Updates for R-1.9.0 & R-2.0.0

Data Analysis and Graphics Using R
– An Example-Based Approach

John Maindonald and John Braun

These notes draw attention to changes in R-1.9.0 and R-2.0.0 that have implications for the discussion in our book (September 9, 2006)

Contents

The script editor window.

The \texttt{data()} function

Automatic generation of keys when using lattice functions

Functions for working with dates

Use of a factor as the \texttt{labels} argument to \texttt{text()}

Changes to default packages

The script editor window

Windows and Mac OS X versions of R now have a script editor window that can be opened from the menu.

The \texttt{data()} function

Datasets that are in the \texttt{datasets}, \texttt{MASS}, \texttt{DAAG} (recent revisions) and most (not all) other packages can, as of R-2.0.0, be accessed directly when the package is attached. Use of the function \texttt{data()} is not required. A current exception is the \texttt{mclust} package, where the function \texttt{data()} must be used to load datasets.

Chapter 2

\textit{p.38-41, Section 2.1.5: Lattice graphics}

There are now better abilities for the automatic generation of keys, using the parameter \texttt{auto.key}:

\begin{verbatim}
xyplot(csoa ~ it | sex*agegp, data=tinting,
       groups=target, auto.key=list(columns=2, between=1))
xyplot(csoa ~ it | sex*agegp, data=tinting,
       groups=tint, auto.key=list(columns=3, between=1))
\end{verbatim}

The function \texttt{trellis.par.set()} has replaced \texttt{lset()}, for use in setting graphics parameters. For examples of its use, see a current version of the “Additional Notes”.

Chapter 9 – Section 9.5 on Time Series

Omit use of \texttt{library(ts)}, wherever this appears. The \texttt{stats} package, which has absorbed \texttt{ts}, is attached automatically in vanilla installations.
Chapter 12
p.310, Subsection 12.2.3: Functions for working with dates

In version 1.9.0, the `date` package has been superseded by functions for working with dates that are in R base. See `help(Dates)` and `help(as.Date)` and `help(format.Date)` for information that is more complete than we provide below.

Use `as.Date()` to convert text strings into dates. Use `format()` to set or change the way that a date is formatted for printing. The following are a selection of the symbols that are available:

- `%d`: day, as number
- `%a`: abbreviated weekday name (%A: unabbreviated)
- `%m`: month (00-12)
- `%b`: month, abbreviated name (%B: unabbreviated)
- `%y`: final two digits of year (%Y: all four digits)

The default format is "%Y-%m-%d"

The function `as.Date()` takes a vector of character strings that have an appropriate format, and converts it into a dates object. Dates are stored using January 1 1970 as origin, as is apparent from printing the value that results when `as.integer()` is used to convert a date into an integer value. Here are examples:

```r
> as.Date("1/1/1960", format="%d/%m/%Y")
[1] "1960-01-01"
> as.Date("1:12:1960", format="%d:%m:%Y")
[1] "1960-12-01"
> as.Date("1960-12-1")-as.Date("1960-1-1")
    as.Date("31/12/1960","%d/%m/%Y")
[1] "1960-12-31"
> as.integer(as.Date("1/1/1970","%d/%m/%Y")
[1] 0
> as.integer(as.Date("1/1/2000","%d/%m/%Y")
[1] 10957
```

The function `format()` allows control of the formatting of dates in printed output. See `help(format.Date)`.

```r
> dec1 <- as.Date("2004-12-1")
> format(dec1, format="%b %d %Y")
[1] "Dec 01 2004"
> format(dec1, format="%a %b %d %Y")
[1] "Wed Dec 01 2004"
```

Such formatting may be used to give meaningful labels on graphs, thus

```r
data(jobs)   # DAAG package
Jobs <- stack(jobs, select=-7)
startofmonth <- seq(from=as.Date("1Jan1990", format="%d%b%Y"), by="1 month", length=24)
Jobs$Date <- rep(startofmonth, 6)
names(Jobs) <- c("Number", "Province", "Date")
atdates <- seq(from=as.Date("1Jan1990", format="%d%b%Y"), by="3 month", length=8)
datelabs <- format(atdates, "%b%y")
```
Changes in R-1.9.0 and R-2.0.0

```r
xyplot(Number ~ Date, groups=Province, data=Jobs,
       scale=list(x=list(at=as.numeric(atdates), labels=datelabs)),
       auto.key=list(columns=6, between=1))
```

Note the use of the function `seq()` to give a regular sequence of dates. See `help(seq.Date)` for details of the options that are available.

The function `date()` returns the current date and time, while `Sys.Date()` returns the date.

**p.315, Section 12.4: Factors**

**line -19**

To extract the codes 1, 2, ..., specify `unclass(country)`. 

**lines -12 and -11**

If the `labels` argument to `text()` is a factor, as of 2.0.0 it is the levels that are plotted.

**p.330, Section 12.9: Changes to default packages**

On starting R-1.9.0, the search path that appears is

```r
> search()
[1] ".GlobalEnv" "package:methods" "package:stats"
[7] "package:base"
```

These names reflect a reorganization of `base` and of the packages that are loaded by default.

In R-2.0.0, there have been further changes. The search path that appears upon startup is:

```r
> search()
[1] ".GlobalEnv" "package:methods" "package:stats"
[7] "package:datasets" "Autoloads" "package:base"
```

In version 1.9.0, the former `base` package has been split between the new packages `base`, `graphics`, `stats` and `utils`. All four are loaded in a default installation. Packages `ctest`, `eda`, `modreg`, `mle`, `mva`, `nls`, `stepfun` and `ts` have been merged into `stats`. The code from `lqs` is incorporated into `MASS`. Note the introduction, with R-2.0.0, of the new packages `grDevices` (graphical devices) and `datasets`. Datasets that were formerly in `base` and `stats` have been moved to this package.

For the time being, use of the former names in the `library()` command will ensure that the requisite package is loaded. Thus `library(lqs)` will have the effect of loading the `MASS` package, and a warning will be issued.

**Implications for DAAGUR**

There is mention of the packages `modreg`, `ts` and `eda`. These no longer exist. They have been merged into the `stats` package, which is attached by default.