spectrum f11")
  sp.plot(f22, y.lab = "Spectrum f22")

  ilam2 <- c(2,2./3.,.795,.539,.586,1,1.2,1.66)

  const <- (n - M*ilam2[ioptw])/(M*ilam2[ioptw])
  fstat <- const*coh/(1-coh)
```

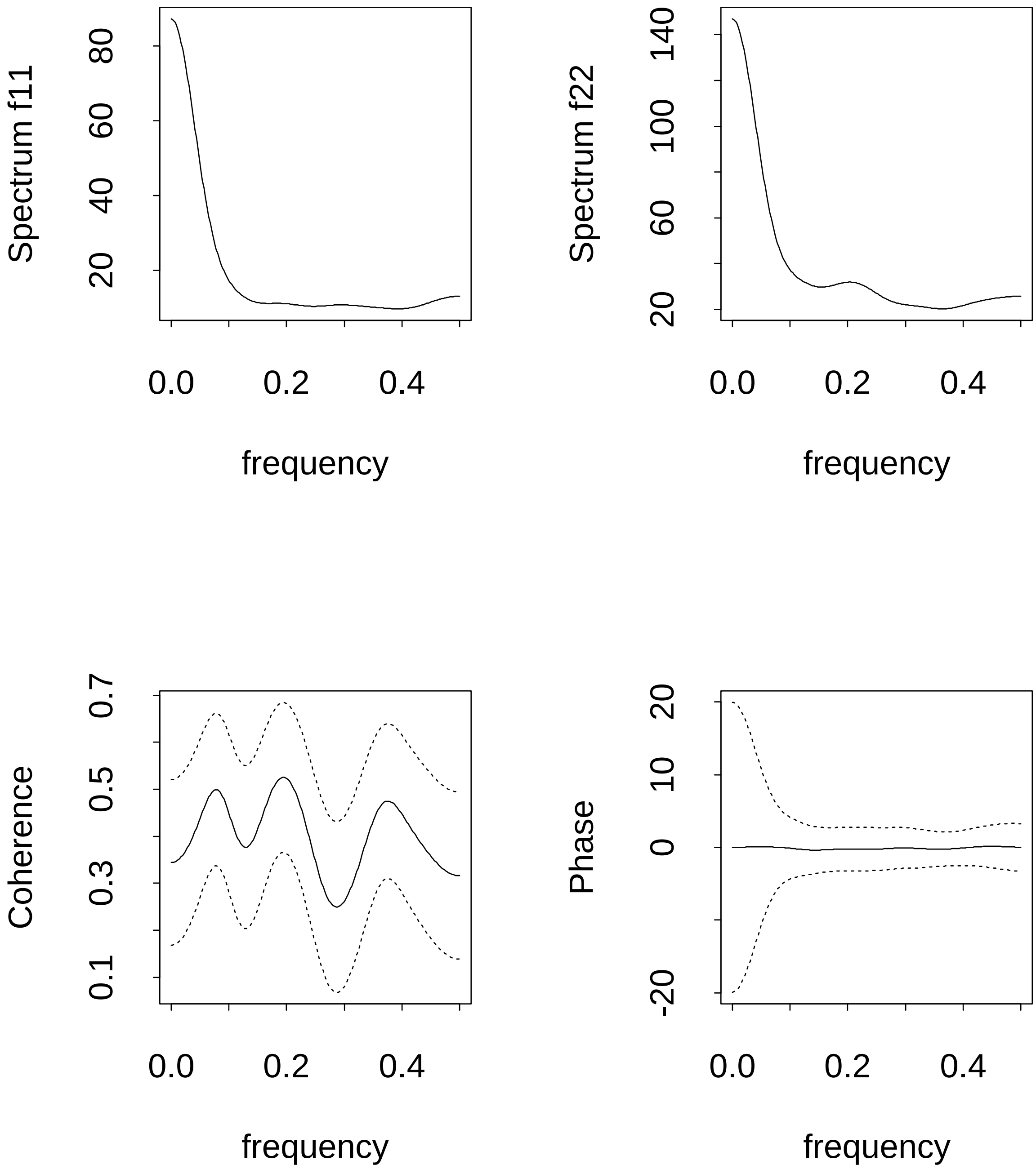


Fig 6.12 *Estimating the coherence and phase of the blood pressure data using* `coher`.

```
  nc <- 2*const
  ft <- qf(.95,2,nc)
  sp.plot(fstat, y.lab = "F-stat")
  lines(c(0:(n - 1))/(2 * (n - 1)), rep(ft,n),lty=2)

  sp.plot(phi, y.lab = "Phase")
  par(mfrow=c(1,1))
  invisible()
}
```

Examples

```
# Gas furnace data
# attach(gasfurn)
cohtest (rate,co2,ioptw=4,M=15)       # Fig 6.13

# Blood pressure
# attach(bp)
cohtest(dbp,sbp,ioptw=4,M=15)       # Fig 6.14
```

The results indicate that bothe series are not independent.

6.10 Estimation of the transfer function

For the transfer function model $y_t = \sum_u a_u x_{t-u} + w_t$, we recall that $A(\lambda) = \sum_u a_u \exp(-iu\lambda)$ and

$$f_{21}(\lambda) = A(\lambda) f_{11}(\lambda).$$

Thus $A(\lambda) = f_{21}(\lambda)/f_{11}(\lambda)$ and

$$a_u = \frac{1}{2\pi} \int \frac{f_{21}(\lambda)}{f_{11}(\lambda)} \exp(iu\lambda)\, d\lambda.$$

Let $\hat{f}_{21}(\lambda)$ and $\hat{f}_{11}(\lambda)$ denote smoothed estimates of $f_{21}(\lambda)$ and $f_{11}(\lambda)$. Then a natural estimate of a_u is given by

$$\hat{a}_u = \frac{1}{2\pi} \int \frac{\hat{f}_{21}(\lambda)}{\hat{f}_{11}(\lambda)} \exp(iu\lambda)\, d\lambda \doteq \frac{1}{K} \sum_k \frac{\hat{f}_{21}(\omega_k)}{\hat{f}_{11}(\omega_k)} \exp(iu\omega_k),$$

where $\omega_k = 2\pi k/K$, $k = 0, 1, \ldots, K$. Observe that a_u can be easily computed as the inverse Fourier transformation of $\hat{f}_{21}(\omega_k)/\hat{f}_{11}(\omega_k)$. This is carried out in the function `estfilt`. Note that in some applications, the index u can take on negative values as in lead-lag modeling. In this case, we have $\hat{a}_{-u} = \hat{a}_{K-u}$, $u \geq 0$.

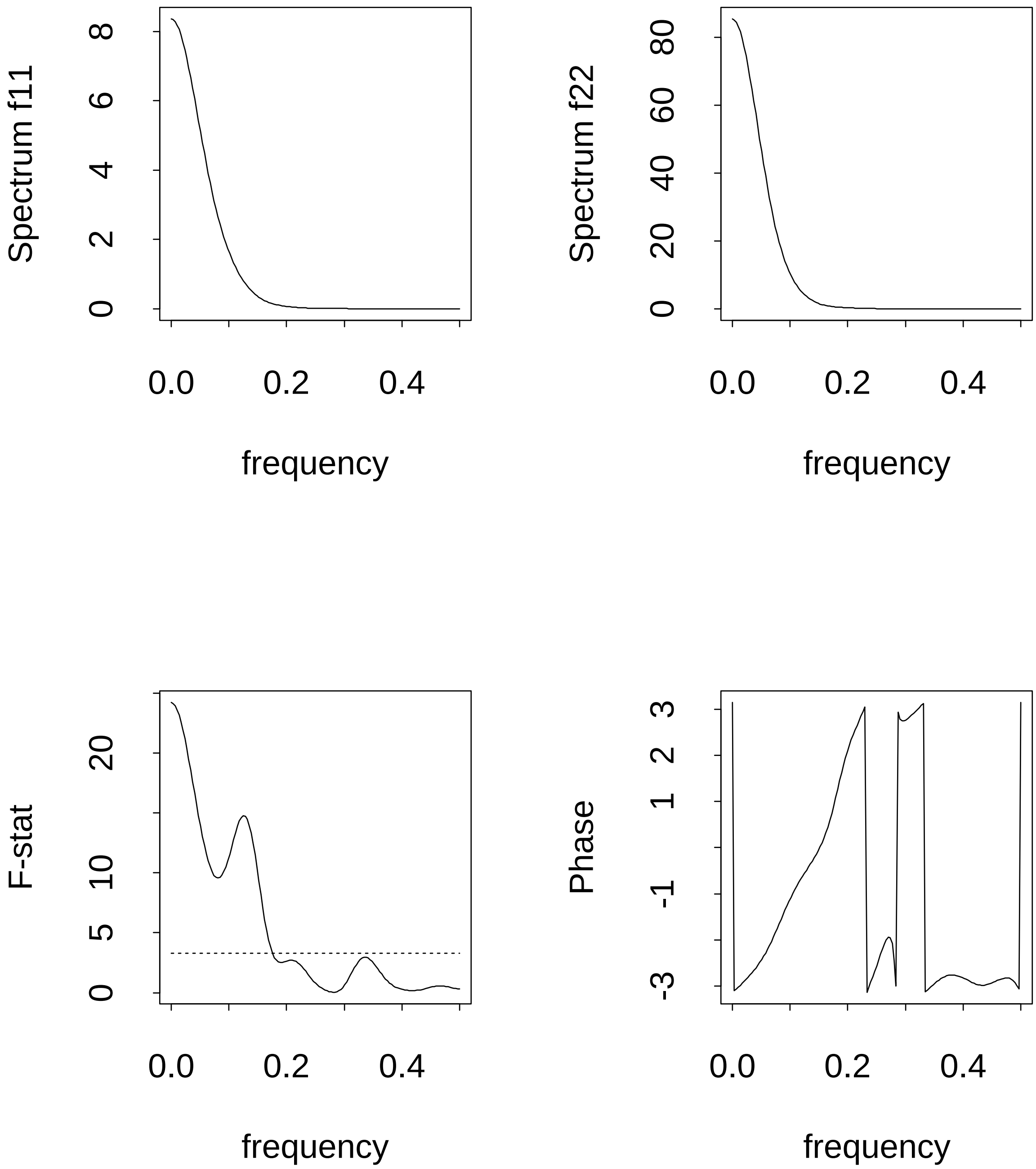


Fig 6.13 *Apply* `cohtest` *to test the linear relationship in the gas furnace data.*

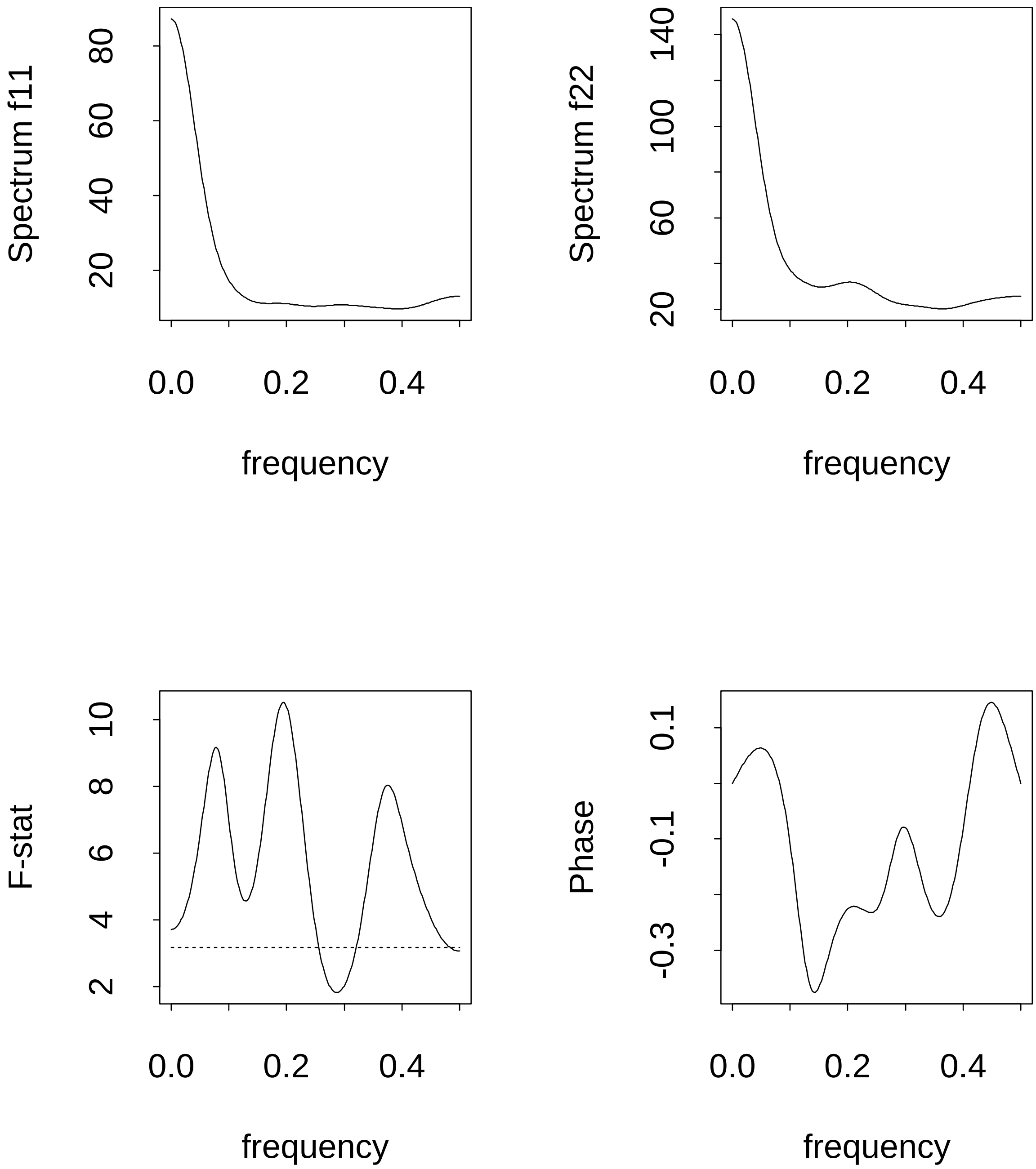


Fig 6.14 *Apply* `cohtest` *to test the linear relationship in the blood pressure data.*

```
estfilt <- function(x1,x2,ioptw,M,kn)
#----------------------------------------------------------
#
#  Function to estimate the best filter coefficients
#  of bivariate series x1 and x2 using window ioptw and M.
#  It uses the inverse Fourier transform on A=f21/f11.
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#    kn+1:        number of beta's
#
#  Value: beta[1:(kn+1)] (coefficients of the transfer function)
#
#  See Theorem 4.1.6 of TIMESLAB (p.321)
#
#----------------------------------------------------------
{
  f11 <- spe(x1,ioptw,M)$f
  f22 <- spe(x2,ioptw,M)$f
  f21 <- xspe(x2,x1,ioptw,M)
  c21 <- f21$re
  q21 <- f21$im

  c211 <- c21/f11
  q211 <- q21/f11

  q <- length(c21)
  c211 <- c(c211,c211[(q-1):2])      # extend to [0,1] via sym
  q211 <- c(q211,-q211[(q-1):2])     # extend to [0,1] via antisym

  z <- c211 + q211*1i                # Form complex array for FFT

  beta <- Re(fft(z,inv=T))/(2*q-2)
  beta[1:(kn+1)]
}
```

6.11 Alignment

In practice, it appears to be the case that cross-spectra vary more substantially than the power spectra. Consider the bivariate process

$$y_t = x_{t-v}, \qquad t = 1, \ldots, n.$$

Then

$$I_{12}(\lambda) = \frac{1}{2\pi} \sum C_{12}(u) e^{-iu\lambda},$$

where

$$C_{xy}(u) = \frac{1}{n} \sum x_{t+u} y_t = \frac{1}{n} \sum x_{t+u} x_{t-v} = C_{11}(u+v).$$

Thus

$$I_{12}(\lambda) \approx e^{iv\lambda} I_{11}(\lambda).$$

Now if v is large, then averaging the cross periodogram $I_{12}(\lambda)$ is equivalent to smooth the highly varying component $e^{iv\lambda}$, the result is that they cancel each other out. Hence

$$I_{12}(\lambda) \approx 0,$$

or equivalently,

$$w_{12}(\lambda) \approx 0.$$

But we know that y and x are perfectly related.

One remedy to this problem is to align the two series using the function `prewhite`, then estimate the transfer function via `estfilt`. Specifically, we prewhite and align (via `prewhite`) the series so that $\alpha(B)y$ and $\beta(B)x$ are nearly white. Then invoke the function `estfilt` to estimate the transfer function between $\alpha(B)y$ and $\beta(B)x$, yielding

$$\alpha(B)y = b(B)\beta(B)x.$$

Then the estimates of the impulse responses are given by

$$\hat{a}(B) = b(B)\beta(B)/\alpha(B).$$

This is carried out in the function `filtest`, which uses `multpoly` and `invpoly` for mulitplying and inverting polynomials.

```
filtest <- function(x1,x2,ioptw,M,kn)
#------------------------------------------------------
#
#  Function to estimate the transfer function of a filter
#  of bivariate series x1 and x2 using estfilt and prewhitening
#
#  Arguments:
```

```
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#    kn+1:        number of beta's
#
#  Value: coef  (coefficients of the transfer function)
#
#  See TIMESLAB (p.344)
#
#-------------------------------------------------------
{
  z <- prewhite(x1,x2)
  x1 <- z$x
  x2 <- z$y
  ax <- -z$ax
  ay <- -z$ay

  beta <- estfilt(x1,x2,ioptw,M,kn)

  beta0 <- beta[1]
  beta <- beta/beta0

  kn1 <- kn+1
  gama <- multpoly(beta[2:kn1],ax)
  betay <- divpoly(a=ay,b=c(),40)
  gama <- multpoly(betay,gama)
  gama <- gama*beta0

  list(coef=gama[1:kn], b0=beta0)
}
```

Examples

We estimate a transfer model for the gas furnace and blood pressure data.

```
# Gas Furnace data
x <- matrix(scan("gasfurn"),ncol=2,byrow=T)
x1 <- x[,1]
x2 <- x[,2]
> filtest(x1,x2,ioptw=4,M=34,kn=10)
$coef:
 [1]  0.08561873 -0.03720568 -0.50389333 -0.60908855 -0.72936814
 [6] -0.43054377 -0.29498369 -0.09430614 -0.03606797 -0.06124109
```

```
$b0:
[1] -0.06401081       # Agree with timeslab (p.345)
```

The above result yields $\beta_0 = -0.06401081$, $\beta_1 = 0.08561873$, $\beta_2 = -0.03720568$, ... These results agree with the model found by Box-Jenkins (1970, p.399):

$$(1 - .57B)x_2(t) = -(.53 + .37B + .51B^2)x_1(t-3).$$

In fact, expanding the rational function using `multpoly` and `invpoly`, the coefficients are given by (exercise)

```
0.00  0.00  -.53  -.67  -.89  -.51  -.29  -.16 ...
```

```
# Blood pressure

> x <- matrix(scan("bp"),byrow=T,ncol=2)
> x1 <- x[,1]
> x2 <- x[,2]
> filtest(x1,x2,ioptw=4,M=36,kn=10)
$coef:
 [1] -0.061356561 -0.026158000 -0.006566397 -0.012613797
 [5]  0.168476327 -0.016126354 -0.053620315 -0.062732866
 [9] -0.068074833  0.017976434

$b0:
[1] 0.6158987
```

The above result indicates that the relationship between DBP and SBP is given by `sbp - mean(sbp) = 0.6158987 (dbp - mean(dbp))`.

Simulation

The function `exfilt` carries out a simulation for examing the performance of `filtest`. The transfer function model is given by

$$y_t = \beta(B)x_t + w_t, \qquad x_t \sim AR(p), \qquad w_t \sim N(0, .5^2).$$

The parameters for the transfer function and the AR are given respectively by $\beta(z) = 0.7 + 0.1z$ and $\alpha(z) = 0.5z + 0.25z^2$ with $x = \alpha(B)\epsilon$. That is,

$$y_t = .7x_t + .1x_{t-1} + w_t, \qquad x_t - .5x_{t-1} - .25x_{t-2} = \epsilon_t, \qquad \epsilon_t \sim N(0, 1).$$

```
exfilt <- function(nsim=10, n=400, beta=c(.7,.1),
            alpha=c(.5,.25), sig=.5, ioptw=4, M=36, kn=7)
# -----------------------------------------------------
#
#  Test filtest using simulated series
#
# -----------------------------------------------------
{
  bhat <- rep(0,1+kn)
  k <- length(beta)

  for (j in 1:nsim)
  {
    z <- arima.sim(list(order(length(alpha),0,0),ar=alpha),n=n+k)
    fz <- filter(z,beta,sides=1)    # fz = beta(B)*z(t)
    fz <- fz[(1+k):(n+k)]           # the first k-1 are NA's
    fz <- fz + sig*rnorm(n)
    z <- z[(1+k):(n+k)]             # align them
    z <- filtest(z,fz,ioptw,M,kn)
    bhat <- ((j-1)*bhat+c(z$b0,z$coef))/j
  }
  bhat
}
```

The mean of the estimates of β in 10 simulations with sample size equal to 400 are given in the following lists.

```
# Examples
> exfilt()
[1]  0.7004292491  0.0958350490  0.0056724285  0.0098596371
[5]  0.0003233507 -0.0203610886  0.0173464051 -0.0002426299

> exfilt(beta=c(0,0,.5))
[1] -0.0070259864 -0.0004812013  0.4893230615  0.0080252134
[5] -0.0073506045 -0.0150583004  0.0028846726  0.0056862068
```

The second example corresponds to the model $y_t = .5x_{t-2} + w_t$ whose phase spectrum effectively capture the lead-lag relationship as we have discussed before.

```
ex <- function(n=400, beta=c(0,0,.5), alpha=c(.5,.25),
               sig=0, ioptw=4, M=36)
# -----------------------------------------------------
#
```

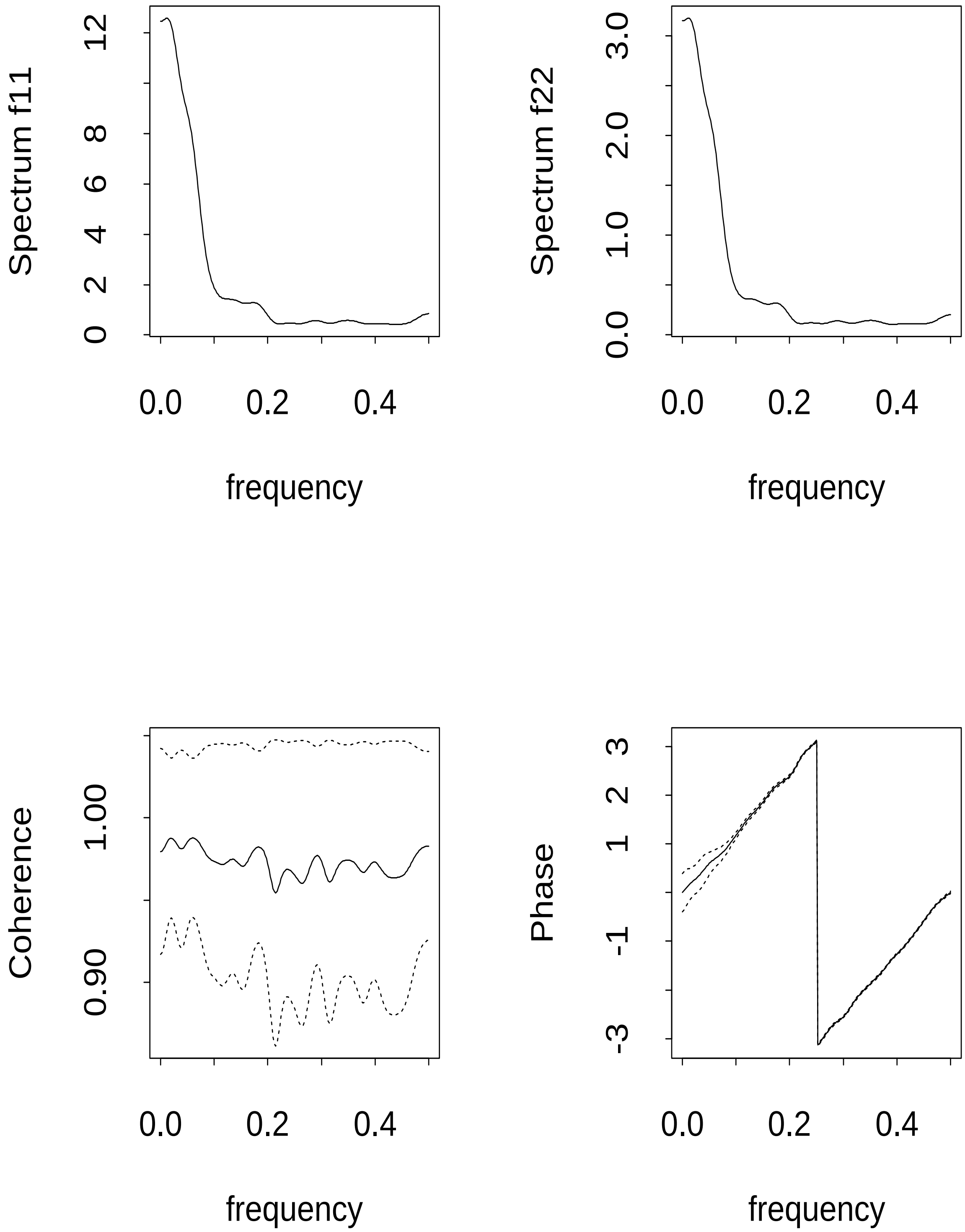


Fig 6.15 *Interpreting the phase spectrum.*

```
#  Interpreting the phase spectrum
#
# -----------------------------------------------------
{
  k <- length(beta)
  z <- arima.sim(list(order(length(alpha),0,0),ar=alpha),n=n+k)
  fz <- filter(z,beta,sides=1)     # fz = beta(B)*z(t)
  fz <- fz[(1+k):(n+k)]            # the first k-1 are NA's
  fz <- fz + sig*rnorm(n)
  z <- z[(1+k):(n+k)]              # align them
  coher(z,fz,ioptw,M)
}

ex()    # Fig 6.15
```

Observe that the phase spectrum has a straight line with a slope of $\pi/.25 = 2\times 2\pi$, which agrees with the theory discussed above.

6.12 Regression analysis

Given a multiple time series $(\mathbf{x}_t, y_t)$ with $\mathbf{x}_t = (x_{t1}, \ldots, x_{tq})'$. The classical time series regression deals with the model

$$y_t = \sum_1^q \beta_j x_{tj} + v_t = \boldsymbol{\beta}'\mathbf{x}_t + v_t,$$

where v_t is an error process assumed to have mean zero and covariance structure given by

$$E(v_s v_t) = \sigma_v^2 R(s-t).$$

When $R(s-t) = \delta_{st}$, we have a situation with uncorrelated errors and this is the ordinary regression problem. To estimate the coefficients β_j, recall the least squares approach in which the estimates are obtained by miniming the residual sum of squares (RSS)

$$\text{RSS} := \sum_t (y_t - \boldsymbol{\beta}'\mathbf{x}_t)^2.$$

The estimates are given by

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y},$$

where the matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, x_n)'$ is of rank q and $\mathbf{Y} = (y_1, \ldots, y_n)'$. The RSS can also be written by

$$\text{RSS} = (\mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}})'(\mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}}) = \mathbf{Y}'\mathbf{Y} - \hat{\boldsymbol{\beta}}'\mathbf{X}'\mathbf{Y}.$$

Note that the OLS $\hat{\beta}$ is unbiased:

$$E(\hat{\boldsymbol{\beta}}) = \boldsymbol{\beta}$$

and that

$$\operatorname{cov}(\hat{\boldsymbol{\beta}}) = \sigma_v^2(\mathbf{X}'\mathbf{X})^{-1}.$$

If the error v_t has a normal distribution $N(0, \sigma_v^2)$, then

$$\frac{\hat{\beta}_j - \beta_j}{\hat{\sigma}_v\sqrt{c_{jj}}} \sim t_{T-q},$$

where c_{jj} is the j-th diagonal element of the matrix $(\mathbf{X}'\mathbf{X})^{-1}$ and

$$\hat{\sigma}_v^2 = \frac{\text{RSS}}{T-q}$$

is an estimate of σ_v^2, which has a chi squared distribution and is independent of $\hat{\beta}$.

Suppose $(p < q)$ and that we are interested in testing the hypothesis $H_0 : \beta_{p+1} = \beta_{p+2} = \cdots = \beta_q = 0$. This can be accomplished by the test

$$\frac{(\text{RSS}_H - \text{RSS})/(q-p)}{\text{RSS}/(T-q)} \sim F_{q-p,T-q}.$$

When $R(s-t) \neq \delta_{st}$, i.e. the error v_t is a time series. Then the coeficients can be estimated by the weighted least squares method just like the linear regression case with general covariance structure. The estimates are given by

$$\hat{\boldsymbol{\beta}}_w = (\mathbf{X}'\mathbf{R}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{R}^{-1}\mathbf{Y}.$$

However, some simulation (Hannan, 1970) had shown that no efficieny is lost by considering the special case $R(s-t) = \delta_{st}$.

$c_{11}(\cdot)$ is a symmetric function with maximum equal to $c_{11}(0)$.

Examples

The following S functions illustrates the use of `arima.mle` in fitting a regression model to `lung`.

```
> attach(lung)
> x1 <- atm   # mortality
> x2 <- smk   # British smoke
> x3 <- so2   # sulfur dioxide
> x4 <- tmp   # temperature (C)
> x5 <- rhu   # relative humidity
```

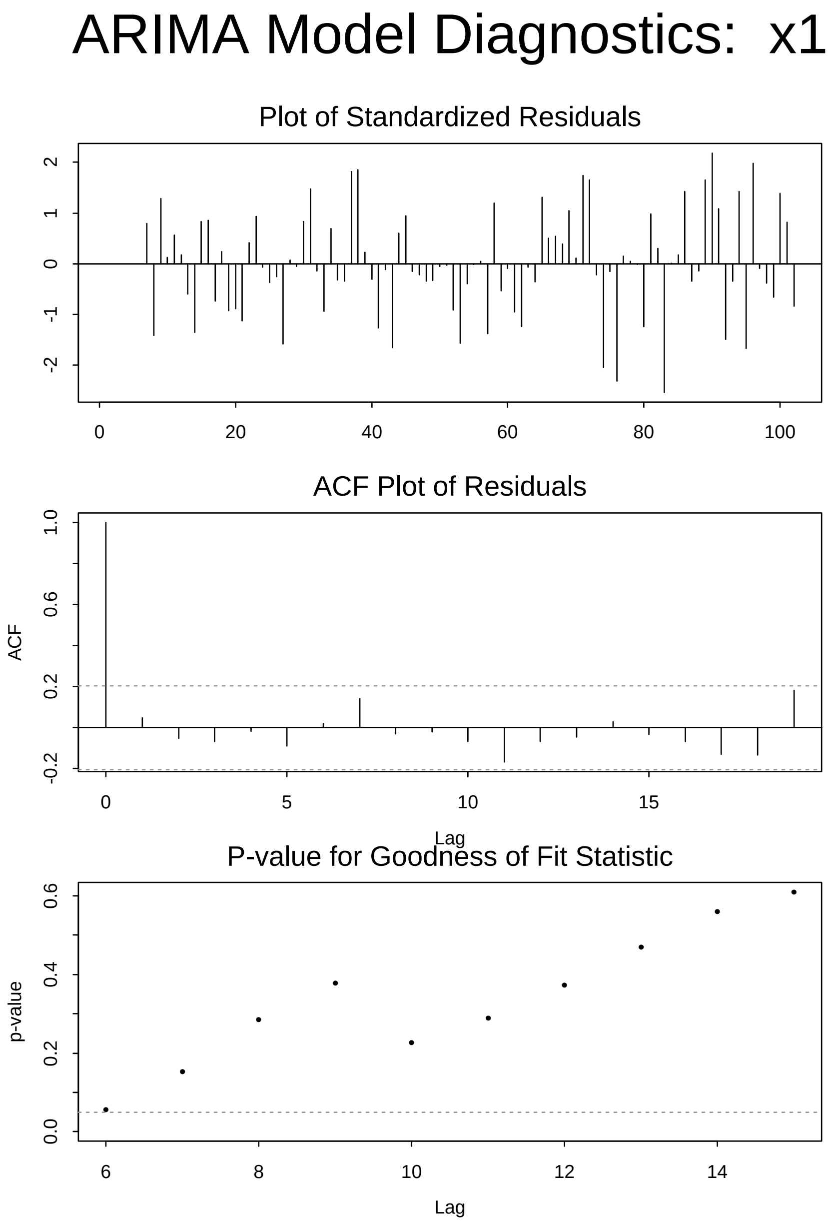


Fig 6.16 *Diagnostics analysis.*

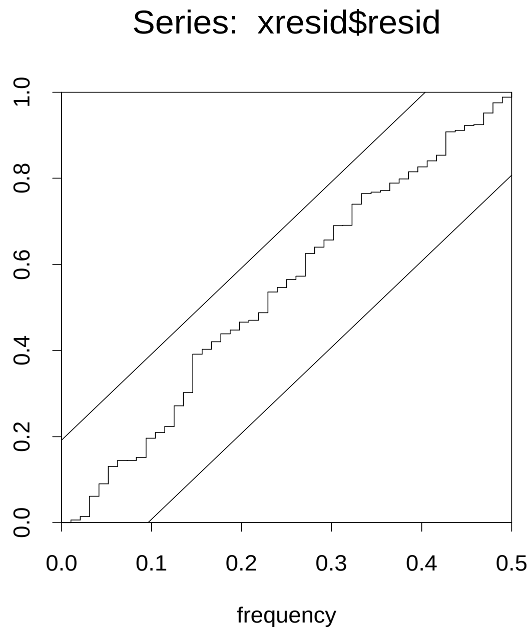


Fig 6.17 *Test of residuals.*

```
> x1 <- x1-mean(x1)
> x2 <- x2-mean(x2)
> x3 <- x3-mean(x3)
> x4 <- x4-mean(x4)
> x5 <- x5-mean(x5)

> x.lm <- lm(x1 ~ x2 + x3 + x4 + x5)    # fit a linear model
> summary(x.lm)

Call: lm(formula = x1 ~ x2 + x3 + x4 + x5)
Residuals:
    Min     1Q Median    3Q   Max
 -52.25 -14.23 -2.355 14.59 51.31

Coefficients:
             Value Std. Error t value Pr(>|t|)
(Intercept) 0.0000 2.1392     0.0000  1.0000
         x2 0.0213 0.0116     1.8339  0.0697
         x3 0.0338 0.0175     1.9276  0.0568
         x4 0.6389 0.7215     0.8855  0.3781
         x5 0.2596 0.3226     0.8046  0.4230
```

```
Residual standard error: 21.61 on 97 degrees of freedom
Multiple R-Squared: 0.3101
F-statistic: 10.9 on 4 and 97 degrees of freedom, the p-value is
 2.436e-007

Correlation of Coefficients:
   (Intercept)      x2      x3      x4
x2  0.0000
x3  0.0000     -0.8164
x4  0.0000      0.0746  0.0164
x5  0.0000     -0.1981  0.0648  0.0403

> X <- matrix(c(x2,x3,x4,x5),ncol=4)

# Previous analysis suggests AR(6)
# fit all predictors using arima.mle

> xmle <- arima.mle(x=x1,model=list(order=c(6,0,0)),xreg=X)

> xmle$reg.coef
[1]  0.02684285  0.01684208 -2.61317564  0.17837581

> arima.diag(xmle)    # Fig 6.16

> xresid <- arima.diag(xmle,plot=F,acf.resid=F, gof.lag=0, resid=T)
> cpgram(xresid$resid)    # Fig 6.17
> xmle$sigma2
[1] 340.5254


# Start the analysis with temperature (t-2):
#
x4 <- x4[1:100]
x1 <- x1[3:102]
x2 <- x2[3:102]
x3 <- x3[3:102]
x5 <- x5[3:102]

 x.lm <- lm(x1 ~ x2 + x3 + x4 + x5)
 summary(x.lm)

Call: lm(formula = x1 ~ x2 + x3 + x4 + x5)
```

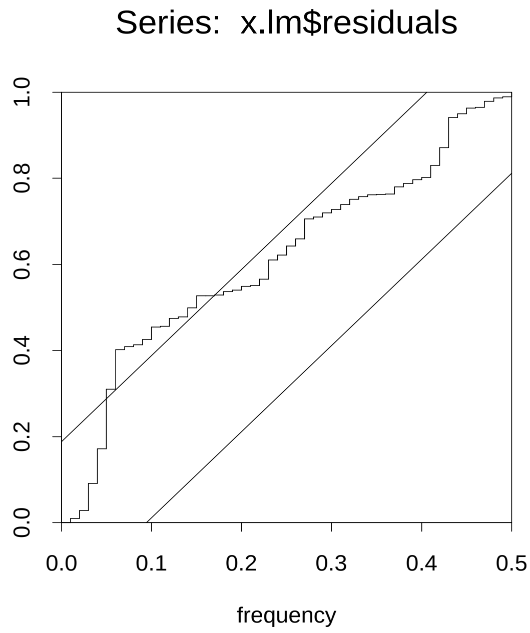


Fig 6.18 *cpgram analysis.*

```
Residuals:
    Min     1Q Median    3Q  Max
 -49.83 -10.75 -2.937 13.92 44.2

Coefficients:
              Value Std. Error t value Pr(>|t|)
(Intercept)  0.3938  2.0881     0.1886  0.8508
         x2  0.0184  0.0112     1.6363  0.1051
         x3  0.0321  0.0170     1.8887  0.0620
         x4 -2.0073  0.6937    -2.8936  0.0047
         x5  0.2215  0.3135     0.7065  0.4816

Residual standard error: 20.88 on 95 degrees of freedom
Multiple R-Squared: 0.3519
F-statistic: 12.9 on 4 and 95 degrees of freedom, the p-value is
 1.998e-008

Correlation of Coefficients:
   (Intercept)      x2      x3      x4
x2  0.0002
x3 -0.0082     -0.8165
x4 -0.0083      0.0697  0.0066
```

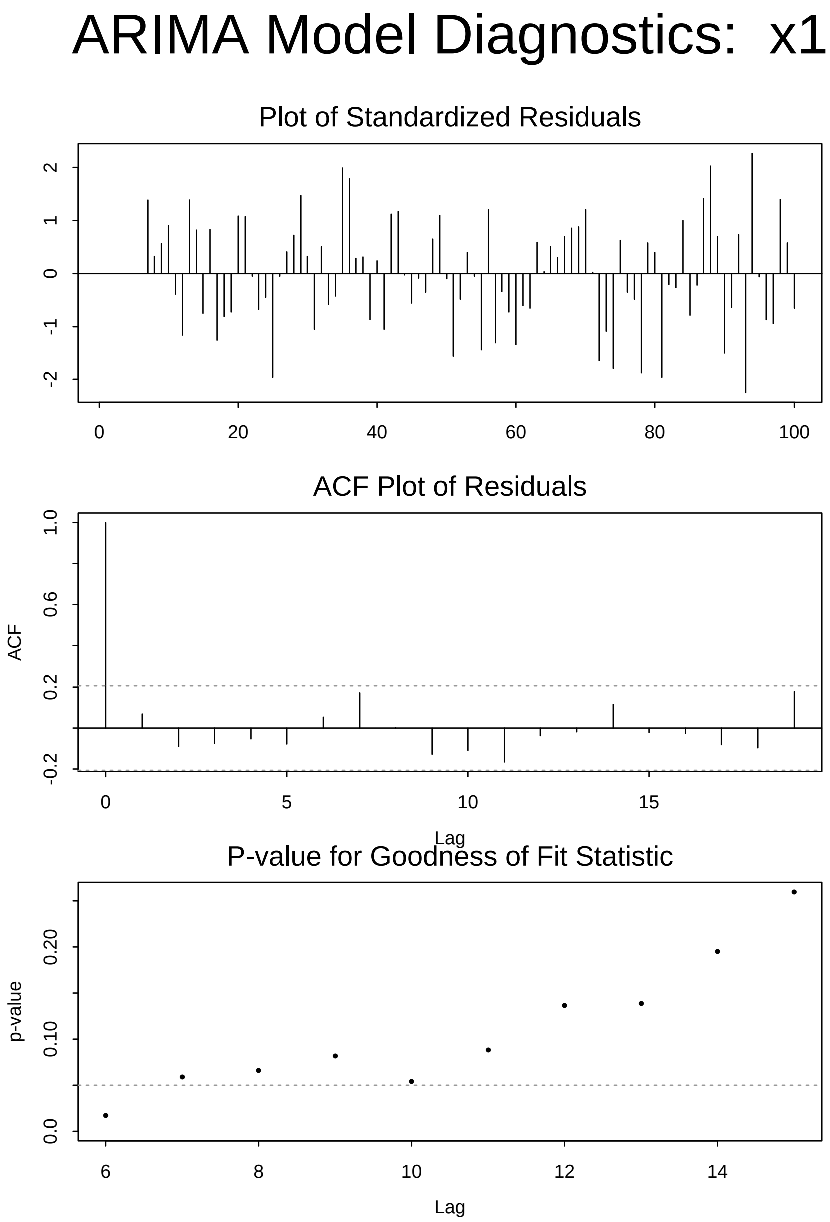


Fig 6.19 *Diagnostics.*

```
x5 -0.0156     -0.2016  0.0697 -0.0147

> cpgram(x.lm$residuals)    # Fig 6.18

> ar(x.lm$residuals)$order
[1] 6

> xmle <- arima.mle(x=x1,model=list(order=c(6,0,0)),xreg=x4)

> xmle$reg.coef
[1] -2.429177
> arima.diag(xmle)
> xresid <- arima.diag(xmle,plot=F,acf.resid=F, gof.lag=0, resid=T)
> cpgram(xresid$resid)

> xmle$model
$order:
[1] 6 0 0

$ar:
[1]  0.17214601  0.18241654  0.06839325  0.07482068  0.03983901
[6] -0.33678468

$ndiff:
[1] 0

> 2*sqrt(diag(xmle$var.coef))       # Nothing is significant!
[1] 0.1942335 0.1972819 0.2002446 0.2002446 0.1972819 0.1942335

# ----------------------------------------------
X <- matrix(c(x2,x3,x4,x5),ncol=4)

# Previous analysis suggests AR(6)
# fit all predictors

> xmle <- arima.mle(x=x1,model=list(order=c(6,0,0)),xreg=X)
> xmle$sigma2       # smaller sigma2
[1] 289.3448

> xmle$reg.coef
[1]  0.02684285  0.01684208 -2.61317564  0.17837581
```

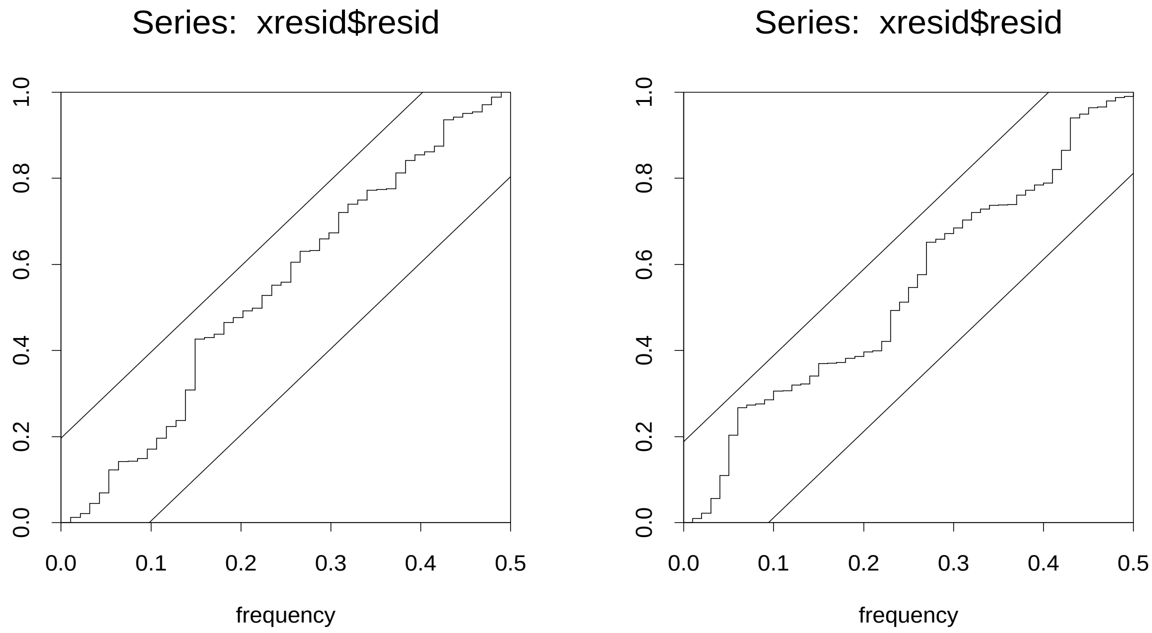


Fig 6.20 *Residual analysis using cpgram.*

```
> arima.diag(xmle)          # Fig 6.19

> xresid <- arima.diag(xmle,plot=F,acf.resid=F, gof.lag=0, resid=T)
> cpgram(xresid$resid)          # Fig 6.20 (left)

# Try low order MA
> xmle <- arima.mle(x=x1,model=list(order=c(0,0,2)),xreg=X)

> xmle$reg.coef
[1]  0.02247670  0.02689484 -2.17186135  0.11100214

> arima.diag(xmle)          # Fig 6.21: Not a good model

> xresid <- arima.diag(xmle,plot=F,acf.resid=F, gof.lag=0, resid=T)
> cpgram(xresid$resid)          # Fig 6.20 (right)
```

Problems

1. The gas furnace data is stored in `gasfurn`. Plot the series and comment on the stationarity. Estimate the cross-correlation and discuss the correlated structures of the series.

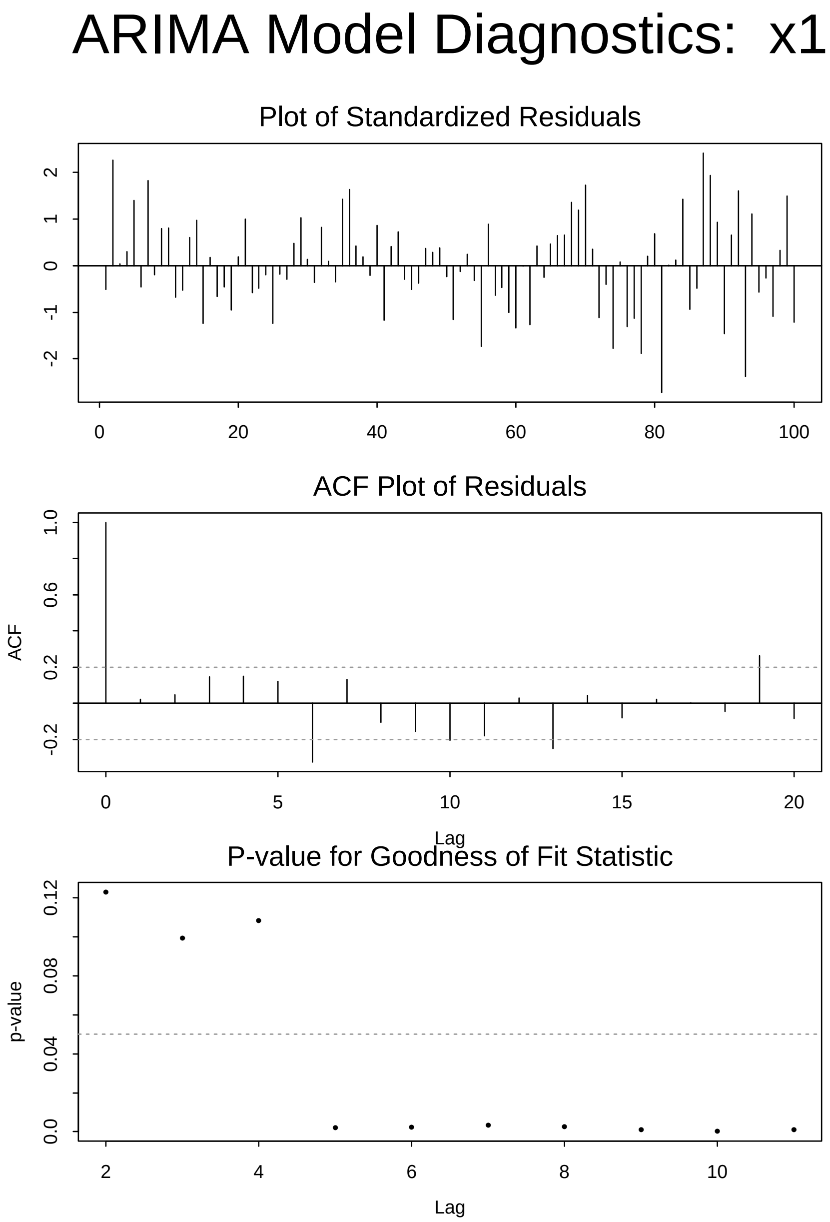


Fig 6.21 *Diagnostics.*

2. An alternative way to implement the function `estfilt` is given by

```
estfilt <- function(x1,x2,ioptw,M,kn)
#----------------------------------------------------------
#
#  Function to estimate the best filter coefficients
#  of bivariate series x1 and x2 using window ioptw and M
#  Alternative to estfilt() in biv.s. This function uses
#  the direct Fourier transform via Conj(A)=f12/f11.
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#    kn+1:        number of beta's
#
#  Value: beta[1:(kn+1)] (coefficients of the transfer function)
#
#----------------------------------------------------------
{
  f11 <- spe(x1,ioptw,M)$f
  f22 <- spe(x2,ioptw,M)$f
  f12 <- xspe(x1,x2,ioptw,M)
  c12 <- f12$re
  q12 <- f12$im

  c121 <- c12/f11
  q121 <- q12/f11

  q <- length(c12)
  c121 <- c(c121,c121[(q-1):2])     # extend to [0,1] via sym
  q121 <- c(q121,-q121[(q-1):2])    # extend to [0,1] via antisym

  z <- c121 + q121*1i               # Form complex array for FFT

  beta <- Re(fft(z))/(2*q-2)
  beta[1:(kn+1)]
}
```

Verify that the two method are equivalent.

3. Use `multpoly` and `divpoly` to expand the following rational function up to 10

terms:

$$(.53 + .37z + .51z^2)/(1 - .57z).$$

4. In the example, we establish that the simple relationship in the `bp` is given by `sbp - mean(sbp) = 0.6158987 (dbp - mean(dbp))`. Use spectral method to study the residuals obtained from this model.

5. Use `filter` to extract the residual from the transfer function model fitted to the gas furnace data. Study the goodness of fit by examing the residuals using spectral analysis. Comment briefly.

6. In analyzing `lung`, we noted that adjusted total mortality is correlated with the temperature series at lag -2. Verify this by using `coher` and `filtest`.

7. The dataset `d12` describes the mortality among male (`mdeaths`) and female (`fdeaths`) [MASS].

   (a) Comment on the stationarity of both series.

   (b) Estimate the auto and cross-correlations.

   (c) Fit parametric models to each series.

   (d) Establish their dependence using `indtest`, `crossp`, `coher` and `cohtest`.

   (e) Fit a transfer function model using `fdeaths` as the response.

   (f) Use `filter` to extract the residual series from the transfer function model. Examine the goodness of fit by carrying a spectral analysis on the residual series.

   (g) Fit a parametric time series model to the residuals using AR(p) and identify the order p. Use `cpgram` and `arima.diag` to examine the fit.

   (h) Based on the above analysis, build a model for describing the relationship between these series.

6.13 SAS Examples

```
/*
    Transfer Model: Gas Furnace Data

    Box, G.E.P., and Jenkins, G.M. (1976).  Time Series
    Analysis: Forecasting and Control.
    Oakland,  CA: Holden-Day.
*/

options ls=70 ps=100 nodate pageno=1;
```

```
title 'Box and Jenkins Series J: Gas Furnace Data';
data seriesj;
   infile 'c:/z2/b259/sas/gasfurn.dat';
   input x y @@;
   label X='Input Gas Rate' Y='Output CO2';

* Fit a Transfer Function Model;
proc arima;

   identify var=x nlag=10;  * Look at the Input Process;
   estimate p=3;            * Fit a Model for the Input;

   identify var=y crosscor=(x) nlag=10; * Crosscorr Prewhitened Series;

   estimate input=(3$ (1,2)/(1,2)x) plot maxit=30;
           * Fit Transfer Function Part of Model and Look at Residuals;

   estimate p=2 input=(3$ (1,2)/(1)x);  * Estimate Full Model;
quit;
```

```
              Box and Jenkins Series J: Gas Furnace Data            1

                             ARIMA Procedure


                  Name of variable = X.

                  Mean of working series = -0.05683
                  Standard deviation     = 1.070952
                  Number of observations =      296

                             Autocorrelations

Lag Covariance Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
  0   1.146938    1.00000  |                    |********************|
  1   1.092430    0.95247  |                  . |******************* |
  2   0.956652    0.83409  |                .   |*****************   |
  3   0.782051    0.68186  |               .    |**************      |
  4   0.609291    0.53123  |               .    |***********         |
  5   0.467380    0.40750  |              .     |********            |
  6   0.364957    0.31820  |              .     |******              |
```

```
 7   0.298427    0.26019  |              .     |*****.              |
 8   0.260943    0.22751  |              .     |*****.              |
 9   0.244378    0.21307  |              .     |**** .              |
10   0.238942    0.20833  |              .     |**** .              |
                           "." marks two standard errors


                        Inverse Autocorrelations

           Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
             1   -0.71090  |      **************| .                  |
             2    0.26217  |                  . |*****               |
             3   -0.13005  |                 ***| .                  |
             4    0.14777  |                  . |***                 |
             5   -0.06803  |                  .*| .                  |
             6   -0.01147  |                  . | .                  |
             7   -0.01649  |                  . | .                  |
             8    0.06108  |                  . |*.                  |
             9   -0.04490  |                  .*| .                  |
            10    0.01100  |                  . | .                  |


                        Partial Autocorrelations

           Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
             1    0.95247  |                  . |******************* |
             2   -0.78796  |    ****************| .                  |
             3    0.33897  |                  . |*******             |
             4    0.12121  |                  . |**                  |
             5    0.05896  |                  . |*.                  |
             6   -0.11147  |                  **| .                  |
             7    0.04862  |                  . |*.                  |
             8    0.09945  |                  . |**                  |
             9    0.01587  |                  . | .                  |
            10   -0.06973  |                  .*| .                  |


                 Autocorrelation Check for White Noise

   To   Chi                    Autocorrelations
  Lag  Square DF   Prob
    6  786.35  6  0.000  0.952  0.834  0.682  0.531  0.408  0.318
```

```
            Box and Jenkins Series J: Gas Furnace Data           2

                            ARIMA Procedure


                  Conditional Least Squares Estimation

                                        Approx.
           Parameter   Estimate      Std Error   T Ratio  Lag
           MU          -0.12280        0.10902     -1.13   0
           AR1,1        1.97607        0.05499     35.94   1
           AR1,2       -1.37499        0.09967    -13.80   2
           AR1,3        0.34336        0.05502      6.24   3

           Constant Estimate  = -0.0068229

           Variance  Estimate = 0.03579652
           Std Error Estimate = 0.18919968
           AIC                = -141.66743*
           SBC                =   -126.906*
           Number of Residuals=        296
           * Does not include log determinant.

                      Correlations of the Estimates


      Parameter           MU        AR1,1        AR1,2        AR1,3

      MU               1.000       -0.017        0.014       -0.016
      AR1,1           -0.017        1.000       -0.941        0.790
      AR1,2            0.014       -0.941        1.000       -0.941
      AR1,3           -0.016        0.790       -0.941        1.000


                  Autocorrelation Check of Residuals

  To   Chi                      Autocorrelations
 Lag  Square DF   Prob
   6   10.30  3  0.016 -0.042  0.068  0.056 -0.145 -0.009  0.059
  12   19.89  9  0.019  0.014  0.002 -0.055  0.035  0.143 -0.079
  18   27.92 15  0.022  0.099  0.043 -0.082  0.017  0.066 -0.052
  24   31.05 21  0.073 -0.078  0.024  0.015  0.030  0.045  0.004
```

```
  30    34.58 27  0.150 -0.007 -0.004  0.073 -0.038 -0.062  0.003
  36    38.84 33  0.223  0.010  0.002  0.082  0.045  0.056 -0.023
  42    41.18 39  0.375  0.002  0.033 -0.061 -0.003 -0.006 -0.043
  48    42.73 45  0.569  0.018  0.051 -0.012  0.015 -0.027  0.020


Model for variable X

Estimated Mean = -0.1228034

Autoregressive Factors
Factor 1: 1 - 1.9761 B**(1) + 1.375 B**(2) - 0.34336 B**(3)

            Box and Jenkins Series J: Gas Furnace Data            3

                         ARIMA Procedure


                Name of variable = Y.

                Mean of working series = 53.50912
                Standard deviation     = 3.196707
                Number of observations =      296

                         Autocorrelations

Lag Covariance Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
  0  10.218937    1.00000  |                    |********************|
  1   9.920101    0.97076  |                  . |******************* |
  2   9.156572    0.89604  |                .   |******************  |
  3   8.099002    0.79255  |               .    |****************    |
  4   6.948508    0.67996  |             .      |**************      |
  5   5.870550    0.57448  |             .      |************        |
  6   4.960762    0.48545  |             .      |**********          |
  7   4.251890    0.41608  |             .      |********            |
  8   3.736119    0.36561  |            .       |*******             |
  9   3.376155    0.33038  |            .       |*******             |
 10   3.132316    0.30652  |            .       |******.             |
                           "." marks two standard errors
```

```
                     Inverse Autocorrelations

 Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
   1    -0.60406  |        ************| .                  |
   2     0.03304  |                  . |*.                  |
   3     0.05747  |                  . |*.                  |
   4     0.04795  |                  . |*.                  |
   5    -0.01833  |                  . | .                  |
   6    -0.04534  |                  .*| .                  |
   7     0.05122  |                  . |*.                  |
   8    -0.03710  |                  .*| .                  |
   9     0.02130  |                  . | .                  |
  10    -0.00607  |                  . | .                  |


                     Partial Autocorrelations

 Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
   1     0.97076  |                  . |******************* |
   2    -0.80388  |    ****************| .                  |
   3     0.18833  |                  . |****                |
   4     0.25999  |                  . |*****               |
   5     0.05949  |                  . |*.                  |
   6    -0.06258  |                  .*| .                  |
   7    -0.01435  |                  . | .                  |
   8     0.05490  |                  . |*.                  |
   9     0.00545  |                  . | .                  |
  10     0.03141  |                  . |*.                  |


                Autocorrelation Check for White Noise

 To    Chi                   Autocorrelations
Lag  Square DF   Prob
  6 1023.15  6  0.000  0.971  0.896  0.793  0.680  0.574  0.485


            Correlation of Y and X
            Both series have been prewhitened.
            Variance of transformed series = 0.131438 and
            0.035357
            Number of observations =      296
```

```
                          Crosscorrelations

Lag Covariance Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
-10  0.0015683    0.02301  |                   . | .                  |
 -9 0.00013502    0.00198  |                   . | .                  |
 -8 -0.0060480   -0.08872  |                  **| .                   |
 -7 -0.0017624   -0.02585  |                  .*| .                   |
 -6 -0.0080539   -0.11814  |                  **| .                   |
 -5 -0.0000944   -0.00138  |                   . | .                  |
 -4 -0.0012802   -0.01878  |                   . | .                  |
 -3 -0.0031078   -0.04559  |                  .*| .                   |
 -2 0.00065212    0.00957  |                   . | .                  |
 -1 -0.0019166   -0.02811  |                  .*| .                   |
  0 -0.0003673   -0.00539  |                   . | .                  |
  1  0.0038939    0.05712  |                   . |*.                  |
  2 -0.0016971   -0.02489  |                   . | .                  |
  3  -0.019231   -0.28210  |              ******| .                   |
  4  -0.022479   -0.32974  |             *******| .                   |
  5  -0.030909   -0.45341  |           *********| .                   |
  6  -0.018122   -0.26583  |               *****| .                   |
  7  -0.011426   -0.16761  |                 ***| .                   |
  8 -0.0017355   -0.02546  |                  .*| .                   |
  9  0.0022590    0.03314  |                   . |*.                  |
 10 -0.0035152   -0.05156  |                  .*| .                   |
                           "." marks two standard errors

              Box and Jenkins Series J: Gas Furnace Data            4

                           ARIMA Procedure


                Crosscorrelation Check Between Series

   To   Chi                      Crosscorrelations
  Lag  Square DF   Prob
    5  117.75  6  0.000 -0.005  0.057 -0.025 -0.282 -0.330 -0.453


  Both variables have been prewhitened by the following filter:
```

```
Prewhitening Filter


Autoregressive Factors
Factor 1: 1 - 1.9761 B**(1) + 1.375 B**(2) - 0.34336 B**(3)

            Box and Jenkins Series J: Gas Furnace Data            5

                         ARIMA Procedure


               Conditional Least Squares Estimation

                              Approx.
 Parameter    Estimate    Std Error   T Ratio  Lag  Variable Shift
 MU          53.32237      0.04932   1081.24    0   Y            0
 NUM1        -0.62868      0.25385     -2.48    0   X            3
 NUM1,1       0.47258      0.62253      0.76    1   X            3
 NUM1,2       0.73660      0.81006      0.91    2   X            3
 DEN1,1       0.15411      0.90483      0.17    1   X            3
 DEN1,2       0.27774      0.57345      0.48    2   X            3

 Constant Estimate  = 53.3223702

 Variance  Estimate = 0.70424109
 Std Error Estimate = 0.83919073
 AIC                =  729.72486*
 SBC                = 751.764799*
 Number of Residuals=        291
 * Does not include log determinant.

                   Correlations of the Estimates


                             Y          X          X          X          X
Variable  Parameter         MU       NUM1     NUM1,1     NUM1,2     DEN1,1

Y         MU             1.000      0.013      0.002     -0.002      0.004
X         NUM1           0.013      1.000      0.755     -0.447      0.089
X         NUM1,1         0.002      0.755      1.000      0.121     -0.538
X         NUM1,2        -0.002     -0.447      0.121      1.000     -0.892
X         DEN1,1         0.004      0.089     -0.538     -0.892      1.000
```

```
X         DEN1,2       -0.006    -0.065     0.565     0.870    -0.998

                                             X
                    Variable  Parameter    DEN1,2

                    Y         MU           -0.006
                    X         NUM1         -0.065
                    X         NUM1,1        0.565
                    X         NUM1,2        0.870
                    X         DEN1,1       -0.998
                    X         DEN1,2        1.000


                  Autocorrelation Check of Residuals

    To   Chi                   Autocorrelations
   Lag  Square DF   Prob
     6  496.45  6  0.000  0.893  0.711  0.502  0.312  0.167  0.064
    12  498.58 12  0.000 -0.003 -0.040 -0.054 -0.040 -0.022 -0.021
    18  539.38 18  0.000 -0.045 -0.083 -0.131 -0.170 -0.196 -0.195
    24  561.87 24  0.000 -0.163 -0.102 -0.026  0.047  0.106  0.142
    30  585.90 30  0.000  0.158  0.156  0.131  0.081  0.013 -0.037
    36  592.42 36  0.000 -0.048 -0.018  0.038  0.070  0.079  0.067
    42  593.44 42  0.000  0.042  0.025  0.013  0.004  0.006  0.019
    48  601.94 48  0.000  0.043  0.068  0.084  0.082  0.061  0.023
              Box and Jenkins Series J: Gas Furnace Data            6

                           ARIMA Procedure


                  Autocorrelation Plot of Residuals

Lag Covariance Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
  0   0.704241    1.00000  |                    |********************|
  1   0.628846    0.89294  |                  . |******************  |
  2   0.500490    0.71068  |                .   |**************      |
  3   0.353404    0.50182  |                .   |**********          |
  4   0.219895    0.31224  |               .    |******              |
  5   0.117330    0.16660  |               .    |*** .               |
  6   0.044967    0.06385  |               .    |*   .               |
  7 -0.0023551   -0.00334  |               .    |    .               |
  8  -0.028030   -0.03980  |               .   *|    .               |
```

```
 9  -0.037891   -0.05380   |                .   *|    .              |
10  -0.028378   -0.04030   |                .   *|    .              |
                           "." marks two standard errors


                       Inverse Autocorrelations

            Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
              1    -0.57346   |         ***********| .                  |
              2     0.02264   |                  . | .                  |
              3     0.03631   |                  . |*.                  |
              4     0.03941   |                  . |*.                  |
              5    -0.01256   |                  . | .                  |
              6    -0.01618   |                  . | .                  |
              7     0.02680   |                  . |*.                  |
              8    -0.05895   |                  .*| .                  |
              9     0.07043   |                  . |*.                  |
             10    -0.02987   |                  .*| .                  |


                       Partial Autocorrelations

            Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
              1     0.89294   |                  . |******************  |
              2    -0.42765   |           *********| .                  |
              3    -0.13463   |                 ***| .                  |
              4     0.02199   |                  . | .                  |
              5     0.03891   |                  . |*.                  |
              6    -0.02219   |                  . | .                  |
              7    -0.02249   |                  . | .                  |
              8     0.01538   |                  . | .                  |
              9     0.00634   |                  . | .                  |
             10     0.07737   |                  . |**                  |


            Crosscorrelation Check of Residuals with Input X

      To    Chi                   Crosscorrelations
     Lag  Square DF   Prob
       5    0.48  2  0.785 -0.009 -0.005  0.026  0.013 -0.017 -0.022
      11    0.93  8  0.999 -0.006  0.008  0.022  0.023 -0.017 -0.013
      17    2.63 14  1.000  0.012  0.035  0.037  0.039 -0.005 -0.040
```

```
 23   19.19 20  0.509 -0.076 -0.108 -0.122 -0.122 -0.094 -0.041
 29   20.12 26  0.786 -0.039 -0.013  0.010 -0.020 -0.031 -0.005
 35   24.22 32  0.836 -0.022 -0.031 -0.074 -0.036  0.014  0.076
 41   30.66 38  0.795  0.108  0.091  0.046  0.018  0.003  0.009
 47   31.65 44  0.918  0.008 -0.011 -0.040 -0.030 -0.002  0.028


        Model for variable Y

        Estimated Intercept = 53.3223702

        Input Number 1 is X with a shift of 3.

        The Numerator Factors are
        Factor 1: -0.6287 - 0.47258 B**(1) - 0.7366 B**(2)


        The Denominator Factors are
        Factor 1: 1 - 0.15411 B**(1) - 0.27774 B**(2)

              Box and Jenkins Series J: Gas Furnace Data            7

                             ARIMA Procedure


                 Conditional Least Squares Estimation

                             Approx.
   Parameter   Estimate    Std Error   T Ratio  Lag  Variable Shift
   MU          53.26307      0.11926    446.63   0   Y            0
   AR1,1        1.53292      0.04754     32.25   1   Y            0
   AR1,2       -0.63297      0.05006    -12.64   2   Y            0
   NUM1        -0.53522      0.07482     -7.15   0   X            3
   NUM1,1       0.37602      0.10287      3.66   1   X            3
   NUM1,2       0.51894      0.10783      4.81   2   X            3
   DEN1,1       0.54842      0.03822     14.35   1   X            3

   Constant Estimate  = 5.32937102

   Variance  Estimate = 0.05882782
   Std Error Estimate = 0.24254447
```

```
AIC               = 8.29281119*
SBC               = 34.0060741*
Number of Residuals=       291
* Does not include log determinant.
```

Correlations of the Estimates

Variable	Parameter	Y MU	Y AR1,1	Y AR1,2	X NUM1	X NUM1,1
Y	MU	1.000	-0.063	0.047	-0.008	-0.016
Y	AR1,1	-0.063	1.000	-0.927	-0.003	0.007
Y	AR1,2	0.047	-0.927	1.000	0.023	-0.005
X	NUM1	-0.008	-0.003	0.023	1.000	0.713
X	NUM1,1	-0.016	0.007	-0.005	0.713	1.000
X	NUM1,2	0.017	-0.002	0.005	-0.178	-0.467
X	DEN1,1	-0.049	0.015	-0.022	-0.013	-0.039

Variable	Parameter	X NUM1,2	X DEN1,1
Y	MU	0.017	-0.049
Y	AR1,1	-0.002	0.015
Y	AR1,2	0.005	-0.022
X	NUM1	-0.178	-0.013
X	NUM1,1	-0.467	-0.039
X	NUM1,2	1.000	-0.720
X	DEN1,1	-0.720	1.000

Autocorrelation Check of Residuals

To Lag	Chi Square	DF	Prob	Autocorrelations					
6	8.61	4	0.072	0.024	0.055	-0.073	-0.054	-0.054	0.119
12	15.43	10	0.117	0.032	0.028	-0.081	0.047	0.022	0.107
18	21.13	16	0.173	-0.038	0.052	-0.093	-0.013	-0.073	-0.005
24	27.52	22	0.192	-0.118	-0.002	-0.007	0.076	0.024	-0.004
30	36.94	28	0.120	0.034	-0.021	0.020	0.094	-0.118	0.065
36	44.26	34	0.112	-0.025	-0.057	0.113	0.022	0.030	0.065
42	45.62	40	0.250	-0.017	-0.036	-0.029	-0.013	-0.033	0.017

```
   48   48.60 46  0.369  0.024  0.069  0.024  0.017  0.022 -0.044


              Crosscorrelation Check of Residuals with Input X

   To   Chi                        Crosscorrelations
  Lag  Square DF   Prob
    5    0.93  3  0.819  0.008  0.004  0.010  0.008 -0.045  0.030
   11    6.60  9  0.678  0.075 -0.024 -0.019 -0.026 -0.111  0.013
   17   13.86 15  0.537  0.050  0.043  0.014  0.014 -0.141 -0.028
   23   18.55 21  0.614 -0.074 -0.078  0.023 -0.016  0.021  0.060
   29   27.99 27  0.411 -0.071 -0.001  0.038 -0.156  0.031  0.035
   35   35.18 33  0.365 -0.014  0.015 -0.039  0.028  0.046  0.142
   41   37.15 39  0.554  0.031 -0.029 -0.070 -0.006  0.012 -0.004
   47   42.42 45  0.582  0.036 -0.038 -0.053  0.107  0.029  0.021



        Model for variable Y

        Estimated Intercept = 53.2630713

        Autoregressive Factors
        Factor 1: 1 - 1.5329 B**(1) + 0.63297 B**(2)


        Input Number 1 is X with a shift of 3.

        The Numerator Factors are
        Factor 1: -0.5352 - 0.37602 B**(1) - 0.51894 B**(2)


        The Denominator Factors are
        Factor 1: 1 - 0.54842 B**(1)
```

6.14 SAS Examples

```
/*
  Regression Models with ARMA Errors

  Maddala, G.S. (1977).  Econometrics.  New York: McGraw-Hill.
```

```
     Chapter 12, pp. 280-281.
*/

options ls=70 ps=100 nodate pageno=1;

title 'Grunfeld''s Investment Models With Autoregressive Errors';
data grunfeld;
   infile 'c:/z2/b259/sas/regar.dat';
   input year gei gef gec wi wf wc;
   label gei='Gross Investment GE'
         gec='Capital Stock Lagged GE'
         gef='Value of Outstanding Shares GE Lagged'
         wi ='Gross Investment W'
         wc ='Capital Stock Lagged W'
         wf ='Value of Outstanding Shares Lagged W';

/* GE stands for General Electric and W for Westinghouse */

proc arima;
   identify var=gei crosscor=(gef gec) noprint;
   estimate p=1 method=ml input=(gef gec) plot;
   estimate p=2 method=ml input=(gef gec);
   estimate q=1 method=ml input=(gef gec);
   estimate q=1 p=1 method=ml input=(gef gec);
quit;
```

```
       Grunfeld's Investment Models With Autoregressive Errors       1

                          ARIMA Procedure


                    Maximum Likelihood Estimation

                           Approx.
   Parameter   Estimate    Std Error   T Ratio  Lag  Variable Shift
   MU         -18.38223    34.37520     -0.53    0   GEI          0
   AR1,1        0.47281     0.25657      1.84    1   GEI          0
   NUM1         0.03341     0.01775      1.88    0   GEF          0
   NUM2         0.13852     0.04257      3.25    0   GEC          0

   Constant Estimate  = -9.6909166
```

```
Variance  Estimate = 639.324853
Std Error Estimate =  25.284874
AIC                = 189.755947
SBC                = 193.738876
Number of Residuals=         20
```

Correlations of the Estimates

Variable	Parameter	GEI MU	GEI AR1,1	GEF NUM1	GEC NUM2
GEI	MU	1.000	-0.261	-0.831	-0.150
GEI	AR1,1	-0.261	1.000	0.462	-0.418
GEF	NUM1	-0.831	0.462	1.000	-0.350
GEC	NUM2	-0.150	-0.418	-0.350	1.000

Autocorrelation Check of Residuals

To Lag	Chi Square	DF	Prob	Autocorrelations					
6	23.01	5	0.000	0.325	-0.363	-0.587	-0.157	0.370	0.344
12	32.27	11	0.001	0.074	-0.355	-0.292	-0.038	0.136	0.039
18	32.38	17	0.013	-0.008	0.003	-0.015	0.007	-0.003	0.019

Autocorrelation Plot of Residuals

```
Lag Covariance Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
  0    639.325    1.00000  |                    |********************|
  1    207.470    0.32451  |           .        |******  .           |
  2   -231.912   -0.36275  |          .  *******|         .          |
  3   -375.585   -0.58747  |        ************|          .         |
  4   -100.124   -0.15661  |       .         ***|            .       |
  5    236.859    0.37048  |       .            |*******     .       |
                           "." marks two standard errors
```

Inverse Autocorrelations

```
Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
  1   -0.23076  |           .    *****|         .          |
```

```
   2    0.10436  |           .        |**      .           |
   3    0.21891  |           .        |****    .           |
   4    0.04512  |           .        |*       .           |
   5   -0.10791  |           .      **|        .           |


                       Partial Autocorrelations

 Lag Correlation -1 9 8 7 6 5 4 3 2 1 0 1 2 3 4 5 6 7 8 9 1
   1    0.32451  |           .        |******  .           |
   2   -0.52315  |          **********|        .           |
   3   -0.36687  |           . *******|        .           |
   4    0.02859  |           .        |*       .           |
   5    0.14045  |           .        |***     .           |


        Model for variable GEI

        Estimated Intercept = -18.382234

        Autoregressive Factors
        Factor 1: 1 - 0.47281 B**(1)


        Input Number 1 is GEF.
        Overall Regression Factor  =  0.03341

        Input Number 2 is GEC.
        Overall Regression Factor  = 0.138519
  Grunfeld's Investment Models With Autoregressive Errors       2

                            ARIMA Procedure


                     Maximum Likelihood Estimation

                             Approx.
Parameter    Estimate     Std Error    T Ratio  Lag  Variable Shift
MU         -15.66776     25.90965      -0.60    0   GEI          0
AR1,1        0.76428      0.19770       3.87    1   GEI          0
AR1,2       -0.61548      0.18978      -3.24    2   GEI          0
```

```
NUM1         0.03054      0.01307      2.34   0   GEF          0
NUM2         0.14742      0.02306      6.39   0   GEC          0

Constant Estimate  = -13.336355

Variance  Estimate = 388.812518
Std Error Estimate = 19.7183295
AIC                = 181.471456
SBC                = 186.450117
Number of Residuals=         20
```

Correlations of the Estimates

Variable	Parameter	GEI MU	GEI AR1,1	GEI AR1,2	GEF NUM1	GEC NUM2
GEI	MU	1.000	-0.188	0.008	-0.915	-0.204
GEI	AR1,1	-0.188	1.000	-0.451	0.250	-0.156
GEI	AR1,2	0.008	-0.451	1.000	0.011	-0.057
GEF	NUM1	-0.915	0.250	0.011	1.000	-0.155
GEC	NUM2	-0.204	-0.156	-0.057	-0.155	1.000

Autocorrelation Check of Residuals

To Lag	Chi Square	DF	Prob	Autocorrelations					
6	3.08	4	0.545	-0.024	0.125	-0.117	-0.078	0.194	-0.184
12	9.40	10	0.494	0.044	-0.348	-0.137	-0.106	0.048	-0.092
18	10.81	16	0.821	-0.014	0.069	0.007	0.091	-0.002	0.026

```
Model for variable GEI

Estimated Intercept = -15.667763

Autoregressive Factors
Factor 1: 1 - 0.76428 B**(1) + 0.61548 B**(2)
```

```
        Input Number 1 is GEF.
        Overall Regression Factor  = 0.030537

        Input Number 2 is GEC.
        Overall Regression Factor  = 0.147421
    Grunfeld's Investment Models With Autoregressive Errors       3

                         ARIMA Procedure


                  Maximum Likelihood Estimation

                               Approx.
   Parameter   Estimate     Std Error   T Ratio  Lag  Variable Shift
   MU         -24.95118     31.47548     -0.79    0   GEI          0
   MA1,1       -0.78516      0.18842     -4.17    1   GEI          0
   NUM1         0.03710      0.01527      2.43    0   GEF          0
   NUM2         0.13773      0.03326      4.14    0   GEC          0

   Constant Estimate  = -24.951177

   Variance  Estimate = 482.284785
   Std Error Estimate = 21.9609832
   AIC                = 184.823664
   SBC                = 188.806593
   Number of Residuals=         20

                   Correlations of the Estimates


                                 GEI         GEI         GEF         GEC
Variable    Parameter             MU       MA1,1        NUM1        NUM2

GEI         MU                 1.000       0.339      -0.867      -0.252
GEI         MA1,1              0.339       1.000      -0.422       0.133
GEF         NUM1              -0.867      -0.422       1.000      -0.185
GEC         NUM2              -0.252       0.133      -0.185       1.000


                 Autocorrelation Check of Residuals

    To   Chi                    Autocorrelations
```

```
Lag  Square DF   Prob
  6    9.85  5  0.080  0.109 -0.068 -0.467 -0.116  0.290  0.177
 12   18.13 11  0.079  0.094 -0.363 -0.218 -0.108  0.120 -0.010
 18   18.53 17  0.356  0.027 -0.034  0.013  0.025  0.002  0.029


             Model for variable GEI

             Estimated Intercept = -24.951177

             Moving Average Factors
             Factor 1: 1 + 0.78516 B**(1)


             Input Number 1 is GEF.
             Overall Regression Factor  = 0.037102

             Input Number 2 is GEC.
             Overall Regression Factor  =  0.13773
       Grunfeld's Investment Models With Autoregressive Errors       4

                          ARIMA Procedure


                      Maximum Likelihood Estimation

                          Approx.
Parameter   Estimate    Std Error   T Ratio  Lag  Variable Shift
MU         -26.46872     32.98901     -0.80   0   GEI          0
MA1,1       -0.68149      0.26596     -2.56   1   GEI          0
AR1,1        0.19627      0.34355      0.57   1   GEI          0
NUM1         0.03922      0.01618      2.42   0   GEF          0
NUM2         0.13103      0.04164      3.15   0   GEC          0

Constant Estimate  = -21.273814

Variance  Estimate = 507.075357
Std Error Estimate = 22.5183338
AIC                = 186.491876
SBC                = 191.470537
Number of Residuals=         20
```

```
                    Correlations of the Estimates


                             GEI         GEI         GEI         GEF         GEC
Variable  Parameter           MU       MA1,1       AR1,1        NUM1        NUM2

GEI       MU               1.000       0.247      -0.039      -0.817      -0.245
GEI       MA1,1            0.247       1.000       0.534      -0.258      -0.005
GEI       AR1,1           -0.039       0.534       1.000       0.211      -0.332
GEF       NUM1            -0.817      -0.258       0.211       1.000      -0.278
GEC       NUM2            -0.245      -0.005      -0.332      -0.278       1.000


                 Autocorrelation Check of Residuals

   To   Chi                     Autocorrelations
  Lag  Square DF   Prob
    6    9.39  4  0.052  0.066 -0.088 -0.429 -0.128  0.321  0.167
   12   17.60 10  0.062  0.115 -0.360 -0.221 -0.087  0.121 -0.009
   18   18.22 16  0.311  0.022 -0.044 -0.008  0.034 -0.006  0.038



                Model for variable GEI

                Estimated Intercept = -26.468724

                Autoregressive Factors
                Factor 1: 1 - 0.19627 B**(1)


                Moving Average Factors
                Factor 1: 1 + 0.68149 B**(1)


                Input Number 1 is GEF.
                Overall Regression Factor  = 0.039217

                Input Number 2 is GEC.
                Overall Regression Factor  = 0.131033
```

Chapter 7

State Space Models and the Kalman Filter

7.1 Measurement equation

$$\underset{N\times 1}{\mathbf{y}_t} = \underset{N\times m}{\mathbf{Z}_t}\ \underset{m\times 1}{\alpha_t} + \underset{N\times n}{\mathbf{S}_t},\ \underset{n\times 1}{\xi_t}, \qquad \xi_t \sim_{\text{iid}} (0, \mathbf{H}_t),$$

where $\mathbf{y}_t$ is the vector of obsrvations and α_t is the unobserved *state vector*.

7.2 Transition equation

$$\underset{m\times 1}{\alpha_t} = \underset{m\times m}{\mathbf{T}_t}\ \underset{m\times 1}{\alpha_{t-1}} + \underset{m\times g}{\mathbf{R}_t},\ \underset{g\times 1}{\eta_t}, \qquad \eta_t \sim_{\text{iid}} (0, \mathbf{Q}_t).$$

7.3 ARMA via State Space

The following stationary ARMA process

$$x_t - \phi_1 x_{t-1} - \cdots - \phi_m x_{t-m} = e_t + \theta_1 e_{t-1} + \cdots + \theta_m e_{t-m},$$

where $m = \max(p, q+1)$ and $\phi_j = 0, (j > p)$ and $\theta_k = 0, (k > q)$ can be cast in a state space model via

$$\alpha_t = \begin{pmatrix} \phi_1 & \\ \vdots & \mathbf{I}_{m-1} \\ \phi_{m-1} & \\ \phi_m & \mathbf{0}'_{m-1} \end{pmatrix} \alpha_{t-1} + \begin{pmatrix} 1 \\ \theta_1 \\ \vdots \\ \theta_{m-1} \end{pmatrix} e_t,$$

and

$$x_t = (1 \quad \mathbf{0}')'\alpha_t.$$

7.4 Minimum Mean Square Estimation

Consider

$$\underset{N\times 1}{\mathbf{y}} = \underset{N\times m}{\mathbf{Z}}\ \underset{m\times 1}{\alpha} + \underset{n\times 1}{\xi}, \qquad \xi \sim_{\text{iid}} (0, \sigma^2\Omega).$$

If α were not random, then the GLS estimator

$$\tilde{\alpha} = (\mathbf{Z}'\Omega^{-1}\mathbf{Z})^{-1}\mathbf{Z}'\Omega^{-1}\mathbf{y}$$

is unbiased with covariance $\sigma^2(\mathbf{Z}'\Omega^{-1}\mathbf{Z})^{-1}$, i.e.

$$\tilde{\alpha} - \alpha \sim (0, \sigma^2(\mathbf{Z}'\Omega^{-1}\mathbf{Z})^{-1}).$$

The above expression is also valid when α is random.

7.4.1 Combining Prior and Sample Information

Suppose now that

$$a_0 - \alpha \sim (0, \sigma^2 P_0).$$

Then

$$\begin{pmatrix} a_0 \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} \mathbf{I} \\ \mathbf{Z} \end{pmatrix}\alpha + \begin{pmatrix} a_0 - \alpha \\ \xi \end{pmatrix}.$$

This may be written as

$$\tilde{y} = \tilde{Z}\alpha + \tilde{\xi},$$

where

$$E(\tilde{\xi}) = 0, \qquad \text{and} \qquad E(\tilde{\xi}\tilde{\xi}') = \sigma^2 V = \sigma^2 \begin{pmatrix} P_0 & 0 \\ 0 & \Omega \end{pmatrix}.$$

From this

$$\hat{\alpha} = (\tilde{\mathbf{Z}}'V^{-1}\tilde{\mathbf{Z}})^{-1}\tilde{\mathbf{Z}}'V^{-1}\tilde{\mathbf{y}} = P(P_0^{-1}a_0 + \mathbf{Z}'\Omega^{-1}\mathbf{y}),$$

where

$$P = (P_0^{-1} + \mathbf{Z}'\Omega^{-1}\mathbf{Z})^{-1}.$$

Thus

$$\hat{\alpha} - \alpha \sim (0, \sigma^2 P).$$

7.5 Updating and Prediction Equations

In most applications, the matrices $\mathbf{S}_t$, $\mathbf{T}_t$ and $\mathbf{R}_t$ will be time-invariant. Thus

$$x_t = z_t'\alpha_t + \xi_t, \quad \xi \sim_{\text{iid}} (0, \sigma^2 h_t),$$

$$\alpha_t = T\alpha_{t-1} + R\eta_t, \quad \eta \sim_{\text{iid}} (0, \sigma^2 Q).$$

Let a_t denote the minimum mean square linear estimator (MMSLE) of α_t based on information upto and including time t, and $a_{t|t-1}$ denote the MMSLE of α_t based on information at time $t-1$.

7.5.1 Prediction

The transition equation can be used to predict the value of α_t based on information available at time $t-1$. Namely,

$$a_{t|t-1} = Ta_{t-1}.$$

$$a_{t|t-1} - \alpha_t = T(a_{t-1} - \alpha_{t-1}) + R\eta_t.$$

Note that

$$E(a_{t|t-1} - \alpha_t) = 0$$

and

$$E((a_{t|t-1} - \alpha_t)(a_{t|t-1} - \alpha_t)') = \sigma^2 TP_{t-1}T' + \sigma^2 RQR'.$$

Thus

$$a_{t|t-1} - \alpha_t \sim (0, \sigma^2 P_{t|t-1}),$$

where

$$P_{t|t-1} = TP_{t-1}T' + RQR'.$$

The MMSLE of y_t at time $t-1$ is

$$\tilde{y}_{t|t-1} = z_t' a_{t|t-1}.$$

The associated prediction error is

$$\nu_t = y_t - \tilde{y}_{t|t-1},$$

with

$$\text{var}(\nu_t) = \sigma^2(z_t' P_{t|t-1} z_t + h_t) = \sigma^2 f_t.$$

7.5.2 Updating

It can be shown that

$$a_t = a_{t|t-1} + P_{t|t-1} z_t(\mathbf{y}_t - z_t' a_{t|t-1})/f_t,$$

$$f_t = z_t' P_{t|t-1} z_t + h_t,$$

$$P_t = P_{t|t-1} - P_{t|t-1} z_t z_t' P_{t|t-1}/f_t.$$

7.6 Examples

7.6.1 Example 1

Consider the following MA(1) model:

$$x_t = e_t + \theta e_{t-1}, \qquad e_t \sim (0, \sigma^2).$$

Then

$$\begin{pmatrix} \alpha_{t,1} \\ \alpha_{t,2} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha_{t-1,1} \\ \alpha_{t-1,2} \end{pmatrix} + \begin{pmatrix} 1 \\ \theta \end{pmatrix} e_t.$$

Thus

$$\alpha_{t,1} = \alpha_{t,2} + e_t, \qquad \alpha_{t,2} = \theta e_t.$$

Or,

$$\alpha_{t,1} = e_t + \theta e_{t-1}.$$

To get x_t from α_t, we simply let

$$x_t = (1 \quad 0)\alpha_t.$$

Starting from $a_0 = 0$, we have

$$P_{1|0} = P_0 = \sigma^{-2} E(\alpha_0 \alpha_0') = E \begin{pmatrix} x_1 \\ \theta e_1 \end{pmatrix} (x_1 \quad \theta e_1) = \begin{pmatrix} 1+\theta^2 & \theta \\ \theta & \theta^2 \end{pmatrix}.$$

By the predicting equations,

$$\tilde{x}_{1|0} = z_1 a_{1|0} = 0,$$
$$\nu_1 = x_1 - \tilde{x}_{1|0} = x_1,$$
$$\mathrm{var}(\nu_1) = \sigma^2(1+\theta^2) = \sigma^2 f_1.$$

The updating equations give

$$a_1 = a_{1|0} + P_{1|0} z_1 (y_1 - a_{1|0})/f_1 = \begin{pmatrix} x_1 \\ x_1\theta/(1+\theta^2) \end{pmatrix},$$

and

$$P_1 = \begin{pmatrix} 1+\theta^2 & \theta \\ \theta & \theta^2 \end{pmatrix}.$$

The prediction eqs then yield

$$a_{2|1} = \begin{pmatrix} x_1\theta/(1+\theta^2) \\ 0 \end{pmatrix} \quad \text{and} \quad P_{2|1} = \begin{pmatrix} (1+\theta^2+\theta^4)/(1+\theta^2) & \theta \\ \theta & \theta^2 \end{pmatrix},$$

and so

$$\nu_2 = x_2 - y_1\theta/(1+\theta^2) \qquad \text{and} \qquad f_2 = (1+\theta^2+\theta^4)/(1+\theta^2).$$

7.6.2 Example 2

Consider the ARMA(1,1) model:

$$x_t = \phi x_{t-1} + e_t + \theta e_{t-1}, \qquad e_t \sim (0, \sigma^2).$$

Then

$$\begin{pmatrix} \alpha_{t,1} \\ \alpha_{t,2} \end{pmatrix} = \begin{pmatrix} \phi & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \alpha_{t-1,1} \\ \alpha_{t-1,2} \end{pmatrix} + \begin{pmatrix} 1 \\ \theta \end{pmatrix} e_t,$$

and

$$x_t = (1 \quad 0)\alpha_t.$$

Starting with $a_{1|0} = a_0 = 0$, we have

$$\tilde{x}_{1|0} = 0, \quad \nu_1 = x_1,$$

$$\sigma^2 f_1 = \text{var}(\nu_1) = \text{var}(x_1) = c(0),$$

$$\sigma^2 P_0 = \text{cov} \begin{pmatrix} x_1 \\ \theta e_1 \end{pmatrix} = \begin{pmatrix} c(0) & \sigma^2\theta \\ \sigma^2\theta & \sigma^2\theta^2 \end{pmatrix},$$

where

$$c(0) = \sigma^2 \frac{\theta^2 + 2\theta\phi + 1}{1 - \phi^2} = \sigma^2 f_1.$$

Also,

$$P_1 = \begin{pmatrix} f_1 & \theta \\ \theta & \theta^2/f_1 \end{pmatrix},$$

$$a_1 = \begin{pmatrix} x_1 \\ \theta x_1/f_1 \end{pmatrix}.$$

Note. The above should be verified directly as an exercise.

7.7 Computing the likelihood function

The likelihood function of an ARMA(p,q) process can be obtained via the state space approach as illustrated through the examples considered in the last section. It may be also helpful to review the section on computing the autocovariance function of ARMA process, which is needed for the updating and predicting equations. See ARMA(1,1) in Example 2.

7.8 Examples

```
/*
  STATE SPACE EXAMPLE 1
*/

options ls=70 ps=100 nodate pageno=1;
title 'SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS';
*  Box-Jenkins page 525;

data seriesa;
  infile 'c:/z2/b259/sas/chem.dat';
  n=_n_;
  input x :3.1 @@;

* proc arima to fit model and forecast;
proc arima data=seriesa;
   identify variable=x noprint;
   estimate p=1 q=1;
   forecast lead=10 out=out1 noprint;
run;

* proc statespace to select and fit;
proc statespace data=seriesa cancorr printout=long;
   var x;
run;

* proc statespace to fit and forecast;
proc statespace data=seriesa printout=short
       lead=10 out=out2 outmodel=model;
   var x;
   restrict f(2,1)=0;
run;

proc print data=model;         * print model data set;
   title 'MODEL FROM PROC STATESPACE';
run;

data;
   merge out1 out2;
run;
```

```
proc print data=_last_(firstobs=180);  * compare forecasts;
run;
```

```
          SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS           1

                          ARIMA Procedure


                 Conditional Least Squares Estimation

                                    Approx.
           Parameter   Estimate    Std Error   T Ratio  Lag
           MU          17.03883      0.09510    179.17   0
           MA1,1        0.55797      0.08828      6.32   1
           AR1,1        0.90365      0.04670     19.35   1

           Constant Estimate  = 1.64168984

           Variance  Estimate = 0.09965051
           Std Error Estimate = 0.31567469
           AIC                =  107.73974*
           SBC                = 117.589351*
           Number of Residuals=        197
           * Does not include log determinant.

                     Correlations of the Estimates


            Parameter           MU        MA1,1       AR1,1

            MU               1.000        0.004      -0.025
            MA1,1            0.004        1.000       0.738
            AR1,1           -0.025        0.738       1.000


                  Autocorrelation Check of Residuals

    To   Chi                   Autocorrelations
   Lag  Square DF   Prob
     6    4.90  4  0.297  0.041  0.004 -0.090 -0.082 -0.085  0.021
    12   14.75 10  0.141  0.168  0.039  0.065  0.021 -0.071 -0.089
    18   22.43 16  0.130  0.000  0.110 -0.107  0.014  0.061  0.091
```

```
24   27.25 22  0.202 -0.036  0.094 -0.085 -0.009 -0.046  0.046
30   36.99 28  0.119  0.042  0.043  0.118 -0.090  0.039 -0.121
36   50.37 34  0.035  0.030  0.122  0.119  0.068 -0.145 -0.016


                  Model for variable X

                  Estimated Mean = 17.0388318

                  Autoregressive Factors
                  Factor 1: 1 - 0.90365 B**(1)


                  Moving Average Factors
                  Factor 1: 1 - 0.55797 B**(1)

         SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS         2

                        STATESPACE Procedure

                            Nobs = 197

                   Variable      Mean       Std
                   X         17.06244  0.399247


          Information Criterion for Autoregressive Models


          Lag=0         Lag=1         Lag=2         Lag=3
    -361.761017   -437.216070   -448.124845   -447.046264


          Lag=4         Lag=5         Lag=6         Lag=7
    -445.994023   -444.848471   -445.884051   -448.755121


          Lag=8         Lag=9        Lag=10
    -446.953416   -444.972197   -443.025084
```

```
                 Var/Cov Matrices of the Input Series


              Lag=0         Lag=1         Lag=2         Lag=3
                  X             X             X             X
      X    0.159398      0.090883      0.078912      0.063433


              Lag=4         Lag=5         Lag=6         Lag=7
                  X             X             X             X
      X    0.056697      0.052105      0.055751      0.062466


              Lag=8         Lag=9        Lag=10
                  X             X             X
      X    0.051247      0.048478      0.040626



               Correlation Matrices of the Input Series


         Lag=0      Lag=1      Lag=2      Lag=3      Lag=4      Lag=5
             X          X          X          X          X          X
 X     1.00000    0.57016    0.49506    0.39795    0.35570    0.32688


         Lag=6      Lag=7      Lag=8      Lag=9     Lag=10
             X          X          X          X          X
 X     0.34976    0.39189    0.32150    0.30413    0.25487



               Schematic Representation of Correlations

             Name/Lag  0  1  2  3  4  5  6  7  8  9 10
             X         +  +  +  +  +  +  +  +  +  +  +

       + is > 2*std error,  - is < -2*std error,  . is between
```

```
                    Partial Autoregressive Matrices


              Lag=1          Lag=2          Lag=3          Lag=4
                  X              X              X              X
X         0.5701648      0.2518453      0.0683106      0.0692777


              Lag=5          Lag=6          Lag=7          Lag=8
                  X              X              X              X
X         0.0657869      0.1236565      0.1562788     -0.0317185


              Lag=9         Lag=10
                  X              X
X         0.0097637     -0.0163838



          Schematic Representation of Partial Autocorrelations

                Name/Lag   1  2  3  4  5  6  7  8  9 10
                X          +  +  .  .  .  .  +  .  .  .

   + is > 2*std error,  - is < -2*std error,  . is between

          SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS          3

                          STATESPACE Procedure

                 Yule-Walker Estimates for the Min AIC


              Lag=1          Lag=2          Lag=3          Lag=4
                  X              X              X              X
X           0.37262        0.19682        0.02035        0.01375


              Lag=5          Lag=6          Lag=7
                  X              X              X
X          -0.01484        0.06240        0.15628
```

```
                 Variance/Covariance of the Residuals


              Lag=0          Lag=1          Lag=2          Lag=3
                  X              X              X              X
     X      0.15940        0.10758        0.10076        0.10029


              Lag=4          Lag=5          Lag=6          Lag=7
                  X              X              X              X
     X      0.09980        0.09937        0.09785        0.09546


              Lag=8          Lag=9         Lag=10
                  X              X              X
     X      0.09537        0.09536        0.09533



                 Residual Correlations from AR Models


        Lag=0       Lag=1       Lag=2       Lag=3       Lag=4       Lag=5
            X           X           X           X           X           X
X     1.00000     1.00000     1.00000     1.00000     1.00000     1.00000


        Lag=6       Lag=7       Lag=8       Lag=9      Lag=10
            X           X           X           X           X
X     1.00000     1.00000     1.00000     1.00000     1.00000

          SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS          4

                         STATESPACE Procedure

                   Canonical Correlations Analysis

    State vector            Correlations        Infor.    Chisq  D.F.
```

```
X(T;T),X(T+1;T)           1.0000 0.3369          9.737346  23.31562   7

X(T;T),X(T+1;T),X(T+2;T) 1.0000 0.3699 0.1675   -6.39369  5.520937   6
          SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS           5

                         STATESPACE Procedure

         Selected Statespace Form and Preliminary Estimates

                             State Vector

                    X(T;T)           X(T+1;T)

                  Estimate of the Transition Matrix

                               0       1

                          -0.036   0.932


                   Input Matrix for the Innovation

                                   1

                                0.373


                  Variance Matrix for the Innovation

                            0.09546355

          SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS           6

                         STATESPACE Procedure

              Selected Statespace Form and Fitted Model

                             State Vector

                    X(T;T)           X(T+1;T)

                  Estimate of the Transition Matrix
```

```
                 0       1

            -0.166   1.109


      Input Matrix for the Innovation

                     1

                 0.392


     Variance Matrix for the Innovation

              0.09866764


             Parameter Estimates

   Parameter Estimate  Std. Err.  T value

   F(2,1)    -0.16564  0.117181  -1.41352
   F(2,2)    1.108647  0.146643  7.560155
   G(2,1)    0.391536  0.069791  5.610135
SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS          7

             STATESPACE Procedure

                 Nobs = 197

         Variable       Mean       Std
         X          17.06244  0.399247


  Information Criterion for Autoregressive Models


     Lag=0          Lag=1          Lag=2          Lag=3
-361.761017   -437.216070   -448.124845   -447.046264
```

```
       Lag=4         Lag=5         Lag=6         Lag=7
 -445.994023   -444.848471   -445.884051   -448.755121


       Lag=8         Lag=9        Lag=10
 -446.953416   -444.972197   -443.025084



          Schematic Representation of Correlations

        Name/Lag  0  1  2  3  4  5  6  7  8  9 10
        X         +  +  +  +  +  +  +  +  +  +  +

 + is > 2*std error,  - is < -2*std error,  . is between



    Schematic Representation of Partial Autocorrelations

          Name/Lag  1  2  3  4  5  6  7  8  9 10
          X         +  +  .  .  .  .  +  .  .  .

 + is > 2*std error,  - is < -2*std error,  . is between



           Yule-Walker Estimates for the Min AIC


           Lag=1         Lag=2         Lag=3         Lag=4
               X             X             X             X
X        0.37262       0.19682       0.02035       0.01375


           Lag=5         Lag=6         Lag=7
               X             X             X
X       -0.01484       0.06240       0.15628

     SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS         8

                    STATESPACE Procedure
```

```
               Selected Statespace Form and Preliminary Estimates

                                  State Vector

                           X(T;T)          X(T+1;T)

                        Estimate of the Transition Matrix

                                        0       1

                                    0.000   0.932


                         Input Matrix for the Innovation

                                            1

                                        0.373


                        Variance Matrix for the Innovation

                                    0.09546355

                SERIES-A: CHEMICAL PROCESS CONCENTRATION READINGS          9

                               STATESPACE Procedure

                    Selected Statespace Form and Fitted Model

                                  State Vector

                           X(T;T)          X(T+1;T)

                        Estimate of the Transition Matrix

                                        0       1

                                    0.000   0.883
```

```
                Input Matrix for the Innovation

                                1

                              0.366


               Variance Matrix for the Innovation

                            0.0992835


                      Parameter Estimates

             Parameter Estimate  Std. Err.  T value

             F(2,2)    0.882604  0.049766  17.73507
             G(2,1)    0.365824  0.063382  5.771711
                   MODEL FROM PROC STATESPACE                    10

   S
   T
   A
   T                                                         S
   E                                      F                  I
O  V        F     F     F  F  F  F  F  F  F  _     G         G
B  E        _     _     _  _  _  _  _  _  _  1     _         _
S  C        1     2     3  4  5  6  7  8  9  0     1         1

1  X(T;T)   0  1.00000  .  .  .  .  .  .  .  .  1.00000  0.099284
2  STD      .  .        .  .  .  .  .  .  .  .   .          .
3  X(T+1;T) 0  0.88260  .  .  .  .  .  .  .  .  0.36582    .
4  STD      .  0.04977  .  .  .  .  .  .  .  .  0.06338    .
                   MODEL FROM PROC STATESPACE                    11

    OBS      X     FORECAST      STD        L95        U95

    180    17.4     17.1276    0.31567    16.5089    17.7463
    181    17.5     17.2132    0.31567    16.5945    17.8319
    182    17.9     17.2955    0.31567    16.6768    17.9143
    183    17.0     17.4798    0.31567    16.8611    18.0985
    184    17.0     17.2714    0.31567    16.6527    17.8901
```

185	17.0	17.1552	0.31567	16.5365	17.7739
186	17.2	17.0903	0.31567	16.4716	17.7090
187	17.3	17.1233	0.31567	16.5046	17.7420
188	17.4	17.1762	0.31567	16.5575	17.7949
189	17.4	17.2403	0.31567	16.6216	17.8591
190	17.0	17.2761	0.31567	16.6574	17.8948
191	18.0	17.1578	0.31567	16.5391	17.7765
192	18.2	17.4375	0.31567	16.8188	18.0562
193	17.6	17.6627	0.31567	17.0439	18.2814
194	17.8	17.5809	0.31567	16.9622	18.1996
195	17.7	17.6044	0.31567	16.9857	18.2231
196	17.2	17.5830	0.31567	16.9642	18.2017
197	17.4	17.3981	0.31567	16.7794	18.0169
198	.	17.3642	0.31567	16.7455	17.9829
199	.	17.3328	0.33400	16.6782	17.9875
200	.	17.3045	0.34826	16.6219	17.9871
201	.	17.2789	0.35948	16.5743	17.9835
202	.	17.2558	0.36838	16.5338	17.9778
203	.	17.2349	0.37550	16.4989	17.9708
204	.	17.2160	0.38122	16.4688	17.9631
205	.	17.1989	0.38582	16.4427	17.9551
206	.	17.1835	0.38954	16.4200	17.9470
207	.	17.1696	0.39255	16.4002	17.9389

OBS	RESIDUAL	FOR1	RES1	STD1
180	0.27241	17.1274	0.27263	0.31509
181	0.28680	17.2195	0.28052	0.31509
182	0.60446	17.3037	0.59633	0.31509
183	-0.47976	17.4935	-0.49350	0.31509
184	-0.27143	17.2624	-0.26236	0.31509
185	-0.15519	17.1429	-0.14291	0.31509
186	0.10967	17.0812	0.11882	0.31509
187	0.17672	17.1224	0.17755	0.31509
188	0.22377	17.1804	0.21964	0.31509
189	0.15965	17.2469	0.15314	0.31509
190	-0.27612	17.2812	-0.28123	0.31509
191	0.84219	17.1527	0.84733	0.31509
192	0.76252	17.4520	0.74795	0.31509
193	-0.06266	17.6799	-0.07993	0.31509
194	0.21911	17.5782	0.22180	0.31509
195	0.09559	17.5988	0.10121	0.31509

```
       196    -0.38296     17.5728    -0.37285    0.31509
       197     0.00185     17.3765     0.02347    0.31509
       198      .          17.3482      .         0.31509
       199      .          17.3147      .         0.33552
       200      .          17.2851      .         0.35060
       201      .          17.2589      .         0.36192
       202      .          17.2359      .         0.37049
       203      .          17.2155      .         0.37704
       204      .          17.1975      .         0.38206
       205      .          17.1817      .         0.38592
       206      .          17.1677      .         0.38891
       207      .          17.1553      .         0.39122

/*
   STATE SPACE EXAMPLE 2: Gas Furnace Data
*/

options ls=70 ps=100 nodate pageno=1;
title1 '-------------------Gas Furnace Data--------------';
title2 '----------------------Series J------------------';
data seriesj;
   infile 'c:/z2/b259/sas/gasfurn.dat';
   input x y @@;
   label X='Input Gas Rate' Y='Output CO2';

*  default model and descriptive statistics;
proc statespace cancorr printout=long;
   var x y;
run;

* transfer function model derived from the default model;
proc statespace residest printout=none outmodel=model1;
   var x y;
   restrict f(3,2)=0 f(3,4)=0 g(3,2)=0 g(4,1)=0 g(5,1)=0;
run;


* more complicated transfer function model;
proc statespace residest printout=none outmodel=model2
   data=seriesj maxit=30;
   var x y;
   form x 3 y 3;
   restrict f(5,2)=0 f(5,4)=0 f(5,6)=0
```

```
            g(3,2)=0 g(4,1)=0 g(5,2)=0 g(6,1)=0;
run;
```

```
          -------------------Gas Furnace Data--------------          1
          ----------------------Series J------------------

                       STATESPACE Procedure

                           Nobs = 296

                    Variable      Mean       Std
                    X         -0.05683  1.072766
                    Y         53.50912  3.202121


           Information Criterion for Autoregressive Models


         Lag=0         Lag=1         Lag=2         Lag=3
    651.386156   -1033.57211   -1632.95659   -1645.12498


         Lag=4         Lag=5         Lag=6         Lag=7
   -1651.52305   -1648.91248   -1649.34335   -1643.15323


         Lag=8         Lag=9        Lag=10
   -1638.55591   -1634.79898   -1633.58501



                 Var/Cov Matrices of the Input Series


              Lag=0                      Lag=1
                  X           Y              X           Y
   X       1.150826   -1.664149       1.096133   -1.351609
   Y      -1.664149   10.253578      -2.055596    9.953729


              Lag=2                      Lag=3
                  X           Y              X           Y
```

X	0.959895	-1.128583	0.784702	-0.983929
Y	-2.490580	9.187612	-2.895192	8.126456

	Lag=4		Lag=5	
	X	Y	X	Y
X	0.611356	-0.894337	0.468964	-0.834292
Y	-3.176091	6.972062	-3.264466	5.890450

	Lag=6		Lag=7	
	X	Y	X	Y
X	0.366194	-0.778798	0.299439	-0.707868
Y	-3.141743	4.977579	-2.848818	4.266303

	Lag=8		Lag=9	
	X	Y	X	Y
X	0.261827	-0.616452	0.245206	-0.510214
Y	-2.461337	3.748784	-2.059646	3.387600

	Lag=10	
	X	Y
X	0.239752	-0.405873
Y	-1.700399	3.142934

Correlation Matrices of the Input Series

	Lag=0		Lag=1		Lag=2	
	X	Y	X	Y	X	Y
X	1.00000	-0.48445	0.95247	-0.39347	0.83409	-0.32854
Y	-0.48445	1.00000	-0.59841	0.97076	-0.72503	0.89604

	Lag=3		Lag=4		Lag=5	
	X	Y	X	Y	X	Y
X	0.68186	-0.28643	0.53123	-0.26035	0.40750	-0.24287
Y	-0.84282	0.79255	-0.92459	0.67996	-0.95032	0.57448

```
         Lag=6                Lag=7                Lag=8
             X        Y           X        Y           X        Y
X      0.31820 -0.22672     0.26019 -0.20607     0.22751 -0.17946
Y     -0.91459  0.48545    -0.82932  0.41608    -0.71652  0.36561


         Lag=9               Lag=10
             X        Y           X        Y
X      0.21307 -0.14853     0.20833 -0.11815
Y     -0.59958  0.33038    -0.49500  0.30652



            Schematic Representation of Correlations

    Name/Lag    0   1   2   3   4   5   6   7   8   9  10
    X          +-  +-  +-  +-  +-  +-  +-  +-  +-  +-  +-

      -------------------Gas Furnace Data--------------         2
      -----------------------Series J------------------

                      STATESPACE Procedure

            Schematic Representation of Correlations
    Y          -+  -+  -+  -+  -+  -+  -+  -+  -+  -+  -+

   + is > 2*std error,  - is < -2*std error,  . is between



                Partial Autoregressive Matrices


                Lag=1                      Lag=2
                    X            Y             X            Y
    X       0.9954945    0.0297498    -0.9367187    0.0335753
    Y      -0.4997101    0.8896541    -0.7787835   -0.5139065


                Lag=3                      Lag=4
```

```
                X           Y              X           Y
X       0.2595691   0.0051259      0.1042360   0.0032683
Y      -0.4619783   0.0047656      0.1954108   0.1334175


            Lag=5                      Lag=6
                X           Y              X           Y
X       0.0791885   0.0016635     -0.2023202   0.0186976
Y       0.2013794   0.0585304      0.1726081  -0.0291720


            Lag=7                      Lag=8
                X           Y              X           Y
X      -0.0395721   0.0192073      0.0956312  -0.0183033
Y      -0.1181590  -0.0169549     -0.0818077   0.0473716


            Lag=9                     Lag=10
                X           Y              X           Y
X       0.0961719  -0.0023803     -0.0034968  -0.0225286
Y      -0.1790565   0.0470629     -0.0206584   0.1038925



  Schematic Representation of Partial Autocorrelations

   Name/Lag    1    2    3    4    5    6    7    8    9   10
   X          +.   -.   +.   ..   ..   -.   ..   ..   ..   ..
   Y          -+   --   -.   .+   ..   ..   ..   ..   ..   .+

+ is > 2*std error,  - is < -2*std error,  . is between



         Yule-Walker Estimates for the Min AIC


            Lag=1                      Lag=2
                X           Y              X           Y
X         1.92589    -0.00124       -1.20166     0.00422
Y         0.05050     1.29979       -0.02046    -0.32770
```

```
          Lag=3                    Lag=4
              X          Y             X          Y
X       0.11692   -0.00867       0.10424    0.00327
Y      -0.71182   -0.25701       0.19541    0.13342


            Variance/Covariance of the Residuals


          Lag=0                    Lag=1
              X          Y             X          Y
X       1.15083   -1.66415       0.09984    0.08606
Y      -1.66415   10.25358       0.08606    0.37100


          Lag=2                    Lag=3
              X          Y             X          Y
X       0.03679   -0.00926       0.03547   -0.00691
Y      -0.00926    0.10581      -0.00691    0.10161


          Lag=4                    Lag=5
              X          Y             X          Y
X       0.03527   -0.00734       0.03516   -0.00763
Y      -0.00734    0.09757      -0.00763    0.09627


          Lag=6                    Lag=7
              X          Y             X          Y
X       0.03434   -0.00687       0.03424   -0.00689
Y      -0.00687    0.09554      -0.00689    0.09526

   -------------------Gas Furnace Data--------------          3
   -----------------------Series J------------------

                    STATESPACE Procedure

            Variance/Covariance of the Residuals
```

	Lag=8 X	Lag=8 Y	Lag=9 X	Lag=9 Y
X	0.03400	-0.00658	0.03384	-0.00625
Y	-0.00658	0.09470	-0.00625	0.09369

	Lag=10 X	Lag=10 Y
X	0.03375	-0.00582
Y	-0.00582	0.09169

Residual Correlations from AR Models

	Lag=0 X	Lag=0 Y	Lag=1 X	Lag=1 Y	Lag=2 X	Lag=2 Y
X	1.00000	-0.48445	1.00000	0.44718	1.00000	-0.14837
Y	-0.48445	1.00000	0.44718	1.00000	-0.14837	1.00000

	Lag=3 X	Lag=3 Y	Lag=4 X	Lag=4 Y	Lag=5 X	Lag=5 Y
X	1.00000	-0.11515	1.00000	-0.12508	1.00000	-0.13112
Y	-0.11515	1.00000	-0.12508	1.00000	-0.13112	1.00000

	Lag=6 X	Lag=6 Y	Lag=7 X	Lag=7 Y	Lag=8 X	Lag=8 Y
X	1.00000	-0.11993	1.00000	-0.12074	1.00000	-0.11595
Y	-0.11993	1.00000	-0.12074	1.00000	-0.11595	1.00000

	Lag=9 X	Lag=9 Y	Lag=10 X	Lag=10 Y
X	1.00000	-0.11096	1.00000	-0.10461
Y	-0.11096	1.00000	-0.10461	1.00000

```
-------------------Gas Furnace Data--------------         4
----------------------Series J------------------
```

```
                         STATESPACE Procedure

                   Canonical Correlations Analysis

     State vector             Correlations         Infor.    Chisq  D.F.

X(T;T),Y(T;T),X(T+1;T)   1.0000 1.0000 0.8049   292.9228  304.7481   8

X(T;T),Y(T;T),X(T+1;T),  1.0000 1.0000 0.9067   122.3358  134.7237   7
Y(T+1;T)                 0.6075

X(T;T),Y(T;T),X(T+1;T),  1.0000 1.0000 0.9094   -1.54701  10.34705   6
Y(T+1;T),X(T+2;T)        0.6103 0.1863

X(T;T),Y(T;T),X(T+1;T),  1.0000 1.0000 0.9101   0.940392  12.80924   6
Y(T+1;T),Y(T+2;T)        0.6189 0.2068

X(T;T),Y(T;T),X(T+1;T),  1.0000 1.0000 0.9130   -7.94103  2.041584   5
Y(T+1;T),Y(T+2;T),       0.6288 0.2266 0.0833
Y(T+3;T)

            -------------------Gas Furnace Data--------------          5
            -----------------------Series J------------------

                         STATESPACE Procedure

          Selected Statespace Form and Preliminary Estimates

                             State Vector

   X(T;T)           Y(T;T)           X(T+1;T)          Y(T+1;T)
   Y(T+2;T)

                   Estimate of the Transition Matrix

                        0        0        1        0        0

                        0        0        0        1        0

                   -0.847    0.027    1.712   -0.050        0

                        0        0        0        0        1
```

```
         -0.198   0.334  -0.182  -1.236   1.787


          Input Matrix for the Innovation

                         1       0

                         0       1

                     1.926  -0.001

                     0.050   1.300

                     0.142   1.362


         Variance Matrix for the Innovation

                 0.0352738 -0.0073378

                -0.0073378 0.09756908

--------------------Gas Furnace Data--------------          6
-----------------------Series J------------------

                STATESPACE Procedure

        Selected Statespace Form and Fitted Model

                    State Vector

X(T;T)          Y(T;T)          X(T+1;T)        Y(T+1;T)
Y(T+2;T)

          Estimate of the Transition Matrix

             0       0       1       0       0

             0       0       0       1       0

         -0.862   0.031   1.724  -0.055       0
```

0	0	0	0	1
-0.348	0.292	-0.094	-1.098	1.671

Input Matrix for the Innovation

1	0
0	1
1.924	-0.004
0.016	1.258
0.081	1.353

Variance Matrix for the Innovation

0.03557876	-0.007285
-0.007285	0.09557739

Parameter Estimates

Parameter	Estimate	Std. Err.	T value
F(3,1)	-0.86192	0.072961	-11.8135
F(3,2)	0.030609	0.026167	1.16975
F(3,3)	1.724235	0.061599	27.99136
F(3,4)	-0.05483	0.030169	-1.81726
F(5,1)	-0.34839	0.135253	-2.57582
F(5,2)	0.292124	0.046299	6.309469
F(5,3)	-0.09435	0.096527	-0.9774
F(5,4)	-1.09823	0.109525	-10.0272
F(5,5)	1.671418	0.083737	19.9604
G(3,1)	1.92442	0.058162	33.08744
G(3,2)	-0.00416	0.035255	-0.11812
G(4,1)	0.015621	0.095771	0.163107

```
G(4,2)    1.258495  0.055742  22.57733
G(5,1)     0.08058  0.151622  0.531452
G(5,2)    1.353204  0.091388  14.80719

-------------------Gas Furnace Data--------------          7
-----------------------Series J------------------

               STATESPACE Procedure

                   Nobs = 296

         Variable       Mean       Std
         X          -0.05683  1.072766
         Y          53.50912  3.202121

-------------------Gas Furnace Data--------------          8
-----------------------Series J------------------

               STATESPACE Procedure

  Selected Statespace Form and Preliminary Estimates

                   State Vector

X(T;T)          Y(T;T)          X(T+1;T)        Y(T+1;T)
Y(T+2;T)

         Estimate of the Transition Matrix

              0       0       1       0       0

              0       0       0       1       0

         -0.847   0.000   1.712   0.000       0

              0       0       0       0       1

         -0.198   0.334  -0.182  -1.236   1.787


         Input Matrix for the Innovation
```

```
                                  1       0

                                  0       1

                              1.926   0.000

                              0.000   1.300

                              0.000   1.362


                     Variance Matrix for the Innovation

                          0.0352738 -0.0073378

                         -0.0073378 0.09756908

WARNING: No improvement after 10 step halvings.
Convergence has been assumed.

          -------------------Gas Furnace Data--------------         9
          -----------------------Series J------------------

                            STATESPACE Procedure

                  Selected Statespace Form and Fitted Model

                                State Vector

   X(T;T)           Y(T;T)           X(T+1;T)         Y(T+1;T)
   Y(T+2;T)

                     Estimate of the Transition Matrix

                          0       0       1       0       0

                          0       0       0       1       0

                     -0.689   0.000   1.599   0.000       0

                          0       0       0       0       1
```

```
-0.181   0.354  -0.196  -1.260   1.796


 Input Matrix for the Innovation

                 1       0

                 0       1

             1.928   0.000

             0.000   1.526

             0.000   1.787


Variance Matrix for the Innovation

        0.03758031 -0.0032196

        -0.0032196 0.05849685


       Parameter Estimates

Parameter Estimate  Std. Err.  T value

F(3,1)    -0.68918  0.051094  -13.4884
F(3,3)    1.598829  0.051505  31.04201
F(5,1)    -0.18075  0.158734   -1.1387
F(5,2)     0.35445  0.074939  4.729819
F(5,3)    -0.19586  0.099653  -1.96543
F(5,4)    -1.25974  0.187437   -6.7209
F(5,5)    1.795796  0.137032  13.10494
G(3,1)    1.927878  0.057183  33.71444
G(4,2)     1.52605  0.057969  26.32547
G(5,2)    1.786529  0.106696  16.74413


-------------------Gas Furnace Data--------------         10
----------------------Series J------------------
```

```
                    STATESPACE Procedure

                        Nobs = 296

               Variable      Mean       Std
               X         -0.05683  1.072766
               Y         53.50912  3.202121


     -------------------Gas Furnace Data--------------          11
     -----------------------Series J------------------

                    STATESPACE Procedure

     Selected Statespace Form and Preliminary Estimates

                        State Vector

X(T;T)           Y(T;T)           X(T+1;T)         Y(T+1;T)
X(T+2;T)         Y(T+2;T)

             Estimate of the Transition Matrix

              0       0       1       0       0       0

              0       0       0       1       0       0

              0       0       0       0       1       0

              0       0       0       0       0       1

          0.432   0.000  -1.913   0.000   2.360   0.000

          0.331   0.281  -1.350  -1.108   0.702   1.719


              Input Matrix for the Innovation

                              1       0

                              0       1
```

```
                        1.926   0.000

                        0.000   1.300

                        2.507   0.000

                        0.000   1.362


               Variance Matrix for the Innovation

                    0.0352738 -0.0073378

                   -0.0073378 0.09756908

WARNING: Estimates give unstable system.

WARNING: No improvement after 10 step halvings.
Convergence has been assumed.
          -------------------Gas Furnace Data--------------         12
          -----------------------Series J------------------

                      STATESPACE Procedure

            Selected Statespace Form and Fitted Model

                          State Vector

   X(T;T)           Y(T;T)           X(T+1;T)         Y(T+1;T)
   X(T+2;T)         Y(T+2;T)

               Estimate of the Transition Matrix

                0       0       1       0       0       0

                0       0       0       1       0       0

                0       0       0       0       1       0

                0       0       0       0       0       1

              0.566   0.000  -1.844   0.000   2.245   0.000
```

```
0.886   0.302  -2.392  -1.156   1.295   1.771


        Input Matrix for the Innovation

                        1       0

                        0       1

                    1.928   0.000

                    0.000   1.531

                    2.516   0.000

                    0.000   1.790


       Variance Matrix for the Innovation

               0.03626623 -0.0032934

               -0.0032934 0.05790786


               Parameter Estimates

 Parameter Estimate  Std. Err.  T value

 F(5,1)    0.565729    0.1084  5.218901
 F(5,3)    -1.84413  0.246644  -7.47688
 F(5,5)    2.244939  0.154789   14.5032
 F(6,1)    0.886417  0.560639  1.581084
 F(6,2)    0.301564  0.092662  3.254434
 F(6,3)    -2.39174   1.21216  -1.97312
 F(6,4)    -1.15601  0.239587    -4.825
 F(6,5)    1.295381  0.733398  1.766273
 F(6,6)    1.770833  0.167794  10.55362
 G(3,1)    1.928018  0.058873  32.74869
 G(4,2)    1.530783  0.058552  26.14422
 G(5,1)    2.516393  0.126382  19.91106
```

```
G(6,2)      1.78995   0.10783  16.59975
```

Chapter 8

Advanced Topics

8.1 A closer look at the inconsistency of $I_n(\lambda)$

From previous the section, we know that the periodograms are very erratic. This is due to the inconsistency caused by the fact that

$$\text{var}(I_n(\lambda)) \doteq f(\lambda)^2.$$

It is instructive to examine closely why the variance of the periodogram $I_n(\lambda)$ would not tend to zero as n becomes large. Recall that

$$I_n(\lambda) = \frac{1}{2\pi} \sum_{-n+1}^{n-1} \hat{R}(u) e^{-iu\lambda},$$

and

$$f(\lambda) = \frac{1}{2\pi} \sum_{-\infty}^{\infty} \hat{R}(u) e^{-iu\lambda}.$$

At first glance, one would speculate that the insconsistency was caused by the truncation in the periodogram. This is not really the reason as can be seen below. The variance may be due to the fact that $\hat{R}(u)$ (unbiased version) has a variance proportional to $1/(n-u)$ which becomes large as u tends to n. However, the version we used in computing the periodogram is the biased one, its bias becomes large when u tends to n. Note also the biased estimator has a variance proportional to $1/n$. Thus the variance of the periodogram can not be contributed by the bias of $\hat{R}(u)$. The main reason lies in the fact that the variance of the periodogram is equal to the sum of the variance of each $\hat{R}(u)$, and since each of these has the order of $1/n$, the sum becomes $O(1)$. Our asymptotic result indicates that this is proportional to $f(\lambda)^2$. To remedy this problem, one suggestion is to remove a number of the $\hat{R}(u)$ in computing the periodogram, which leads to the following

$$\hat{f}(\lambda) = \frac{1}{2\pi} \sum_{-n+1}^{n-1} w(u) \hat{R}(u) e^{-iu\lambda},$$

where $w(u)$ (called *the lag window*) is a positive function with support on a finite interal. For example, $w(u) = 1_{[-M,M]}(u)$. By

$$\hat{R}(u) = \int_{-\pi}^{\pi} I_n(\omega) e^{iu\omega} d\omega,$$

the above estimator can be written as follows:

$$\hat{f}(\lambda) = \int I_n(\omega) \left\{ \frac{1}{2\pi} \sum_{-n+1}^{n-1} w(u) e^{-iu(\lambda-\omega)} \right\} d\omega.$$

Deinfe *the spectral window* by

$$W(\theta) = \frac{1}{2\pi} \sum_{-n+1}^{n-1} w(u) e^{-iu(\theta)}.$$

Then we obtain another smooth spectral density esitmate:

$$\hat{f}(\lambda) = \int I_n(\omega) W(\omega - \lambda) d\omega.$$

Table 8.1 *Window generators*

`ioptw`	$w(u)$	$W(\omega)$	$\int w^2(u)\,du$
1	1	$\frac{\sin\omega}{\pi\omega}$	2
2	$1-\lvert u\rvert$	$\frac{1}{2\pi}\left(\frac{\sin\omega/2}{\omega/2}\right)^2$	2/3
3	$.54+.46\cos\pi u$	$\frac{1}{2\pi}\frac{\sin\omega}{\omega}\frac{\pi^2}{\pi^2-\omega^2}$	.795
4	(cf below)	$\frac{3}{8\pi}\left(\frac{\sin\omega/4}{\omega/4}\right)^2$	.539
5	$(1-\lvert u\rvert)\cos\pi u+\frac{1}{\pi}\sin\pi\lvert u\rvert$	$\frac{2\pi(1+\cos\omega)}{(\pi^2-\omega^2)^2}$	.586
6	$\frac{\sin\pi u}{\pi u}$	$\frac{1}{2\pi}$	1
7	$\frac{3}{(\pi u)^2}(\frac{\sin\pi u}{\pi u}-\cos\pi u)$	$\frac{3}{4\pi}(1-(\frac{\omega}{\pi})^2)$	1.2
8	$1/(1+u^4)$	$\frac{1}{2}e^{-\lvert\omega\rvert/\sqrt{2}}\sin(\frac{\lvert\omega\rvert}{\sqrt{2}}+\frac{\pi}{4}$	1.66

For option 4, we have

$$w(u) = \begin{cases} 1 - 6u^2 + 6|u|^3, & |u| \le .5 \\ 2(1-|u|)^3, & .5 \le |u| \le 1. \end{cases}$$

The window estimate is given by

$$\hat{f}_{w,M}(\omega) = \sum_v w\left(\frac{v}{M}\right) \hat{R}(v) e^{-2\pi i v\omega},$$

where M is the window scale parameter. The estimate is smooth when M is small while it is rough when M is large. The following functions perform the window estimate of the spectral density function using one of the eight methods (`ioptw=1,...,8`).

The estimate also depends on the scale parameter `M` in a crucial manner, which will be discussed shortly. The function `auspe` plots the estimate and its confidence intervals based on either logarithmic (`logs=T`) or (default) ordinary scale. It uses another function `spe` for computing the estimates.

```
x <- arima.sim(list(order(2,0,0),ar=c(.5,.25)),n=200)
par(mfrow=c(4,4))
for (i in 1:8) auspe(x,ioptw=i)         # Fig 8.2
for (i in 1:8) auspe(x,M=i*10)           # Fig 8.2
```

Note the increasing roughness of the resulting estimates with `M`.

Under appropriate conditions, it can be shown that

$$\sqrt{\frac{n}{M}}[\log \hat{f}_{w,M}(\omega) - \log f(\omega)] \to N\Big(0, \int w^2(u)\,du\Big).$$

Choice of M

Here `M` is the scale parameter and is used to control the smoothness of the estimates. Large `M` yields rough estimate which means less biased and large variance, while small `M` yields smoother estimate with more biased and less variance. In practice one should try several values of `M` and the graphs to make general statement about the underlying spectral density function. Nevertheless, Parzen (1967) suggested using the following three values

$$.05 \leq \frac{M_1}{n} \leq .10, \qquad .10 \leq \frac{M_1}{n} \leq .25, \qquad .25 \leq \frac{M_1}{n} \leq .75.$$

8.2 The LOGSPLINE methodology

Given the integer $K \geq 3$, the numbers L and U with $-\infty \leq L < U \leq \infty$, and the sequence $t_1, \ldots, t_K$ with $L < t_1 < \cdots < t_K < U$, let G be the space of twice continuously differentiable functions s on (L, U), such that the restrictions of s to $[t_1, t_2], \ldots, [t_{K-1}, t_K]$ are cubic polynomials and the restrictions of s to $(L, t_1]$ and $[t_K, U)$ are linear. The space G is K-dimensional. Set $J = K - 1$. Then G has a basis of the form $1, B_1, \ldots, B_J$. We can choose $B_1, \ldots, B_J$ such that B_1 is linear with negative slope on $(L, t_1]$, $B_2, \ldots, B_J$ are constant on $(L, t_1]$, B_J is linear with positive slope on $[t_K, U)$, and $B_1, \ldots, B_{J-1}$ are constant on $[t_K, U)$.

A column vector $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_J)^T \in \mathbb{R}^J$ is said to be *feasible* if

$$\int_L^U \exp(\beta_1 B_1(y) + \cdots + \beta_J B_J(y))dy < \infty$$

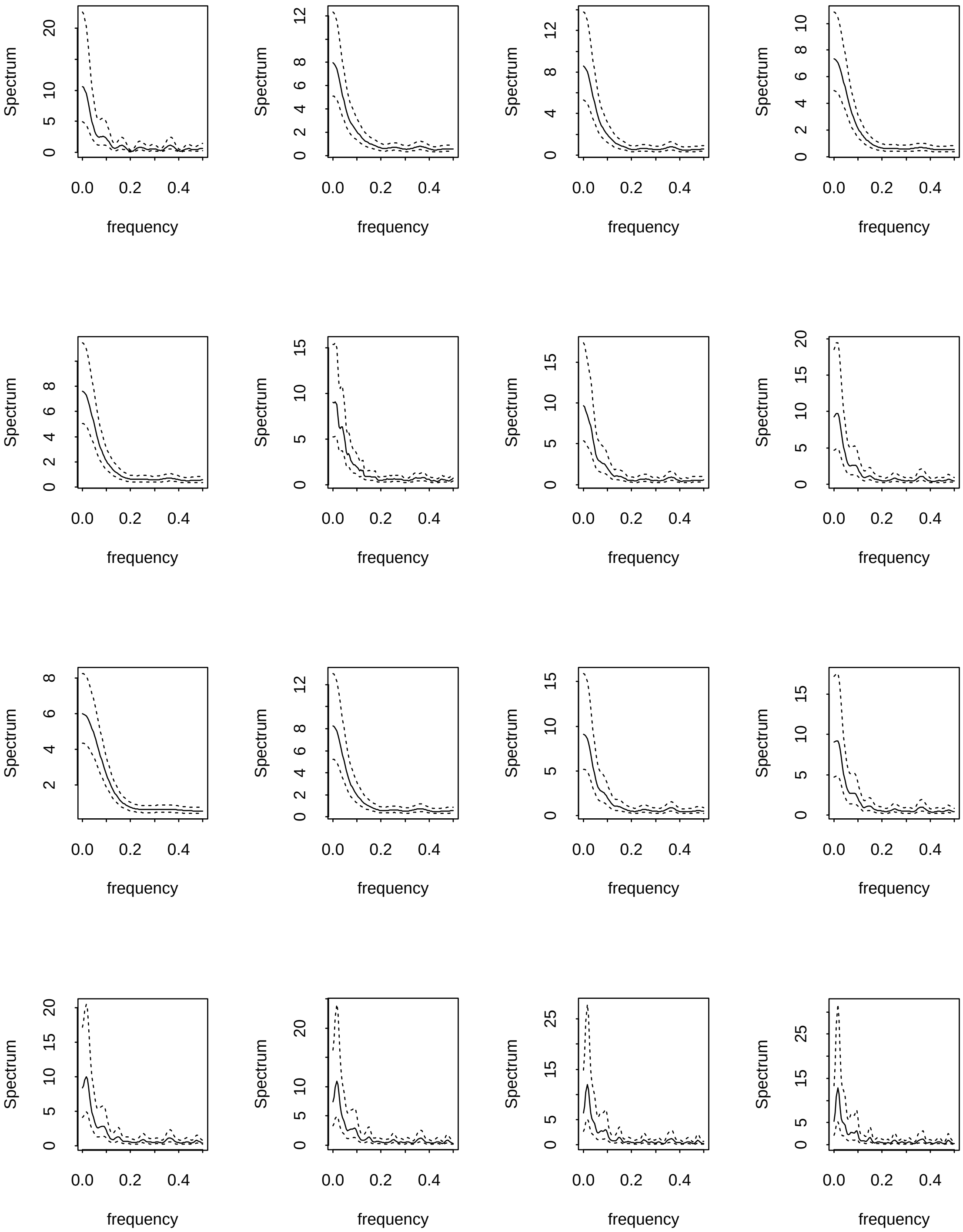


Fig 8.1 *Examples of* `auspe`.

or, equivalently, if (i) either $L > -\infty$ or $\beta_1 < 0$ and (ii) either $U < \infty$ or $\beta_J < 0$. Let $\mathcal{B}$ denote the collection of such feasible column vectors. Given $\boldsymbol{\beta} \in \mathcal{B}$, set

$$f(y;\boldsymbol{\beta}) = \exp(\beta_1 B_1(y) + \cdots + \beta_J B_J(y) - C(\boldsymbol{\beta})), \qquad L < y < U,$$

where

$$C(\boldsymbol{\beta}) = \log\left(\int_L^U \exp(\beta_1 B_1(y) + \cdots + \beta_J B_J(y))dy\right).$$

Then $f(\cdot;\boldsymbol{\beta})$ is a positive density function on (L, U) for $\boldsymbol{\beta} \in \mathcal{B}$. If $U = \infty$, then the density function is exponential on $[t_K, \infty)$; if $L = -\infty$, then the density function is exponential on $(-\infty, t_1]$.

Let $Y_1, \ldots, Y_n$ be a random sample of size n from a distribution on (L, U) having density function f. Let $A_1, \ldots, A_n$ be subintervals of (L, U) such that it is known only that $Y_i \in A_i$ for $1 \le i \le n$. If Y_i is uncensored, then $A_i = \{Y_i\}$. If Y_i is right censored at $C_i < Y_i$, then $A_i = (C_i, U)$. If Y_i is left censored at $C_i > Y_i$, then $A_i = (L, C_i)$. In either case, we refer to C_i as the censoring value of Y_i. If Y_i is interval censored, then its censoring interval A_i is a subinterval of (L, U). Under the usual assumption that the random sample is independent of the censoring mechanism, the log-likelihood function corresponding to the LOGSPLINE model has the form given by

$$\ell(\boldsymbol{\beta}) = \sum_i \varphi(A_i;\boldsymbol{\beta}), \qquad \boldsymbol{\beta} \in \mathcal{B};$$

here

$$\varphi(y;\boldsymbol{\beta}) = \log f(y;\boldsymbol{\beta}) = \sum_j \beta_j B_j(y) - C(\boldsymbol{\beta}), \qquad \boldsymbol{\beta} \in \mathcal{B},$$

if A is the one-point set $\{y\}$ and

$$\varphi(A;\boldsymbol{\beta}) = \log\left(\int_A f(y;\boldsymbol{\beta})dy\right) = \log\left(\int_A \exp\varphi(y;\boldsymbol{\beta})dy\right), \qquad \boldsymbol{\beta} \in \mathcal{B},$$

if A has positive length. Formulas for the score function and Hessian can be found in Kooperberg and Stone (1992, Section 2). These formulas become rather complicated when A has positive length.

The maximum likelihood estimate $\widehat{\boldsymbol{\beta}}$ is given by $\ell(\widehat{\boldsymbol{\beta}}) = \max_{\boldsymbol{\beta}\in\mathcal{B}} \ell(\boldsymbol{\beta})$, and the log-likelihood of the fitted model is given by $\widehat{\ell} = \ell(\widehat{\boldsymbol{\beta}})$. The corresponding maximum likelihood estimate of f is given by $\widehat{f}(y) = f(y;\widehat{\boldsymbol{\beta}})$ for $L < y < U$.

Model selection

The knot selection methodology involves initial knot placement, stepwise knot addition, stepwise knot deletion, and final model selection based on AIC. In this subsection we assume that all the data are uncensored; that is, $A_i = \{Y_i\}$ for all i.

Initially we start with K knots, with $K = \min(2.5n^{\cdot 2}, n/4, N, 25)$, where N is the number of distinct Y_i's. These K knots are positioned according to the rule described in Kooperberg and Stone (1992). This rule places knots at selected order statistics of the data. (The rule is suitably modified when some data are censored.) If $L = -\infty$ and $U = \infty$, the extreme knots are placed at the extreme observations and the interior knots are positioned such that the distances (on an order statistic scale) between knots near the extremes of the data are fairly small and almost independent of the sample size, while the knots in the interior are positioned approximately equidistantly. If $L > -\infty$ or $U < \infty$, the procedure is suitably modified.

The knot-addition/knot-deletion procedure that we employ is essentially the procedure described in Section 3. In particular, at each addition step of the algorithm we first find a good location for a new knot in each of the intervals (L, t_1), (t_1, t_2), ..., (t_{K-1}, t_K), (t_K, U) determined by the existing knots $t_1, \ldots, t_K$. To do this we maximize in each interval the Rao statistic for potential knots located at the quartiles of the data within each interval. The location is then further optimized, which may involve computing a few more Rao statistics [see Section 11.3 of Kooperberg, Stone and Truong (1995a) for our current implementation]. The search algorithm then selects among the best candidates within the various intervals. The default value for the maximum number of knots in a model is $K_{\max} = \min(4n^{\cdot 2}, n/4, N, 30)$.

During knot deletion we successively remove the least significant knot, where Wald statistics are used to measure significance. We continue this procedure until only three knots are left. (Rarely, with extremely heavy tailed densities, there are numerical problems when the number of knots is too small. In such a situation we terminate the procedure as soon as these problems occur.)

Among all models that are fit during the sequence of knot addition and knot deletion we choose the model that minimizes AIC with default penalty parameter $a = \log n$, as described in Section 3.

S Functions

```
library(logspline)  # Load the library
X11()

  myfit <- logspline.fit(rnorm(100))
  logspline.plot(myfit)

  x <- cauchy(n)
  x.fit _ logspline.fit(x)
  logspline.plot(x.fit) # Fig 8.3

  x <- arima.sim(n=200,model=list(order=c(1,0,0),ar=.6))
  xar <- ar(x)
```

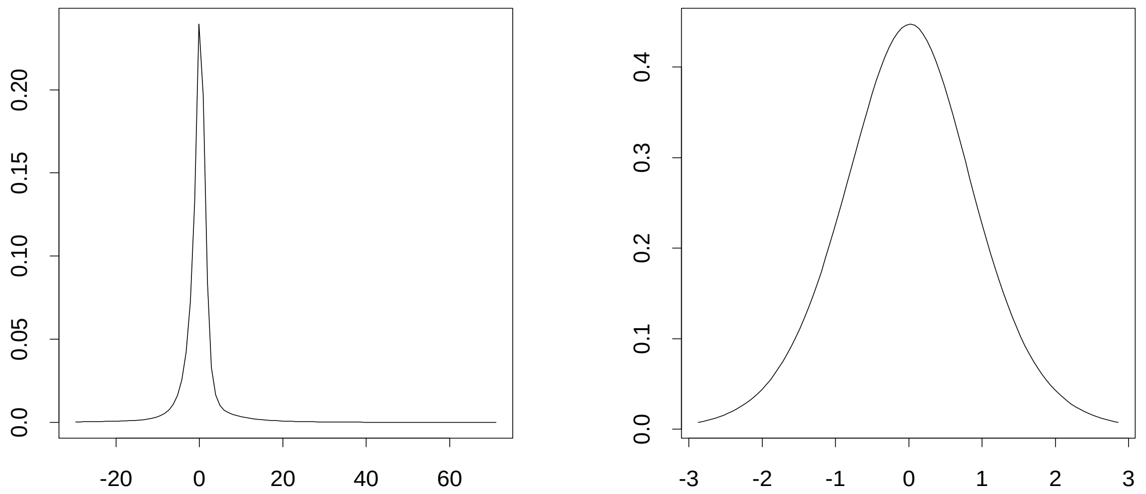


Fig 8.3 *Examples of* `logspline.fit`.

```
xres <- xar$resid
x.fit _ logspline.fit(xres)
logspline.plot(x.fit) # Fig 8.3
```

8.3 The LSPEC methodology

For stationary times series, it is known that the periodogram ordinates at the Fourier frequencies are approximately independent and have an exponential distribution with mean equal to the spectral density function. This implies that the periodogram is not a consistent estimate, but consistency can be achieved by smoothing the periodogram ordinates [see Brillinger (1981)]. In this section we present our version of the spectral estimate by treating it as a special case of the generalized regression problem discussed in Section 2. Specifically, we use the theory and methodology of extended linear models to estimate the logarithm of the mean of the exponential distribution function. Here the mean is the spectral density function.

To describe the possibly mixed spectral distribution, consider a real-valued, second-order stationary time series X_t with mean $E(X_t) = E(X_0)$ and covariance function $\gamma(u) = \mathrm{cov}(X_t, X_{t+u})$. Assume that the time series has the form

$$X_t = \sum_{j=1}^{p} R_j \cos(t\lambda_j + \varphi_j) + Y_t.$$

Here $0 < \lambda_j \leq \pi$; φ_j are independent and uniformly distributed on $[-\pi, \pi]$; R_j are independent, nonnegative random variables such that R_j^2 has positive mean $4\rho_j$; and Y_t is a second-order stationary time series with $E(Y_t) = E(X_0)$ and autocovariance function $\gamma_c(u) = \operatorname{cov}(Y_t, Y_{t+u})$ satisfying $\sum_u |\gamma_c(u)| < \infty$.

The spectral distribution function of X_t is given by

$$F(\lambda) = \int_{-\pi}^{\lambda} f_c(\omega)d\omega + \sum_{\omega \leq \lambda} f_d(\omega), \qquad |\lambda| \leq \pi,$$

where

$$f_c(\lambda) = \frac{1}{2\pi} \sum_{u=-\infty}^{\infty} \gamma_c(u) \exp(iu\lambda), \qquad |\lambda| \leq \pi,$$

and

$$f_d(\lambda) = \begin{cases} \rho_j & \text{if } \lambda = \pm\lambda_j, \\ 0 & \text{otherwise.} \end{cases}$$

The functions f_c and f_d are referred to as the *spectral density function* and *line spectrum* of the time series X_t.

Note that f_c and f_d are nonnegative and symmetric about zero and that they can be extended to periodic functions on $(-\infty, \infty)$ with period 2π. From now on we limit our attention to the interval $[0, \pi]$. Observe that if the indicated derivatives of f_c exist, then $f_c'(0)$, $f_c'''(0)$, $f_c'(\pi)$ and $f_c'''(\pi)$ all equal zero.

Let $\delta_a(\lambda)$ equal one or zero according as $\lambda = a$ or $\lambda \neq a$. Given a time series $X_1, X_2, \ldots, X_{T-1}$, set $f = f_c + \frac{T}{2\pi} f_d$, $\phi = \log f$ and $\phi_c = \log f_c$. Then $\phi = \phi_c + \phi_d$, where $\phi_d = \beta_1\delta_{\lambda_1} + \cdots + \beta_p\delta_{\lambda_p}$ with $\beta_1, \ldots, \beta_p > 0$. Moreover, $f_d = (2\pi/T)(\exp \phi_d - 1)f_c$. In the following discussion, we will use cubic splines to obtain a finite-dimensional approximation to ϕ_c and hence to ϕ.

First, we describe the space of splines that will be used to model the logarithm of the spectral density function. Given the positive integer J_c, let G_c be the J_c-dimensional space of twice continuously differentiable, cubic spline functions s with the knot sequence $0 \leq t_1 < \cdots < t_{J_c} \leq \pi$. We require that $s'(0) = s'(\pi) = 0$. Also, $s'''(0) = 0$ unless $t_1 = 0$, and $s'''(\pi) = 0$ unless $t_{J_c} = \pi$. Let $B_1, \ldots, B_{J_c}$ be a basis of G_c. Then functions in G_c can be extended to splines on $(-\infty, \infty)$ that are symmetric about zero, periodic with period 2π, have a knot at zero if and only if $t_1 = 0$, and have a knot at π if and only if $t_{J_c} = \pi$.

Next, we describe the space that will be used indirectly to model the line spectrum. Given the nonnegative integer J_d and the increasing sequence $a_1, \ldots, a_{J_d}$ of members of $\{2\pi j/T \ : \ 1 \leq j \leq T/2\}$, let G_d be the J_d-dimensional space of nonnegative functions s on $[0, \pi]$ such that $s = 0$ except at $a_1, \ldots, a_{J_d}$. Set $B_{j+J_c}(\lambda) = \delta_{a_j}(\lambda)$ for $1 \leq j \leq J_d$. Then $B_{J_c+1}, \ldots, B_J$ form a basis of G_d, where $J = J_c + J_d$.

Let G be the space spanned by $B_1, \ldots, B_J$. The collection $\mathcal{G}$ of such J-dimensional spaces G form a family of allowable spaces. Set

$$\phi_c(\cdot; \boldsymbol{\beta}_c) = \beta_1 B_1(\cdot) + \cdots + \beta_{J_c} B_{J_c}(\cdot), \qquad \boldsymbol{\beta}_c = (\beta_1, \ldots, \beta_{J_c}) \in \mathbb{R}^{J_c},$$

$\phi_d(\cdot;\boldsymbol{\beta}_d) = \beta_{J_c+1}B_{J_c+1}(\cdot)+\cdots+\beta_J B_J(\cdot), \quad \boldsymbol{\beta}_d = (\beta_{J_c+1},\ldots,\beta_J)$ with $\beta_{J_c+1},\ldots,\beta_J \geq 0$,

and

$$\phi(\cdot;\boldsymbol{\beta}) = \phi_c(\cdot;\boldsymbol{\beta}_c) + \phi_d(\cdot;\boldsymbol{\beta}_d), \qquad \boldsymbol{\beta} = (\beta_1,\ldots,\beta_J).$$

We use $\phi_c(\cdot;\boldsymbol{\beta}_c)$ to model the logarithm of the spectral density function and $\phi(\cdot;\boldsymbol{\beta})$ to model $\log f$. Thus, $f_c(\cdot;\boldsymbol{\beta}_c) = \exp\phi_c(\cdot;\boldsymbol{\beta}_c)$, $f(\cdot;\boldsymbol{\beta}) = \exp\phi(\cdot;\boldsymbol{\beta})$, and

$$f_d(\cdot;\boldsymbol{\beta}_c) = \frac{2\pi}{T}[\exp\ \phi_d(\cdot;\boldsymbol{\beta}_d) - 1]f_c(\cdot;\boldsymbol{\beta}_c).$$

Denote the Fourier frequencies by $\lambda_k = 2\pi k/T$ for $k = 0, 1, \ldots [T/2]$. Let I_k denote the k-th ordinate of the periodogram, which is given by

$$I_k = I^{(T)}(\lambda_k) = (2\pi T)^{-1}\Big|\sum_{t=0}^{T-1}\exp(-i\lambda_k t)X_t\Big|^2.$$

For Gaussian time series, I_k, $1 \leq k \leq [T/2]$, are independent and have the exponential distribution with mean equal to $f(\lambda_k) = \exp\phi(\lambda_k)$. Hence, the log-likelihood function is given by

$$\ell(\boldsymbol{\beta}) = \frac{1}{[T/2]}\sum_{k=1}^{[T/2]}\Big(\frac{\delta_\pi(\lambda_k)}{2} - 1\Big)[\phi(\lambda_k;\boldsymbol{\beta}) + I_k\exp(-\phi(\lambda_k;\boldsymbol{\beta}))], \qquad \boldsymbol{\beta} \in \mathbb{R}^J.$$

Observe that the log-likelihood is a concave function of $\boldsymbol{\beta}$.

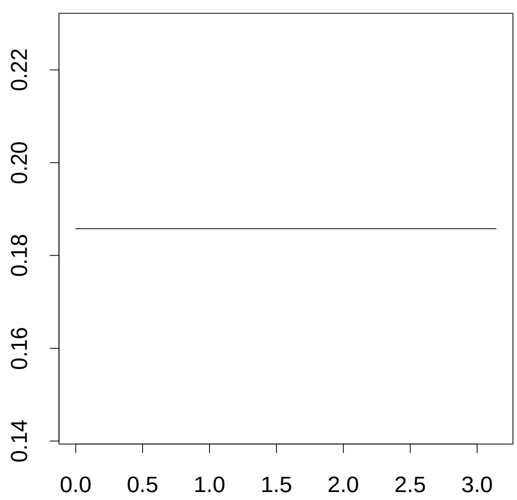
Let $\widehat{\boldsymbol{\beta}}$ denote the maximum likelihood estimate of $\boldsymbol{\beta}$, which is obtained as usual by the Newton–Raphson method. The corresponding estimate of the function f is given by $\widehat{f}(\lambda) = f(\lambda;\widehat{\boldsymbol{\beta}})$. Similarly, estimates of the spectral density function and line spectrum are given by $\widehat{f_c}(\cdot) = f_c(\cdot;\widehat{\boldsymbol{\beta}}_c)$ and $\widehat{f_d}(\cdot) = f_d(\cdot,\widehat{\boldsymbol{\beta}}_d)$, where $\widehat{\boldsymbol{\beta}}_c = (\widehat{\beta}_1,\ldots,\widehat{\beta}_{J_c})$ and $\widehat{\boldsymbol{\beta}}_d = (\widehat{\beta}_{J_c+1},\ldots,\widehat{\beta}_J)$.

As in other cases discussed in this paper, our spectral estimate depends on G. We follow the procedure described in Section 3 (with $d = 1$) to select G adaptively from $\mathcal{G}$. This methodology is referred to as LSPEC in Kooperberg, Stone and Truong (1995c). (In the current implementation of LSPEC, if an atom has a frequency that is not of the form $2\pi\, k/T$, then it is typically replaced by the two closest adjacent atoms with frequencies of this form. Also, LSPEC prevents atoms with small mass from entering the model.)

In the absence of atoms, the rate of convergence of the maximum likelihood estimate $\widehat{\phi}_c$ is given in Kooperberg, Stone and Truong (1995d). This result lends theoretical support to LSPEC.

S Functions

```
library(lspec) # Load the library
```



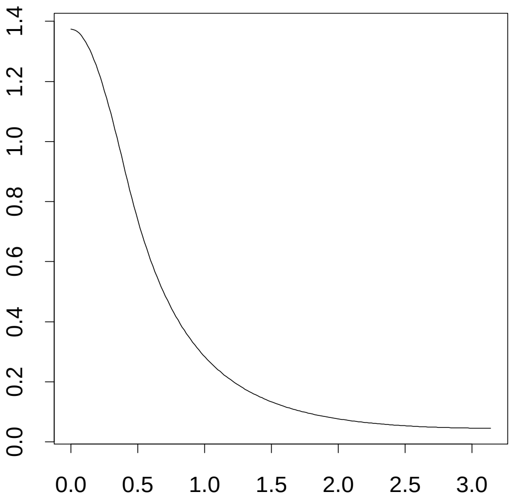


Fig 8.4 *Spectral estimates from* `indtest`.

```
X11()

x.fit <- lspec.fit(rnorm(200))
lspec.plot(x.fit) # Fig 8.4

x <- arima.sim(n=200,model=list(order=c(1,0,0),ar=.6))
x.fit <- lspec.fit(x)
lspec.plot(x.fit) # Fig 8.4
```

An example

We will use LSPEC to analyze the result of a neurophysiological experiment consisting of 30 trials of electrical potential (EP) measurements [see Durka, Kelly and Blinowska (1995)]. It started with a 24 Hz (cycles/sec), $500\mu m$ peak to peak sinusoidal stimulus applied to the right fingertip. The responses are the EP measurements at the scalp and wrist. Each EP measurement lasted for 6 seconds, with the stimulus coming on at 2 seconds and staying on for the remainder of the trial. The channels were sampled at 256 times/sec, giving a total of 1536 sampling points per channel.

Since the stimulus was not active for the first 2 seconds, our analyses were based on the last 4 seconds of recordings, so that $T = 1024$. Figure 11 shows the averages of 30 EP responses from the scalp and wrist, which appear to be stationary. The left side of Figure 12 shows the LSPEC estimate of the scalp EP spectrum. We observe

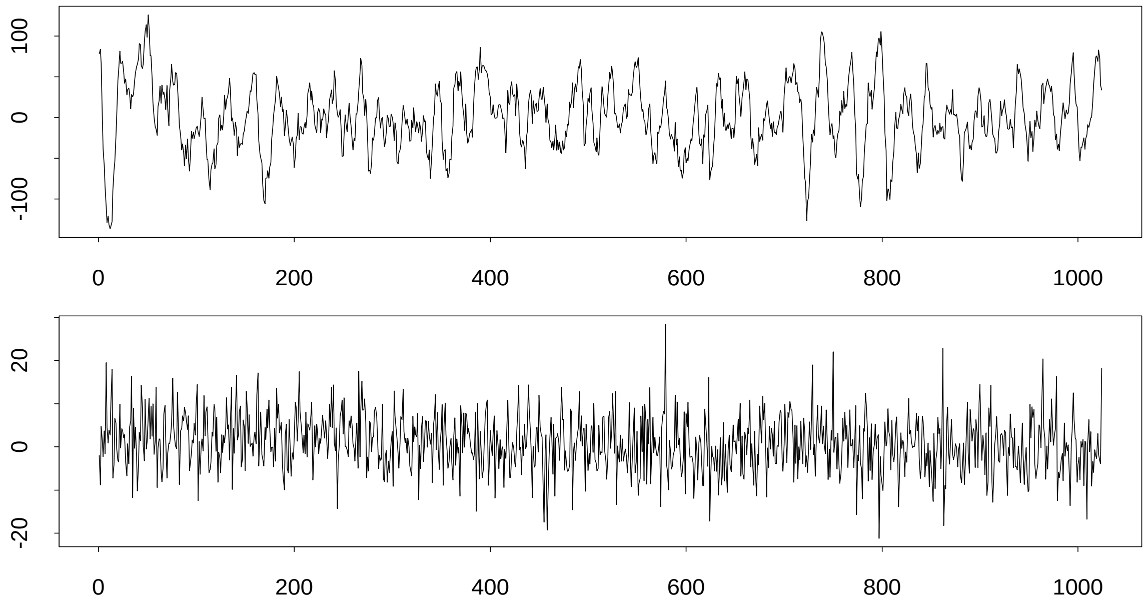


Fig. 8.5 *Averages of 30 series of electrical potential (EP) measurements from the scalp (top) and wrist (bottom).*

two lines with frequencies of 9.25 Hz and 9.75 Hz [the former frequency corresponds to $k = 4(9.25) = 37$ and $\lambda = 2\pi(37)/1024 \doteq 0.227$, and the latter frequency corresponds to $k = 39$ and $\lambda \doteq 0.239$]. These are approximately the alpha-rhythm frequencies. There is also a peak with a frequency of 48 Hz ($\lambda \doteq 1.178$), corresponding to the second harmonic of the stimulus frequency 24 Hz. In the right side of Figure 12, we observe that the wrist EP responded with a frequency (the first line) at 24 Hz, while it also picked up the electrical power line frequency at 60 Hz. Note that the background noise level (the continuous spectrum) is much higher in the scalp EP than in the wrist EP.

The responses were then filtered to remove the unwanted (alpha-rhythm, electrical power line) signals and low frequency components of background noise and sampled at 128 times/sec, yielding a total of 512 sampling points. Applications of LSPEC to the filtered observations are illustrated in Figure 13. For the scalp EP data, the resulting fit is a spline with seven knots and three lines in the model. The first line has a frequency of 24 Hz ($\lambda \doteq 1.178$), showing that LSPEC has located the desired signal. The other two lines correspond to the second harmonic. The fit for the wrist EP data shows a spline with eight knots and one line (at 24 Hz) in the model.

In summary, in this example the LSPEC methodology yielded a precise estimate of the stimulus frequency (24 Hz) and provided an informative description of the neurophysiological data. More generally, in the light of the present example and those given in Kooperberg et al. (1995c), we find the LSPEC methodology to be both effective and of considerable practical value.

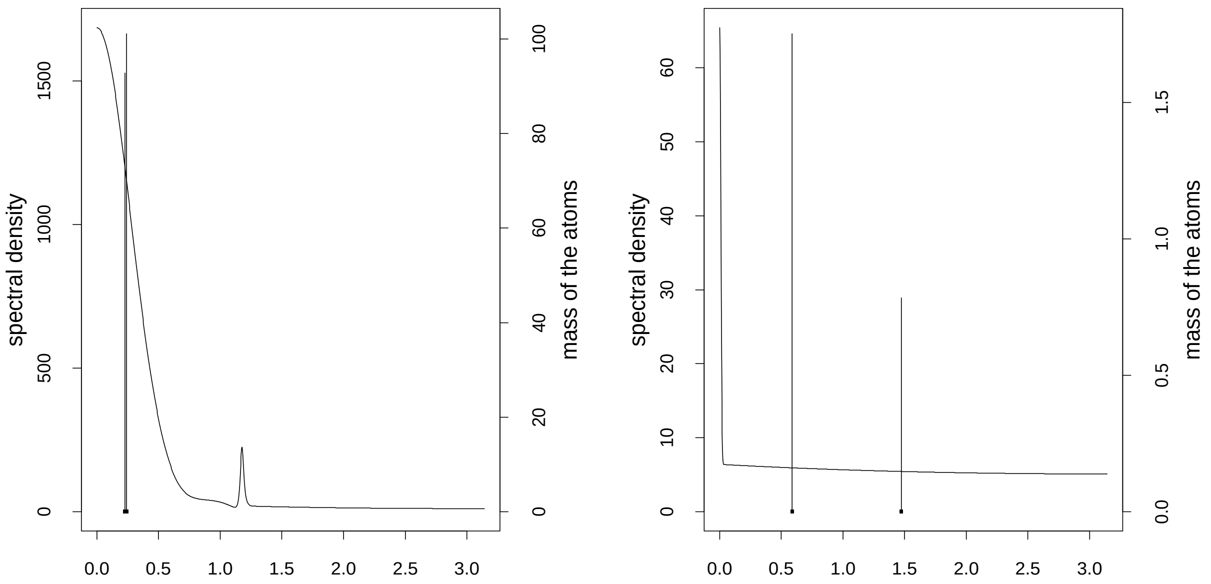


Fig. 8.6 *The scalp EP spectrum (left side) has line frequencies = 9.25 Hz and 9.75 Hz; the peak has a frequency = 48 Hz. The line frequencies of the wrist EP spectrum (right side) are 24 Hz and 60 Hz.*

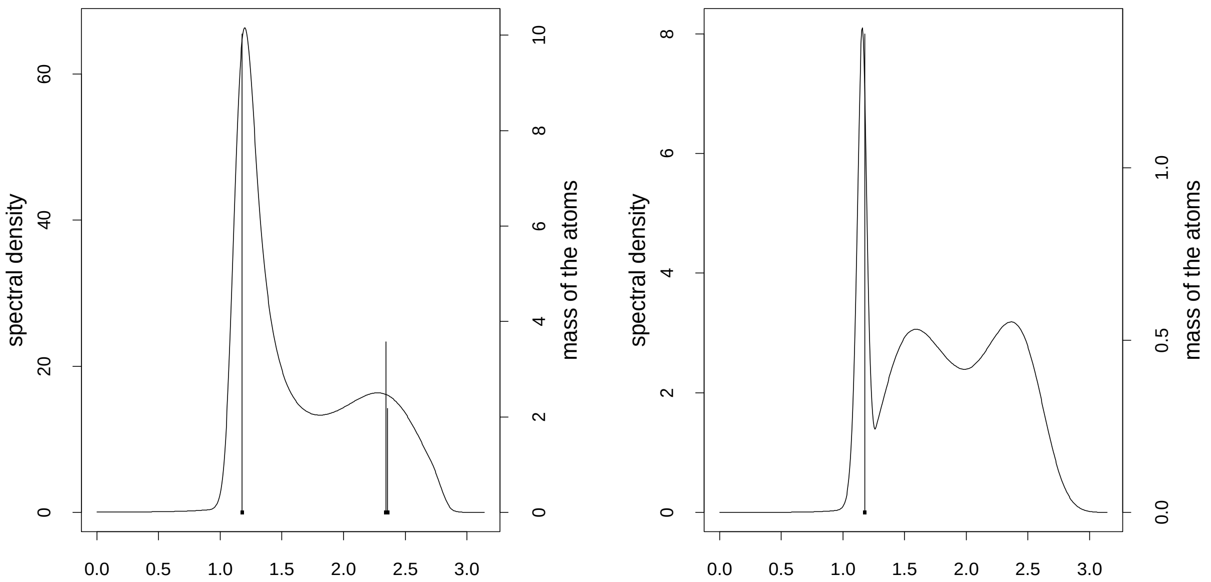


Fig. 8.7 *Spectra of the filtered EP data. The scalp (left side) has line frequencies equal to 24 Hz and 48 Hz. The wrist (right side) has a line frequency equal to 24 Hz.*

8.4 Trend estimation

Linear trend can be modeled via the function `lm()`. More gerenally, a mean function can be estimated via the function `loess` and `stl`. The following yields an estimate of the nonlinear trend.

```
attach(births)
y <- births
tsplot(y,xlab="months",ylab="number of births")
x <- c(1:373)
y.loess _ loess(y~x,span=.4)
lines(x,fitted(y.loess),lty=2)
# title("US Births")
```

The following example uses the function `stl()` model the trend, then plot it along with the residuals.

```
deaths.stl <- stl(deaths, "periodic")
ts.plot(deaths, deaths.stl$sea, deaths.stl$rem)
```

The theory for estimating the mean function via functional approach can be found in Truong (1991) and the references given therein. See also the next section.

8.5 Functional modeling in regression

Let X and Y denote random variables. We consider the problem of estimating the regression function of Y on X: $m(x_0) = E(Y|X = x_0)$. Note that the regression function has the following property:

$$m(x_0) = \text{argmin}_m E[(Y - m(x_0))^2|X = x_0].$$

We use this property to derive a functional estimate of $m(\cdot)$. Observe that

$$E[(Y - m(x_0))^2|X = x_0] = \frac{1}{f_X(x_0)} \int [y - m(x_0)]^2 f(x_0, y)\, dy,$$

where $f_X(\cdot)$ and $f(\cdot, \cdot)$ are the density and joint density functions of X and (X, Y), respectively. Set $g(x) = \int [y - m(x)]^2 f(y, x)\, dy$. Let $(X_1, Y_1), \ldots, (X_n, Y_n)$ denote a random sample from the distribution $F(\cdot, \cdot)$ of (X, Y). Then by the approximation result,

$$g(x_0) = \lim_{h \to 0} \int g(x) K\Big(\frac{x - x_0}{h}\Big)\, dx,$$

where $K(\cdot)$ is a kernel function and h is the bandwidth. Thus

$$\begin{aligned} g(x_0) &\doteq \int\int [y-m(x)]^2 f(y,x)\,dy K\Big(\frac{x-x_0}{h}\Big)\,dx \\ &= \int\int [y-m(x)]^2 K\Big(\frac{x-x_0}{h}\Big)\,dF(x,y) \\ &= \frac{1}{n}\sum_i [Y_i - m(X_i)]^2 K\Big(\frac{X_i-x_0}{h}\Big). \end{aligned}$$

By setting $m(x)=x_0$ and solve the following optimization problem:

$$\min \sum_i [Y_i - m(X_i)]^2 K\Big(\frac{X_i-x_0}{h}\Big),$$

we obtain the usual kernel estimate:

$$\widehat{m}(x_0) = \frac{\sum_i Y_i K\left(\frac{X_i-x_0}{h}\right)}{\sum_i K\left(\frac{X_i-x_0}{h}\right)}. \tag{8.1}$$

More interestingly, if we set $m(x)=a_0+a_1(x-x_0)$ and solve (8.1) for a_0, we obtain the local linear estimate of the regression function [Stone (1977), Cleveland (1979) and Fan (1992).] Note that one can easily get the local polynomial estimate of the regression function by setting $m(x)=a_0+a_1(x-x_0)+\cdots+a_p(x-x_0)^p$ [Stone (1980, 1982).]

Example 1. The observations were generated from:

$$Y_i = \sin(2\pi t_i) + Z_i, \quad Z_i = .5Z_{i-1} + \varepsilon_i,$$

$$\varepsilon_i \sim_{\text{iid}} N(0, .5^2), \quad i = 0, 1, \ldots, 200.$$

A local average estimate of $\theta(t) = \sin(2\pi t)$ is obtained by choosing a smoothing parameter $\delta_n = 0.17$. [In fact, a sequence of local average estimates were constructed with bandwidths ranging from 0.12 to 0.20. $\delta_n = 0.17$ was chosen because it provided the most suitable smooth estimate. One may also try to select the bandwidth via the method considered by Chiu (1989), which, however, requires the knowledge of time series component.]

After the data were smoothed, the residuals $Y_i - \widehat{\theta}_n(t_i)$ were used to provide parametric estimates for the time series Z_i. The estimated model is given by

$$Z_i = 0.46305 Z_{i-1} + \varepsilon_i, \quad \varepsilon_i \sim_{\text{iid}} N(0, 0.23136).$$

Indeed, this was obtained by fitting four autoregressive models $[AR(p),\ p=1,\ldots,4]$ to the residuals using Brillinger's DRBLIB. These results are given in Table 1. Here,

$HQ(p)$ is defined by $HQ(p) = \log(s^2) + 2pn^{-1} \log\log(n)$ (Hannan and Quinn, 1979) with s^2 denoting the residual variance. The order of the fitted process is chosen to be the value of p at which $HQ(p)$ is minimum. In this case, the minimum value is $HQ(1)$. See Table 1 of Truong (1991).

[The Hannan-Quinn criterion is preferred over other model selection criteria because (1) it is a consistent rule, and (2) in a simulation study, Lütkepohl (1985) has shown that this criterion leads most often to correct estimates for model order and has the smallest prediction error.]

Table 1. Parametric Time Series Model Fitting for the Error Structure

p	$\hat{\mathbf{b}}$	s^2	$HQ(p)$
1	$\hat{b}_1 = 0.46305$	0.23136	-1.4304
2	$\hat{b}_1 = 0.43035, \hat{b}_2 = 0.07063$	0.23021	-1.4187
3	$\hat{b}_1 = 0.43124, \hat{b}_2 = 0.07605$ $\hat{b}_3 = 0.01260$	0.23017	-1.4022
4	$\hat{b}_1 = 0.43051, \hat{b}_2 = 0.08041$ $\hat{b}_3 = 0.01212, \hat{b}_4 = 0.05733$	0.22942	-1.3888

Let $(\mathbf{X}_t, Y_t)$, $t = 0, \pm 1, \ldots$ denote a (strictly) stationary time series with $\mathbf{X}_t$ being $\mathbf{R}^d$-valued and Y_t being real-valued. Let $\theta(\cdot)$ denote either the conditional mean (regression function) on $\mathbf{R}^d$, which is given by $\theta(\mathbf{X}_0) = E(Y_0|\mathbf{X}_0)$, or the conditional median function on $\mathbf{R}^d$, which is given by $\theta(\mathbf{X}_0) = \mathrm{med}(Y_0|\mathbf{X}_0)$. Here $E(Y_0|\mathbf{X}_0)$ and $\mathrm{med}(Y_0|\mathbf{X}_0)$ denote the mean and median respectively of the conditional distribution of Y_0 given $\mathbf{X}_0$.

EXAMPLE 1 (*Univariate time-series*). Let $X_t,\ t = 0, \pm 1, \pm 2, \ldots$ be a real-valued stationary time series, let d be a positive integer and let m be an integer. Set

$$\mathbf{X}_t = (X_{t+1}, \ldots, X_{t+d}) \text{ and } Y_t = X_{t+d+m}.$$

Then $(\mathbf{X}_t, Y_t)$, $t = 0, \pm 1, \ldots$ is a stationary time series,

$$E(Y_0|\mathbf{X}_0) = E(X_{d+m}|X_1, \ldots, X_d)$$

and

$$\mathrm{med}(Y_0|\mathbf{X}_0) = \mathrm{med}(X_{d+m}|X_1, \ldots, X_d).$$

In the context of forecasting m units of time into the future, m is a positive integer.

EXAMPLE 2 (*Bivariate time-series*). Let (X_t, Z_t), $t = 0, \pm 1, \ldots$ be an $\mathbf{R}^2$-valued stationary time series, let d be a positive integer, and m a nonnegative integer. Set

$$\mathbf{X}_t = (X_{t+1}, \ldots, X_{t+d}) \text{ and } Y_t = Z_{t+d+m}.$$

Then $(\mathbf{X}_t, Y_t)$, $t = 0, \pm 1, \ldots$ is a stationary time series,

$$E(Y_0|\mathbf{X}_0) = E(Z_{d+m}|X_1, \ldots, X_d)$$

and

$$\operatorname{med}(Y_0|\mathbf{X}_0) = \operatorname{med}(Z_{d+m}|X_1, \ldots, X_d).$$

EXAMPLE 3 (*Bivariate time-series*). Let (X_t, Z_t), $t = 0, \pm 1, \ldots$ be an $\mathbf{R}^2$-valued stationary time series, let d, k and m be positive integers such that $k \leq d$. Set

$$\mathbf{X}_t = (X_{t+1}, \ldots, X_{t+k}, Z_{t+k+1}, \ldots, Z_{t+d}) \text{ and } Y_t = Z_{t+d+m}.$$

Then $(\mathbf{X}_t, Y_t)$, $t = 0, \pm 1, \ldots$ is a stationary time series,

$$E(Y_0|\mathbf{X}_0) = E(Z_{d+m}|X_1, \ldots, X_k, Z_{k+1}, \ldots, Z_d)$$

and

$$\operatorname{med}(Y_0|\mathbf{X}_0) = \operatorname{med}(Z_{d+m}|X_1, \ldots, X_k, Z_{k+1}, \ldots, Z_d).$$

Nonlinear time series modelling provides a useful alternative to traditional data analysis and it has been an active area of research lately. Comprehensive accounts of this important direction are described in Priestley (1988) and Tong (1990), where specific properties of models, testing and estimation are also obtained. There are, however, fewer results on model identification. This important issue has led a number of authors to consider nonparametric problems of estimating conditional mean and median functions. The usefulness of this approach in model identification is discussed, for example, in Robinson (1983, 1984), Auestad and Tjøstheim (1990), Truong and Stone (1992), and the references cited therein.

The present paper considers a semiparametric approach to regression problems involving bivariate stationary time series. To motivate this approach, let (X_i, Y_i) be a bivariate stationary time series in which

$$\begin{aligned} Y_i &= \theta(X_i) + Z_i, \\ Z_i &= \beta Z_{i-1} + \varepsilon_i, \quad |\beta| < 1, \quad \varepsilon_i \sim_{\text{iid}} (0, \sigma^2). \end{aligned} \tag{8.2}$$

Suppose $\{\varepsilon_i\}$ and $\{X_i\}$ are independent sequences of random variables, so that $\theta(X_i) = E(Y_i|X_i)$. Consider now the problem of estimating the parameter β as well as the regression function $\theta(\cdot)$. The solution to such problems would lead to a better understanding of the underlying structures of the response series Y_i and the bivariate series (X_i, Y_i).

In classical time series analysis, the estimation is carried out by putting parametric (linear) assumptions on $\theta(\cdot)$. This approach, known as parametric regression analysis, is described in detail by Brillinger (1981), Brockwell and Davis (1991), Fuller (1976), Hannan (1970), Priestley (1981), Shumway (1988), and the references cited therein; see also Cochran and Orcutt (1949). The approach can be made more flexible or general by adopting nonlinear models as considered by Priestley (1988) and Tong (1983, 1990). In practical applications, one would have to address the issue of model identification.

To develop a general approach to the above problem, we extend model (8.2) as follows. Let $(\mathbf{X}_i, Y_i)$, $i = 0, \pm 1, \ldots$ denote a bivariate stationary time series with $\mathbf{X}_i$ being $\mathbf{R}^d$-valued and Y_i being real-valued. Also, let $\theta(\cdot)$ be a smooth real-valued function on $\mathbf{R}^d$ and consider the following regression model:

$$Y_i = \theta(\mathbf{X}_i) + Z_i, \tag{8.3}$$

where $Z_i = \sum_{u=-\infty}^{\infty} a_{i-u}\varepsilon_u$. Here $\{a_u : u = 0, \pm 1, \pm 2, \ldots\}$ is a sequence of unknown parameters and $\{\varepsilon_u : u = 0, \pm 1, \ldots\}$ is a sequence of iid random variables with mean zero and variance σ^2. Furthermore, suppose the sequences $\{\mathbf{X}_i, i = 0, \pm 1, \pm 2, \ldots\}$ and $\{\varepsilon_i, i = 0, \pm 1, \pm 2, \ldots\}$ are independent, so that $\theta(\mathbf{X}_i) = E(Y_i|\mathbf{X}_i)$.

To enhance the flexibility in describing the relationship between the response and predictive series, as well as the ability to identify various nonlinear models, it is desirable to eliminate the parametric assumptions on the mean function $\theta(\cdot)$. With this semiparametric approach, the present paper generalizes the results presented in Truong and Stone (1992) in several directions. In particular, inference for the autocorrelated structures in the response series [such as the estimation of β in model (8.2)] can be given via the usual parametric approach. Moreover, results on rates of convergence in the estimation of the regression function $\theta(\cdot)$ can be improved. More specifically, under appropriate conditions to be given in the following sections, it is shown that local polynomial estimators of the regression function $\theta(\cdot)$ can be chosen to achieve both local and global optimal rates of convergence as given in Stone (1980, 1982). Furthermore, if the errors Z_i form part of a parametric time series model with finitely many unknown parameters, then these parameters can be estimated with the usual root-n rate of convergence.

Example. We consider the following bivariate time series model:

$$X_i = 1.2X_{i-1} - 0.4X_{i-2} + \xi_i, \quad \xi_i \sim_{\text{iid}} N(0, 0.2832^2),$$

$$Y_i = \theta(X_i) + Z_i,$$

$$Z_i = -0.5Z_{i-1} - 0.25Z_{i-2} + \varepsilon_i, \quad \varepsilon_i \sim_{\text{iid}} N(0, 0.4437^2), \quad i = 1, \ldots, 1000,$$

$$\theta(x) = x + 2\exp(-16x^2).$$

The parameters have been chosen to ensure stationarity in both series; see Figures 1 and 2 of Truong and Stone (1994).

Observe that the variances of ξ_i and ε_i are chosen so that the standard deviations of X_i and Z_i equal to 0.5 and 0.6, respectively. The function $\theta(\cdot)$ is estimated by using the local linear estimator described in Section 2. Figure 3 of Truong and Stone (1994) shows the scatterplot of Y and X and the local linear estimate of $\theta(\cdot)$ obtained by choosing $\delta_n = 0.13$. Note that the high variability in the tails of the estimate occur near the tails of the distribution of the X series. The estimate is compared with the underlying model $\theta(\cdot)$ in Figure 4 of Truong and Stone (1994).

To fit a parametric model to the series $Z_i, i = 1, 2, \ldots$, we construct the residuals $\hat{Z}_i, i = 1, 2, \ldots$ as in Section 4 with the sequence of bandwidths $\delta_n = 0.10, 0.13, 0.16, 0.19, 0.22$. Then autoregressive models of orders up to 20 are fitted, and the final model is selected by the AIC criterion. The results are summarized in TABLE 1.

TABLE 1
SUMMARY OF FITTING AUTOREGRESSIVE MODELS OF ORDERS UP TO 20 TO THE RESIDUAL SERIES $\hat{Z}_i, i = 1, 2, \ldots$ WITH BANDWIDTHS $\delta_n = 0.10, 0.13, 0.16, 0.19, 0.22$.

δ_n	Selected order	β_1	β_2
0.10	2	-0.4717	-0.2592
0.13	2	-0.4691	-0.2599
0.16	2	-0.4646	-0.2593
0.19	2	-0.4554	-0.2553
0.22	2	-0.4422	-0.2489

We also fitted autoregressive models of orders up to 20 to the series $Z_i, i = 1, 2, \ldots$, and AIC selected AR(2) with $(\beta_1, \beta_2) = (-0.4767, -0.2598)$. Observe that the simulated X_i's do not have compact support as required by Theorem 4, but this does not seem to have a noticeable effect on the parametric estimates using reconstructed residuals. Figure 5 of Truong and Stone (1994) shows the time series plot of $\{\hat{Z}_i\}$, which was obtained using the bandwidth of 0.13. The differences $\{Z_i - \hat{Z}_i\}$ of these residuals are shown in Figure 6 of Truong and Stone (1994).

8.6 Functional modeling via ELM

Extended linear modeling – forthcoming ...

```
library(hare)
library(heft)
```

Chapter 9

S Functions for Time Series Analysis

9.1 Standard S functions

acf	Estimates and displays autocovariance	acf

```
DESCRIPTION:
       Estimates and displays autocovariance, autocorrelation  or
       partial autocorrelation functions.

USAGE:
       acf(x, lag.max=<<see below>>, type="correlation", plot=T)

REQUIRED ARGUMENTS:
x:     a regular or calendar time series, or a vector or  matrix.
       Time  series  may  be univariate or multivariate.  Missing
       values are allowed only at the beginning or end of series.
       Factor  data are not allowed.  If x is a matrix, rows are
       treated as timepoints and columns as univariate series.

OPTIONAL ARGUMENTS:
lag.max:    the maximum number of lags at which to  estimate  the
       autocovariance.   If  this is not supplied, it is a number
       proportional to the logarithm of the length of the series.
type:    a character  string:  "covariance"  to  estimate  the
       autocovariance    function,    "correlation"    for   the
       autocorrelation function, or "partial",  if  the  partial
```

autocorrelation function is desired. The start of one of the strings will suffice.

plot: If TRUE, the autocovariance or autocorrelation function between pairs of univariate series will be plotted in an array of at most 5 by 5 plots per page. If type is "correlation" or "partial" approximate 95% confidence limits are drawn on the plots.

VALUE:

a list with the following components:

acf: a three-dimensional array containing the autocovariance or autocorrelation function estimates. acf[i,j,k] is the covariance (or correlation) between the j-th series at time t and the k-th series at time t+1-i.

lag: an array the same shape as acf containing the lags (as fractions of the sampling period) at which acf is calculated. If j > k and i > 1, then lag[i,j,k] is negative.

n.used: the number of observations in which no missing values occur.

type: a character string indicating the type of function, "covariance", "correlation" or "partial".

series: the name of x, including transformations.

`ar` Fit Univariate or Multivariate Autoregressive Model `ar`

DESCRIPTION:

Fits a model of the form x(t) = a(1)x(t-1) + ... + a(p)x(t-p) + e(t).

USAGE:

```
ar(x, aic=T, order.max=<<see below>>, method="yule-walker")
```

REQUIRED ARGUMENTS:

x: a univariate or multivariate time series, or a vector, or a matrix with columns representing univariate components of a multivariate time series. Missing values are allowed only at the beginning or end of series.

OPTIONAL ARGUMENTS:

```
aic:    logical flag: if TRUE', use the Akaike information
        criterion to choose the best order not greater than
        order.max'. If FALSE', order.max' will be the order of
        the fitted model.
order.max:    the maximum order of autoregression to fit to the
        time series. The default value is 10 *
        log10(n.used/ncol(x))'.
method:   either the string "yule-walker"' or "burg"' depending
        on whether Yule-Walker equations or Burg's algorithm is to
        be used in estimating the autoregression coefficients.
        Only the first character is necessary.

VALUE:
        a list with the following components:
order:   the order of the autoregression fitted. If aic=TRUE',
        then this is the order less than or equal to order.max'
        which minimizes the AIC, otherwise, it is order.max'.
ar:     the autoregressive coefficients. These are in an order' by
        "nser" by "nser" array. where "nser" is the number of
        univariate components of x'. If order' is 0', ar' will
        have dimensions 1' by "nser" by "nser" and will be filled
        with zeros. The first level of the first dimension
        corresponds to one observation back in time, the second
        level corresponds to two observations back, etc. The
        second dimension corresponds to the predicted series, and
        the third corresponds to the predicting series.
var.pred:  the estimated prediction variance ("nser" by "nser"
        matrix) of the process with AR coefficients ar'.
aic:    vector of the values of Akaike's information criterion for
        orders 0' through order.max'. These have the minimum
        value subtracted from all of them so the minimum is always
        zero.
n.used:   the length of the nonmissing part of the input time
        series.
order.max:    the input or default value of order.max'.
partialacf:     vector of the partial autocorrelation
        coefficients, useful for choosing the proper order of the
        autoregression. This is computed only if method="yule-
        walker"'.
resid:    a time series representing the estimate of e[t] for the
        model above. The estimates contained in ar' are used in
        the forward direction on the series with mean(s) removed.
```

```
method:  the string "yule-walker"' or "burg"'.
series:  the name used for the input x'.  It includes
       transformations.
```

arima.diag	Compute Diagnostics for ARIMA Model	arima.diag

```
DESCRIPTION:
       Computes diagnostics for an ARIMA model.  The diagnostics
       include the autocorrelation function of the residuals, the
       standardized residuals, and the portmanteau goodness of
       fit test statistic.

USAGE:
       arima.diag(z, acf.resid=T, gof.lag=10, lag.max=<<see below>>,
                  resid=F, std.resid=T, plot=T, ...)

REQUIRED ARGUMENTS:
z:     a list like the output from arima.mle'.

OPTIONAL ARGUMENTS:
acf.resid:    logical flag: if TRUE', the autocorrelation of the
       residuals will be returned.
lag.max:   the maximum number of lags at which to estimate the
       autocovariance.  If this is not supplied, it is the
       maximum between gof.lag' plus the number of model
       parameters and a number proportional to the logarithm of
       the length of the series.
gof.lag:   if gof.lag>0', then gof.lag' plus the number of model
       parameters is the number of lags to use for computing the
       Portmanteau goodness of fit statistic.  If gof.lag=0',
       then the statistic will not be computed
resid:   logical flag: if TRUE', then the residuals will be
       returned.
std.resid:    logical flag: if TRUE', then the standardized
       residuals will be returned.
plot:    logical flag: if TRUE', the diagnostics will be plotted
       using the function arima.diag.plot'
...:   additional arguments may be passed to the function
       arima.diag.plot'
```

```
VALUE:
       a list (which is returned invisibly when plot=TRUE')  with
       the following elements:
acf.list:   a list representing the autocorrelation  function  of
       the residuals.  See acf' for details.
gof:   a  list  representing  the  Portmanteau  goodness  of  fit
       statistics  computed  for  a  range of lags.  The list has
       four elements: lag', statistic', df', p.value'.  lag' is a
       vector   of  the  number  of  lags  used  to  compute  the
       statistics.   statistic'  is  the  vector  of   statistics
       corresponding  to  each  lag  used.   df' is the number of
       degrees of freedom the test statistics have under the null
       hypothesis  that  the  model  is  correct.   p.value' is a
       vector of the p-values for the  statistics  using  a  Chi-
       Squared  distribution  with  the  appropriate  degrees  of
       freedom.
resid:    the residuals or innovations for the process.
std.resid:     the standardized  residuals.   The  residuals  are
       standardized  to  have  unit variance under the assumption
       that the model is correct and the process is Gaussian.
series:   the name of x, including transformations.
```

arima.forecast	Forecast a Time Series via ARIMA	arima.forecast

```
DESCRIPTION:
       Forecasts a univariate time series using an  ARIMA  model.
       Under  the  assumption  that the model is known, predicted
       values and their standard errors are computed  for  future
       values.

USAGE:
       arima.forecast(x, model, n, end, sigma2=<<see below>>, xreg=NULL,
                      reg.coef=NULL)

REQUIRED ARGUMENTS:
x:     a univariate time series  or  a  vector.   Missing  values
       (NAs) are allowed.
model:    a list specifying an  ARIMA  model  as  in  arima.mle'.
       Note  that  the  coefficients must be provided through the
       elements ar' and ma' (otherwise the coefficients  are  set
```

```
       to zero).

       Either end' or n' must be specified.  If both are
       specified, they must agree.

OPTIONAL ARGUMENTS:
n:     the number of time periods to forecast beyond the end of
       the series (optional if end' is provided).
end:   the last date for which forecasts should be produced.
       Forecasts will be produced for every time period between
       end(x)' and end'.
sigma2:   the estimated innovations variance.  If omitted,
       sigma2' will be the concentrated prediction error variance
       computed from the model.
xreg:     a univariate or multivariate time series or a vector or
       matrix with univariate time series per column.  These will
       be used as additive regression variables.  xreg' must be
       consistent with the number of time periods forecast:
       e.g., if n=10' and length(x)=20', then the number of rows
       of xreg' should be 30'.
reg.coef:   a vector of regression coefficients corresponding to
       xreg'.

VALUE:
       a list with the following elements:
mean:     the estimated mean of the forecasts.
std.err:    the approximate forecast error.  In a Gaussian series
       with known ARIMA model structure, then each row of the
       matrix tsmatrix(mean-1.96*std.err, mean+1.96*std.err)' is
       a 95% (non-simultaneous) confidence interval for the
       forecast in that time period.  Note that std.err' does not
       take into account the variability due to estimation of the
       ARIMA model.

DETAILS:
       The ARIMA model is put into state space form and the
       Kalman filter is applied to obtain the forecast intervals.
       This is done using the function arima.filt'.

REFERENCES:
       Harvey, A. C. (1981).  Time Series Models.  Wiley, New
       York.
```

```
EXAMPLES:
       # Fit an ARIMA model to the ship data and forecast one year
       #  beyond the end of the data.  Plot the series along with
       #  forecast means +/- two standard errors

       model <- list(list(order=c(0,1,1)),list(order=c(0,1,1),period=12))
       ship.fit <- arima.mle(ship,model=model)
       ship.fore <- arima.forecast(ship,n=12,model=ship.fit$model)
       tsplot(ship,ship.fore$mean,ship.fore$mean+2*ship.fore$std.err,
                      ship.fore$mean-2*ship.fore$std.err)
```

`arima.mle`	ARIMA Modeling via Gaussian Maximum Likelihood	`arima.mle`

```
DESCRIPTION:
       Returns a list representing a univariate ARIMA model
       estimated by Gaussian maximum likelihood.

USAGE:
       arima.mle(x, model, n.cond=<<see below>>, xreg=NULL,  ...)

REQUIRED ARGUMENTS:
x:     a univariate time series or a vector.  Missing values
       (NAs) are allowed.
model:    a list specifying an ARIMA model.  If a simple ARIMA
       model is specified (i.e., only one component), then model'
       should be a list with component names from: order',
       period',  ar',  ma', ndiff', ar.opt', ma.trans', and
       ma.opt'.  See the DETAILS section for further details.

OPTIONAL ARGUMENTS:
n.cond:   the number of observations on which to condition the
       likelihood.  n.cond' must be at least P+D (the default),
       where P and D are the orders of the expanded
       autoregressive and differencing polynomials.
xreg:     a univariate or multivariate time series or a vector or
       matrix with univariate time series per column.  These will
       be used as additive regression variables.
...:   additional arguments can be passed to nlmin'.
```

VALUE:

a list representing the result of the fitted model:

model: the same as the input model with the estimated coefficients substituted in the components ar' and ma'.

var.coef: the variance-covariance matrix for the autoregressive and moving average coefficients.

loglik: -2 times the log likelihood of the model (up to a constant factor).

aic: Akaike's information criteria, which is loglik' plus 2 times the number of parameters fit.

sigma2: the estimated innovations variance.

n.used: the number of observations used to compute the likelihood.

n.cond: the number of observations on which the likelihood is conditioned.

reg.coef: the estimated regression coefficients (returned if xreg' is provided).

converged: if the optimizer has apparently successfully converged to a minimum, then converged' is TRUE'; otherwise converged' is FALSE'.

conv.type: a character string describing the type of convergence.

series: a character string giving the name of x', including transformations.

reg.series: a character string giving the name of xreg', including transformations (only returned if xreg' is provided).

DETAILS:

SPECIFYING A MODEL. The components of the model' argument should be some or all of: order', period', ar', ma', ndiff', ar.opt', ma.trans', and ma.opt'. order' is a vector of length 3 specifying the order of an ARIMA(p,d,q); the elements of the vector are (p, d, q), where p and q are the order of the autoregressive and moving average operators and d is the number of differences. period' specifies the period of the ARIMA operators; e.g., a seasonal operator for monthly data would have a period of 12' (the default is 1'). ar' and ma' are vectors of initial values for autoregressive and moving average coefficients respectively (if not provided, the initial values to the optimizer are set to 0').

ndiff' is an alternative way to specify the number of differences. A model can be specified using either order' or ar', ma', and ndiff', or both (but they must agree if both are specified). The model doesn't explicitly allow for a non-zero mean of the series, though the estimation is invariant to the mean of the data for many of the possible models; if you estimate an AR model, you must subtract the mean yourself or use the xreg' argument in order to get sensible estimates.

If ma.trans' is TRUE' (the default), the moving average coefficients will be transformed before passing them to the optimizer, ensuring invertibility of the model. ar.opt' and ma.opt' are logical vectors of length p and q respectively. If the ith element is TRUE', then the optimizer will optimize over the ith autoregressive or moving average coefficient. By default, the vectors are TRUE'. This option is useful to fit models in which some low order coefficients are set to 0'.

If a multiplicative ARIMA model is specified (i.e., more than one component), then model' is a list of lists. Each list represents a component of the multiplicative ARIMA model, and is specified in the same manner as above (i.e., the components of each list are ar', ma', ndiff', order', period', ma.trans', ar.opt', ma.opt').

THE ALGORITHM. The likelihood is conditioned on n.cond' observations. If no missing values are present, the likelihood is computed by forming the Choleski decomposition of the covariance matrix of the process (see Ansley, 1979). If missing values are present, then the likelihood is computed using the Kalman filter applied to a state space representation of the likelihood. The state space representation is the same as that used by Kohn and Ansley (1986). The method of initializing the Kalman filter recursions is that given by Bell and Hillmer (1987). The regression parameters are concentrated out of the likelihood, as in Kohn and Ansley (1985). The autoregressive and moving average parameters are transformed to ensure stationarity and invertibility as in Jones (1980).

```
WARNING:
       Unlike the ar' function, the mean of the series  will  not
       be  estimated  by  arima.mle'  unless  you  use  the xreg'
       argument.  arima.mle' assumes a zero mean series.

SEE ALSO:
       ar', arima.diag',  arima.filt',  arima.forecast',  nlmin',
       sabl'.
```

arima.sim	Simulate a Univariate ARIMA Series	arima.sim

```
DESCRIPTION:
       Simulates a univariate ARIMA time series.  By default  the
       innovations are Gaussian.

USAGE:
       arima.sim(model, n=100, innov=NULL, n.start=100, start.innov=NULL,
                 rand.gen=rnorm, xreg=NULL, reg.coef=NULL, ...)

REQUIRED ARGUMENTS:
model:    a list specifying an  ARIMA  model  as  in  arima.mle'.
       Note  that  the  coefficients must be provided through the
       elements ar' and ma' (otherwise the coefficients  are  set
       to zero).

OPTIONAL ARGUMENTS:
n:     the length of the series  to  be  simulated  (optional  if
       innov' is provided).
innov:    a univariate time series or vector  of  innovations  to
       produce  the  series.   If  not  provided,  innov' will be
       generated  using  rand.gen'.   Missing  values   are   not
       allowed.
n.start:     the  number  of  start-up  values   discarded   when
       simulating    non-stationary    models.     The   start-up
       innovations will be generated by rand.gen' if start.innov'
       is not provided.
start.innov:   a univariate time series or vector of  innovations
       to  be  used  as  start up values.  Missing values are not
       allowed.
```

rand.gen: a function which is called to generate the innovations. Usually, rand.gen' will be a random number generator.
xreg: a univariate or multivariate time series, or a vector, or a matrix with univariate time series per column. These will be used as additive regression variables.
reg.coef: a vector of regression coefficients corresponding to xreg'.
...: additional arguments may be passed to rand.gen'

VALUE:
a univariate time series, the simulated series.

demod	Complex Demodulation with Least Squares Lowpass Filter	demod

DESCRIPTION:
Estimates the instantaneous amplitude and phase of a given harmonic component in order to analyze nonstationary time series. This function can also be used to construct lowpass filters.

USAGE:
```
demod(x, centerfreq, passfreq, stopfreq, wrapphase=F)
```

REQUIRED ARGUMENTS:
x: a univariate or multivariate time series, or a vector, or a matrix with one univariate series per column. Missing values are allowed only at the ends.
centerfreq: the frequency at which to demodulate the series.
passfreq: the pass frequency of the lowpass filter. passfreq' must be less than stopfreq'.
stopfreq: the stop frequency of the lowpass filter.

OPTIONAL ARGUMENTS:
wrapphase: logical flag: if TRUE', then phase' is wrapped into the interval [- pi, pi]. When FALSE', phase' is made continuous by not allowing jumps greater than 2' times pi.

VALUE:
a list with the following components:

```
amp:    a time series like x' containing the amplitude at
       frequency centerfreq'.
phase:    a time series like x' containing the phase in radians
       at frequency centerfreq'.
filter:  vector of the lowpass filter coefficients.
centerfreq:   the same as the input centerfreq'.
stopfreq:  the same as the input stopfreq'.
passfreq:  the same as the input passfreq'.
series:  the name of the input time series as a character
       string.

DETAILS:
       Missing values at the ends of the series are excluded from
       the computations.

       The phase and amplitude of the demodulation are smoothed
       with a least squares approximation to a lowpass filter.
       The cutoff of the filter is half-way between passfreq' and
       stopfreq'.  The number of terms in the filter is
       floor(2*max(2,d)+1)', where
       d=frequency(x)/(stopfreq-passfreq)'.

BACKGROUND:
       Complex demodulation is a technique for analyzing
       nonstationary time series by estimating the instantaneous
       amplitude and phase of a given harmonic component.  See
       Bloomfield (1976) for an account of the method.  To better
       understand the results of complex demodulation several
       lowpass filters should be tried:  the smaller the pass
       band, the less instantaneous in time but more specific in
       frequency is the result.

REFERENCES:
       Bloomfield, P. (1976).  Fourier Analysis of Time Series:
       An Introduction.  Wiley, New York.

EXAMPLES:
   p <- spectrum(sunspots, span=c(7,5), plot=F)
   fpeak <- p$freq[p$spec==max(p$spec)] # the frequency of the spectral peak
   d <- demod(sunspots, fpeak, .02, .04)

   par(mar=c(5,4,4,4)+.1)
```

```
tsplot(d$amp, lty=1, ylab="amplitude")
legend(1750, 40, c("amplitude", "phase"), lty=1:2)
par(new=T)
tsplot(d$phase, lty=2, main="Sunspot Demodulation", axes=F)
axis (4)
mtext(side=4, line=3, "phase")
  # a sloped phase plot indicates that the frequency used
  # is slightly off the mark
```

density	Estimate Probability Density Function	density

DESCRIPTION

Returns x and y coordinates of a non-parametric estimate of the probability density of the data. Options include the choice of the window to use and the number of points at which to estimate the density.

USAGE

```
density(x, n=50, window="g", na.rm=F, width=<<see below>>,
         from=<<see below>>, to=<<see below>>, cut=<<see below>>)
```

REQUIRED ARGUMENTS

x   vector of observations from the distribution whose density is to be estimated. Missing values are allowed if na.rm is TRUE.

OPTIONAL ARGUMENTS

n   the number of equally spaced points at which to estimate the density.

window   character string giving the type of window used in the computations. One of: "cosine", "gaussian", "rectangular", "triangular" (one character is sufficient).

na.rm   logical flag: should missing values be removed before estimation?

```
 width  width of the window.  The default is the width of a histogram
        bar which is determined by log(length(x), base=2) + 1 bars to
        cover the range of x.  The standard error of a Gaussian window
        is width/4. For the other windows width is the width of the
        interval on which the window is non-zero.

from, to  the n estimated values of density are equally spaced between
          from and to. The default is the range of the data extended
          by width*cut.

   cut  The fraction of the window width that the x values are to be
        extended by. The default is .75 for the Gaussian window and
        .5 for the other windows.

VALUE

list with two components, x and y, suitable for giving as an
argument to one of the plotting functions.

x vector of n points at which the density is estimated.
y density estimate at each x point.
```

fft	Fast Fourier Transform	fft

```
DESCRIPTION:
       Performs the fast Fourier transform  on  a  vector  or  an
       array of either numeric or complex values.

USAGE:
       fft(z, inverse=F)

REQUIRED ARGUMENTS:
z:     numeric or complex vector or array.

OPTIONAL ARGUMENTS:
inverse:    logical flag: if TRUE, the inverse of  the  transform
       is computed.

VALUE:
       the unnormalized discrete Fourier transform of  the  input
```

data z (or the inverse transform if inverse=TRUE). This is of mode "complex".

DETAILS:

The fast Fourier transform algorithm is used. No padding of the input data is done; length(z) is factored, if possible, and the factorization is used in the algorithm. Therefore, if length(z) is prime, there will be no advantage in using fft over computing the transform explicitly.

If z is an array, fft will return the multi-dimensional unnormalized discrete Fourier transform of z-a complex array with the same shape as z. Therefore, using fft on a multivariate time series will not compute the time transform.

filter	Apply a Filter to a Time Series	filter

DESCRIPTION:

Returns a time series that is the filtered version of x'.

USAGE:

```
filter(x, filter, method="convolution", sides=2, circular=F,
       init=<<see below>>)
```

REQUIRED ARGUMENTS:

x: a univariate or multivariate regular time series, or a vector, or a matrix with each column representing a univariate component of a multivariate series. Missing values are allowed only at the ends of series. Factor data are not allowed.

filter: a vector of filter coefficients - the first element is for the last timepoint used and the last element of filter' is for the earliest timepoint used. The same filter coefficients are applied to each univariate component of the time series, i.e., to each column of x'.

OPTIONAL ARGUMENTS:

method: either the string "convolution"' or "recursive"' (only

```
       the start of a string is necessary).
sides:    for convolution filters only: if sides=2',  the  filter
       coefficients are centered at lag 0' (so there should be an
       odd number  of  coefficients).  If  sides=1',  the  first
       coefficient  is  the  lag  0'  coefficient so the filtered
       series depends only on past  and  present  values  of  the
       original series.
circular:   for convolution filters  only:  if  TRUE',  wrap  the
       filter  around  the ends of the series.  If FALSE', do not
       wrap and set any undefined elements of the filtered series
       to NA'.
init:     for recursive filters only: matrix of the values of the
       filtered  series  just prior to the beginning of the input
       time series.  Must have nrow(init) =  length(filter)'  and
       ncol(init)'  equal  to number of univariate time series in
       x' (ncol(x)').  The first  row  of  init'  refers  to  the
       values  of the series one time step back from the start of
       x'.  The default is a matrix of zeros.   A  typical  init'
       might  be  the  mean  of each input series repeated to the
       length of the filter, i.e.,

       init=matrix(rep(apply(x,2,mean),length(filter)),
              nrow=length(filter),byrow=T)'.

VALUE:
       a regular time series with the same  dimensions  and  time
       parameters as x' containing the filtered series.
```

spectrum	Estimate Spectra	spectrum

```
DESCRIPTION:
       Estimates spectra using either a smoothed  periodogram  or
       autoregressive parameters.

USAGE:
       spectrum(x, method="pgram", plot=T, ...)

REQUIRED ARGUMENTS:
x:     a univariate or multivariate time series, or a vector,  or
       a  matrix  with  a  univariate series per column.  Missing
```

```
       values are allowed only at the ends.  If method="ar", then
       this can also be a list as returned by the ar function.

OPTIONAL ARGUMENTS:
method:   either the string "ar" or "pgram" depending on  whether
       an autoregressive model or a smoothed periodogram is to be
       used in estimating the spectrum.
...:   additional arguments.

       If method="pgram", then the possible arguments  and  their
       defaults  are:  spans=1,  taper=0.1,  pad=0, detrend=TRUE,
       demean=FALSE. The most useful  of  these  is  spans  which
       should  be  a  vector  of odd integers specifying spans of
       Daniell smoothers used on the periodogram.  See spec.pgram
       for further information.

       If method="ar", additional arguments can  be  n.freq  (the
       number  of  frequencies  at  which  the  spectrum is to be
       estimated - the default is 1 plus half the length  of  the
       series),  and frequency (the frequency of the observations
       of the time series - the default is the frequency  of  x).
       See spec.ar.

VALUE:
       a list as returned by spec.pgram if method="pgram", or  as
       returned  by  spec.ar  if method="ar".  The most important
       components are freq (the frequencies at which the spectrum
       is  estimated)  and  spec  (a vector or matrix of spectral
       estimates in decibels).  If the  series  is  multivariate,
       components  coh  and phase give the coherency and phase of
       the series.
```

stl	Seasonal Decomposition of a Time Series	stl

```
DESCRIPTION:
       Decompose  a  time  series  into  frequency  components  of
       variation by a sequence of loess smoothings.

USAGE:
       stl(time.series, ss.window = NULL, ss.degree = 0, ss.robust = F,
```

```
            s.window = NULL, s.degree = 2, fc.window = NULL, fc.degree = 2,
            control, ...)

REQUIRED ARGUMENTS:
time.series:   a univariate time series that is to be decomposed.

OPTIONAL ARGUMENTS:
ss.window:     a single value that is character or  numeric.   If
       it is the character value "periodic"', an exactly periodic
       seasonal  component  is  returned.  If  the  argument   is
       numeric,  it  specifies  the  length  of the window in the
       loess smoothing for each seasonal subseries in  phase  one
       of  the  decomposition.  This  must  be  an odd integer. A
       seasonal subseries consists of the values of the  seasonal
       component at a given position in the seasonal cycle (e.g.,
       January values of a monthly series with a  yearly  cycle);
       the  number  of  subseries is frequency(time.series)'.  As
       ss.window' increases, each subseries becomes smoother.  If
       missing, no seasonal component is computed.
ss.degree:     degree of locally-fitted polynomial in  the  loess
       smoothing  for  the seasonal subseries in phase one of the
       decomposition.  Values of 0 (locally-constant fitting) and
       1  (locally-linear  fitting) are allowed.  Using the first
       results in increased stability of the seasonal  component.
       The default is 0.
ss.robust:     a logical.  If TRUE', a robust  fitting  procedure
       is used in phase one.
s.window:   length of the window of the phase-two loess smoothing
       of the seasonal component from phase one.
s.degree:   degree of  locally-fitted  polynomial  in  the  loess
       smoothing  of  the  seasonal  in  phase  two.  Values of 1
       (locally-linear fitting) and 2 (locally-quadratic fitting)
       are allowed. Normally, the value should be 2, the default.
fc.window:     vector of lengths of windows in  loess  smoothings
       for  other  frequency  components  in  phase  two  of  the
       decomposition. If a seasonal component is present,  it  is
       subtracted from time.series'; if not, the procedure begins
       with time.series'. A frequency component is computed by  a
       loess  fit  with  the  window  length  equal  to the first
       element of fc.window', the component is  removed,  another
       component  is computed with the window length equal to the
       second element of fc.window', and so forth. In most cases,
```

the values of the argument should be decreasing, that is, the frequency bands of the fitted components should increase. If the argument ss.robust' is TRUE', the robustness weights from the phase one seasonal fit are used as weights in the loess fitting.

fc.degree: vector of degrees of locally-fitted polynomial in the loess smoothings for the frequency components specified in fc.window'. Values of 1 and 2 are allowed. If the length of fc.degree' is less than that of fc.window', the former is expanded to the length of the latter using rep'; thus, giving the value 1 specifies a degree of 1 for all components.

control: a list that controls certain parameters in both phase-one and phase-two fitting. The list can be created by the function stl.control', whose documentation describes the options. Most users do not need to alter the default values for these parameters.

...: arguments of the function stl.control' can also be specified directly in the call to stl' without using the argument control'. See help file for stl.control' for details.

VALUE:

an object of class "stl"' representing the fitted model.

DETAILS:

There are methods for these objects including print.stl', plot.stl', and specs.stl'. The object is a list whose components include the computed frequency components as time series with the same attributes as time.series'. The names of these components are included in the printing of print.stl', and the components together with the series are graphed by plot.stl'.

specs.stl' can be used to recall the specifications used to compute an object of class stl'. Some of these values may be slightly different from the input values given the algorithm requirements.

stl' carries out a decomposition of a time series into frequency components of variation by a sequence of loess smoothings. In phase one, the backfitting algorithm produces a seasonal component with a period equal to that of frequency(time.series)'. The subseries of this component are smooth though time; that is, for a monthly series with period 12, the January values are smooth, the February values are smooth and so forth. But the values by design do not have to be smooth from one month to the next. If this is desired, the phase-one seasonal is replaced by a smoothing of it in the first step of phase two. The seasonal is subtracted from time.series' and other frequency components are computed sequentially. stl' is a replacement for the sabl' procedure.

REFERENCES:

Cleveland, R. B., Cleveland, W. S., McRae, J. E. and Terpening, I. (1990). STL: A Seasonal-Trend Decomposition Procedure Based on Loess. Journal of Official Statistics, 6, 3-73.

9.2 Additional S functions

armasp	Estimate the spectrum via ARMA method	armasp

DESCRIPTION:

```
Find spectral density for
x(t) - phi(1)X(t-1) - ... - phi(p)x(t-p) =
+ e(t) - theta(1)e(t-1) - ... - theta(q)e(t-q).
```

USAGE:

```
armasp(phi=c(0), theta=c(0), sigma=1, Q=256)
```

REQUIRED ARGUMENTS:

```
phi: AR coefficients
theta: MA coefficients
sigma: Variance of the innovation
Q:      number of frequencies
```

cos.sim	Simulate a harmonic series	cos.sim

```
DESCRIPTION:

        Generate a harmonic series consisting of several frequencies
of length n with Gaussian white noise.

USAGE:
       cos.sim(a=c(2,1,6,7),
                 b=c(3,3,1,4),f=c(1/12,1/36,1/6,1/3),n=200)

REQUIRED ARGUMENTS:
a,b: Amplitudes of the cosine and sine
f: The corresponding frequencies
n:      number of observations
```

pgram	Compute the periodogram	pgram

```
DESCRIPTION:
       Compute the periodogram of a time series.

USAGE:
       pgram(x,rm.mean=T)

REQUIRED ARGUMENTS:
x:     A time series
rm.mean:   Remove the mean if it is TRUE.
```

pgram.test	Fisher's test	pgram.test

```
DESCRIPTION:
       Compute Fisher's exact and approximated test.

USAGE:
       pgram.test(x,exact=T)

REQUIRED ARGUMENTS:
x:     A time series
exact:   exact test if it is TRUE, otherwise it's an approximated test.
```

fisther.p	P-value for Fisher's test	fisther.p

```
DESCRIPTION:
       Compute the P-value for Fisher's exact test.

USAGE:
       fisher.p(a,n)

REQUIRED ARGUMENTS:
a:     the periodogram
n:     length of a time series
```

minmax	Find the max, min, and the locations	minmax

```
USAGE:
       minmax(y)

REQUIRED ARGUMENTS:
y:     an array of numbers
```

auspe	Estimate and display the spectra	auspe

```
DESCRIPTION:
       Function to estimate and plot the auto spectral density
       of a time series x using window ioptw and M

USAGE:
       auspe(x, ioptw=4, M=15, logs=F, ci=T)

REQUIRED ARGUMENTS:
x:     A time series
ioptw: window option 1 ... 8
M:     scale parameter for window
logs:  flag for logarithmic scale
ci:    confidence intervals
```

sp.plot	Plotting spectral quantities	sp.plot

```
DESCRIPTION:
       Function to plot the spectral quantities

USAGE:
       sp.plot(f, y.lab="Spectrum", p.type="l")

REQUIRED ARGUMENTS:
f:     spectrum
y.lab: any character string
p.type:   A character string for plotting options
```

ci.plot	Plotting spectral quantities and CI	ci.plot

```
DESCRIPTION:
       Function to plot the spectral quantities and confidence intervals.

USAGE:
       ci.plot(f,fl,fu,y.lab="Spectrum")

REQUIRED ARGUMENTS:
f:     spectrum
fl:    lower CI
fu:    upper CI
y.lab: any character string
```

coher	Estimate and display the coherence and phase	coher

```
DESCRIPTION:
       Estimate the spectra, coherence and phase
       of bivariate series x1 and x2 using window ioptw and M

USAGE:
       coher(x1,x2,ioptw,M)

REQUIRED ARGUMENTS:
```

```
x1,x2:  two time series
ioptw:  window option 1 ... 8
M:      scale parameter for window
```

`cohtest`	Display and test the coherence	`cohtest`

```
DESCRIPTION:

       Function to estimate and test the coherence
       of bivariate series x1 and x2 using window ioptw and M

USAGE:
       cohtest(x1,x2,ioptw,M)

REQUIRED ARGUMENTS:
x1,x2:  two time series
ioptw:  window option 1 ... 8
M:      scale parameter for window
```

`crossp`	Estimate and display the co- and quad-spectra	`crossp`

```
DESCRIPTION:

       Function to estimate and plot the co- and quad-spectra
       of bivariate series x1 and x2 using window ioptw and M

USAGE:
       crossp(x1,x2,ioptw,M)

REQUIRED ARGUMENTS:
x1,x2:  two time series
ioptw:  window option 1 ... 8
M:      scale parameter for window
```

`divpoly`	Divide two polynomials	`divpoly`

```
DESCRIPTION:

       Function to find the first n coefficients of
       (1+sum_1^q b(j)z^j) / (1+sum_1^p a(i)z^i).

USAGE:
       divpoly(a,b,n)

REQUIRED ARGUMENTS:
a,b:    coefficients of the two polynomials
n:      number of coefficients to be listed
```

estfilt	Estimate the best filter coefficients	estfilt

```
DESCRIPTION:

       Function to estimate the best filter coefficients
       of bivariate series x1 and x2 using window ioptw and M.
       It uses the inverse Fourier transform on A=f21/f11.

USAGE:
       estfilt(x1,x2,ioptw,M,kn)

REQUIRED ARGUMENTS:
x1,x2:  two time series
ioptw:  window option 1 ... 8
M:      scale parameter for window
kn+1:   number of beta's

VALUE:
       beta[1:(kn+1)] (coefficients of the transfer function)
```

filtest	Estimate the transfer functions	filtest

```
DESCRIPTION:

       Function to estimate the transfer function of a filter
       of bivariate series x1 and x2 using estfilt and prewhitening
```

```
USAGE:
       filtest(x1,x2,ioptw,M,kn)

REQUIRED ARGUMENTS:
x1,x2:  two time series
ioptw:  window option 1 ... 8
M:      scale parameter for window
kn+1:   number of beta's

VALUE:
       coef (coefficients of the transfer function)
```

indtest	Test the independence of two TS	indtest

```
DESCRIPTION:

       Function to test the indepdence of x1 and x2 using autocorr.

USAGE:
       indtest(x1,x2, nlag=20, alpha=.05)

REQUIRED ARGUMENTS:
x1,x2:  two time series
nlag:   max lag
alpha:  alpha level

VALUE:
       simultaneous upper confidence limit
```

prewhite	Prewhittening	prewhite

```
DESCRIPTION:

       Function to use AR filters to prewhiten and align two
       time series

USAGE:
```

```
       prewhite(x,y)

REQUIRED ARGUMENTS:
x,y:  two time series

VALUE:
n:    the length of the final prewhited series.
x:    the prewhited x series
y:    the prewhited y series
ax:   the AR coef of the prewhited x series
ay:   the AR coef of the prewhited y series
```

spe	Auto-spectrum estimation	spe

```
DESCRIPTION:

        Function to find window estimate of the auto-spectrum.

USAGE:
       spe(x, ioptw, M, alpha=.05)

REQUIRED ARGUMENTS:
x:     A time series
M:     scale parameter for window
ioptw:  window option 1 ... 8

1   Truncated
2   Bartlett
3   Tukey
4   Parzen
5   Bohman
6   Daniell
7   Bartlett-Priestley
8   Parzen-Cogburn-Davis
alpha: for confidence interval

VALUE:
f: spectrum
c: critical value for confidence intervals
```

xspe	Cross-spectrum estimation	xspe

DESCRIPTION:

```
        Function to find window estimate of the cross-spectra.

USAGE:
       spe(x1, x2, ioptw, M, alpha=.05)

REQUIRED ARGUMENTS:
x1,x2:  time series
M:      scale parameter for window
ioptw:  window option 1 ... 8

        1   Truncated
        2   Bartlett
        3   Tukey
        4   Parzen
        5   Bohman
        6   Daniell
        7   Bartlett-Priestley
        8   Parzen-Cogburn-Davis
alpha: for confidence interval

VALUE:
re:     co-spectrum (real part)
im:     (-1) quad-spectrum (imadinary part)
c:      critical value for confidence intervals
```

wn	Simulate white noise series	wn

DESCRIPTION:

```
        Generate a white noise time series of length n having
distribution specified by dist.

USAGE:
       wn(seed,n,dist=1)
```

```
REQUIRED ARGUMENTS:

seed: is a nonnegative integer seed for the random number generator
(if seed=0, then the output seed from last call is used)
n: number of observations
dist =  1: normal (0,1)
2: unif (0,1)
3: exponential (1)
4: logistic (mean 0 and var pi^2/3)
5: Cauchy
6: Extreme value (log(exp(1))
7: lognormal
8: double exponential
```

Chapter 10

Appendix: Data sets and S functions

The following datasets are part of `library(tslib)`.

1. `ahh` – Speech record of the syllable AHH sampled at 10,000 points per second (n=1020). (Shumway, 1988)

2. `births` – Monthly registered live births in thousands from January 1948 through January 1979 (n=373). Data were taken from the Monthly Vital Statistics Reporter, National Center for Health Statistics, US Public Health Service (current values may have been adjusted). (Shumway, 1988)

3. `bp` – 460 blood pressure measurements (twice/day) (Shumway, 1988)

4. `d12` – mortality among male (`mdeaths`) and female (`fdeaths`). See MASS.

5. `ep` – Averages of 30 electral potential (EP) measurements: at scalp (col 1), wrist (col 2), stimulant (col 3).

6. `eriel` – Levels of Lake Erie, 1921-1970.

7. `gasfurn` – Gas furnace (Box and Jenkins, 1971).

   ```
   rate # input rate
   co2 # carbon dioxide
   ```

8. `lh` – Four time series of lutenizing hormone in blood samples taken from a woman at ten-minutes intervals over an eight-hour period. The cyclic patterns are important features for studying the human reproductive system (Murdoch et al, 1985).

```
lh1
lh2     # cycle 1, late follicular phase
lh3     # cycle 1, early follicular phase
lh4     # cycle 2, late follicular phase
```

9. `lung` – London mortality, weather, and pollution for the winter of 1958 (n=102). Taken from Table A.1 of Shumway (1988).

```
atm     # adjusted total mortality
smk     # British smoke
so2     # sulfur dioxide
tmp     # temperature (C)
rhu     # relative humidity in percent
```

10. `pm` – Mortality and environmental series. The daily deaths of residents of the Birmingham, Alabama, metropolitan area from 1985–1988.

```
id      # day within dataset
day     # day within month
mon     # month (1=Jan, 12=dec, etc)
yr      # year
tmax    # daily max temp (C)
tmax1   # tmax lagged 1 day
tmax2   # tmax lagged 2 days
tmin    # daily min temp (C)
tmin1   # tmax lagged 1 day
tmin2   # tmax lagged 2 days
mnsh    # ave daily specific humidity (g/kg)
mnsh1   # mnsh lagged 1 day
mnsh2   # mnsh lagged 2 days
awnd    # ave daily wind speed (m/s)
awnd1   # awnd lagged 1 day
awnd2   # awnd lagged 2 days
pres    # ave daily station pressure (mb)
pres1   # pres lagged 1 day
pres2   # pres lagged 2 days
pm10    # PM10 reading, ave over 3 days
tmort   # total non-accidental deaths for the day
mort65  # total non-accidental deaths, aged 65+
```

11. `unemp` – Monthly values of economic indicators, 1948-1979 (n=372). Taken from Table A.5 of Shumway (1988).

```
     index   # Federal Reserve Board Production Index
     noune   # Number of unemployed
```

Examples

```
> attach(lh)
> pgram.test(sunspots)
$test:
[1] 229.3131

$pval:
[1] 0

> pgram.test(sunspots,exact=F)
$test:
[1] 229.3131

$pval:
[1] 0

> pgram.test(lh4)
$test:
[1] 3.657647

$pval:
[1] 0.4547011

> pgram.test(lh4,exact=F)
$test:
[1] 3.657647

$pval:
[1] 0.4517516
```

```
pgram <- function(x, rm.mean=T)
#----------------------------------------------------------
#
# Compute the periodogram
#
#----------------------------------------------------------
{
   if (rm.mean) x <- x - mean(x)
```

```
   per <- Mod(fft(x))^2 / (n <- length(x))
    per[1:((n / 2) + 1)]
}

pgram.test <- function(x,exact=T)
#----------------------------------------------------------
#
#  Fisher's exact and approximated test
#
#----------------------------------------------------------
{
  n <- length(x)
  q <- (n-1) %/% 2 # number of freq's
  q1 <- q+1
  per <- pgram(x) # peridogram
  a <- q*max(per[2:q1])/sum(per[2:q1]) # Fisher's test

  if (exact) p <- fisher.p(a,n) # exact P
     else p <- 1-(1-exp(-a))^q # apprx P

  list(test=a, pval=p)
}

fisher.p <- function(a,n)
#----------------------------------------------------------
#
#  Compute the P-value for Fisher's test
#  Brockwell and Davis (1991, p.339)
#
#----------------------------------------------------------
{
 q <- (n-1) %/% 2
 bj <- 1 # first binomial coef is 1
 pr <- 1 # first term
 for (j in 1:q)
  {
   bj <- bj*(q-j+1)/j
   if (1-j*a/q <= 0) return (1-pr)
   else pr <- pr + (-1)^j*bj*(1-j*a/q)^(q-1)
  }
  1-pr
}
```

```
minmax <- function(y)
#---------------------------------------------------------
#
# Find the max, min, and the locations
#
#---------------------------------------------------------
{
  n <- length(y)
  yo <- order(y)
  ymin.f <- (yo[1]-1)/(2*n)
  ymax.f <- (yo[n]-1)/(2*n)
  list(ymin = y[yo[1]], ymin.f = ymin.f, ymax = y[yo[n]], ymax.f = ymax.f)
}

auspe <- function(x, ioptw=4, M=15, logs=F, ci=T)
#---------------------------------------------------------
#
#  Function to estimate and plot the auto spectral density
#  of a time series x using window ioptw and M
#
#  Arguments:
#    x:         time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#    logs:      flag for logarithmic scale
#    ci:        confidence intervals
#
#  Value: None
#
#---------------------------------------------------------
{
  x.spe <- spe(x, ioptw, M)
  f11 <- x.spe$f
  crv <- x.spe$c
  n <- length(f11)

  if ( logs )
  {
    f11 <- log(f11)
    fl <- f11-crv
    fu <- f11+crv
```

```
    if (ci) ci.plot(f11,fl,fu,y.lab="Log-spectrum")
     else sp.plot(f11,y.lab="Log-spectrum")
  }
  else
  {
    crv <- exp(crv)
    fl <- f11/crv
    fu <- f11*crv

    if (ci) ci.plot(f11,fl,fu,y.lab="Spectrum")
     else sp.plot(f11,y.lab="Spectrum")
  }
}

sp.plot <- function(f, y.lab="Spectrum", p.type="l")
# -----------------------------------------------------
#
#   Function for plotting spectral quantities
#
# -----------------------------------------------------
{
   q <- length(f)
   plot(c(0:(q-1))/(2*(q-1)),f,xlab="frequency",
        ylab=y.lab,type=p.type)
}

ci.plot <- function(f,fl,fu,y.lab="Spectrum")
# -----------------------------------------------------
#
#   Function for plotting spectral quantities and CI
#
# -----------------------------------------------------
{
  n <- length(f)
  plot(c(0,.5), c(min(fl), max(fu)), xlab = "frequency",
      ylab = y.lab, type = "n")
  lines(c(0:(n - 1))/(2 * (n - 1)), f)
  lines(c(0:(n - 1))/(2 * (n - 1)), fl, lty=2)
  lines(c(0:(n - 1))/(2 * (n - 1)), fu, lty=2)
}
```

```
coher <- function(x1,x2,ioptw,M)
#---------------------------------------------------------
#
#  Function to estimate the spectra, coherence and phase
#  of bivariate series x1 and x2 using window ioptw and M
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#
#  Value: None
#
#---------------------------------------------------------
{
 f11 <- spe(x1,ioptw,M)$f
 f22 <- spe(x2,ioptw,M)$f
 f12 <- xspe(x1,x2,ioptw,M)
 c12 <- f12$re
 q12 <- f12$im

 amp <- polar(c12,q12)$amp
 phi <- polar(c12,q12)$phase
 coh <- amp^2/(f11*f22)

 par(mfrow=c(2,2))

 sp.plot(f11, y.lab = "Spectrum f11")
 sp.plot(f22, y.lab = "Spectrum f22")

 ilam2 <- c(2,2./3.,.795,.539,.586,1,1.2,1.66)
 n <- length(x1)

 w12 <- sqrt(coh)
 se <- sqrt(abs(1-coh)*M*ilam2[ioptw]/(2*n))
 lo <- w12-2*se
 up <- w12+2*se
 ci.plot(w12, lo, up, y.lab="Coherence")

 se <- amp*sqrt(abs((1/coh)-1)*M*ilam2[ioptw]/(2*n))
 lo <- phi-2*se
 up <- phi+2*se
```

```
 ci.plot(phi, lo, up, y.lab="Phase")

 par(mfrow=c(1,1))
 invisible()
}

cohtest <- function(x1,x2,ioptw,M)
#---------------------------------------------------------
#
#  Function to estimate and test the coherence
#  of bivariate series x1 and x2 using window ioptw and M
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#
#  Value: None
#
#---------------------------------------------------------
{
  f11 <- spe(x1,ioptw,M)$f
  f22 <- spe(x2,ioptw,M)$f
  f12 <- xspe(x1,x2,ioptw,M)
  c12 <- f12$re
  q12 <- f12$im

  amp <- polar(c12,q12)$amp
  phi <- polar(c12,q12)$phase
  coh <- amp^2/(f11*f22)

  n <- length(f11)
  par(mfrow=c(2,2))

  sp.plot(f11, y.lab = "Spectrum f11")
  sp.plot(f22, y.lab = "Spectrum f22")

  ilam2 <- c(2,2./3.,.795,.539,.586,1,1.2,1.66)

  const <- (n - M*ilam2[ioptw])/(M*ilam2[ioptw])
  fstat <- const*coh/(1-coh)
  nc <- 2*const
```

```
  ft <- qf(.95,2,nc)
  sp.plot(fstat, y.lab = "F-stat")
  lines(c(0:(n - 1))/(2 * (n - 1)), rep(ft,n),lty=2)

  sp.plot(phi, y.lab = "Phase")
  par(mfrow=c(1,1))
  invisible()
}

crossp <- function(x1,x2,ioptw,M, ci=T)
#----------------------------------------------------------
#
#  Function to estimate, plot the co and quad spectra
#  of bivariate series x1 and x2 using window ioptw and M
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#
#  Value: None
#
# TIMESLAB (p.338)
#----------------------------------------------------------
{
  f11 <- spe(x1,ioptw,M)$f
  f22 <- spe(x2,ioptw,M)$f
  f12 <- xspe(x1,x2,ioptw,M)
  crv <- f12$c
  c12 <- f12$re
  q12 <- -f12$im

# Compute the avar

  v1 <- f11*f22
  v2 <- c12^2-q12^2
  sd <- matrix(c(sqrt((v1+v2)/2),sqrt((v1-v2)/2)),ncol=2)

  f <- matrix(c(c12,q12),ncol=2)
  n <- length(f[,1])

  par(mfrow=c(1,2))
```

```
  for (j in (1:2))
  {
      s <- f[,j]
      se <- sd[,j]*crv
      cil <- s - se
      ciu <- s + se
      if (j==1)
        {
          y.lab <- "Co-spectrum"
        }
      else
        {
          y.lab <- "Quad-spectrum"
        }
      if (ci) ci.plot(s, cil, ciu, y.lab = y.lab)
        else sp.plot(s,y.lab=y.lab)
  }
  par(mfrow=c(1,1))
  invisible()
}

divpoly <- function(a,b,n)
# ------------------------------------------------
#
#  Find the first n coefficients of
#      (1+sum_1^q b(j)z^j) / (1+sum_1^p a(i)z^i)
#
# ------------------------------------------------
{
  p <- length(a)
  q <- length(b)
  if (p+q == 0) return(rep(0,p+q))
  g <- c(b,rep(0,n-q))
  if (p > 0)
  {
    for (i in 1:n)
    {
      c0 <- g[i]
      for (j in 1:min(i,p))
      {
        c1 <- 1
        if (i-j>0) c1 <- g[i-j]
```

```
        c0 <- c0 - a[j]*c1
      }
      g[i] <- c0
    }
  }
  g
}

estfilt <- function(x1,x2,ioptw,M,kn)
#----------------------------------------------------------
#
#  Function to estimate the best filter coefficients
#  of bivariate series x1 and x2 using window ioptw and M.
#  It uses the inverse Fourier transform on A=f21/f11.
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#    kn+1:        number of beta's
#
#  Value: beta[1:(kn+1)] (coefficients of the transfer function)
#
#  See Theorem 4.1.6 of TIMESLAB (p.321)
#
#----------------------------------------------------------
{
  f11 <- spe(x1,ioptw,M)$f
  f22 <- spe(x2,ioptw,M)$f
  f21 <- xspe(x2,x1,ioptw,M)
  c21 <- f21$re
  q21 <- f21$im

  c211 <- c21/f11
  q211 <- q21/f11

  q <- length(c21)
  c211 <- c(c211,c211[(q-1):2])      # extend to [0,1] via sym
  q211 <- c(q211,-q211[(q-1):2])     # extend to [0,1] via antisym

  z <- c211 + q211*1i                # Form complex array for FFT
```

```
  beta <- Re(fft(z,inv=T))/(2*q-2)
  beta[1:(kn+1)]
}

filtest <- function(x1,x2,ioptw,M,kn)
#----------------------------------------------------------
#
#  Function to estimate the transfer function of a filter
#  of bivariate series x1 and x2 using estfilt and prewhitening
#
#  Arguments:
#    x1, x2:    time series
#    ioptw:     window option 1 ... 8
#    M:         scale parameter for window
#    kn+1:        number of beta's
#
#  Value: coef  (coefficients of the transfer function)
#
#  See TIMESLAB (p.344)
#
#----------------------------------------------------------
{
  z <- prewhite(x1,x2)
  x1 <- z$x
  x2 <- z$y
  ax <- -z$ax
  ay <- -z$ay

  beta <- estfilt(x1,x2,ioptw,M,kn)

  beta0 <- beta[1]
  beta <- beta/beta0

  kn1 <- kn+1
  gama <- multpoly(beta[2:kn1],ax)
  betay <- divpoly(a=ay,b=c(),40)
  gama <- multpoly(betay,gama)
  gama <- gama*beta0

  list(coef=gama[1:kn], b0=beta0)
}
```

```
indtest <- function(x1, x2, nlag=20, alpha=.05)
# -----------------------------------------------
#
#  To test the independence of two time series
#  using cross-correlations
#
#  Argument: x1, x2, nlag, alpha
#
#  Value:    upper confidence limit
#
# -----------------------------------------------
{
  z <- prewhite(x1,x2)
  zz <- matrix(c(z$x,z$y),ncol=2)
  zacf <- acf(zz, lag.max=nlag)         # plot acf

  n <- length(z$x)      # for CI
  qnorm( (1+(1-alpha)^(1/(2*nlag+1)))/2) / (n^.5)       # upper CI
}

multpoly <- function(a,b)
# -----------------------------------------------
#
#  Multiply 1+sum_1^p a(i)z^i and 1+sum_1^q b(j)z^j
#
# -----------------------------------------------
{
  p <- length(a)
  q <- length(b)
  g <- rep(0,p+q)
  g <- g + c(a,rep(0,q))
  g <- g + c(b,rep(0,p))
  for (i in 1:p)
  {
    for (j in 1:q) g[i+j] <- g[i+j] + a[i]*b[j]
  }
  g
}

polar <- function(zr,zi)
#-----------------------------------------------------
#
```

```
#  Function for converting to polar representation
#
#-----------------------------------------------------
{
 return(list(amp=sqrt(zr^2+zi^2), phase=atan(zi,zr)))
}

prewhite <- function(x,y)
# -----------------------------------------------------
#
# Use AR filters to prewhiten and align two time series
#
# -----------------------------------------------------
{
  n <- length(x)
  x <- x - mean(x)
  y <- y - mean(y)

  x.ar <- ar(x)
  y.ar <- ar(y)
  px <- x.ar$order
  py <- y.ar$order

  if (px == 0) xx <- x
  if (px > 0) {
    xx <- x.ar$resid
    xx <- xx[(px+1):n]
  }

  if (py == 0) yy <- y
  if (py > 0) {
    yy <- y.ar$resid
    yy <- yy[(py+1):n]
  }

  nx <- n - px
  ny <- n - py
  nn <- nx

  if (nx > ny) {
    nn <- ny
    n1 <- nx-ny
```

```
    n1 <- n1 + 1
    xx <- xx[n1:nx]
  }

  if (nx < ny) {
    nn <- nx
    n1 <- ny-nx
    n1 <- n1 + 1
    yy <- yy[n1:ny]
  }

  list(n=nn, x=xx, y=yy, ax=x.ar$ar, ay=y.ar$ar)
}

spe <- function(x, ioptw, M, alpha=.05)
#------------------------------------------------------------
#
#   Function to find auto-spectra using scale parameter M,
#   window number ioptw:
#
#   1   Truncated
#   2   Bartlett
#   3   Tukey
#   4   Parzen
#   5   Bohman
#   6   Daniell
#   7   Bartlett-Priestley
#   8   Parzen-Cogburn-Davis
#
#   The spectral estimator and the constant c used in confidence
#   intervals are returned in a list.
#
#   TIMESLAB (p.185)
#
#--------------------------------------------------------------
{
   n <- length(x)
   if(M >= n || ioptw < 1 || ioptw > 8
      || alpha <= 0. || alpha >= 1.)
   stop("Illegal Input to spe()")

   x.acf <- acf(x,lag.max=n-1,type="cov",plot=F)
```

```
z <- x.acf$acf[,1,1] # c11

z1 <- rep(0,n)

if(ioptw == 1) z1[1:(M+1)] <- rep(1, M+1)

else if(ioptw == 2)  z1[1:(M+1)] <- 1 - (c(0:M) / M)

else if(ioptw == 3)  z1[1:(M+1)] <- .54+.46*cos(pi*c(0:M)/M)

else if(ioptw == 4) {
   u  <- c(1:M)/M
   u1 <- u[u<=.5]
   u2 <- u[u>.5]
   z1[1:(M+1)]  <- c(1,1-6*u1^2+6*u1^3,2*(1-u2)^3)
}

else if(ioptw == 5) {
   u   <- c(0:M) / M
   piu <- pi * u
   z1[1:(M+1)]  <- (1 - u) * cos(piu) + sin(piu) / pi
}

else if(ioptw == 6) {
   piu <- pi * (c(1:(n-1)) / M)
   z1[1:n] <- c(1,sin(piu)/piu)
}

else if(ioptw == 7) {
   piu <- pi * (c(1:(n-1)) / M)
   z1[1:n] <- c(1,3*((sin(piu)/piu) - cos(piu)) / (piu*piu))
}

else if(ioptw == 8) {
   u <- c(0:(n-1))/M
   z1[1:n] <- 1 /(1+u^4)
}

z <- c(z1[1] * z[1], 2 * z1[2:n] * z[2:n])

z <- Re(fft(z))
z <- z[1:((n / 2) + 1)]
```

```
   ilam2 <- c(2,2./3.,.795,.539,.586,1,1.2,1.66)
   fac <- qnorm(1-(alpha/2)) * sqrt(M*ilam2[ioptw]/n)

   return(list(f=z,c=fac))

}

xspe <- function(x1, x2, ioptw, M, alpha=.05)
#------------------------------------------------------------
#
#   Function to find  cross-spectra using scale parameter M,
#   window number ioptw:
#
#   1   Truncated
#   2   Bartlett
#   3   Tukey
#   4   Parzen
#   5   Bohman
#   6   Daniell
#   7   Bartlett-Priestley
#   8   Parzen-Cogburn-Davis
#
#   The spectral estimator and the constant c used in confidence
#   intervals are returned in a list.
#
#   TIMESLAB (p.335)
#
#------------------------------------------------------------
{
   if (length(x1) != length(x2))
     stop("Series with unequal length!")
   n <- length(x1)

   if(M >= n || ioptw < 1 || ioptw > 8
      || alpha <= 0. || alpha >= 1.)
   stop("Illegal Input to xspe()")

 x <- matrix(c(x1,x2),n,2)
   x.acf <- acf(x,lag.max=n-1,type="cov",plot=F)
 c12 <- x.acf$acf[,1,2]
 c21 <- x.acf$acf[,2,1]
```

```
z1 <- rep(0,n)

if(ioptw == 1) z1[1:(M+1)] <- rep(1, M+1)

else if(ioptw == 2)  z1[1:(M+1)] <- 1 - (c(0:M) / M)

else if(ioptw == 3)  z1[1:(M+1)] <- .54+.46*cos(pi*c(0:M)/M)

else if(ioptw == 4) {
   u  <- c(1:M)/M
   u1 <- u[u<=.5]
   u2 <- u[u>.5]
   z1[1:(M+1)]  <- c(1,1-6*u1^2+6*u1^3,2*(1-u2)^3)
}

else if(ioptw == 5) {
   u   <- c(0:M) / M
   piu <- pi * u
   z1[1:(M+1)]  <- (1 - u) * cos(piu) + sin(piu) / pi
}

else if(ioptw == 6) {
   piu <- pi * (c(1:(n-1)) / M)
   z1[1:n] <- c(1,sin(piu)/piu)
}

else if(ioptw == 7) {
   piu <- pi * (c(1:(n-1)) / M)
   z1[1:n] <- c(1,3*((sin(piu)/piu) - cos(piu)) / (piu*piu))
}

else if(ioptw == 8) {
   u <- c(0:(n-1))/M
   z1[1:n] <- 1 /(1+u^4)
}

z <- fft(z1[1:n]*c12[1:n])+Conj(fft(c(0,z1[2:n]*c21[2:n])))
z <- z[1:((n / 2) + 1)]

ilam2 <- c(2,2./3.,.795,.539,.586,1,1.2,1.66)
fac <- qnorm(1-(alpha/2)) * sqrt(M*ilam2[ioptw]/n)
```

```
  return(list(re=Re(z), im=Im(z), c=fac))
}
```

References

1. Auestad, B. and Tjøstheim, D. (1990). Identification of nonlinear time series: First order characterization and order determination. *Biometrika* **77**, 669–687.

2. Ansley, C. F. (1979). An algorithm for the exact likelihood of a mixed autoregressive-moving average process. *Biometrika* **66** 59–65.

3. Bell, W. and Hillmer, S. (1987). Initializing the Kalman filter in the nonstationary case. Research Report CENSUS/SRC/RR-87/33, Statistical Research Division, Bureau of the Census, Washington, DC, 20233.

4. Bowerman, B. L. and O'Connell, R. T. (1993). *Forecasting and Time Series.* Duxbury, CA.

5. Box, G. E. P. and Jenkins, G. M. (1976). Time Series Analysis: Forecasting and Control. Holden-Day, Oakland, CA.

6. Box, G. E. P. and Pierce, D. A. (1970). Distribution of residual autocorrelations in autoregressive-interated moving average time series models. *JASA* **65** 1509–1526.

7. Brillinger, D. R. (1981). *Time series: Data analysis and theory.* Holden-Day, San Francisco.

8. Brockwell, P. J. and Davis, R. A. (1991). *Time Series: Theory and Methods.* 2nd ed., Springer, New York.

9. Cochran, D. and Orcutt, G. H. (1949). Applications of least squares regression to relationships containing autocorrelated errors. *J. Amer. Stat. Assoc.* **44**, 32–61.

10. Diggle, P. J. (1990). *Time Series: A Biostatistical Introduction.* Oxford University Press, NY.

11. Fuller, W. A. (1976). *Introduction to statistical time series analysis.* Wiley, New York.

12. Gorodetskii, V. V. (1977). On the strong mixing property for linear sequences. *Th. Prob. Appl.* **22**, 411–413.

13. Hannan, E. T. (1970). *Multiple time series.* Wiley, New York.

14. Hannan, E. J. and Heyde, C. C. (1972). On limit theorems for quadratic functions of discrete time series. *Ann. Math. Stat.* **43**, 2058–2066.

15. Heckman, N. E. (1986). Spline smoothing in a partly linear model. *J. R. Statist. Soc.* B **48**, 244–248.

16. Hidalgo, F. J. (1992). Adaptive semiparametric estimation in the presence of autocorrelation of unknown from. *J. Time Series Anal.* **13**, 48–78.

17. Hurvich, C. M. and Tsai, C. L. (1989). *Biometrika* **76** 297–307.

18. Jones, R. H. (1975). Fitting autoregressions. *JASA* **70** 590–592.

19. Jones, R. H. (1980). Maximum likelihood fitting of ARMA models to time series with missing observations. *Technometrics* **22** 389–395.

20. Kohn, R. and Ansley, C. F. (1985). Efficient estimation and prediction in time series regression models. *Biometrika* **72** 694–697.

21. Kohn, R. and Ansley, C. F. (1986). Estimation, prediction, and interpolation for ARIMA models with missing data. *JASA* **81** 751–761.

22. Müller, H. G. and Stadtmüller, U. (1987). Estimation of heteroscedasticity in regression analysis. *Ann. Statist.* **15**, 610–625.

23. Newton, H. J. (1988). *TIMESLAB: A Time Series Analysis Laboratory.* Wadsworth, CA.

24. Powell, J. L., Stock, J. H. and Stoker, T. M. (1989). Semiparametric estimation of index coefficients. *Econometrica* **57**, 1403–1430.

25. Priestley, M. B. (1981). *Spectral analysis and time series.* Academic Press, New York.

26. Priestley, M. B. (1988). *Non-linear and Non-stationary Time series.* Academic Press, New York.

27. Robinson, P. M. (1983). Nonparametric estimators for time series. *J. Time Series Anal.* **4**, 185–207.

28. Robinson, P. M. (1984). Robust nonparametric regression. *Robust and nonlinear time series analysis.* Franke, J., Härdle, W. and Martin, D., (eds.) Springer Lecture Notes, New York.

29. Robinson, P. M. (1988). Root-N-consistent semiparametric regression. *Econometrica* **56**, 931–954.

30. Schuster, A. (1898). On the investigation of hidden periodicities with application to a supposed 26-day period of meteorological phenomena. *Terr. Mag. Atmos. Elect.* **3** 13–41.

31. Schwartz, J. (1993). Air polution and daily mortality in Birmingham, Alabama. *Amer. J. Epidem.* 1136–1147.

32. Shibata, R. (1976). *Biometrika* **63** 117–126.

33. Shibata, R. (1980). *Ann. Statist.* **8** 147–164.

34. Shiryayev, A. N. (1984). *Probability.* Springer, New York.

35. Shumway, R. H. (1988). *Applied statistical time series analysis.* Prentice Hall, New Jersey.

36. Silverman, B. W. (1986). Density Estimation for Statistics and Data Analysis. Chapman and Hall, London.

37. Stone, C. J. (1977). Consistent nonparametric regression. *Ann. Statist.* **5**, 595–620.

38. Stone, C. J. (1980). Optimal rates of convergence for nonparametric estimators. *Ann. Statist.* **8**, 1348–1360.

39. Stone, C. J. (1982). Optimal global rates of convergence for nonparametric regression. *Ann. Statist.* **10**, 1040–1053.

40. Tong, H. (1983). *Threshold Models in Non-linear Time Series Analysis.* Lecture Notes in Statistics **21**. Springer, New York.

41. Tong, H. (1990). *Non-linear Time Series: A Dynamical System Approach.* Oxford University Press.

42. Truong, Y. (1989). Asymptotic properties of kernel estimators based on local medians. *The Annals of Statistics* **17** 606–617.

43. Truong, Y. K. (1991). Nonparametric curve estimation with time series errors. *J. Statist. Planning and Inference* **28**, 167–183.

44. Truong, Y. K. and Stone, C. J. (1992). Nonparametric function estimation involving time series. *Ann. Statist.* **20**, 77–97.

45. Truong, Y. (1992). A nonparametric framework for time series analysis. *New Directions in Time Series Analysis*, ed. by M. Parzen.

46. Truong, Y. (1992). Robust nonparametric time series regression. *J. Mult. Anal.* **41** 163–177.

47. Truong, Y. and Stone, C. J. (1994). Semi-parametric time series regression. *Journal of Time Series Analysis* **15** 405–428.

48. Truong, Y. (1994). Nonparametric time series regression. *Ann. Inst. Statist. Math.* **46** 279–293.

49. Venables, W. N. and Ripley, B. D. (1994). *Modern Applied Statistics with S-Plus.* Springer, NY.

50. Wegman, E. J. (1972). Nonparametric probability density estimation. Technometrics, 14, 533-546.

51. Withers, C. S. (1981). Conditions for Linear Processes to be Strong Mixing. *Z. Wahrsch. verw. Gebiete* **57**, 477–480.