
APPENDIX: AN INTRODUCTION TO S AND S-Plus

This appendix gives the script for a possible introductory session intended for first time users of S-Plus. Imagine that you are trying to learn enough of S to use it in a Data Analysis course based on this book, and let us imagine that the instructor of the course decided to help the students by devoting one of her first lectures to a first introduction to S-Plus. This appendix should be understood as a summary of all the commands and examples used in this hypothetical introductory lecture.

But before we start, a little bit of terminology might be helpful to explain why we often use S and S-Plus interchangeably. The reader needs to keep in mind that S is the programming environment/language developed at AT&T, and S-Plus is the commercial package implementing this language. Consult the online manuals, and especially the *Guide to Statistics* for more information on S and S-Plus.

STARTING S-PLUS (UNDER Windows AND UNDER UNIX)

The instructions given below are restricted to versions 6.0 and higher of S-Plus.

Under Windows

After starting the program, you will be faced with a dialog box asking you to choose the S-Plus Chapter in which you want to work. If you are using the program for the first time, there is no existing chapter, and you will need to browse your hard disk and choose a directory (folder) – or possibly create one – in which you will work on this project. It is recommended that you choose a special chapter which you will use exclusively for the purpose of running the examples and doing the homework problems of this book. The notion of S-Plus chapter replaces the the notion of project or workspace of the previous Windows versions of S-Plus (for example in S-Plus 2000).

The first thing need to do is to go to the menu item **File** ▷ **Chapters** and choose **File** ▷ **Chapters** ▷ **Attach/Create New Chapter** if you want to create a new chapter

(which you will want to do the first time you run S-Plus), or if you want to attach an existing chapter to be able to access its functions and data. You should choose **File** ▷ **Chapters** ▷ **New Working Chapter ...** if you want to switch to an existing chapter different from the default chapter started with S-Plus.

Under Unix

Let us now imagine that the introductory session was intended for Unix or Linux users (we will use the notation \$ to denote the Unix prompt), and let us imagine that the name of the class is ORF405.

The first thing we want to do is to create a subdirectory ORF405 in your home directory. The plan is to store all the files relative to the course in this directory, and in particular, to do our S-Plus based homework in this directory. Next we move to this directory, and run the command CHAPTER

```
$ mkdir ORF405
$ cd ORF405
$ S-Plus CHAPTER
Creating data directory for chapter .
S-PLUS chapter ORF405 initialized.
```

The purpose of this command is to customize the directory for future use. Among other things, it creates a subdirectory .Data, and all the objects created by S-Plus when you run it in the directory ORF405 will be saved (automatically) in this subdirectory .Data and all the objects present in this directory will be accessible each time you run S-Plus from ORF405. You can now start the program S-Plus.

```
$ S-Plus
S-PLUS : Copyright (c) 1988, 2002 Insightful Corp.
S : Copyright Lucent Technologies, Inc.
Version 6.1.2 Release 2 for Linux 2.2.12 : 2002
Working data will be in .Data
>
```

The four lines of text reproduced above are a sign that S-Plus started without problem. The next thing we do is to open a graphic window in which we shall display the graphic objects produced by S-Plus.

```
> trellis.device(motif)
```

At this stage a motif window should appear on the desktop. If this is not the case, especially if you get an error message stating that a display cannot be open, one reason could be that you are running S-Plus remotely, and you did not set up the environment correctly. In such a case, you need to type the following commands needed for the X window system to behave appropriately:

```
$ setenv DISPLAY ''network name of the terminal''
```

on the remote server on which you are running *S-Plus* to make sure that the server knows where to send the X graphic objects and

```
xhost +
```

to make sure that your terminal will accept the X commands to display the graphics.

Important Remark

Most every work with *S* can be described independently of the platform if one limits ourselves to the list of *S*-commands used to perform whatever tasks one is interested in. We will adhere to this practice. This is not much of an issue for *Unix* or *Linux* users who are traditionally used to doing most of the work through typing. But it is clear that doing so, we are missing on most of the *goodies* of the Graphic User Interface, also called GUI (and pronounced "gooohie"). In any case, I remain convinced that the users preferring the mouse to the keyboard will find easily, and take full advantage of, the GUI equivalents offered by *S-Plus*.

Despite the merits of this discourse on the virtues of platform independence, I will presumably err on the side of *Windows* users since after all, most of the participants in this hypothetical introductory class are working with a laptop running a version of the *Windows* operating system.

CREATING S OBJECTS

Typing the command

```
> X <- 1:16
```

in the *Command Window*, or at the *S-Plus* prompt under *Unix* or *Linux*, creates a vector of length 16 containing the first 16 integers in increasing order. Notice the arrow "`< -`" which is typed by using first the "less than" key "`<`" and then the minus sign "`-`". This combination reads "X gets ... to avoid the possible confusion with left arrow key. It is very important in *S-Plus*, since it provides an assignment command: whatever is on the left of this "gets" is an *S* object created by *S-Plus* with the result of the command on the right of the "gets" sign. If an object with this name already exists, it is automatically replaced. In fact, any symbol appearing on the left of the "gets" sign, will be used as the name of an object which *S-Plus* creates and saves in the local `.Data` directory. All these saved objects can be found in the *Object Explorer* under *Windows*.

Note that the same result would be obtained with the command:

```
> X <- seq(from=1,to=16,by=1)
```

To understand what the function `seq` does, we can use the online help by typing:

```
> help(seq)
```

Now that we know what the function `seq` can do for us, we can type:

```
> help
```

and notice that, instead of getting a help file of some kind, we get something which looks more like code. This is a general rule: if one types the name of an S method or function without parentheses, the system returns the S-code of the function in question. Only if we add the parentheses, will we see the command be executed (or an error message returned telling us that we did not included the parameters expected by the function). We can experiment further by typing:

```
> ls()
```

which lists all the S objects contained in the `_Data` subdirectory, and

```
> ls
```

which merely gives the code of the `ls` method. The command:

```
> dim(X) <- c(4,4)
```

reshapes the one dimensional vector `X` into a 4 by 4 matrix. The command:

```
> Y <- X*X
```

creates a new S-object. It is also a 4 by 4 matrix. Its entries are obtained by multiplying `X` by itself *entry by entry*. The symbol `*` does not give the usual matrix product. The latter is obtained by sandwiching the `*` symbol in between `%`'s as in:

```
> Z <- X%%X
```

`Z` is now the 4 by 4 matrix equal to the usual matrix product of `X` by itself. To illustrate further the difference between these products we can do:

```
> dim(X) <- c(8,2)
> Y <- X*X
> Z <- X%%X
```

and even though `Y` is computed correctly as the *entry by entry* product of `X` by itself, the computation of `Z` is not performed and we get an error message. After all, one cannot multiply (in the usual sense of the product of matrices) a 8 by 2 matrix by a 8 by 2 matrix. But notice that the command:

```
> Z <- X%% t(X)
```

does not trigger an error message. This is because the `S` command `t(X)` computes the transpose of the matrix X , and that consequently, the dimensions of the matrices involved in this matrix product match the usual requirements of the product of matrices.

Exiting `S-Plus` can be done at any time with the menu item **File** ▷ **Exit**, or by typing the command:

```
> q()
```

in the command window, or even by closing the `S-Plus` window. There is not need to save your work if you are working under `Unix` or `Linux`. Indeed `S-Plus` automatically saves all the objects you created during the session. You can check that these objects have been saved by listing the objects in your `.Data` directory if you are working under `Unix`. As a consequence of this automatic save, the `Unix` users of `S-Plus` need to clean up house regularly in order to avoid using too much disk space with all the `S` objects saved silently by the program. Things are slightly different in the `Windows` environment. When you quit the program, a dialog box prompts you to decide which, among the objects created during the session, do you want to save to the disk.

RANDOM GENERATION AND WHITE NOISE

The commands

```
> WN <- rnorm(1024)
> tsplot(WN)
```

create a vector `WN` of length 1024, and produce a sequential plot of the entries of this vector. The entries of `WN` are realizations of independent normal random variables with the same distribution $N(0, 1)$. We specified the length of the random vector but we did not specify the values of the parameters of the distribution. `S-Plus` uses the default values 0 and 1 for the mean and the standard deviation of the normal distribution. The function `tsplot` gives a sequential plot of the vector `WN`, by plotting its values against the variable which takes the values 1, 2, ..., 1024.

NB: A command `WN <- rnorm(1024, 1.2, 4.0)` would have created a sample of the same size 1024 of realizations of i.i.d. variates from the normal distribution with mean 1.2 and standard deviation 4, in other words the distribution $N(1.2, 16)$ if we use the standard notation used in the text. The commands:

```
> par(mfrow=c(2, 1))
> tsplot(WN)
> tsplot(WN[1:64])
> par(mfrow=c(2, 1))
```

divide the graphics window into two subplot areas, one on the top of the other, give a time series plot of the full WN vector on top, a plot of the time series of the first 64 entries of the vector WN , and reset the graphic window to a single plotting area. The results are shown in Figure 7.21. The top plot shows what we should expect from a

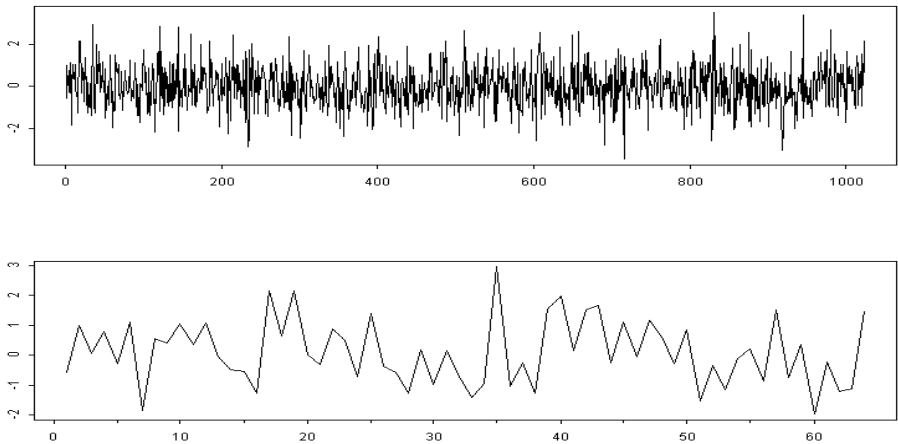


Fig. 7.21. Time series plot of the full white noise series WN (top) and of the series of its first 64 entries (bottom).

white noise: very erratic because of the lack of correlation of the successive entries of the series. Identifying a white noise is of crucial importance to a statistician. Indeed, most model fitting efforts are devoted to the identification and the extraction from the data of organized structures. This is usually done up until the remainder terms (which we call the residuals) form a white noise. From that point on, it is unreasonable to try to go any further, and statistical inference should take place at that time, and definitely before we start trying to fit the noise by mistake.

Strangely enough, the bottom plot of Figure 7.21 looks smoother. The reason is simple: it is viewed at a different scale. We plotted one sixteenth of the data on a plot with the same width! The *white noise look* which was identified earlier has practically disappeared. It is important to realize that, looking at the same data at different scales will leave different visual impressions. Don't be fooled by this artifact of the graphic settings used.

In the above commands, we did subscript the vector WN to consider only the set of its first 64 entries. To do so, we specified the range of the indices we wanted to keep. Subscripting is also conveniently used to extract sub-matrices of matrices. For example, the command `X[1:3, 1:3]` will produce the 3 by 3 matrix in the left most

corner in the top of X (recall that X was defined earlier) while $X[, 2]$ will produce a vector equal to the second column of X .

MORE FUNCTIONS AND for LOOPS

The function `diff` should be viewed as a discrete analog of the operation of differentiation for functions of continuous variables. We learned in calculus that the inverse operation is integration. The discrete analog is given by the function `cumsum`. If we apply to the white noise vector `WN`:

```
> WN <- rnorm(1024)
> RW <- cumsum(WN)
```

we get a numeric vector `RW` which represents a sample of length 1024 from a random walk.. The sequential plot of `RW` (i.e. the plot of the values of `RW` against the successive integers) is given in the left pane of Figure 7.22. Let us now try to create an object according to the Samuelson's model for stocks. According to this model, the time evolution of a stock is represented by the exponential of a random walk (with a given volatility) with drift given by the rate of appreciation of the stock. More precisely, we define the objects `SIG`, `MU`, `TIME` and `STOCK` by:

```
> DELTAT <- 1/252
> SIG <- .2*sqrt(DELTAT)
> MU <- .15*DELTAT
> TIME <- (1:1024)/252
```

for the volatility, the rate of return, the time (expressed in years) and an initial zero vector for the stock. With these objects at hand, one should be able to fill in the entries of the vector `STOCK` with the commands:

```
> for (I in 1:1024) STOCK[I] <- exp(SIG*RW[I]+MU*TIME[I])
```

This command is typical of the use of the `for` loops in `S-Plus`. Unfortunately, in `S-Plus` like in all other interpreted languages, computations with loops are very slow, and they should be avoided whenever possible. There is a very simple way to avoid the above loop. Indeed, most numerical functions in `S-Plus` when applied to a vector (resp. matrix, array, ...) will create a vector (resp. matrix, array, ...) with entries given by the computations of the function on the entries of the vector (resp. matrix, array, ...) passed as argument to the function. So in the above example, the command:

```
> STOCK <- exp(SIG*RW+MU*TIME)
```

will give the same result, and the computations will be much faster. The plot of the entries of this vector is given in the right pane of Figure 7.22. The latter was produced with the commands:

```

> par(mfrow=c(2,1))
> tsplot(RW)
> title("Sample of size 1024 from a random walk")
> plot(TIME,STOCK,type="l")
> title("Corresponding geometric random walk with drift")
> par(mfrow=c(1,1))

```

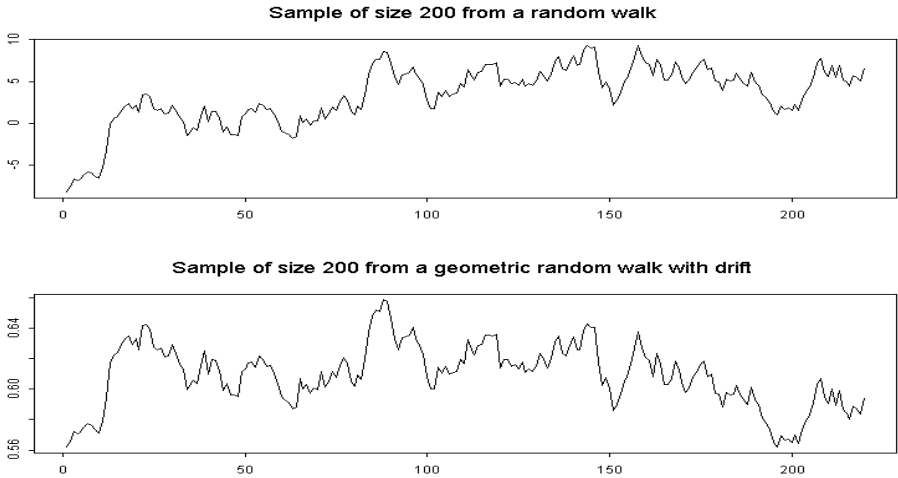


Fig. 7.22. Sequential plot of random walk sample RW (left) and of the corresponding geometric random walk with drift (right).

Notice that we use the command `plot` instead of `tsplot`. We need to pass two vectors to the command `plot`. The first one gives the values of the first coordinates of the points to plot while the second vector gives the second coordinates of these points. We use the option `type="l"` to joint the points by straight line segments to get the visual impression of a continuous curve instead of isolated points. The plots look very similar, until we notice the difference in scale on the vertical axes.

IMPORTING DATA

In most data analysis applications, the data are not created within the statistical package used for the analysis: they have to be imported. We give several examples as illustrations. These examples can be used as templates for further applications and to work out the problems (homework assignments) given in the text.

Using Data Already in S-Plus

In some instances in the text, we use data sets contained in the S-Plus distribution package. Let us consider for example, the data of the record times of the Scottish races. The data set is in the form of a data frame called `hills`, and it is part of the library `mass` which we need to link to our S-Plus session. See below for the way to link the library. We can choose to manipulate the variables by using their names `hills$dist`, `hills$climb` and `hills$time`, or we can decide to avoid having to type over and over the data frame name `hills`, in which case we need to "attach" the data frame to the S-Plus session. This is done in the following way.

```
> library(mass)
> attach(hills)
```

Doing so makes it possible to use all the variables in this frame by simply referring to their names. For example, the following commands produce the results shown in Figure 7.23.

```
> plot(climb,time)
> plot(dist,time)
```

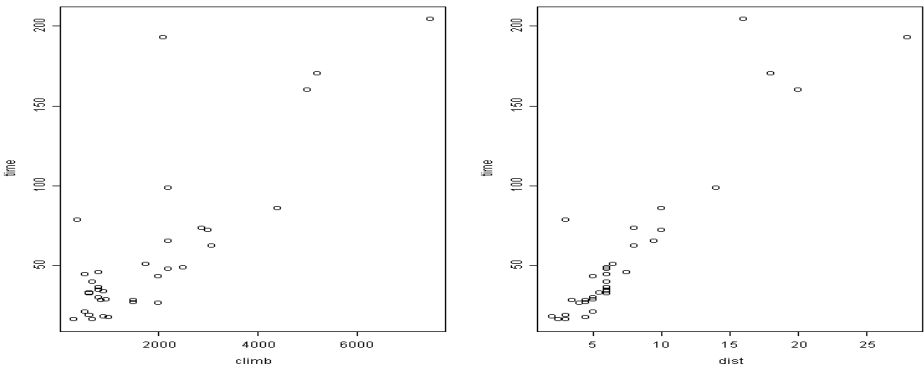


Fig. 7.23. Scatterplot of the record time versus `climb` (left) from the `hills` data frame, and of the time versus the distance variable `dist` (right).

Importing Text Files in S-Plus

As an example, we show how to import the text (ASCII) file `HOWAREYOU` which is part of the directory `C:\My Documents\ORF405\book\data` containing all

the data sets used in the book. We can open the file with a text editor to check its contents. We show the first seven rows of the file for the sake of illustration.

```
0.0359678408504965
0.0635196843758394
0.0750752080020106
0.0628095887777292
0.031177878417794
-0.00520235015344319
-0.0294474884975495
```

Under Windows, we import this file into S-Plus with the dialog created by the **File** ▷ **Import Data** ▷ **From File**, and clicking on the button BROWSE to navigate to the location where the file is, and selecting it. The full pathname of the file HOWAREYOU appears, its being an ASCII file also shows, and OK'ing all that allows S-Plus to create an S object called HOWAREYOU in the current data base. The same result could be obtained with the command:

```
> HOWAREYOU <- scan("C:\\My Documents\\ORF405\\book
                    \\data\\HOWAREYOU")
```

which could also be used under Unix if the double backslashes \\ were to be replaced by single ones \. As the result of the next command shows, it is a vector of length 5151 and its first 8 elements are given by:

```
> length(HOWAREYOU)
[1] 5151
> HOWAREYOU[1:10]
[1] 0.035968 0.063520 0.075075 0.062810
[5] 0.031178 -0.005202 -0.029447 -0.031487 -0.013135
```

We use this vector in Chapter 5 to illustrate the construction of signal series objects in S-Plus

Importing S-Plus Dumps

The previous example was concerned with plain text files. Next, we consider the case of text files produced by S-Plus, and containing information about S-objects they were issued from. It is very easy to port files from one version of S-Plus to another. For example, files created and manipulated under UNIX or LINUX can be easily read and manipulated under Windows. S objects can be dumped to a text file with the dump command (see Help file for details), and the format of these text files is such that they can be read by any other version of S-Plus irrespective of the platform in question. A text file resulting from dumping S objects should be open under Windows from the dialog created by **File** ▷ **Open**, or by double-clicking the open folder icon in the task bar. S-Plus creates a script window, and displays the contents of the text file in this window. Hitting the **F10** key, or running the script

using the menu item **Script** ▷ **Run F10**, is what is needed to create the desired object. These dumps are text files and as such, they can be opened and edited with a text editor. We shall use the extension `.asc` to distinguish them from regular text files for which we try to use the extension `.txt`. Most of the data sets provided for illustration purposes or because they are needed for the problem sets, will be in the form of script files with the extension `.asc`.

Getting Data from the Web

We now show how to import data from the web into *S-Plus*. In the process, we show how starting from scratch, we can perform the analysis of the daily S&P 500 log-returns which we presented in Chapter 1.

We first retrieve the data from the internet. Let us start our favorite web browser and go to

`http://www.yahoo.com.`

Once there, we double click on the Finance special icon/button at the top of the page, or on the link Finance from the list of links right underneath the search box in the middle of the page. Once on the Finance page, we double click on the link US Indices under the search box. Next we double click on the Chart link of the Standard and Poor's 500 Index section. Finally, we double click on the link Daily Historical Quotes at the bottom of the page. We choose the start date January 1, 1960 and end date June 6, 2003 (now you know when this part of the book was prepared !) and we download the resulting table in spreadsheet format. We choose a location where to save the file. For the sake of definiteness, I call `filename` the full pathname of the file.

Warning. Web sites change frequently. Yahoo is no exception, so some of the detailed instructions given above may be obsolete by the time the book appear in print, or the reader tries to follow them: improvisation may be the only way out then !

The following instructions are for Windows only. See for example the next subsection below for instructions appropriate for the Unix/Linux platforms.

Next we switch to *S-Plus* and we open the dialog box under **File** ▷ **Import Data** ▷ **From File** and use the button **Browse** to get the pathname of the file to appear in the dialog box. Once ASCII file - comma delimited (`csv`) and `DSP500` appear in the File format and Data set spaces we click OK, and *voilà* ! the data frame `SP500` appears on the desktop. A sequential plot of the daily close prices can be obtained with the command:

```
> tsplit(DSP500$Close)
```

Note that the dollar sign is used to extract a column from a data frame using the name of the column. The graphic window appears on the desktop and shows the graph reproduced in the left pane of Figure 7.24. We shall revisit this procedure when we study time series and we introduce the `timeSeries` objects used by *S-Plus* for

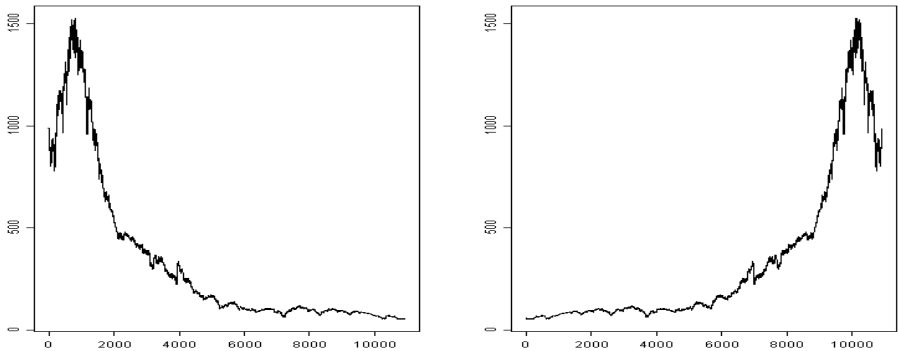


Fig. 7.24. Daily S&P 500 closing prices as imported from the web (left) and same data after we revert to the natural order (right).

their analyses. Obviously, something is wrong. The data were read in the reverse order of what would be natural. So we revert to the natural order using the function `rev` provided by `S-Plus`. We create a new `S` object of the numerical vector class:

```
> DSP <- rev(SP500$Close)
> tsplot(DSP)
```

and we compute and plot the daily log returns with the command: The result is reproduced in the right pane of Figure 7.24. Things look right now.

```
> DSPLR <- diff(log(DSP))
> tsplot(DSPLR)
> title("S&P 500 daily log-returns")
```

The logarithm of the daily close is computed with the function `log`, and the difference of the resulting vector is computed with the function `diff`. Check the help file for details on the function `diff`. Notice that we ignore the weekends (i.e. Monday's close follows the preceding Friday's close) and holidays, and more generally the days the market was close like the week of September 12, 2001 following the terrorist attack on the World Trade Center in New York.

Using the functions `mean` and `var` we compute the mean and the standard deviation of the daily log return. We use the function `sqrt` to compute the square root of the variance.

```
> mean(DSPLR)
[1] 0.0002717871
> sqrt(var(DSPLR))
[1] 0.009175474
```

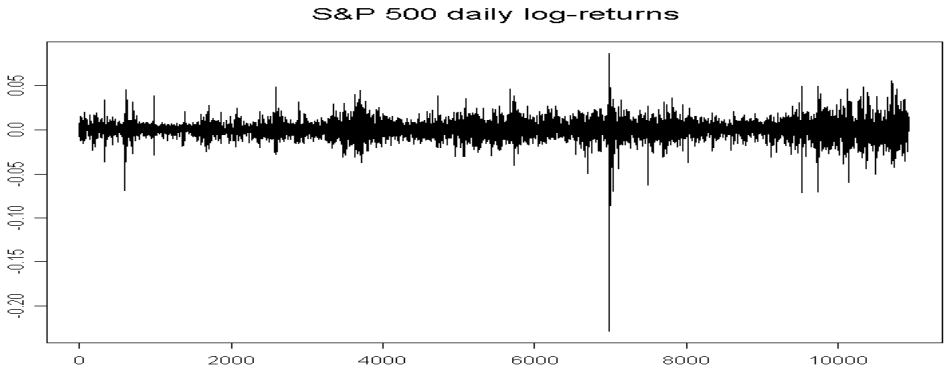


Fig. 7.25. Sequential plot of the daily S&P 500 log-returns.

Looking at the sequential plot of the daily log return (as reproduced in Figure 7.25) we notice we notice a few very large negative values. Looking at the largest of these down moves we see that:

```
> min(DSPLR)
[1] -0.2289972
> (min(DSPLR) - mean(DSPLR)) / sqrt(var(DSPLR))
[1] -24.98716
```

which shows that this down move was essentially 25 standard deviations away from the mean daily move. So much for the normal distribution as a model for the daily moves of this index. This single day of October 1987, cannot be accounted for by a normal model for the daily log returns !

Still Another Example

Let us illustrate one more of the typical features of data sets which we have to face before we can actually begin the statistical analysis. We show how to handle the presence of headers. The text file `BondPrices61300.txt` contains information downloaded from Data Stream about the prices of the treasuries traded on June 6, 2000. Opening this file with a text editor shows the existence of a 12 lines header which we would like to ignore when importing the data into S-Plus. As illustrated before, the file can be imported using the menu or a command which can also be used under Unix. One can use **File** \triangleright **Import Data** \triangleright **from File** from the S-Plus menu if we are working under Windows, then browse the disk to locate the file as before, but this time, we should state after clicking the Options tab, that we want to start on row 13 only. Equivalently, one can use the command:

```
> BP61300 <- read.table('BondPrices61300.txt', skip=12,
```

```
col.names=c('NAME', 'YEAR', 'EXPIRY', 'PRICE',
            'YIELD', RED.YIELD 'ACCR.INT.', 'LIFE', 'VARIATION')
```

which can also be used if we are working under Unix. This command creates a data frame, whose first five rows can be printed in the command window. We only reproduce the first six columns.

```
> BP61300[1:5,]
      NAME YEAR EXPIRY PRICE YIELD RED.YIELD
1 US.TRYSY.BONDS 1997  2027  99.50  6.16  6.161
2 US.TRYSY.BONDS 1975  2005 100.03  8.25  8.241
3 US.TRYSY.BONDS 1977  2007 101.56  7.51  7.317
4 US.TRYSY.BONDS 1977  2007 102.78  7.66  7.375
5 US.TRYSY.BONDS 1978  2008 106.43  8.22  7.691
```

Notice that S-Plus labels the rows by the successive integers (in increasing order) by default, since we did not provide the `read.table` command with a value for the parameter `row.names`. The `skip=12` part of the command line forced S-Plus to ignore the first 12 lines of the ASCII file `w6.txt`. We can get the same result under Windows by setting appropriately the relevant options appearing in the **Options** tab of the dialog box. In particular, one would have to decide which row should the reading start from, one should have to give the name of the column variables, If we want to plot the yield against the remaining life of these fixed income instruments, we can use the command:

```
> plot(BondPrices61300$LIFE, BondPrices61300$YIELD, type="l")
```

The result is shown in Figure 7.26. The extension `$LIFE` added to the name of the data.frame `BondPrices61300` identifies the variable `LIFE` from this data frame. Similarly `BondPrices61300$YIELD` identify the `YIELD` variable. We used the option `type="l"` to join the points of the scatterplot by line segments, producing in this way a continuous curve, akin the graph of the yield against the time remaining to maturity. If we want to work for a while with these data without having to type the reference `BondPrices61300` to the data frame over and over, we can attach the data frame with the command:

```
> attach(BondPrices61300)
```

and work with the variables in the columns of the data frame by referring directly to their names. For example the plot of Figure 7.26 can be equivalently produced by the command:

```
> plot(LIFE, YIELD)
```

NB: It is important to keep in mind the fact that, contrary to most Windows programs, S-Plus is case sensitive. This feature is presumably inherited from the Unix version of S-Plus. Consequently, `BondPrices61300` is not the same thing as `bondprices61300` !

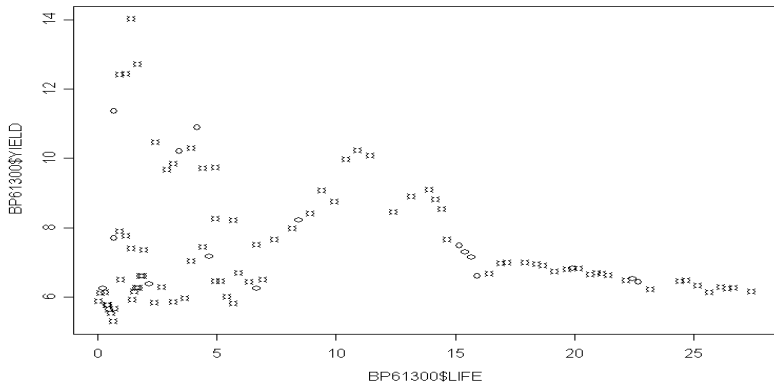


Fig. 7.26. Plot of the yield against the remaining life of the fixed income instruments contained in the text file `BondPrices61300`.

PROGRAMMING IN S-PLUS: A FIRST FUNCTION

When we suspect that a set of S-Plus commands will be needed frequently, it is advantageous to encapsulate the individual commands in a script which we can save as a script, and which we will be able to re-use at our convenience. We can re-run individual commands or entire sets of commands by selecting them (highlighting them) and hitting the key **F10** or using the **Script** ▷ **Run F10** menu.

We can also structure the set of commands to be able to take parameters as arguments and save it as an S object which will be called a function whenever it acts on an argument. We did just that to create functions `qexp` and `myqqnorm` which we used earlier in the first chapter. We now give the details of an example which we borrow from the S-Plus manual. , we define function `eda.shape` defined by typing the following in the command window.

```
> eda.shape <- function(x)
+ {
+   par(mfrow = c(2,2))
+   hist(x)
+   boxplot(x)
+   iqd <- summary(x)[5] -summary(x)[2]
+   plot(density(x,width=2*iqd),xlab="x",ylab="",type="l")
+   qqnorm(x)
+   qqline(x)
+   par(mfrow=c(1,1))
+ }
```

The argument of this function needs to be a numeric vector x containing the numeric data which we want to visualize. Except for the boxplots which are hardly used in the text, all of these commands are used extensively throughout the book. We illustrate the use of this function with a sample of size 1024 which we generate from the standard normal distribution.

```
> eda.shape(rnorm(1024))
```

The result is reproduced in Figure 7.27. As expected, the histogram tries to reproduce

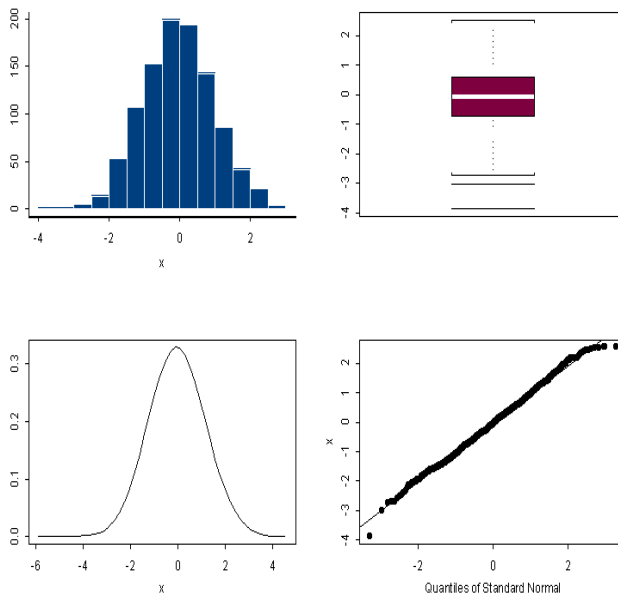


Fig. 7.27. Exploratory data analysis of a sample of size 1024 from the standard normal distribution.

the bell shape normal density, the kernel density estimate gives a faithful representation of the normal density function, and the points of the Q-Q plot are tightly packed around the diagonal. This shows that the random number generator used to create this sample did a reasonable job. At this stage, it is important to emphasize that the function `eda.shape` is useful for samples whose distributions are nearly normal. This is especially true for the histogram and density estimation parts of the exploratory data analysis. We will use it to check graphically, if the distribution of a sample is normal or does not depart too much from a normal distribution. Indeed, it can give strange results for distributions with drastically different features.

NOTES & COMPLEMENTS

When I first started to use *S-Plus* in teaching, I used to recommend the small book of Phil Spector to newcomers to *S* and *S-Plus*, especially if they were to work under *Unix*. More recently, I changed my recommendation to the thicker book by Krause and Olson [55]. It is best for an introduction to the language *S*, and to the functionalities (especially of the *Windows* version) of *S-Plus*. Advanced users can consult the book entitled *S Programming* by Venables and Ripley, however the *bible* remains the green book *Programming with Data* by John Chambers.[23]

References

1. Y. Ait Sahalia and A.W. Lo: Nonparametric Estimation of State Price Densities Implicit in Financial Asset Prices. *The Journal of Finance* **53** (1998) 499-547.
2. S. Amari: *Differential Geometrical Methods in Statistics*. Lect. Notes in Statistics # **28** (1985) Springer Verlag, New York, NY.
3. N. Anderson, F. Breedon, M. Deacon, A. Derry, and G. Murphy: *Estimating and Interpreting the Yield Curve*. (1996) Wiley, Chichester.
4. A. Antoniadis, J. Berruyer and R. Carmona: *Régression Non-linéaire et Applications*. (1992) Economica.
5. S.P. d'Arcy and V.G. France: Catastrophe Futures: A Better Hedge for Insurers. *J. of Risk and Insurance*, **59** (1992)
6. L. Bachelier: *Théorie de la spéculation*. Thesis, (1900) Paris.
7. Bank for International Settlements: Zero-coupon Yield Curves. Technical Report March (1999).
8. E. Banks ed.: *Weather Risk Management*. (2002) Palgrave, New York, NY.
9. T. Bollerslev: Generalized Auto Regressive Heteroskedasticity. *J. of Econometrics* **31** (1986) 307-327.
10. T. Bollerslev, R.F Engle and J.M. Wooldridge: ARCH Models. In *Handbook of Econometrics*, **IV** (1994) 2959-3038.
11. G.E.P. Box and G.M. Jenkins: *Time Series Analysis: Forecasting and Control*. Revised Edition, (1976) Holden Day, San Francisco.
12. L. Breiman, J.H. Friedman, R.A. Olshen and C.I. Stone: *Classification and Regression Trees*. Wadsworth and Brooks/Cole, (1984) Monterey, CA.
13. P.J. Brockwell and R.A. Davis (1996): *Introduction to Time Series and Forecasting*. Springer Texts in Statistics, Springer Verlag, New York, NY.
14. R.L. Brown, J. Durbin and J.M. Evans: Techniques for testing the constancy of regression relationship over time (with comments). *J. Royal Statistical Society, ser. B* **37** (1975) 149-192.
15. A. Bruce and H.Y. Gao: *Applied Wavelet Analysis with S-plus*. Springer Verlag (1996) New York, N.Y.
16. J.Y. Campbell, A.W. Lo, and A.C. MacKinlay: *The Econometrics of Financial Markets*. Princeton University Press (1997) Princeton, NJ.
17. R. Carmona and J. Morrisson (2000): EVANESCE, an S-Plus Library for Heavy Tail Distributions and Copulas. *Dept. of Operations Research & Financial Engineering, Princeton University*. <http://www.princeton.edu/rcarmona>.

18. R. Carmona, C. Chen, R.D. Frostig and S. Zhong: Brain Function Imaging: a Comparative Study. *Tech. Rep.* (1996) Princeton Univ.
19. R. Carmona, W. Hwang and R.D. Frostig: Wavelet Analysis for Brain Function Imaging. *IEEE Trans. on Medical Imaging* **14** (1995) 556-564.
20. R. Carmona, W. Hwang and B. Torresanni: *Time Frequency Analysis: Continuous Wavelet and Gabor Transform, with an implementation in Splus* (1998) Academic Press, New York, N.Y.
21. R. Carmona: *Interest Rates Mathematical Models: from Parametric Statistics to Infinite Dimensional Stochastic Analysis*. SIAM Monographs (2003) (to appear) SIAM, Philadelphia, PA
22. R. Carmona: *Stochastic Analysis for Financial Engineering*. Lecture Notes, Princeton University (2001) Princeton NJ.
23. J.M. Chambers: *Programming with Data: A Guide to the S Language*. MathSoft (1998) Seattle WA
24. N.H. Chan: *Time Series: Applications to Finance*. John Wiley & Sons, (2002) New York, N.Y.
25. Chicago Board of Trade: *A User Guide: PCS Catastrophe Insurance Options*. (1995)
26. W.S. Cleveland: *The Elements of Graphing Data*. Hobart Press, (1994) Summit, N.J.
27. L. Clewlow and C. Strickland: *Energy Derivatives, Pricing and Risk management*. Lacima Publications (2000) London.
28. C. de Boor: *A Practical Guide to Splines*. Springer Verlag, (1978) New York, NY.
29. D. Drouot Mari and S. Kotz: *Correlation and Dependence*. Imperial College Press (2001) London
30. B. Dupire: Pricing with a Smile. *Risk Magazine*, **7** (1994).
31. P. Embrechts, C. Kluppelberg and T. Mikosch: *Modelling Extremal Events for Insurance and Finance*. Springer Verlag (1997) New York, N.Y.
32. R.F. Engle: Autoregressive conditional heteroskedasticity with estimates of the variance of the United Kingdom inflation. *Econometrica*, **50** (1982) 987-1006.
33. R.F. Engle and T. Bollerslev: Modeling the Persistence of Conditional Variances. *Econometric Reviews*, **5** (1986) 1-50.
34. R.F. Engle and C. Granger: Cointegration and Error-Correction: Representation, Estimation and Testing. *Econometrica*, **55** (1987) 251-276.
35. E.F. Fama and R.R. Biss: The Information in Long-Maturity Forward Rates. *American Economic Rev.* **77** (1987) 680-692.
36. J. Fan and Q. Yao: *Nonlinear Time Series: Nonparametric and Parametric Methods*. Springer Verlag (2003) New York, N.Y.
37. H. Föllmer and A. Schied: *Stochastic Finance: An Introduction in Discrete Time*. (2002) De Gruyter Studies in Mathematics. Berlin.
38. J.P Fouque, G. Papanicolaou and R. Sircar: *Derivatives in Financial Markets with Stochastic Volatility*. (2000) Cambridge University Press, London.
39. R. Gençay, F. Selçuk and B. Whitcher: *An Introduction to Wavelets and Other Filtering Methods in Finance and Economics*. Academic Press (2002) New York, NY.
40. C. Gouriéroux: *ARCH Models and Financial Applications*. Springer Verlag (1997) New York, N.Y.
41. C. Gouriéroux and J. Jasiak: *Financial Econometrics*. Princeton University Press (2002) Princeton, N.J.
42. W. Härdle: *Non-parametric Regression*. Springer Verlag (1994) New York, N.Y.
43. J.D. Hamilton: *Time Series Analysis*. Princeton University Press (1994) Princeton, NJ.

44. Hastie, Tibshirani, J. Friedman: *The Elements of Statistical Learning, Data mining, Inference and Prediction*. Springer Verlag (2001) New York, N.Y.
45. Hausdorff: Dimension und äusseres Mass. *Math. Annalen* **79** (1919) 157-179.
46. P.J. Huber: *Robust Statistics*. J. Wiley & Sons (1981) New York, NY.
47. J.C. Hull: *Options, Futures, and Other Derivatives*. Prentice Hall, 5th edition (2002) New York, NY
48. J.M. Hutchinson, A.W. Lo and T. Poggio: A Nonparametric Approach to the Pricing and Hedging of Derivative Securities via Learning Networks. *Journal of Finance*, **49** (1994)
49. S. Johansen: *Likelihood-Based Inference in Cointegrated Vector Autoregressive Models*. Oxford University Press (1995) Oxford
50. P. Jorion: *Value at Risk: The New Benchmark for Managing Financial Risk*. 2nd ed. McGraw Hill, (2000) New York, NY
51. S. Karlin and H.W. Taylor: *A First Course in Stochastic Processes*. 2nd ed. Academic Press (1975) New York, N.Y.
52. C.J. Kim and C.R. Nelson: *State-Space Models with Regime Switching*. MIT Press, (1999) Cambridge MA.
53. G. Kitagawa: Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models. *Journal of Computational and Graphical Statistics*, **5**, (1996) 1-25.
54. T. Kohonen: *Self-organizing Maps*. Springer Verlag (1995) New York, NY
55. A. Krause and M. Olson: *The Basics of S and S-PLUS*. 2nd Edition. Springer Verlag (2000) New York, N.Y.
56. D. Lambertson and B. Lapeyre: *Introduction to Stochastic Calculus Applied to Finance*. (1996) CRC Press.
57. R. Litterman and J. Scheinkman: Common factors affecting bond returns. *J. of Fixed Income*, **1**, 49-53.
58. B.B. Mandelbrot: New Methods in Statistical Economics. *J. of Political Economy*, **71** (1963) 421-440.
59. B.B. Mandelbrot: The Variation of Certain Speculative Prices. *J. of Business*, **36** (1963) 394-419.
60. B.B. Mandelbrot: *Fractal, Form Dimension and Chance*. W.H. Freeman & Co. (1977) San Francisco, CA.
61. K.V. Mardia, J.T. Kent and J.M. Bibby: *Multivariate Analysis*. Academic Press, (1979) New York, NY
62. T.C. Mills: *The Econometric Modelling of Financial Time Series*. 2nd Ed. Cambridge University Press, (1999) London
63. K.R. Miltersen and E.S. Schwartz: Pricing of Options on Commodity Futures with Stochastic Term Structure of Convenience Yields and Interest Rates. *J. of Financial and Quantitative Analysis*, (2000)
64. D.C. Montgomery and E.A. Peck: *Introduction to Linear Regression Analysis*. 2nd ed. John Wiley & Sons (1992) New York, NY
65. R.B. Nelsen: *An Introduction to Copulas*. Lect. Notes in Statistics **139** (1999) Springer Verlag, New York, NY.
66. C.R. Nelson and A.F. Siegel: Parsimonious Modeling of Yield Curves. *J. of Business* **60** (1987) 473-489.
67. D. Pilipovic: *Energy Risk: Valuing and Managing Energy Derivatives*. Mc Graw Hill (1997) New York, NY
68. S.R. Pliska: *Introduction to Mathematical Finance. Discrete Time Models*. Blackwell (1997) Oxford, UK.

69. M.B. Priestley: *Nonlinear and Non-stationary Time Series Analysis*. Academic Press, (1988) New York, NY.
70. R. Rebonato: *Interest-Rate Option Models: Understanding, Analyzing and Using Models for Exotic Interest-Rate Options*. Wiley (1996) New York, NY
71. P.A. Ruud: *An Introduction to Classical Econometric Theory*. Oxford University Press, (2000) New York, NY.
72. E. Renault and N. Touzi (1996): Option Hedging and Implied Volatility in a Stochastic Volatility Model. *Mathematical Finance* **6** 215-236.
73. J.Rice: *Mathematical Statistics and Data Analysis*. 2nd ed. Duxbury Press (1995) Belmont CA
74. R. Roll: A Critique of the Asset Pricing Theory's Test, Part One: Past and Potential Testability of the Theory. *Journal of Financial Economics* **4** (1977) 129-176.
75. M. Rosenblatt: *Gaussian and Non-Gaussian Linear Time Series and Random Fields*. Springer Verlag (2000), New York N.Y.
76. P.J. Rousseeuw and A.M. Leroy: *Robust Regression and Outlier Detection*. J. Wiley & Sons (1984) New York, NY.
77. L.O. Scott: Option pricing when the variance changes randomly: Theory, estimation and application. *Journal of Financial and Quantitative Analysis* **22** (1987) 419-438.
78. R.H. Shumway and D.S. Stoffer: *Time Series Analysis and Its Applications*. Springer Verlag (2000) New York, NY.
79. B.W. Silverman: *Density Estimation for Statistics and Data Analysis*. Chapman & Hall (1986) London.
80. P. Spector: *An Introduction to S and S-Plus*. Duxbury Press (1994) Belmont, CA.
81. J.M. Steele: *Stochastic Calculus and Financial Applications*. Springer Verlag (2000) New York NY
82. L.E.O. Svensson: Estimating and Interpreting Forward Interest Rates: Sweden 1992-94. *NBER working Paper Series # 4871* (1994)
83. R.S. Tsay: *Analysis of Financial Time Series*. John Wiley & Sons, (2002) New York, N.Y.
84. O. Vasicek and G. Fong: Term structure estimation using exponential splines. *J. of Finance*, **38** (1982) 339-348.
85. W.N. Venables and B.D. Ripley: *Modern Applied Statistics with S-PLUS*. 2nd Edition. Springer Verlag (1997) New York, N.Y.
86. H. von Storch and F.W. Zwiers: *Statistical Analysis in Climate Research*. Cambridge University Press (1999) New York, NY.
87. M.V. Wickerhauser: *Adapted Wavelet Analysis: from Theory to Software*. A.K.Peters LTd (1994) Wellesley, MA
88. I.H. Witten and E. Frank: *Data Mining*. Academic Press (2000) San Diego CA.
89. E. Zivot and J. Wang: *Modeling Financial Time Series with S-PLUS*. Springer Verlag (2003) New York, N.Y.

Notation Index

- $(x - K)^+$, 209
 $AvgT_t$, 290
 B , 275, 312, 356
 $C(m, \lambda)$, 12
 CDD_t , 290
 CO_2 , 261
 $C_{T,K}(t, S)$, 208
 $E(\lambda)$, 11
 ES_q , 27
 $E_n(\mathbf{Z})$, 326
 F_X , 7
 $F_\lambda(x)$, 11
 $F_{a,b}(x)$, 8
 HDD_t , 290
 $I(1)$, 260
 $I(p)$, 260
 I_n , 327
 Max_t , 290
 Min_t , 290
 $N(x)$, 181
 $N(x_j)$, 183
 RC_t , 26
 R^2 , 131
 $U(0, 1)$, 8
 $U(a, b)$, 8
 Var_q , 27
 $W(u)$, 181
 $WN(0, \sigma^2)$, 267
 $X \sim AR(p)$, 268
 $X \sim ARMA(p, q)$, 281
 $X \sim MA(q)$, 272
 Δt , 240, 381
 $\hat{\Phi}$, 209
 $\hat{\Phi}_{\mu, \sigma^2}$, 9
 Σ_0 , 334
 \mathbf{H} , 135
 $\mathcal{L}(\varphi)$, 179
 $\mathcal{L}_{JUS}(\varphi)$, 189
 $\mathcal{L}(\varphi)$, 115
 γ_X , 253
 $\hat{F}_n(x)$, 28
 $\hat{\epsilon}_i^*$, 138
 $\hat{\epsilon}'_i$, 137
 μ_0 , 334
 $\mu_X(t)$, 253
 ∇ , 116, 260, 312
 $\rho_K(X, Y)$, 69
 ρ_P , 53
 $\rho_S(X, Y)$, 69
 ρ_X , 253
 $\sigma_X(t)$, 253
 $\tau(X, Y)$, 69
 $\text{var}_X(t)$, 253
 $\varphi^{(m)}(x)$, 179
 $\hat{\mathbf{X}}_{n+1}$, 327
 b , 183
 $d(x)$, 181
 d_1 , 209
 d_2 , 209
 $f_x(X)$, 8
 $f_\lambda(x)$, 11
 $f_{\mu, \sigma^2}(x)$, 8
 $f_{a,b}(x)$, 8
 h_t , 378
 $h_{i,i}$, 136
 w_i , 189

- $w_x(x_i)$, 181
 acf, 270, 360
 AEP, 144
 AIC, 150
 AR(p), 268
 ARCH, 362, 410
 ARIMA, 282
 ARIMA(p,d,q), 282
 ARMA(p,q), 281

 BIS, 88, 160, 173

 CAPM, 142, 332, 337
 CDD, 290
 cdf, 7, 8
 CIR, 403
 CME, 247, 290, 291

 EM, 410

 GARCH, 410
 GARCH(p,q), 363
 GDP, 12, 32
 GMM, 410
 GPD, 36, 38, 66

 HDD, 290
 HMM, 393

 i.i.d., 13, 28, 264
 ISO, 4

 L1, 111, 113
 L2, 111, 113
 LAD, 111, 113
 LIBOR, 90
 LS, 111, 113

 MA(q), 272
 MCMC, 410
 MLE, 358

 NLC, 85
 NYMEX, 248
 NYSE, 241, 248

 OTC, 290, 291

 P&L, 26
 PCA, 84, 85
 PCS, 4, 291
 POT, 37

 SDE, 380
 SSE, 131, 137
 SUR, 142
 SV, 375

 TSS, 132
 TXU, 144

 VaR, 25, 26, 82

Data Set Index

AFT.mat, 193
BLIND, 233
DSP, 44
dsp.asc, 44
DSP500, 423
DSPLR, 20, 35
enron.tab, 405
FRWRD, 146, 177, 178, 180, 182, 185
GEYSER, 225
hills, 421
HOWAREYOU, 240, 421
IBM, 372
ibmqeps, 351
intrasp.asc, 44
longactive.txt, 307
longnonactive.txt, 307
lugs90, 226
MIND, 228
MORN.mat, 193
MSP, 44
nb, 226
nbticks, 193
PCS, 4
pepsiqeeps, 351
powerspot.txt, 45
PSPOT, 45
range, 193
SAFT.mat, 228
SHIP, 225
SMORN.mat, 228
SPfutures.txt, 99
SUBSP, 232
swap, 91
TRGSP, 213, 214
trgsp.asc, 213
trgsp2.asc, 229
TRGSP3.asc, 229
TSTSP, 213, 214
tstsp.asc, 213
tstsp2.asc, 229
TSTSP3.asc, 229
us.bis.yield, 88
UTIL.index, 107
utilities.txt, 98
VINEYARD, 226
WSP, 6
wsp.asc, 6
WSPLRet, 14, 30
wspts.asc, 5

S-Plus Index

Symbols

.Data, 415
%*%, 416

A

abline, 112
acf, 265, 271
align, 250, 252, 372
apply, 193, 194
ar, 284, 286, 288, 297, 303
arima.diag, 288
arima.forecast, 289
arima.mle, 287, 288, 303
arima.sim, 285, 286, 288, 289, 303
attach, 106, 167, 421

B

bandwidth, 185
begday, 252
bivd, 75
bns, 164
box, 18, 185
bscall, 214, 217

C

c, 120, 330, 416
cbind, 62, 129
chapter, 413
col.names, 244
command window, 415
concat, 244
constructor, 244
copula.family, 77

copula.object, 79
cor.test, 69
cosine, 18
cov, 59
cumsum, 267, 384, 419

D

data, 130, 242
data frame, 106, 423
dcauchy, 33
density, 18, 20, 428
dexp, 11, 13, 33
df, 177, 178, 180
diag\$gov, 288
diff, 7, 121, 127, 128, 262, 419, 424
dim, 107, 194
DISPLAY, 415
dnorm, 33
dunif, 8, 33

E

eda.shape, 64, 428
eigen, 100
ENRON.ts, 406
esq, 198
estimate, 69
EVANESCE, 101
exit, 417

F

f, 183
fit.copula, 77
fitted, 147, 167, 183
floor, 406

fns, 163
From File, 423

G

garch, 372
Gaussian, 18
gdp, 38
gets, 415
gpd.lq, 40
gpd.tail, 37, 39, 42, 65
GUI, 415

H

hat, 138
help, 200, 416
hills, 421
hills\$climb, 421
hills\$dist, 421
hills\$time, 421
hist, 15, 20, 428
how, 252

I

idq, 428
Import Data, 423
innov, 288
integrated of order p , 260
integrated of order one, 260
isig, 214

J

julian, 406

K

kalman, 330
kdest, 52, 59
kernel, 185
knots, 178
kreg, 192
ks.gof, 45
ksmooth, 18, 20, 185, 189

L

llfit, 129
lag, 302
lag.plot, 274
layout, 274
legend, 178
length, 62, 422

library, 421
lines, 147, 167, 169
lm, 112, 125, 130, 131, 166, 173, 177
lm.diag, 138
loadings, 89, 91
localvar, 370
loess, 181, 182
loess.smooth, 167
log, 7, 424
lower, 42
lowess, 182
ls.diag, 125, 138
lsfit, 111, 125, 129
lty, 120

M

m.max, 198
m.min, 198
ma, 286, 288, 303
makedate, 406
mass, 421
mean, 35, 59, 62, 194, 425
mfrow, 15, 107, 418
min, 35, 425
model, 286, 288, 303
motif, 414
mvnorm, 101
mysqrt, 101

N

n, 20
n.points, 20
n.start, 288
names, 167
nls, 151
noon, 252
normal, 185
NOx, 167
ns, 177, 178
NULL, 330

O

Object Explorer, 415
one unit-root, 260
one.tail, 38, 42
Open, 423
options, 244
optlog, 38
order, 169, 284, 288

P

pairs, 108
 par, 15, 418
 partial, 271, 297
 parzen, 185
 pcauchy, 33
 PCS, 4
 persp.dbivd, 75
 persp.pbivd, 76
 pexp, 11, 13, 33
 plot, 41, 107, 419, 426
 pnorm, 9, 33
 points, 20
 poly, 148, 177
 poly.transform, 167
 positions, 242
 ppreg, 198
 pred.ar, 319
 princomp, 87, 88, 91
 probability, 20
 project, 413
 punif, 8, 33

Q

q, 417
 qcauchy, 25, 33
 qexp, 33
 qnorm, 24, 33
 qqexp, 31
 qqline, 428
 qqnorm, 428
 qunif, 33

R

range, 214
 rcauchy, 33
 rcopula, 79
 read.table, 242, 244, 425
 readindex, 406
 rev, 424
 rexp, 33
 rho, 59, 79
 rlnorm, 383
 rmvnorm, 58, 59, 62
 rnorm, 33, 101, 166, 288, 417, 428
 Run F10, 423
 runif, 33, 166

S

scan, 422
 script, 423
 sd, 59
 seq, 416
 seriesMerge, 302
 setenv, 415
 shape.plot, 39, 65
 shift, 302
 sigma.t, 369
 signalSeries, 302
 simulate.garch, 370, 373
 smooth.spline, 180
 solve, 330
 source, 330
 span, 181, 182
 spar, 180, 181
 sqrt, 35, 425
 start, 151, 370, 373
 stdres, 139
 stl, 262, 294
 studres, 139
 summary, 151, 428
 supsmu, 183, 198

T

t, 330, 416
 tailplot, 38
 tau, 79
 timeDate, 242
 timeSeq, 384
 timeSeries, 20, 138, 242, 384
 title, 420, 424
 trellis.device, 414
 triangle, 18, 185
 tsplot, 33, 138, 417, 423
 type, 419, 426

U

unclass, 121, 127, 128
 unitroot, 309, 320
 upper, 39, 42
 upper.par.est, 39
 USBN041700, 188

V

var, 35, 62, 425
 VaR.exp.portf, 82
 VaR.exp.sim, 83

W
window, 18
workspace, 413

X
xhost, 415

xpred, 192

Y
ylim, 20
yrs, 163

Author Index

A

Ait-Sahalia, 235
Akaike, 150
Amari, 173
Anderson, 102
Antoniadis, 173

B

Bachelier, 381
Bellman, 192
Berruyer, 173
Bibby, 173
Biss, 233
Black, 175, 207
Bollerslev, 410
Box, 308
Breedon, 102
Breiman, 236
Brown, 336, 352
Bucy, 326

C

Cérou, 411
Campbell, 309
Cantelli, 29
Carmona, 173
Chambers, 173, 429
Chan, 352
Cleveland, 46
Clewlow, 411
Crisan, 411

D

de Boor, 233

Deacon, 102
Del Moral, 411
Derry, 102
Dickey, 258, 281, 305, 309
Drouet Mari, 101
Durbin, 336, 352

E

Einstein, 381
El Karoui, 173
Engle, 351, 410
Euler, 383
Evans, 336, 352

F

Föllmer, 47
Fama, 233
Fan, 410
Fong, 173
Fouque, 410
Fourier, 309
Frank, 236
Friedman, 234, 236
Frostig, 309
Fuller, 258, 281, 305, 309

G

Gabor, 309
Gençay, 352
Glivenko, 29
Gourieroux, 47, 309, 410
Granger, 351
Guillounet, 411

H

Haerdle, 234
 Hamilton, 351
 Hastie, 234, 236
 Heston, 410
 Hill, 37
 Hooke, 389
 Huber, 173
 Hull, 309, 410

I

Ito, 208

J

Jasiak, 47, 309, 410
 Jenkins, 308
 Johansen, 351
 Jorion, 47

K

Kalman, 326, 411
 Karhunen, 102
 Kendall, 69, 79
 Kent, 173
 Kimberdoff, 233
 Kitagawa, 411
 Kohonen, 236
 Kotz, 101
 Krause, 46, 429
 Kushner, 411

L

Lamberton, 410
 Lapeyre, 410
 Le Gland, 411
 Leroy, 173
 Lintner, 173
 Litterman, 102
 Lo, 235, 309
 Loève, 102
 Lyons, 411

M

Macaulay, 158
 Mallat, 234
 Mandelbrot, 46, 355, 410
 Mardia, 173
 Markowitz, 173
 Mc Neil, 47

McKinlay, 309
 Merton, 207
 Miltersen, 411
 Montgomery, 173
 Morrisson, 47, 101
 Murphy, 102

N

Nelsen, 101
 Nelson, 160, 173

O

Olshen, 236
 Olson, 46, 429
 Ornstein, 382, 388, 409

P

Papanicolaou, 410
 Papavasiliou, 411
 Pareto, 32, 36, 64
 Parzen, 185
 Pearson, 53
 Peck, 173
 Pilipovic, 411
 Poisson, 46
 Priestley, 308

R

Raleigh, 86
 Rebonato, 102
 Renault, 410
 Rice, 173
 Ripley, 46, 234, 429
 Ritz, 86
 Roll, 173
 Rosenblatt, 309
 Rousseeuw, 173
 Ruud, 173

S

Sales, 411
 Samuelson, 207, 268, 380, 400, 419
 Scheinkmann, 102
 Schied, 47
 Scholes, 175, 207
 Schwartz, 411
 Selçuk, 352
 Shannon, 309
 Sharpe, 173

Shumway, 352
Siegel, 160, 173
Silverman, 236
Sircar, 410
Spearman, 79
Spector, 429
Spierman, 69
Steele, 410
Stettner, 410
Stoffer, 352
Stone, 236
Strickland, 411
Stuetze, 234
Swensson, 161, 173

T

Tibshirani, 234, 236
Touzi, 410
Tsay, 352

U

Uhlenbeck, 382, 388, 409

V

Vasicek, 173, 388, 389
Venables, 46, 234, 429
von Storch, 309

W

Wang, 309, 352
Whaba, 233
Whitcher, 352
White, 410
Wiener, 381
Witten, 236

Y

Yao, 410

Z

Zakai, 411
Zhang, 234
Zivot, 309, 352
Zwiers, 309

Subject Index

A

accrued interest, 158
acf, 128
AIC criterion, 284
alignment, 253
American Electric Power, 144
AR-representation, 279
arbitrage, 208
ARCH
 factor model, 325
 model, 410
ARIMA
 process, 282
 time series, 282
ASH, 16
Asian option, 293
asymptotically
 normal estimate, 359
 stationary, 356
at the money, 292
augmented Dickey-Fuller test, 281
auto-correlation function, 253
auto-covariance function, 253
auto-regressive, 268
 representation, 279
auto-regressive moving average, 281

B

B-spline, 233
back testing, 204, 205
backward shift operator, 275
backwardation, 151
bagging, 236

bandwidth, 17, 51, 183
Bank of International Settlements, 88, 160,
 173
basket options, 317
beta, 143
betas, 324
bid-ask spread, 156
bin, 14
Black-Scholes formula, 208
bond, 155
 coupon, 155
 discount, 155
 price equation, 156
 zero coupon, 155
boosting, 236
bootstrap, 191, 205
bootstrap method, 186
Brownian motion, 381, 386
 fractional, 355
 geometric, 400

C

calendar time series, 241
California crisis, 144
cap, 292
Capital Asset Pricing Model, 142, 337
capitalization, 405
categorical data, 14, 84
Cauchy distribution, 12, 24, 32, 36
causal process, 277
causality, 277
Central Bank, 150
charting, 309

- classification, 236
 - classification tree, 236
 - clean price, 157, 158
 - clustering, 236
 - coefficient of determination, 125, 132
 - cointegration, 319, 391
 - vector, 319
 - confidence band, 205
 - consistent estimate, 358
 - constructor, 244
 - contango, 151
 - contingent claim, 205
 - convex, 115
 - cooling degree day, 290
 - copula, 70–72
 - Archimedean, 96
 - B1Mix, 98
 - BB1, 97
 - BB2, 97
 - BB3, 97
 - BB4, 97
 - BB5, 98
 - BB6, 98
 - BB7, 98
 - density, 74
 - extreme value, 95
 - fitting, 76
 - Frank, 96
 - Galambos, 96
 - Gaussian, 72, 96
 - Gumbel, 73, 96
 - Hüsler and Reiss, 97
 - independent, 72
 - Kimeldorf and Sampson, 96
 - logistic, 73
 - normal, 72, 96
 - Twan, 97
 - correlation coefficient, 53
 - cost function, 110, 179
 - coupon
 - bond, 155
 - payment, 156
 - covariance, 54
 - cross validation, 180, 183, 191
 - cumulative distribution function, 7
 - curse of dimensionality, 177, 192, 197
 - CUSUM test, 336, 352
- D**
- data mining, 236
 - Data Stream, 91, 425
 - degree day, 290
 - degree of freedom, 180
 - density, 8
 - historical, 207
 - objective, 207
 - state price, 208
 - dependent variable, 109
 - design matrix, 134
 - diagnostics, 288
 - Dickey-Fuller statistic, 281, 305
 - Dickey-Fuller test, 281, 305
 - difference operator, 260
 - direct product, 190
 - discont rate, 155
 - discount
 - bond, 155
 - curve, 156
 - rate, 155
 - discounting, 208
 - discretization step, 382
 - distribution, 7
 - empirical, 28
 - Gaussian, 8
 - normal, 8
 - P&L, 26
 - shortfall, 27
 - standard Gaussian, 9
 - standard normal, 9
 - uniform, 8
 - distribution function
 - cumulative, 8
 - dividend, 6
 - double exponential distribution, 125
 - Dow Jones Indexes, 411
 - drift, 268
 - duration, 158
- E**
- eigenvalue
 - nondegenerate, 86
 - simple, 86
 - empirical
 - auto-covariance function, 258
 - cdf, 51
 - correlation, 54
 - covariance, 54

- distribution function, 28
- endogenous, 325
- Enron, 144, 365
- ergodic time series, 257
- estimate
 - asymptotically normal, 359
 - consistent, 358
- Euler scheme, 383
- European
 - call option, 205
 - option, 293
- EVANESCE, 47
- excess kurtosis, 358, 377, 406
- exogenous, 325
- exotic option, 205
- expected shortfall, 27
- explanatory variable, 109
- exponential distribution, 10
- extended Kalman filter, 323
- extreme value, 35
- extreme value theory, 36

F

- face value, 156
- factor
 - endogenous, 325
 - exogenous, 325
- filtering, 322
- first order condition, 114
- floor, 293
- Fourier
 - analysis, 259
 - transform, 309
- fractional
 - Brownian motion, 355
 - differentiation, 356
 - process, 356
 - time series, 356
- frequency
 - sampling, 19, 381
- futures contract, 244

G

- GARCH model, 410
- Gaussian, 123
- Gaussian distribution, 8
- generalized Pareto distribution, 12, 32, 36, 38, 64
- generalized Vasicek family, 172

- geometric
 - Brownian motion, 208, 380, 400
 - Ornstein Uhlenbeck, 409
 - Ornstein-Uhlenbeck process, 403
- global minimum, 116
- goodness of fit, 288
- gradient, 116
- Gumbel copula, 77

H

- hat matrix, 135, 138
- heating degree day, 290
- heavy tail, 32, 289
- heteroskedasticity, 361
- hidden Markov model, 393
- Hill's estimator, 37
- histogram, 14, 51
- historical
 - average, 292
 - density, 207
 - volatility, 210
- hockey stick, 209
- Hooke's law, 389

I

- implied volatility, 211, 213, 379
- importance sampling, 410
- independent random variables, 14
- independent variable, 109
- inflation rate, 389
- influence, 136
- influence measure, 138
- information, 375
- innovation, 256, 327, 363
- instrument, 154
- interest rate
 - long, 385
 - real, 389
 - short, 161, 385
 - spot, 155
- invertibility, 277
- invertible, 303
 - process, 279
- investment
 - beta, 143
 - grade, 411
- irregular time series, 241
- iterative extraction, 186
- Ito correction, 208

J

joint cdf, 50
 jointly normal random variables, 56

K

k-nearest neighbors, 235
 Kalman
 filtering, 326
 gain, 328
 Kalman filter
 extended, 323
 Kalman gain matrix, 335
 Karhunen-Loève decomposition, 102
 Kendall's τ , 69
 kernel, 17, 51, 183
 density estimation, 52
 function, 184
 knots, 177
 kurtosis, 377
 excess, 358, 377

L

L1 regression, 111
 L2 regression, 111
 lag, 257, 270
 Laplace distribution, 125
 least absolute deviations regression, 110
 least squares regression, 110
 left tail, 37
 leverage effect, 377, 378, 401
 likelihood function, 124
 linear
 dependence, 55
 factor model, 325
 model, 125, 130, 134
 process, 276
 regression, 124
 linearization, 323
 loading, 85, 86, 88
 local minimum, 116
 locally weighted regression, 181
 location, 118
 parameter, 32
 log-return, 6, 207
 lognormal
 distribution, 100
 random variable, 100
 long
 interest rate, 385

range dependence, 356
 range memory, 356

loss, 115

M

machine learning, 236
 Markov chain, 394
 martingale difference, 379
 matching pursuit, 234
 matrix
 design, 134
 hat, 135
 prediction, 135
 square root, 58
 maturity, 155
 date, 154, 155, 205
 maximum likelihood, 334
 mean, 31
 function, 253
 median, 119
 method of moments, 410
 model, 240
 moneyiness, 217
 Monte Carlo, 393
 Markov Chain, 410
 simulations, 82
 Moody's, 411
 moving average, 270, 272
 representation, 278
 multiple regression, 124, 129
 multiscale volatility, 405

N

natural spline, 233
 Nelson-Siegel family, 160
 neural network, 234
 Nobel prize, 207, 351, 410
 noise, 124
 colored, 313
 white, 313
 nominal
 pay-off rate, 292
 value, 155
 non-anticipative, 255, 338, 369
 nondegenerate eigenvalue, 86
 nonnegative definite, 258
 nonparametric regression, 115, 124
 normal
 copula, 71

distribution, 8
normalized linear combination, 85

O

objective
 density, 207
 function, 179
observation equation, 322
option, 205, 291
 American, 205
 Asian, 293
 at the money, 217, 380
 basket, 317
 European, 293
 exotic, 205
 expiration, 205
 in the money, 217, 380
 maturity, 205
 out of the money, 217, 380
 strike price, 205
 temperature, 291
order statistic, 29
Ornstein-Uhlenbeck process, 382, 388, 409
 geometric, 409
out of the money, 292
outlier, 138
outlying observation, 136, 138

P

parallel shift, 89
parametric regression, 115, 124
Pareto distribution, 32, 36
partial auto-correlation function, 253, 256, 270
partial derivative, 116
particle approximation, 393
pay-off, 205
pay-off function, 291
PCS Index, 4
peak over threshold, 37
penalty, 115
percentile, 23
perfect observation, 340
persistence, 356
plain vanilla, 91
Poisson distribution, 46
pormanteau test, 266
prediction, 289, 322
 linear, 326

 matrix, 135
 quadratic error, 327
principal, 154
 component, 87
 component analysis, 84, 85, 160, 236
 value, 156
process
 fractional, 356
projection pursuit, 197
 regression, 198
pseudo-MLE, 358

Q

Q-Q plot, 24
quantile, 23
 function, 23
quasi-MLE, 358

R

random
 variable, 7
 walk, 267, 419
 walk with drift, 268
rate, 11, 31
raw
 residuals, 136
 return, 6
real interest rate, 389
recursive filtering equations, 326
regime switching, 405
regression
 diagnostics, 128
 function, 123, 176
 L1, 111
 L2, 111
 linear, 124
 multiple, 124, 129
 nonparametric, 115, 124
 parametric, 115, 124
 polynomial, 115
 projection pursuit, 198
 semi-parametric, 213
 significant, 118
 simple, 124
 time series, 123
regressor, 109
regular time series, 240
regularization, 179
relaxation, 389

required capital, 26
 residual, 131
 raw, 136
 standardized, 137
 studentized, 137
 response
 surface, 195
 variable, 109
 return, 6
 log, 6
 raw, 6
 rho, 69
 right tail, 37
 risk neutral, 208
 RiskMetrics, 47
 robust, 173
 robustness, 120, 122
 root one, 267

S

S&P 500, 423
 S&P 500 index, 5
 sample
 auto-correlation function, 258
 mean, 119
 size, 111
 sampling
 frequency, 19, 240, 381
 interval, 240
 theorem, 309
 scale parameter, 32
 scatterplot, 108, 426
 smoother, 178
 scenario, 385
 script, 427
 seasonal component, 258, 259, 343
 seemingly unrelated regressions, 142
 self-similarity, 44, 356
 semi-parametric, 213
 shape
 index, 37
 parameter, 37, 38
 shift operator, 312
 short interest rate, 161, 385
 shortfall
 distribution, 27
 expected, 27
 signal, 240
 significant, 118
 simple
 eigenvalue, 86
 regression, 124
 simulation, 288
 sliding window, 369
 slot, 242
 smile, 212, 400
 effect, 379
 smoother, 178
 smoothing, 115, 322
 parameter, 179, 181
 spline, 179, 233
 spectral analysis, 259
 spline
 B, 233
 natural, 233
 smoothing, 179
 spot interest rate, 155
 spread, 391
 stable distribution, 36
 standard
 deviation function, 253
 Gaussian distribution, 9
 normal distribution, 9
 standardized residuals, 137
 state
 equation, 321, 322
 price density, 208, 221
 stationarity, 254, 277, 312
 strong, 254
 test, 258
 weak, 254
 stationary, 254, 303
 stochastic
 differential equation, 380
 volatility, 375, 409
 strictly convex, 115
 strike price, 205
 strong stationarity, 254
 structural model, 343
 Student copula, 71
 studentized residuals, 137
 survival function, 38
 Swensson family, 161

T

tail, 15, 22, 35
 tau, 69

technical analysis, 309
 temperature
 option, 291
 tensor product, 190
 test
 augmented Dickey-Fuller, 281
 Dickey-Fuller, 281, 305
 sample, 205
 stationarity, 258
 unit-root, 258, 280, 309
 testing, 204
 Texas Utilities, 144
 threshold, 39
 tick data, 245
 time series
 alignment, 253
 calendar, 241
 ergodicity, 257
 fractional, 356
 irregular, 241
 model, 240
 regression, 123
 regular, 240
 trade data, 245
 training, 204
 sample, 205
 transaction data, 245
 tree, 236
 trend, 258, 259, 343

U

uniform distribution, 8
 unimodal distribution, 12, 15
 unit root test, 320

unit-root test, 258, 280, 309

V

Value at Risk, 25
 value at risk, 27
 VaR, 47
 variable
 dependent, 109
 explanatory, 109
 independent, 109
 variance function, 253
 Vasicek model, 388
 volatility, 207, 381
 historical, 210
 implied, 211
 ratio, 193, 195
 smile, 212, 380
 stochastic, 409
 volvol, 400

W

wavelet, 309
 wavelet packets, 236
 weakly stationary, 254
 weekly return, 7
 weights, 179
 well posed, 115
 white noise, 34, 264
 Wiener process, 381, 383, 386

Y

Yahoo, 423
 yield curve, 150, 159
 Yule-Walker equation, 269, 313