USING THE R STATISTICAL COMPUTING ENVIRONMENT TO TEACH SOCIAL STATISTICS COURSES*

John Fox and Robert Andersen
Department of Sociology
McMaster University

January 2005

*** Acknowledgements:** Support for developing the `Rcmdr` package discussed in this paper was provided by the Arts Research Board of McMaster University. We wish to thank Tony Christensen, a research assistant who systematically examined several introductory-statistics texts to ensure that the `Rcmdr` package contains the procedures commonly used in introductory statistics courses, and helped to test the program.

USING THE R STATISTICAL COMPUTING ENVIRONMENT TO TEACH SOCIAL STATISTICS COURSES

ABSTRACT

R is a free, cooperatively developed, open-source implementation of the S statistical programming language and computing environment, a language that has become a de-facto standard among statisticians for the development of statistical software. The goal of this paper is to show the value of using R to teach social statistics courses. The paper briefly introduces R, and describes its use in four such courses: two introductory courses (one for undergraduates and the other for graduate students), in which students interact with R through a graphical-user interface; a graduate course on applied regression and generalized linear models; and a more advanced graduate course that takes up several topics, including structural-equation models, survival analysis, and mixed-effects models for hierarchical and longitudinal data.

keywords: social-statistics education, statistical computing, S language, open-source software

USING THE R STATISTICAL COMPUTING ENVIRONMENT TO TEACH SOCIAL STATISTICS COURSES

R (Ihaka and Gentleman 1996) is a free, cooperatively developed, open-source implementation of S, a powerful and flexible statistical programming language and computing environment that has become the effective standard among statisticians. Despite its strong points, and its large user base among statisticians, it is fair to say that R and its commercial cousin S-PLUS are not widely used by sociologists and other social scientists. Nevertheless, interest in R among social scientists is increasing. For example, since 2002 several relatively advanced courses at the Summer Program of the Inter-University Consortium for Political and Social Research (ICPSR) have coordinated their use of R, and the Program has introduced a lecture series on S that focuses on R. Similarly, the Oxford Spring School in Quantitative Methods for Social Research also emphasizes the use of R, along with Stata, in its workshops. [1]

The principal goal of this paper is to demonstrate the advantages of using R in social statistics courses, both at the undergraduate and graduate level. The paper is divided into three sections: (1) a general discussion of R, its benefits, and how it can be obtained; (2) a brief introduction to computing with R; and (3) a description of how R is currently being used in four social statistics courses in the Sociology Department at McMaster University in Hamilton, Ontario, Canada.

WHAT IS R AND WHY USE IT?

The S programming language was originally developed at Bell Labs by John Chambers and others (Becker, Chambers and Wilks 1988; Chambers 1998; Chambers and Hastie 1992). A commercial implementation of S, called S-PLUS, is sold by Insightful Corporation (see http://www.insightful.com/products/splus/) and has been available, in one or another form, for two decades. More recently, R, which has facilities generally comparable to those of S-PLUS, has become a popular implementation of S. The basic R system, and contributed packages, are available from the Comprehensive R Archive Network (CRAN) at http://cran.r-project.org/.

Aside from its price (i.e., zero), there are several other reasons to adopt R in preference to other commonly used statistics programs:

- R is high quality software. Several of the leading figures in statistical computing (e.g., Douglas Bates, John Chambers, Brian Ripley, and Luke Tierney) are closely associated with the development of R.
- The S statistical computing language is very well designed, both from the point of view of the practicing data analyst and from that of the computer scientist. Indeed, John Chambers received the 1998 Software System Award from the Association for Computing Machinery (ACM – the principal professional organization of computer scientists) for developing S. Chambers is now a member of the R core development team.
- The basic R system is supplemented by several "recommended packages" which are part of the standard R distribution. In addition to the recommended packages, at the time of writing more than 400 "contributed packages" are freely available from CRAN, many representing the state of the art in various areas of statistical computing. The basic R system augmented by the contributed packages is arguably the most extensive resource for statistical computing currently available.
- Reflecting its origin at Bell Labs, S (including R) incorporates excellent facilities for drawing statistical graphs, including "Trellis graphics" (Cleveland 1993).
- Because it incorporates a true programming language, R is relatively easily extended. This is an attractive characteristic for instructors, and for students in advanced classes.
- R runs on all of the commonly used computer platforms – including Windows systems, Unix/Linux systems, and Macintoshes – and installs in the normal manner on each platform. (S-PLUS, in contrast, has no Macintosh version.)

Unlike statistical packages such as SPSS and Minitab, which provide point-and-click graphical-user interfaces ("GUIs") to most of their statistical capabilities, R is command-oriented: Users normally type commands at a command-prompt (or in a programming editor, such as WinEdt or Emacs), and the R interpreter responds interactively to these commands. R's somewhat misleadingly named "Rgui" for Windows provides some basic functions, such as menus for downloading and loading packages (in the Packages menu), as alternatives to commands, but it does not incorporate a GUI for statistical procedures; Rgui does, however, provide a basic programming editor in the form of "script" windows.

A command-oriented interface to statistical software provides a number of important advantages, especially to advanced or regular users:

- It is easier in a command-driven system to correct, modify, and replicate analyses. It is true that many statistical programs (e.g., SPSS) have GUIs that are capable of generating commands, but in our experience users who rely on a GUI rarely take advantage of this feature, preferring instead to point and click their way through the GUI rather than editing commands.
- It is simpler, and natural, to create a permanent record of one's work, including both input and output. Indeed, R accommodates sophisticated facilities for "literate programming" (Knuth, 1992) in which commands and output can be mixed in a natural manner with explanatory text (Leisch, 2002).
- GUIs to complex statistical systems are either incomplete (necessitating the use of commands for more advanced or non-standard analyses) or Byzantine (requiring the user to drill down through a complex maze of menus, dialog boxes, and graphical controls) – or both.

Those who migrate to R from point-and-click statistical software may not be immediately comfortable with the command-oriented S programming language, but in many respects R is initially much simpler for someone entirely new to statistical computing. More importantly, R implicitly encourages a deeper understanding of the statistical procedures being used than do some other programs, such as SPSS and Minitab. Notwithstanding these points, R does include tools to construct graphical interfaces, most prominently via the `tcltk` package (one of the "recommended" R packages, which are a part of the standard R distribution; see Dalgaard 2001, 2002), which provides access to the Tcl/Tk GUI builder (Welch 2000). Tcl/Tk systems are freely available for all major computing platforms, and the standard distribution of R for Windows includes a basic Tcl/Tk system that is installed along with R. A GUI interface built using these tools is provided by the `Rcmdr` ("R Commander") package, which is described later in this paper.

A QUICK TOUR OF R

Although it is not feasible to provide a general introduction to R within the confines of this paper, we nevertheless attempt to convey enough background information to render the remainder of the paper intelligible. S is a functional programming language: programs – both those supplied with R and those written by the user – take the form of functions, and S expressions entail function calls. Even operators, such as the + operator for addition and * operator for multiplication, implicitly invoke functions. When the R program is started, and after it prints an introductory message, the R interpreter prompts for input with > (the greater-than sign). The interpreter executes expressions that are typed at the command prompt. For example:

```
> 1 + 2*3
[1] 7
> 1:3
[1] 1 2 3
> 1:3 + c(2, 5, -1)
[1] 3 7 2
> x <- 1:3
> 5*x
[1]  5 10 15
```

Most of the preceding interaction with the R interpreter is likely self-explanatory, with the following exceptions:

- Simple ("vector") output is prefixed by `[1]`. Were output to extend over several lines, the index number of the first element in each line would appear in square brackets at the beginning of the line.
- `c()` is the "combine" function, which creates a vector from its arguments. In general, the arguments to functions in the S language are specified within parentheses and separated by commas; where arguments to a function are named, they are specified as *`name = value`*.
- `<-` is the assignment operator; in the illustration, the vector of integers from 1 to 3 is assigned to a variable named `x`. The equals sign (`=`) may also be used for assignment. Variables are created and memory is allocated to them dynamically. Variable names can consist of any combination of lower- and upper-case letters, numerals, periods, and (as of version 1.9.0 of R[2]) underscores, but cannot begin with a numeral or an underscore; R is case-sensitive, so the names `X` and `x` are distinguished, and there is no limit on the number of characters in a name.
- All of the objects in these examples are numeric vectors, but R supports other kinds of data as well and many other data structures, including those generated by two object-oriented programming systems.

Data Input

Variables composing small data sets can be entered directly at the keyboard, as above, or in a spreadsheet-like data editor, but this approach is clearly limited. R has many other – and more generally convenient – facilities for data input: Data can be read from text files, binary files, the clipboard, the Internet, and attached packages; imported from other statistical packages and spreadsheet programs; and input through connections to scientific devices. Moreover, a transparent link can be established between R and a database-management system, permitting the manipulation and analysis of very large data sets (R normally keeps active data in memory, a limitation that is not very restrictive in an era of multi-Gigabyte computers).

One of the simplest ways to get data into R is from a package that is attached via the `library` function; packages of data can be created for course use, for example. To illustrate, the `car` package (associated with Fox 2002) includes a "data frame" (rectangular data set) named `Prestige`, which contains data on the prestige and some other characteristics of 102 Canadian occupations circa 1970. The variables in the data set, which are further discussed in Fox (1997, 2002), are as follows: `prestige` is the average prestige rating of the occupation, on a 0 to 100 scale; `income` is the average income of occupational incumbents, in dollars; `education` is the average years of schooling of occupational incumbents; and `type` is a "factor" (categorical variable) with three "levels" (categories) – `bc` (blue collar), `wc` (white collar), and `prof` (professional and managerial). Three occupations have missing `type`. The commands below load the `car` package, read the data set into memory, and display the observations (the ellipses, `. . .`, represent elided lines, and `NA` is a missing-data indicator):

```
> library(car)
> data(Prestige)
> Prestige
                          education income women prestige census type
GOV.ADMINISTRATORS            13.11  12351 11.16     68.8   1113 prof
GENERAL.MANAGERS              12.26  25879  4.02     69.1   1130 prof
ACCOUNTANTS                   12.77   9271 15.70     63.4   1171 prof
. . .
NURSING.AIDES                  9.45   3485 76.14     34.9   3135   bc
PHYSIO.THERAPSTS              13.62   5092 82.66     72.1   3137 prof
PHARMACISTS                   15.21  10432 24.71     69.3   3151 prof
MEDICAL.TECHNICIANS           12.79   5180 76.04     67.5   3156   wc
COMMERCIAL.ARTISTS            11.09   6197 21.03     57.2   3314 prof
RADIO.TV.ANNOUNCERS           12.71   7562 11.15     57.6   3337   wc
ATHLETES                      11.44   8206  8.13     54.1   3373 <NA>
. . .
BOOKBINDERS                    8.55   3617 70.87     35.2   9517   bc

> dim(Prestige)
[1] 102   6
```

The R “search path” determines where the interpreter will look for objects; the path initially includes the “global environment” – which contains objects created during the R session – and several packages that are attached by default. As with most aspects of R, the list of initially attached packages is configurable, and can easily be adapted to the needs of a specific course. A convenient way to compute with the variables in a data set is to attach the data frame to the search path. Variables in the data frame will then be accessible much as if they had been entered at the keyboard:

```
> attach(Prestige)
> mean(income)
[1] 6797.902
> median(income)
[1] 5930.5
> summary(type)
  bc prof   wc NA's
  44   31   23    4
> summary(education)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  6.380   8.445  10.540  10.740  12.650  15.970
```

`summary` is an example of a “generic function,” a functions that adapts its behavior to its input. Thus, `summary` produces counts for the factor `type`, but a five-number summary plus the mean for the numeric variable `education`.

It is also very easy to read data from a text file. Suppose, for example, that the Canadian occupational-prestige data reside in the file `c:\data\prestige.txt` with the following contents:

```
education income women prestige census type
GOV.ADMINISTRATORS           13.11 12351 11.16 68.8  1113  prof
GENERAL.MANAGERS             12.26 25879  4.02 69.1  1130  prof
ACCOUNTANTS                  12.77  9271 15.70 63.4  1171  prof
. . .
```

To read the data into a data frame named `Canadian.Prestige`:

```
> Canadian.Prestige <- read.table("c:/data/Prestige.txt")
```

The `read.table` function detects that the first line in the file contains variable names and that the first entry in each row is the row name. Data entries in each row are separated by “white space” (spaces or tabs); this is a convenient form in which to prepare data for `read.table`, but the function is flexible enough to read many different input-data formats. Note that forward slashes (`/`) are used to separate directories, even in Windows systems.

Statistical Models

The statistical modeling facilities in S are one of the great strengths of the language: S provides a version of the Wilkinson and Rogers (1973) notation for describing linear and related models (another version of which is used in SAS), and S functions for statistical modeling return objects that can then be subjected to further computations – to extract residuals, plot various quantities, print an analysis-of-variance table, print a model summary, and so on. This approach contrasts with that in a statistical package, such as SPSS or SAS, which is primarily oriented towards turning data sets into printed and plotted output. Moreover, the handling of factors and interactions is especially convenient in S. Unlike many commonly used programs such as SPSS and Stata, when the data are input, R by default treats character data as factors, thus avoiding the hassle of having to specify labels—i.e., category names can be used directly rather than assigning numbers which then need to be given value labels. There is also no need to construct product variables for interaction terms directly because they are specified simply and more naturally as part of a model formula. A brief and partial description of model formulas is shown in Table 1.

[Table 1 about here]

To illustrate, the following command invokes the `lm` function to create a linear-model object, regressing the variable `prestige` on `income`, `education`, and `type` of occupation, and permitting `income` and `education` to interact with `type`:

```
> mod <- lm(prestige ~ (income + education)*type)
```

Nothing is printed by this command, which returns a linear-model object that is saved in the variable `mod`. The generic `summary` function may be used to obtain a standard regression printout (the `e`'s in the output below represent scientific notation; e.g., `9.62e-09` = 9.62×10^{-9}):

```
> summary(mod)

Call:
lm(formula = prestige ~ (income + education) * type)

Residuals:
    Min      1Q  Median      3Q     Max 
-13.462  -4.225   1.346   3.826  19.631 

Coefficients:
                      Estimate  Std. Error t value Pr(>|t|)    
(Intercept)          2.2757530   7.0565809   0.323   0.7478    
income               0.0035224   0.0005563   6.332 9.62e-09 ***
education            1.7132747   0.9572405   1.790   0.0769 .  
typeprof            15.3518963  13.7152577   1.119   0.2660    
typewc             -33.5366519  17.6536726  -1.900   0.0607 .  
income:typeprof     -0.0029026   0.0005989  -4.847 5.28e-06 ***
income:typewc       -0.0020719   0.0008940  -2.318   0.0228 *  
education:typeprof   1.3878090   1.2886282   1.077   0.2844    
education:typewc     4.2908748   1.7572512   2.442   0.0166 *  
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1 

Residual standard error: 6.318 on 89 degrees of freedom
Multiple R-Squared: 0.8747,     Adjusted R-squared: 0.8634 
F-statistic: 77.64 on 8 and 89 DF,  p-value: < 2.2e-16
```

Because `type` is a factor, it automatically gives rise to a set of dummy-coded (0/1) contrasts. Other contrast-coding schemes, including customized contrasts, may be employed as well. To obtain an analysis of variance table for the regression,

```
> Anova(mod)
Anova Table (Type II tests)

Response: prestige
               Sum Sq Df F value    Pr(>F)    
income         1131.9  1 28.3544 7.511e-07 ***
education      1068.0  1 26.7532 1.413e-06 ***
type            591.2  2  7.4044  0.001060 ** 
income:type     951.8  2 11.9210 2.588e-05 ***
education:type  238.4  2  2.9859  0.055574 .  
Residuals      3552.9 89                      
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1
```

Statistical Graphics

There are exceptional facilities for statistical graphics in R, both in the basic R system and in add-on packages. The generic `plot` function, for example, is extremely flexible and powerful. Graphs of varying complexity can be built in a sequence of typically simple commands that give the user complete control over the appearance of the graph. For example, in sequential order, the commands below draw a scatterplot, add a least-squares line, and place a title above the graph (producing Figure 1). The last line allows us to identify points on the graph interactively using the mouse.

```
> plot(income, prestige)
> abline(lm(prestige ~ income))
> title("Scatterplot of Prestige vs. Income")
> identify(income, prestige, row.names(Prestige))
```

Even this simple graph is of near publication quality, but customization is also straightforward. Point symbols, sizes and colors, line types, axes labels, and many other graphical elements can be changed with simple commands. Even type-set-quality mathematical annotation within in a graph is easily produced (Murrell and Ihaka, 2000). Moreover, more advanced two-dimensional plots, such as Trellis graphics (see Cleveland 1993), can be constructed using the `lattice` package, which is part of the standard R distribution. Sophisticated dynamic 3D graphs can be constructed using the `rgl` package.

[Figure 1 about here]

Yet another feature of R that sets it apart from programs commonly used by sociologists is its facilities for model diagnostics. Simple functions extract or compute various quantities from model objects that can then be used to perform diagnostics and construct graphs — e.g., `residuals(mod)` returns residuals, while `fitted(mod)` returns the fitted values. Simple diagnostic plots can also be directly constructed using the generic `plot` function on a model object. Following from the example above, the command `plot(mod)` produces a four-panel graph of diagnostic plots (shown in Figure 2): (1) a plot of residuals versus fitted values; (2) a normal quantile-comparison plot of the residuals; (3) a scale-location plot of the square root of the absolute residuals against fitted values; and (4) a plot of Cook's distances versus observation indices.

A more comprehensive set of functions for diagnostic plots—including added-variable plots, component-plus-residual plots, leverage plots, and spread-level plots—is available in the `car` package. The `car` package also includes functions for testing for autocorrelated errors, for non-constant error variance, and for outliers, and functions for Box-Cox and Box-Tidwell transformations, respectively, of the response and explanatory variables in a regression. Other common diagnostics are available in the `lmtest` package.

[Figure 2 about here]

Contributed Packages

As mentioned earlier, the flexibility of R has led to the development of a wide range of add-on packages. In combination, these packages extend R to include functions for all statistical models commonly used by social scientists. Here is a partial list of these methods (with associated packages in parentheses): generalized linear models (`base`), multinomial logit models (`nnet`), social network analysis (`sna`), structural equation modeling with latent variables (`sem`), latent class analysis (`e1071`), smoothing splines and generalized additive models (`mgcv`), mixed-effects models (`nlme`), event-history analysis (`survival`), bootstrapping (`boot`), cluster analysis (`mclust`). (This association of methods with packages understates the richness of R: In most of these instances, relevant software is provided in other packages as well. For example, the `gam` package also provides facilities for generalized additive models.) Some of these packages will be described in more detail when we discuss the use of R in graduate statistics courses later in the paper.

If the user's computer is connected to the Internet, contributed packages can be installed or updated directly from CRAN within R itself, either by using the *Packages* menu in the *R Console* or from the command prompt; for example, to download and install the `car` package:

```
> install.packages("car")
```

Packages can also be installed from zip files located on a local disk.

Getting Help

Despite being free, R is extensively documented. For example, the standard R distribution comes with a complete set of manuals, including a decent introductory manual. Moreover, contributed R packages incorporate a standard method of providing documentation that generates help files appropriate to the user's computing platform. (In fact, contributed packages are not posted on CRAN unless they have the required documentation.) Standard

Windows help is available via a *Help* menu, but help is can also be obtained in a number of ways from the command prompt:

- `help(lm)` or `?lm`: Opens the help file for an object – in this case, for the `lm` function.
- `help.search("logistic")`: Searches all installed help files for a particular sequence of characters – here, for the word "logistic."
- `apropos("norm")`: Searches for a character sequence ("norm") in the names of all objects available in the current R session.

Help files for individual packages can also be printed as manuals.

In addition to the free documentation included with R itself, S and R are the subject of several texts (see, in particular, Fox 2002; Venables and Ripley 2000, 2002), and a number of other books are associated with software that is available in R packages (e.g., Davison and Hinkley 1997; Efron and Tibshirani 1993; Pinheiro and Bates 2000; Therneau and Grambsch 2000). Finally, R users have access to excellent support in the r-help email list (see, http://www.r-project.org/mail). Before posting a question to the r-help list, however, please read the posting guide at http://www.r-project.org/posting-guide.html.

SOCIAL STATISTICS COURSES AT MCMASTER UNIVERSITY

The remainder of this paper describes the use of R in four social-statistics courses taught at McMaster University in Hamilton, Ontario, Canada:

- Two of these courses (Sociology 3H06 and 6Z03) are introductory social-statistics courses, for undergraduates and graduate students, respectively. The courses have nearly the same content. The undergraduate course is required of all Sociology honours majors (which, in Canada, is the four-year bachelor's degree programme). The graduate course is essentially remedial.
- The third course (Sociology 740) covers linear and generalized linear models, and is required of all of our PhD students.
- The fourth course (Sociology 761) is a more advanced social-statistics course, and is optional.

Introductory Social Statistics Courses

Sociology 3H06 is a two-semester undergraduate introductory social-statistics. Sociology 6Z03 is a one-semester version of the same course for graduate students who have a weak background in statistics. The current text for these courses is Moore's *The Basic Practice of Statistics, Third Edition* (Moore 2003). The course covers all of Moore's text. In addition, we supplement Moore's treatment of regression (with lecture material on multiple regression) and contingency tables. Detailed course outlines are available at
http://socserv.socsci.mcmaster.ca/jfox/Courses/soc3h6/index.html,
http://socserv.socsci.mcmaster.ca/andersen/soc3H6/index.htm, and
http://socserv.socsci.mcmaster.ca/andersen/soc6Z3/index.htm.

Students in these classes are provided with a Windows CD/ROM with the software and data sets for the class. The software can be run directly from the CD or installed on the student's computer. R software and course data sets can also be accessed in university computer labs.

Despite the benefits of R's command language for the advanced user, a command-driven system has some disadvantages in an introductory statistics course:

- The learning curve is relatively "steep." In contrast, a well-designed GUI is essentially self-teaching.
- New and casual users frequently must look up commands. In contrast, a GUI presents users with options and consequently does not tax memory.
- It is easy to make mistakes – such as typing errors, syntax errors, and the specification of nonsensical analyses – in a command-driven system. In contrast, a GUI eliminates most syntax and typing errors, and should, at least to an extent, protect users from nonsense.

The `Rcmdr` ("R Commander") package (described in more detail in Fox, 2004) provides a basic-statistics GUI for R, including methods of analysis typically covered in an introductory statistics class together with linear and generalized linear models. The `Rcmdr` loads in the same manner as any R package, and is useable on all of the platforms on which R runs. Moreover, commands can still be entered directly at the command prompt or via the log/script window in the R Commander. The R Commander consequently makes it easy to carry out statistical analyses while at the same time facilitating the learning of R functions.

Figure 3 shows part of a Windows XP desktop with the *R Commander* window. Commands are generated via menus and dialog boxes accessed from the *R Commander* window. These commands are logged to the script window in the R Commander. Commands are also echoed to the R Commander output window along with printed output that is generated. Graphs appear in a separate graphics-device window, as illustrated in Figure 3. The script window incorporates a simple text editor, which allows the user to re-execute (e.g., via the *Submit* button), modify, enter, and save commands. Other elements of the interface include buttons to view and edit the "active" data set in a spreadsheet-like data editor; to change the active data set; and to change the active statistical model. Figure 4 shows a typical R Commander dialog box, to create the parallel boxplots shown in Figure 3.

[Figures 3 and 4 about here]

The R Commander was designed to cover the contents of Moore's text, and of several other introductory statistics texts that were consulted. The complete menu tree for the R Commander is shown in Figure 5. Use of R through the R Commander is not essentially different from the use of any menu-driven statistical package. Students in the graduate version of the course (Sociology 6Z3) are also encouraged to learn some R commands, a task that is facilitated by the commands that appear in the R Commander log window.

[Figure 5 about here]

To illustrate how the R Commander provides a graphical interface to R functions for linear and generalized linear models, we recall our example of `prestige` regressed on `income`, `education`, and `type` of occupation, including interactions of `income` and `education` with `type.` The same model can be specified using the R Commander GUI in the linear-model dialog box shown in Figure 6. In this case, the summary of the model will be printed after the model is fit, and the resulting linear-model object becomes the "active model." Items in the *Model* menu (see Figure 5) apply to the active model. For example, the basic diagnostic plots produced by plotting a linear-model object, as shown previously in Figure 2, can also be constructed via the R Commander menus: *Models* → *Graphs* → *Basic diagnostic plots* (where the arrows denote menu selection).

[Figure 6 about here]

An Applied Regression Course

Sociology 740 is a one-semester course in applied regression analysis required of all of our doctoral students who have not already taken an equivalent course. The course is oriented towards the use of statistical models for data analysis, and takes up exploratory data analysis and variable transformations, linear statistical models, and generalized linear models (e.g., for categorical responses). There are two texts for the class: *Applied Regression* (Fox 1997) and a book on R and S-PLUS (Fox 2002). A detailed course outline is available at http://socserv.socsci.mcmaster.ca/jfox/Courses/soc740/index.html.

The `Rcmdr` package covers almost all of the computing for the course, including several topics on linear-model and generalized-linear-model diagnostics. Students in the class – particularly those who intend to use quantitative methods in their work – are strongly encouraged, however, to learn to write R commands and to understand the rudiments of programming in R. As in Sociology 3H06 and 6Z03 (and in Sociology 761, described below), students in Sociology 740 are provided with a Windows CD/ROM with the software and data for the class.

A More Advanced Course in Social Statistics

Sociology 761 is a one-semester course in social statistics, covering several selected topics. Students in the class also write a paper on a statistical topic of their choice. In the Fall semester of 2004, the class began with introductions/refreshers on R, basic matrix algebra, and basic calculus, and proceeded to take up structural-equation models (including models with latent variables), survival ("event-history") analysis, and linear mixed models for hierarchical and longitudinal data. In the last two years, students in the class wrote papers on imputation of missing data, nonlinear regression, nonparametric regression, exploratory factor analysis, interactions in latent-variable structural-equation models, and time-series regression. Again, a detailed course outline is available at http://socserv.socsci.mcmaster.ca/jfox/Courses/soc761/index.html.

With the exception of structural-equation modeling, R has state-of-the-art capabilities in all of these areas:

- R includes excellent facilities for numerical linear algebra. In this class, students just scratch the surface, using R for basic matrix computations – products, inverses, determinants, etc.
- One of us (Fox) is the author of the `sem` (structural-equation modeling) package in R, which is described in an on-line appendix to Fox (2002a), available at http://socserv.socsci.mcmaster.ca/jfox/Books/Companion/appendix.html. (Several other appendices, on topics such as nonlinear regression, nonparametric regression, bootstrapping, etc., are also available at this location.) The `sem` package is capable of fitting observed-variable structural-equation models (SEMs) by two-stage least squares, and both observed-variable and latent-variable models by full-information maximum likelihood, assuming multinormality. Although adequate for a basic introduction to modern structural-equation modeling, the package is substantially more limited than special-purpose SEM software, such as LISREL, EQS, or Amos.
- The `survival` package, originally written by Terry Therneau for S-PLUS and ported to R (see Therneau and Grambsch 2000), is arguably the most capable facility for survival analysis in any statistical software.
- The `nlme` (nonlinear mixed-effects) package, by Pinheiro and Bates (2000), is also cutting-edge software that fits linear and nonlinear mixed-effects models. Software for generalized linear mixed models is also available in R – e.g., using the `glmmPQL` function in the `MASS` package (which is associated with Venables and Ripley, 2002), or the `GLMM` function in Bates's `lme4` package.
- There are equally impressive facilities in R for missing-data imputation (e.g., the software associated with Shafer, 1997), for nonlinear regression (Bates's `nlm` package), for nonparametric regression (e.g., the `mgcv` and `gam` packages for fitting generalized additive models, Wood, 2000, 2003; Hastie and Tibshirani, 1990), and for time-series regression (the `gls` function in the `nlme` package).

Students in Sociology 761 learn to interact with R through the command-line interface, and acquire basic programming skills – for example to manage data. We take advantage of the extensibility of R by providing students with simple programs (available from the web site for the course) for computing life tables and for preparing survival data for analysis.

An illustration: Cox regression for survival data. To convey a sense of what it is like to use R for more advanced statistical modeling, we provide an example of survival analysis from Sociology 761. The file `Rossi.txt` contains data from an experimental study of recidivism of 432 male prisoners, who were observed for a year after being released from prison (Rossi, Berk, and Lenihan 1980). The following variables are included in the data; the variable names are those used by Allison (1995), from whom this example and variable descriptions are adapted:

- `week`: week of first arrest after release, or censoring time.
- `arrest`: the event indicator, equal to `1` for those arrested during the period of the study and `0` for those who were not arrested.
- `fin`: a dummy variable, equal to `1` if the individual received financial aid after release from prison, and `0` if he did not; financial aid was a randomly assigned factor manipulated by the researchers.
- `age`: in years at the time of release.
- `race`: a dummy variable coded `1` for blacks and `0` for others.
- `wexp`: a dummy variable coded `1` if the individual had full-time work experience prior to incarceration and `0` if he did not.
- `mar`: a dummy variable coded `1` if the individual was married at the time of release and `0` if he was not.
- `paro`: a dummy variable coded `1` if the individual was released on parole and `0` if he was not.
- `prio`: number of prior convictions.
- `educ`: education, a categorical variable, with codes `2` (grade 6 or less), `3` (grades 6 through 9), `4` (grades 10 and 11), `5` (grade 12), or `6` (some post-secondary).
- `emp1` – `emp52`: dummy variables coded `1` if the individual was employed in the corresponding week of the study and `0` otherwise. Employment is therefore a time-varying covariate, and is ignored in the example developed below; in the course, the example is subsequently extended to consider the impact of employment.

We read the data file into a data frame directly from the course web site, and print the first few observations (omitting the variables `emp1` – `emp52`, which are in columns 11–62 of the data frame):

```
> Rossi <-read.table(
+    "http://socserv.socsci.mcmaster.ca/jfox/Courses/soc761/Rossi.txt",
+    header=TRUE)

> Rossi[1:5, 1:10]      # print the first 5 observations
  week arrest fin age race wexp mar paro prio educ
1   20      1   0  27    1    0   0    1    3    3
2   17      1   0  18    1    0   0    1    8    4
3   25      1   0  19    0    1   0    1   13    3
4   52      0   1  23    1    1   1    1    1    5
5   52      0   0  19    0    1   0    1    3    3
```

Thus, for example, the first individual was arrested in week 20 of the study, while the fourth individual was never rearrested, and hence has a censoring time of 52. (The argument `header=TRUE` is necessary here because although variable names are included in the first row of the data file, there are no row names. Since the first and subsequent lines therefore have the same number of fields, `read.table` cannot deduce that the first line contains variable names.).

Following Allison, a Cox regression of time to rearrest on the time-constant covariates is specified as follows:

```
> mod.allison <- coxph(Surv(week, arrest) ~
+     fin + age + race + wexp + mar + paro + prio,
+     data=Rossi)
> summary(mod.allison)
Call:
coxph(formula = Surv(week, arrest) fin + age + race + wexp +
    mar + paro + prio, data = Rossi)

  n= 432

        coef exp(coef) se(coef)      z      p
fin  -0.3794     0.684   0.1914 -1.983 0.0470
age  -0.0574     0.944   0.0220 -2.611 0.0090
race  0.3139     1.369   0.3080  1.019 0.3100
wexp -0.1498     0.861   0.2122 -0.706 0.4800
mar  -0.4337     0.648   0.3819 -1.136 0.2600
paro -0.0849     0.919   0.1958 -0.434 0.6600
prio  0.0915     1.096   0.0286  3.195 0.0014

     exp(coef) exp(-coef) lower .95 upper .95
fin      0.684      1.461     0.470     0.996
age      0.944      1.059     0.904     0.986
race     1.369      0.731     0.748     2.503
wexp     0.861      1.162     0.568     1.305
```

```
mar      0.648      1.543     0.307     1.370
paro     0.919      1.089     0.626     1.348
prio     1.096      0.913     1.036     1.159

Rsquare= 0.074   (max possible= 0.956 )
Likelihood ratio test= 33.3  on 7 df,   p=2.36e-05
Wald test            = 32.1  on 7 df,   p=3.86e-05
Score (logrank) test = 33.5  on 7 df,   p=2.11e-05
```

- Notice here that as an alternative to attaching the `Rossi` data frame to the search path, we have specified a `data` argument to the `coxph` function.
- The column marked `z` in the output records the ratio of each regression coefficient to its standard error, a Wald statistic which is asymptotically standard normal under the hypothesis that the corresponding β is zero. The covariates `age` and `prio` (prior convictions) have highly statistically significant coefficients, while the coefficient for `fin` (financial aid – the focus of the study) is marginally significant.
- The exponentiated coefficients in the second column of the first panel (and in the first column of the second panel) of the output are interpretable as multiplicative effects on the hazard. Thus, for example, holding the other covariates constant, an additional year of age on average reduces the weekly hazard of rearrest by a factor of $e^{b_{age}} = 0.944$ – that is, by 5.6 percent. Similarly, each prior conviction increases the hazard by a factor of 1.096, or 9.6 percent.
- The likelihood-ratio, Wald, and score chi-square statistics at the bottom of the output are asymptotically equivalent tests of the omnibus null hypothesis that all of the β's are zero. In this instance, the test statistics are in close agreement, and the hypothesis is soundly rejected.

Having fit a Cox model to the data, we now examine the estimated distribution of survival times. The R function `survfit` estimates the survival function $S(t)$, by default at the mean values of the covariates. The `plot` method for objects returned by `survfit` graphs the estimated surivival function, along with a point-wise 95-percent confidence band. For example, for the model just fit to the recidivism data:

```
> plot(survfit(mod.allison), ylim=c(.7, 1), xlab='Weeks',
+     ylab='Proportion Not Rearrested')
```

This command produces Figure 7. [The limits for the vertical axis, set by `ylim=c(.7, 1)`, were selected after examining an initial plot.]

[Figure 7 about here]

Even more cogently, we may wish to display how estimated survival depends upon the value of a covariate. Because the principal purpose of the recidivism study was to assess the impact of financial aid on rearrest, we focus on this covariate. We construct a new data frame with two rows, one for each value of `fin`; the other covariates are fixed to their average values. (In the case of a dummy covariate, such as `race`, the average value is the proportion coded 1 in the data set – for `race`, the proportion of blacks). This data frame is passed to `survfit` via the `newdata` argument:

```
> attach(Rossi)
> Rossi.fin <- data.frame(fin=c(0,1), age=rep(mean(age),2),
+     race=rep(mean(race),2), wexp=rep(mean(wexp),2),
+     mar=rep(mean(mar),2), paro=rep(mean(paro),2),
+     prio=rep(mean(prio),2))
> detach()
> plot(survfit(mod.allison, newdata=Rossi.fin),
+     conf.int=TRUE, lty=c(1,2), ylim=c(.6, 1))
> legend(locator(1), legend=c('fin = 0', 'fin = 1'),
+     lty=c(1,2))
```

We specified two additional arguments to `plot:` `lty=c(1,2)` indicates that the survival function for the first group (i.e., for `fin = 0`) will be plotted with a solid line, while that for the second group (`fin = 1`) will be plotted with a broken line (line types can be specified by number – as here – or by name); `conf.int=TRUE` requests that confidence envelopes be drawn around each estimated survival function (which is not the default when more than one survival function is plotted). Notice, as well, the use of the `legend` function (along with `locator`) to place a legend on the plot: We clicked the left mouse button to position the legend. The resulting graph, which appears in Figure 8, shows the higher estimated "survival" of those receiving financial aid, but the two confidence envelopes overlap substantially, even after 52 weeks.

[Figure 8 about here]

CONCLUSIONS

R is excellent software that happens to be free. The wide array of statistical methods available in R makes it an attractive platform for teaching a variety of courses, well beyond those covered in this paper. All techniques commonly used by social scientists are present in R, and due to its flexibility, methods that are not currently available can be programmed – typically, more easily in S than in other programming languages. Indeed, one of the principal attractions of R is the rapidity with which new methods are incorporated. Given that S is the standard for statistical programming among statisticians, there is every reason to believe that R will remain at the cutting edge of statistical computing for quite some time. Learning to use R, therefore, is time well spent by the serious quantitative social scientist.

The object-oriented style of S programming may be unfamiliar to users migrating from programs such as SPSS, but the strengths of this approach quickly become apparent. Standards for expressing statistical models, for handling data, and for providing documentation generally make it straightforward to use R and its contributed packages. In fact, those new to statistical computing should not find R any more difficult than any other command-driven program. More importantly, motivated students are likely to get a better understanding of statistical procedures by using R than they will by using GUI-oriented statistical software. It has been our experience that our most capable students very much enjoy using R, largely because they gain a sense of accomplishment from learning the rudiments of R programming. In any event, even students with who are less adept at computing and statistics should have no more trouble using the graphical user interface of the `Rcmdr` package than the point-and-click interfaces to programs such as SPSS or Minitab. Consequently, the standard command-line interface to R need not be an impediment to its use even in basic social-statistics courses.

In conclusion, it is difficult to over-emphasize the advantage of adopting high-quality free software for instructional use. Students need to “get their hands dirty with data” to understand and to become competent users of statistical methods. The increasing financial pressures on both public and private universities make R very attractive. R can be distributed to students for the cost of a CD, without having to pay for a site license. Students can work on their own computers, spending as much time as is necessary to learn what is required. They can, moreover, continue to use and update the software after they leave university. In short, R is an outstanding teaching resource, as well as a formidable tool for data analysis, presentation, and the development of new statistical methods.

REFERENCES

Allison, Paul D. 1995. *Survival Analysis Using the SAS System: A Practical Guide*. Cary NC: SAS Institute.

Andersen, Bob, John Fox, Charles Franklin and Jeff Gill. 2003. “The times they R achanging” *The Political Methodologist*, 11(2):22-24.

Becker, Richard A., John M. Chambers, and Allan R. Wilks. 1988. *The New S Language: A Programming Environment for Data Analysis and Graphics*. Pacific Grove CA: Wadsworth.

Chambers, John M. 1998. *Programming with Data: A Guide to the S Language*. New York: Springer.

Chambers, John M. and Trevor J. Hastie. 1992. *Statistical Models in S*. Pacific Grove CA: Wadsworth.

Cleveland, William S. 1993. *Visualizing Data*. Summit NJ: Hobart Press.

Dalgaard, Peter. 2001. “A primer on the R-Tcl/Tk package.” *R News* 1(3):27-31.

______. 2002. “Changes to the R-Tcl/Tk package.” *R News,* 2(3):25-27.

Davison, A.C. and D.V. Hinkley. 1997. *Bootstrap Methods and their Application*. Cambridge: Cambridge University Press.

Efron, Bradley and Robert J. Tibshirani. 1993. *An Introduction to the Bootstrap*. New York: Chapman and Hall.

Fox, John. 1997. *Applied Regression Analysis, Linear Models, and Related Methods*. Thousand Oaks CA: Sage.

______. 2002a. *An R and S-PLUS Companion to Applied Regression*. Thousand Oaks CA: Sage.

______. 2002b. “R at the ICPSR.” *R News*, 2(3): 39-40.

______. Unpublished. “Getting started with the R Commander: A basic-statistics graphical user interface to R.” Paper to be read at the useR Conference, Vienna, May 2004.

Hastie, T. J. and R. J. Tibshirani. 1990. *Generalized Additive Models.* London: Chapman and Hall.

Ihaka, Ross and Robert Gentleman. 1996. “R: A language for data analysis and graphics.” *Journal of Computational and Graphical Statistics* 5:299-314.

Knuth, Donald E. 1992. Literate Programming. Stanford CA: Center for the Study of Language and Information.

Leisch, Friedrich. 2002. “Sweave, part I: Mixing R and LaTex.” *R News,* 2(3): 28-31.

Moore, David S. 2003. *The Basic Practice of Statistics, Third Edition*. New York: Freeman.

Murrell, Paul and Ross Ihaka. 2000. “An approach to providing mathematical annotation in plots.” *Journal of Computational and Graphical Statistics*, 9:582–599.

Pinheiro, Jose C. and Douglas M. Bates. 2000. *Mixed-Effects Models in S and S-PLUS*. New York: Springer.

Rossi, Peter H., Richard A. Berk, and Kenneth J. Lenihan. 1980. *Money, Work and Crime: Some Experimental Results*. New York: Academic Press.

Schafer, Joe L. 1997. *Analysis of Incomplete Multivariate Data*. London: Chapman and Hall.

Therneau, Terry M. and Patricia M. Grambsch. 2000. *Modeling Survival Data: Extending the Cox Model*. New

York: Springer.

Venables, W.N. and B.D. Ripley. 2000. *S Programming*. New York: Springer.

______. 2000. *Modern Applied Statistics with S, Fourth Edition*. New York: Springer.

Welch, Brent B. 2000. *Practical Programming in Tcl and Tk, Third Edition*. Upper Saddle River NJ: Prentice-Hall.

Wilkinson, G.N. and C.E. Rogers. 1973. "Symbolic description of factorial models for analysis of variance." *Applied Statistics* 22:392-399.

Wood, Simon N. 2000. "Modelling and smoothing parameter estimation with multiple quadratic penalties." *Journal of the Royal Statistical Society, Series B* 62:413-428.

______. 2003. "Thin plate regression splines." *Journal of the Royal Statistical Society, Series B* 65:95-114.

Table 1. Statistical Modeling Functions and Model Formulas in R

A. Some Statistical Modeling Functions

Model Type	*Function*	*Package*[a]
Linear models	`lm`	`stats`
Generalized linear models	`glm`	`stats`
Generalized additive models	`gam`	`mgcv, gam`
Linear mixed-effect models	`lme`	`nlme`

B. Model Formulas

R Specification	*Model Description*
`lm(Y ~ X1 + X2 + X3)`	Main effects only (no interactions) for a linear model of `Y` regressed (~) on `X1`, `X2` and `X3.`
`lm(Y ~ X1 + X2 + X3 + X2:X3)` *or* `lm(Y ~ X1 + X2*X3)`	All main effects and the interaction between `X2` and `X3` only.
`lm(Y ~ X1*X2*X3)`	All main effects and all possible interactions (i.e., both the two-way and three-way interactions).
`lm(Y ~ X1 + X2*(X3 + X4))`	All main effects and the two-way interactions for `X2` with each of `X3` and `X4`.
`lm(Y ~ X1 + poly(X2, 3) + ns(X4, df=4))`	Regression of `Y` on a linear term in `X1`, a third-degree (i.e., cubic) polynomial in `X2`, and a natural cubic regression spline in `X4` with four degrees of freedom.
`lm(Y ~ X1 + X2 - 1)`	Main effects only, excluding the constant (i.e., regression through the origin).
`glm(Y ~ X1 + X2, family=binomial)`	Logit model of Y regressed on X1 and X2.
`gam(Y ~ s(X1, X2) + s(X3) + X4)`	Additive regression of `Y` on a two-dimensional smoothing spline in `X1` and `X2`, a smoothing spline in `X3`, and a linear term in `X4`.

[a] The `stats` package is part of the basic R system; `mgcv` and `nlme` are "recommend" packages, which are part of the standard R distribution.

Figure 1. A basic scatterplot in R

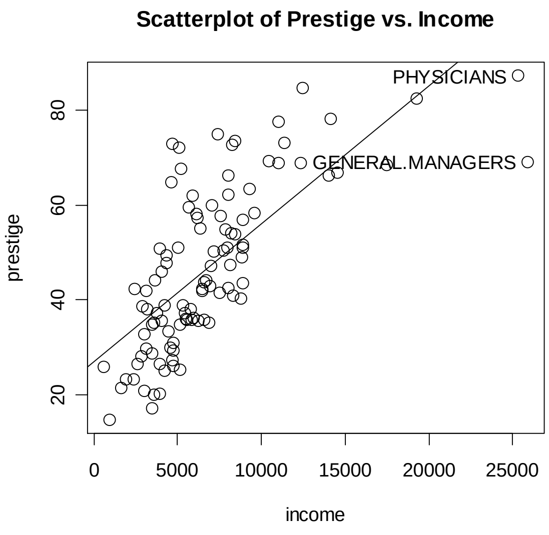


Figure 2. Simple model diagnostic plots in R

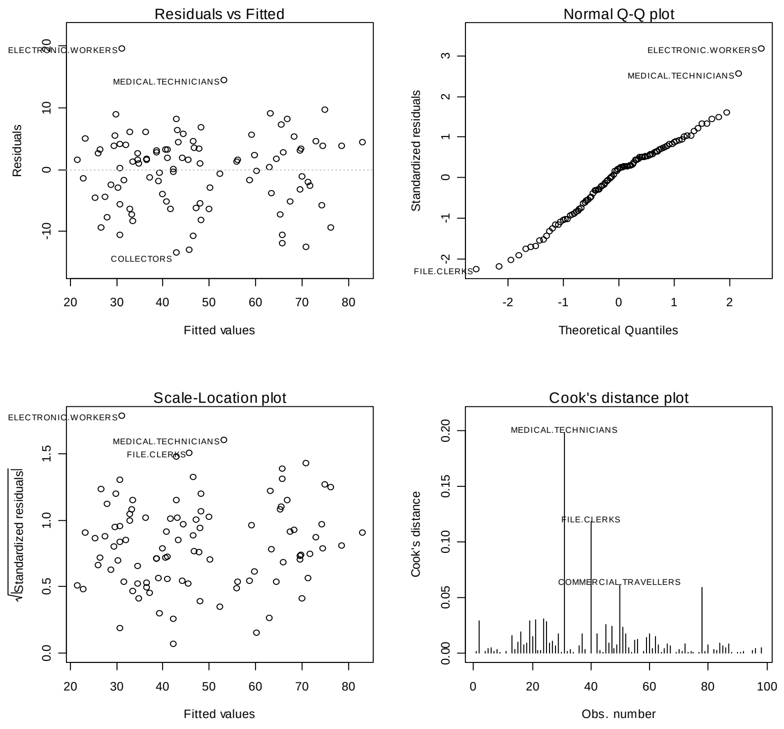
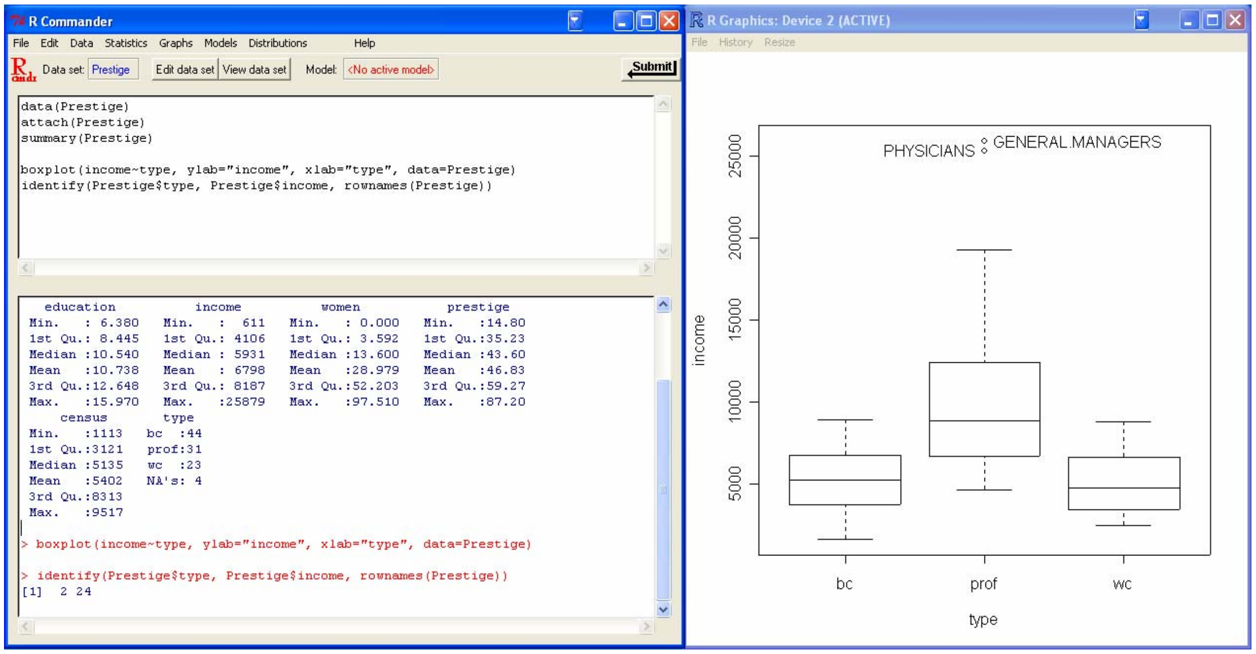


Figure 3. The R Commander GUI window and a graphics-device window.



```
data(Prestige)
attach(Prestige)
summary(Prestige)

boxplot(income~type, ylab="income", xlab="type", data=Prestige)
identify(Prestige$type, Prestige$income, rownames(Prestige))
```

```
   education          income           women            prestige
 Min.   : 6.380   Min.   :  611   Min.   : 0.000   Min.   :14.80
 1st Qu.: 8.445   1st Qu.: 4106   1st Qu.: 3.592   1st Qu.:35.23
 Median :10.540   Median : 5931   Median :13.600   Median :43.60
 Mean   :10.738   Mean   : 6798   Mean   :28.979   Mean   :46.83
 3rd Qu.:12.648   3rd Qu.: 8187   3rd Qu.:52.203   3rd Qu.:59.27
 Max.   :15.970   Max.   :25879   Max.   :97.510   Max.   :87.20
     census          type
 Min.   :1113   bc  :44
 1st Qu.:3121   prof:31
 Median :5135   wc  :23
 Mean   :5402   NA's: 4
 3rd Qu.:8313
 Max.   :9517

> boxplot(income~type, ylab="income", xlab="type", data=Prestige)

> identify(Prestige$type, Prestige$income, rownames(Prestige))
[1]  2 24
```

Figure 4. An R Commander dialog box to generate side-by-side boxplots.

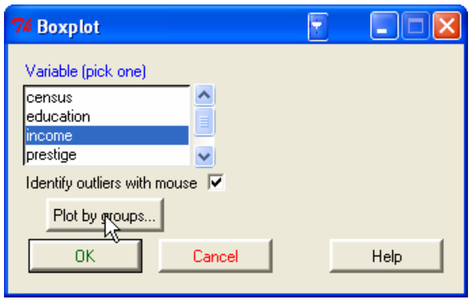
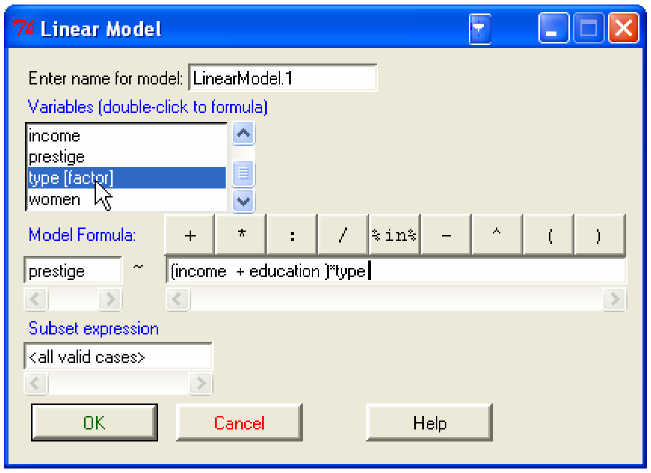


Figure 5. The R Commander menu tree.

```
File - Load script from file
    |- Save script
    |- Save script as
    |- Save output
    |- Save output as
    |- Save R workspace
    |- Save R workspace as
    |- Options
    |- Reset output width
    |- Exit - from Commander
           |- from Commander and R

Edit - Cut
    |- Copy
    |- Paste
    |- Delete
    |- Find
    |- Select all

Data - New data set
       |- Import data - from text file
    |             |- from SPSS data set
    |             |- from Minitab data set
    |             |- from STATA data set
    |- Data in packages - List data sets in packages
    |                  |- Read data set from attached package
    |- Active data set - Select active data set
    |                 |- Help on active data set (if available)
    |                 |- Variables in active data set
    |                 |- Set case names
    |                 |- Subset active data set
    |                 |- Remove cases with missing data
    |                 |- Export active data set
    |- Manage variables in active data set - Recode variable
                                          |- Compute new variable
                                          |- Standardize variables
                                          |- Convert numeric variable to factor
                                          |- Bin numeric variable
                                          |- Reorder factor levels
                                          |- Define contrasts for a factor
                                          |- Rename variables
                                          |- Delete variables from data set

Statistics - Summaries - Active data set
          |           |- Numerical summaries
          |           |- Frequency distribution
          |           |- Table of statistics
          |           |- Correlation matrix
          |- Contingency Tables - Two-way table
          |                    |- Multi-way table
          |                    |- Enter and analyze two-way table
          |- Means - Single-sample t-test
          |       |- Independent-samples t-test
          |       |- Paired t-test
          |       |- One-way ANOVA
          |       |- Multi-way ANOVA
          |- Proportions - Single-sample proportion test
          |             |- Two-sample proportions test
          |- Variances - Two-variances F-test
          |           |- Bartlett's test
          |           |- Levene's test
          |- Nonparametric tests - Two-sample Wilcoxon test
          |                     |- Paired-samples Wilcoxon test
          |                     |- Kruskal-Wallis test
          |- Dimensional analysis - Scale reliability
          |                      |- Principal-components analysis
          |                      |- Factor analysis
          |                      |- Cluster analysis - k-means cluster analysis
          |                                         |- Hierarchical cluster analysis
          |                                         |- Summarize hierarchical clustering
          |                                         |- Add hierarchical clustering to data set
          |- Fit models - Linear regression
                       |- Linear model
                       |- Generalized linear model
                       |- Multinomial logit model
                       |- Proportional-odds logit model

Graphs - Index plot
      |- Histogram
      |- Stem-and-leaf display
      |- Boxplot
      |- Quantile-comparison plot
      |- Scatterplot
      |- Scatterplot matrix
      |- 3D scatterplot
      |- Line graph
      |- Plot of means
      |- Bar graph
      |- Pie chart
      |- Save graph to file - as bitmap
                           |- as PDF/Postscript/EPS
                           |- 3D RGL graph

Models - Select active model
      |- Summarize model
      |- Add observation statistics to data
      |- Hypothesis tests - ANOVA table
      |                  |- Compare two models
      |                  |- Linear hypothesis
      |- Numerical diagnostics - Variance-inflation factors
      |                       |- Breusch-Pagan test for heteroscedasticity
      |                       |- Durbin-Watson test for autocorrelation
      |                       |- RESET test for nonlinearity
      |                       |- Bonferroni outlier test
      |- Graphs - Basic diagnostic plots
               |- Residual quantile-comparison plot
               |- Component+residual plots
               |- Added-variable plots
               |- Influence plot
               |- Effect plots

Distributions - Normal distribution - Normal quantiles
             |                     |- Normal probabilities
             |                     |- Plot normal distribution
             |- t distribution - t quantiles
             |                |- t probabilities
             |                |- Plot t distribution
             |- Chi-squared distribution - Chi-squared quantiles
             |                          |- Chi-squared probabilities
             |                          |- Plot chi-squared distribution
             |- F distribution - F quantiles
             |                |- F probabilities
             |                |- Plot F distribution
             |- Binomial distribution - Binomial quantiles
             |                       |- Binomial tail probabilities
             |                       |- Binomial probabilities
             |                       |- Plot binomial distribution
             |- Poisson distribuition - Poisson probabilities
                                     |- Plot Poisson distribution

Help - Commander help
    |- About Rcmdr
    |- Introduction to the R Commander
    |- Help on active data set (if available)
```

Figure 6. A linear-model dialog box from the R Commander.



Enter name for model: LinearModel.1

prestige ~ (income + education)*type

<all valid cases>

Figure 7: Estimated survival function at the mean values of the covariates.

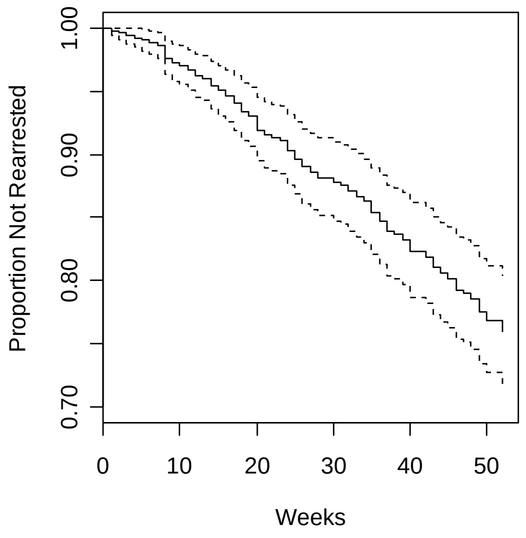
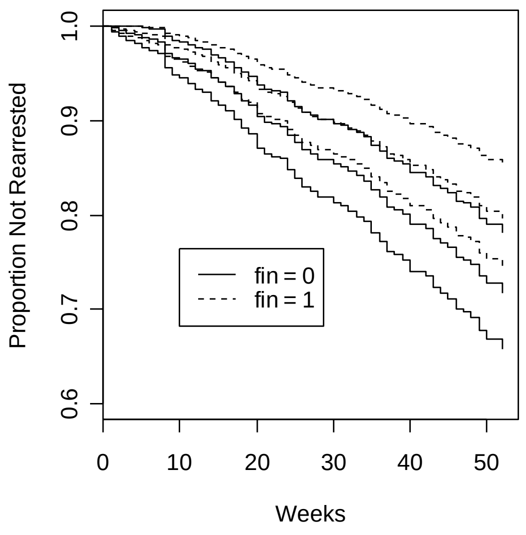


Figure 8: Estimated survival curves with 95-percent confidence envelopes for those receiving (`fin = 1`) and not receiving (`fin = 0`) financial aid.



NOTES

[1] See the 2003 ICPSR courses in Bayesian Methods for the Social and Behavioral Sciences, Maximum Likelihood Estimation for Generalized Linear Models, and Regression Analysis III: Advanced Methods, and the lecture series on Statistical Computing in S, at http://www.icpsr.umich.edu/training/summer/biblio/2003/index.html. Information about the 2004 Oxford Spring School is available at http://springschool.politics.ox.ac.uk/. Also see Fox (2002b) and Andersen, Fox, Franklin, and Gill (2003).

[2] At the time of writing, the current version of R is 2.0.1.