R FAQ

Kurt Hornik

Table of Contents

1 Introduction

This document contains answers to some of the most frequently asked questions about R.

1.1 Legalese

This document is copyright © 1998–2006 by Kurt Hornik.

This document is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This document is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

A copy of the GNU General Public License is available via WWW at

> http://www.gnu.org/copyleft/gpl.html.

You can also obtain it by writing to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, U.S.A.

1.2 Obtaining this document

The latest version of this document is always available from

> http://CRAN.R-project.org/doc/FAQ/

From there, you can obtain versions converted to plain ASCII text, DVI, GNU info, HTML, PDF, PostScript as well as the Texinfo source used for creating all these formats using the GNU Texinfo system.

You can also obtain the R FAQ from the 'doc/FAQ' subdirectory of a CRAN site (see Section 2.10 [What is CRAN?], page 9).

1.3 Citing this document

In publications, please refer to this FAQ as Hornik (2006), "The R FAQ", and give the above, *official* URL and the ISBN 3-900051-08-9.

1.4 Notation

Everything should be pretty standard. 'R>' is used for the R prompt, and a '$' for the shell prompt (where applicable).

1.5 Feedback

Feedback via email to Kurt.Hornik@R-project.org is of course most welcome.

In particular, note that I do not have access to Windows or Macintosh systems. Features specific to the Windows and Mac OS X ports of R are described in the "R for Windows FAQ" and the "R for Mac OS X FAQ". If you have information on Macintosh or Windows systems that you think should be added to this document, please let me know.

2 R Basics

2.1 What is R?

R is a system for statistical computation and graphics. It consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files.

The design of R has been heavily influenced by two existing languages: Becker, Chambers & Wilks' S (see Section 3.1 [What is S?], page 11) and Sussman's Scheme. Whereas the resulting language is very similar in appearance to S, the underlying implementation and semantics are derived from Scheme. See Section 3.3 [What are the differences between R and S?], page 12, for further details.

The core of R is an interpreted computer language which allows branching and looping as well as modular programming using functions. Most of the user-visible functions in R are written in R. It is possible for the user to interface to procedures written in the C, C++, or FORTRAN languages for efficiency. The R distribution contains functionality for a large number of statistical procedures. Among these are: linear and generalized linear models, nonlinear regression models, time series analysis, classical parametric and nonparametric tests, clustering and smoothing. There is also a large set of functions which provide a flexible graphical environment for creating various kinds of data presentations. Additional modules ("add-on packages") are available for a variety of specific purposes (see Chapter 5 [R Add-On Packages], page 21).

R was initially written by Ross Ihaka and Robert Gentleman at the Department of Statistics of the University of Auckland in Auckland, New Zealand. In addition, a large group of individuals has contributed to R by sending code and bug reports.

Since mid-1997 there has been a core group (the "R Core Team") who can modify the R source code archive. The group currently consists of Doug Bates, John Chambers, Peter Dalgaard, Robert Gentleman, Kurt Hornik, Stefano Iacus, Ross Ihaka, Friedrich Leisch, Thomas Lumley, Martin Maechler, Duncan Murdoch, Paul Murrell, Martyn Plummer, Brian Ripley, Duncan Temple Lang, Luke Tierney, and Simon Urbanek.

R has a home page at http://www.R-project.org/. It is free software distributed under a GNU-style copyleft, and an official part of the GNU project ("GNU S").

2.2 What machines does R run on?

R is being developed for the Unix, Windows and Mac families of operating systems. Support for Mac OS Classic ended with R 1.7.1.

The current version of R will configure and build under a number of common Unix platforms including *cpu*-linux-gnu for the i386, alpha, arm, hppa, ia64, m68k, mips/mipsel, powerpc, s390, sparc (e.g., http://buildd.debian.org/build.php?&pkg=r-base), and x86_64 CPUs, powerpc-apple-darwin, mips-sgi-irix, rs6000-ibm-aix, and sparc-sun-solaris.

If you know about other platforms, please drop us a note.

2.3 What is the current version of R?

The current released version is 2.3.0. Based on this 'major.minor.patchlevel' numbering scheme, there are two development versions of R, a patched version of the current release ('r-patched') and one working towards the next minor or eventually major ('r-devel') releases of R, respectively. Version r-patched is for bug fixes mostly. New features are typically introduced in r-devel.

2.4 How can R be obtained?

Sources, binaries and documentation for R can be obtained via CRAN, the "Comprehensive R Archive Network" (see Section 2.10 [What is CRAN?], page 9).

Sources are also available via `https://svn.R-project.org/R/`, the R Subversion repository, but currently not via anonymous rsync (nor CVS).

Tarballs with daily snapshots of the r-devel and r-patched development versions of R can be found at `ftp://ftp.stat.math.ethz.ch/Software/R`.

2.5 How can R be installed?

2.5.1 How can R be installed (Unix)

If R is already installed, it can be started by typing `R` at the shell prompt (of course, provided that the executable is in your path).

If binaries are available for your platform (see Section 2.6 [Are there Unix binaries for R?], page 5), you can use these, following the instructions that come with them.

Otherwise, you can compile and install R yourself, which can be done very easily under a number of common Unix platforms (see Section 2.2 [What machines does R run on?], page 2). The file 'INSTALL' that comes with the R distribution contains a brief introduction, and the "R Installation and Administration" guide (see Section 2.7 [What documentation exists for R?], page 5) has full details.

Note that you need a FORTRAN compiler or perhaps `f2c` in addition to a C compiler to build R. Also, you need Perl version 5 to build the R object documentations. (If this is not available on your system, you can obtain a PDF version of the object reference manual via CRAN.)

In the simplest case, untar the R source code, change to the directory thus created, and issue the following commands (at the shell prompt):

```
$ ./configure
$ make
```

If these commands execute successfully, the R binary and a shell script front-end called 'R' are created and copied to the '`bin`' directory. You can copy the script to a place where users can invoke it, for example to '`/usr/local/bin`'. In addition, plain text help pages as well as HTML and LaTeX versions of the documentation are built.

Use *make dvi* to create DVI versions of the R manuals, such as '`refman.dvi`' (an R object reference index) and '`R-exts.dvi`', the "R Extension Writers Guide", in the '`doc/manual`' subdirectory. These files can be previewed and printed using standard programs such as `xdvi` and `dvips`. You can also use *make pdf* to build PDF (Portable Document Format)

version of the manuals, and view these using e.g. Acrobat. Manuals written in the GNU Texinfo system can also be converted to info files suitable for reading online with Emacs or stand-alone GNU Info; use `make info` to create these versions (note that this requires Makeinfo version 4.5).

Finally, use `make check` to find out whether your R system works correctly.

You can also perform a "system-wide" installation using `make install`. By default, this will install to the following directories:

'`${prefix}/bin`'
> the front-end shell script

'`${prefix}/man/man1`'
> the man page

'`${prefix}/lib/R`'
> all the rest (libraries, on-line help system, . . .). This is the "R Home Directory" (`R_HOME`) of the installed system.

In the above, `prefix` is determined during configuration (typically '`/usr/local`') and can be set by running `configure` with the option

```
$ ./configure --prefix=/where/you/want/R/to/go
```

(E.g., the R executable will then be installed into '`/where/you/want/R/to/go/bin`'.)

To install DVI, info and PDF versions of the manuals, use `make install-dvi`, `make install-info` and `make install-pdf`, respectively.

2.5.2 How can R be installed (Windows)

The '`bin/windows`' directory of a CRAN site contains binaries for a base distribution and a large number of add-on packages from CRAN to run on Windows 95, 98, ME, NT4, 2000, and XP (at least) on Intel and clones (but not on other platforms). The Windows version of R was created by Robert Gentleman and Guido Masarotto, and is now being developed and maintained by Duncan Murdoch and Brian D. Ripley.

For most installations the Windows installer program will be the easiest tool to use.

See the "R for Windows FAQ" for more details.

2.5.3 How can R be installed (Macintosh)

The '`bin/macosx`' directory of a CRAN site contains a standard Apple installer package inside a disk image named '`R.dmg`'. Once downloaded and executed, the installer will install the current non-developer release of R. RAqua is a native Mac OS X Darwin version of R with a R.app Mac OS X GUI. Inside '`bin/macosx/powerpc/contrib/x.y`' there are prebuilt binary packages (for powerpc version of Mac OS X) to be used with RAqua corresponding to the "$x.y$" release of R. The installation of these packages is available through the "Package" menu of the R.app GUI. This port of R for Mac OS X is maintained by Stefano Iacus. The "R for Mac OS X FAQ has more details.

The '`bin/macos`' directory of a CRAN site contains bin-hexed ('`hqx`') and stuffit ('`sit`') archives for a base distribution and a large number of add-on packages of R 1.7.1 to run under Mac OS 8.6 to Mac OS 9.2.2. This port of R for Macintosh is no longer supported.

2.6 Are there Unix binaries for R?

The 'bin/linux' directory of a CRAN site contains the following packages.

	CPU	Versions	Provider
Debian et al.	i386	stable/oldstable	Christian Steigies
Mandrake	i386	10.0	Michele Alzetta
Red Hat	i386	FC3/FC4/FC5	Martyn Plummer
	x86_64	FC3	Brian Ripley
	x86_64	FC4/FC5	Martyn Plummer
	i386	Enterprise Linux	Matthew P. Cox
	x86_64	Enterprise Linux	Matthew P. Cox
SuSE	i386	7.3/8.0/8.1/8.2	Detlef Steuer
	i586	9.0/9.1/9.2/9.3/10.0	Detlef Steuer
	x86_64	9.2/9.3/10.0	Detlef Steuer
VineLinux	i386	3.2	Susunu Tanimura

Debian packages, maintained by Dirk Eddelbuettel and Doug Bates, have long been part of the Debian distribution, and can be accessed through APT, the Debian package maintenance tool. Use e.g. `apt-get install r-base r-recommended` to install the R environment and recommended packages. If you also want to build R packages from source, also run `apt-get install r-base-dev` to obtain the additional tools required for this. So-called "backports" of the current R packages for the *stable* distribution of Debian are provided by Christian Steigies, and available from CRAN. Simply add the line

```
deb http://cran.R-project.org/bin/linux/debian stable/
```

(feel free to use a CRAN mirror instead of the master) to the file '/etc/apt/sources.list', and install as usual. More details on installing and administering R on Debian Linux can be found at http://cran.r-project.org/bin/linux/debian/README. While the *stable* backport was built for Debian, it has also been found to be directly usable for Ubuntu/Kubuntu, and should be suitable for other Debian derivatives.

No other binary distributions are currently publically available via CRAN.

A "live" Linux distribution with a particular focus on R is *Quantian*, which provides a directly bootable and self-configuring "Live DVD" containing numerous applications of interests to scientists and researchers, including several hundred CRAN and Bioconductor packages, the "ESS" extensions for Emacs, the "JGR" Java GUI for R, the Ggobi visualization tool as well as several other R interfaces. The *Quantian* website at http://dirk.eddelbuettel.com/quantian/ contains more details as well download information.

2.7 What documentation exists for R?

Online documentation for most of the functions and variables in R exists, and can be printed on-screen by typing *help(name)* (or *?name*) at the R prompt, where *name* is the name of the topic help is sought for. (In the case of unary and binary operators and control-flow special forms, the name may need to be be quoted.)

This documentation can also be made available as one reference manual for on-line reading in HTML and PDF formats, and as hardcopy via LaTeX, see Section 2.5 [How can

R be installed?], page 3. An up-to-date HTML version is always available for web browsing at http://stat.ethz.ch/R-manual/.

Printed copies of the R reference manual for some version(s) are available from Network Theory Ltd, at http://www.network-theory.co.uk/R/base/. For each set of manuals sold, the publisher donates USD 10 to the R Foundation (see Section 2.13 [What is the R Foundation?], page 10).

The R distribution also comes with the following manuals.

- "An Introduction to R" ('R-intro') includes information on data types, programming elements, statistical modeling and graphics. This document is based on the "Notes on S-Plus" by Bill Venables and David Smith.
- "Writing R Extensions" ('R-exts') currently describes the process of creating R add-on packages, writing R documentation, R's system and foreign language interfaces, and the R API.
- "R Data Import/Export" ('R-data') is a guide to importing and exporting data to and from R.
- "The R Language Definition" ('R-lang'), a first version of the "Kernighan & Ritchie of R", explains evaluation, parsing, object oriented programming, computing on the language, and so forth.
- "R Installation and Administration" ('R-admin').

Books on R include

P. Dalgaard (2002), "Introductory Statistics with R", Springer: New York, ISBN 0-387-95475-9.
http://www.biostat.ku.dk/~pd/ISwR.html.

J. Fox (2002), "An R and S-Plus Companion to Applied Regression", Sage Publications, ISBN 0-761-92280-6 (softcover) or 0-761-92279-2 (hardcover),
http://socserv.socsci.mcmaster.ca/jfox/Books/Companion/.

J. Maindonald and J. Braun (2003), "Data Analysis and Graphics Using R: An Example-Based Approach", Cambridge University Press, ISBN 0-521-81336-0,
http://wwwmaths.anu.edu.au/~johnm/.

S. M. Iacus and G. Masarotto (2002), "Laboratorio di statistica con R", McGraw-Hill, ISBN 88-386-6084-0 (in Italian),
http://www.ateneonline.it/LibroAteneo.asp?item_id=1436.

P. Murrell (2005), "R Graphics", Chapman & Hall/CRC, ISBN: 1-584-88486-X,
http://www.stat.auckland.ac.nz/~paul/RGraphics/rgraphics.html.

The book

W. N. Venables and B. D. Ripley (2002), "Modern Applied Statistics with S. Fourth Edition". Springer, ISBN 0-387-95457-0

has a home page at http://www.stats.ox.ac.uk/pub/MASS4/ providing additional material. Its companion is

W. N. Venables and B. D. Ripley (2000), "S Programming". Springer, ISBN 0-387-98966-8

and provides an in-depth guide to writing software in the S language which forms the basis of both the commercial S-Plus and the Open Source R data analysis software systems. See http://www.stats.ox.ac.uk/pub/MASS3/Sprog/ for more information.

In addition to material written specifically or explicitly for R, documentation for S/S-Plus (see Chapter 3 [R and S], page 11) can be used in combination with this FAQ (see Section 3.3 [What are the differences between R and S?], page 12). Introductory books include

> P. Spector (1994), "An introduction to S and S-Plus", Duxbury Press.

> A. Krause and M. Olsen (2005), "The Basics of S-Plus" (Fourth Edition). Springer, ISBN 0-387-26109-5.

The book

> J. C. Pinheiro and D. M. Bates (2000), "Mixed-Effects Models in S and S-Plus", Springer, ISBN 0-387-98957-0

provides a comprehensive guide to the use of the **nlme** package for linear and nonlinear mixed-effects models.

As an example of how R can be used in teaching an advanced introductory statistics course, see

> D. Nolan and T. Speed (2000), "Stat Labs: Mathematical Statistics Through Applications", Springer Texts in Statistics, ISBN 0-387-98974-9

This integrates theory of statistics with the practice of statistics through a collection of case studies ("labs"), and uses R to analyze the data. More information can be found at http://www.stat.Berkeley.EDU/users/statlabs/.

Last, but not least, Ross' and Robert's experience in designing and implementing R is described in Ihaka & Gentleman (1996), "R: A Language for Data Analysis and Graphics", *Journal of Computational and Graphical Statistics*, **5**, 299–314.

An annotated bibliography (BibTeX format) of R-related publications which includes most of the above references can be found at

> http://www.R-project.org/doc/bib/R.bib

2.8 Citing R

To cite R in publications, use

```
@Manual{,
  title        = {R: A Language and Environment for Statistical
                   Computing},
  author       = {{R Development Core Team}},
  organization = {R Foundation for Statistical Computing},
  address      = {Vienna, Austria},
  year         = 2006,
  note         = {{ISBN} 3-900051-07-0},
  url          = {http://www.R-project.org}
}
```

Citation strings (or BibTeX entries) for R and R packages can also be obtained by citation().

2.9 What mailing lists exist for R?

Thanks to Martin Maechler, there are four mailing lists devoted to R.

R-announce

> A moderated list for major announcements about the development of R and the availability of new code.

R-packages

> A moderated list for announcements on the availability of new or enhanced contributed packages.

R-help   The 'main' R mailing list, for discussion about problems and solutions using R, announcements (not covered by 'R-announce' and 'R-packages') about the development of R and the availability of new code.

R-devel   This list is for questions and discussion about code development in R.

Please read the posting guide *before* sending anything to any mailing list.

Note in particular that R-help is intended to be comprehensible to people who want to use R to solve problems but who are not necessarily interested in or knowledgeable about programming. Questions likely to prompt discussion unintelligible to non-programmers (e.g., questions involving C or C++) should go to R-devel.

Convenient access to information on these lists, subscription, and archives is provided by the web interface at `http://stat.ethz.ch/mailman/listinfo/`. One can also subscribe (or unsubscribe) via email, e.g. to R-help by sending '`subscribe`' (or '`unsubscribe`') in the *body* of the message (not in the subject!) to `R-help-request@lists.R-project.org`.

Send email to `R-help@lists.R-project.org` to send a message to everyone on the R-help mailing list. Subscription and posting to the other lists is done analogously, with '`R-help`' replaced by '`R-announce`', '`R-packages`', and '`R-devel`', respectively. Note that the R-announce and R-packages lists are gatewayed into R-help. Hence, you should subscribe to either of them only in case you are not subscribed to R-help.

It is recommended that you send mail to R-help rather than only to the R Core developers (who are also subscribed to the list, of course). This may save them precious time they can use for constantly improving R, and will typically also result in much quicker feedback for yourself.

Of course, in the case of bug reports it would be very helpful to have code which reliably reproduces the problem. Also, make sure that you include information on the system and version of R being used. See Chapter 9 [R Bugs], page 83 for more details.

See `http://www.R-project.org/mail.html` for more information on the R mailing lists.

The R Core Team can be reached at `R-core@lists.R-project.org` for comments and reports.

Many of the R project's mailing lists are also available via Gmane, from which they can be read with a web browser, using an NNTP news reader, or via RSS feeds. See `http://dir.gmane.org/index.php?prefix=gmane.comp.lang.r.` for the available mailing lists, and `http://www.gmane.org/rss.php` for details on RSS feeds.

2.10 What is CRAN?

The "Comprehensive R Archive Network" (CRAN) is a collection of sites which carry identical material, consisting of the R distribution(s), the contributed extensions, documentation for R, and binaries.

The CRAN master site at TU Wien, Austria, can be found at the URL

> http://cran.R-project.org/

Daily mirrors are available at URLs including

http://cran.at.R-project.org/	(TU Wien, Austria)
http://cran.au.R-project.org/	(PlanetMirror, Australia)
http://cran.br.R-project.org/	(Universidade Federal de Paraná, Brazil)
http://cran.ch.R-project.org/	(ETH Zürich, Switzerland)
http://cran.dk.R-project.org/	(SunSITE, Denmark)
http://cran.es.R-project.org/	(Spanish National Research Network, Madrid, Spain)
http://cran.fr.R-project.org/	(INRA, Toulouse, France)
http://cran.hu.R-project.org/	(Semmelweis U, Hungary)
http://cran.pt.R-project.org/	(Universidade do Porto, Portugal)
http://cran.uk.R-project.org/	(U of Bristol, United Kingdom)
http://cran.us.R-project.org/	(pair Networks, USA)
http://cran.za.R-project.org/	(Rhodes U, South Africa)

See http://cran.R-project.org/mirrors.html for a complete list of mirrors. Please use the CRAN site closest to you to reduce network load.

From CRAN, you can obtain the latest official release of R, daily snapshots of R (copies of the current source trees), as gzipped and bzipped tar files, a wealth of additional contributed code, as well as prebuilt binaries for various operating systems (Linux, Mac OS Classic, Mac OS X, and MS Windows). CRAN also provides access to documentation on R, existing mailing lists and the R Bug Tracking system.

To "submit" to CRAN, simply upload to ftp://cran.R-project.org/incoming/ and send an email to cran@R-project.org. Note that CRAN generally does not accept submissions of precompiled binaries due to security reasons. In particular, binary packages for Windows and Mac OS X are provided by the respective binary package maintainers.

> **Note:** It is very important that you indicate the copyright (license) information (GPL, BSD, Artistic, . . .) in your submission.

Please always use the URL of the master site when referring to CRAN.

2.11 Can I use R for commercial purposes?

R is released under the GNU General Public License (GPL). If you have any questions regarding the legality of using R in any particular situation you should bring it up with your legal counsel. We are in no position to offer legal advice.

It is the opinion of the R Core Team that one can use R for commercial purposes (e.g., in business or in consulting). The GPL, like all Open Source licenses, permits all and any use of the package. It only restricts distribution of R or of other programs containing code

from R. This is made clear in clause 6 ("No Discrimination Against Fields of Endeavor") of the Open Source Definition:

> The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

It is also explicitly stated in clause 0 of the GPL, which says in part

> Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program.

Most add-on packages, including all recommended ones, also explicitly allow commercial use in this way. A few packages are restricted to "non-commercial use"; you should contact the author to clarify whether these may be used or seek the advice of your legal counsel.

None of the discussion in this section constitutes legal advice. The R Core Team does not provide legal advice under any circumstances.

2.12 Why is R named R?

The name is partly based on the (first) names of the first two R authors (Robert Gentleman and Ross Ihaka), and partly a play on the name of the Bell Labs language 'S' (see Section 3.1 [What is S?], page 11).

2.13 What is the R Foundation?

The R Foundation is a not for profit organization working in the public interest. It was founded by the members of the R Core Team in order to provide support for the R project and other innovations in statistical computing, provide a reference point for individuals, institutions or commercial enterprises that want to support or interact with the R development community, and to hold and administer the copyright of R software and documentation. See http://www.R-project.org/foundation/ for more information.

3 R and S

3.1 What is S?

S is a very high level language and an environment for data analysis and graphics. In 1998, the Association for Computing Machinery (ACM) presented its Software System Award to John M. Chambers, the principal designer of S, for

> the S system, which has forever altered the way people analyze, visualize, and manipulate data . . .

> S is an elegant, widely accepted, and enduring software system, with conceptual integrity, thanks to the insight, taste, and effort of John Chambers.

The evolution of the S language is characterized by four books by John Chambers and coauthors, which are also the primary references for S.

- Richard A. Becker and John M. Chambers (1984), "S. An Interactive Environment for Data Analysis and Graphics," Monterey: Wadsworth and Brooks/Cole.

  This is also referred to as the "*Brown Book*", and of historical interest only.

- Richard A. Becker, John M. Chambers and Allan R. Wilks (1988), "The New S Language," London: Chapman & Hall.

  This book is often called the "*Blue Book*", and introduced what is now known as S version 2.

- John M. Chambers and Trevor J. Hastie (1992), "Statistical Models in S," London: Chapman & Hall.

  This is also called the "*White Book*", and introduced S version 3, which added structures to facilitate statistical modeling in S.

- John M. Chambers (1998), "Programming with Data," New York: Springer, ISBN 0-387-98503-4 (http://cm.bell-labs.com/cm/ms/departments/sia/Sbook/).

  This "*Green Book*" describes version 4 of S, a major revision of S designed by John Chambers to improve its usefulness at every stage of the programming process.

See http://cm.bell-labs.com/cm/ms/departments/sia/S/history.html for further information on "Stages in the Evolution of S".

There is a huge amount of user-contributed code for S, available at the S Repository at CMU.

3.2 What is S-Plus?

S-Plus is a value-added version of S sold by Insightful Corporation. Based on the S language, S-Plus provides functionality in a wide variety of areas, including robust regression, modern non-parametric regression, time series, survival analysis, multivariate analysis, classical statistical tests, quality control, and graphics drivers. Add-on modules add additional capabilities.

See the Insightful S-Plus page for further information.

3.3 What are the differences between R and S?

We can regard S as a language with three current implementations or "engines", the "old S engine" (S version 3; S-Plus 3.x and 4.x), the "new S engine" (S version 4; S-Plus 5.x and above), and R. Given this understanding, asking for "the differences between R and S" really amounts to asking for the specifics of the R implementation of the S language, i.e., the difference between the R and S *engines*.

For the remainder of this section, "S" refers to the S engines and not the S language.

3.3.1 Lexical scoping

Contrary to other implementations of the S language, R has adopted an evaluation model in which nested function definitions are lexically scoped. This is analogous to the evalutation model in Scheme.

This difference becomes manifest when *free* variables occur in a function. Free variables are those which are neither formal parameters (occurring in the argument list of the function) nor local variables (created by assigning to them in the body of the function). In S, the values of free variables are determined by a set of global variables (similar to C, there is only local and global scope). In R, they are determined by the environment in which the function was created.

Consider the following function:

```
cube <- function(n) {
  sq <- function() n * n
  n * sq()
}
```

Under S, `sq()` does not "know" about the variable `n` unless it is defined globally:

```
S> cube(2)
Error in sq():  Object "n" not found
Dumped
S> n <- 3
S> cube(2)
[1] 18
```

In R, the "environment" created when `cube()` was invoked is also looked in:

```
R> cube(2)
[1] 8
```

As a more "interesting" real-world problem, suppose you want to write a function which returns the density function of the r-th order statistic from a sample of size n from a (continuous) distribution. For simplicity, we shall use both the cdf and pdf of the distribution as explicit arguments. (Example compiled from various postings by Luke Tierney.)

The S-Plus documentation for `call()` basically suggests the following:

```
dorder <- function(n, r, pfun, dfun) {
  f <- function(x) NULL
  con <- round(exp(lgamma(n + 1) - lgamma(r) - lgamma(n - r + 1)))
  PF <- call(substitute(pfun), as.name("x"))
  DF <- call(substitute(dfun), as.name("x"))
  f[[length(f)]] <-
    call("*", con,
         call("*", call("^", PF, r - 1),
              call("*", call("^", call("-", 1, PF), n - r),
                   DF)))
  f
}
```

Rather tricky, isn't it? The code uses the fact that in S, functions are just lists of special mode with the function body as the last argument, and hence does not work in R (one could make the idea work, though).

A version which makes heavy use of `substitute()` and seems to work under both S and R is

```
dorder <- function(n, r, pfun, dfun) {
  con <- round(exp(lgamma(n + 1) - lgamma(r) - lgamma(n - r + 1)))
  eval(substitute(function(x) K * PF(x)^a * (1 - PF(x))^b * DF(x),
                  list(PF = substitute(pfun), DF = substitute(dfun),
                       a = r - 1, b = n - r, K = con)))
}
```

(the `eval()` is not needed in S).

However, in R there is a much easier solution:

```
dorder <- function(n, r, pfun, dfun) {
  con <- round(exp(lgamma(n + 1) - lgamma(r) - lgamma(n - r + 1)))
  function(x) {
    con * pfun(x)^(r - 1) * (1 - pfun(x))^(n - r) * dfun(x)
  }
}
```

This seems to be the "natural" implementation, and it works because the free variables in the returned function can be looked up in the defining environment (this is lexical scope).

Note that what you really need is the function *closure*, i.e., the body along with all variable bindings needed for evaluating it. Since in the above version, the free variables in the value function are not modified, you can actually use it in S as well if you abstract out the closure operation into a function `MC()` (for "make closure"):

```
dorder <- function(n, r, pfun, dfun) {
  con <- round(exp(lgamma(n + 1) - lgamma(r) - lgamma(n - r + 1)))
  MC(function(x) {
       con * pfun(x)^(r - 1) * (1 - pfun(x))^(n - r) * dfun(x)
     },
     list(con = con, pfun = pfun, dfun = dfun, r = r, n = n))
}
```

Given the appropriate definitions of the closure operator, this works in both R and S, and is much "cleaner" than a substitute/eval solution (or one which overrules the default scoping rules by using explicit access to evaluation frames, as is of course possible in both R and S).

For R, `MC()` simply is

```
MC <- function(f, env) f
```

(lexical scope!), a version for S is

```
MC <- function(f, env = NULL) {
  env <- as.list(env)
  if (mode(f) != "function")
    stop(paste("not a function:", f))
  if (length(env) > 0 && any(names(env) == ""))
    stop(paste("not all arguments are named:", env))
  fargs <- if(length(f) > 1) f[1:(length(f) - 1)] else NULL
  fargs <- c(fargs, env)
  if (any(duplicated(names(fargs))))
    stop(paste("duplicated arguments:", paste(names(fargs)),
          collapse = ", "))
  fbody <- f[length(f)]
  cf <- c(fargs, fbody)
  mode(cf) <- "function"
  return(cf)
}
```

Similarly, most optimization (or zero-finding) routines need some arguments to be optimized over and have other parameters that depend on the data but are fixed with respect to optimization. With R scoping rules, this is a trivial problem; simply make up the function with the required definitions in the same environment and scoping takes care of it. With S, one solution is to add an extra parameter to the function and to the optimizer to pass in these extras, which however can only work if the optimizer supports this.

Nested lexically scoped functions allow using function closures and maintaining local state. A simple example (taken from Abelson and Sussman) is obtained by typing `demo("scoping")` at the R prompt. Further information is provided in the standard R reference "R: A Language for Data Analysis and Graphics" (see Section 2.7 [What documentation exists for R?], page 5) and in Robert Gentleman and Ross Ihaka (2000), "Lexical Scope and Statistical Computing", *Journal of Computational and Graphical Statistics*, **9**, 491–508.

Nested lexically scoped functions also imply a further major difference. Whereas S stores all objects as separate files in a directory somewhere (usually '`.Data`' under the current directory), R does not. All objects in R are stored internally. When R is started up it grabs a piece of memory and uses it to store the objects. R performs its own memory management of this piece of memory, growing and shrinking its size as needed. Having everything in memory is necessary because it is not really possible to externally maintain all relevant "environments" of symbol/value pairs. This difference also seems to make R *faster* than S.

The down side is that if R crashes you will lose all the work for the current session. Saving and restoring the memory "images" (the functions and data stored in R's internal memory at any time) can be a bit slow, especially if they are big. In S this does not happen, because everything is saved in disk files and if you crash nothing is likely to happen to them. (In fact, one might conjecture that the S developers felt that the price of changing their approach to persistent storage just to accommodate lexical scope was far too expensive.) Hence, when doing important work, you might consider saving often (see Section 7.2 [How can I save my workspace?], page 71) to safeguard against possible crashes. Other possibilities are logging your sessions, or have your R commands stored in text files which can be read in using `source()`.

> **Note:** If you run R from within Emacs (see Chapter 6 [R and Emacs], page 69), you can save the contents of the interaction buffer to a file and conveniently manipulate it using `ess-transcript-mode`, as well as save source copies of all functions and data used.

3.3.2 Models

There are some differences in the modeling code, such as

- Whereas in S, you would use `lm(y ~ x^3)` to regress `y` on `x^3`, in R, you have to insulate powers of numeric vectors (using `I()`), i.e., you have to use `lm(y ~ I(x^3))`.
- The glm family objects are implemented differently in R and S. The same functionality is available but the components have different names.
- Option `na.action` is set to `"na.omit"` by default in R, but not set in S.
- Terms objects are stored differently. In S a terms object is an expression with attributes, in R it is a formula with attributes. The attributes have the same names but are mostly stored differently.
- Finally, in R `y ~ x + 0` is an alternative to `y ~ x - 1` for specifying a model with no intercept. Models with no parameters at all can be specified by `y ~ 0`.

3.3.3 Others

Apart from lexical scoping and its implications, R follows the S language definition in the Blue and White Books as much as possible, and hence really is an "implementation" of S. There are some intentional differences where the behavior of S is considered "not clean". In general, the rationale is that R should help you detect programming errors, while at the same time being as compatible as possible with S.

Some known differences are the following.

- In R, if `x` is a list, then `x[i] <- NULL` and `x[[i]] <- NULL` remove the specified elements from `x`. The first of these is incompatible with S, where it is a no-op. (Note that you can set elements to `NULL` using `x[i] <- list(NULL)`.)
- In S, the functions named `.First` and `.Last` in the '`.Data`' directory can be used for customizing, as they are executed at the very beginning and end of a session, respectively.

  In R, the startup mechanism is as follows. R first sources the system startup file '`$R_HOME/library/base/R/Rprofile`'. Then, it searches for a site-wide startup profile unless the command line option '`--no-site-file`' was given. The name of this file

is taken from the value of the `R_PROFILE` environment variable. If that variable is unset, the default is '`$R_HOME/etc/Rprofile.site`' ('`$R_HOME/etc/Rprofile`' in versions prior to 1.4.0). This code is loaded in package **base**. Then, unless '`--no-init-file`' was given, R searches for a file called '`.Rprofile`' in the current directory or in the user's home directory (in that order) and sources it into the user workspace. It then loads a saved image of the user workspace from '`.RData`' in case there is one (unless '`--no-restore`' was specified). If needed, the functions `.First()` and `.Last()` should be defined in the appropriate startup profiles.

- In R, `T` and `F` are just variables being set to `TRUE` and `FALSE`, respectively, but are not reserved words as in S and hence can be overwritten by the user. (This helps e.g. when you have factors with levels `"T"` or `"F"`.) Hence, when writing code you should always use `TRUE` and `FALSE`.

- In R, `dyn.load()` can only load *shared objects*, as created for example by `R CMD SHLIB`.

- In R, `attach()` currently only works for lists and data frames, but not for directories. (In fact, `attach()` also works for R data files created with `save()`, which is analogous to attaching directories in S.) Also, you cannot attach at position 1.

- Categories do not exist in R, and never will as they are deprecated now in S. Use factors instead.

- In R, `For()` loops are not necessary and hence not supported.

- In R, `assign()` uses the argument '`envir=`' rather than '`where=`' as in S.

- The random number generators are different, and the seeds have different length.

- R passes integer objects to C as `int *` rather than `long *` as in S.

- R has no single precision storage mode. However, as of version 0.65.1, there is a single precision interface to C/FORTRAN subroutines.

- By default, `ls()` returns the names of the objects in the current (under R) and global (under S) environment, respectively. For example, given

      x <- 1; fun <- function() {y <- 1; ls()}

  then `fun()` returns `"y"` in R and `"x"` (together with the rest of the global environment) in S.

- R allows for zero-extent matrices (and arrays, i.e., some elements of the `dim` attribute vector can be 0). This has been determined a useful feature as it helps reducing the need for special-case tests for empty subsets. For example, if `x` is a matrix, `x[, FALSE]` is not `NULL` but a "matrix" with 0 columns. Hence, such objects need to be tested for by checking whether their `length()` is zero (which works in both R and S), and not using `is.null()`.

- Named vectors are considered vectors in R but not in S (e.g., `is.vector(c(a = 1:3))` returns `FALSE` in S and `TRUE` in R).

- Data frames are not considered as matrices in R (i.e., if `DF` is a data frame, then `is.matrix(DF)` returns `FALSE` in R and `TRUE` in S).

- R by default uses treatment contrasts in the unordered case, whereas S uses the Helmert ones. This is a deliberate difference reflecting the opinion that treatment contrasts are more natural.

- In R, the argument of a replacement function which corresponds to the right hand side must be named 'value'. E.g., `f(a) <- b` is evaluated as `a <- "f<-"(a, value = b)`. S always takes the last argument, irrespective of its name.

- In S, `substitute()` searches for names for substitution in the given expression in three places: the actual and the default arguments of the matching call, and the local frame (in that order). R looks in the local frame only, with the special rule to use a "promise" if a variable is not evaluated. Since the local frame is initialized with the actual arguments or the default expressions, this is usually equivalent to S, until assignment takes place.

- In S, the index variable in a `for()` loop is local to the inside of the loop. In R it is local to the environment where the `for()` statement is executed.

- In S, `tapply(simplify=TRUE)` returns a vector where R returns a one-dimensional array (which can have named dimnames).

- In S(-Plus) the C locale is used, whereas in R the current operating system locale is used for determining which characters are alphanumeric and how they are sorted. This affects the set of valid names for R objects (for example accented chars may be allowed in R) and ordering in sorts and comparisons (such as whether `"aA" < "Bb"` is true or false). From version 1.2.0 the locale can be (re-)set in R by the `Sys.setlocale()` function.

- In S, `missing(arg)` remains `TRUE` if *arg* is subsequently modified; in R it doesn't.

- From R version 1.3.0, `data.frame` strips `I()` when creating (column) names.

- In R, the string `"NA"` is not treated as a missing value in a character variable. Use `as.character(NA)` to create a missing character value.

- R disallows repeated formal arguments in function calls.

- In S, `dump()`, `dput()` and `deparse()` are essentially different interfaces to the same code. In R from version 2.0.0, this is only true if the same `control` argument is used, but by default it is not. By default `dump()` tries to write code that will evaluate to reproduce the object, whereas `dput()` and `deparse()` default to options for producing deparsed code that is readable.

- In R, indexing a vector, matrix, array or data frame with `[` using a character vector index looks only for exact matches (whereas `[[` and `$` allow partial matches). In S, `[` allows partial matches.

- S has a two-argument version of `atan` and no `atan2`. A call in S such as `atan(x1, x2)` is equivalent to R's `atan2(x1, x2)`. However, beware of named arguments since S's `atan(x = a, y = b)` is equivalent to R's `atan2(y = a, x = b)` with the meanings of `x` and `y` interchanged. (R used to have undocumented support for a two-argument `atan` with positional arguments, but this has been withdrawn to avoid further confusion.)

- Numeric constants with no fractional and exponent (i.e., only integer) part are taken as integer in S-Plus 6.x or later, but as double in R.

There are also differences which are not intentional, and result from missing or incorrect code in R. The developers would appreciate hearing about any deficiencies you may find (in a written report fully documenting the difference as you see it). Of course, it would be useful if you were to implement the change yourself and make sure it works.

3.4 Is there anything R can do that S-Plus cannot?

Since almost anything you can do in R has source code that you could port to S-Plus with little effort there will never be much you can do in R that you couldn't do in S-Plus if you wanted to. (Note that using lexical scoping may simplify matters considerably, though.)

R offers several graphics features that S-Plus does not, such as finer handling of line types, more convenient color handling (via palettes), gamma correction for color, and, most importantly, mathematical annotation in plot texts, via input expressions reminiscent of TeX constructs. See the help page for `plotmath`, which features an impressive on-line example. More details can be found in Paul Murrell and Ross Ihaka (2000), "An Approach to Providing Mathematical Annotation in Plots", *Journal of Computational and Graphical Statistics*, **9**, 582–599.

3.5 What is R-plus?

There is no such thing.

4 R Web Interfaces

Rweb is developed and maintained by Jeff Banfield. The Rweb Home Page provides access to all three versions of Rweb—a simple text entry form that returns output and graphs, a more sophisticated Javascript version that provides a multiple window environment, and a set of point and click modules that are useful for introductory statistics courses and require no knowledge of the R language. All of the Rweb versions can analyze Web accessible datasets if a URL is provided.

The paper "Rweb: Web-based Statistical Analysis", providing a detailed explanation of the different versions of Rweb and an overview of how Rweb works, was published in the Journal of Statistical Software (`http://www.jstatsoft.org/v04/i01/`).

Ulf Bartel has developed **R-Online**, a simple on-line programming environment for R which intends to make the first steps in statistical programming with R (especially with time series) as easy as possible. There is no need for a local installation since the only requirement for the user is a JavaScript capable browser. See `http://osvisions.com/r-online/` for more information.

Rcgi is a CGI WWW interface to R by MJ Ray. It had the ability to use "embedded code": you could mix user input and code, allowing the HTML author to do anything from load in data sets to enter most of the commands for users without writing CGI scripts. Graphical output was possible in PostScript or GIF formats and the executed code was presented to the user for revision. However, it is not clear if the project is still active. Currently, a modified version of **Rcgi** by Mai Zhou (actually, two versions: one with (bitmap) graphics and one without) as well as the original code are available from `http://www.ms.uky.edu/~statweb/`.

CGI-based web access to R is also provided at `http://hermes.sdu.dk/cgi-bin/go/`. There are many additional examples of web interfaces to R which basically allow to submit R code to a remote server, see for example the collection of links available from `http://biostat.mc.vanderbilt.edu/twiki/bin/view/Main/StatCompCourse`.

David Firth has written **CGIwithR**, an R add-on package available from CRAN. It provides some simple extensions to R to facilitate running R scripts through the CGI interface to a web server, and allows submission of data using both GET and POST methods. It is easily installed using Apache under Linux and in principle should run on any platform that supports R and a web server provided that the installer has the necessary security permissions. David's paper "CGIwithR: Facilities for Processing Web Forms Using R" was published in the Journal of Statistical Software (`http://www.jstatsoft.org/v08/i10/`). The package is now maintained by Duncan Temple Lang and has a web page at `http://www.omegahat.org/CGIwithR/`.

Rpad, developed and actively maintained by Tom Short, provides a sophisticated environment which combines some of the features of the previous approaches with quite a bit of Javascript, allowing for a GUI-like behavior (with sortable tables, clickable graphics, editable output), etc.

Jeff Horner is working on the R/Apache Integration Project which embeds the R interpreter inside Apache 2 (and beyond). A tutorial and presentation are available from the project web page at `http://biostat.mc.vanderbilt.edu/twiki/bin/view/Main/RApacheProject`.

Rserve is a project actively developed by Simon Urbanek. It implements a TCP/IP server which allows other programs to use facilities of R. Clients are available from the web site for Java and C++ (and could be written for other languages that support TCP/IP sockets).

OpenStatServer is being developed by a team lead by Greg Warnes; it aims "to provide clean access to computational modules defined in a variety of computational environments (R, SAS, Matlab, etc) via a single well-defined client interface" and to turn computational services into web services.

Two projects use PHP to provide a web interface to R. R_PHP_Online by Steve Chen (though it is unclear if this project is still active) is somewhat similar to the above Rcgi and Rweb. R-php is actively developed by Alfredo Pontillo and Angelo Mineo and provides both a web interface to R and a set of pre-specified analyses that need no R code input.

Finally, webbioc is "an integrated web interface for doing microarray analysis using several of the Bioconductor packages" and is designed to be installed at local sites as a shared computing resource.

5 R Add-On Packages

5.1 Which add-on packages exist for R?

5.1.1 Add-on packages in R

The R distribution comes with the following packages:

base   Base R functions (and datasets before R 2.0.0).

datasets   Base R datasets (added in R 2.0.0).

grDevices   Graphics devices for base and grid graphics (added in R 2.0.0).

graphics   R functions for base graphics.

grid   A rewrite of the graphics layout capabilities, plus some support for interaction.

methods   Formally defined methods and classes for R objects, plus other programming tools, as described in the Green Book.

splines   Regression spline functions and classes.

stats   R statistical functions.

stats4   Statistical functions using S4 classes.

tcltk   Interface and language bindings to Tcl/Tk GUI elements.

tools   Tools for package development and administration.

utils   R utility functions.

These "base packages" were substantially reorganized in R 1.9.0. The former **base** was split into the four packages **base**, **graphics**, **stats**, and **utils**. Packages **ctest**, **eda**, **modreg**, **mva**, **nls**, **stepfun** and **ts** were merged into **stats**, package **lqs** returned to the recommended package **MASS**, and package **mle** moved to **stats4**.

5.1.2 Add-on packages from CRAN

The following packages are available from the CRAN 'src/contrib' area. (Packages denoted as *Recommended* are to be included in all binary distributions of R.)

AMORE   A MORE flexible neural network package, providing the TAO robust neural network algorithm.

AlgDesign   Algorithmic experimental designs. Calculates exact and approximate theory experimental designs for D, A, and I criteria.

AnalyzeFMRI
                Functions for I/O, visualisation and analysis of functional Magnetic Resonance Imaging (fMRI) datasets stored in the ANALYZE format.

ArDec   Time series autoregressive decomposition.

BACCO   Bayesian Analysis of Computer Code Output. Contains **approximator**, **calibrator**, and **emulator**, for Bayesian prediction of complex computer codes, calibration of computer models, and emulation of computer programs, respectively.

BHH2   Functions and data sets reproducing some examples in "Statistics for Experimenters II" by G. E. P. Box, J. S. Hunter, and W. C. Hunter, 2005, John Wiley and Sons.

BMA   Bayesian Model Averaging for linear models, generalizable linear models and survival models (Cox regression).

BRugs   OpenBUGS and its R interface BRugs.

BayesTree   Bayesian methods for tree based models.

BayesValidate
Bayesian software validation using posterior quantiles.

Bhat   Functions for general likelihood exploration (MLE, MCMC, CIs).

Biodem   A number of functions for biodemographycal analysis.

Bolstad   Functions and data sets for the book "Introduction to Bayesian Statistics" by W. M. Bolstad, 2004, John Wiley and Sons.

BradleyTerry
Specify and fit the Bradley-Terry model and structured versions.

BSDA   Data sets for the book "Basic Statistics and Data Analysis" by L. J. Kitchens, 2003, Duxbury.

BsMD   Bayes screening and model discrimination follow-up designs.

CDNmoney
Components of Canadian monetary aggregates.

CGIwithR   Facilities for the use of R to write CGI scripts.

CTFS   The CTFS large plot forest dynamics analyses.

CVThresh   Level-dependent Cross-Validation Thresholding.

CircStats   Circular Statistics, from "Topics in Circular Statistics" by S. Rao Jammalamadaka and A. SenGupta, 2001, World Scientific.

CoCo   Graphical modeling for contingency tables using CoCo.

DAAG   Various data sets used in examples and exercises in "Data Analysis and Graphics Using R" by John H. Maindonald and W. John Brown, 2003.

DBI   A common database interface (DBI) class and method definitions. All classes in this package are virtual and need to be extended by the various DBMS implementations.

DCluster   A set of functions for the detection of spatial clusters of diseases using count data.

DDHFm   Variance stabilization by Data-Driven Haar-Fisz (for microarrays).

DEoptim   Differential Evolution Optimization.

DICOM   Import and manipulate medical imaging data using the Digital Imaging and Communications in Medicine (DICOM) Standard.

DPpackage
Semiparametric Bayesian analysis using Dirichlet process priors.

Davies   Functions for the Davies quantile function and the Generalized Lambda distribution.

DescribeDisplay
R interface to the DescribeDisplay GGobi plugin.

Design   Regression modeling, testing, estimation, validation, graphics, prediction, and typesetting by storing enhanced model design attributes in the fit. Design is a collection of about 180 functions that assist and streamline modeling, especially for biostatistical and epidemiologic applications. It also contains new functions for binary and ordinal logistic regression models and the Buckley-James multiple regression model for right-censored responses, and implements penalized maximum likelihood estimation for logistic and ordinary linear models. Design works with almost any regression model, but it was especially written to work with logistic regression, Cox regression, accelerated failure time models, ordinary linear models, and the Buckley-James model.

Devore5   Data sets and sample analyses from "Probability and Statistics for Engineering and the Sciences (5th ed)" by Jay L. Devore, 2000, Duxbury.

Devore6   Data sets and sample analyses from "Probability and Statistics for Engineering and the Sciences (6th ed)" by Jay L. Devore, 2003, Duxbury.

EMV   Estimation of missing values in a matrix by a k-th nearest neighboors algorithm.

EbayesThresh
Empirical Bayes thresholding and related methods.

Ecdat   Data sets from econometrics textbooks.

ElemStatLearn
Data sets, functions and examples from the book "The Elements of Statistical Learning: Data Mining, Inference, and Prediction" by Trevor Hastie, Robert Tibshirani and Jerome Friedman (2001), Springer.

Epi   Statistical analysis in epidemiology, with functions for demographic and epidemiological analysis in the Lexis diagram.

FLCore   Core package of FLR, fisheries modeling in R.

FactoMineR
Factor analysis and data mining with R.

Fahrmeir   Data from the book "Multivariate Statistical Modelling Based on Generalized Linear Models" by Ludwig Fahrmeir and Gerhard Tutz (1994), Springer.

FortranCallsR
Simple Fortran/C/R interface example.

FracSim   Simulation of one- and two-dimensional fractional and multifractional Levy motions.

FunCluster
Functional profiling of cDNA microarray expression data.

GDD Platform and X11 independent device for creating bitmaps (png, gif and jpeg) using the GD graphics library.

GPArotation
Gradient Projection Algorithm rotation for factor analysis.

GRASS An interface between the GRASS geographical information system and R, based on starting R from within the GRASS environment and chosen LOCA-TION_NAME and MAPSET. Wrapper and helper functions are provided for a range of R functions to match the interface metadata structures.

GammaTest
Gamma Test data analysis.

GenKern Functions for generating and manipulating generalised binned kernel density estimates.

GeneNT Relevance or Dependency network and signaling pathway discovery.

GeneTS A package for analysing multiple gene expression time series data. Currently, implements methods for cell cycle analysis and for inferring large sparse graphical Gaussian models.

Geneland MCMC inference from individual genetic data based on a spatial statistical model.

GroupSeq Computations related to group-seqential boundaries.

HI Simulation from distributions supported by nested hyperplanes.

HSAUR Functions, data sets, analyses and examples from the book "A Handbook of Statistical Analyses Using R" by Brian S. Everitt and Torsten Hothorn (2006), Chapman & Hall/CRC.

HTMLapplets
Functions inserting dynamic scatterplots and grids in documents generated by **R2HTML**.

HighProbability
Estimation of the alternative hypotheses having frequentist or Bayesian probabilities at least as great as a specified threshold, given a list of p-values.

Hmisc Functions useful for data analysis, high-level graphics, utility operations, functions for computing sample size and power, importing datasets, imputing missing values, advanced table making, variable clustering, character string manipulation, conversion of S objects to LaTeX code, recoding variables, and bootstrap repeated measures analysis.

HyperbolicDist
Basic functions for the hyperbolic distribution: probability density function, distribution function, quantile function, a routine for generating observations from the hyperbolic, and a function for fitting the hyperbolic distribution to data.

ICE Iterated Conditional Expectation: kernel estimators for interval-censored data.

IDPmisc   Utilities from the Institute of Data Analyses and Process Design, IDP/ZHW.

ISwR   Data sets for "Introductory Statistics with R" by Peter Dalgaard, 2002, Springer.

Icens   Functions for computing the NPMLE for censored and truncated data.

JLLprod   Nonparametric estimation of homothetic and generalized homothetic production functions.

JGR   Java Gui for R.

JavaGD   Java Graphics Device.

JointGLM   Joint modeling of mean and dispersion through two interlinked GLM's.

KMsurv   Data sets and functions for "Survival Analysis, Techniques for Censored and Truncated Data" by Klein and Moeschberger, 1997, Springer.

Kendall   Kendall rank correlation and Mann-Kendall trend test.

KernSmooth
Functions for kernel smoothing (and density estimation) corresponding to the book "Kernel Smoothing" by M. P. Wand and M. C. Jones, 1995. *Recommended.*

LDheatmap
Heat maps of linkage disequilibrium measures.

LMGene   Date transformation and identification of differentially expressed genes in gene expression arrays.

Lmoments   Estimation of L-moments and the parameters of normal and Cauchy polynomial quantile mixtures.

LogicReg   Routines for Logic Regression.

LoopAnalyst
A collection of tools to conduct Levins' Loop Analysis.

LowRankQP
Low Rank Quadratic Programming: QP problems where the hessian is represented as the product of two matrices.

MASS   Functions and datasets from the main package of Venables and Ripley, "Modern Applied Statistics with S". Contained in the 'VR' bundle. *Recommended.*

MBESS   Methods for the Behavioral, Educational, and Social Sciences.

MCMCpack
Markov chain Monte Carlo (MCMC) package: functions for posterior simulation for a number of statistical models.

MEMSS   Data sets and sample analyses from "Mixed-effects Models in S and S-PLUS" by J. Pinheiro and D. Bates, 2000, Springer.

MNP   Fitting Bayesian Multinomial Probit models via Markov chain Monte Carlo. Along with the standard Multinomial Probit model, it can also fit models with different choice sets for each observation and complete or partial ordering of all the available alternatives.

MPV   Data sets from the book "Introduction to Linear Regression Analysis" by D. C. Montgomery, E. A. Peck, and C. G. Vining, 2001, John Wiley and Sons.

MSBVAR   Bayesian vector autoregression models, impulse responses and forecasting.

MarkedPointProcess
Non-parametric analysis of the marks of marked point processes.

MatchIt   Select matched samples of the original treated and control groups with similar covariate distributions.

Matching   Multivariate and propensity score matching with formal tests of balance.

Matrix   A Matrix package.

NADA   Methods described in "Nondetects And Data Analysis: Statistics for Censored Environmental Data" by Dennis R. Helsel, 2004, John Wiley and Sons.

NISTnls   A set of test nonlinear least squares examples from NIST, the U.S. National Institute for Standards and Technology.

NORMT3   Evaluates complex erf, erfc and density of sum of Gaussian and Student's t.

Oarray   Arrays with arbitrary offsets.

PBSmapping
Software evolved from fisheries research conducted at the Pacific Biological Station (PBS) in Nanaimo, British Columbia, Canada. Draws maps and implements other GIS procedures.

PHYLOGR
Manipulation and analysis of phylogenetically simulated data sets (as obtained from PDSIMUL in package PDAP) and phylogenetically-based analyses using GLS.

PK   Estimation of pharmacokinetic parameters.

PKfit   A nonlinear regression (including a genetic algorithm) program designed to deal with curve fitting for pharmacokinetics.

POT   Generalized Pareto distribution and Peaks Over Threshold.

PTAk   A multiway method to decompose a tensor (array) of any order, as a generalisation of SVD also supporting non-identity metrics and penalisations. Also includes some other multiway methods.

ProbForecastGOP
Probabilistic weather field forecasts using the Geostatistical Output Perturbation method introduced by Gel, Raftery and Gneiting (2004).

QCA   Qualitative Comparative Analysis for crisp sets.

R.matlab   Read and write of MAT files together with R-to-Matlab connectivity.

R.oo   R object-oriented programming with or without references.

R.utils   Utility classes and methods useful when programming in R and developing R packages.

R2HTML   Functions for exporting R objects & graphics in an HTML document.

R2WinBUGS

Running WinBUGS from R: call a BUGS model, summarize inferences and convergence in a table and graph, and save the simulations in arrays for easy access in R.

RArcInfo   Functions to import Arc/Info V7.x coverages and data.

RBloomberg

Fetch data from a Bloomberg API using COM.

RColorBrewer

ColorBrewer palettes for drawing nice maps shaded according to a variable.

RFA   Regional Frequency Analysis.

RGrace   Mouse/menu driven interactive plotting application.

RGraphics

Data and functions from the book "R Graphics" by Paul Murrell, 2005, Chapman & Hall/CRC.

RII   Estimation of the relative index of inequality for interval-censored data using natural cubic splines.

RLMM   A genotype calling algorithm for Affymetrix SNP arrays.

RMySQL   An interface between R and the MySQL database system.

RNetCDF   An interface to Unidata's NetCDF library functions (version 3) and furthermore access to Unidata's udunits calendar conversions.

ROCR   Visualizing the performance of scoring classifiers.

RODBC   An ODBC database interface.

ROracle   Oracle Database Interface driver for R. Uses the ProC/C++ embedded SQL.

RQuantLib

Provides access to (some) of the QuantLib functions from within R; currently limited to some Option pricing and analysis functions. The QuantLib project aims to provide a comprehensive software framework for quantitative finance.

RSQLite   Database Interface R driver for SQLite. Embeds the SQLite database engine in R.

RScaLAPACK

An interface to ScaLAPACK functions from R.

RSvgDevice

A graphics device for R that uses the new w3.org XML standard for Scalable Vector Graphics.

RUnit   Functions implementing a standard Unit Testing framework, with additional code inspection and report generation tools.

RWeka   An R interface to Weka, a rich collection of machine learning algorithms for data mining tasks.

RWinEdt   A plug in for using WinEdt as an editor for R.

RXshrink   Maximum Likelihood Shrinkage via Ridge or Least Angle Regression.

RadioSonde
A collection of programs for reading and plotting SKEW-T,log p diagrams and wind profiles for data collected by radiosondes (the typical weather balloon-borne instrument).

RandVar   Implementation of random variables by means of S4 classes and methods.

RandomFields
Creating random fields using various methods.

Rcmdr   A platform-independent basic-statistics GUI (graphical user interface) for R, based on the **tcltk** package.

RcppTemplate
An illustration of the use of the Rcpp R/C++ interface library.

ResistorArray
Electrical properties of resistor networks.

Rfwdmv   Forward Search for Multivariate Data.

Rlab   Functions and data sets for the NCSU ST370 class.

Rlsf   Interface to the LSF queuing system.

Rmdr   R-Multifactor Dimensionality Reduction.

Rmpi   An interface (wrapper) to MPI (Message-Passing Interface) APIs. It also provides an interactive R slave environment in which distributed statistical computing can be carried out.

Rpad   Utility functions for the Rpad workbook-style interface.

Rwave   An environment for the time-frequency analysis of 1-D signals (and especially for the wavelet and Gabor transforms of noisy signals), based on the book "Practical Time-Frequency Analysis: Gabor and Wavelet Transforms with an Implementation in S" by Rene Carmona, Wen L. Hwang and Bruno Torresani, 1998, Academic Press.

SAGx   Retrieval, preparation and analysis of data from the Affymetrix GeneChip.

SASmixed   Data sets and sample linear mixed effects analyses corresponding to the examples in "SAS System for Mixed Models" by R. C. Littell, G. A. Milliken, W. W. Stroup and R. D. Wolfinger, 1996, SAS Institute.

SIN   A SINful approach to selection of Gaussian Graphical Markov Models.

SciViews   A bundle of packages to implement a full reusable GUI API for R. Contains **svGUI** with the main GUI features, **svDialogs** for the dialog boxes, **svIO** for data import/export, **svMisc** with miscellaneous supporting functions, and **svViews** providing views and report features (views are HTML presentations of the content of R objects, combining text, tables and graphs in the same document).

SemiPar   Functions for semiparametric regression analysis, to complement the book "Semiparametric Regression" by R. Ruppert, M. P. Wand, and R. J. Carroll, 2003, Cambridge University Press.

SenSrivastava
          Collection of datasets from "Regression Analysis, Theory, Methods and Applications" by A. Sen and M. Srivastava, 1990, Springer.

SensoMineR
          Sensory data analysis.

SeqKnn   Sequential KNN imputation.

SharedHT2
          Shared Hotelling T^2 test for small sample microarray experiments.

SoPhy   Soil Physics Tools: simulation of water flux and solute transport in soil.

SparseLogReg
          Sparse logistic regression.

SparseM   Basic linear algebra for sparse matrices.

StatDataML
          Read and write StatDataML.

StoppingRules
          Stopping rules for microarray classifiers.

SuppDists   Ten distributions supplementing those built into R (Inverse Gauss, Kruskal-Wallis, Kendall's Tau, Friedman's chi squared, Spearman's rho, maximum F ratio, the Pearson product moment correlation coefficient, Johnson distributions, normal scores and generalized hypergeometric distributions).

SwissAir   Air quality data of Switzerland for one year in 30 min resolution.

TeachingDemos
          A set of demonstration functions that can be used in a classroom to demonstrate statistical concepts, or on your own to better understand the concepts or the programming.

UNF   Tools for creating universal numeric fingerprints for data.

USPS   Unsupervised and Supervised methods of Propensity Score adjustment for bias.

UsingR   Data sets to accompany the textbook "Using R for Introductory Statistics" by J. Verzani, 2005, Chapman & Hall/CRC.

VDCutil   Utilities supporting VDC, an open source digital library system for quantitative data.

VLMC   Functions, classes & methods for estimation, prediction, and simulation (bootstrap) of VLMC (Variable Length Markov Chain) models.

VaR   Methods for calculation of Value at Risk (VaR).

WhatIf   Software for evaluating counterfactuals.

XML   Tools for reading XML documents and DTDs.

Zelig   Everyone's statistical software: an easy-to-use program that can estimate, and help interpret the results of, an enormous range of statistical models.

aaMI   Mutual information for protein sequence alignments.

abind   Combine multi-dimensional arrays.

accuracy   A suite of tools designed to test and improve the accuracy of statistical computation.

acepack   ACE (Alternating Conditional Expectations) and AVAS (Additivity and VAriance Stabilization for regression) methods for selecting regression transformations.

actuar   Functions related to actuarial science applications.

adapt   Adaptive quadrature in up to 20 dimensions.

ade4   Multivariate data analysis and graphical display.

adehabitat
               A collection of tools for the analysis of habitat selection by animals.

adlift   Adaptive Wavelet transforms for signal denoising.

agce   Analysis of growth curve experiments.

akima   Linear or cubic spline interpolation for irregularly gridded data.

allelic   A fast, unbiased and exact allelic exact test.

alr3   Methods and data to accompany the textbook "Applied Linear Regression" by S. Weisberg, 2005, Wiley.

amap   Another Multidimensional Analysis Package.

aod   Analysis of Overdispersed Data.

apTreeshape
               Analyses of phylogenetic treeshape.

ape   Analyses of Phylogenetics and Evolution, providing functions for reading and plotting phylogenetic trees in parenthetic format (standard Newick format), analyses of comparative data in a phylogenetic framework, analyses of diversification and macroevolution, computing distances from allelic and nucleotide data, reading nucleotide sequences from GenBank via internet, and several tools such as Mantel's test, computation of minimum spanning tree, or the population parameter theta based on various approaches.

aplpack   Another PLot PACKage: stem.leaf, bagplot, faces, spin3R, ...

arules   Mining association rules and frequent itemsets with R.

ash   David Scott's ASH routines for 1D and 2D density estimation.

assist   A suite of functions implementing smoothing splines.

aster   Functions and datasets for Aster modeling (forest graph exponential family conditional or unconditional canonical statistic models for life history trait modeling).

asypow   A set of routines that calculate power and related quantities utilizing asymptotic likelihood ratio methods.

aws   Functions to perform adaptive weights smoothing.

bayesSurv Bayesian survival regression with flexible error and (later on also random effects) distributions.

bayesm   Bayes Inference for Marketing/Micro-econometrics.

bayesmix   Bayesian mixture models of univariate Gaussian distributions using JAGS.

betareg   Beta regression for modeling rates and proportions.

bicreduc   Reduction algorithm for the NPMLE for the distribution function of bivariate interval-censored data.

biglm   Linear regression for data too large to fit in memory.

bim   Bayesian interval mapping diagnostics: functions to interpret QTLCart and Bmapqtl samples.

bindata   Generation of correlated artificial binary data.

biopara   Self-contained parallel system for R.

bitops   Functions for Bitwise operations on integer vectors.

bivpois   Bivariate Poisson models using the EM algorithm.

blighty   Function for drawing the coastline of the United Kingdom.

blockrand Randomization for block random clinical trials.

boa   Bayesian Output Analysis Program for MCMC.

boolean   Boolean logit and probit: a procedure for testing Boolean hypotheses.

boost   Boosting methods for real and simulated data, featuring 'BagBoost', 'LogitBoost', 'AdaBoost', and 'L2Boost'.

boot   Functions and datasets for bootstrapping from the book "Bootstrap Methods and Their Applications" by A. C. Davison and D. V. Hinkley, 1997, Cambridge University Press. *Recommended.*

bootstrap Software (bootstrap, cross-validation, jackknife), data and errata for the book "An Introduction to the Bootstrap" by B. Efron and R. Tibshirani, 1993, Chapman and Hall.

bqtl   QTL mapping toolkit for inbred crosses and recombinant inbred lines. Includes maximum likelihood and Bayesian tools.

brlr   Bias-reduced logistic regression: fits logistic regression models by maximum penalized likelihood.

butler   Unit testing, profiling and benchmarking for R.

caMassClass
Processing and Classification of protein mass spectra (SELDI) data.

caTools Miscellaneous utility functions, including reading/writing ENVI binary files, a LogitBoost classifier, and a base64 encoder/decoder.

calibrate Calibration of biplot axes.

car Companion to Applied Regression, containing functions for applied regession, linear models, and generalized linear models, with an emphasis on regression diagnostics, particularly graphical diagnostic methods.

cat Analysis of categorical-variable datasets with missing values.

catspec Special models for categorical variables.

cba Clustering for Business Analytics, including implementations of Proximus and Rock.

cclust Convex clustering methods, including k-means algorithm, on-line update algorithm (Hard Competitive Learning) and Neural Gas algorithm (Soft Competitive Learning) and calculation of several indexes for finding the number of clusters in a data set.

cfa Analysis of configuration frequencies.

chplot Augmented convex hull plots: informative and nice plots for grouped bivariate data.

changeLOS
Change in length of hospital stay (LOS).

chron A package for working with chronological objects (times and dates).

circular Circular statistics, from "Topics in Circular Statistics" by Rao Jammalamadaka and A. SenGupta, 2001, World Scientific.

clac Clust Along Chromosomes, a method to call gains/losses in CGH array data.

class Functions for classification (k-nearest neighbor and LVQ). Contained in the 'VR' bundle. *Recommended.*

classInt Choose univariate class intervals for mapping or other graphics purposes.

classPP Projection Pursuit for supervised classification.

clim.pact Climate analysis and downscaling for monthly and daily data.

climatol Functions to fill missing data in climatological (monthly) series and to test their homogeneity, plus functions to draw wind-rose and Walter&Lieth diagrams.

clines Calculates Contour Lines.

clue CLUster Ensembles.

cluster Functions for cluster analysis. *Recommended.*

clusterRepro
Reproducibility of gene expression clusters.

clustvarsel Variable selection for model-based clustering.

cmprsk Estimation, testing and regression modeling of subdistribution functions in competing risks.

cobs Constrained B-splines: qualitatively constrained (regression) smoothing via linear programming.

cocorresp Co-correspondence analysis ordination methods for community ecology.

coda Output analysis and diagnostics for Markov Chain Monte Carlo (MCMC) simulations.

coin COnditional INference procedures for the general independence problem including two-sample, K-sample, correlation, censored, ordered and multivariate problems.

colorspace Mapping between assorted color spaces.

combinat Combinatorics utilities.

compositions
Functions for the consistent analysis of compositional data (e.g., portions of substances) and positive numbers (e.g., concentrations).

concor Concordance, providing "SVD by blocks".

concord Measures of concordance and reliability.

conf.design
A series of simple tools for constructing and manipulating confounded and fractional factorial designs.

connectedness
Find disconnected sets for two-way classification.

copula Classes of commonly used copulas (including elliptical and Archimedian), and methods for density, distribution, random number generators, and plotting.

corpora Utility functions for the statistical analysis of corpus frequency data.

corpcor Efficient estimation of covariance and (partial) correlation.

covRobust Robust covariance estimation via nearest neighbor cleaning.

coxrobust Robust Estimation in the Cox proportional hazards regression model.

cramer Routine for the multivariate nonparametric Cramer test.

crossdes Functions for the construction and randomization of balanced carryover balanced designs, to check given designs for balance, and for simulation studies on the validity of two randomization procedures.

crq Quantile regression for randomly censored data.

cslogistic Likelihood and posterior analysis of conditionally specified logistic regression models.

ctv Server-side and client-side tools for CRAN task views.

cwhmisc   Miscellaneous functions by Christian W. Hoffmann.

cyclones   Cyclone identification.

date   Functions for dealing with dates. The most useful of them accepts a vector of input dates in any of the forms '8/30/53', '30Aug53', '30 August 1953', ..., 'August 30 53', or any mixture of these.

dblcens   Calculates the NPMLE of the survival distribution for doubly censored data.

deal   Bayesian networks with continuous and/or discrete variables can be learned and compared from data.

debug   Debugger for R functions, with code display, graceful error recovery, line-numbered conditional breakpoints, access to exit code, flow control, and full keyboard input.

deldir   Calculates the Delaunay triangulation and the Dirichlet or Voronoi tesselation (with respect to the entire plane) of a planar point set.

delt   Estimation of multivariate densities with adaptive histograms.

denpro   Visualization of multivariate density functions and estimates with level set trees and shape trees, and visualization of multivariate data with tail trees.

depmix   Dependent Mixture Models: fit (multi-group) mixtures of latent Markov models on mixed categorical and continuous (time series) data.

dglm   Double generalized linear models.

diamonds   Functions for illustrating aperture-4 diamond partitions in the plane, or on the surface of an octahedron or icosahedron, for use as analysis or sampling grids.

dichromat   Color schemes for dichromats: collapse red-green distinctions to simulate the effects of colour-blindness.

digest   Two functions for the creation of "hash" digests of arbitrary R objects using the md5 and sha-1 algorithms permitting easy comparison of R language objects.

diptest   Compute Hartigan's dip test statistic for unimodality.

dispmod   Functions for modelling dispersion in GLMs.

distr   An object orientated implementation of distributions and some additional functionality.

distrEx   Extensions of package **distr**.

distrSim   Simulation classes based on package **distr**.

distrTEst   Estimation and Testing classes based on package **distr**.

diveMove   Dive analysis and calibration.

doBy   Facilities for groupwise computations.

dprep   Data preprocessing and visualization functions for classification.

dr   Functions, methods, and datasets for fitting dimension reduction regression, including pHd and inverse regression methods SIR and SAVE.

drc   Non-linear regression analysis for multiple curves with focus on concentration-response, dose-response and time-response curves.

drfit   Dose-response data evaluation.

dse   Dynamic System Estimation, a multivariate time series package bundle. Contains **dse1** (the base system, including multivariate ARMA and state space models), **dse2** (extensions for evaluating estimation techniques, forecasting, and for evaluating forecasting model), **tframe** (functions for writing code that is independent of the representation of time), and **setRNG** (a mechanism for generating the same random numbers in S and R).

dyn   Time series regression.

dynamicGraph
                 Interactive graphical tool for manipulating graphs.

dynlm   Dynamic linear models and time series regression.

e1071   Miscellaneous functions used at the Department of Statistics at TU Wien (E1071), including moments, short-time Fourier transforms, Independent Component Analysis, Latent Class Analysis, support vector machines, and fuzzy clustering, shortest path computation, bagged clustering, and some more.

eba   Fitting and testing probabilistic choice models, especially the BTL, elimination-by-aspects (EBA), and preference tree (Pretree) models.

eco   Fitting Bayesian models of ecological inference in 2 by 2 tables.

ecodist   Dissimilarity-based functions for ecological analysis.

edci   Edge Detection and Clustering in Images.

effects   Graphical and tabular effect displays, e.g., of interactions, for linear and generalised linear models.

eha   A package for survival and event history analysis.

elasticnet   Elastic net regularization and variable selection.

ellipse   Package for drawing ellipses and ellipse-like confidence regions.

elliptic   A suite of elliptic and related functions including Weierstrass and Jacobi forms.

emme2   Functions to read from and write to an EMME/2 databank.

emplik   Empirical likelihood ratio for means/quantiles/hazards from possibly right censored data.

energy   E-statistics (energy) tests for comparing distributions: multivariate normality, Poisson test, multivariate k-sample test for equal distributions, hierarchical clustering by e-distances.

ensembleBMA
                 Probabilistic forecasting using Bayesian Model Averaging of ensembles using a mixture of normal distributions.

epitools   Basic tools for applied epidemiology.

epsi   Edge Preserving Smoothing for Images.

equivalence
          Tests and graphics for assessing tests of equivalence.

evd   Functions for extreme value distributions. Extends simulation, distribution, quantile and density functions to univariate, bivariate and (for simulation) multivariate parametric extreme value distributions, and provides fitting functions which calculate maximum likelihood estimates for univariate and bivariate models.

evdbayes   Functions for the bayesian analysis of extreme value models, using MCMC methods.

evir   Extreme Values in R: Functions for extreme value theory, which may be divided into the following groups; exploratory data analysis, block maxima, peaks over thresholds (univariate and bivariate), point processes, gev/gpd distributions.

exactLoglinTest
          Monte Carlo exact tests for log-linear models.

exactRankTests
          Computes exact p-values and quantiles using an implementation of the Streitberg/Roehmel shift algorithm.

exactmaxsel
          Exact methods for maximally selected statistics for binary response variables.

fBasics   The Rmetrics module for "Markets, basic statistics, and stylized facts". Rmetrics is an environment and software collection for teaching financial engineering and computational finance (http://www.Rmetrics.org/).

fCalendar   The Rmetrics module for "Date, Time and Calendars".

fExtremes   The Rmetrics module for "Beyond the Sample, Dealing with Extreme Values".

fMultivar   The Rmetrics module for "Multivariate Data Analysis".

fOptions   The Rmetrics module for "The Valuation of Options".

fPortfolio The Rmetrics module for "Pricing and Hedging of Options".

fSeries   The Rmetrics module for "The Dynamical Process Behind Financial Markets".

far   Modelization for Functional AutoRegressive processes.

faraway   Functions and datasets for books by Julian Faraway.

fastICA   Implementation of FastICA algorithm to perform Independent Component Analysis (ICA) and Projection Pursuit.

fda   Functional Data Analysis: analysis of data where the basic observation is a function of some sort.

fdim   Functions for calculating fractal dimension.

femmeR   Output analysis of FEMME model results.

fgac   Families of Generalized Archimedean Copulas.

fields   A collection of programs for curve and function fitting with an emphasis on spatial data. The major methods implemented include cubic and thin plate splines, universal Kriging and Kriging for large data sets. The main feature is that any covariance function implemented in R can be used for spatial prediction.

filehash   Simple file-based hash table.

financial   Solving financial problems in R.

flexclust   Flexible cluster algorithms.

flexmix   Flexible Mixture Modeling: a general framework for finite mixtures of regression models using the EM algorithm.

foreign   Functions for reading and writing data stored by statistical software like Minitab, S, SAS, SPSS, Stata, Systat, etc. *Recommended.*

fork   Functions for handling multiple processes: simple wrappers around the Unix process management API calls.

fortunes   R fortunes.

forward   Forward search approach to robust analysis in linear and generalized linear regression models.

fpc   Fixed point clusters, clusterwise regression and discriminant plots.

fracdiff   Maximum likelihood estimation of the parameters of a fractionally differenced ARIMA(p, d, q) model (Haslett and Raftery, Applied Statistics, 1989).

frailtypack
                 Fit a shared gamma frailty model and Cox proportional hazards model using a Penalized Likelihood on the hazard function.

ftnonpar   Features and strings for nonparametric regression.

fuzzyRankTests
                 Fuzzy rank tests and confidence intervals.

g.data   Create and maintain delayed-data packages (DDP's).

gRbase   A package for graphical modelling in R. Defines S4 classes for graphical meta data and graphical models, and illustrates how hierarchical log-linear models may be implemented and combined with **dynamicGraph**.

gafit   Genetic algorithm for curve fitting.

gam   Functions for fitting and working with Generalized Additive Models, as described in chapter 7 of the White Book, and in "Generalized Additive Models" by T. Hastie and R. Tibshirani (1990).

gamair   Data sets used in the book "Generalized Additive Models: An Introduction with R" by S. Wood (2006).

gamlss   Functions to fit Generalized Additive Models for Location Scale and Shape.

gap   Genetic analysis package for both population and family data.

gbm Generalized Boosted Regression Models: implements extensions to Freund and Schapire's AdaBoost algorithm and J. Friedman's gradient boosting machine. Includes regression methods for least squares, absolute loss, logistic, Poisson, Cox proportional hazards partial likelihood, and AdaBoost exponential loss.

gclus Clustering Graphics. Orders panels in scatterplot matrices and parallel coordinate displays by some merit index.

gcmrec Parameters estimation of the general semiparametric model for recurrent event data proposed by Peña and Hollander.

gdata Various functions to manipulate data.

gee An implementation of the Liang/Zeger generalized estimating equation approach to GLMs for dependent data.

geepack Generalized estimating equations solver for parameters in mean, scale, and correlation structures, through mean link, scale link, and correlation link. Can also handle clustered categorical responses.

genalg R based genetic algorithm for binary and floating point chromosomes.

genetics Classes and methods for handling genetic data. Includes classes to represent genotypes and haplotypes at single markers up to multiple markers on multiple chromosomes, and functions for allele frequencies, flagging homo/heterozygotes, flagging carriers of certain alleles, computing disequlibrium, testing Hardy-Weinberg equilibrium, . . .

geoR Functions to perform geostatistical data analysis including model-based methods.

geoRglm Functions for inference in generalised linear spatial models.

geometry Mesh generation and surface tesselation, based on the Qhull library.

ggm Functions for defining directed acyclic graphs and undirected graphs, finding induced graphs and fitting Gaussian Markov models.

ggplot Grammar of graphics based plots for R.

giRaph Data structures and algorithms for computations on graphs.

gld Basic functions for the generalised (Tukey) lambda distribution.

gllm Routines for log-linear models of incomplete contingency tables, including some latent class models via EM and Fisher scoring approaches.

glmmML A Maximum Likelihood approach to generalized linear models with random intercept.

glmpath L1 regularization path for Generalized Linear Models.

glpk Interface to the GNU Linear Programming Kit (GLPK).

gmodels Various functions to manipulate models.

gmp Arithmetic "without limitations" using the GNU Multiple Precision library.

gmt Interface between the GMT 4.0 map-making software and R.

gnm   Functions to specify and fit generalized nonlinear models, including models with multiplicative interaction terms such as the UNIDIFF model from sociology and the AMMI model from crop science.

gpclib   General polygon clipping routines for R based on Alan Murta's C library.

gpls   Classification using generalized partial least squares for two-group and multi-group (more than 2 group) classification.

gplots   Various functions to draw plots.

grasper   Generalized Regression Analysis and Spatial Predictions for R.

gregmisc   Miscellaneous functions written/maintained by Gregory R. Warnes.

gridBase   Integration of base and grid graphics.

grnnR   A Generalized Regression Neural Network.

grouped   Regression models for grouped and coarse data, under the Coarsened At Random assumption.

gsl   Wrapper for special functions of the Gnu Scientific Library (GSL).

gss   A comprehensive package for structural multivariate function estimation using smoothing splines.

gstat   multivariable geostatistical modelling, prediction and simulation. Includes code for variogram modelling; simple, ordinary and universal point or block (co)kriging, sequential Gaussian or indicator (co)simulation, and map plotting functions.

gsubfn   Miscellaneous string utilities.

gtkDevice   GTK graphics device driver that may be used independently of the R-GNOME interface and can be used to create R devices as embedded components in a GUI using a Gtk drawing area widget, e.g., using RGtk.

gtools   Various functions to help manipulate data.

hapassoc   Likelihood inference of trait associations with SNP haplotypes and other attributes using the EM Algorithm.

haplo.ccs   Estimate haplotype relative risks in case-control data.

haplo.stats
                Statistical analysis of haplotypes with traits and covariates when linkage phase is ambiguous.

hapsim   Haplotype data simulation.

hddplot   Use known groups in high-dimensional data to derive scores for plots.

hdf5   Interface to the NCSA HDF5 library.

hdrcde   Highest Density Regions and Conditional Density Estimation.

hett   Functions for the fitting and summarizing of heteroscedastic t-regression.

hier.part   Hierarchical Partitioning: variance partition of a multivariate data set.

hierfstat   Estimation of hierarchical F-statistics from haploid or diploid genetic data with any numbers of levels in the hierarchy, and tests for the significance of each F and variance components.

hmm.discnp
                Hidden Markov models with discrete non-parametric observation distributions.

hoa   A bundle of packages for higher order likelihood-based inference. Contains **cond** for approximate conditional inference for logistic and loglinear models, **csampling** for conditional simulation in regression-scale models, **marg** for approximate marginal inference for regression-scale models, and **nlreg** for higher order inference for nonlinear heteroscedastic models.

homals   Homogeneity Analysis (HOMALS) package with optional Tcl/Tk interface.

hopach   Hierarchical Ordered Partitioning and Collapsing Hybrid (HOPACH).

howmany   A lower bound for the number of correct rejections.

httpRequest
                Implements HTTP Request protocols (GET, POST, and multipart POST requests).

hwde   Models and tests for departure from Hardy-Weinberg equilibrium and independence between loci.

hybridHclust
                Hybrid hierarchical clustering via mutual clusters.

ifs   Iterated Function Systems distribution function estimator.

igraph   Routines for simple graphs.

iid.test   Testing whether data is independent and identically distributed.

impute   Imputation for microarray data (currently KNN only).

ineq   Inequality, concentration and poverty measures, and Lorenz curves (empirical and theoretic).

intcox   Implementation of the Iterated Convex Minorant Algorithm for the Cox proportional hazard model for interval censored event data.

ipred   Improved predictive models by direct and indirect bootstrap aggregation in classification and regression as well as resampling based estimators of prediction error.

irr   Coefficients of Interrater Reliability and Agreement for quantitative, ordinal and nominal data.

ismev   Functions to support the computations carried out in "An Introduction to Statistical Modeling of Extreme Values;' by S. Coles, 2001, Springer. The functions may be divided into the following groups; maxima/minima, order statistics, peaks over thresholds and point processes.

its   An S4 class for handling irregular time series.

kappalab The "laboratory for capacities", an S4 tool box for capacity (or non-additive measure, fuzzy measure) and integral manipulation on a finite setting.

kernlab Kernel-based machine learning methods including support vector machines.

kinship Mixed-effects Cox models, sparse matrices, and modeling data from large pedigrees.

kknn Weighted k-nearest neighbors classification and regression.

klaR Miscellaneous functions for classification and visualization developed at the Department of Statistics, University of Dortmund.

km.ci Confidence intervals for the Kaplan-Meier estimator.

knnFinder Fast nearest neighbor search.

knnTree Construct or predict with k-nearest-neighbor classifiers, using cross-validation to select k, choose variables (by forward or backwards selection), and choose scaling (from among no scaling, scaling each column by its SD, or scaling each column by its MAD). The finished classifier will consist of a classification tree with one such k-nn classifier in each leaf.

knncat Nearest-neighbor classification with categorical variables.

kohonen Supervised and unsupervised self-organising maps.

ks Kernel smoothing: bandwidth matrices for kernel density estimators and kernel discriminant analysis for bivariate data.

kza Kolmogorov-Zurbenko Adpative filter for locating change points in a time series.

kzft Kolmogorov-Zurbenko Fourier Transform and application.

labdsv Laboratory for Dynamic Synthetic Vegephenomenology.

labstatR Functions for the book "Laboratorio di statistica con R" by S. M. Iacus and G. Masarotto, 2002, McGraw-Hill. Function names and documentation in Italian.

lars Least Angle Regression, Lasso and Forward Stagewise: efficient procedures for fitting an entire lasso sequence with the cost of a single least squares fit.

lasso2 Routines and documentation for solving regression problems while imposing an L1 constraint on the estimates, based on the algorithm of Osborne et al. (1998).

latentnet Latent position and cluster models for statistical networks.

lattice Lattice graphics, an implementation of Trellis Graphics functions. *Recommended.*

latticeExtra
 Generic functions and standard methods for Trellis-based displays.

lazy Lazy learning for local regression.

ldDesign Design of experiments for detection of linkage disequilibrium,

ldbounds Lan-DeMets method for group sequential boundaries.

leaps A package which performs an exhaustive search for the best subsets of a given set of potential regressors, using a branch-and-bound algorithm, and also performs searches using a number of less time-consuming techniques.

lgtdl A set of methods for longitudinal data objects.

limma LInear Models for MicroArray data.

linprog Solve linear programming/linear optimization problems by using the simplex algorithm.

lme4 Fit linear and generalized linear mixed-effects models.

lmeSplines
Fit smoothing spline terms in Gaussian linear and nonlinear mixed-effects models.

lmm Linear mixed models.

lmomco L-moments and L-comoments.

lmtest A collection of tests on the assumptions of linear regression models from the book "The linear regression model under test" by W. Kraemer and H. Sonnberger, 1986, Physica.

locfdr Computation of local false discovery rates.

locfit Local Regression, likelihood and density estimation.

lodplot Assorted plots of location score versus genetic map position.

logistf Firth's bias reduced logistic regression approach with penalized profile likelihood based confidence intervals for parameter estimates.

logspline Logspline density estimation.

lokern Kernel regression smoothing with adaptive local or global plug-in bandwidth selection.

longmemo Datasets and Functionality from the textbook "Statistics for Long-Memory Processes" by J. Beran, 1994, Chapman & Hall.

lpSolve Functions that solve general linear/integer problems, assignment problems, and transportation problems via interfacing Lp_solve.

lpridge Local polynomial (ridge) regression.

lsa Latent Semantic Analysis.

lspls LS-PLS (least squares — partial least squares) models.

ltm Analysis of multivariate Bernoulli data using latent trait models (including the Rasch model) under the Item Response Theory approach.

mAr Estimation of multivariate AR models through a computationally efficient stepwise least-squares algorithm.

maanova Analysis of N-dye Micro Array experiments using mixed model effect. Contains anlysis of variance, permutation and bootstrap, cluster and consensus tree.

magic A variety of methods for creating magic squares of any order greater than 2, and various magic hypercubes.

mapLD Linkage Disequilibrium mapping.

mapdata Supplement to package **maps**, providing the larger and/or higher-resolution databases.

mapproj Map Projections: converts latitude/longitude into projected coordinates.

maps Draw geographical maps. Projection code and larger maps are in separate packages.

maptools Set of tools for manipulating and reading geographic data, in particular ESRI shapefiles.

maptree Functions with example data for graphing and mapping models from hierarchical clustering and classification and regression trees.

mathgraph
Tools for constructing and manipulating objects from a class of directed and undirected graphs.

matlab Emulate MATLAB code using R.

maxstat Maximally selected rank and Gauss statistics with several p-value approximations.

mblm Median-based Linear models, using Theil-Sen single or Siegel repeated medians.

mcgibbsit Warnes and Raftery's MCGibbsit MCMC diagnostic.

mclust Model-based cluster analysis: the 2002 version of MCLUST.

mcmc Functions for Markov Chain Monte Carlo (MCMC).

mda Code for mixture discriminant analysis (MDA), flexible discriminant analysis (FDA), penalized discriminant analysis (PDA), multivariate additive regression splines (MARS), adaptive back-fitting splines (BRUTO), and penalized regression.

merror Accuracy and precision of measurements.

meta Fixed and random effects meta-analysis, with functions for tests of bias, forest and funnel plot.

mfp Multiple Fractional Polynomials.

mgcv Routines for GAMs and other genralized ridge regression problems with multiple smoothing parameter selection by GCV or UBRE. *Recommended.*

micEcdat Data sets for micro-econometrics.

micEcon Tools for microeconomic analysis and microeconomic modelling.

mice Multivariate Imputation by Chained Equations.

mimR An R interface to MIM for graphical modeling in R.

minpack.lm

R interface for two functions from the MINPACK least squares optimization library, solving the nonlinear least squares problem by a modification of the Levenberg-Marquardt algorithm.

misc3d   A collection of miscellaneous 3d plots, including rgl-based isosurfaces.

mitools   Tools to perform analyses and combine results from multiple-imputation datasets.

mix   Estimation/multiple imputation programs for mixed categorical and continuous data.

mixreg   Functions to fit mixtures of regressions.

mlbench   A collection of artificial and real-world machine learning benchmark problems, including the Boston housing data.

mlica   Independent Component Analysis using Maximum Likelihood.

mlmRev   Examples from Multilevel Modelling Software Review.

mmlcr   Mixed-mode latent class regression (also known as mixed-mode mixture model regression or mixed-mode mixture regression models) which can handle both longitudinal and one-time responses.

mnormt   The multivariate normal and t distributions.

moc   Fits a variety of mixtures models for multivariate observations with user-difined distributions and curves.

modeltools

A collection of tools to deal with statistical models.

moments   Moments, skewness, kurtosis and related tests.

monreg   Estimation of monotone regression and variance functions in nonparametric models.

monoProc   Strictly monotone smoothing procedure.

mprobit   Multivariate probit model for binary/ordinal response.

msm   Functions for fitting continuous-time Markov multi-state models to categorical processes observed at arbitrary times, optionally with misclassified responses, and covariates on transition or misclassification rates.

muhaz   Hazard function estimation in survival analysis.

multcomp   Multiple comparison procedures for the one-way layout.

multilevel Analysis of multilevel data by organizational and social psychologists.

multinomRob

Overdispersed multinomial regression using robust (LQD and tanh) estimation.

multtest   Resampling-based multiple hypothesis testing.

mvbutils Utilities by Mark V. Bravington for project organization, editing and backup, sourcing, documentation (formal and informal), package preparation, macro functions, and more.

mvnmle ML estimation for multivariate normal data with missing values.

mvnormtest
Generalization of the Shapiro-Wilk test for multivariate variables.

mvoutlier Multivariate outlier detection based on robust estimates of location and covariance structure.

mvpart Multivariate partitioning.

mvtnorm Multivariate normal and t distributions.

nFDR Nonparametric Estimate of FDR Based on Bernstein polynomials.

ncdf Interface to Unidata netCDF data files.

ncomplete Functions to perform the regression depth method (RDM) to binary regression to approximate the minimum number of observations that can be removed such that the reduced data set has complete separation.

ncvar High-level R interface to netCDF datasets.

negenes Estimating the number of essential genes in a genome on the basis of data from a random transposon mutagenesis experiment, through the use of a Gibbs sampler.

network Tools to create and modify network objects, which can represent a range of relational data types.

neural RBF and MLP neural networks with graphical user interface.

nice Get or set UNIX priority (niceness) of running R process.

nlme Fit and compare Gaussian linear and nonlinear mixed-effects models. *Recommended.*

nlmeODE Combine the **nlme** and **odesolve** packages for mixed-effects modelling using differential equations.

nltm NonLinear Transformation Models for survival analysis.

nnet Software for single hidden layer perceptrons ("feed-forward neural networks"), and for multinomial log-linear models. Contained in the 'VR' bundle. *Recommended.*

nor1mix One-dimensional normal mixture models classes, for, e.g., density estimation or clustering algorithms research and teaching; providing the widely used Marron-Wand densities.

norm Analysis of multivariate normal datasets with missing values.

normalp A collection of utilities for normal of order p distributions (General Error Distributions).

nortest Five omnibus tests for the composite hypothesis of normality.

noverlap	Functions to perform the regression depth method (RDM) to binary regression to approximate the amount of overlap, i.e., the minimal number of observations that need to be removed such that the reduced data set has no longer overlap.
npmc	Nonparametric Multiple Comparisons: provides simultaneous rank test procedures for the one-way layout without presuming a certain distribution.
numDeriv	Accurate numerical derivatives.
nws	Functions for NetWorkSpaces and Sleigh.
odesolve	An interface for the Ordinary Differential Equation (ODE) solver lsoda. ODEs are expressed as R functions.
onion	A collection of routines to manipulate and visualize quaternions and octonions.
optmatch	Functions to perform optimal matching, particularly full matching.
orientlib	Representations, conversions and display of orientation SO(3) data.
ouch	Ornstein-Uhlenbeck models for phylogenetic comparative hypotheses.
outliers	A collection of some tests commonly used for identifying outliers.
oz	Functions for plotting Australia's coastline and state boundaries.
pamr	Pam: Prediction Analysis for Microarrays.
pan	Multiple imputation for multivariate panel or clustered data.
panel	Functions and datasets for fitting models to Panel data.
papply	Parallel apply function using MPI.
partsm	Periodic AutoRegressive Time Series Models.
party	Unbiased recursive partitioning in a conditional inference framework.
pastecs	Package for Analysis of Space-Time Ecological Series.
pbatR	Frontend to PBAT to run within R.
pcaPP	Robust PCA by Projection Pursuit.
pcurve	Fits a principal curve to a numeric multivariate dataset in arbitrary dimensions. Produces diagnostic plots. Also calculates Bray-Curtis and other distance matrices and performs multi-dimensional scaling and principal component analyses.
pear	Periodic Autoregression Analysis.
permax	Functions intended to facilitate certain basic analyses of DNA array data, especially with regard to comparing expression levels between two types of tissue.
permtest	Permutation test to compare variability within and distance between two groups.
perturb	Perturbation analysis for evaluating collinearity.
pgam	Poisson-Gamma Additive Models.

pgirmess Functions for analysis and display of ecological and spatial data.

pheno Some easy-to-use functions for time series analyses of (plant-) phenological data sets.

phpSerialize
Serialize R to PHP associative array.

pinktoe Converts S trees to HTML/Perl files for interactive tree traversal.

pixmap Functions for import, export, plotting and other manipulations of bitmapped images.

plotrix Various useful functions for enhancing plots.

plugdensity
Kernel density estimation with global bandwidth selection via "plug-in".

pls Partial Least Squares Regression (PLSR) and Principal Component Regression (PCR).

plsgenomics
PLS analyses for genomics.

polspline Routines for the polynomial spline fitting routines hazard regression, hazard estimation with flexible tails, logspline, lspec, polyclass, and polymars, by C. Kooperberg and co-authors.

polyapost Simulating from the Polya posterior.

polycor Polychoric and polyserial correlations.

polynom A collection of functions to implement a class for univariate polynomial manipulations.

popgen Statistical and POPulation GENetics.

portfolio Classes for analyzing and implementing portfolios.

powerpkg Power analyses for the affected sib pair and the TDT design.

ppc Sample classification of protein mass spectra by peak probabilty contrasts.

pps Functions to select samples using PPS (probability proportional to size) sampling, for stratified simple random sampling, and to compute joint inclusion probabilities for Sampford's method of PPS sampling.

prabclus Distance based parametric bootstrap tests for clustering, mainly thought for presence-absence data (clustering of species distribution maps). Jaccard and Kulczynski distance measures, clustering of MDS scores, and nearest neighbor based noise detection.

princurve Fits a principal curve to a matrix of points in arbitrary dimension.

proto An object oriented system using prototype or object-based (rather than class-based) object oriented ideas.

pscl R in the Political Science Computational Laboratory, Stanford University.

pspline Smoothing splines with penalties on order m derivatives.

psy Various procedures used in psychometry: Kappa, ICC, Cronbach alpha, screeplot, PCA and related methods.

pwr Basic functions for power analysis.

pwt The Penn World Table providing purchasing power parity and national income accounts converted to international prices for 168 countries for some or all of the years 1950–2000.

pvclust Hierarchical clustering with p-value.

qcc Quality Control Charts. Shewhart quality control charts for continuous, attribute and count data. Cusum and EWMA charts. Operating characteristic curves. Process capability analysis. Pareto chart and cause-and-effect chart.

qtl Analysis of experimental crosses to identify QTLs.

qtlDesign Tools for the design of QTL experiments.

quadprog For solving quadratic programming problems.

quantchem
 Quantitative chemical analysis: calibration and evaluation of results.

quantreg Quantile regression and related methods.

quantregForest
 Quantile Regression Forests, a tree-based ensemble method for estimation of conditional quantiles.

qvalue Q-value estimation for false discovery rate control.

qvcalc Functions to compute quasi-variances and associated measures of approximation error.

rJava Low-level R to Java interface. Allows creation of objects, calling methods and accessing fields.

race Implementation of some racing methods for the empirical selection of the best.

rake Raking survey datasets by re-weighting.

randaes Random number generator based on AES cipher.

randomForest
 Breiman's random forest classifier.

rankreg Rank regression estimator for the AFT model with right censored data.

rbugs Functions to prepare files needed for running BUGS in batch mode, and running BUGS from R. Support for Linux systems with Wine is emphasized.

rcdd C Double Description for R, an interface to the CDD computational geometry library.

rcom R COM Client Interface and internal COM Server.

rda Shrunken Centroids Regularized Discriminant Analysis.

ref	Functions for creating references, reading from and writing ro references and a memory efficient refdata type that transparently encapsulates matrices and data frames.
regress	Fitting Gaussian linear models where the covariance structure is a linear combination of known matrices by maximising the residual log likelihood. Can be used for multivariate models and random effects models.
relaimpo	RELAtive IMPOrtance of regressors in linear models.
relax	Functions for report writing, presentation, and programming.
reldist	Functions for the comparison of distributions, including nonparametric estimation of the relative distribution PDF and CDF and numerical summaries as described in "Relative Distribution Methods in the Social Sciences" by Mark S. Handcock and Martina Morris, 1999, Springer.
relimp	Functions to facilitate inference on the relative importance of predictors in a linear or generalized linear model.
relsurv	Various functions for regression in relative survival.
reshape	Flexibly reshape data.
resper	Sampling from restricted permutations.
rgdal	Provides bindings to Frank Warmerdam's Geospatial Data Abstraction Library (GDAL).
rgenoud	R version of GENetic Optimization Using Derivatives.
rggobi	Interface between R and GGobi.
rgl	3D visualization device system (OpenGL).
rimage	Functions for image processing, including Sobel filter, rank filters, fft, histogram equalization, and reading JPEG files.
riv	Robust Instrumental Variables estimators based on high breakpoint point S-estimators of multivariate location and scatter matrices.
rlecuyer	R interface to RNG with multiple streams.
rmeta	Functions for simple fixed and random effects meta-analysis for two-sample comparison of binary outcomes.
rmetasim	An interface between R and the metasim simulation engine. Facilitates the use of the metasim engine to build and run individual based population genetics simulations.
roblm	Robust regression estimators.
robust	Insightful robust package.
robustbase	Basic Robust Statistics.
rpart	Recursive PARTitioning and regression trees. *Recommended.*

rpart.permutation
> Permutation tests of rpart models.

rpvm   R interface to PVM (Parallel Virtual Machine). Provides interface to PVM APIs, and examples and documentation for its use.

rqmcmb2 Markov chain marginal bootstrap for quantile regression.

rrcov   Functions for robust location and scatter estimation and robust regression with high breakdown point.

rrp   Random Recursive Partitioning.

rsprng Provides interface to SPRNG (Scalable Parallel Random Number Generators) APIs, and examples and documentation for its use.

rstream Unified object oriented interface for multiple independent streams of random numbers from different sources.

rtiff   Read TIFF format images and return them as pixmap objects.

rv   Simulation-based random variable object class.

rwt   Rice Wavelet Toolbox wrapper, providing a set of functions for performing digital signal processing.

sac   Semiparametric empirical likelihood ratio based test of changepoint with one-change or epidemic alternatives with data-based model diagnostic.

sampling A set of tools to select and to calibrate samples.

sampfling Implements a modified version of the Sampford sampling algorithm. Given a quantity assigned to each unit in the population, samples are drawn with probability proportional to te product of the quantities of the units included in the sample.

samr   Significance Analysis of Microarrays.

sandwich Model-robust standard error estimators for time series and longitudinal data.

sca   Simple Component Analysis.

scape   functions to import and plot results from statistical catch-at-age models, used in fisheries stock assessments.

scapeMCMC
> Markov-chain Monte Carlo diagnostic plots, accompanying the **scape** package.

scatterplot3d
> Plots a three dimensional (3D) point cloud perspectively.

scope   Data manipulation using arbitrary row and column criteria.

scuba   Scuba diving calculations and decompression models.

seacarb Calculates parameters of the seawater carbonate system.

seas   Detailed seasonal plots of temperature and precipitation data.

seewave Time wave analysis and graphical representation.

segmented Functions to estimate break-points of segmented relationships in regression models (GLMs).

sem Functions for fitting general linear Structural Equation Models (with observed and unobserved variables) by the method of maximum likelihood using the RAM approach.

seqinr Exploratory data analysis and data visualization for biological sequence (DNA and protein) data.

seqmon Sequential monitoring of clinical trials.

session Functions for interacting with, saving and restoring R sessions.

setRNG Set (normal) random number generator and seed.

sfsmisc Utilities from Seminar fuer Statistik ETH Zurich.

sgeostat An object-oriented framework for geostatistical modeling.

shapefiles Functions to read and write ESRI shapefiles.

shapes Routines for the statistical analysis of shapes, including procrustes analysis, displaying shapes and principal components, testing for mean shape difference, thin-plate spline transformation grids and edge superimposition methods.

simex SIMEX and MCSIMEX algorithms for measurement error models.

simpleboot
Simple bootstrap routines.

skewt Density, distribution function, quantile function and random generation for the skewed t distribution of Fernandez and Steel.

sm Software linked to the book "Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations" by A. W. Bowman and A. Azzalini, 1997, Oxford University Press.

sma Functions for exploratory (statistical) microarray analysis.

smatr (Standardized) Major Axis estimation and Testing Routines.

smoothSurv
Survival regression with smoothed error distribution.

sn Functions for manipulating skew-normal probability distributions and for fitting them to data, in the scalar and the multivariate case.

sna A range of tools for social network analysis, including node and graph-level indices, structural distance and covariance methods, structural equivalence detection, p* modeling, and network visualization.

snow Simple Network of Workstations: support for simple parallel computing in R.

snowFT Fault Tolerant Simple Network of Workstations.

som Self-Organizing Maps (with application in gene clustering).

sound A sound interface for R: Basic functions for dealing with '`.wav`' files and sound samples.

sp A package that provides classes and methods for spatial data, including utility functions for plotting data as maps, spatial selection, amd much more.

spatial Functions for kriging and point pattern analysis from "Modern Applied Statistics with S" by W. Venables and B. Ripley. Contained in the 'VR' bundle. *Recommended.*

spatialCovariance
  Computation of spatial covariance matrices for data on rectangles using one dimensional numerical integration and analytic results.

spatstat Data analysis and modelling of two-dimensional point patterns, including multitype points and spatial covariates.

spc Statistical Process Control: evaluation of control charts by means of the zero-state, steady-state ARL (Average Run Length), setting up control charts for given in-control ARL, and plotting of the related figures.

spdep A collection of functions to create spatial weights matrix objects from polygon contiguities, from point patterns by distance and tesselations, for summarising these objects, and for permitting their use in spatial data analysis; a collection of tests for spatial autocorrelation, including global Moran's I and Geary's C, local Moran's I, saddlepoint approximations for global and local Moran's I; and functions for estimating spatial simultaneous autoregressive (SAR) models. (Was formerly the three packages: **spweights**, **sptests**, and **spsarlm**.)

spe Stochastic Proximity Embedding.

spectralGP
  Approximate Gaussian processes using the Fourier basis.

spectrino Spectra organizer, visualization and data extraction from within R.

spgwr Geographically weighted regression.

splancs Spatial and space-time point pattern analysis functions.

ssanv Sample Size Adjusted for Nonadherence or Variability of input parameters.

sspir State SPace models In R.

startupmsg
  Utilities for start-up messages.

statmod Miscellaneous biostatistical modelling functions.

stepwise A stepwise approach to identifying recombination breakpoints in a sequence alignment.

strucchange
  Various tests on structural change in linear regression models.

subselect A collection of functions which assess the quality of variable subsets as surrogates for a full data set, and search for subsets which are optimal under various criteria.

sudoku Sudoku puzzle solver.

supclust   Methodology for supervised grouping of predictor variables.

superpc   Supervised principal components.

survBayes   Fits a proportional hazards model to time to event data by a Bayesian approach.

surveillance
          Outbreak detection algorithms for surveillance data.

survey   Summary statistics, generalized linear models, and general maximum likelihood estimation for stratified, cluster-sampled, unequally weighted survey samples.

survival   Functions for survival analysis, including penalised likelihood. *Recommended.*

survrec   Survival analysis for recurrent event data.

svmpath   Computes the entire regularization path for the two-class svm classifier with essentialy the same cost as a single SVM fit.

systemfit   Contains functions for fitting simultaneous systems of equations using Ordinary Least Sqaures (OLS), Two-Stage Least Squares (2SLS), and Three-Stage Least Squares (3SLS).

tapiR   Tools for accessing (UK) parliamentary information in R.

taskPR   Task-Parallel R package.

tcltk2   A series of widgets and functions to supplement **tcltk**.

tdist   Computes the distribution of a linear combination of independent Student's t variables.

tdthap   Transmission/disequilibrium tests for extended marker haplotypes.

tensor   Tensor product of arrays.

tgp   Bayesian regression and adaptive sampling with Treed Gaussian Process models.

time   Time tracking for developers.

tkrplot   Simple mechanism for placing R graphics in a Tk widget.

tlnise   Two-level normal independent sampling estimation.

tree   Classification and regression trees.

treeglia   Stem analysis functions for volume increment and carbon uptake assessment from tree-rings.

tripack   A constrained two-dimensional Delaunay triangulation package.

truncgof   Goodness-of-fit tests allowing for left truncated data.

trust   Local optimization using two derivatives and trust regions.

tseries   Package for time series analysis with emphasis on non-linear modelling.

tseriesChaos
          Routines for the analysis of non-linear time series.

tsfa   Time Series Factor Analysis.

tuneR   Collection of tools to analyze music, handle wave files, transcription, etc.

twang   Toolkit for Weighting and Analysis of Nonequivalent Groups.

tweedie   Maximum likelihood computations for Tweedie exponential family models.

udunits   Interface to Unidata's routines to convert units.

ump   Uniformly Most Powerful tests.

untb   Ecological drift under the UNTB (Unified Neutral Theory of Biodiversity).

urca   Unit root and cointegration tests for time series data.

urn   Functions for sampling without replacement (simulated urns).

uroot   Unit root tests and graphics for seasonal time series.

vabayelMix
               Variational Bayesian mixture model.

varSelRF   Variable selection using random forests.

vardiag   Interactive variogram diagnostics.

varmixt   Mixture model on the variance for the analysis of gene expression data.

vcd   Functions and data sets based on the book "Visualizing Categorical Data" by Michael Friendly.

vegan   Various help functions for vegetation scientists and community ecologists.

verification
               Utilities for verification of discrete and probabilistic forecasts.

verify   Construction of test suites using verify objects.

vioplot   Violin plots, which are a combination of a box plot and a kernel density plot.

wavelets   Functions for computing wavelet filters, wavelet transforms and multiresolution analyses.

waveslim   Basic wavelet routines for time series analysis.

wavethresh
               Software to perform 1-d and 2-d wavelet statistics and transforms.

wccsom   SOM networks for comparing patterns with peak shifts.

wle   Robust statistical inference via a weighted likelihood approach.

xgobi   Interface to the XGobi and XGvis programs for graphical data analysis.

xtable   Export data to LaTeX and HTML tables.

zicounts   Fit classical, zero-inflated and interval censored count data regression models.

zoo   A class with methods for totally ordered indexed observations such as irregular time series.

See CRAN '`src/contrib/PACKAGES`' for more information.

There is also a CRAN '`src/contrib/Devel`' directory which contains packages still "under development" or depending on features only present in the current development versions of R. Volunteers are invited to give these a try, of course. This area of CRAN currently contains

GLMMGibbs
> Generalised Linear Mixed Models by Gibbs sampling.

RPgSQL Provides methods for accessing data stored in PostgreSQL tables.

dseplus Extensions to **dse**, the Dynamic Systems Estimation multivariate time series package. Contains PADI, juice and monitoring extensions.

ensemble Ensembles of tree classifiers.

runStat Running median and mean.

write.snns Function for writing a SNNS pattern file from a data frame or matrix.

5.1.3 Add-on packages from Omegahat

The Omegahat Project for Statistical Computing provides a variety of open-source software for statistical applications, with special emphasis on web-based software, Java, the Java virtual machine, and distributed computing. A CRAN style R package repository is available via `http://www.omegahat.org/R/`.

Currently, there are the following packages.

Aspell An interface to facilities in the aspell library.

CGIwithR Facilities for the use of R to write CGI scripts.

CORBA Dynamic CORBA client/server facilities for R. Connects to other CORBA-aware applications developed in arbitrary languages, on different machines and allows R functionality to be exported in the same way to other applications.

Combinations
> Compute the combinations of choosing r items from n elements.

IDocs Infrastructure for interactive documents.

OOP OOP style classes and methods for R and S-Plus. Object references and class-based method definition are supported in the style of languages such as Java and C++.

RCurl Allows one to compose HTTP requests to fetch URIs, post forms, etc., and process the results returned by the Web server.

RDCOMClient
> Provides dynamic client-side access to (D)COM applications from within R.

RDCOMEvents
> Provides facilities to use R functions and objects as handlers for DCOM events.

RDCOMServer
> Facilities for exporting S objects and functions as COM objects.

REmbeddedPostgres

Allows R functions and objects to be used to implement SQL functions —
per-record, aggregate and trigger functions.

REventLoop

An abstract event loop mechanism that is toolkit independent and can be used
to to replace the R event loop.

Rexif   Extract meta-information from JPEG files.

RGdkPixbuf

S language functions to access the facilities in the GdkPixbuf library for ma-
nipulating images.

RGnumeric

A plugin for the Gnumeric spreadsheet that allows R functions to be called
from cells within the sheet, automatic recalculation, etc.

RGtk   Facilities in the S language for programming graphical interfaces using Gtk, the
Gnome GUI toolkit.

RGtkBindingGenerator

A meta-package which generates C and R code to provide bindings to a Gtk-
based library.

RGtkExtra

A collection of S functions that provide an interface to the widgets in the
gtk+extra library such as the GtkSheet data-grid display, icon list, file list and
directory tree.

RGtkGlade

S language bindings providing an interface to Glade, the interactive Gnome
GUI creator.

RGtkHTML

A collection of S functions that provide an interface to creating and controlling
an HTML widget which can be used to display HTML documents from files or
content generated dynamically in S.

RGtkIPrimitives

A collection of low-level primitives for interactive use with R graphics and the
gtkDevice using **RGtk**.

RGtkViewers

A collection of tools for viewing different S objects, databases, class and widget
hierarchies, S source file contents, etc.

RJavaDevice

A graphics device for R that uses Java components and graphics. APIs.

RMatlab   A bi-directional interface between R and Matlab.

RObjectTables

The C and S code allows one to define R objects to be used as elements of
the search path with their own semantics and facilities for reading and writing

variables. The objects implement a simple interface via R functions (either methods or closures) and can access external data, e.g., in other applications, languages, formats, . . .

RSMethods
    An implementation of S version 4 methods and classes for R, consistent with the basic material in "Programming with Data" by John M. Chambers, 1998, Springer NY.

RSPerl   An interface from R to an embedded, persistent Perl interpreter, allowing one to call arbitrary Perl subroutines, classes and methods.

RSPython   Allows Python programs to invoke S functions, methods, etc., and S code to call Python functionality.

Rlibstree   Suffix Trees in R via the libstree C library.

RXLisp   An interface to call XLisp-Stat functions from within R.

Rstem   Interface to Snowball implementation of Porter's word stemming algorithm.

SASXML   Example for reading XML files in SAS 8.2 manner.

SJava   An interface from R to Java to create and call Java objects and methods.

SLanguage
    Functions and C support utilities to support S language programming that can work in both R and S-Plus.

SNetscape   Plugin for Netscape and JavaScript.

SSOAP   A client interface to SOAP (Simple Object Access Protocol) servers from within S.

SWinRegistry
    Provides access from within R to read and write the Windows registry.

SWinTypeLibs
    Provides ways to extract type information from type libraries and/or DCOM objects that describes the methods, properties, etc., of an interface.

SXalan   Process XML documents using XSL functions implemented in R and dynamically substituting output from R.

Slcc   Parses C source code, allowing one to analyze and automatically generate interfaces from S to that code, including the table of S-accessible native symbols, parameter count and type information, S constructors from C objects, call graphs, etc.

Sxslt   An extension module for libxslt, the XML-XSL document translator, that allows XSL functions to be implemented via R functions.

XML   Tools for reading XML documents and DTDs.

5.1.4 Add-on packages from Bioconductor

The Bioconductor Project produces an open source software framework that will assist biologists and statisticians working in bioinformatics, with primary emphasis on inference using DNA microarrays. A CRAN style R package repository is available via http://www.bioconductor.org/.

The following R packages are contained in the current release of Bioconductor, with more packages under development.

AnnBuilder
: Assemble and process genomic annotation data, from databases such as Gen-Bank, the Gene Ontology Consortium, LocusLink, UniGene, the UCSC Human Genome Project.

Biobase
: Object-oriented representation and manipulation of genomic data (S4 class structure).

Biostrings
: Class definitions and generics for biological sequences along with pattern matching algorithms.

Category
: A collection of tools for performing category analysis.

ChromoViz
: Draw gene expression profile onto chromosome using SVG.

CoCiteStats
: A collection of software tools for dealing with co-citation data.

DEDS
: Differential Expression via Distance Summary for microarray data.

DNAcopy
: Segments DNA copy number data using circular binary segmentation to detect regions with abnormal copy number.

DynDoc
: Functionality to create and interact with dynamic documents, vignettes, and other navigable documents.

EBImage
: R image processing toolkit.

EBarrays
: Empirical Bayes tools for the analysis of replicated microarray data across multiple conditions.

GEOquery
: Get data from NCBI Gene Expression Omnibus (GEO).

GLAD
: Gain and Loss Analysis of DNA.

GOstats
: Tools for manipulating GO and microarrays.

GeneMeta
: A collection of meta-analysis tools for analyzing high throughput experimental data.

GeneR
: Package manipulating nucleotidic sequences (Embl, Fasta, GenBank).

GeneSpring
: Functions and class definitions to be able to read and write GeneSpring specific data objects and convert them to Bioconductor objects.

GeneTS   A package for analysing multiple gene expression time series data. Currently, implements methods for cell cycle analysis and for inferring large sparse graphical Gaussian models.

GeneTraffic
            GeneTraffic R integration functions.

GenomeBase
            Base functions for genome data package manipulation.

GlobalAncova
            Calculates a global test for differential gene expression between groups.

GraphAT   Graph theoretic Association Tests.

HEM   Heterogeneous Error Model for analysis of microarray data.

Heatplus   A heat map displaying covariates and coloring clusters.

Icens   Functions for computing the NPMLE for censored and truncated data.

KEGGSOAP
            Client-side SOAP access KEGG.

LMGene   Analysis of microarray data using a linear model and glog data transformation.

LPE   Significance analysis of microarray data with small number of replicates using the Local Pooled Error (LPE) method.

MANOR   Micro-Array NORmalization.

MCRestimate
            Misclassification error estimation with cross-validation.

MLInterfaces
            Uniform interfaces to machine learning code for the exprSet class from Bioconductor.

MVCClass
            Model-View-Controller (MVC) classes.

MantelCorr
            Compute Mantel Cluster Correlations.

MeasurementError.cor
            Two-stage measurement error model for correlation estimation with smaller bias than the usual sample correlation.

MergeMaid
            Cross-study comparison of gene expression array data.

Mfuzz   Soft clustering of time series gene expression data.

MiPP   Misclassification Penalized Posterior Classification.

OCplus   Operating characteristics plus sample size and local fdr for microarray experiments.

OLIN   Optimized Local Intensity-dependent Normalisation of two-color microarrays.

OLINgui   Graphical user interface for **OLIN**.

OrderedList
> Similarities of ordered gene lists.

PROcess   Ciphergen SELDI-TOF processing.

RBGL   An interface between the graph package and the Boost graph libraries, allowing for fast manipulation of graph objects in R.

RLMM   A genotype calling algorithm for Affymetrix SNP arrays.

RMAGEML
> Functionality to handle MAGEML documents.

RMAPPER
> Interface to mapper.chip.org.

ROC   Receiver Operating Characteristic (ROC) approach for identifying genes that are differentially expressed in two types of samples.

RSNPper   Interface to chip.org::SNPper for SNP-related data.

RankProd   Rank Product method for identifying differentially expressed genes.

RdbiPgSQL
> Methods for accessing data stored in PostgreSQL tables.

Rdbi   Generic framework for database access in R.

Resourcerer
> Read annotation data from TIGR Resourcerer or convert the annotation data into Bioconductor data package.

Rgraphviz   An interface with Graphviz for plotting graph objects in R.

Ruuid   Creates Universally Unique ID values (UUIDs) in R.

SAGElyzer
> Locates genes based on SAGE tags.

SBMLR   Systems Biology Markup Language (SBML) interface and biochemical system analysis tools.

SNAData   Data from the book "Social Network Analysis" by Wasserman & Faust, 1999.

ScISI   In Silico Interactome.

aCGH   Classes and functions for Array Comparative Genomic Hybridization data.

adSplit   Annotation-driven clustering.

affxparser   Package for parsing Affymetrix files (CDF, CEL, CHP, BPMAP, BAR).

affy   Methods for Affymetrix Oligonucleotide Arrays.

affyQCReport
> QC Report Generation for affyBatch objects.

affyPLM   For fitting Probe Level Models.

affycomp   Graphics toolbox for assessment of Affymetrix expression measures.

affycoretools

Functions useful for those doing repetitive analyses.

affydata   Affymetrix data for demonstration purposes.

affyio   Tools for parsing Affymetrix data files.

affylmGUI

Graphical User Interface for affy analysis using package **limma**.

affypdnn   Probe Dependent Nearest Neighbors (PDNN) for the affy package.

altcdfenvs   Utilities to handle cdfenvs.

annaffy   Functions for handling data from Bioconductor Affymetrix annotation data packages.

annotate   Associate experimental data in real time to biological metadata from web databases such as GenBank, LocusLink and PubMed. Process and store query results. Generate HTML reports of analyses.

apComplex

Estimate protein complex membership using AP-MS protein data.

applera   Tools for Applied Biosystems microarrays AB1700 data analysis and quality controls.

arrayMagic

Utilities for quality control and processing for two-color cDNA microarray data.

arrayQuality

Performing print-run and array level quality assessment.

beadarray   Quality control and low-level analysis of BeadArrays.

bim   Bayesian interval mapping diagnostics: functions to interpret QTLCart and Bmapqtl samples.

bioDist   A collection of software tools for calculating distance measures.

biocViews   Categorized views of R package repositories.

biomaRt   Interface to BioMart databases (e.g., Ensembl)

bridge   Bayesian Robust Inference for Differential Gene Expression.

cellHTS   Analysis of cell-based screens.

cghMCR   Find chromosome regions showing common gains/losses.

clusterStab

Compute cluster stability scores for microarray data.

codelink   Manipulation of Codelink Bioarrays data.

convert   Convert Microarray Data Objects.

copa   Functions to perform cancer outlier profile analysis.

ctc   Tools to export and import Tree and Cluster to other programs.

daMA   Functions for the efficient design of factorial two-color microarray experiments
                 and for the statistical analysis of factorial microarray data.

diffGeneAnalysis
                 Performs differential Gene expression Analysis.

ecolitk   Metadata and tools to work with E. coli.

edd   Expression density diagnostics: graphical methods and pattern recognition al-
                 gorithms for distribution shape classification.

exprExternal
                 Implementation of exprSet using externalVectors.

externalVector
                 Basic class definitions and generics for external pointer based vector objects for
                 R.

factDesign   A set of tools for analyzing data from factorial designed microarray experiments.
                 The functions can be used to evaluate appropriate tests of contrast and perform
                 single outlier detection.

fdrame   FDR Adjustments of Microarray Experiments (FDR-AME).

gcrma   Background adjustment using sequence information.

genArise   A tool for dual color microarray data.

geneRecommender
                 A gene recommender algorithm to identify genes coexpressed with a query set
                 of genes.

genefilter   Tools for sequentially filtering genes using a wide variety of filtering functions.
                 Example of filters include: number of missing value, coefficient of variation of
                 expression measures, ANOVA p-value, Cox model p-values. Sequential applica-
                 tion of filtering functions to genes.

geneplotter
                 Graphical tools for genomic data, for example for plotting expression data along
                 a chromosome or producing color images of expression data matrices.

gff3Plotter
                 Plotting data of experiments on the genomic layout.

globaltest   Testing globally whether a group of genes is significantly related to some clinical
                 variable of interest.

goCluster   Analysis of clustering results in conjunction with annotation data.

goTools   Functions for description/comparison of oligo ID list using the Gene Ontology
                 database.

gpls   Classification using generalized partial least squares for two-group and multi-
                 group classification.

graph   Classes and tools for creating and manipulating graphs within R.

gtkWidgets
        Widgets built using **RGtk**.

hexbin   Binning functions, in particular hexagonal bins for graphing.

hopach   Hierarchical Ordered Partitioning and Collapsing Hybrid (HOPACH).

hypergraph
        Capabilities for representing and manipulating hypergraphs.

iSPlot   Link views that are based on the same data set.

idiogram   Plotting genomic data by chromosomal location.

impute   Imputation for microarray data (currently KNN only).

limma   Linear models for microarray data.

limmaGUI
        Graphical User Interface for package **limma**.

logicFS   Identification of SNP interactions.

maCorrPlot
        Visualize artificial correlation in microarray data.

maDB   Microarray database and utility functions for microarray analysis.

maSigPro Significant gene expression profile differeneces in time course microarray data.

maanova   Tools for analyzing micro array experiments.

macat   MicroArray Chromosome Analysis Tool.

made4   Multivariate analysis of microarray data using ADE4.

makecdfenv
        Two functions. One reads a Affymetrix chip description file (CDF) and creates a hash table environment containing the location/probe set membership mapping. The other creates a package that automatically loads that environment.

marray   Exploratory analysis for two-color spotted microarray data.

matchprobes
        Tools for sequence matching of probes on arrays.

metaArray
        Integration of microarray data for meta-analysis.

mmgmos   Multi-chip Modified Gamma Model of Oligonucleotide Signal.

multtest   Multiple testing procedures for controlling the family-wise error rate (FWER) and the false discovery rate (FDR). Tests can be based on t- or F-statistics for one- and two-factor designs, and permutation procedures are available to estimate adjusted p-values.

nnNorm   Spatial and intensity based normalization of cDNA microarray data based on robust neural nets.

nudge   Normal Uniform Differential Gene Expression detection.

ontoTools   Graphs and sparse matrices for working with ontologies; formal objects for nomenclatures with provenance management.

pairseqsim

Pairwise sequence alignment and scoring algorithms for global, local and overlap alignment with affine gap penalty.

pamr   Pam: Prediction Analysis for Microarrays.

panp   Presence-Absence calls from Negative strand matching Probesets.

pathRender

Render molecular pathways.

pdmclass   CLASSification of microarray samples using Penalized Discriminant Methods.

pgUtils   Utility functions for PostgreSQL databases.

pickgene   Adaptive gene picking for microarray expression data analysis.

plgem   Power Law Global Error Model.

plier   Implements the Affymetrix PLIER (Probe Logarithmic Error Intensity Estimate) algorithm.

prada   Tools for analyzing and navigating data from high-throughput phenotyping experiments based on cellular assays and fluorescent detection.

qvalue   Q-value estimation for false discovery rate control.

rama   Robust Analysis of MicroArrays: robust estimation of cDNA microarray intensities with replicates using a Bayesian hierarchical model.

reb   Regional Expression Biases.

reposTools

Tools for dealing with file repositories and allow users to easily install, update, and distribute packages, vignettes, and other files.

rfcdmin   Data sets for RFlowCyt examples.

rflowcyt   Statistical tools and data structures for analytic flow cytometry.

safe   Significance Analysis of Function and Expression.

sagenhaft   Functions for reading and comparing SAGE (Serial Analysis of Gene Expression) libraries.

siggenes   Identifying differentially expressed genes and estimating the False Discovery Rate (FDR) with both the Significance Analysis of Microarrays (SAM) and the Empirical Bayes Analyses of Microarrays (EBAM).

simpleaffy   Very simple high level analysis of Affymetrix data.

simulatorAPMS

Computationally simulates the AP-MS technology.

sizepower   Sample size and power calculation in microarray studies.

snapCGH   Segmentation, normalization and processing of aCGH data.

splicegear   A set of tools to work with alternative splicing.

spotSegmentation
          Microarray spot segmentation and gridding for blocks of microarray spots.

sscore   S-score algorithm for Affymetrix oligonucleotide microarrays.

ssize   Estimate microarry sample size.

stam   STructured Analysis of Microarray data.

stepNorm   Stepwise normalization functions for cDNA microarrays.

tilingArray
          Analysis of tiling arrays.

timecourse
          Statistical analysis for developmental microarray time course data.

tkWidgets   Widgets in Tcl/Tk that provide functionality for Bioconductor packages.

twilight   Estimation of local false discovery rate.

vsn   Calibration and variance stabilizing transformations for both Affymetrix and cDNA array data.

webbioc   Integrated web interface for doing microarray analysis using several of the Bioconductor packages.

widgetInvoke
          Evaluation widgets for functions.

widgetTools
          Tools for creating Tcl/Tk widgets, i.e., small-scale graphical user interfaces.

xcms   LC/MS and GC/MS data analysis: framework for processing and visualization of chromatographically separated mass spectral data.

y2hStat   Analysis of Yeast 2-Hybrid data sets.

5.1.5 Other add-on packages

Jim Lindsey has written a collection of R packages for nonlinear regression and repeated measurements, consisting of **event** (event history procedures and models), **gnlm** (generalized nonlinear regression models), **growth** (multivariate normal and elliptically-contoured repeated measurements models), **repeated** (non-normal repeated measurements models), **rmutil** (utilities for nonlinear regression and repeated measurements), and **stable** (probability functions and generalized regression models for stable distributions). All analyses in the new edition of his book "Models for Repeated Measurements" (1999, Oxford University Press) were carried out using these packages. Jim has also started **dna**, a package with procedures for the analysis of DNA sequences. Jim's packages can be obtained from http://popgen.unimaas.nl/~jlindsey/rcode.html.

More code has been posted to the R-help mailing list, and can be obtained from the mailing list archive.

5.2 How can add-on packages be installed?

(Unix only.)   The add-on packages on CRAN come as gzipped tar files named
pkg_version.`tar.gz`, which may in fact be "bundles" containing more than one package.
Provided that `tar` and `gzip` are available on your system, type

```
$ R CMD INSTALL /path/to/pkg_version.tar.gz
```

at the shell prompt to install to the library tree rooted at the first directory given in
`R_LIBS` (see below) if this is set and non-null, and to the default library (the '`library`'
subdirectory of '`R_HOME`') otherwise. (Versions of R prior to 1.3.0 installed to the default
library by default.)

To install to another tree (e.g., your private one), use

```
$ R CMD INSTALL -l lib /path/to/pkg_version.tar.gz
```

where *lib* gives the path to the library tree to install to.

Even more conveniently, you can install and automatically update packages from
within R if you have access to repositories such as CRAN. See the help page for
`available.packages()` for more information.

You can use several library trees of add-on packages. The easiest way to tell R to use
these is via the environment variable `R_LIBS` which should be a colon-separated list of
directories at which R library trees are rooted. You do not have to specify the default
tree in `R_LIBS`. E.g., to use a private tree in '`$HOME/lib/R`' and a public site-wide tree in
'`/usr/local/lib/R-contrib`', put

```
R_LIBS="$HOME/lib/R:/usr/local/lib/R-contrib"; export R_LIBS
```

into your (Bourne) shell profile or even preferably, add the line

```
R_LIBS="~/lib/R:/usr/local/lib/R-contrib"
```

your '`~/.Renviron`' file. (Note that no `export` statement is needed or allowed in this file;
see the on-line help for `Startup` for more information.)

5.3 How can add-on packages be used?

To find out which additional packages are available on your system, type

```
library()
```

at the R prompt.

This produces something like

```
Packages in '/home/me/lib/R':

mystuff       My own R functions, nicely packaged but not documented

Packages in '/usr/local/lib/R/library':

KernSmooth    Functions for kernel smoothing for Wand & Jones (1995)
MASS          Main Package of Venables and Ripley's MASS
base          The R Base package
boot          Bootstrap R (S-Plus) Functions (Canty)
class         Functions for Classification
cluster       Functions for clustering (by Rousseeuw et al.)
datasets      The R datasets Package
foreign       Read data stored by Minitab, S, SAS, SPSS, Stata, ...
grDevices     The R Graphics Devices and Support for Colours and Fonts
graphics      The R Graphics Package
grid          The Grid Graphics Package
lattice       Lattice Graphics
methods       Formal Methods and Classes
mgcv          GAMs with GCV smoothness estimation and GAMMs by REML/PQ
nlme          Linear and nonlinear mixed effects models
nnet          Feed-forward Neural Networks and Multinomial Log-Linear
              Models
rpart         Recursive partitioning
spatial       Functions for Kriging and Point Pattern Analysis
splines       Regression Spline Functions and Classes
stats         The R Stats Package
stats4        Statistical functions using S4 classes
survival      Survival analysis, including penalised likelihood
tcltk         Tcl/Tk Interface
tools         Tools for Package Development
utils         The R Utils Package
```

You can "load" the installed package *pkg* by

```
library(pkg)
```

You can then find out which functions it provides by typing one of

```
library(help = pkg)
help(package = pkg)
```

You can unload the loaded package *pkg* by

```
detach("package:pkg")
```

5.4 How can add-on packages be removed?

Use

```
$ R CMD REMOVE pkg_1 ... pkg_n
```

to remove the packages *pkg_1*, ..., *pkg_n* from the library tree rooted at the first directory given in `R_LIBS` if this is set and non-null, and from the default library otherwise. (Versions of R prior to 1.3.0 removed from the default library by default.)

To remove from library *lib*, do

```
$ R CMD REMOVE -l lib pkg_1 ... pkg_n
```

5.5 How can I create an R package?

A package consists of a subdirectory containing the files 'DESCRIPTION' and 'INDEX', and the subdirectories 'R', 'data', 'demo', 'exec', 'inst', 'man', 'src', and 'tests' (some of which can be missing). Optionally the package can also contain script files 'configure' and 'cleanup' which are executed before and after installation.

See section "Creating R packages" in *Writing R Extensions*, for details. This manual is included in the R distribution, see Section 2.7 [What documentation exists for R?], page 5, and gives information on package structure, the configure and cleanup mechanisms, and on automated package checking and building.

R version 1.3.0 has added the function package.skeleton() which will set up directories, save data and code, and create skeleton help files for a set of R functions and datasets.

See Section 2.10 [What is CRAN?], page 9, for information on uploading a package to CRAN.

5.6 How can I contribute to R?

R is in active development and there is always a risk of bugs creeping in. Also, the developers do not have access to all possible machines capable of running R. So, simply using it and communicating problems is certainly of great value.

One place where functionality is still missing is the modeling software as described in "Statistical Models in S" (see Section 3.1 [What is S?], page 11); some of the nonlinear modeling code is not there yet.

The R Developer Page acts as an intermediate repository for more or less finalized ideas and plans for the R statistical system. It contains (pointers to) TODO lists, RFCs, various other writeups, ideas lists, and CVS miscellanea.

Many (more) of the packages available at the Statlib S Repository might be worth porting to R.

If you are interested in working on any of these projects, please notify Kurt Hornik.

6 R and Emacs

6.1 Is there Emacs support for R?

There is an Emacs package called ESS ("Emacs Speaks Statistics") which provides a standard interface between statistical programs and statistical processes. It is intended to provide assistance for interactive statistical programming and data analysis. Languages supported include: S dialects (R, S 3/4, and S-Plus 3.x/4.x/5.x/6.x/7.x), LispStat dialects (XLispStat, ViSta), SAS, Stata, and BUGS.

ESS grew out of the need for bug fixes and extensions to S-mode 4.8 (which was a GNU Emacs interface to S/S-Plus version 3 only). The current set of developers desired support for XEmacs, R, S4, and MS Windows. In addition, with new modes being developed for R, Stata, and SAS, it was felt that a unifying interface and framework for the user interface would benefit both the user and the developer, by helping both groups conform to standard Emacs usage. The end result is an increase in efficiency for statistical programming and data analysis, over the usual tools.

R support contains code for editing R source code (syntactic indentation and highlighting of source code, partial evaluations of code, loading and error-checking of code, and source code revision maintenance) and documentation (syntactic indentation and highlighting of source code, sending examples to running ESS process, and previewing), interacting with an inferior R process from within Emacs (command-line editing, searchable command history, command-line completion of R object and file names, quick access to object and search lists, transcript recording, and an interface to the help system), and transcript manipulation (recording and saving transcript files, manipulating and editing saved transcripts, and re-evaluating commands from transcript files).

The latest stable version of ESS are available via CRAN or the ESS web page. The HTML version of the documentation can be found at http://stat.ethz.ch/ESS/.

ESS comes with detailed installation instructions.

For help with ESS, send email to ESS-help@stat.math.ethz.ch.

Please send bug reports and suggestions on ESS to ESS-bugs@stat.math.ethz.ch. The easiest way to do this from is within Emacs by typing `M-x ess-submit-bug-report` or using the [ESS] or [iESS] pulldown menus.

6.2 Should I run R from within Emacs?

Yes, *definitely*. Inferior R mode provides a readline/history mechanism, object name completion, and syntax-based highlighting of the interaction buffer using Font Lock mode, as well as a very convenient interface to the R help system.

Of course, it also integrates nicely with the mechanisms for editing R source using Emacs. One can write code in one Emacs buffer and send whole or parts of it for execution to R; this is helpful for both data analysis and programming. One can also seamlessly integrate with a revision control system, in order to maintain a log of changes in your programs and data, as well as to allow for the retrieval of past versions of the code.

In addition, it allows you to keep a record of your session, which can also be used for error recovery through the use of the transcript mode.

To specify command line arguments for the inferior R process, use `C-u M-x R` for starting R.

6.3 Debugging R from within Emacs

To debug R "from within Emacs", there are several possibilities. To use the Emacs GUD (Grand Unified Debugger) library with the recommended debugger GDB, type `M-x gdb` and give the path to the R *binary* as argument. At the `gdb` prompt, set `R_HOME` and other environment variables as needed (using e.g. `set env R_HOME /path/to/R/`, but see also below), and start the binary with the desired arguments (e.g., `run --quiet`).

If you have ESS, you can do `C-u M-x R` (RET) `- d` (SPC) `g d b` (RET) to start an inferior R process with arguments '`-d gdb`'.

A third option is to start an inferior R process via ESS (`M-x R`) and then start GUD (`M-x gdb`) giving the R binary (using its full path name) as the program to debug. Use the program `ps` to find the process number of the currently running R process then use the `attach` command in gdb to attach it to that process. One advantage of this method is that you have separate `*R*` and `*gud-gdb*` windows. Within the `*R*` window you have all the ESS facilities, such as object-name completion, that we know and love.

When using GUD mode for debugging from within Emacs, you may find it most convenient to use the directory with your code in it as the current working directory and then make a symbolic link from that directory to the R binary. That way '`.gdbinit`' can stay in the directory with the code and be used to set up the environment and the search paths for the source, e.g. as follows:

```
set env R_HOME /opt/R
set env R_PAPERSIZE letter
set env R_PRINTCMD lpr
dir /opt/R/src/appl
dir /opt/R/src/main
dir /opt/R/src/nmath
dir /opt/R/src/unix
```

7 R Miscellanea

7.1 How can I set components of a list to NULL?

You can use

```
x[i] <- list(NULL)
```

to set component `i` of the list `x` to NULL, similarly for named components. Do not set `x[i]` or `x[[i]]` to NULL, because this will remove the corresponding component from the list.

For dropping the row names of a matrix `x`, it may be easier to use `rownames(x) <- NULL`, similarly for column names.

7.2 How can I save my workspace?

`save.image()` saves the objects in the user's `.GlobalEnv` to the file '.RData' in the R startup directory. (This is also what happens after `q("yes")`.) Using `save.image(file)` one can save the image under a different name.

7.3 How can I clean up my workspace?

To remove all objects in the currently active environment (typically `.GlobalEnv`), you can do

```
rm(list = ls(all = TRUE))
```

(Without '`all = TRUE`', only the objects with names not starting with a '.' are removed.)

7.4 How can I get eval() and D() to work?

Strange things will happen if you use `eval(print(x), envir = e)` or `D(x^2, "x")`. The first one will either tell you that `"x"` is not found, or print the value of the wrong `x`. The other one will likely return zero if `x` exists, and an error otherwise.

This is because in both cases, the first argument is evaluated in the calling environment first. The result (which should be an object of mode `"expression"` or `"call"`) is then evaluated or differentiated. What you (most likely) really want is obtained by "quoting" the first argument upon surrounding it with `expression()`. For example,

```
R> D(expression(x^2), "x")
2 * x
```

Although this behavior may initially seem to be rather strange, is perfectly logical. The "intuitive" behavior could easily be implemented, but problems would arise whenever the expression is contained in a variable, passed as a parameter, or is the result of a function call. Consider for instance the semantics in cases like

```
D2 <- function(e, n) D(D(e, n), n)
```

or

```
g <- function(y) eval(substitute(y), sys.frame(sys.parent(n = 2)))
g(a * b)
```

See the help page for `deriv()` for more examples.

7.5 Why do my matrices lose dimensions?

When a matrix with a single row or column is created by a subscripting operation, e.g.,
`row <- mat[2, ]`, it is by default turned into a vector. In a similar way if an array with
dimension, say, 2 x 3 x 1 x 4 is created by subscripting it will be coerced into a 2 x 3 x 4
array, losing the unnecessary dimension. After much discussion this has been determined
to be a *feature*.

To prevent this happening, add the option 'drop = FALSE' to the subscripting. For
example,

```
rowmatrix <- mat[2, , drop = FALSE]   # creates a row matrix
colmatrix <- mat[, 2, drop = FALSE]   # creates a column matrix
a <- b[1, 1, 1, drop = FALSE]          # creates a 1 x 1 x 1 array
```

The 'drop = FALSE' option should be used defensively when programming. For example,
the statement

```
somerows <- mat[index, ]
```

will return a vector rather than a matrix if `index` happens to have length 1, causing errors
later in the code. It should probably be rewritten as

```
somerows <- mat[index, , drop = FALSE]
```

7.6 How does autoloading work?

R has a special environment called `.AutoloadEnv`. Using *autoload(name, pkg)*, where
name and *pkg* are strings giving the names of an object and the package containing it,
stores some information in this environment. When R tries to evaluate *name*, it loads the
corresponding package *pkg* and reevaluates *name* in the new package's environment.

Using this mechanism makes R behave as if the package was loaded, but does not occupy
memory (yet).

See the help page for `autoload()` for a very nice example.

7.7 How should I set options?

The function `options()` allows setting and examining a variety of global "options" which
affect the way in which R computes and displays its results. The variable `.Options` holds
the current values of these options, but should never directly be assigned to unless you want
to drive yourself crazy—simply pretend that it is a "read-only" variable.

For example, given

```
test1 <- function(x = pi, dig = 3) {
  oo <- options(digits = dig); on.exit(options(oo));
  cat(.Options$digits, x, "\n")
}
test2 <- function(x = pi, dig = 3) {
  .Options$digits <- dig
  cat(.Options$digits, x, "\n")
}
```

we obtain:

```
R> test1()
3 3.14
R> test2()
3 3.141593
```

What is really used is the *global* value of `.Options`, and using `options(OPT = VAL)` correctly updates it. Local copies of `.Options`, either in `.GlobalEnv` or in a function environment (frame), are just silently disregarded.

7.8 How do file names work in Windows?

As R uses C-style string handling, '\' is treated as an escape character, so that for example one can enter a newline as '\n'. When you really need a '\', you have to escape it with another '\'.

Thus, in filenames use something like `"c:\\data\\money.dat"`. You can also replace '\' by '/' (`"c:/data/money.dat"`).

7.9 Why does plotting give a color allocation error?

On an X11 device, plotting sometimes, e.g., when running `demo("image")`, results in "Error: color allocation error". This is an X problem, and only indirectly related to R. It occurs when applications started prior to R have used all the available colors. (How many colors are available depends on the X configuration; sometimes only 256 colors can be used.)

One application which is notorious for "eating" colors is Netscape. If the problem occurs when Netscape is running, try (re)starting it with either the '`-no-install`' (to use the default colormap) or the '`-install`' (to install a private colormap) option.

You could also set the `colortype` of `X11()` to `"pseudo.cube"` rather than the default `"pseudo"`. See the help page for `X11()` for more information.

7.10 How do I convert factors to numeric?

It may happen that when reading numeric data into R (usually, when reading in a file), they come in as factors. If `f` is such a factor object, you can use

```
as.numeric(as.character(f))
```

to get the numbers back. More efficient, but harder to remember, is

```
as.numeric(levels(f))[as.integer(f)]
```

In any case, do not call `as.numeric()` or their likes directly for the task at hand (as `as.numeric()` or `unclass()` give the internal codes).

7.11 Are Trellis displays implemented in R?

The recommended package **lattice** (which is based on another recommended package, **grid**) provides graphical functionality that is compatible with most Trellis commands.

You could also look at `coplot()` and `dotchart()` which might do at least some of what you want. Note also that the R version of `pairs()` is fairly general and provides most of the functionality of `splom()`, and that R's default plot method has an argument `asp` allowing to specify (and fix against device resizing) the aspect ratio of the plot.

(Because the word "Trellis" has been claimed as a trademark we do not use it in R. The name "lattice" has been chosen for the R equivalent.)

7.12 What are the enclosing and parent environments?

Inside a function you may want to access variables in two additional environments: the one that the function was defined in ("enclosing"), and the one it was invoked in ("parent").

If you create a function at the command line or load it in a package its enclosing environment is the global workspace. If you define a function `f()` inside another function `g()` its enclosing environment is the environment inside `g()`. The enclosing environment for a function is fixed when the function is created. You can find out the enclosing environment for a function `f()` using `environment(f)`.

The "parent" environment, on the other hand, is defined when you invoke a function. If you invoke `lm()` at the command line its parent environment is the global workspace, if you invoke it inside a function `f()` then its parent environment is the environment inside `f()`. You can find out the parent environment for an invocation of a function by using `parent.frame()` or `sys.frame(sys.parent())`.

So for most user-visible functions the enclosing environment will be the global workspace, since that is where most functions are defined. The parent environment will be wherever the function happens to be called from. If a function `f()` is defined inside another function `g()` it will probably be used inside `g()` as well, so its parent environment and enclosing environment will probably be the same.

Parent environments are important because things like model formulas need to be evaluated in the environment the function was called from, since that's where all the variables will be available. This relies on the parent environment being potentially different with each invocation.

Enclosing environments are important because a function can use variables in the enclosing environment to share information with other functions or with other invocations of itself (see the section on lexical scoping). This relies on the enclosing environment being the same each time the function is invoked. (In C this would be done with static variables.)

Scoping *is* hard. Looking at examples helps. It is particularly instructive to look at examples that work differently in R and S and try to see why they differ. One way to describe the scoping differences between R and S is to say that in S the enclosing environment is *always* the global workspace, but in R the enclosing environment is wherever the function was created.

7.13 How can I substitute into a plot label?

Often, it is desired to use the value of an R object in a plot label, e.g., a title. This is easily accomplished using `paste()` if the label is a simple character string, but not always obvious in case the label is an expression (for refined mathematical annotation). In such a case, either use `parse()` on your pasted character string or use `substitute()` on an expression. For example, if `ahat` is an estimator of your parameter a of interest, use

```
title(substitute(hat(a) == ahat, list(ahat = ahat)))
```

(note that it is '==' and not '='). Sometimes `bquote()` gives a more compact form, e.g.,

```
    title(bquote(hat(a) = .(ahat)))
```
where subexpressions enclosed in '.()' are replaced by their values.

There are more worked examples in the mailing list achives.

7.14 What are valid names?

When creating data frames using `data.frame()` or `read.table()`, R by default ensures that the variable names are syntactically valid. (The argument 'check.names' to these functions controls whether variable names are checked and adjusted by `make.names()` if needed.)

To understand what names are "valid", one needs to take into account that the term "name" is used in several different (but related) ways in the language:

1. A *syntactic name* is a string the parser interprets as this type of expression. It consists of letters, numbers, and the dot and (for version of R at least 1.9.0) underscore characters, and starts with either a letter or a dot not followed by a number. Reserved words are not syntactic names.

2. An *object name* is a string associated with an object that is assigned in an expression either by having the object name on the left of an assignment operation or as an argument to the `assign()` function. It is usually a syntactic name as well, but can be any non-empty string if it is quoted (and it is always quoted in the call to `assign()`).

3. An *argument name* is what appears to the left of the equals sign when supplying an argument in a function call (for example, `f(trim=.5)`). Argument names are also usually syntactic names, but again can be anything if they are quoted.

4. An *element name* is a string that identifies a piece of an object (a component of a list, for example.) When it is used on the right of the '$' operator, it must be a syntactic name, or quoted. Otherwise, element names can be any strings. (When an object is used as a database, as in a call to `eval()` or `attach()`, the element names become object names.)

5. Finally, a *file name* is a string identifying a file in the operating system for reading, writing, etc. It really has nothing much to do with names in the language, but it is traditional to call these strings file "names".

7.15 Are GAMs implemented in R?

Package **gam** from CRAN implements all the Generalized Additive Models (GAM) functionality as described in the GAM chapter of the White Book. In particular, it implements backfitting with both local regression and smoothing splines, and is extendable. There is a `gam()` function for GAMs in package **mgcv**, but it is not an exact clone of what is described in the White Book (no `lo()` for example). Package **gss** can fit spline-based GAMs too. And if you can accept regression splines you can use `glm()`. For gaussian GAMs you can use `bruto()` from package **mda**.

7.16 Why is the output not printed when I source() a file?

Most R commands do not generate any output. The command

```
1+1
```

computes the value 2 and returns it; the command

```
summary(glm(y~x+z, family=binomial))
```

fits a logistic regression model, computes some summary information and returns an object of class `"summary.glm"` (see Section 8.1 [How should I write summary methods?], page 82).

If you type '1+1' or 'summary(glm(y~x+z, family=binomial))' at the command line the returned value is automatically printed (unless it is `invisible()`), but in other circumstances, such as in a `source()`d file or inside a function it isn't printed unless you specifically print it.

To print the value use

```
print(1+1)
```

or

```
print(summary(glm(y~x+z, family=binomial)))
```

instead, or use `source(file, echo=TRUE)`.

7.17 Why does outer() behave strangely with my function?

As the help for `outer()` indicates, it does not work on arbitrary functions the way the `apply()` family does. It requires functions that are vectorized to work elementwise on arrays. As you can see by looking at the code, `outer(x, y, FUN)` creates two large vectors containing every possible combination of elements of `x` and `y` and then passes this to `FUN` all at once. Your function probably cannot handle two large vectors as parameters.

If you have a function that cannot handle two vectors but can handle two scalars, then you can still use `outer()` but you will need to wrap your function up first, to simulate vectorized behavior. Suppose your function is

```
foo <- function(x, y, happy) {
  stopifnot(length(x) == 1, length(y) == 1) # scalars only!
  (x + y) * happy
}
```

If you define the general function

```
wrapper <- function(x, y, my.fun, ...) {
  sapply(seq(along = x), FUN = function(i) my.fun(x[i], y[i], ...))
}
```

then you can use `outer()` by writing, e.g.,

```
outer(1:4, 1:2, FUN = wrapper, my.fun = foo, happy = 10)
```

7.18 Why does the output from anova() depend on the order of factors in the model?

In a model such as `~A+B+A:B`, R will report the difference in sums of squares between the models `~1`, `~A`, `~A+B` and `~A+B+A:B`. If the model were `~B+A+A:B`, R would report differences between `~1`, `~B`, `~A+B`, and `~A+B+A:B` . In the first case the sum of squares for `A` is comparing `~1` and `~A`, in the second case it is comparing `~B` and `~B+A`. In a non-orthogonal design (i.e., most unbalanced designs) these comparisons are (conceptually and numerically) different.

Some packages report instead the sums of squares based on comparing the full model to the models with each factor removed one at a time (the famous 'Type III sums of squares'

from SAS, for example). These do not depend on the order of factors in the model. The question of which set of sums of squares is the Right Thing provokes low-level holy wars on R-help from time to time.

There is no need to be agitated about the particular sums of squares that R reports. You can compute your favorite sums of squares quite easily. Any two models can be compared with `anova(`*`model1, model2`*`)`, and `drop1(`*`model1`*`)` will show the sums of squares resulting from dropping single terms.

7.19 How do I produce PNG graphics in batch mode?

Under Unix, the `png()` device uses the X11 driver, which is a problem in batch mode or for remote operation. If you have Ghostscript you can use `bitmap()`, which produces a PostScript file then converts it to any bitmap format supported by Ghostscript. On some installations this produces ugly output, on others it is perfectly satisfactory. In theory one could also use Xvfb from X.Org, which is an X11 server that does not require a screen; and the **GDD** package from CRAN, which produces PNG, JPEG and GIF bitmaps without X11.

7.20 How can I get command line editing to work?

The Unix command-line interface to R can only provide the inbuilt command line editor which allows recall, editing and re-submission of prior commands provided that the GNU readline library is available at the time R is configured for compilation. Note that the 'development' version of readline including the appropriate headers is needed: users of Linux binary distributions will need to install packages such as `libreadline-dev` (Debian) or `readline-devel` (Red Hat).

7.21 How can I turn a string into a variable?

If you have

```
varname <- c("a", "b", "d")
```

you can do

```
get(varname[1]) + 2
```

for

```
a + 2
```

or

```
assign(varname[1], 2 + 2)
```

for

```
a <- 2 + 2
```

or

```
eval(substitute(lm(y ~ x + variable),
                list(variable = as.name(varname[1]))
```

for

```
lm(y ~ x + a)
```

At least in the first two cases it is often easier to just use a list, and then you can easily index it by name

```
vars <- list(a = 1:10, b = rnorm(100), d = LETTERS)
vars[["a"]]
```

without any of this messing about.

7.22 Why do lattice/trellis graphics not work?

The most likely reason is that you forgot to tell R to display the graph. Lattice functions
such as `xyplot()` create a graph object, but do not display it (the same is true of Trellis
graphics in S-Plus). The `print()` method for the graph object produces the actual display.
When you use these functions interactively at the command line, the result is automatically
printed, but in `source()` or inside your own functions you will need an explicit `print()`
statement.

7.23 How can I sort the rows of a data frame?

To sort the rows within a data frame, with respect to the values in one or more of the
columns, simply use `order()`.

7.24 Why does the help.start() search engine not work?

The browser-based search engine in `help.start()` utilizes a Java applet. In order for this
to function properly, a compatible version of Java must installed on your system and linked
to your browser, and both Java *and* JavaScript need to be enabled in your browser.

There have been a number of compatibility issues with versions of Java and of browsers.
For further details please consult section "Enabling search in HTML help" in *R Installation
and Administration*. This manual is included in the R distribution, see Section 2.7 [What
documentation exists for R?], page 5, and its HTML version is linked from the HTML search
page.

7.25 Why did my .Rprofile stop working when I updated R?

Did you read the 'NEWS' file? For functions that are not in the **base** package you need to
specify the correct package namespace, since the code will be run *before* the packages are
loaded. E.g.,

```
ps.options(horizontal = FALSE)
help.start()
```

needs to be

```
grDevices::ps.options(horizontal = FALSE)
utils::help.start()
```

(`graphics::ps.options(horizontal = FALSE)` in R 1.9.x).

7.26 Where have all the methods gone?

Many functions, particularly S3 methods, are now hidden in namespaces. This has the
advantage that they cannot be called inadvertently with arguments of the wrong class, but
it makes them harder to view.

To see the code for an S3 method (e.g., `[.terms`) use

```
getS3method("[", "terms")
```

To see the code for an unexported function `foo()` in the namespace of package `"bar"` use `bar:::foo`. Don't use these constructions to call unexported functions in your own code—they are probably unexported for a reason and may change without warning.

7.27 How can I create rotated axis labels?

To rotate axis labels (using base graphics), you need to use `text()`, rather than `mtext()`, as the latter does not support `par("srt")`.

```
## Increase bottom margin to make room for rotated labels
par(mar = c(7, 4, 4, 2) + 0.1)
## Create plot with no x axis and no x axis label
plot(1 : 8, xaxt = "n",  xlab = "")
## Set up x axis with tick marks alone
axis(1, labels = FALSE)
## Create some text labels
labels <- paste("Label", 1:8, sep = " ")
## Plot x axis labels at default tick marks
text(1:8, par("usr")[3] - 0.25, srt = 45, adj = 1,
     labels = labels, xpd = TRUE)
## Plot x axis label at line 6 (of 7)
mtext(1, text = "X Axis Label", line = 6)
```

When plotting the x axis labels, we use `srt = 45` for text rotation angle, `adj = 1` to place the right end of text at the tick marks, and `xpd = TRUE` to allow for text outside the plot region. You can adjust the value of the `0.25` offset as required to move the axis labels up or down relative to the x axis. See `?par` for more information.

Also see Figure 1 and associated code in Paul Murrell (2003), "Integrating grid Graphics Output with Base Graphics Output", *R News*, **3/2**, 7–12.

7.28 Why is read.table() so inefficient?

By default, `read.table()` needs to read in everything as character data, and then try to figure out which variables to convert to numerics or factors. For a large data set, this takes condiderable amounts of time and memory. Performance can substantially be improved by using the `colClasses` argument to specify the classes to be assumed for the columns of the table.

7.29 What is the difference between package and library?

A *package* is a standardized collection of material extending R, e.g. providing code, data, or documentation. A *library* is a place (directory) where R knows to find packages it can use (i.e., which were *installed*). R is told to use a package (to "load" it and add it to the search path) via calls to the function `library`. I.e., `library()` is employed to load a package from libraries containing packages.

See Chapter 5 [R Add-On Packages], page 21, for more details. See also Uwe Ligges (2003), "R Help Desk: Package Management", *R News*, **3/3**, 37–39.

7.30 I installed a package but the functions are not there

To actually *use* the package, it needs to be *loaded* using `library()`.

See Chapter 5 [R Add-On Packages], page 21 and Section 7.29 [What is the difference between package and library?], page 79 for more information.

7.31 Why doesn't R think these numbers are equal?

The only numbers that can be represented exactly in R's numeric type are integers and fractions whose denominator is a power of 2. Other numbers have to be rounded to (typically) 53 binary digits accuracy. As a result, two floating point numbers will not reliably be equal unless they have been computed by the same algorithm, and not always even then. For example

```
R> a <- sqrt(2)
R> a * a == 2
[1] FALSE
R> a * a - 2
[1] 4.440892e-16
```

The function `all.equal()` compares two objects using a numeric tolerance of `.Machine$double.eps ^ 0.5`. If you want much greater accuracy than this you will need to consider error propagation carefully.

For more information, see e.g. David Goldberg (1991), "What Every Computer Scientist Should Know About Floating-Point Arithmetic", *ACM Computing Surveys*, **23/1**, 5–48, also available via http://docs.sun.com/source/806-3568/ncg_goldberg.html.

7.32 How can I capture or ignore errors in a long simulation?

Use `try()`, which returns an object of class `"try-error"` instead of an error, or preferably `tryCatch()`, where the return value can be configured more flexibly. For example

```
beta[i,] <- tryCatch(coef(lm(formula, data)),
                     error = function(e) rep(NaN, 4))
```

would return the coefficients if the `lm()` call succeeded and would return `c(NaN, NaN, NaN, NaN)` if it failed (presumably there are supposed to be 4 coefficients in this example).

7.33 Why are powers of negative numbers wrong?

You are probably seeing something like

```
R> -2^2
[1] -4
```

and misunderstanding the precedence rules for expressions in R. Write

```
R> (-2)^2
[1] 4
```

to get the square of −2.

The precedence rules are documented in `?Syntax`, and to see how R interprets an expression you can look at the parse tree

```
R> as.list(quote(-2^2))
[[1]]
`-`


[[2]]
2^2
```

7.34 How can I save the result of each iteration in a loop into a separate file?

One way is to use `paste()` (or `sprintf()`) to concatenate a stem filename and the iteration number while `file.path()` constructs the path. For example, to save results into files 'result1.rda', ..., 'result100.rda' in the subdirectory 'Results' of the current working directory, one can use

```
for(i in 1:100) {
  ## Calculations constructing "some_object" ...
  fp <- file.path("Results", paste("result", i, ".rda", sep = ""))
  save(list = "some_object", file = fp)
}
```

8 R Programming

8.1 How should I write summary methods?

Suppose you want to provide a summary method for class `"foo"`. Then `summary.foo()` should not print anything, but return an object of class `"summary.foo"`, *and* you should write a method `print.summary.foo()` which nicely prints the summary information and invisibly returns its object. This approach is preferred over having `summary.foo()` print summary information and return something useful, as sometimes you need to grab something computed by `summary()` inside a function or similar. In such cases you don't want anything printed.

8.2 How can I debug dynamically loaded code?

Roughly speaking, you need to start R inside the debugger, load the code, send an interrupt, and then set the required breakpoints.

See section "Finding entry points in dynamically loaded code" in *Writing R Extensions*. This manual is included in the R distribution, see Section 2.7 [What documentation exists for R?], page 5.

8.3 How can I inspect R objects when debugging?

The most convenient way is to call `R_PV` from the symbolic debugger.

See section "Inspecting R objects when debugging" in *Writing R Extensions*.

8.4 How can I change compilation flags?

Suppose you have C code file for dynloading into R, but you want to use `R CMD SHLIB` with compilation flags other than the default ones (which were determined when R was built).

Starting with R 2.1.0, users can provide personal Makevars configuration files in '`$HOME/.R`' to override the default flags. See section "Add-on packages" in *R Installation and Administration*.

For earlier versions of R, you could change the file '`$R_HOME/etc/Makeconf`' to reflect your preferences, or (at least for systems using GNU Make) override them by the environment variable `MAKEFLAGS`. See section "Creating shared objects" in *Writing R Extensions*.

8.5 How can I debug S4 methods?

Use the `trace()` function with argument `signature=` to add calls to the browser or any other code to the method that will be dispatched for the corresponding signature. See `?trace` for details.

9 R Bugs

9.1 What is a bug?

If R executes an illegal instruction, or dies with an operating system error message that indicates a problem in the program (as opposed to something like "disk full"), then it is certainly a bug. If you call `.C()`, `.Fortran()`, `.External()` or `.Call()` (or `.Internal()`) yourself (or in a function you wrote), you can always crash R by using wrong argument types (modes). This is not a bug.

Taking forever to complete a command can be a bug, but you must make certain that it was really R's fault. Some commands simply take a long time. If the input was such that you *know* it should have been processed quickly, report a bug. If you don't know whether the command should take a long time, find out by looking in the manual or by asking for assistance.

If a command you are familiar with causes an R error message in a case where its usual definition ought to be reasonable, it is probably a bug. If a command does the wrong thing, that is a bug. But be sure you know for certain what it ought to have done. If you aren't familiar with the command, or don't know for certain how the command is supposed to work, then it might actually be working right. Rather than jumping to conclusions, show the problem to someone who knows for certain.

Finally, a command's intended definition may not be best for statistical analysis. This is a very important sort of problem, but it is also a matter of judgment. Also, it is easy to come to such a conclusion out of ignorance of some of the existing features. It is probably best not to complain about such a problem until you have checked the documentation in the usual ways, feel confident that you understand it, and know for certain that what you want is not available. If you are not sure what the command is supposed to do after a careful reading of the manual this indicates a bug in the manual. The manual's job is to make everything clear. It is just as important to report documentation bugs as program bugs. However, we know that the introductory documentation is seriously inadequate, so you don't need to report this.

If the online argument list of a function disagrees with the manual, one of them must be wrong, so report the bug.

9.2 How to report a bug

When you decide that there is a bug, it is important to report it and to report it in a way which is useful. What is most useful is an exact description of what commands you type, starting with the shell command to run R, until the problem happens. Always include the version of R, machine, and operating system that you are using; type *version* in R to print this.

The most important principle in reporting a bug is to report *facts*, not hypotheses or categorizations. It is always easier to report the facts, but people seem to prefer to strain to posit explanations and report them instead. If the explanations are based on guesses about how R is implemented, they will be useless; others will have to try to figure out what the facts must have been to lead to such speculations. Sometimes this is impossible. But in any case, it is unnecessary work for the ones trying to fix the problem.

For example, suppose that on a data set which you know to be quite large the command

```
R> data.frame(x, y, z, monday, tuesday)
```

never returns. Do not report that `data.frame()` fails for large data sets. Perhaps it fails when a variable name is a day of the week. If this is so then when others got your report they would try out the `data.frame()` command on a large data set, probably with no day of the week variable name, and not see any problem. There is no way in the world that others could guess that they should try a day of the week variable name.

Or perhaps the command fails because the last command you used was a method for `"["()` that had a bug causing R's internal data structures to be corrupted and making the `data.frame()` command fail from then on. This is why others need to know what other commands you have typed (or read from your startup file).

It is very useful to try and find simple examples that produce apparently the same bug, and somewhat useful to find simple examples that might be expected to produce the bug but actually do not. If you want to debug the problem and find exactly what caused it, that is wonderful. You should still report the facts as well as any explanations or solutions. Please include an example that reproduces the problem, preferably the simplest one you have found.

Invoking R with the '`--vanilla`' option may help in isolating a bug. This ensures that the site profile and saved data files are not read.

Before you actually submit a bug report, you should check whether the bug has already been reported and/or fixed. First, try the "Search Existing Reports" facility in the Bug Tracking page at http://bugs.R-project.org/. Second, consult https://svn.R-project.org/R/trunk/NEWS, which records changes that will appear in the *next* release of R, including some bug fixes that do not appear in Bug Tracking. (Windows users should additionally consult https://svn.R-project.org/R/trunk/src/gnuwin32/CHANGES.) Third, if possible try the current r-patched or r-devel version of R. If a bug has already been reported or fixed, please do not submit further bug reports on it. Finally, check carefully whether the bug is with R, or a contributed package. Bug reports on contributed packages should be sent first to the package maintainer, and only submitted to the R-bugs repository by package maintainers, mentioning the package in the subject line.

On Unix systems a bug report can be generated using the function `bug.report()`. This automatically includes the version information and sends the bug to the correct address. Alternatively the bug report can be emailed to R-bugs@R-project.org or submitted to the Web page at http://bugs.R-project.org/. Please try including results of `sessionInfo()` in your bug report.

There is a section of the bug repository for suggestions for enhancements for R labelled '`wishlist`'. Suggestions can be submitted in the same ways as bugs, but please ensure that the subject line makes clear that this is for the wishlist and not a bug report, for example by starting with '`Wishlist:`'.

Comments on and suggestions for the Windows port of R should be sent to R-windows@R-project.org.

Corrections to and comments on message translation should be sent to the last translator (listed at the top of the appropriate '`.po`' file) or to the translation team as listed at http://developer.R-project.org/TranslationTeams.html.

10 Acknowledgments

Of course, many many thanks to Robert and Ross for the R system, and to the package writers and porters for adding to it.

Special thanks go to Doug Bates, Peter Dalgaard, Paul Gilbert, Stefano Iacus, Fritz Leisch, Jim Lindsey, Thomas Lumley, Martin Maechler, Brian D. Ripley, Anthony Rossini, and Andreas Weingessel for their comments which helped me improve this FAQ.

More to come soon . . .