Part 1: Getting Started

140.776 Statistical Computing

Ingo Ruczinski

Thanks to Thomas Lumley and Robert Gentleman of the R-core group (http://www.r-project.org/) for providing some tex files that appear in part in the following slides.

Some S facts

- S is a language and system for organizing, visualizing, and analyzing data.
- S has been a project of statistics research at Bell Labs since 1976.
- The language has evolved through several major versions to become the most widely used environment for research in data analysis and statistics.
- In 1998, S became the first statistical system to receive the Software System Award, the top software award from the ACM.

Products and projects based on S

- The S-Plus language is based on the S software from Bell Labs. S-Plus products are distributed by the Insightful Corporation, which has an exclusive license to distribute software based on Bell Labs' S.

- The R language is an open-source system distributed under the GPL license. It is a separate project based on the S language, with a number of differences to Splus.

- Omegahat is also an open-source project for development of ("next-generation") software for statistical applications, with emphasis on web-based software.

- Bioconductor is an open source software project for the analysis and comprehension of genomic data.

Some R facts

- R is an environment for data analysis and visualization.

- R is an open source implementation of the S language (S-Plus is a commercial implementation of the S language).

- The current version of R (September 2004) is 1.9.1.

- The R Core group consists of Doug Bates, John Chambers, Peter Dalgaard, Robert Gentleman, Kurt Hornik, Stefano Iacus, Ross Ihaka, Friedrich Leisch, Thomas Lumley, Martin Maechler, Guido Masarotto, Paul Murrell, Brian Ripley, Duncan Temple Lang, and Luke Tierney.

- If you use R extensively, be a good citizen and join the R Foundation for Statistical Computing http://www.r-project.org/ .

The R history

1991	Ross Ihaka and Robert Gentleman begin work on a project that will ultimately become R.
1992	Design and implementation of pre-R.
1993	The first announcement of R.
1995	R available by ftp under the GPL.
1996	A mailing list is started and maintained by Martin Maechler at ETH.
1997	The R core group is formed.
1999	DSC meeting in Vienna, the first time many R core members meet.
2000	R 1.0.0 is released.
2004	R 1.9.1 is the current release.

The R license

- R is both open source and open development.
- You can look at the source code and you can propose changes.
- R is not in the public domain.
- You are given a license to run the software (currently GPL).

The R software

- R is mainly written in C.
- R is available for many platforms:
  - Unix of many flavors, including Linux, Solaris, FreeBSD, AIX.
  - Windows 95 and later.
  - MacOS X.
- Binaries and source code are available from www.r-project.org .
- R "talks" to data bases, programming languages, and other statistical packages.
- R should be source code compatible with most of the Splus code written.

How do I get R?

- The informational web site http://www.r-project.org/ .
- CRAN - the Comprehensive R Archive Network:
  - The primary site is http://cran.r-project.org/ .
  - Mirror sites are available for many countries. For example: http://cran.us.r-project.org/ .
- CRAN sites have source code andbinary distributions for Windows 95, 98, ME, NT4, 2000 and XP on Intel and compatible processor, for the Macintosh (System 8.6 to 9.1 and MacOS X), and for several Linux distributions.
- New releases occur frequently, at least twice a year. Be prepared to re-install frequently.

Installing R

- Windows:
  Download and run the `SetupR.exe` installer.

- Macintosh:
  Download `R.dmg`, and double-click on the `R-1.9.1.pkg` icon on the `R.dmg` disk image.

- Linux:
  RPM files are available for RedHat, SuSE, and Mandrake. Deb files are available for Debian.

- Unix/Linux:
  Download and expand the compressed tar file of the sources. Run `./configure`, and then `make`, `make check`, and `make install`.

Installing R with a script

Create the directory `R-lr` in your root directory (for example, mine is `/home/iruczins/`). Edit the following script and copy it to your `bin/` directory.

```
rsync -rC rsync.r-project.org::r-release /home/iruczins/R-lr
cd /home/iruczins/R-lr
./configure --prefix=/home/iruczins/R-lr
make
make install
echo Done!
```

Make sure you can execute it (`chmod 755 [filename]` will do). When you run it, `rsync` will automatically grab the latest released R version for you.

Create an alias in your `.tcshrc` file, or whatever shell you use:

```
alias R '/home/iruczins/R-lr/bin/R'
```

Sources of information about R

- The web site http://www.r-project.org/ and CRAN.
- Kurt Hornik's page http://www.ci.tuwien.ac.at/~hornik/R/R-FAQ.html mirrored at http://cran.r-project.org/doc/FAQ/R-FAQ.html . Most of the contents of these slides (and many other topics) are covered in this FAQ site.
- The manuals at http://cran.r-project.org/manuals.html . In particular, check out `R-intro.pdf` and `R-data.pdf`.
- Karl's R page at http://www.biostat.jhsph.edu/~kbroman/Rintro/
- More detailled notes are on the Statistical Computing class page at http://www.biostat.jhsph.edu/~bcaffo/statcomp/

A sample session

To start a session, type `R` at the prompt or click on the GUI.

```
> log(64)
[1] 4.158883

> log2(64)
[1] 6

> sqrt(2)
[1] 1.414214

> sqrt(-1)
[1] NaN
Warning message:
NaNs produced in: sqrt(-1)

> sqrt(-1+0i)
[1] 0+1i
```

A sample session

```
> x <- 5
> x
[1] 5

> x = 5
> x
[1] 5

> x <- c(1,2,3,4)
> x
[1] 1 2 3 4

> x <- 1:4
> x
[1] 1 2 3 4

> x <- seq(1,4)
> x
[1] 1 2 3 4
```

A sample session

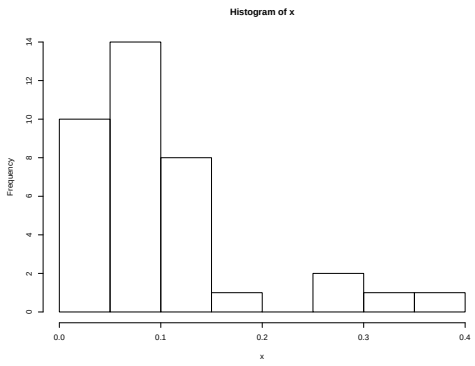
```
> x <- c(0.008, 0.018, 0.056, 0.055, 0.135,
+        0.052, 0.077, 0.026, 0.044, 0.300,
+        0.025, 0.036, 0.043, 0.100, 0.120,
+        0.110, 0.100, 0.350, 0.100, 0.300,
+        0.011, 0.060, 0.070, 0.050, 0.080,
+        0.110, 0.110, 0.120, 0.133, 0.100,
+        0.100, 0.155, 0.370, 0.019, 0.100,
+        0.100, 0.116)

> x
 [1] 0.008 0.018 0.056 0.055 0.135 0.052 0.077 0.026 0.044
[10] 0.300 0.025 0.036 0.043 0.100 0.120 0.110 0.100 0.350
[19] 0.100 0.300 0.011 0.060 0.070 0.050 0.080 0.110 0.110
[28] 0.120 0.133 0.100 0.100 0.155 0.370 0.019 0.100 0.100
[37] 0.116

> class(x)
[1] "numeric"

> length(x)
[1] 37
```

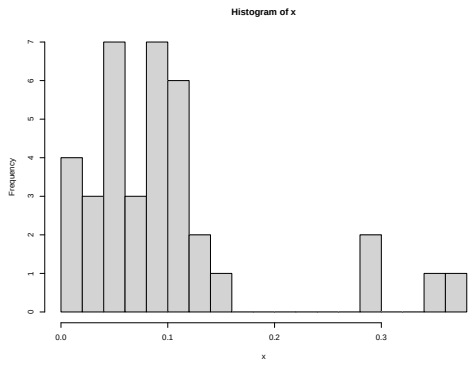
A sample session

```
> summary(x)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0080  0.0500  0.1000  0.1043  0.1160  0.3700

> quantile(x, c(0.1, 0.95))
   10%    95%
0.0226 0.3100

> mean(x);median(x);sd(x);var(x)
[1] 0.1042973
[1] 0.1
[1] 0.08895344
[1] 0.007912715

> round(c(mean(x),median(x),sd(x),var(x)),3)
[1] 0.104 0.100 0.089 0.008
```

A sample session

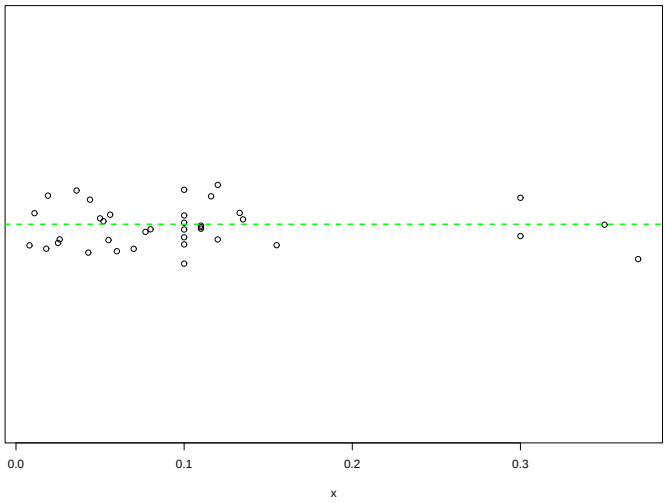
```
> exp(mean(log(x)))
[1] 0.07463571

> 1/mean(1/x)
[1] 0.04841714

> hist(x)
> hist(x,breaks=15,col="lightgrey")
```





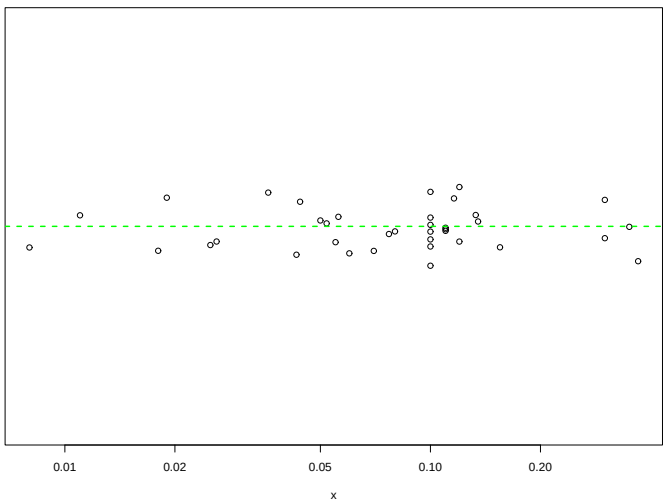
A sample session

```
> y <- runif(length(x))
> plot(x, y, ylim=c(-2,3), yaxt="n", ylab="")
> abline(h=0.5, lty=2, col="green",lwd=2)
```



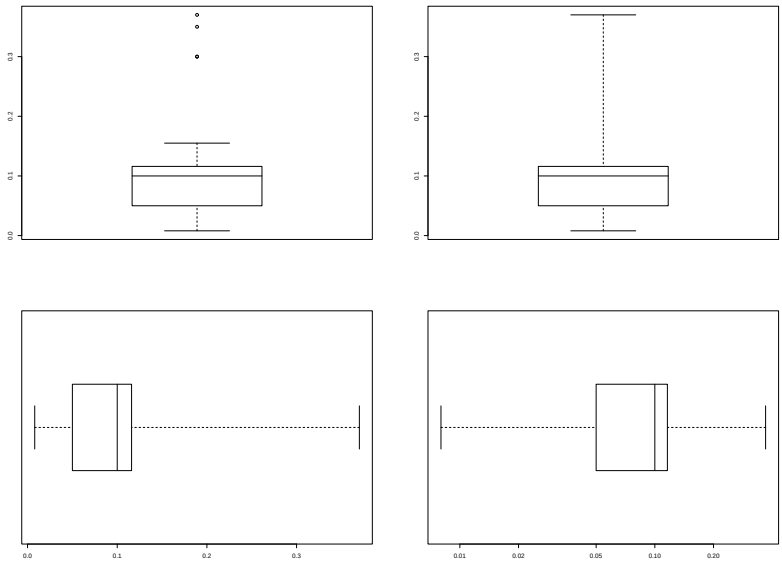
A sample session

```
> plot(x, y, ylim=c(-2,3), yaxt="n", ylab="", log="x")
> abline(h=0.5, lty=2, col="green",lwd=2)
```



A sample session

```
> boxplot(x)
> boxplot(x, range=0)
> boxplot(x, range=0, horizontal=TRUE)
> boxplot(x, range=0, horizontal=TRUE, log="x")
```



A sample session

```
> x <- 1:4
> x
[1] 1 2 3 4

> x*x
[1]  1  4  9 16

> z <- x %*% x
> z
     [,1]
[1,]   30

> drop(z)
[1] 30
```

A sample session

```
> y <- diag(x)
> y
     [,1] [,2] [,3] [,4]
[1,]    1    0    0    0
[2,]    0    2    0    0
[3,]    0    0    3    0
[4,]    0    0    0    4

> solve(y)
     [,1] [,2]      [,3] [,4]
[1,]    1  0.0 0.0000000 0.00
[2,]    0  0.5 0.0000000 0.00
[3,]    0  0.0 0.3333333 0.00
[4,]    0  0.0 0.0000000 0.25

> det(y)
[1] 24
```

A sample session

```
> z <- matrix(sample(1:12), ncol = 3, nrow = 4)
> z
     [,1] [,2] [,3]
[1,]    7   12    8
[2,]    4    5    9
[3,]    2    1    6
[4,]   10   11    3

> t(z)
     [,1] [,2] [,3] [,4]
[1,]    7    4    2   10
[2,]   12    5    1   11
[3,]    8    9    6    3
```

A sample session

```
> y %*% z
     [,1] [,2] [,3]
[1,]    7   12    8
[2,]    8   10   18
[3,]    6    3   18
[4,]   40   44   12

> y %*% x
     [,1]
[1,]    1
[2,]    4
[3,]    9
[4,]   16

> x %*% z
     [,1] [,2] [,3]
[1,]   61   69   56
```

A sample session

```
> solve(t(z)%*%z) %*% (t(z)%*%x)

             [,1]
[1,]  1.1210197
[2,] -0.6962714
[3,]  0.1637485

> A <- t(z)%*%z
> b <- t(z)%*%x
> solve(A,b)
           [,1]
[1,]  1.1210197
[2,] -0.6962714
[3,]  0.1637485
```

Getting help

- Check the help files and manuals.
  - For example, to check out how the function `weighted.mean()` works, type `?weighted.mean` or `help(weighted.mean)` at the R prompt.
  - If you do not know the exact name of the function, type for example `help.search(''mean'')` or use the html search engine, which you can start by typing `help.start()`.
- Ask some R wizard.
- Use the mailing list archives and search facilities.
- Post your question to the mailing list.

The R package system

- Packages are self-contained units of code with documentation.
- The packages are simple to obtain, understand, and update. Try commands like `install.packages()`, `example()`, and `update.packages()`.
- All functions must have examples and the examples must run.
- There are automatic testing features built in.
- You can write your own packages ($\rightarrow$ Rafa will talk about this).

Downloading packages and bundles

```
> install.packages("rpart")
> library(rpart)
> data(kyphosis)

> kyphosis
   Kyphosis Age Number Start
1    absent  71      3     5
2    absent 158      3    14
3   present 128      4     5
4    absent   2      5     1
5    absent   1      4    15
6    absent   1      2    16
7    absent  61      2    17
8    absent  37      3    16
9    absent 113      2    16
10  present  59      6    12
11  present  82      5    14
12   absent 148      3    16
13   absent  18      5     2
...
```

Downloading packages and bundles

```
> install.packages("VR")
> library(MASS)
> data(hills)
> ?hills
hills                  package:MASS                  R Documentation

Record Times in Scottish Hill Races

Description:
     The record times in 1984 for 35 Scottish hill races.

Usage:
     data(hills)

Format:
     The components are:
     'dist' distance in miles (on the map)
     'climb' total height gained during the route, in feet.
     'time' record time in minutes.
```

9 basic R functions you should know

Typing a function name and hitting the <RET> button simply displays the function. To invoke the function you must include an argument list in `()`, even if the list is empty. That is, use `q()` <RET> and not just `q` <RET>.

- `q` Quit the session
- `help` Get help on a function or object
- `help.start` Allow the use of a web browser for reading help
- `example` Run the example from the help page for an object
- `data` List the available data sets or import a data set
- `library` List available packages or attach a package
- `objects` List the objects in the workspace
- `summary` Summarize an object
- `str` Show the low-level structure of an object