

# R for Scientific Computing

Stephen Eglen

Cambridge Computational Biology Institute  
Department of Applied Mathematics and Theoretical Physics  
University of Cambridge  
<http://www.damtp.cam.ac.uk/user/eglen/>

July 2007

# What is R?

- Computing environment, similar to matlab.
- Very popular in many areas of statistics, computational biology.
- High-level language, based on scheme (lisp dialect) and S. Interfaces to other languages for:
  1. speed.
  2. access to existing code, databases.
- “Programming with data” (Chambers)
- Approach: command-line for one-liners; write scripts/functions for larger work (edit/run cycle).

# History

- S language came from Bell Labs (Becker, Chambers and Wilks). Commercial version S-plus (1988).
- R emerged as a combination of S and Scheme: Ross Ihaka and Robert Gentleman (NZ).
- 1993: first announcement.
- 1995: 0.60 release, now under GPL.
- Jul 2007: release 2.5.1. Stable, multi-platform.
- R-core now 17 people, key academics in field, including John Chambers.

## Brief comparison to matlab

- Flexible language, similar to matlab, but definitely not “everything is a matrix”. Frames, lists, vectors ...
- From matlab to R:  
<http://cran.r-project.org/doc/contrib/R-and-octave.txt>
- Use `x[i]` not `x(i)` for indexing vectors.
- Making vectors: `x <- c(10, 9, 5, 1)`
- Assignment: best to use `<-` rather than `=`. Stay away from underscore!

```
x <- 10
```

```
x = 10    ## equivalent, more readable?
```

```
lo_val <- 100    ## not lo_val <= 100
```

## Strengths of R

- GPL'd, available on many platforms.
- Excellent development team with Apr/Oct release cycle.
- Source always available to examine/edit.
- Fast for vectorized calculations.
- Foreign-language interface (C/Fortran) when speed crucial.
- Good collection of numerical/statistical routines.
- Comprehensive R Archive Network (CRAN)  $\sim$  1100 packages.
- On-line doc, with examples.
- High-quality graphics (pdf, postscript, quartz, x11, bitmaps). Often used just for plotting ...
- Passing arguments to functions is nice ...



## Argument passing (named arguments + lazy evaluation)

```
add.n <- function(x, n=10, keep.first) {  
  res <- x + n  
  if (!missing(keep.first)) {  
    res <- res[1:keep.first]  
  }  
  res  
}  
  
> dat <- c(5, 4, 3, 2, 1)  
> add.n(dat) ## assume defaults for arg 2,3  
[1] 15 14 13 12 11  
> add.n(dat, n=100) ## change arg 2  
[1] 105 104 103 102 101  
> add.n(dat, keep.first=3) ## change arg 3  
[1] 15 14 13  
> add.n(keep=3, x=dat) ## swap order, abbrev  
[1] 15 14 13
```

## Weaknesses of R

- Slow for loops. Learn how to vectorize solutions or use apply family of functions.
- No compiler (yet).
- No decent GUI. Tk is “temporary” solution.

# Using R

- Start-up: type 'R' at command line.
- Type commands interactively, and get results.
- Type commands into a file; **source**('myfile.R'); edit file ...
- Mac/Win has a GUI for interactive use, with internal editors.
- All platforms have a command-line interface
- Many external editors have support for R, including Emacs (<http://ess.r-project.org>).

## Key commands

- **args(ls)** to see arguments of a function.
- **help(ls)** to see help file (or **?ls**).
- **example(boxplot)** run examples in help page.
- **help.start()** starts web-browser for help/ on-line docs.
- **help.search("matrix")**
- **demo()** to list all demos, e.g. **demo(graphics)**
- **q()** to exit.

## Reproducible research: Sweave and vignettes

- Use one file to store code and document. Best shown by way of example... `estimate.Rnw`
- Vignettes often used in Bioconductor to document packages.
- `> library(tkWidgets); vExplorer()`  
Interactively explore vignettes.

## Starting points

- Rnews: 2/year newsletter.
- R-help mailing list: 50–100 messages/day.
- useR meeting: every 2 years.
- Perhaps we can have a local users group within this forum?