



A Skimpy Intro to R/S/S-Plus¹

History of Programs	1
Why shouldn't I use R/S/S-Plus?	2
Why should I use R/S/S-Plus?	2
Which software package should I get and where do I get it?	2
Where do I obtain more info on using R?	2
How does R work?	4
What do I need to know to get started?	5
I can't always see my data, how do I know what it looks like?	7
Ok, but I need to know how to get my data into R. How can I do this?	7
How do I get my output to a separate file?	8
How do I create or save projects?	8
How do I save the history of the command entries for a session?	9
Show me some examples, please?	9
But you said, R is great for graphics, show me?	12

¹ *Note:* Material is liberally lifted from numerous sources including memory – because this is in part based on memory there is no guarantee on the final quality of this product! The goals of this document are to provide a brief introduction and to provide summary information on where to find help/more information.

History of Programs

S was first developed by statisticians at Bell Labs. S is the programming language underlying all of these programs. S-Plus is the commercial version, complete with graphical user interface (GUI; or the point and click stuff), maintained and distributed by Insightful Corporation (located in Seattle, WA very near Microsoft). The current version of S-Plus is 7.0. R is the FREEWARE program developed by a consortium of statisticians and continuously updated. The current version of R is 2.3.1. Most, but not all, programs or syntax written for one will work in the other. These programs have become the choice of statisticians. There must be a reason.

Why shouldn't I use R/S/S-Plus?

- Requires patience
- Somewhat steep learning curve for R (no GUI)
- With flexibility comes responsibility
- Most of the time you actually need to understand what it is you are trying to do. There is much less taken for granted. If you simply want un-interpretable output, this is not the program for you.

Why should I use R/S/S-Plus?

- Graphics capabilities are remarkable
- Extremely flexible in abilities
- Fast and efficient
- S-Plus interfaces with Microsoft Office (PowerPoint and Excel)
- Once over the learning hurdle, can program almost anything AND save and repeat when you have multiple applications
- Can create your own functions/libraries/packages for future use
- Many add-on libraries exist. Examples: bootstrapping, multilevel analyses, ...

Which software package should I get and where do I get it?

If you are comfortable with using a command line interface (all syntax) then download R for free. I use both for different tasks. R and all libraries can be downloaded from the R website (see below). If you need the assistance of the GUI, then a student version (with limitations on data) can be downloaded from the Insightful website (see below). The cost of S-PLUS (last I checked) was about \$2k. Students may purchase S-Plus (including all the manuals) for about \$200. Libraries that were not packaged with your version of S-Plus may also be downloaded at the Insightful website.

**** From this point on, R/S/S-PLUS is shortened to R unless otherwise noted. ****

Where do I obtain more info on using R?

Help files

All R/S/S-Plus functions within libraries or packages have similarly formatted help files. These files can be both invaluable and uninterpretable. They are often written by a programmer that knows exactly how the functions work. For those of us in the dark, we have to hope to be able to interpret them. Fortunately there are other sources of help (see below). I find the online help files to be invaluable for ‘remembering’ how to specify various functions. Each of the following will return information from the help file on a given *function*:

```
?function  
help(function)  
help.search("key word in function")
```

```
# Comments are set apart by the ‘#’ key.  
# R/S/S-Plus is case sensitive!
```

In R can call the help menu by:

```
help.start()
```

here you will find all the search engines built into R and the online manuals

Websites

R project website:
www.r-project.org

S-Plus commercial website:
www.insightful.com

UCLA Statistical Website (has tons of useful info)
www.ats.ucla.edu/stat/seminars

When all else fails, Google it.

Books

Spector, P. (1994). *An introduction to S and SPLUS*. Belmont, CA: Duxbury Press.

William N. Venables and Brian D. Ripley. *Modern Applied Statistics with S. Fourth Edition*. Springer, 2002. ISBN 0-387-95457-0.

Jose C. Pinheiro and Douglas M. Bates. *Mixed-Effects Models in S and S-Plus*. Springer, 2000. ISBN 0-387-98957-0.

John Fox. *An R and S-Plus Companion to Applied Regression*. Sage Publications, Thousand Oaks, CA, USA, 2002. ISBN 0761922792

Schumaker, R. E. & Akers, A. (2001). *Understanding statistical concepts using S-Plus*. Mahwah, NJ: Lawrence Erlbaum Associates.

Richard M. Heiberger and Burt Holland. *Statistical Analysis and Data Display: An Intermediate Course with Examples in S-Plus, R, and SAS*. Springer Texts in Statistics. Springer, 2004. ISBN 0-387-40270-5.

Online manuals (can be printed or purchased)

Free downloads

R manuals page:

<http://cran.r-project.org/manuals.html>

An introduction to R can be downloaded at:

<http://cran.r-project.org/doc/manuals/R-intro.html>

The R language definition can be downloaded at:

<http://cran.r-project.org/doc/manuals/R-lang.html>

Additional manuals (PDFs) can be found at:

<http://cran.r-project.org/other-docs.html>

R manual for Beginners:

http://cran.r-project.org/doc/contrib/rdebuts_en.pdf

Listsers

** As with all listsers, do your homework first. You will receive helpful comments, but, you may also receive caustic remarks from individuals lurking about waiting to tell you how lazy you are (i.e., “read the online documentation first”). The listsers are archived and are a wonderful resource when you utilize the search feature.

R listserv

www.r-project.org/mail.html

S listserv

www.biostat.wustl.edu/s-news

How does R work?

R works by manipulating objects. From scalars, vectors, matrices to output from various statistical models, everything within R is considered an object game for manipulation. This means that when output is generated for a particular model, it is useful to store the model itself as an object for later use.

Because of the way in which R works with memory, it is far superior (faster) than SPSS and other similar canned packages. Computers with larger memory (i.e., > MB RAM) will tend to run programs much faster and more efficiently.

What do I need to know to get started?

Key data terms:

- Scalar – single variable/value
- Vector – a row or column of data
- List – collection of scalars or vectors or other objects – `c(1,2,3,4)`
- Array – multidimensional arrangement of lists
- Matrix – usually a 2-dimensional array (think spreadsheet); operating on matrices is computationally faster than operating on data.frames
- Data.frame – matrix with embedded formats such as a dataset from SPSS, Excel; operating on data.frames is usually syntactically simpler.

Key command line features:

>

Command prompt is where the next statement is typed

<-

Assignment operator. Similar to the “equals” sign (=), it denotes a left and right side expression. This is used for naming, specifying models, and other things. THERE IS NO OOPS BUTTON!
Example:

```
> x <- rnorm(10)
```

This creates a vector of 10 random normal values named ‘x’

```
> x
```

produces x, the 10 random values

Typing the command `ls()` or alternatively `objects()` lists all the objects in the current scope (what is available)

```
rm()
```

removes whatever is in the parentheses ()

Example:

```
rm(x)
```

Mathematical operations are straightforward:

```
> 2+2
```

```
4
```

```
> 2*8
```

```
16
```

```
> 16/4
```

```
4
```

```
> 2^3
```

```
8
```

```
...
```

Some operations are built in. These are called functions. For example, to get the mean of the x values generated above, (which should be zero, 0), simply use the mean function.

```
mean(x)
```

or, for the standard deviation

```
sd(x)
```

if you removed x , it's not there, (gone forever – no redo button)

you will have to create another object to use the mean as an example.

You can create your own function such as standardizing data if you desire

```
Z <- function (data)
{
  d <- data
  z <- (d - mean(d))/sd(d)
}
```

Notice that unfinished lines are followed by a '+' on the next line indicating to be continued

to check our function, lets create a data.frame of 10 values with mean 20, sd = 2.

to see how to use the `rnorm` function, let's call the help file

```
?rnorm
```

```
x <- rnorm(10,20,2)
```

```
x <- data.frame(x) # changes format of 'x' object to a data.frame
```

creates a temporary object called `zx` which contains the z scores

```
zx <- Z (x)
```

```
zx
```

```
x <- cbind(zx,x) # put zx in the x data.frame by combining columns
```

```
# Now on your own check the first value to see that it is indeed a z score of the first value # of x.
check to see that the mean of zx is 0 and the sd of zx is 1.
# Check to see what objects exist now?
ls()
```

I can't always see my data, how do I know what it looks like?

```
# Suppose you have a large matrix or data.frame, say 10 columns and 100 cases.
```

```
tmat <- matrix(rnorm(1000),ncol=10)
dim(tmat)
[1] 100 10    # tells us that the matrix has 100 rows and 10 columns
```

```
# suppose you want to extract the 3 & 4 columns and only the first 10 rows.
# subscripts for a matrix or object are in brackets [ ]
```

```
tmat2 <- tmat[1:10,c(3,4)]
# or alternatively, since 3 & 4 are adjacent
tmat2 <- tmat[1:10,3:4]
```

```
# check, look at tmat2 in its entirety
tmat2
```

```
# and then look at the first 12 rows of tmat
tmat[1:12, ]    # this gives the first 12 rows and all columns
```

```
# In R, use the data editor by selecting EDIT, Data Editor from the menu
# fix(tmat) also opens the editor
```

```
# In S-Plus,
# Use the data menu on the toolbar
```

Ok, but I need to know how to get my data into R. How can I do this?

Here, the programs R and S-Plus differ a bit. Since S-Plus has a pull-down menu for this, I would use it, unless you need to create a program that gets data in the middle – I do this a lot.

R uses the R library (foreign) to read SPSS and other data formats or the function *read.table* to read text files of various formats.

```
library(foreign)

help(read.spss)

help(read.table)
```

```
# Notice the specification of the slash, in R the slash is either a double back or single forward.
```

Check the latest version of R. The data entry functions are continuously updated.

In both R and S-Plus you can manipulate the data via a spreadsheet like appearance. See pull-down menus.

As an example ... suppose you have the following small dataset:

```
1 2 3
4 5 6
7 8 9
```

You can copy the data to the 'clipboard' by highlighting the data and right-clicking and selecting 'copy'

Then in R ...

```
tdat <- read.table("clipboard")
tdat # check to see that the data read properly
```

See also `library(Rcmdr)` for a nice point-and-click interface

How do I get my output to a separate file?

```
sink("C:\\temp.txt")
# when finished be sure to sink back to the command line
sink()
```

How do I create or save projects?

In R

File save workspace as ...

Clicking on the save icon will only save the project to the default location. You will now not have a 'clean' R file to work with. Everytime you open R, you will have the stuff you saved. Don't click on the save icon!

I find it nice to have a 'clean' R file, then save each project

I often keep a separate R project in whatever folder I am working on in my USB drive. This project will have all objects created (data, models, ... everything)

From the command line

```
> save.image ("myNewProject.Rdata")
```

In S-Plus

Can create separate project folders under the File menu. Data are saved in a folder named .data. Preferences (settings) are saved in a folder named .prefs.

Another method is to force S-Plus to ask each time it is opened where to start.

Yet another, create a shortcut and put somewhere (e.g., the desktop) and right click, under Target, enter **S_PROJ="C:\splus files"** after the location of the .exe, where everything in the "" is the location of the project you want to create/open ...

How do I save the history of the command entries for a session?

In R -> File – save history as ...

In S-Plus -> history button on tool bar and File – save history as ...

Show me some examples, please?

Example 1

Specify a linear regression model.

Call data *women* and check the names and dimensions in the data.frame

```
> data(women)
```

```
> names(women)
```

```
> dim(women)
```

Specify a linear model using height as a predictor of weight

```
> tmod1 <- lm (weight ~ height, women)
```

various functions will assist in interpreting the model and assessing fit

```
tmod1
```

```
coef(tmod1)
```

```
summary(tmod1)
```

```
resid(tmod1)
```

```
plot(tmod1)
```

```
anova(tmod1)
```

What does the object tmod1 contain?

```
objects(tmod1)    or,
```

```
names(tmod1)     or,
```

```
str(tmod1)
```

Suppose you only want the SE of the height effect; enables you to use the SE in subsequent programming. Notice the use of subscripts for the object

```
summary(tmod1)$coefficients[2,2]
```

```
summary(tmod1)$coefficients[2,3]    # extracts the t-value for the height effect
```

Specify models with interactions

In SPSS or other such programs, to specify an interaction, you must create the product term. In R, you simply specify the interaction and the program will ‘take care of the computations.’

```
lm.mod <- lm(Y ~ X*Z, data)          # is  $Y = X + Z + XZ$ 
```

```
lm.mod <- lm(Y ~X:Z, data)          # is  $Y = XZ$ 
```

Example 2

```
# A one-way ANOVA
# Create grouped data
mat <- data.frame(rep(c(1:4),5),rep(c("a","b","c","d"),5))
names(mat) <- c("Y","F")

# what is the mean of each level

by(mat$Y,mat$F,FUN=mean)

# In S-Plus can generate plot of means, medians, or other summary statistic by factor level
plot.factor(mat$Y~mat$F) # this isn't working for the current example
plot.design(mat,FUN=mean) # this isn't working for the current example

# ANOVA
tmod3 <- aov (Y ~ F, mat)
summary (tmod3)

# GLM
tmod3b <- lm (Y ~ F, mat)
summary (tmod3b)
anova (tmod3b)

# Note. You have to ask for Type III SS ...
```

Example 3

```
# A multilevel example using NLME (LME4 is available in R, haven't tried it yet)

library(nlme)

# This example uses data created by P. Bliese. Information and download is available at:
# http://www.sio.org/Klein/Klein.aspx

klein2000 <- read.csv("http://www.sio.org/Klein/simcomma.txt")
names(klein2000) # see what variables are in the data

# Ultimately, test the model that Job Satisfaction = Group-level cohesion and individual-level
work load.

# Create a unit level variable from individual data and merge files into one data.frame

g.cohes <- aggregate(klein2000$COHES,by=list(klein2000$GRPID),FUN=mean)

g.cohes[1:10,] # look at what we created

names(g.cohes) <- c("GRPID","G.COHES") # Change names in g.cohes
names(g.cohes) # Check names

# Create new merged data.frame
nklein <- merge(klein2000,g.cohes,by=("GRPID"))
```

```

# klein2000 <- merge(klein2000,g.cohes,by=("GRPID")) # This works too

# Assess 'null' model
# One-way 'ANOVA' to assess non-independence

js.null <- lme (JOBSAT ~ 1, random=~1|GRPID, data=klein2000,
na.action=na.omit)

VarCorr(js.null)

# ICC =  $\tau_{00} / \tau_{00} + \sigma^2 =$ 
> 0.6835257 / (0.6835257 + 5.4741228)
[1] 0.1110043 # 11% of variance in JS is due to group membership

intervals(js.null) # provides the 95% CI for model components
# tells us that with 95% certainty, the variability in
# means(intercepts) is non-zero.
# We have non-independence and variability at group level to account for.

# Then analyze a multilevel model

js.mod1 <- lme (JOBSAT ~ G.COHERS + WLOAD, random=~1|GRPID, data=nklein,
na.action=na.omit)

summary(js.mod1)
VarCorr(js.mod1)

# Variance Explained (use VarCorr results for both between and within)

#  $R^2 = 1 - ((\text{var w/ predictor}) / (\text{var w/o predictor}))$ 

# Level 2 or Between group variance explained
> 1 - (0.5127268 / 0.6835257)
[1] 0.2498793

# Level 1 or within group variance explained
> 1 - (5.1539412 / 5.4741228)
[1] 0.05849003

# Compare nested models

js.mod2 <- lme (JOBSAT ~ G.COHERS + WLOAD, random=~ WLOAD |GRPID, data=nklein,
na.action=na.omit)

```

```
# Use ANOVA to obtain deviance test (i.e., chi-sq difference)
# ANOVA does not work for lme with REML estimation IF fixed effects are not the same

anova(js.mod1, js.mod2)

# What can we conclude about the relationship of workload with job satisfaction in these data?
```

But you said, R is great for graphics, show me?

Univariate displays

```
> x -> rnorm(100)
```

```
# There are 3 types of graphics in R/S-Plus: editable, traditional and Trellis (see online manual
for distinctions
```

```
# In R must call library(lattice) to use the trellis features
```

```
library(lattice)
densityplot(x)
plot(density(x))
qqnorm(x)
bwplot(x)
bwplot(mat[,1]) # column 1 of the data.frame named mat
bwplot(x[1:20]) # plots only first 20 values of x
```

```
# One can create a function to produce four plots at once useful to assessing normality
```

```
eda.shape <- function(x,lab) {
  par(mfrow = c(2, 2))
  hist(x,col=0, main=lab)
  boxplot(x,boxcol=-1,medcol=1)
  iqd <- summary(x)[5] - summary(x)[2]
  plot(density(x, width = 2 * iqd),
  xlab = "x", ylab = "", type = "l")
  qqnorm(x, pch = 1)
  qqline(x)
  invisible()
}
```

```
# Create positively skewed data and graph
```

```
x <- rchisq(100,2)

eda.shape (x,"X-label")
```

Multivariate displays

Create data for graphing

```
tmat <- matrix(rnorm(300),ncol=3)
tmat <- data.frame(tmat)
names(tmat) <- c("x","z","e")
attach(tmat)
tmat$y <- .5*x +.5*z + x*z + .25*e
detach(tmat)
```

plot(x,y)

```
plot(tmat$x,tmat$y,xlab="X",ylab="Y",ylim=c(-1,1),col="Blue")
# or, equivalently ...
attach(tmat)
plot(x,y) # without the fluff
detach(tmat)
```

3D plots

See help file on `cloud` (3d scatterplot) run IRIS data example
Then, examine the tmat data with a 3d plot

```
cloud(y~x*z,data=tmat)
```

Plot the fitted model, create a lm and then plot fitted values instead of y

```
t1m <- lm(y~x*z,data=tmat)
cloud (t1m$fitted~x*z,data=tmat)
```

Panel plots

Based on Klein2000 data from Example 3 above

```
library(lattice)
trellis.device(device="windows",bg="white")
xyplot(JOBSAT~WLOAD|as.factor(GRPID),data=klein2000,panel=function(x,y)
{panel.xyplot(x,y)
panel.abline(lm(y~x,na.action=na.omit))}),
xlab="Work Load",ylab="Job Satisfaction")
```

What does this tell us about our models from Example 3 above?

Does the JobSat ~ Pay relationship differ by group?

```
trellis.device(device="windows",bg="white")
xyplot(JOBSAT~PAY|as.factor(GRPID),data=klein2000,panel=function(x,y)
{panel.xyplot(x,y)
```

```

        panel.abline(lm(y~x,na.action=na.omit))},
        xlab="Pay",
        ylab="Job Satisfaction")

# Another multilevel graphing example ## Plot LME Model

# specify model
tmod<- lmList(JOBSAT~PAY|GRPID,data=klein2000)

attach(klein2000)
#open plot
plot(PAY,JOBSAT,xlab="Pay", ylab="Job Satisfaction",type="n")

#create function to run all groups
lmplot<-function(X){
  for (i in 1:50){
    abline(X[[i]])
  }}

#plot
lmplot(tmod)

# Let's plot multiple relationships on the same page
# use klein2000 and put above plot example on 4 plots JS ~ pay, wload, posaff, tasksig

# Create 4 models to plot
tmod1 <- lmList(JOBSAT~PAY|GRPID,klein2000)
tmod2 <- lmList(JOBSAT~WLOAD|GRPID,klein2000)
tmod3 <- lmList(JOBSAT~POSAFF|GRPID,klein2000)
tmod4 <- lmList(JOBSAT~TASKSIG|GRPID,klein2000)

attach(klein2000)      #makes klein2000 'visible' to search path

# Creates plot 2 by 2
par (mfrow = c(2,2), bg="tan2")

# Plots in cell 1,1
par (mfg = c(1,1,2,2))
plot(PAY,JOBSAT,xlab="Pay",ylab="Job Satisfaction",type="n")
lmplot(tmod1)

# Plots in cell 2,1
par (mfg = c(2,1,2,2))
plot(WLOAD,JOBSAT,xlab="WorkLoad",ylab="Job Satisfaction",type="n")
lmplot(tmod2)

# Plots in cell 1,2
par (mfg = c(1,2,2,2))
plot(POSAFF,JOBSAT,xlab="Pos. Affect",ylab="Job Satisfaction",type="n")
lmplot(tmod3)

```

```
# Plots in cell 2,2
  par (mfg = c(2,2,2,2))
  plot(TASKSIG,JOBSAT,xlab="TaskSig",ylab="Job Satisfaction",type="n")
  lplot(tmod4)

  detach(klein2000)      # removes klein2000 from search path

# Going back to the sim data from before (tmat) create a page with all four possible x,y plots
attach(tmat)
par(mfrow=c(2,2))
plot(x,z)
plot(x,y)
plot(z,y)
plot(x*z,y)
detach(tmat)

# Notice the order of the graphs without specifying which cells to plot them in.
```