The R Project: A Brief History and Thoughts About the Future

Ross Ihaka

The University of Auckland

The R Language and Environment

- R is a computer language and run-time environment which can be used to carry out statistical (or other quantitative) computations.
- The base part of R comes with a wide range of standard statistical and graphical analyses built in.
- There are a large number of user-developed extension *packages* which provide an even richer set of capabilities.

Licensing

- R is free software released under the Free Software Foundation's General Public License.
- This means that R is free of any restrictions on how it can be disseminated.
- Versions of R can be obtained without charge and can be redistributed to others.
- The license prevents the creation of encumbered derived works (i.e. commercial versions).

Uptake

- Because of its license, it is very hard to determine what the installed base of R might be.
- The R development group has confined itself to estimates of the form: “somewhere in excess of 50,000.”
- A recent New York Times article presented the estimates: one million (Intel Capital) and two million (Revolution computing).

The R Language

- R is an expression-based language.
  - Users type language *expressions* at the R prompt.
  - These expressions are *evaluated* by the R *interpreter*..
  - The computed values of the expressions are printed.
- R is extensible.
  - Users can implement new functionality in the form of *functions*.
  - Developers can implement new *packages* of functionality that extends the base system.

An Example

Read a data set into R (from a local file or network URL).

```
> rats = read.csv("rats.csv")
```

An Example

Read a data set into R (from a local file or network URL).

```
> rats = read.csv("rats.csv")
```

Examine the basic stucture of the data.

```
> summary(rats)
   WeightGain          Group
 Min.   :-16.90   Control:23
 1st Qu.: 10.10   Ozone  :22
 Median : 18.30
 Mean   : 16.83
 3rd Qu.: 26.00
 Max.   : 54.60
```

Example (Continued)

```
> with(rats, tapply(WeightGain, Group, mean))
 Control    Ozone
22.40435 11.00909
```

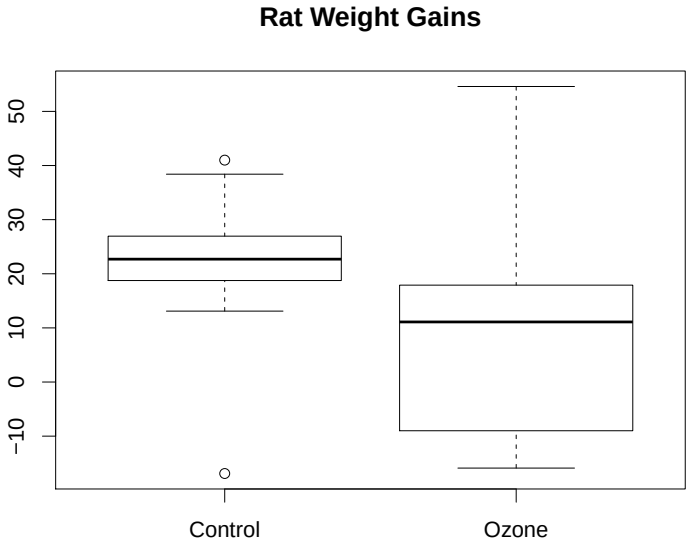
Example (Continued)

```
> with(rats, tapply(WeightGain, Group, mean))
 Control    Ozone
22.40435 11.00909
```

```
> summary(aov(WeightGain ~ Group, data = rats))
            Df  Sum Sq Mean Sq F value  Pr(>F)
Group        1  1460.1 1460.11  6.1872 0.01682
Residuals   43 10147.5  235.99
```

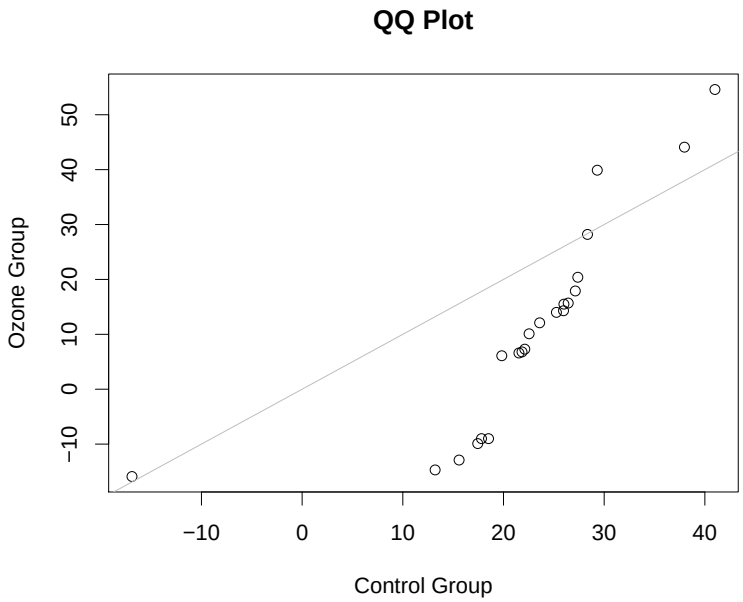
Example (Continued)

```
> with(rats, tapply(WeightGain, Group, mean))
 Control    Ozone
22.40435 11.00909

> summary(aov(WeightGain ~ Group, data = rats))
            Df  Sum Sq Mean Sq F value  Pr(>F)
Group        1  1460.1 1460.11  6.1872 0.01682
Residuals   43 10147.5  235.99

> boxplot(WeightGain ~ Group, data = rats,
          main = "Rat Weight Gains")
```


Rat Weight Gains
−10
0
10
20
30
40
50
Control
Ozone

Example Continued

```
> with(rats,
       qqplot(WeightGain[Group == "Control"],
              WeightGain[Group == "Ozone"],
              main = "QQ Plot",
              xlab = "Control Group",
              ylab = "Ozone Group"))

> abline(0, 1, col = "gray")
```


QQ Plot
Ozone Group
Control Group
−10
0
10
20
30
40
50
−10
0
10
20
30
40

The Empirical Shift Function

The standard two sample t-test makes the hypothesis the the x sample and the y sample have the same distribution up to a constant shift. In other words:

$$F_X(x) = F_Y(x + \Delta).$$

The value Δ measures how much the x distribution must be shifted to obtain the y distribution.

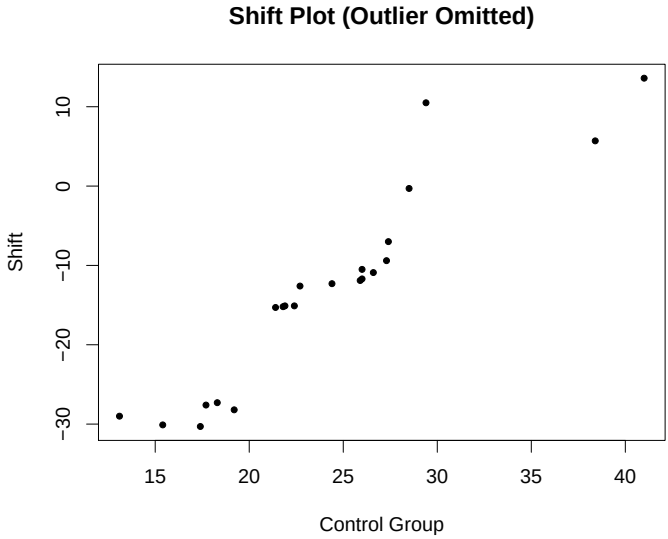
A more general model is the general *shift-function* model:

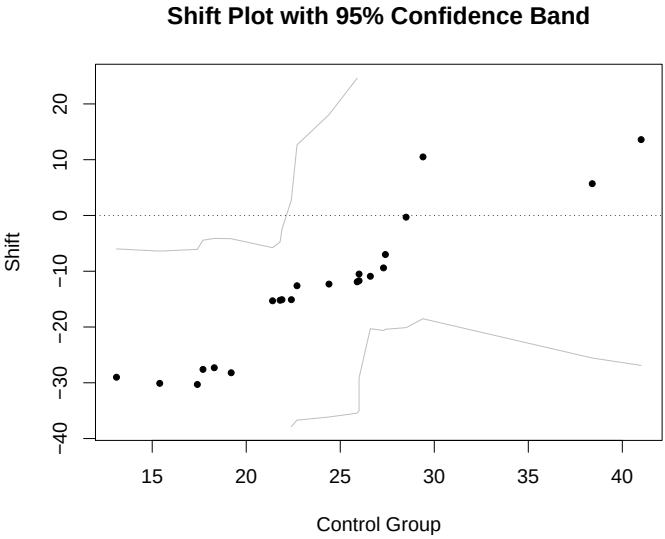
$$F_X(x) = F_Y(x + \Delta(x))$$

where $\Delta(x)$ is a function which indicates the adjustments which must be made to the x distribution to produce the y distribution.

Shift Function Code

```
> quantilefun =
      function(y)
      approxfun(seq(0, 1, length = length(y)),
                sort(y), yleft = NA, yright = NA)

> shiftplot =
      function(x, y, pch = 20, xlab = "x",
               ylab = "Shift", main = NULL, ...)
      {
          x = sort(na.omit(x))
          y = sort(na.omit(y))
          qy = quantilefun(y)
          p = seq(0, 1, length = length(x))
          yq = qy(p) - x
          plot(x, yq, pch = pch,
               xlab = xlab, ylab = ylab,
               main = main, ...)
      }
```


Shift Plot (Outlier Omitted)
Shift
10
0
−10
−20
−30
15
20
25
30
35
40
Control Group


Shift Plot with 95% Confidence Band
Shift
20
10
0
−10
−20
−30
−40
15
20
25
30
35
40
Control Group

Early History - 1990

- Ross Ihaka joins the Department of Statistics at the University of Auckland.
- Robert Gentleman spends sabbatical from the University of Waterloo.
- During a chance encounter in the corridor, the following exchange takes place:

  Gentleman: "Let's write some software."
  Ihaka: "Sure, that sounds like fun."

- The initial goal is to build a testbed for trying out ideas and to publish a paper or two.

The Initial Language

```
> (set x (seq 10))
(1 2 3 4 5 6 7 8 9 10)

> (sum x)
55

> (set factorial (lambda (x)
  (if (< x 1)
      1
      (* x (factorial (- x 1))))))
<closure>

> (factorial 5)
120
```

Early History - 1992

- Robert Gentleman joins the department at Auckland.
- A decision is made to develop enough of a language to teach introductory statistics courses at Auckland.
  - It is decided to adopt the syntax of the S language developed at Bell Laboratories.
  - As a joke, the name “R” is coined for the language (standing for Robert and Ross).

Early History - 1994

- An initial version of the language is complete.
- Colleagues overseas encourage us to release the language as "free software."
- A little thought convinces us that there are limited prospects for the software as a commerical product.
- We adopt the Free Software Foundation GPL as our license and begin to make releases via the internet.
- We start a small email list so that we and our users can discuss R.

The original R developers plotting world domination.

Early History - 1996

- By 1996 we were becoming victims of our own success.
- We were being supplied with a continual stream of bug reports and suggestions for improvement.
- Maintaining the mailing list was becoming problematic.
- It was beginning to be clear that the project was getting close to the limit of what two of us could handle.

Success! - 1997

- The mailing list turned out to be very successful and our user base increased enormously (to nearly 100!).
- The list was so successful that was split into the present `r-help` and `r-devel` lists.
- Kurt Hornik and Fritz Leisch established the CRAN archive at TU Vienna as a repository for user contributions.
- We became so deluged with patches and requests for enhancements that we decided to open up the development process by giving a selected “core” of developers direct access to the CVS archive.

R Becomes A GNU Project

From: Richard Stallman <rms@gnu.ai.mit.edu>
To: ihaka@stat.auckland.ac.nz
cc: rms@gnu.ai.mit.edu
Subject: Re: Seen on your wishlist
Date: Tue, 16 Sep 1997 21:56:06 -0400

> So [explicitly], yes we would like R to be considered as a GNU program.

I hereby dub R GNU software!

A Free Software Project

- Since we opened up the project, it has gone ahead in leaps and bounds.
- On February 29, 2000, the software was deemed fully featured enough and stable enough for the 1.0 release to take place.
- There are now nearly 20 core developers maintaining and extending the language interpreter and its basic functionality.
- The group includes a number of well-known researchers in Statistical Computing.
- The software now has a regular six-monthly release cycle and will shortly see the release of version 2.10.

The intense software development effort leading up to R version 1.

R Core Developers

Peter Dalgaard	University of Copenhagen
John Chambers	Bell Labs and Stanford University
Robert Gentleman	Fred Hutchinson Cancer Researh Center
Kurt Hornik	University of Vienna
Stefano Iacus	University of Milan
Ross Ihaka	University of Auckland
Friedrich Leisch	University of Munich
Thomas Lumley	University of Washington
Martin Mächler	ETH Zurich
Duncan Murdoch	University of Waterloo
Paul Murrell	University of Auckland
Martyn Plummer	International Agency for Research on Cancer
Brian Ripley	Oxford University
Duncan Temple Lang	University of California
Luke Tierney	University of Iowa
Simon Urbanek	AT&T

Current Status

- The *R Project* is an international collaboration of researchers in statistical computing.
- The formal structure for the project is provided by the *R Foundation*, a non-profit foundation based in Vienna.
- Development is carried out by the roughly 20 members of the "R Core Team."
- Releases of the R environment are made through the CRAN (comprehensive R archive network) twice per year.
- The software continues to be released under a "free software" license.

Current Status

- There are some 50 books which have been published (or are in preparation) dealing with R and its applications.
- Springer has a book series dedicated to R (currently there are 20 titles in the series).
- The "R Newsletter" is about to be relaunched as the "R Journal."
- There are over 1700 extension *packages* which have been contributed to CRAN.

Limitations

- R is a useful piece of software, but it does have limitations.
- Two major complaints are:
  - "It's too slow for my analysis."
  - "It can't handle my multigigabyte data set."
- Help is on the way for the first of these problems.
- The second issue is more fundamental.

Why Speed Can be Improved

- Multicore machines are becoming commonplace, soon they will be ubiquitous.
- Within a year or two this should provide an order of magnitude improvement for many statistical problems.
- The improvement is possible because many of R's computations are vectorised and it is possible to partition them and assign the subproblems to separate processors.

Why Size is a Problem

- R uses a “call by value” evaluation model.
- This means that data values are copied whenever they take part in computations.
- In the worst case, large data objects can be copied multiple times.
- In the case of fitting a linear model, the design matrix is copied 6 times during the fitting process.
- This problem is fundamental. Changing the evaluation model means rewriting the entire code base.

A New Language?

- Because of the performance and resource consumption problems with R, a new language is needed.
- Initial work indicates that it is possible to build a language which will perform two orders of magnitude faster than R for scalar computations and use significantly less memory than R for tasks such as model fitting.
- At the moment there are just three people working part-time on the project (Ihaka, Duncan Temple Lang and Brendan McArdle).
- Progress is slow because the research is unsupported.
- When the base language is in place, a collaborative model like that used with R can be used to add functionality.