A Guide to Doing Statistics in Second Language Research Using R

Answers to Application Activities

Note: The name of the section is given before the answers.

Chapter 1: Getting Started with R and R Commander

Application Activities for Practicing Entering DATA into R

1. Answer:
Using R Commander, follow the menu sequence DATA > NEW DATA SET (remember, if you have only the R Console open, to get R Commander open you will need to type the sequence `>library(Rcmdr)` first). Enter the name "Count" in the first box that appears. A spreadsheet pops up. Click on "var1" and replace "var1" with the name "Score." Change the type to numeric and close the window. Click on "var2" and change the name to "Group." Now enter the numbers in the spreadsheet.

2. Answer:
Using R Commander, follow the menu sequence DATA > IMPORT DATA > FROM TEXT FILE, CLIPBOARD OR URL. Replace the default name for the file with "read." Keep the check mark on the box "Variable names in file," but change the Field Separator to "Other" and enter at least one space (you can enter more if you like!) in the white box next to this choice. Click OK. Navigate to where you have placed the file, and open it. Use the "View data set" button on R Commander to view it. It should have opened with 12 columns, all topped by the appropriate name.

3. Answer:
Using R Commander, follow the menu sequence DATA > IMPORT DATA > FROM SPSS DATA SET. Replace the default name with "dekeyser" and then press OK. Navigate to where the file exists on your computer and open it. Go back to R Commander and verify that the data set looks good.

Application Activities in Practicing Saving Data, Recognizing Missing Data, and Attaching and Detaching Data Sets

1. Answer:
Using R Commander, follow the menu sequence DATA > NEW DATA SET (remember, if you have only the R Console open, to get R Commander open you will need to type the sequence `>library(Rcmdr)` first). Enter the name "ClassTest" in the first box that appears. A spreadsheet pops up. Click on "var1" and replace "var1" with the name "Pretest." Change the type to numeric and close the window. Click on "var2" and change the name to "Posttest," and again change the type to numeric. Repeat the process for the last variable, "DelayedPosttest." Now enter the numbers in the spreadsheet. When you are finished, close the window. To check for missing data, type `>is.na(ClassTest)`. You should see two entries that say "TRUE."

2. Answer:
For example:

```
> ClassTest$Pretest
> attach(ClassTest)
> Pretest
```

3. Answer:
Using R Commander, first make sure your "ClassTest" is the active data set by looking at the window in R Commander. Then follow the menu sequence DATA > ACTIVE DATA SET > EXPORT ACTIVE DATA SET. Click on "Commas" for Field Separator. Erase the .txt portion of the name and add .csv. Click OK. If you now navigate to where you saved your file, you will see your file marked as an Excel spreadsheet, and that is what it will open up as if you click it.

Application Activities with Using R as a Calculator

1. Answer:
```
> 88+2689+331+389+2
[1] 3499
```

2. Answer:
```
>25338-89.32
[1] 25248.68
```

3. Answer:
```
> (59*26)/sqrt(24)
[1] 313.1264
```

4. Answer:
```
> 2500-(800+500+150+136+55+260+35+225)
[1] 339
```

5. Answer:
```
> ((275+38)^3)/6
[1] 5110716
```

6. Answer:
```
> ((258367/268)+245+897)*(4^20)
```
[1] 2.315633e+15 (which means take the decimal point and move it over to the right 15 places, adding zeros so that you can! The result is 2315633000000000).

7. Answer:
Hit the arrow up button to go back to your original calculation in activity 4 and easily change the 800 to 850; then add the three more expenses to the end of the calculation (this might be faster than retyping it!).
```
>2500-(850+500+150+136+55+260+35+225+63.24+ 35.10+180)
[1] 10.66
```

8. Answer:
```
> ((42*84)/(90+266+35))^3
[1] 734.6077
```

Application Activities in Creating Objects

1. Answer:
```
> Years=c(1976, 2003, 2009, 1900)
```

2. Answer:
For example:

```
>MyFamily=c("Jenifer", "Calvin")
>MyFamily
[1] "Jenifer" "Calvin"
```

3. Answer:
For example:

```
> Salary=c(2500)
> Salary
[1] 2500
> Expenses=c(700, 350, 225, 96, 42, 33)
> Budget=c(Salary-sum(Expenses))
> Budget
[1] 1054
```

Application Activities with Types of Data

1. Answer:
First, import beq using R Commander sequence DATA > IMPORT DATA > FROM SPSS DATA SET and call it beq.

```
> class(beq)
[1] "data.frame"
```

2. Answer:

```
> class(beq$CatDominance)
[1] "factor"
```

3. Answer:

```
> summary(beq$CatDominance) #There are 4 levels
YES NO YESPLUS NA's
558 105 373 542
```

4. Answer:

```
> class(Years)
[1] "numeric"
> class(MyFamily)
[1] "character"
```

Application Activities with Functions

1. Answer:
First, import dekeyser using R Commander sequence DATA > IMPORT DATA > FROM SPSS DATA SET and call it dekeyser.

```
> var(dekeyser$GJTScore)
[1] 876.3127
> sd(dekeyser$GJTScore)
```

```
[1] 29.60258
> 29.60258^2
[1] 876.3127
```

2. Answer:

```
> mean(dekeyser$Age)
[1] 21.56140
```

3. Answer:

```
> max(dekeyser$Age)
[1] 40
```

4. Answer:
Import LarsonHallPartial.sav using R Commander sequence DATA > IMPORT DATA > FROM SPSS DATA SET and name this file partial. Use the names() function to see what the variables are.

```
> names(partial)
[1] "id" "age" "LOR" "aptitude" "R_L_accuracy"
> length(partial$id)
[1] 15 # 15 participants
> mean(partial$R_L_accuracy)
[1] 3.783389
> sd(partial$R_L_accuracy)
[1] 0.722244
```

Application Activities for Manipulating Variables

Combining or Recalculating Variables

1. Answer:
Import torres data frame into R using R Commander sequence DATA > IMPORT DATA > FROM SPSS DATA SET and name this file partial.

DATA > MANAGE VARIABLES IN ACTIVE DATA SET > COMPUTE NEW VARIABLE

In the box "Expression to compute" put: (listening+ speaking)/2. (Note: there seem to be two variables, one named "listening" and one named "listenin," which are exactly the same; use either one!) In the box "New variable name" put "OralPreferences." Use the "View data set" button on R Commander to verify this column was added.

```
> range(torres$OralPreferences)
[1] 1 5
```

2. Answer:
DATA > MANAGE VARIABLES IN ACTIVE DATA SET > COMPUTE NEW VARIABLE

In the box "Expression to compute" put: Post.test- Pre.test. In the box "New variable name" put "gainscore."

```
> mean(mcguireSR$gainscore)
```

```
[1] 11.35053
```

3. Answer:
DATA > MANAGE VARIABLES IN ACTIVE DATA SET > COMPUTE NEW VARIABLE

In the box "Expression to compute" put: (R_L_accuracy*100)/7. In the box "New variable name" put "PercentAccuracy."

```
> max(partial$PercentAccuracy)
[1] 68.03571
```

Creating Categorical Groups

1. Answer:
DATA > MANAGE VARIABLES IN ACTIVE DATA SET > RECODE VARIABLES

Select the NumberOfLang variable. Put in new variable name "Glots." In box the labeled "Enter recode directives" write:

```
"Two"="Minorglots"
"Three"="Minorglots"
"Four"="Majorglots"
"Five"="Majorglots"
```

Note that these directives must be written *exactly* as shown here, with the same capitalization and punctuation!

In the R Console, use the `summary()` command.

```
> summary(beq$Glots)
```

2. Answer:

```
> mean(mcguireSR$Pre.test)
[1] 137.7547
> max(mcguireSR$Pre.test)
[1] 187.2
```

DATA > MANAGE VARIABLES IN ACTIVE DATA SET > RECODE VARIABLES

Select the Pre.test variable. Put in new variable name ("rate"). In the box labeled "Enter recode directives" write:

```
0:137.74="slow"
137.75:187.2="fast"
```
Verify: > mcguireSR$rate

3. Answer:
DATA > MANAGE VARIABLES IN ACTIVE DATA SET > RECODE VARIABLES

Select the "beginner" variable. Put in new variable name. In the box labeled "Enter recode directives" write:

1="strong"
2="moderate"
3="neutral"
4="moderate"
5="strong"

R code:

```
> summary(torres$TypeOfPreference)
```

Deleting Parts of a Data Set

1. Answers:
a. sw[1:3] #shows first 3 columns, all 5 rows
b. sw[1:3,] #shows first 3 rows, all 4 columns
c. sw[,1:3] #shows first 3 columns, all 5 rows
d. sw[4:5, 1:3] #shows 4th and 5th rows, columns 1 thru 3
e. sw[[1]] #shows the data for the first column, all rows
f. sw[1,] #shows the data for the first row, all columns
g. sw["C",] #shows data for the row that starts with a "C"

2. Answer:
R code:

```
> sw$Examination<-NULL
> sw #examine the data set again to make sure this column is gone
```

3. Answer:

```
> length(ChickWeight$Diet)
[1] 578 #Original number of entries
> ChickDiet<-subset(ChickWeight, subset=Diet!=4)
> length(ChickDiet$Diet)
[1] 460 #Number who are left after excluding those with Diet #4
```

Getting Data in the Correct Form for Statistical Tests

1. Answer:
First make sure dekeyser is your active data set! In R Commander: DATA > ACTIVE DATA SET > SUBSET ACTIVE DATA SET. Click off "Include all variables" because I only want the score; select GJTScore

Subset expression is:

Status=="Under 15" (has 15 entries)
Status=="Over 15" (has 42 entries)

In R code:

```
> DkCats1<-subset(dekeyser,subset=Status=="Under 15", select=c(GJTScore))
> DkCats2<-subset(dekeyser,subset=Status=="Over 15", select=c(GJTScore))
```

2. Answer:
In R Commander: DATA > ACTIVE DATA SET > SUBSET ACTIVE DATA SET. Click off "Include all variables" because I only want the gain score; select gnsc1.1. In order to get the correct subset expression, first look at the levels of the Treatment variable.

```
> levels(OStory$Treatment)
[1] "No music No pics" "No music Yes pics"
[3] "Yes music No pics" "Yes music Yes pics"
```

Subset expression is:

Treatment1=="No music Yes pics"; called this NoMusicYesPics as name of new data set (this has 17 entries)

Be sure to click on "Data set" on the R Commander window to get back to OStory before subsetting again.

Treatment1=="Yes music Yes pics"; name is YesMusicYesPics (17 entries)
Treatment1=="Yes music No pics"; name is YesMusicNoPics (15 entries)
Treatment1=="No music No pics"; name is NoMusicNoPics (15 entries)

In R code:

```
NoMusicYesPics <- subset(OStory, subset=Treatment1=="No music Yes pics",
select=c(gnsc1.1))
YesMusicYesPics <- subset(OStory, subset=Treatment1=="Yes music Yes pics",
select=c(gnsc1.1))
YesMusicNoPics <- subset(OStory, subset=Treatment1=="Yes music No pics",
select=c(gnsc1.1))
NoMusicNoPics <- subset(OStory, subset=Treatment1=="No music No pics",
select=c(gnsc1.1))
```

3. Answer:
In R Commander, choose DATA > ACTIVE DATA SET > STACK VARIABLES IN ACTIVE DATA SET. Choose all four variables; change name to "EffortLong"; change name for variable to Score and name for factor to Condition. Press OK and you will have a two-column data frame.

```
> mean(EffortLong$Score) #to get the average
```

In R Code:

```
verbsLong <- stack(verbs[, c("AdultIrregV","AdultRegVerb","ChildIrregVerb",
"ChildRegVerb")])
names(verbsLong) <- c("Score", "Group")
```

4. Answer:
In R Commander, choose DATA > ACTIVE DATA SET > STACK VARIABLES IN ACTIVE DATA SET. Choose all four variables; change name to "ArabicLong"; change name for variable to Score and name for factor to Contrast. Press OK and you will have a two-column data frame.

```
> mean(ArabicLong$Score)
```

In R code:

```
ArabicLong <- stack(Arabic[, c(“dh_D”,”GLOTTALSTOP_PHARYNGEALFRIC”,”t_T”,
“th_dh”)])
names(ArabicLong) <- c(“Score”, “Contrast”)
```

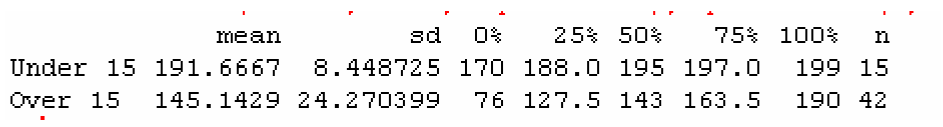
Chapter 3: Describing Data Numerically and Graphically

Application Activities with Numerical Summaries

1. Answer:
In R commander, first make sure the dekeyser file is the active data set. Then go to STATISTICS > SUMMARIES > NUMERICAL SUMMARIES. Choose the variable “GJTScore” and in the “Summarize by groups” button choose “Status.” This gives mean, standard deviation, quantiles, and N.

Here is the R code for this command:

```
numSummary(dekeyser[,"GJTScore"], groups=dekeyser$Status,
statistics=c("mean", "sd", "quantiles"), quantiles=c(0,.25,.5,.75,1))
```

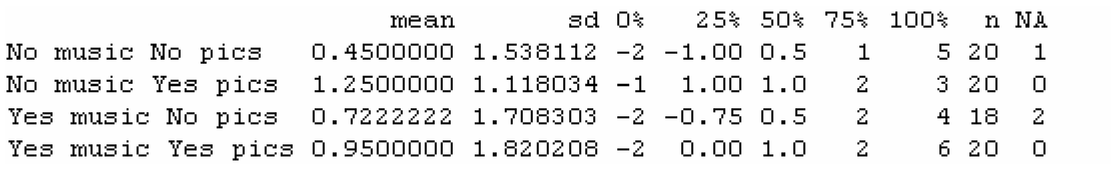


```
              mean        sd  0%   25% 50%   75% 100%  n
Under 15 191.6667  8.448725 170 188.0 195 197.0  199 15
Over 15  145.1429 24.270399  76 127.5 143 163.5  190 42
```

2. Answer:
In R commander, go to STATISTICS > SUMMARIES > NUMERICAL SUMMARIES. Choose the variable “gnsc1.1” and in the “Summarize by groups” button choose “trtmnt1.” This gives mean, standard deviation, quantiles, and N.

Here is the R code for this command:

```
numSummary(obarow[,"gnsc1.1"], groups=obarow$trtmnt1, statistics=c("mean",
"sd", "quantiles"), quantiles=c(0,.25,.5,.75,1))
```



```
                          mean       sd 0%   25% 50% 75% 100%  n NA
No music No pics     0.4500000 1.538112 -2 -1.00 0.5   1    5 20  1
No music Yes pics    1.2500000 1.118034 -1  1.00 1.0   2    3 20  0
Yes music No pics    0.7222222 1.708303 -2 -0.75 0.5   2    4 18  2
Yes music Yes pics   0.9500000 1.820208 -2  0.00 1.0   2    6 20  0
```

Application Activities for Exploring Assumptions

Checking Normality

1. Answer:
To look at Q-Q plots:

```
library(lattice) #if not open yet
qqmath(~Pronunciation|Group,data=FYL, layout=c(3,3),
xlab= "Score on pronunciation test",
prepanel=prepanel.qqmathline,
panel=function(x, ...){
panel.qqmathline(x,...)
panel.qqmath(x,...)
})
```

The reason this doesn't produce nice names on each box is that the Group variable is not a factor, whereas it was for the "forget" data set. You can make the Group variable a factor and try it again:

```
FYL$Group=factor(FYL$Group)
is.factor(FYL$Group) #this will check that it is indeed a factor now
```

It looks as though there is departure from normality in a number of groups, especially the oldest groups (the ones at the top of the figure).

2. Answer:

```
Hist(dekeyser$GJTScore[1:15], col="darkgrey") #Rows 1–15 are the "Under 15" group
```

This histogram is heavily negatively skewed, with most participants doing quite well.

```
Hist(dekeyser$GJTScore[16:57], col="darkgrey") #Rows 16–57 are the "Over 15" group
```

This histogram is more normally distributed, with the majority of answers between 120 and 160.

Stem and Leaf:

Start by using R Commander. Choose GRAPHS > STEM AND LEAF DISPLAY. The problem is this does not divide according to groups. Use the R code found in the Script Window of R Commander and modify it for the two groups by giving specific row numbers as used above for the histogram.

```
stem.leaf(dekeyser$GJTScore[1:15], trim.outliers=FALSE, na.rm=TRUE)
stem.leaf(dekeyser$GJTScore[16:57], trim.outliers=FALSE, na.rm=TRUE)
```

The stem and leaf for the "Under 15" is definitely skewed. The stem and leaf for the "Over 15" doesn't look quite as normally distributed as the histogram did.

3. Answer:

```
library(lattice) #if not open yet
qqmath(~CompGain1|Cond,data=lyster,
prepanel=prepanel.qqmathline,
panel=function(x, ...){
panel.qqmathline(x,...)
panel.qqmath(x,...)
})
```

Data here look pretty close to the line and fairly normally distributed.

```
qqmath(~CompGain2|Cond,data=lyster,
prepanel=prepanel.qqmathline,
panel=function(x, ...){
panel.qqmathline(x,...)
panel.qqmath(x,...)
})
```

Data here seem less normally distributed, with some curvature at the ends of the lines, and some serious departure from the line for the FFIprompt group.

Checking Homogeneity of Variance

1. Answer:
Make sure FYL is the active data set in R Commander. Choose STATISTICS > SUMMARIES > NUMERICAL SUMMARIES. Choose the variable of "Pronunciation" and in the "Summarize by groups" box choose "Group" This "should" work but doesn't, because Group has not been defined as a factor. You can make Group a factor or use the R code instead. Here's how to make Group into a factor using R code:

```
FYL$Group=factor(FYL$Group)
is.factor(FYL$Group) #this will check that it is indeed a factor now
```

If you now want to go back and use the R Commander syntax, you'll have to choose a different data set than FYL; then go back and choose FYL (this then reattaches the updated file).

If you wanted to just use the R code without changing Group to a factor, it works just fine:

```
numSummary(FYL[,"Pronunciation"], groups=FYL$Group, statistics=c("mean",
"sd", "quantiles"), quantiles=c(0,.25,.5,.75,1))
```

There is some substantial variation in standard deviations, especially for the youngest groups. After about group 4 or 5 the variance for the rest of the groups is fairly homogeneous.

2. Answer:
Make sure "dekeyser" is the active data set in R Commander. Choose STATISTICS > SUMMARIES > NUMERICAL SUMMARIES. Choose the variable of "GJTScore" and in the "Summarize by groups" box choose "Status." The summary shows that the standard deviations are extremely different. The groups do not have homogeneous variances.

R code:

```
numSummary(dekeyser[,"GJTScore"], groups=dekeyser$Status,
statistics=c("mean", "sd", "quantiles"), quantiles=c(0,.25,.5,.75,1))
```

3. Answer:
Make sure "lyster" is the active data set in R Commander. Choose STATISTICS > SUMMARIES > NUMERICAL SUMMARIES. Choose the variable of "CompsGain2" and in the "Summarize by groups" box choose "Cond." The summary shows that the standard deviations are extremely

different, ranging from a low of 0.4 to a high of 8.8. Even if the Comparison group were removed, the other groups do not appear to have homogeneous variances.

R code:

```
numSummary(lyster[,"CompGain2"], groups=lyster$Cond, statistics=c("mean",
"sd", "quantiles"), quantiles=c(0,.25,.5,.75,1))
```

Application Activity for Transformations

To correct substantial positive skewing Tabachnick and Fidell (2001) recommend using the square root transformation.

We want to work only with Group 11 and the variable of Pronunciation, so first subset by choosing DATA > ACTIVE DATA SET > SUBSET DATA SET. Click off "Include all variables" and just select Pronunciation. Then in the "Subset expression" box put "Group=="11" and give it a new name. Here is the R code for this:

```
FYL11 <- subset(FYL, subset=Group=="11", select=c(Pronunciation))
```

To do the transformation, make sure FYL11 is the active data set; then in R Commander choose DATA > MANAGE ACTIVE VARIABLES IN DATA SET > COMPUTE NEW VARIABLE.

I put the expression "sqrt(Pronunciation)" in the "Expression to compute" box and called mine "PronT." Here is the R code for this:

```
FYL11$PronT <- with(FYL11, sqrt(Pronunciation))
```

Now I can examine a histogram of the non-transformed Pronunciation variable and then the transformed one. Choose GRAPHS > HISTOGRAM. The R code is:

```
Hist(FYL11$Pronunciation)
Hist(FYL11$PronT)
```

The transformed variable is certainly more normal than the original variable, although it may not be totally normally distributed (and close is not really "good enough" for the assumption of normality, as I've stated!).

Chapter 6: Correlation

Application Activities with Creating Scatterplots

1. Answer:
For DeKeyser (2000) in R Commander choose GRAPHS > SCATTERPLOT. Choose Age for one variable and GJTScore for the other. Three boxes are ticked; you can switch the ticks off or leave them on (marginal boxplots, least squares line—that's the regression line—and smooth line). I asked you to identify outliers, so tick the box that says "Identify points" under Options. Press OK.

R code is:

```
scatterplot(GJTScore~Age, reg.line=lm, smooth=TRUE,
labels=rownames(dekeyser), boxplots='xy', span=0.5,
data=dekeyser)
```

One extreme point is 48, clearly an outlier. Regression line may not be a good fit to Loess. Loess seems to go flat around age 20 (although there's not a lot of data between 14 and 20, as DeKeyser points out in his article).

2. Answer:
For DeKeyser (2000) data split by groups. If necessary, change the Status variable to "character" instead of "numeric" with this command:

```
dekeyser$STATUS=factor(dekeyser$STATUS)
```

To make the scatterplot, choose GRAPHS > SCATTERPLOT. Pick Age for the x-axis variable and GJT Score for the y-axis variable. Open the Plot by groups button and pick Status; click OK. Untick the Boxplots default (the boxplots chart the entire data set, not the separate groups). Click OK and the graph appears.

The R code for this is:

```
scatterplot(GJTScore~Age | Status, reg.line=lm, smooth=TRUE,
labels=FALSE, boxplots=FALSE, span=0.5, by.groups=TRUE,
data=dekeyser)
```

Regression lines on each group of variables show that there is a slightly negative slope on the group of participants age 15 or less, but the regression line on the older group appears to be flat, indicating no decline on GJT scores with age over 15.

3. Answer:
For Larson-Hall's (2008) Japanese learners, make the scatterplot by choosing GRAPHS > SCATTERPLOT. Pick totalhrs for the one variable, and gjtscore for the other variable. I asked you to identify outliers, so tick the box that says "Identify points" under Options. Press OK.

The code for this is:

```
scatterplot(gjtscore~totalhrs, reg.line=lm, smooth=TRUE,
labels=rownames(larsonhall2008), boxplots='xy', span=0.5,
data=larsonhall2008)
```

The data points seem fairly scattered about and do not closely congregate around a line. However, the Loess line and regression line are fairly close together and do trend slightly upwards, so it could be concluded there is a slight linear relationship. As for outliers, the boxplots along the margins show three outliers along the x-axis of total hours. These are identified as cases 83, 199, and 200.

4. Answer:
For Larson-Hall's (2008) Japanese learners divided by groups, follow the same procedure as in activity 4, but this time click the "Plot by Groups" button and choose erlyexp.

R code for this is:

```
scatterplot(totalhrs~gjtscore | erlyexp, reg.line=lm,
smooth=TRUE, labels=FALSE, boxplots='xy', span=0.5,
by.groups=TRUE, data=larsonhall2008)
```

The Loess and regression lines seem to have some significant differences from each other.

5. Answer:
For Dewaele and Pavlenko's BEQ data, in R commander choose GRAPHS > SCATTERPLOT, and then agesec for one variable and l2speak for the other. Press OK.

The R code is:

```
scatterplot(l1speak~agesec, reg.line=lm, smooth=TRUE,
labels=FALSE, boxplots='xy', span=0.5, data=beqSwear)
```

This scatterplot looks odd because the answers are so discrete for the L2 speaking rating (there are only five choices). It would be hard to discern trends here, but one could say they are definitely not linear!

Application Activities with Calculating Correlation Coefficients

Pearson Correlation

1. Answer:
DeKeyser 2000 data
We have already checked this data for linearity by looking at the scatterplot in the Application activity "Correlation.Application Activity_ Creating scatterplots." We assume that the variables were independently gathered. We should check variables for normality in each group (use R Console here):

```
library(fBasics)
basicStats(dekeyser$GJTScore[1:15])
```

"Under 15" group is skewed (skewness is over 1).

```
basicStats(dekeyser$GJTScore[16:57])
```

"Over 15" group seems fine.

```
basicStats(dekeyser$Age[1:15])
```

Kurtosis is over 1.

```
basicStats(dekeyser$Age[16:57])
```

This seems fine.

To obtain the correlation in R Commander, make sure `dekeyser` is the active data set; then choose STATISTICS > SUMMARIES > CORRELATION MATRIX. Choose "Age" and "GJTScore" as variables (use the Ctrl button to choose both); leave the Pearson button marked and click

“Pairwise p-values.” Press OK. This returns the Pearson *r*=−.62 for the entire group, *n*=57, *p*<.0001 (I’m writing this as less than .0001, but in fact the results say the *p*-value is zero—however, it is *never* zero, so just write that it is smaller than a small number); this correlation coefficient indicates a large effect size.

The R code for this was:

```
rcorr.adjust(dekeyser[,c("Age","GJTScore")], type="pearson")
```

To obtain the correlation for the groups, note that “Under 15” are rows 1:15, and “Over 15” are rows 16:57. Use R Console:

```
rcorr.adjust(dekeyser[1:15,c("Age","GJTScore")], type="pearson")
rcorr.adjust(dekeyser[16:57,c("Age","GJTScore")], type="pearson")
```

For the “Under 15,” *r*=−.26, *n*=15, *p*=.35; for the “Over 15,” *r*=−.03, *n*=42, *p*=.86. Effect sizes become small or negligible when the groups are divided in this way.

2. Answer:
Flege, Yeni-Komshian, and Liu (1999) data. We have already checked this data for linearity for the relationship between AOA and PronEng (see the online document “Correlation.Creating scatterplots”), and found that it could be considered linear, although it might best be considered a third-order function. Check the relationship between AOA and PronKor by creating a scatterplot (I leave it to the reader to do this). We should check for normality for each group. This would involve subsetting the data into the different groups or checking the row numbers and putting them in, as was done in activity 1 above. An example for group 11 in English pronunciation, in rows 217–240, would be this code (used in R Console):

```
basicStats(flegeetal1999$PronEng[217-240])
Hist(flegeetal1999$PronEng[217-240])
```

To calculate the correlation coefficient, in R Commander choose STATISTICS > SUMMARIES > CORRELATION MATRIX. Choose AOA, PronEng and PronKor; leave the Pearson button marked, and click “Pairwise p-values.” For AOA and PronEng, *r*=−.85, *n*=240, *p*<.0001; for AOA and PronKor, *r*=.74, *n*=240, *p*<.0001; for PronEng and PronKor, *r*=−.65, *n*=240, *p*<.0001. In other words, there are strong effect sizes among all three of these variables.

3. Answer:
Larson-Hall (2008) data. We assume that variables were independently gathered. We need to check this data for linearity. In R Commander, choose GRAPHS > SCATTERPLOT MATRIX and enter the variables “useeng,” “gjtscore,” and “likeeng.” Choose to have histograms on the diagonal to examine distributions.

R code is:

```
scatterplot.matrix(~gjtscore+likeeng+useeng, reg.line=lm,
smooth=TRUE, span=0.5, diagonal = ‘density’,
data=larsonhall2008)
```

The regression line seems to match the Loess line fairly well except in the case of useeng and gjtscore, where the Loess line is curved. We can also note that in the case of useeng and likeeng the data are not randomly scattered. There is a fan-effect, which means the data are heteroscedastistic (variances are not equal over the entire area). This combination would probably not satisfy the parametric assumptions. There could be some outliers in the likeeng~useeng combination, and also in the useeng~gjt combination.

Next, check for normality of distribution of the variables. In R Commander, choose GRAPHS > HISTOGRAM. Use of English is highly positively skewed. Most Japanese learners of English don't use very much English. The degree to which people enjoy studying English is more normally distributed, although there seems to be too much data on the far right end of the distribution to be a normal distribution. The GJT scores seem fairly normally distributed.

Run the correlation on all of the variables. To return *p*-values I'll use STATISTICS > SUMMARIES > CORRELATION TEST in R Commander and do each of the three correlations one at a time.

R code for one pairing is:

```
cor.test(larsonhall2008$likeeng, larsonhall2008$useeng,
alternative="two.sided", method="pearson")
```

Remember that, with enough N, any statistical test can become "significant"! I have a very large N, so the question is, how large is the effect size? See the correlation coefficient for the answer (I leave it to the reader to test these).

4. Answer:
Dewaele and Pavlenko's BEQ data. Check for linearity with scatterplots, using a scatterplot matrix to examine all three variables at one time.

R code:

```
scatterplot.matrix(~agesec+l1comp+l1speak, reg.line=lm,
smooth=TRUE, span=0.5, diagonal = 'density', data=beqSwear)
```

Because answers are so discrete on L1comp and L1speak, it is difficult to call the data linear at all. We could also look at normality with histograms.

R code for this is:

```
Hist(beqSwear$l1comp, scale="frequency", breaks="Sturges",
col="darkgray")
```

This histogram is highly negatively skewed, as most people report comprehending their L1 quite well (a 5 on the scale!). These results may lead us to reconsider using a correlation to understand this data. (To be fair to the authors, I have used their data for illustration purposes in this book, and I don't know whether they looked at correlations among this data!) I leave it to the reader to look at the histograms of the other variables.

To run the correlation coefficient (and we'll use Spearman's too, since the assumptions for parametric correlation do not seem to hold well for these variables), in R Commander choose STATISTICS > SUMMARIES > CORRELATION TEST.

Here is the R code for Spearman's test for the pairing of "agesec" and "l1comp":

```
cor.test(beqSwear$agesec, beqSwear$l1comp,
alternative="two.sided", method="spearman")
```

The Spearman's coefficient of rho=.08, *p*=.008 would indicate a very small effect size. I leave it to the reader to perform the remaining correlations.

Robust Correlation

1. Answer:
Flege, Yeni-Komshian, and Liu (1999) data. To do robust correlation you'll need to open the "mvoutlier" library. All of the robust commands need to be done using R Console, not R Commander. I'm going to attach the data set first so I don't have to type so much! Here is the code I used:

```
attach(flegeetal1999)
library(mvoutlier)
cor.plot(AOA, PronEng)
```

There is no large difference between classical and robust for this pairing.

```
cor.plot(AOA, PronKor)
```

There is a sizeable difference between the classical and robust correlations, although the correlation is still large and strong.

```
cor.plot(PronKor, PronEng)
```

Again, there is a sizeable difference between the classical and robust correlations. The graphs help show why the correlations are different (robust data have a principled method of excluding some data, while the classical correlation keeps all data in).

```
detach(flegeetal1999)
```

2. Answer:
Larson-Hall (2008) data. Use the same strategy as in activity 1:

```
attach(larsonhall2008)
cor.plot(useeng,gjtscore)
Error in svd(cov(x), nv = 0) : infinite or missing values in 'x'
```

The error message shows there is a problem with missing data. The command will not work with missing data. Go to the online document "Describing data numerically and graphically.Imputing missing data" to review how to impute data (however, I will show it quickly here as well).

```
library(mice)
imp<-mice(larsonhall2008)
complete(imp)
implarsonhall2008<-complete(imp)
detach(larsonhall2008)
attach(implarsonhall2008)
cor.plot(useeng,gjtscore)
```

With the data imputed we are now able to use the `cor.plot()` command successfully. There is quite a large difference between the classical and robust scores; the robust stats find basically no correlation for these two variables.

```
cor.plot(likeeng,gjtscore) #note that this is still using the imputed file
```

The robust correlation is considerably smaller than the classical.

```
cor.plot(likeeng,useeng)
```

There is not too much difference between classical and robust.

```
detach(implarsonhall2008)
```

3. Answer:
Dewaele and Pavlenko's BEQ data. We found in activity 4 above that a parametric correlation was not a good way to deal with this data because of the violation of assumptions. A robust correlation would be a better way, if you decided that correlation was the best way to analyze this data.

```
attach(beqSwear)
cor.plot(agesec,l2speak)
Error in svd(cov(x), nv = 0) : infinite or missing values in 'x'
```

Once again we find that there is missing data, so we need to impute this data set before continuing.

```
imp<-mice(beqSwear)
complete(imp)
impbeqSwear<-complete(imp)
detach(beqSwear)
attach(impbeqSwear)
cor.plot(agesec,l2speak)
```

There is not much difference between correlation coefficients, although the graph shows a large difference in the data that is included.

```
detach(impbeqSwear)
```

Chapter 7: Multiple Regression

Application Activities with Graphs for Understanding Complex Relationships

Coplots

1. Answer:
Lafrance and Gottardo (2005) data. You'll need to use the R Console to perform a coplot. Here is the code I used:

```
attach(lafrance)
coplot(WorkingMemory~Grade1L1Reading| L2PhonemicAwareness, panel=
function(x,y,...) panel.smooth(x,y,span=.8,...),data=lafrance)
```

As phonemic awareness increases, there is less spread in the scatterplots of working memory and G1L2 reading. When phonemic awareness is high, scores on Grade 1 L1 reading are high too, but scores on working memory are somewhat in the middle.

2. Answer:
Lafrance and Gottardo (2005) data. Use R Console:

```
coplot(KinderL2Reading~Grade1L2Reading | L2PhonemicAwareness,
panel=function(x,y, ...)panel.smooth(x,y,span=.8),data=lafrance)
detach(lafrance)
```

The relationship between word reading measures looks essentially flat (no correlation) according to the smooth lines for the first two plots, which represent lower levels of phonological awareness. With increasing phonological awareness, the relationship becomes somewhat more linear and scores on both reading tests increase.

3. Answer:
Larson-Hall (2008) data. Use R Console:

```
attach(larsonhall2008)
coplot(rlwscore~gjtscore|earlyage,
panel=function(x,y, ...)panel.smooth(x,y,span=.8),data=larsonhall2008)
detach(larsonhall2008)
```

At an earlier age (the bottom three scatterplots) there is a positive relationship between scores on the grammar and pronunciation tests, but starting somewhat after age 8 (the upper three scatterplots) the relationship between scores on the grammar and pronunciation tests evens out or perhaps becomes slightly negative. Note that there are a number of missing rows in this data set. This is because only some of the participants had any experience with English at an early age.

3D Plots

1. Answer:
Lafrance and Gottardo (2005) data. 3D plots can be accessed through R Commander. Make sure the "lafrance" data set is the active data set. Then choose GRAPHS > 3D GRAPH > 3D SCATTERPLOT. Pick "Grade1L2Reading" for response variable. Hold down the Ctrl button and mouse-click on "L1PhonemicAwareness" and "KinderL2Reading." Additionally tick "Smooth regression." Press OK. Once the RGL device is open, use the mouse to rotate the figure to look at it in 3D.

Looking at the regression plane, it seems to be affected by all three axes. There is definitely a positive relationship between L2 word reading measures in Kindergarten and first grade—when the scores on Kindergarten measures get bigger, they also get bigger for first grade. It is hard to say how phonological awareness affects scores on both word reading measures, however. There seem to be a lot of points concentrated on the (0,0) line which range in their level of phonological awareness, and in general there don't seem to be really low levels of phonological awareness. This makes it difficult to discern any trend with phonological awareness and the word reading, although looking at the graph from the angle captured below there is a slight angle upward to the plane away from the Phonological awareness axis, which seems to indicate a slight level of positive association.

Tree Models

1. Answer:
Lafrance and Gottardo (2005) data. First, make sure the tree library is open:

```
library(tree)
```

Create the tree model, setting "Grade1L2Reading" as the response variable:

```
model=tree(Grade1L2Reading ~.,lafrance)
plot(model) #call for the tree
text(model) #puts the names on the tree
```

The variables included in the tree are KinderL2Reading, Grade1L1Reading, L2PhonemicAwareness and L1PhonemicAwareness. Kindergarten L2 reading splits the first node, Grade 1 L1 reading splits the next node down on the left, while L1 Phonemic awareness splits the next node down on the right. Phonological awareness in L2 is important for those on the left branch, while KinderL2 reading and Grade 1 L1 reading are important for those on the right branch.

To exclude the variables that were not included in the first run, you can subset the data frame. I did it in R Commander (DATA > ACTIVE DATA SET > SUBSET ACTIVE DATA SET) and just selected the four variables (plus Grade 1 L2 reading, which we are trying to explain by the other variables) that appeared in the first tree mode. Here is the syntax:

```
lafrance.smaller <- subset(lafrance, select=c(Grade1L1Reading,
Grade1L2Reading,KinderL2Reading,L1PhomemicAwareness,
L2PhonemicAwareness))
```

Now I want to run the tree model again in R Console:

```
model2=tree(Grade1L2Reading ~.,lafrance.smaller)
plot(model2)
text(model2)
```

2. Answer:
Dewaele and Pavlenko data, beqSwear file. There are 21 variables (columns) in the data file. We want to model the variables as explaining "weight1" first, so here is our model:

```
model=tree(weight1~.,beqSwear)
```

```
plot(model)
text(model)
```

The variables that are included in the tree are the weight they give to swearing in their L2 (weight2), frequency that the person swears in L1 (swear1), evaluation of their speaking skills in L1 (l1speak), the context of acquisition where they acquired their second language (contextacquisitionsec), age, the frequency they speak in their L2 (l2freq), and the number of languages they speak. The tree shows that many factors are involved in the decision to swear!

Now let's try a model using weight2 as the response variable.

```
model2=tree(weight2~.,beqSwear)
plot(model2)
text(model2)
```

This model has more explanatory variables included than the one with "weight1." It has the frequency of swearing in L1 and L2 (swear1 and swear2), L2 comprehension ability (l2_comp), the weight the person gives swearing in L1 (weight1), L2 reading ability (l2_read), the frequency with which they use the L2 (l2freq), their current age, and the age they learned the second language (agesec).

Application Activities with Multiple Regression

1. Answer:
Lafrance and Gottardo (2005).

```
modelA=lm(Grade1L1Reading~NonVerbalReasoning,data=lafrance)
summary(modelA)
```

The total R^2 value is found in the second line up from the bottom of the summary in the line titled "Multiple R-squared." The change in R^2 will be a straightforward calculation from the first model (here, ModelA). The coefficients for the five variables will be found in the row with the title of the variable (here in ModelA that's "NonVerbal Reasoning" only and the column that says "Estimate." In this way, gather the data from each successive model and draw it into a table (or just check it against the table in the online document).

```
modelB=lm(Grade1L1Reading~NonVerbalReasoning+KinderL2Reading,data=
lafrance)
modelC=lm(Grade1L1Reading~NonVerbalReasoning+KinderL2Reading+
NamingSpeed,data= lafrance)
modelD=lm(Grade1L1Reading~NonVerbalReasoning+KinderL2Reading+
NamingSpeed+WorkingMemory,data= lafrance)
modelE=lm(Grade1L1Reading~NonVerbalReasoning+KinderL2Reading+
+ NamingSpeed+WorkingMemory+L2PhonemicAwareness,data= lafrance)
```

2. Answer:
Lafrance and Gottardo (2005) data. Basically we can just take the final model from activity 1 and replace the response variable with the variable "Grade1L2Reading":

```
model1=lm(Grade1L2Reading~NonVerbalReasoning+KinderL2Reading+
+ NamingSpeed+WorkingMemory+L2PhonemicAwareness,data= lafrance)
```

```
summary(model1)
```

In reporting on the results of a standard regression, we can find the total R^2 value in the second line up from the bottom of the summary in the line titled "Multiple R-squared." Here that is $R^2=.67$. The coefficients for each variable are found at the intersection of the row with the title of the variable and the column that says "Estimate." So, for example, for NamingSpeed the B coefficient you should report is −.52. In order to obtain the more informative estimate of relative importance, use the relaimpo library and the calc.relimp() command.

```
library(relaimpo)
calc.relimp(model1)
```

Use the values from the section entitled "Relative importance metrics" for these values. So, for example, for NamingSpeed the relative importance is 5% (not very important!).

To examine the assumptions of multiple regression, use the plots listed in the summary box at the end of the online document "Multiple Regression.Finding the best fit":

```
plot(model1)
plot(model1, which=4)
plot(model1,which=6)
vif(model1)
library(MASS)
plot(row.names(lafrance), stdres(model1))
```

There do not appear to be any large problems with the assumptions, although there may be some small departures (such as the Q-Q plot showing some departures from normality).

3. Answer:
French and O'Brien (2008). Construct a series of models in the order of entering variables:

```
modelA=lm(GRAM_2~GRAM_1,data=french)
summary(modelA)
modelB=lm(GRAM_2~GRAM_1+INTELLIG,data=french)
modelC=lm(GRAM_2~GRAM_1+INTELLIG+L2CONTA,data=french)
modelD=lm(GRAM_2~GRAM_1+INTELLIG+L2CONTA+ANWR_1,data=french)
modelE=lm(GRAM_2~GRAM_1+INTELLIG+L2CONTA+ANWR_1+ENWR_1,data=fr
ench)
```

Use the summary function to get the information for each successive model (see activity 1 for more detailed information about what information you need to gather from each summary.

In order to obtain the more informative estimate of relative importance, use the relaimpo library and the calc.relimp() command.

```
library(relaimpo)
calc.relimp(modelE)
```

Application Activities for Finding the Best (Minimally Adequate) Fit

Note: For all of these questions there is not one single possible answer or route to finding the minimal adequate model. I give my own attempts here, but variation away from what I did should not necessarily be considered wrong! There should be a logic, however, to the testing that is done to find the minimally adequate fit.

1. Answer:
Howell (2002). The response variable is "Overall," and the explanatory variables are "Teach," "Exam," "Knowledge," "Grade," and "Enroll." First look at scatterplots. In R Commander choose GRAPHS > SCATTERPLOT MATRIX and then choose all of the variables.

The R code for this is:

```
scatterplot.matrix(~Enroll+Exam+Grade+Knowledge+Overall+Teach,
reg.line=lm, smooth=TRUE, span=0.5, diagonal = 'density',
data=howell)
```

The intersection of "Overall" with the other variables looks fairly linear in all cases except for "Enroll," which is highly positively skewed, as seen in the density plot for "Enroll" found on the diagonal. Exclude "Enroll" from the regression.

To start the regression analysis, we note there are 50 data points (length(howell$Overall), which means we should have at most only about 50/3≅18 parameters. Thus, this model is overparameterized. Thus, a different person might proceed differently from here, but here is one way to do it.

If we model a full factorial with four parameters (excluding the "Enroll" variable), there will be 24 parameters, so we will need to split the modeling into at least two parts. First, we'll name all of the two-way and three-way interactions, and then randomize them. There are 6 two-way and 4 three-way interactions, so we'll put half of them in each model. We'll then add the four-way interaction.

```
interactions2=c("Teach:Exam", "Teach:Knowledge", "Teach:Grade",
+ "Exam:Grade", "Knowledge:Grade")
sample(interactions2) #this randomly rearranges the members of the set
[1] "Teach:Exam" "Teach:Knowledge" "Teach:Grade"
[4] "Knowledge:Grade" "Exam:Grade"
interactions3=c("Teach:Exam:Knowledge", "Teach:Exam:Grade",
"Teach:Knowledge:Grade", "Exam:Knowledge:Grade")
sample(interactions3)
[1] "Teach:Exam:Grade" "Exam:Knowledge:Grade"
[3] "Teach:Exam:Knowledge" "Teach:Knowledge:Grade"

attach(howell)

hmodel1=lm(Overall~Teach+Exam+Knowledge+Grade+
Knowledge:Grade +Teach:Knowledge+ Teach:Grade + #2-way interactions
Teach:Exam:Grade +Exam:Knowledge:Grade+ #3-way interactions
Teach:Exam:Knowledge:Grade)
summary(hmodel1)
```

```
hmodel2=lm(Overall~Teach+Exam+Knowledge+Grade+
Teach:Exam + Exam:Grade + #2-way interactions
Teach:Exam:Knowledge + Teach:Knowledge:Grade + #3-way interactions
Teach:Exam:Knowledge:Grade)
summary(hmodel2)
```

In model1 and model2 there are no statistical terms. Let’s make another model with all of the three-way terms, the one four-way term, and all of the main effects:

```
hmodel3= lm(Overall~Teach+Exam+Knowledge+Grade+
Teach:Exam:Grade +Exam:Knowledge:Grade+ Teach:Exam:Knowledge +
Teach:Knowledge:Grade +
Teach:Exam:Knowledge:Grade)
summary(hmodel3)
```

In model3, still nothing is statistical. Let’s try another model with only two-way interactions and main effects:

```
hmodel4= lm(Overall~Teach+Exam+Knowledge+Grade+
Knowledge:Grade +Teach:Knowledge+ Teach:Grade + Teach:Exam + Exam:Grade)
summary(hmodel4)
```

The summary shows that still nothing is statistical. Let’s go down to a model with only main effects:

```
hmodel5= lm(Overall~Teach+Exam+Knowledge+Grade)
summary(hmodel5)
```

Now the main effects for Teach and Knowledge are statistical. Take out the main effect with the highest *p*-value, which is Exam:

```
hmodel6= update(hmodel5,~.-Exam)
anova(hmodel5, hmodel6)
summary(hmodel6)
```

The Grade variable is still not statistical, so take it out.

```
hmodel7= update(hmodel6,~.-Grade)
anova(hmodel6, hmodel7)
summary(hmodel7)
```

Now everything is statistical. The result is a model with only the terms Teach and Knowledge, and the regression equation is: Overall rating=−1.3+.7*Teaching ability+.53*Knowledge of subject matter. The explanatory value of this model is R^2=.74, with a residual standard error of .32 on 47 df.

We can look at the relative importance of the terms using the relaimpo library:

```
library(relaimpo)
calc.relimp(hmodel7)
```

This shows that Teach accounts for 46% and Knowledge for 28% of the variance.

Looking at diagnostic plots for this model:

```
plot(hmodel7)
plot(hmodel7, which=4)
plot(hmodel7,which=6)
vif(hmodel7)
library(MASS)
plot(row.names(howell), stdres(hmodel7))
```

Some outliers are identified, but otherwise variances look constant and data is fairly normally distributed.

2. Answer:
Dewaele and Pavlenko data. First, examine a scatterplot matrix with the variables "swear2", "l2freq", "weight2", "l2speak", and "l2_comp" (in R Commander, GRAPHS > SCATTERPLOT MATRIX). The regression and Loess lines drawn on the scatterplots indicate fairly linear patterns, although the data points look randomly scattered all over the graph at discrete points. We will continue with the analysis.

To do the regression, create a first full factorial model:

```
beqm1=lm(swear2~l2freq*weight2*l2speak*l2_comp,data=beqSwear)
summary(beqm1)
```

Only one term, the main effect of "weight2," is statistical in this full model. Let's give the boot.stepAIC() a try and see what it computes.

```
library(bootStepAIC)
boot.stepAIC(beqm1,data=beqSwear)
```

This procedure gives the following final model:

```
swear2 ~ l2freq + weight2 + l2speak + l2_comp + l2freq:l2speak +
weight2:l2speak + l2speak:l2_comp
```

To make sure this is really the minimal adequate model, we'll model it and take a look at the summary.

```
beqm2=lm(swear2 ~ l2freq + weight2 + l2speak + l2_comp + l2freq:l2speak +
weight2:l2speak + l2speak:l2_comp, data=beqSwear)
summary(beqm2)
```

This model does not contain *only* statistical terms, but, because all three of the interactions which are included are statistical, the main effects contained in those interactions must be retained. Use the relaimpo library to check the relative importance of the terms:

```
library(relaimpo)
```

```
calc.relimp(beqm2)
```

This shows that, although the interactions may be statistical, they do not provide an explanation for even 1% of variance in the data. The large effect comes from the frequency with which the speakers speak the L2 (11% for "l2freq"). If we keep this model, the regression equation is:

Amount of L2 swearing=.78 +.08*Frequency you speak L2+.56*Weight you give to swearing in L2−.10*Your perceived speaking ability in L2−.16*Your perceived listening comprehension in L2+.05*l2sfreq:l2speak−.08*weight2:l2speak+.08*l2speak:l2_comp.

This model accounts for R^2=.31 of the variance in the response variable, with a residual standard error of 1.04 on 872 df.

What would happen if we made a model that took out all of the interaction terms, since we know they do not have much relative importance? I will try to model that:

```
beqm3=lm(swear2~l2freq+weight2+l2speak+l2_comp, data=beqSwear)
summary(beqm3)
anova(beqm2, beqm3)
```

The summary shows that all main effects except "l2_comp" are statistical. The ANOVA shows that this model is statistically different from the one with the interaction effects, and it explains fewer of the variances in the data (R^2 is smaller at 29%, but this is to be expected when there are fewer variables and the difference is not so large). A simplification to only main effects might be a good idea, as it makes the model conceptually easier to understand. However, this is up to the judgment of the reader!

Last of all, we will examine how well the minimal adequate model (beqm2) meets regression assumptions.

```
plot(beqm3)
plot(beqm3, which=4)
plot(beqm3,which=6)
vif(beqm3)
library(MASS)
plot(row.names(beqSwear), stdres(beqm3))
```

The pattern of residuals is very strange and discrete (it is declining in the fitted values), but it doesn't remind us of a pie piece or a pinching of data in a certain region, so it doesn't seem to indicate heteroscedasticity. The Q-Q plot looks fairly linear and normal in its middle portions, with some outliers identified at the ends. The leverage plot indicates a couple of outliers that may be influential. Overall, the data seem to meet the regression assumptions well, with the exception of a few outliers. However, because there are outliers a robust regression analysis is recommended.

3. Answer:
Larson-Hall (2008) data. To look first at a scatterplot matrix, I used R Commander: GRAPHS > SCATTERPLOT MATRIX; choose "gjtscore," "rlwscore," "aptscore," and "totalhrs." In looking at the relationship of aptitude and the phonemic test ("rlwscore") with GJT, both

scatterplots seem to show curvature (the Loess line shows a distinct curve). The hours looks more linear. I will add quadratic terms for both “aptscore” and “rlwscore” to the model.

Starting with (gjtscore) as response variable, construct a model:

```
lhmodel1=lm(gjtscore~totalhrs*aptscore*rlwscore+
I(rlwscore^2)+I(aptscore^2),data=larsonhall2008)
summary(lhmodel1) #shows both quadratic terms are statistical
```

For a next step in the model, remove the largest interaction (the three-way interaction) and compare models:

```
lhmodel2=update(lhmodel1,~.-totalhrs:aptscore:rlwscore)
anova(lhmodel1,lhmodel2) #no difference, keep simpler model
summary(lhmodel2)
```

Now more terms are nearly statistical, but the two-way interaction between “totalhrs” and “aptscore” is not and has the highest *p*-value, so remove it next.

```
lhmodel3=update(lhmodel2,~.-totalhrs:aptscore)
anova(lhmodel2,lhmodel3) #no difference, keep simpler model
summary(lhmodel3)
```

There are still some non-statistical terms, including one two-way interaction, so remove that.

```
lhmodel4=update(lhmodel3,~.-aptscore:rlwscore)
anova(lhmodel3,lhmodel4) #no difference in models
summary(lhmodel4)
```

The *p*-value for the main effect of the “rlwscore” is not under .05, but the main effect must stay in the model if the quadratic term is there, so we will not take out any further terms.

The regression equation is: .02+.01*TotalHours−3.9*Aptitude score−6*RLWScore+.06(RLWScore)^2+.01*(Aptitudescore)^2−.0001*(the interaction between hours and RLW Score).

This model explains R^2=24% of the variance in scores on the GJT. One question, though, is how do we interpret a regression equation with a quadratic term? Crawley (2007) notes that, if a polynomial regression (one that contains a higher power of x) fits the data better than a non-polynomial does, this means the data depart significantly from linearity.

Let’s look at the relative importance of the terms:

```
library(relaimpo)
calc.relimp(lhmodel4)
```

The most important term is the quadratic RLWScore at 8%. No one term provides a large amount of explanation though.

What would happen if we used `boot.stepAIC`?

```
boot.stepAIC(lhmodel1,data=larsonhall2008)
```

The only difference is that the automatic process would subtract out the interaction between hours and RLW Score and keep in the interaction between aptitude score and RLW Score. This is also a plausible model!

Let's look at the diagnostics plots.

```
plot(lhmodel4)
plot(lhmodel4, which=4)
plot(lhmodel4,which=6)
vif(lhmodel4)
library(MASS)
plot(row.names(larsonhall2008), stdres(lhmodel4))
```

Residuals vs. fitted: This looks like a pie piece with a lot of pinching on the right end, indicating a problem with heteroscedasticity.
Q-Q: The main part of the data fits the line quite well; the ends are a little bit off the line.
Leverage: Three outliers are labeled that are away from the bulk of the data.
Cook's distance: There is nothing to worry about here.

A look at VIF shows that scores are too high, with all well over 5! Of course, since one variable is the square of another, high intercorrelation is to be expected in some places here, but high VIFs for all variables are problematic.

In sum, the regression has some serious problems with constant variances (heteroscedasticity) and some outliers. A robust regression is recommended.

Application Activities with Robust Regression

1. Answer:
Lafrance and Gottardo (2005) data. You are asked to use the model you determined in the "Multiple Regression.Application activity_Multiple regression" section for this data. That model was:

```
model1=lm(Grade1L2Reading~NonVerbalReasoning+KinderL2Reading+
+ NamingSpeed+WorkingMemory+L2PhonemicAwareness,data= lafrance)
```

So now I construct a model to compare both LS and robust fits:

```
library(robust)
l2modelfit=fit.models(
list(Robust="lmRob", LS="lm"),
formula= Grade1L2Reading~NonVerbalReasoning+KinderL2Reading+
+ NamingSpeed+WorkingMemory+L2PhonemicAwareness,
data= lafrance)
summary(l2modelfit)
```

The parameters are only slightly different in most cases, but highly different for L2 Phonemic Awareness (the robust parameter is 0.16, while the least squares parameter is −0.25). The R^2 is smaller for the robust fit, but residual error is also smaller.

Look at diagnostic plots:

```
plot.lmRob(l2modelfit) #I got an error message here about "incorrect number of
dimensions"
```

So I will try just using the `plot()` command on the robust fit model.

```
plot(l2modelfit)
```

The Q-Q plot looked pretty crooked for both fits! The density plot was more centered on zero for the robust fit. I would choose to report the robust regression data, since there was a slight difference.

2. Answer:
Larson-Hall 2008 data. You are asked to use the model you determined in the “Multiple Regression.Application activity_Finding the best fit” section for this data. My model was:

```
lm(formula = gjtscore ~ totalhrs + aptscore + rlwscore + I(rlwscore^2) + I(aptscore^2)
+ totalhrs:rlwscore, data = larsonhall2008)
```

Again, construct a model to compare both fits:

```
lhfit=fit.models(
list(Robust="lmRob", LS="lm"),
formula= gjtscore ~ totalhrs + aptscore + rlwscore + I(rlwscore^2) + I(aptscore^2) +
totalhrs:rlwscore, data=larsonhall2008)
summary(lhfit)
```

The parameters, R^2, and residual errors of both fits are almost exactly the same.

Looking at diagnostics:

```
plot(lhfit)
```

I chose number 1 for all plots. The overlaid plots look exactly the same, so in this case there is probably not a lot of reason to choose the robust model.

3. Answer:
Howell (2002) data. You are asked to use the model you determined in the “Multiple Regression.Application activity_Finding the best fit” section for this data. My model was:

```
Overall ~ Teach + Knowledge + Grade
```

```
howellfit=fit.models(
list(Robust="lmRob", LS="lm"),
formula= Overall ~ Teach + Knowledge + Grade,
data=howell)
summary(howellfit)
```

There are some large differences in parameter estimates; the R^2 is quite a bit smaller for the robust fit, but is still rather large at 62%. Look at diagnostic plots:

```
plot(howellfit)
```

The Q-Q plots fit the data quite well for both models. The density plot shows the robust model with a slightly more compact middle than the LS fit.

4. Answer:
Dewaele and Pavlenko data. You are asked to use the model you determined in the "Multiple Regression.Application activity_Finding the best fit" section for this data. My model was:

```
beqm3=lm(swear2~l2freq+weight2+l2speak+l2_comp, data=beqSwear)

swearfit=fit.models(
list(Robust="lmRob", LS="lm"),
formula= swear2~l2freq+weight2+l2speak+l2_comp,
data=beqSwear)
summary(swearfit)
```

There are some small differences between the LS and robust fit parameter estimates. The robust analysis has a slightly smaller residual error and a higher R^2 than the L2 fit. I would report using the robust estimates.

Look at diagnostics:

```
plot(swearfit)
```

Chapter 8: Chi-Square

Application Activities for Summarizing and Visualizing Data

Application Activities with Crosstabs

1. Answer:
Dewaele and Pavlenko (2001–2003) data. In R Commander, choose STATISTICS > CONTINGENCY TABLES > MULTI-WAY TABLE. I chose "NumberOfLang" for row, "CatDominance" for column, and "sex" for control variable. Do not click the radio button for percentages. Press OK.

The R code for this procedure is:

```
.Table <- xtabs(~NumberOfLang+CatDominance+sex, data=beqDom)
.Table
```

In looking at males and females separately, the pattern of increasing numbers responding that dominance is among more than one language as number of languages known increases holds approximately true for both males and females, although it seems stronger in females (for females with 5 lges, YES=99 and YESPLUS=112, while for males with 5 lges, YES=58 and YESPLUS=51).

2. Answer:
Mackey and Silver (2005). In R Commander, make sure the mackey data set is the active one. Then choose STATISTICS > SUMMARIES > FREQUENCY DISTRIBUTIONS. Choose the PRETEST variable. If it is not available, you'll need to make it into a factor first:

```
mackey$PreTest=factor(mackey$PreTest)
```

If using R Commander, you'll need to change active data sets and then come back to mackey. Now you will see PreTest as a choice.

The command for R is:

```
.Table <- table(mackey$PreTest)
.Table
```

You should have N=26 in total, and the most frequent level is 2, while the least frequent is 4.

3. Answer:
Mackey and Silver (2005). In R Commander, choose STATISTICS > CONTINGENCY TABLES > TWO-WAY TABLE. I chose "Group" for row and "PreTest" for column. Press OK.

The command for R is:

```
.Table <- xtabs(~Group+PreTest, data=mackey)
.Table
```

You should have N=26. From the numbers it looks as though there were more participants in the experimental group who were at lower developmental levels.

Application Activities with Barplots

1. Answer:
Mackey and Silver (2005) data. Use R Console for this:

```
attach(mackey)
barplot(tapply(DevelopDelPost, list(Group, DevelopDelPost), length),col=c(2,4),
beside=T)
locator() #use this to click where you want the legend to be; then pull down the Stop
menu and #stop the locator
legend(3.86,9.82,c("Control group", "Experimental group"),fill=c(2,4))
```

Probably you will have different coordinates for your legend! More students from the experimental group developed than did not develop (while the opposite was true for students from the control group).

2. Answer:
Mackey and Silver (2005) data. You could just use the code in the previous item (activity 1), replacing DevelopDelPost with DevelopPost in both places in the code. If you want to

envision this the opposite way, however (with groupings done by group and bars showing how many developed and how many did not), this is easily done by simply switching the order of variables inside the command:

```
barplot(tapply(DevelopPost, list(DevelopPost,Group), length),col=c(2,4), beside=T)
legend(1.55,8.5,c("Developed", "Did not develop"),fill=c(2,4))
detach(mackey)
```

The legend labels and location need to be changed too.

The graph shows that, in the immediate post-test, the pattern seen in activity 1 did *not* hold. For the immediate post-test, both groups had more students who developed than did not develop, although the number who developed is less for the control group than for the experimental group (and the number who didn't develop is roughly equal in both experimental conditions).

3. Answer:

```
attach(languageChoice)
barplot(tapply(Population, list(Population,Language), length),col=c(2,4), beside=T)
legend(.93,28,c("Hometown U", "Big City U"),fill=c(2,4))
detach(languageChoice)
```

4. Answer:

```
attach(motivation)
barplot(tapply(teacher, list(teacher, first), length),col=c(1,2,3,4,5), beside=T,
main="Beginning of term: Do you like this class?")
legend(6.8, 76.8,c("Teacher 1", "Teacher 2", "Teacher 3", "Teacher 4", "Teacher
5"),fill= c(1,2,3,4,5))

barplot(tapply(teacher, list(teacher, last), length),col=c(1,2,3,4,5), beside=T,
main="End of term: Do you like this class?")
legend(6.8, 76.8,c("Teacher 1", "Teacher 2", "Teacher 3", "Teacher 4", "Teacher
5"),fill= c(1,2,3,4,5))
detach(motivation)
```

Application Activities with Association Plots, Mosaic Plots, and Doubledecker Plots

1. Answer:
Dewaele and Pavlenko (2001–2003) data. First open library; then do command in R Console:

```
library(vcd)
(EDU=structable(L1dominance~degree + numberoflanguages,data=beqSwear))
mosaic(EDU,gp=shading_Friendly,labeling_args=list(rep=c(degree=F)))
```

The plot is fairly complicated, but in just looking at those areas where the residuals are higher than 4.00 (there is just one such area) it appears there are more Ph.Ds with L1PLUS dominance who have five languages than would be expected. In other words, we might say Ph.Ds are more represented in the intersection of five languages and having L1PLUS dominance than would be expected if all educational levels were equally likely.

2. Answer:
Try both ways of arranging the data:

```
doubledecker(L1dominance~degree+numberoflanguages,data=beqSwear)
doubledecker(L1dominance~numberoflanguages+ degree,data=beqSwear)
```

The doubledecker plot does not seem as informative to me as the mosaic plot for this data.

3. Answer:
If not already open:

```
library(vcd)
```

Then construct the association plot:

```
(QUEST=structable(DevelopDelPost~DevelopPost+Group,data=mackey))
assoc(QUEST,gp=shading_Friendly)
```

There are slightly more people who developed in the post-test in the experimental group than would be expected if there were no effect for group (the shading is blue but the residuals do not go above 4). This data is quite simple, so I'm not sure I prefer the association plot to the barplot! Let's make a mosaic plot now:

```
mosaic(QUEST,gp=shading_Friendly)
```

What do you think?

4. Answer:
There are several possibilities for structuring this data. Here is mine:

```
(TEACH=structable(first~last+teacher,data=motivation)) #one way to structure it!
mosaic(TEACH, gp=shading_Friendly)
assoc(TEACH, gp=shading_Friendly)
doubledecker(last~first+teacher, data=motivation)
```

I like the doubledecker one best—it seems easiest to figure out.

```
doubledecker(last~first+teacher, data=motivation)
doubledecker(last~first+teacher, data=motivation)
```

Application Activities for Calculating One-Way Goodness-of-Fit and Two-Way Group-Independence Tests

1. Answer:
Geeslin and Guijarro-Fuentes (2006) data. Test the data using R Console:

```
chisq.test(table(geeslin3$Item15))
```

Results show that we should reject the null hypothesis that all choices are equally likely (χ^2=6.421, df=2, *p*=.04).

```
barplot(table(geeslin3$Item15))#one choice for a visual
```

The barplot shows that number 1 (*estar*) was the most frequent choice, number 2 (*ser*) was the second most frequent, and number 3 was the least frequent.

2. Answer:
Geeslin and Guijarro-Fuentes (2006) data.

```
p=c (.45,.45,.1) #set up probabilities as described in problem
chisq.test(table(geeslin3$Item15),p=p)
```

Results show that we cannot now reject the null hypothesis (χ^2=1.468, df=2, *p*=.48).

3. Answer:
Mackey and Silver (2005) data.

```
summary(assocstats(xtabs(~Group+DevelopPost, data=mackey)))
```

There are 26 valid cases. Results show that we cannot reject the null hypothesis (Pearson χ^2=.097, df=1, *p*=.756). We know there is at least one cell with less than expected counts because of the warning. The effect size is phi=.06, quite a small effect size.

4. Answer:
Dewaele and Pavlenko (2001–2003) data.

```
summary(assocstats(xtabs(~NumberOfLang+CatDominance, data=beqDom)))
```

There should be 1,036 valid cases, and results show that we can reject the null hypothesis that there is no relationship between the variables (Pearson=59.58, df=6, *p*<.0001). No cells have less than the expected count. The effect size is small (Cramer's V=.17).

5. Answer:
Smith (2004). Because we are given only summary data, the easiest way to do a group independent chi-square for this data is to use R Commander: STATISTICS > CONTINGENCY TABLES > ENTER AND ANALYZE TWO-WAY TABLE. I entered the data this way:

Strategy	*No Uptake*	*Successful Uptake*
Preemptive	21	2
Negotiation	37	6

Results: Pearson χ^2=.39, df=1, *p*=.53. There is no relationship between strategy and uptake.

Chapter 9: T-Tests

Application Activities with Creating Boxplots

1. Answer:
Leow and Morgan-Short (2004) data. To make new variables, in R Commander choose DATA > MANAGE VARIABLES IN ACTIVE DATA SET > COMPUTE NEW VARIABLE from the drop-down menu. For the Receptive condition, I will call the new variable recfinal and put this name in

the “New variable name” box. Move the “RecPostScore” to the “Expression to compute” box by double-clicking it. Next insert a minus sign from the keyboard; then insert the “RecPreScore” from the “Current variables” box by double-clicking. Press OK. If you click the “View data set” button you will see a new column called “recfinal.” Do the same for the Productive condition (ProPostScore- ProPreScore); I will call the new variable profinal.

The R code for these actions is:

```
leow$recfinal <- with(leow, RecPostScore- RecPreScore)
leow$profinal <- with(leow, ProPostScore- ProPreScore)
```

To plot two variables side by side on a boxplot, we’ll need to use the R Console (R Commander will plot only one variable at a time). We’ll first try:

```
boxplot(leow$recfinal, leow$profinal)
```

This works out but doesn’t look that pretty. I’ve prettified it this way:

```
boxplot(leow$recfinal, leow$profinal,names=c("Receptive",
"Productive"),boxwex=.5,col="darkgrey",medcol="white")
#the names argument puts names underneath the variables
```

The productive condition shows seven points labeled as outliers. The receptive condition does not have any outliers. The distributions are highly different; the distribution of the receptive condition ranges over a large space, while the distribution of the productive condition is much narrower. The receptive condition’s median is higher, at about 3, while the productive condition’s median is around zero. Both distributions are slightly skewed positively, meaning that there is more distribution above the median line than below it.

2. Answer:
Leow and Morgan-Short (2004) data. To plot one variable divided into groups, in R Commander choose GRAPHS > BOXPLOT. Pick the new receptive variable you made; I called mine “recfinal.” Push the “Plot by Groups” button and choose GROUP. Click the “identify outliers” box. Press OK. The R code for this action is:

```
boxplot(recfinal~Group, ylab="recfinal", xlab="Group",
data=leow)
```

The boxplot shows that there are three outliers in the non-think-aloud group but none for the think-aloud group. The median of both groups is equal, at about 3, but the boxplot of the think-aloud group extends further upward than does the boxplot of the non-think-aloud group, which means that more people did better on the measure in the think-aloud group. The think-aloud group has more spread; its IQR box and whiskers cover more space than that of the non-think-aloud group. Having more spread is not necessarily desirable; it means there is more variance and more individual variation in the measure. It also means the groups do not have equal variances, if we want to compare them. However, in this case there were also more people who did better in the think-aloud group, so it looks as though more people made progress in the think-aloud group than in the non-think-aloud group.

3. Answer:

Yates (2003) data. To plot a series of boxplots of different variables, use the R Console. A first try might be:

```
attach(yate)
boxplot(LabBefore, LabAfter, MimicryBefore, MimicryAfter)
```

Again, it would be nice to have labels and prettier boxplots perhaps!

```
boxplot(LabBefore, LabAfter, MimicryBefore, MimicryAfter, names=c("Lab Before",
"Lab After", "Mimicry Before", "Mimicry After"), boxwex=.5, col="darkgrey",
medcol="white")
```

I leave it to the reader to analyze what the boxplots say.

4. Answer:
Inagaki and Long (1999) data. To make a boxplot divided by groups, R Commander works well (GRAPHS > BOXPLOT). The R code is:

```
boxplot(GainScore~Group, ylab="GainScore", xlab="Group",
data=inagaki)
```

Again, I leave it to the reader to analyze what the boxplots say.

Application Activities for the Independent-Samples T-Test

1.Answer:
Larson-Hall (2008) data. First, to explore whether the data are normally distributed and have equal variances, let's look at boxplots of the data. The multiple boxplot is too cluttered to be a good diagnostic, so I suggest looking at each variable ("Total score on aptitude test [aptscore]," "Total score of 4 situations for use of English in life now [useeng]," "gjtscore," "Score on the R/L/W test [rlwscore]") separately. In R Commander, from the drop-down menu, choose GRAPHS > BOXPLOTS. Pick one variable at a time, push the "Plot by groups" button, and pick erlyexp.

Results: Only the GJT score appears to satisfy parametric assumptions.

For Aptitude test: Variances look equal but distribution is negatively skewed and there are outliers. For English use: Again, variances look equal but distribution is positively skewed and there are many outliers. For GJT score: These boxplots look normally distributed and variances are very similar. For R/L/W test: Boxplots look symmetric and variances are similar, but there are some outliers. For any variables with outliers it would be better to use robust methods to remove them in an objective fashion.

To perform an independent-sample t-test with the variables, in R Commander choose STATISTICS > MEANS > INDEPENDENT SAMPLES T-TEST. Choose "erlyexp" for the "Groups" box on the left and "gjtscore" for the "Response Variable" on the right. Do not change any other settings. Press OK. Repeat with the other variables. To get all of the information for the table below, also run the NumSummary test using the "erlyexp" group to split scores (under STATISTICS > SUMMARIES > NUMERICAL SUMMARIES). Actually, to do the multiple values, I took the code from R Commander's script window, put it into R Console, and then just changed the name of the variable each time for the two commands (so, anywhere you find "gjtscore" in these commands, put in the other variables):

```
t.test(gjtscore~erlyexp, alternative='two.sided',
conf.level=.95, var.equal=FALSE, data=larsonhall2008)

numSummary(larsonhall2008[,"gjtscore"],
groups=larsonhall2008$erlyexp, statistics=c("mean", "sd",
"quantiles"), quantiles=c(0,.25,.5,.75,1))
```

Results: Parametric T-Tests

Variable	*95% CI*	*Mean 1 (SD1) Not Early*	*Mean 2 (SD2) Early*	*N1/N2*	T-*Value*	P-*Value*	*Effect Size*
Aptitude Score	−1.58, .99	31.2 (4.4)	31.5 (4.2)	139/61	−.45	.65	.02
Use of English	−.95, .76	7.1 (2.8)	7.1 (2.7)	131/56	−.21	.83	0
GJT Score	−5.63, .76	112.3 (10.1)	114.7 (10.7)	139/61	−1.51	.13	.02
R/L/W Score	−9.74, −1.06	49.3 (13.2)	54.7 (14.7)	139/61	−2.47	.02	.03

Only the RLWScore's CI does not pass through zero (in other words, the comparison between early and later groups for the RLWScore is statistical). However, the effect size for all comparisons is quite small.

2. Answer:
Inagaki and Long (1999) data. We have already looked at boxplots of this data in item 4 of the online document "T-tests: Application activity_Creating boxplots." The data were highly positively skewed and not normally distributed at all. Variances are not too different, though. I used R Commander to run the t-test (STATISTICS > MEANS > INDEPENDENT SAMPLES T-TEST). I left the "Assume equal variances" setting on "No."

The result is a non-statistical difference between groups who heard recasts and those who heard models, $t(13.9)=-.23$, $p=.81$. I used the sequence STATISTICS > SUMMARIES > NUMERICAL SUMMARIES to get the means and standard deviations for both groups in order to calculate the effect size (done online). The effect size is $d=.11$, which is very small. The R code is:

```
t.test(GainScore~Group, alternative='two.sided',
conf.level=.95, var.equal=FALSE, data=inagaki)
numSummary(inagaki[,"GainScore"], groups=inagaki$Group,
statistics=c("mean", "sd", "quantiles"), quantiles=c(0,
.25,.5,.75,1))
```

3. Answer:
Larson-Hall and Connell (2005) data (LarsonHall.Forgotten.sav). To subset data, first check names in the file:

```
names(forget)
```

We want the Status variable, and we want to know the names of the levels *exactly*, so we can use the subset command correctly.

```
levels(forget$Status)
```

Now we're ready to go to R Commander and use the subset command, under DATA > ACTIVE DATA SET > SUBSET DATA SET. Leave the box ticked for "Include all variables," and write the subset condition to exclude just one of the three groups (the sequence "!=" means "not equal to"). I named each group by which group was *not* in that group (this may seem rather perverse, I know, but I did it because I do not know how to choose just two out of the three groups with the subset command), and so, for example, the group without the "Non" had the subset expression "Status!="Non" and I named it "forgetNoNon." Here is the R code for what I did:

```
forgetNoNon <- subset(forget, subset=Status!= "Non") #has 29 rows
forgetNoEarly <- subset(forget, subset=Status!= "Early") #has 30 rows
forgetNoLate <- subset(forget, subset=Status!= "Late") #has 29 rows
```

Now I am ready to run t-tests. Using R Commander, choose STATISTICS > MEANS > INDEPENDENT SAMPLES T-TEST. I want to choose "Status" for the Groups and "Sentence Accent" for the Response variable.

For some reason, when I used the forgetNoNon group, the Status variable wasn't appearing as a choice for a group variable (factor), in spite of the fact that it was a "character" variable (only "sex" appeared in the "Groups" box). I could make this into a factor manually, but what I did was just run a t-test with "sex" and then substituted in the variable "Status" in the R code. This worked fine in R Console:

```
t.test(SentenceAccent~Status, alternative='two.sided', conf.level=.95,
var.equal=FALSE, data=forgetNoNon) #CI [-2.3, -.7]
t.test(SentenceAccent~Status, alternative='two.sided', conf.level=.95,
var.equal=FALSE, data=forgetNoEarly) #CI [-1.3, .13]
t.test(SentenceAccent~Status, alternative='two.sided', conf.level=.95,
var.equal=FALSE, data=forgetNoLate) #CI [-2.9,-1.3]
```

The confidence intervals I give here mean that the Early and Late groups differed statistically (in the "forgetNoNon" group, which contains the Early and Late groups, the CI does not pass through 0 so the comparison is statistical), the Early and Non groups differed statistically, but the Late and Non groups did not differ statistically.

For effect sizes, I need means and standard deviations.

```
numSummary(forget[,"SentenceAccent"], groups=forget$Status,
statistics=c("mean", "sd"))
```

Standard deviations are not very different; I used an online calculator to calculate effect sizes. I leave it to the reader to calculate these and discuss how large they are (see the SPSS book, *A*

Guide to Doing Statistics in Second Language Research Using SPSS, Table 4.7, p. 118 for guidelines on Cohen's *d* effect sizes).

4. Answer:
Larson-Hall (2008) data. To perform robust t-tests, we will first have to subset the groups for each variable. In R Commander go to DATA > ACTIVE DATA SET > SUBSET ACTIVE DATA SET; then pick the four variables we are interested in (aptscore, gjtscore, rlwscore, and useeng; click *off* the "include all variables" box). In "subset expression" write "erlyexp=="Not Early," and name this the lh.not data set. Go back and pick the larsonhall data set (don't leave it set to lh.not), and do this again, this time using "erlyexp=="Early" and calling it lh.early). Here is the R code for the first one:

```
lh.not <- subset(larsonhall2008,
subset=erlyexp=="Not Early", select=c(aptscore,gjtscore,
rlwscore,useeng))
```

Now we can use the trimpb2 command, after we've opened up the WRS library:

```
trimpb2(lh.not$aptscore, lh.early$aptscore, tr=.2)
```

In other words, this command performs a robust t-test on the aptitude variable between the participants who received early English training and those who did not. The test returns a *p*-value (here *p*=.916), a confidence interval (here [−1.49, 1.62]), and the estimated difference in means (here .095). Continue to do this for the other variables (I've collected the answers into the table below).

To get trimmed means:

```
mean(lh.not$aptscore, tr=.2) #for example!
```

Here is the result of all the tests:

Variable	*95% CI*	*Mean 1 Not Early*	*Mean 2 Early*	*N1/N2*	P-*Value*
Aptitude Score	−1.49, 1.62	31.9	31.8	139/61	.92
Use of English	−1.15, .62	6.68	7.08	131/56	.68
GJT Score	−6.16, 1.14	111.8	114.4	139/61	.16
R/L/W Score	−10.75, −1.28	48.7	54.8	139/61	.01

Comparing parametric and robust versions, there is not too much difference, and only the R/L/W test shows a difference in performance between groups, as in the parametric version.

5. Answer:
First, you will need to change from the wide form to the long form of the data. You can easily do this in R Commander by choosing DATA > ACTIVE DATA SET > STACK VARIABLES IN ACTIVE DATA SET. Then rename your variable (I named mind "vocabstack"), choose a name

for the variable (this is the scores on the test, so I named it "score"), and choose a name for the factor (this is the group, so I named it "group"). Here is the R code for this same action:

```
vocabstack2 <- stack(vocabulary[, c("control", "experiment")])
names(vocabstack2) <- c("score", "group")
```

To test the assumptions for parametric tests, check a boxplot of this data (GRAPHS > BOXPLOT). There is one outlier in the experimental group, but variances seem fairly equal. Now perform a parametric t-test on this new variable, using STATISTICS > MEANS > INDEPENDENT SAMPLES T-TEST. The *p*-value is just over .05, meaning the groups are not statistically different. The mean scores of the groups are quite different, but the outlier in the experimental group (the score of 25) increases the variance, which increases the denominator of the t-test equation, which means that the *t*-statistic does not get as large as we would think it should, which results in a *p*-value over .05.

For the robust test, we don't need to subset the data; we can go back to the individual columns in the original "vocabulary" file:

```
trimpb2(vocabulary$control, vocabulary$experiment) #open the WRS library if not
open
```

Using the default 20% means trimming, the test is still not statistical. However, we seem to only have one outlier (that score of 25) so let's try 10% means trimming:

```
trimpb2(vocabulary$control, vocabulary$experiment, tr=.1)
```

The result is now statistical. If this feels like statistical hand-waving and data trolling until we get the answer we want, consider the following facts:

a. The effect size of the difference between groups has not changed. We can calculate effect size by getting the mean and standard deviations of both groups:

```
numSummary(vocabstack[,"score"], groups=vocabstack$group, statistics=c("mean",
"sd"))
```

Using these values and the online effect size calculator, we find that Cohen's *d* is .88, a very large effect size! Since many people who read our research report won't realize that effect size is more important than *p*-value, we want our *p*-value to match what we know is the importance of the study.

b. The small sample size of the study means we do not have a lot of power to find differences. If our sample sizes were 50 in each group, the *p*-value would certainly come out on the smaller side of .05 with the same effect size.

c. Since there is an outlier, the data do not meet the requirements of a parametric test; it is therefore a good idea to use a robust test.

d. It is totally legitimate to examine the results of tests that use differing amounts of means trimming; Wilcox (2003) states that 20% is a good amount as a general rule, but whatever amount we trim is still an objective way of getting rid of outliers (and not a subjective way like just removing one point).

Application Activities for the Paired-Samples T-Test

1. Answer:
French and O'Brien (2008) data. First, examine boxplots of the pairs of variables. This cannot be done with R Commander, so use the R Console. I give the syntax for the GRAM variable first:

```
boxplot(french$GRAM_1, french$GRAM_2)
```

In looking at the pairs of variables, for grammar participants made a lot of progress at Time 2, but there were many outliers at Time 1. The medians are not quite symmetrical in their boxes, but otherwise the distributions look good (besides the outliers). For receptive vocabulary the distributions look very symmetrical, but there is one outlier at Time 2. The distribution for the productive vocabulary looks exactly normal!

To perform the t-test, go to STATISTICS > MEANS > PAIRED T-TEST in R Commander. Choose the first and second variables (choose, say, GRAM_1 for the first variable and then GRAM_2 for the second variable). Make sure there are 104 participants for all of the tests (that will give a df=103). Here is the R code for the GRAM variable:

```
t.test(french$GRAM_1, french$GRAM_2,
alternative='two.sided', conf.level=.95, paired=TRUE)
```

All of the tests are statistical, and from the mean scores we can say the participants improved on all three measures over the course of their study. Use the STATISTICS > SUMMARIES > NUMERICAL SUMMARIES choice in R Commander to get mean scores and standard deviations for all tests. The improvement in grammar was the largest, with an average of 10.7 points out of 45 total. Participants on average improved least on the receptive vocabulary measure, with an average change of 5.3 points out of 60. I did the effect size calculations online, and they are all quite large.

Here are the results in tabular form:

Variable	*95% CI*	*Mean Time 1 (SD1)*	*Mean Time 2 (SD2)*	*N1/N2*	T-*Value*	P-*Value*	*Effect Size*
Grammar	−11.5, −9.8	16.6 (4.5)	27.2 (4.6)	104	−25.2	<.0005	2.3
Receptive vocab	−6.0, −4.7	32.8 (5.8)	38.1 (6.4)	104	−16.1	<.0005	0.9
Productive vocab	−12.9, −11.9	30.6 (5.9)	43.0 (6.5)	104	−53.4	<.0005	2.0

2. Answer:
Yates (2003) data. To perform the t-test, go to STATISTICS > MEANS > PAIRED T-TEST in R Commander. Test the scores for before and after for the lab, and later before and after for mimicry. Here is the R code for the lab scores:

```
t.test(yates$LabBefore, yates$LabAfter,
alternative='two.sided', conf.level=.95, paired=TRUE)
```

Use the numSummary() command to get means and standard deviations in order to calculate effect sizes.

```
numSummary(yates[,c("LabAfter", "LabBefore", "MimicryAfter", "MimicryBefore")],
statistics=c("mean", "sd", "quantiles"), quantiles=c(0,.25,.5,.75,1))
```

The results are gathered into the table below. Neither one of the tests is statistical, although the *p*-value for Mimicry is much smaller than that of Lab. Notice, however, that the effect size for Mimicry is quite large, more than one standard deviation. If I had been reporting the result I would have focused on the effect size and not the *p*-value. The fact that the *p*-value was not below *p*=0.05 was most likely due to the small sample size and not to any of the other factors that Yates posited (such as possibly having too short a period of testing, although it was a whole semester, and three hours a week not being enough to master linguistic mimicry). Clearly, this test needed more power, because the effect size of the mimicry task was quite large!

Variable	*95% CI*	*Mean Time 1 (SD1)*	*Mean Time 2 (SD2)*	*N1/N2*	T-*Value*	P-*Value*	*Effect Size*
Lab	−9.7, 8.1	109.4 (13.9)	110.2 (3.2)	10	−0.2	0.84	.08
Mimicry	−2.9, 13.5	112.0 (5.0)	106.7 (4.2)	10	1.46	0.18	1.15

3. Answer:
Larson-Hall and Connell (2005) data (LarsonHall.Forgotten.sav, imported as "forget"). To perform the t-test, go to STATISTICS > MEANS > PAIRED T-TEST in R Commander. Choose the variables "AccentR" and "AccentL." The R code is:

```
t.test(forget$AccentR, forget$AccentL,
alternative='two.sided', conf.level=.95, paired=TRUE)
```

The 95% CI does not go through zero, and the difference in means between the groups lies, with 95% confidence, somewhere between 3.7 and 2.7 points. So our Japanese users of English produce a word beginning with /l/ with a more native-like accent than a word beginning with /□/ (since the mean score for "AccentL" is larger). The effect size is a medium one.

Variable	*95% CI*	*Mean (SD1)*	*N*	T-*Value*	P-*Value*	*Effect Size*
AccentR	−3.7, −2.7	7.2 (2.0)	44	−13.4	p<.0005	.45
AccentL		10.4 (1.6)	44			

4. Answer:
French and O'Brien (2008) data. Here was the original robust t-test command:

```
rmmcppb(french$ANWR_1, french$ANWR_2, alpha=.05, est=mean, tr=.2, dif=T,
nboot=2000, BA=T, hoch=F)
```

The resultant *p*-value was less than .05. Let's try this test again but change the trimming to 10%:

```
rmmcppb(french$ANWR_1, french$ANWR_2, alpha=.05, est=mean, tr=.1, dif=T,
nboot=2000, BA=T, hoch=F)
```

This test is still statistical. Now let's try changing the "dif" term to F:

```
rmmcppb(french$ANWR_1, french$ANWR_2, alpha=.05, est=mean, tr=.2, dif=F,
nboot=2000, BA=T, hoch=F)
```

Using the hypothesis of equal measures of location (H_0: $\theta_1 = \theta_2$) we find a non-statistical *p*-value (.29). Changing the amount of trimming on this last equation also changes the size of the *p*-value (but not to below .05 for 10% or 15% trimming).

The important question is how to deal with the fact that, by using different tests, one can get different answers. I guess my answer would be that being up-front with your readers is probably the best tack to take. You might tell readers that "if I use this test I find a statistical result, but if I use this other one I get a non-statistical result." If readers are educated about effect sizes, which do *not* change by test (and you may have to educate them about this yourself!), then the test can be seen merely as an artifact and the attention that should really be directed would be to effect sizes.

5. Answer:
Yates (2003) data. Here is the command we would use to do a robust paired-samples t-test:

```
rmmcppb(yates$MimicryBefore, yates$MimicryAfter, alpha=.05, est=mean, tr=.2,
dif=T, nboot=2000, BA=T, hoch=F)
```

I could not find any configurations of this test which would cause the *p*-value to get close to 0! So maybe the tests are not as arbitrary as they seemed with the data above! However, the point that effect sizes are the most important piece of information still holds.

Application Activities for the One-Sample T-Test

1. Answer:
Torres (2004) data. In R Commander, choose STATISTICS > MEANS > SINGLE-SAMPLE T-TEST. Choose one variable at a time from "reading" and "listenin." Change null hypothesis from mu=0 to mu=3. Press OK. Here is the R code for this:

```
t.test(torres$listenin, alternative='two.sided', mu=3,
conf.level=.95)
```

Once finished, repeat for the other variable. The results will give you values for the *t*-value, the df, the *p*-value, and the CI for the table. To calculate the effect size, you'll need to obtain the standard deviation for each set first. In R Commander, choose STATISTICS > SUMMARIES > NUMERICAL SUMMARIES. Choose both "reading" and "listenin."

Results

Area	*Mean (sd)*	T-*Value*	*Df*	P-*Value*	*Effect Size (Cohen's d)*	*95% CI*
Reading	3.2 (1.16)	1.71	101	.196		2.97–3.42
Listening	3.29 (1.03)	2.88	101	.005		3.09–3.40

To calculate the effect size, use the mean and standard deviation given in the output:

$$\text{Reading} = \frac{3.20-3.0}{1.16} = .17,\ \text{Listening} = \frac{3.29-3.0}{1.03} = .28$$

These are small effect sizes.

2. Answer:
Torres (2004) data. Use the commands given in activity 1 to repeat the parametric analysis for the variables "culture" and "pron." To get the robust results, run the following command in R Console (making sure the WRS library is open first):

```
trimpb(torres$culture, tr=.2, null.value=3)
```

Repeat for the other variable of pronunciation. You will receive a *p*-value and a 95% CI from these commands. To calculate 20% trimmed means, use this command:

```
mean(torres$culture,tr=.2)
```

Results

Area	*20% Trimmed Mean*	P-*Value*	*95% CI of 20% Means Trimmed Percentile Bootstrap*
Culture	3.58	0	3.42–3.75
Speaking	3.39	0	3.18–3.56
Grammar	3.23	.09	2.97–3.48
Reading	3.26	.055	2.98–3.52
Listening	3.47	0	3.21–3.68

There is not a large difference between the robust t-test and the classic t-test with this data. I leave it to the reader to look at means and standard deviations with numSummary() and then calculate effect sizes.

3. Answer:
Dewaele and Pavlenko (2001–2003) data. Using R Commander, choose STATISTICS > MEANS > SINGLE-SAMPLE. Go through variables. I've started with "l1speak." Change the null hypothesis to mu=5. Press OK. Here is the R code for this command:

```
t.test(beq$l1speak, alternative='two.sided', mu=5, conf.level=.95)
```

The *p*-value is very small (and the confidence interval small as well, which is great!), meaning we can reject the null hypothesis that there is no difference between the people who rated themselves and a score of 5 for speaking proficiency. In other words, people did *not* rate themselves as fully proficient in their L1. Does this seem strange? Consider the confidence intervals. For speaking, it lies within [4.71, 4.78]. The true mean score lies, with 95% confidence, in this range. It is very high, but, because the group size is so large, any deviation away from 5 is going to be large enough to obtain *statistical* results. I leave it up to the reader to calculate the results for the other variables.

One more thing: it is easy to calculate effect sizes for one-way t-tests! The mean for "l1speak"=4.8, so:

$$\text{L1Speaking} = \frac{5-4.8}{.7} = .29\,.$$

This is a small to medium effect size.

Chapter 10: One-Way ANOVA

Application Activities for Boxplots with Overlaid Dotcharts

Note: If you are copying code directly from my documents and get an error message about an unexpected ')' in ")" then the problem might be the quotation marks! My MS Word document replaces straight quotation marks with "smart" quotation marks, and R doesn't like these!

1. Answer:
The plotting character is a box. Here is the code you will want to use:

```
bx.p<-boxplot(SyntaxVariety~Group, data=ellisyuan)
with (ellisyuan,
{bxp (bx.p, staplewex=1, boxfill="light grey", medlwd=8, medcol="white", boxwex=.5,
ylab="Syntactic variety", outl=F)
points (jitter (rep (1:3, each=14), 1.2),
unlist (split (SyntaxVariety, Group)),
cex=.8, pch=16)
}
)
```

2. Answer:
Here is the code to use in R Console:

```
bx.p<-boxplot(Disfluencies~Group, data=ellisyuan)
with (ellisyuan,
{bxp (bx.p, staplewex=1, notch=T, boxwex=.5, outl=F)
points (jitter (rep (1:3, each=14), 1.2),
unlist (split (Disfluencies, Group)),
cex=.5, pch=16)
}
)
```

It is the same as in activity 1, but a white box and black median line are the default, so all of the commands about median lines widths and colors, and boxplot color can be removed. The notch=T command is added to the bxp command, and the "SyntaxVariety" variable is replaced by "Disfluencies."

Note that you will get a warning about some notches going outside the hinges. Notches may not be the best choice here, but I wanted you to see what they looked like!

3. Answer:
Leow and Morgan-Short (2004) data. The first step is to make sure the non-think-aloud group comes first in a boxplot. So, to do this, first make a boxplot using R Commander, choosing GRAPHS > BOXPLOT. Pick “RecPostScore” in the “Variables” box. Click on “Plot by Groups” and pick “Group.” Press OK. Indeed, the non-think-aloud group does come first.

The shortest box should be the non-think-aloud group on the left-hand side. If it is not, change the order of the variables like this:

```
leow$Group=ordered(leow$Group,levels=c("Think Aloud", "Non Think Aloud"))
```

The boxplot with overlaid dotchart is created using the commands:

```
bx.p<-boxplot(RecPostScore~Group, data=leow)
with (leow,
{bxp (bx.p, staplewex=1, boxwex=.5, boxfill="blue", medlwd=8, medcol="red",
ylim=c(0,20), outl=F)
points (jitter (rep (1:2, each=39, len=77), 1.2),
unlist (split (RecPostScore, Group)),
cex=.5, pch=16)
}
)
```

Application Activities for One-Way ANOVAs

1. Answer:
Ellis and Yuan (2004) data. Examine the data first with boxplots. Use R Commander (GRAPHS > BOXPLOT) or use the basic syntax for the MSTTR variable, for example:

```
boxplot(MSTTR~Group,data=ellisyuan)
```

For error-free clauses, the PTP group has the widest variance, and both NP and PTP have outliers. For MSTTR, the OLP group has the widest variance, and the PTP group has some outliers. For SPM, the PTP group is skewed and has an outlier. The boxplots show that all of the variables violate some of the assumptions of parametric statistics.

To do the one-way ANOVA with R Commander choose STATISTICS > MEANS > ONE-WAY ANOVA. Only “Group” is available for the Group choice. Choose “ErrorFreeClauses,” “MSTTR,” and “SPM” in turn for the response variable, and tick the “Pairwise comparison of means” box. You’ll also need to pick a name for the model that R Commander will create, but you can just let R Commander do this automatically. Below I’ve gathered the results into tables. I calculated effect sizes with an online calculator, using the means and standard deviations given in the tables.

	Omnibus F Test	*Mean difference for NP-PTP + CI*	*Mean difference for NP-OLP + CI*	*Mean difference for PTP-OLP + CI*
SPM	$F_{2,39}$=11.19,	−3.77	.73 (−1.76, 3.21)	−4.50 (−6.98, −2.01)

	p=.000	(−6.25, −1.28)		
MSTTR	$F_{2,39}$=.18, p=.84	−.001 (−.03, .03)	.005 (−.02, .03)	.006 (−.03, .02)
Error-free clauses	$F_{2,39}$=3.04, p=.06	−.03 (−.13, .06)	.09 (.00, .18)	.06 (−.03, .15)

	Mean NP	*Mean PTP*	*Mean OLP*	*ES for NP-PTP*	*ES for NP-OLP*	*ES for PTP-OLP*	*ES for NP-PTP*
SPM	12.5 (2.0)	16.3 (3.3)	11.8 (2.7)	−1.39	.29	1.49	−1.39
MSTTR	.88 (.03)	.88 (.02)	.87 (.03)	0	.33	.39	0
Error-free clauses	.77 (.10)	.81 (.12)	.86 (.07)	−.36	−1.04	−.51	−.36

It's interesting that planning time had no effect on lexical variety (MSTTR). It's interesting to look at effect sizes here and see what effect sizes are found even when differences in mean scores between groups are fairly small. For error-free clauses, it seems that pre-task planning (PTP) had a large effect on improving the number of these. These are just a few comments; you will certainly find more to remark on as you study this data.

2. Answer:
Pandey (2000) data. Visually examine data with boxplots. Use R Commander (GRAPHS > BOXPLOT) to look at the gain scores (in turn) split by group, or use the R code:

```
boxplot(Gain1~Group, ylab="Gain1", xlab="Group", data=pandey)
```

Focus group B is a little bit skewed, and the control group has many outliers. The control group's variance is certainly much different from either Group A or Group B.

For planned comparisons, I wanted to test two hypotheses. The first is that Group A is different from Group B, while the second is that Focus A *plus* Focus B are different from Control A. I used this code to specify the contrasts I wanted:

```
contr=rbind("Focus A-Focus B"=c(1,-1,0), "(Focus A + Focus B)-Control A"=c(1,1,-2))
```

See the section on planned comparisons in the online document "One way ANOVA.One-way ANOVA test" for an explanation of these numbers in the command. Now we're ready to create a model.

```
pandey.anova=lm(Gain1~Group,data=pandey) #create initial model for ANOVA
pandey.pairs=glht(pandey.anova,linfct=mcp(Group=contr)) #create model using planned contrasts
confint(pandey.pairs) #call confidence intervals on planned contrasts
summary(pandey.pairs) #call for p-values
```

The results show that the first contrast just between Group A and Group B is statistical; mean difference=29.5, CI=21.1, 37.9, t=8.13, p<.001.

The second contrast of Groups A and B against the control is also statistical; mean difference=54.0, CI=41.9, 66.0, *t*=10.4, *p*<.001.

3. Answer:
The main problem with this design is that it is repeated measures, so data is not independent. If the researcher were to ask whether there were any difference between groups for only *one* of the three discourse completion tasks, assuming that each of the three classes received different treatments, this would be a valid one-way ANOVA. But to try to put the scores of all three discourse completion tasks together and then perform a one-way ANOVA would compromise the fundamental assumption of independence of groups in the one-way ANOVA (this research design is indicated in the table below).

Participant	*Group*	*DCT Score*
Chi-Wei	1	68
Chi-Wei	2	49
Chi-Wei	3	87
Tsi-Wen	1	53
…	…	…

4. Answer:
Inagaki and Long (1999) data. Visually examine the data. In R Commander, choose a boxplot with “GainAdj” as the variable, split by the groups variable of “AdjModelOrRecast.” Here is the syntax:

```
boxplot(GainAdj~AdjModelOrRecast, ylab="GainAdj", xlab="AdjModelOrRecast",
data=inagaki1999)
```

Repeat this step for the Locative formation. All groups are skewed for adjective and locative, and, since for adjective the control group did not make any gains, they have no variance. They should probably not be compared to the other two groups in that case.

Thus, for Adjective, I’ll conduct a t-test between the two experimental groups. First I’ll need to subset the data. In R Commander, choose DATA > ACTIVE DATA SET > SUBSET DATA SET. Remove the tick from “Include all variables” and choose only “GainAdj.” The subset expression is: “AdjModelOrRecast==“Recast”” and I give it the name il.adj.recast. Here is the R syntax for this step:

```
il.adj.recast <- subset(inagaki1999, subset=AdjModelOrRecast=="Recast",
select=c(GainAdj))
```

Go back to main inagaki1999 data set and repeat with “AdjModelOrRecast==“Model”” (and I named it il.adj.model). Now run a t-test using R syntax:

```
t.test(il.adj.recast,il.adj.model)
```

The result is not statistics, *t*(12.4)=−.34, *p*=.74. Effect size is .17, quite small. We conclude there is no difference between the two groups in how they performed on adjectives.

For Locative, run a one-way ANOVA. First, since you were previously subsetting, make sure inagaki1999 is the active data set in R Commander, and choose STATISTICS > MEANS >

ONE-WAY ANOVA. Choose "LocModelOrRecast" for Groups and "GainLoc" for response variable. Tick the comparison of means and go!

The main ANOVA summary shows that the omnibus test is not statistical, $F_{2,21}$=.37, *p*=.69. We won't look any further to consider the post-hocs then. We conclude there is no difference between groups in how accurate they are on locatives either.

I note that one possible problem with this study was that only three items per structure were used. There might have been more differentiation with a larger number of items!

5. Answer:
Dewaele and Pavlenko (2001–2003) data, BEQ.Context file. Visually examine the data. In R Commander call for a boxplot (GRAPHS > BOXPLOT), or use the syntax:

```
boxplot(l2speak~L2Context, ylab="l2speak", xlab="L2Context", data=beqContext)
```

Repeat this for the "l2_read" variable. Boxplots show that all data are skewed, and there are some outliers.

To conduct the one-way ANOVA in R Commander, make sure beqContext is the active data set in R Commander, and choose STATISTICS > MEANS > ONE-WAY ANOVA. Choose "L2Context" for Groups and "l2speak" for response variable (repeat with "l2_read"). Tick the comparison of means box and press OK.

Results: Looking at the descriptive stats, we see that the highest scores for both speaking and reading were by those who reported learning the L2 both naturalistically and in an instructed way (the "Both" choice). Probably owing to the fact that we have extremely large sample sizes here, the omnibus test for both ANOVAs is statistical (this result is under the line "summary (AnovaModel.X)"). What we should be more concerned about is the post-hocs and their associated effect sizes. These can be seen under the line "confint(.Pairs)" in the output. I've summarized the data for the speaking in the table below.

	Mean Natural	*Mean Instruct*	*Mean Both*	*ES for Natl-Instruct (CI)*	*ES for Natl-Both*	*ES for Instruct-Both*
L2 Speaking	4.3 (.98)	3.8 (1.2)	4.4 (.85)	.53 (.31, −.75)	.08 (−.13, .30)	.61 (.46, .78)

For L2 speaking, differences are statistical for the comparison between instruction and both contexts and between naturalistic and instructed (but not between naturalistic and both contexts). The effect sizes for these are medium to large (calculating effect sizes using an online calculator with the mean scores and standard deviations).

I will leave the discussion of the L2 reading data to the reader to examine.

Chapter 11: Factorial ANOVA

Application Activities with Factorial ANOVA

1. Answer:
Obarow (2004) data. Use Obarow.Story2.sav file. To get the file in shape, you'll need to compute a gain score, recode the "trtmnt2" variable into separate columns regarding the

presence or absence of Music and Pictures, and decide if you want to delete some cases of individuals who scored highly on the pre-test.

To select and take out any cases where individuals scored above 17 on the pre-test, you can subset based on the pre-test score. The R code for doing this if students scored above 17 is:

```
new.obarow <- subset(obarow2, subset=pretest2<18)
```

If you do this, it results in retaining only 61 participants out of an original 81. However, I will not do this here, and will retain the original 81 participants in my further calculations.

To change the "trtmnt2" variable into two different IVs we will need to recode the data. First check out the names of the conditions in the "trtmnt2" variable:

```
levels(obarow2$trtmnt2)
```

Now we are ready to go to DATA > MANAGE VARIABLES IN ACTIVE DATA SET > RECODE VARIABLES in R Commander. Put in the following directives:

```
'"no music no pics"="no"
"no music yes pics"="no"
"yes music yes pics"="yes"
"yes music no pics"="yes"
```

I called this new factor "music." Here is the R code for the process:

```
obarow2$music <- recode(obarow2$trtmnt2,
'"no music no pics"="no"; "no music yes pics"="no"; "yes music yes pics"="yes"; "yes
music no pics"="yes"; ', as.factor.result=TRUE)
```

Repeat the same process for pictures. If you ask for the names() now for the obarow2 data set, you will see separate columns for "music" and "pictures."

Lastly, to compute a gain score choose DATA > MANAGE VARIABLES IN ACTIVE DATA SET > COMPUTE NEW VARIABLE in R Commander. Call variable gain score. In Numeric Expression box, put "posttest2-pretest2." Press OK.

To examine the data visually, make boxplots of the gain score divided into the different groups (music, pictures, gender). You can call up boxplots in R Commander with GRAPHS > BOXPLOT; then choose which variable to split groups by. From the boxplots we see that males have a larger variance than females on the gain score, there are lots of outliers identified for the "pictures present" group, and there are a few outliers for the "no music" group. Thus we see that there is some departure from normality and heteroscedasticity in our data.

To numerically examine the data, use STATISTICS > SUMMARIES > NUMERICAL SUMMARIES in R Commander. Pick the "gainscore" and examine it by differently split groups.

To perform the factorial ANOVA (2×2×2 in this case), we can follow the same steps that were seen in the online document "Factorial ANOVA.Factorial ANOVA test" for the first Obarow test (see Figure 1 in that document). In R Commander go to STATISTICS > MEANS >

MULTI-WAY ANOVA and choose the three factors of "gender," "music," and "pictures." The response variable is "gainscore."

In this full factorial model none of the terms are statistical. Summarize this model:

```
summary(AnovaModel.8)
anova(AnovaModel.8)
```

We find this accounts for only about 5% of the variance in scores (the R^2 value). Of course, if we wanted to stick with this model and just report the results (in the same way SPSS does), run an anova() on the model to get F values, degrees of freedom, and *p*-values for an ANOVA table.

Run a bootstrap step function on the model that R Commander created (for me it was named AnovaModel.8).

```
library(bootStepAIC)
boot.stepAIC(AnovaModel.8,data=obarow2)
```

The step function says that our minimally adequate model will consist of the gain score modeled by the main effect of pictures only. In other words, the regression model will be:

Gainscore=1.3 −.55 (Pictures)

Constructing a new model with just this and doing a summary, we find this accounts for only 3% of the total variance. It seems that the variables that were tested are not doing the best job of explaining the variance in the gain scores.

```
AnovaModel.9=lm(gainscore~pictures, data=obarow2)
summary(AnovaModel.9)
```

2. Answer:
Larson-Hall and Connell (2005) data. Use LarsonHall.Forgotten.sav (forget). First, examine boxplots (the same way as in activity 1), putting "SentenceAccent" in the Variable box and "sex" (and then "Status") in the Category Axis box. All plots show outliers, and variances seem to be quite different for males and females, but data seems symmetric. Variances seem equal for all immersion students, and data seems normally distributed.

To perform the factorial ANOVA (3×2 in this case), in R Commander go to STATISTICS > MEANS > MULTI-WAY ANOVA and choose the two factors of "sex" and "Status." The response variable is "SentenceAccent." The ANOVA finds the main effects of "sex" and "Status" to be statistical but not the interaction between the two. I leave it to the reader to report on the results of a minimally adequate model, if you would rather.

You could report these results as something like:

A factorial ANOVA testing for the interaction between sex and status found a main effect for sex ($F_{1, 38}$=6.6, *p*=.01) and for status ($F_{2, 38}$=10.9, *p*<.001). However, the interaction between sex and status was not statistical ($F_{2, 38}$=0.8, *p*=.46).

Now, however, you want to say something further about status, because there are three levels and you don't know which status is different from the other. From the mean scores for gender (STATISTICS > SUMMARIES > NUMERICAL SUMMARY) we can see that females performed better than males, and since the comparison is statistical we can say females performed statistically better than males. But for status the Early group did best of all, the Non group did worst, and the Late group was in the middle.

Probably the easiest way to go about this in R Commander would be just to do a one-way ANOVA for the "Status" variable, because this will run post-hoc comparisons. Choose STATISTICS > MEANS > ONE-WAY ANALYSIS OF VARIANCE. Choose "Status" for the Group and "SentenceAccent" for the response variable. Tick the "Pairwise comparison of means" box. The tests show that non- and late immersionists are statistically worse than early immersionists, but the non- and late immersionists don't differ from each other statistically). I leave it to the reader to report on those confidence intervals.

On the other hand, for doing post-hoc comparisons, in the R Console it would be easy just to use the glht() command on the existing best model to obtain pairwise comparison, like this:

```
summary(glht(AnovaModel.10, linfct=mcp(Status="Tukey")))
```

This command shows the same results as were reported above for the one-way ANOVA. For assumptions, run the following commands (put in the correct name of your model):

```
plot(AnovaModel.9)
plot(AnovaModel.9)
plot(AnovaModel.9)
vif(AnovaModel.9)
library(MASS)
plot(row.names(forget), stdres(AnovaModel.9))
```

3. Answer:
Eysenck (1974) data. First, rearrange the data from the wide form to the long form. In R Commander, choose DATA > ACTIVE DATA SET > STACK VARIABLES IN ACTIVE DATA SET. Choose all five variables in the "Variables" box. Name this "EyStack." Name the variable "Recall" and the factor "Group." This is great but doesn't actually get us the second variable of "AgeGroup." However, we see that the first ten people in each row are categorized as "Old" and the second ten as "Young," so we can construct our own second factor for the "EyStack" data set this way:

```
AgeGroup=gl(2, 10, 100, c("Old", "Young")) #create the factor (see Appendix A, creating a factor)
AgeGroup #look at it and make sure it's what I want
length(AgeGroup)
length(EyStack$Group) #make sure they're the same length!
EyStack=cbind(EyStack, AgeGroup)
names(EyStack)
```

OK, now we're ready to go! To perform the factorial ANOVA, first make sure your updated version of the data frame is the active data set (since we just changed it, you may have to switch to a different data set and then come back to EyStack to get the update). Now choose

STATISTICS > MEANS > MULTI-WAY ANOVA and choose the two factors of “AgeGroup” and “Group.” The response variable is “Recall.” I leave it to the reader to report on the results of this test!

Chapter 12: Repeated-Measures ANOVA

Application Activities for Interaction (Means) Plots and Parallel Coordinate Plots

1. Answer:
Lyster (2004) data. Here is the syntax for the means plot:

```
interaction.plot(lysterMeans$Cond, lysterMeans$Time, lysterMeans$ClozeTask,
type=c("b"), xlab="Group", ylab="Mean score")
```

The highest mean is on Immediate task=FFIprompt; it is the same for the delayed post-test. Means are parallel for both post-tests. Only the Comparison group did not make gains, but the FFIprompt made almost twice as much gain as the FFIrecast or FFIonly groups. The trends here are faster to discern than the parallel coordinate plot, but of course show only group means. As a reader, I’d like to see both plots!

2. Answer:
Larson-Hall (2004) data. Plot where separate lines are the three contrasts:

```
interaction.plot(LHMeans$level,LHMeans$contrast, LHMeans$score,type=c("b"))
```

This shows that scores were consistently high for F_X (which for some reason is in the middle of 1 and 2, but I’m not sure why!). Scores for R_L went up with increasing proficiency, but scores for SH_SHCH bounced up for Intermediates but came back down for Advanced learners for some reason.

Plot where separate lines are the four groups:

```
interaction.plot(LHMeans$contrast,LHMeans$level, LHMeans$score,type=c("b"))
```

This means plot shows Beginners basically did the worst and native Russians (NR) basically did the best. The Intermediates and Advanced groups, however, bounced around in the same space.

3. Answer:
Lyster (2004) data. Here is the syntax for the parallel coordinate plot (first open the lattice library):

```
parallel(~lyster[3:5]|lyster$Cond)
```

Again there is a sharp increase in scores for the FFIprompt group at the first post-test, markedly so over the other groups. The FFIrecast seems to be in second place, with many participants increasing, at least for the first post-test. The other two groups seem randomly to have increases and decreases. All groups show some backsliding for the delayed post-test, though.

4. Answer:

Murphy (2004) data. Here is the syntax for the plot:

```
parallel(~murphy[2:4]|murphy$group)
parallel(~murphy[2:7]|murphy$group)
```

These plots are rather difficult to read and do not show too much patterning, since there are not that many participants, but one might say that, for the regular verbs, NNS tended to stick toward regular endings more than NS adults did. For irregular verbs, NS seemed to pattern more closely together than did the NNS, who were all over the place. Still, there were not any strong patterns at all.

Application Activities for Mixed-Effects Models

1. Answer:
Murphy (2004) data. First, to check on whether the variables are factors, use the str() command:

```
str(murphyLong)
```

I find that only the factors are duly labeled. In fact, I found that the nlme library commands would not work until the variables of verbtype, similarity, and group were true factors with categorical name (I had to go back into SPSS and change this to get it to work; trying just to make the variable into a factor through R wasn't good enough).

Now let's set up a mixed-effect model using the syntax I proposed in the online document "Repeated Measures ANOVA.Performing an RM ANOVA the fixed effects way" first.

```
library(nlme)
murphy.m1=lme(fixed=verbs~group*verbtype*similarity,
random=~1|similarity/participant, data=murphyLong)
anova(murphy.m1)
```

The ANOVA shows that almost all of the terms in the fixed effects are statistical (the similarity term has no denominator DF and so cannot be tested), with *p*-values below .05. Let's make another model with just participant in the random effects section and compare the two models.

```
murphy.m2=lme(fixed=verbs~group*verbtype*similarity,
random=~1|participant,data=murphyLong)
anova(murphy.m1,murphy.m2)
Model df AIC BIC logLik Test L.Ratio p-value
murphy.m1 1 21 942.8186 1023.3497 -450.4093
murphy.m2 2 20 896.2860 972.9822 -428.1430 1 vs 2 44.53266 <.0001
```

Model 2 has lower scores on all fit criteria (AIC, BIC, log likelihood test); go with Model 2.

```
anova(murphy.m2)
```

All terms except the group versus verbtype interaction are statistical. Try removing this interaction:

```
murphy.m2=lme(fixed=verbs~group*verbtype*similarity,
random=~1|participant,data=murphyLong, method="ML") #change method to make
comparison
murphy.m3=update(murphy.m2,~.-group:verbtype, data=murphyLong,
method="ML")
```

There is no difference between models, so keep the simpler one (m3). Calculate variances for the random effects.

```
summary(murphy.m3)
Random effects:
Formula: ~1 | participant
(Intercept) Residual
StdDev: 0.4725816 0.6805839
```

Now calculate the variances:

```
sd=c(.47, .68)
var=sd^2
100*var/sum(var)
[1] 32.32841 67.67159
```

The subject effect explains 32% of the total variance of the model.

Examining model assumptions:

```
plot(murphy.m3, main="Murphy (2004) data") #discrete but no heteroscedasticity
plot(murphy.m3,resid(.,type="p")~fitted(.)|group) #no problem by group
with(murphy, plot(murphy.m1,verbs~fitted(.))) #linear? hard to tell bec. discrete
qqnorm(murphy.m1,~resid(.)|group) #line not straight, problems with normality
qqnorm(murphy.m1,~resid(.)|verbtype:similarity) #lines are curvy, not normal, clear
outliers for irregular:intermediate group
qqnorm(murphy.m3,~ranef(.)) #outliers
```

2. Answers:
a. (for example) toth.m1=lme(fixed=group*time, random=~time|participant)
b. erdener.m1=lme(fixed=condition*L1*L2, random=~condition|participant)
c. larsonhall.m1=lme(fixed=contrast*level, random=~contrast|participant)

3. Answer:
Lyster (2004) data. Rearrange data from the imported lyster file:

```
names(lyster)
lysterLong<-stack(lyster[,c("PreBinary", "Post1Binary", "Post2Binary")])
head(lysterLong) #just want to look at what I have before I name it
names(lysterLong)=c("binary", "index") #name main response variable ‘binary’
levels(lyster$Cond) #I need to make group variable and want to know what’s there
numSummary(lyster[,"BinaryGain1"], groups=lyster$Cond, statistics=c("mean",
"sd", "quantiles"), quantiles=c(0,.25,.5,.75,1)) # get summary with Ns for each group
group=rep(rep(1:4,c(38,49,42,51)),3) #create a new factor for the variables I stacked
```

```
length(group)
length(lysterLong$binary)
#lengths match up—good!
participant=factor(rep(c(1:180),3)) #I have 180 different participants
time=gl(3,180,540) #check to see if this is going to correspond to 3 different times—it
does!
lysterLong=data.frame(cbind(binary=lysterLong$binary, group, time, participant))
#put it all together
head(lysterLong) #check it out before continuing
str(lysterLong) #no factors here, so fix it!
lysterLong$group=as.factor(lysterLong$group)
levels(lyster$Cond)
[1] "FFIrecast" "FFIprompt" "FFIonly" "Comparison"
levels(lysterLong$group)=c("FFIrecast", "FFIprompt", "FFIonly", "Comparison")
lysterLong$time=as.factor(lysterLong$time)
names(lyster)
[1] "Participant" "Cond" "PreBinary" "Post1Binary"
[5] "Post2Binary" "PreTaskCompl" "Post1TaskCompl" "Post2TaskCompl"
[9] "BinaryGain1" "BinaryGain2" "CompGain1" "CompGain2"
levels(lysterLong$time)=c("PreBinary", "Post1Binary", "Post2Binary")
lysterLong$participant=as.factor(lysterLong$participant)
str(lysterLong)
'data.frame': 540 obs. of 4 variables:
$ binary : num 42 36 35 39 38 36 32 27 37 39 ...
$ group : Factor w/ 4 levels "FFIrecast","FFIprompt",..: 1 1 1 1 1 1 1 1 1 1 ...
$ time : Factor w/ 3 levels "PreBinary","Post1Binary",..: 1 1 1 1 1 1 1 1 1 1 ...
$ participant: Factor w/ 180 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...
```

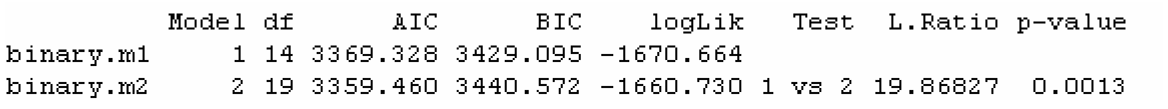
Data set lysterLong ready to go!

Mixed-effects model:

```
binary.m1<-lme(fixed=binary~group*time, random=~1|participant,data=lysterLong)
anova(binary.m1)
numDF denDF F-value p-value
(Intercept) 1 352 6951.671 <.0001
group 3 176 15.496 <.0001
time 2 352 70.544 <.0001
group:time 6 352 14.386 <.0001
```

No need to look for the minimal model, since all variables are statistical. Let's try some other variations on the random part.

```
binary.m2<-lme(fixed=binary~group*time,
random=~time|participant,data=lysterLong)
anova(binary.m1,binary.m2)
```



```
          Model df      AIC      BIC    logLik   Test  L.Ratio p-value
binary.m1     1 14 3369.328 3429.095 -1670.664
binary.m2     2 19 3359.460 3440.572 -1660.730 1 vs 2 19.86827  0.0013
```

Model 2 has lower AIC but higher BIC, just as before. Look at confidence intervals for Model 2:

```
intervals(binary.m2)
```

The intervals are way too wide for random effects; abandon Model 2; stick with Model 1. Intervals are just fine for Model 1.

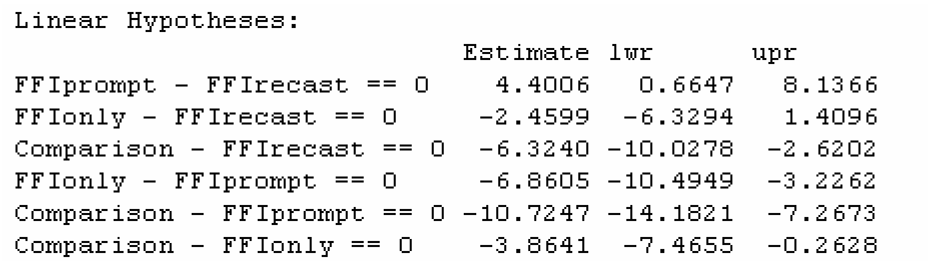
Info for main effects and interaction effects for Model 1 in ANOVA print-out. Variances for this model can be obtained through the summary() command. They are 5.28 for Intercept and 4.06 for Residual.

```
sd=c(5.28, 4.06)
var=sd^2
100*var/sum(var)
```

Effect of participants accounts for 62.8% of variance in the model.

To look at differences in the interaction between groups and conditions in the first post-test, first subset "lysterLong" to contain just data for the immediate post-test:

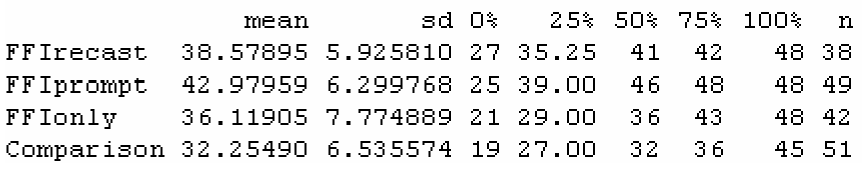
```
lyster.post1=subset(lysterLong,subset=time=="Post1Binary")
lysterPost=lm(binary~group,data=lyster.post1)
library(multcomp)
.Pairs=glht(lysterPost,linfct=mcp(group="Tukey"))
confint(.Pairs)
```



```
Linear Hypotheses:
                            Estimate lwr      upr
FFIprompt - FFIrecast == 0    4.4006   0.6647   8.1366
FFIonly - FFIrecast == 0     -2.4599  -6.3294   1.4096
Comparison - FFIrecast == 0  -6.3240 -10.0278  -2.6202
FFIonly - FFIprompt == 0     -6.8605 -10.4949  -3.2262
Comparison - FFIprompt == 0 -10.7247 -14.1821  -7.2673
Comparison - FFIonly == 0    -3.8641  -7.4655  -0.2628
```

For the first post-test, 95% confidence intervals show differences between all groups except FFI only versus FFI recast. Get mean scores of the conditions on the binary test:

```
numSummary(lyster.post1[,"binary"],groups=lyster.post1$group)
```



```
               mean       sd 0%   25% 50% 75% 100%  n
FFIrecast  38.57895 5.925810 27 35.25  41  42   48 38
FFIprompt  42.97959 6.299768 25 39.00  46  48   48 49
FFIonly    36.11905 7.774889 21 29.00  36  43   48 42
Comparison 32.25490 6.535574 19 27.00  32  36   45 51
```

For the first post-test, FFI PROMPT > FFI RECAST, FFI ONLY > COMPARISON

Examining model assumptions:

```
plot(binary.m1,) #possibly some heteroscedasticity in upper right corner
plot(binary.m1,resid(.,type="p")~fitted(.)|group) #most groups except Comparison
showing heteroscedasticity on right side of graph
with(lysterLong, plot(binary.m1,binary~fitted(.))) #seems linear
qqnorm(binary.m1,~resid(.)|group) #data looks normal
qqnorm(binary.m1,~ranef(.)) #looks good
```

There seems to be a problem with heteroscedasticity but not normality in this data.

Chapter 13: ANCOVA

Application Activities for ANCOVA

1. Answer:
Class Time data set. The structure of this problem is a one-way ANOVA with one covariate. First, check linearity between covariate and dependent variable by looking at a scatterplot between "PreTestScores" and "PostTestScores." Using R Commander, choose GRAPHS > SCATTERPLOT. The Loess line fits the regression line fairly well, so we will assume the data are linear. To test the homogeneity of regression assumption, we set up a regression model with an interaction between the independent variable and the covariate:

```
class.m1=aov(PostTestScores~TimeOfClass*PreTestScores, data=classtime)
summary(class.m1)
```

The interaction is not statistical, so we may proceed. Now let's set up a regression model with the covariate added to the model:

```
class.m2=aov(PostTestScores~TimeOfClass + PreTestScores, data=classtime)
anova(class.m2)
```

Both terms (Time of Class and PreTestScores) are statistical. Thus we can report that there is a statistical effect of time of class ($F_{4,41}=22.0$, $p<.001$) even with pre-test scores on motivation and enthusiasm controlled for (I don't think we are really interested in the fact that the PreTest Scores are statistical, so I'm not reporting on that). Now we would like to say which times of day have higher scores on the motivation and enthusiasm measure than others. We have five classes per day to look at. Using the `numSummary()` command (in R Commander, choosing STATISTICS > SUMMARIES > NUMERICAL SUMMARY) we can see that the 10 a.m. class has the highest mean score, while the 8 a.m. class has the lowest (somehow, not a surprise!).

We can use the `glht()` command to perform pairwise comparisons on our ANCOVA model:

```
summary(glht(class.m2, linfct=mcp(TimeOfClass="Tukey")))
```

Ten comparisons among the five conditions are done. There are statistical differences for two of the comparisons—the 8 a.m. vs. 10 a.m. class and the 10 a.m. class vs. the 2 p.m. class. The *p*-value for the comparison between the 10 a.m. and 1 p.m. class is not statistical, but the *p*-value is low (.08).

We can look at whether this model satisfies assumptions:

```
plot(class.m2) #some homoscedasticity in the first plot
```

Now let's run a robust ANCOVA. Here we can compare only two levels of a group at a time, which is why I asked you to look only at the comparison between the 10 a.m. and 1 p.m. class. We'll need to subset the data for only these classes.

```
classtime.10=subset(classtime,subset=TimeOfClass=="10 a.m. ")
classtime.1=subset(classtime,subset=TimeOfClass=="1 p.m. ")
```

Now we can use ancboot() to compare the scores for the a.m. versus p.m. classes on the "PostTestScores." We'll need to set up our Xs and Ys like this:

```
x1=classtime.10$PreTestScores
y1= classtime.10$ PostTestScores
x2= classtime.1$ PreTestScores
y2= classtime.1$ PostTestScores
ancboot(x1,y1,x2,y2,fr1=1,fr2=1,tr=.2, nboot=599, plotit=T, pts=NA)
```

My attempt to analyze this results in this error message:

```
Error in sample(xcen, size = length(x[[j]]) * nboot, replace = T) :
invalid first argument
```

My sample is too small to sample at different points, so I cannot perform the robust ANCOVA on this data.

2. Answer:
LarsonHall (2008) data. First we'll examine the special assumptions for an ANCOVA. We don't want there to be a strong correlation between the covariates of language aptitude (aptscore) and total amount of input in English (totalhrs). Actually, we already tested that for the model where the "GJTScore" was the dependent variable, so we don't need to test it again. There is no strong correlation between the covariates.

Next, we'll need to test for the assumption of homogeneity of regression slopes by looking for a statistical interaction between the covariates and the grouping variable of "erlyexp." I'll use the same syntax as was used in the online document "ANCOVA: Two-way ANCOVA with two covariates," just substituting the "gjtscore" for the "rlwscore":

```
check1=aov(rlwscore~erlyexp* aptscore, data=larsonhall2008)
summary(check1)
check2=aov(rlwscore~erlyexp* totalhrs, data=larsonhall2008)
summary(check2)
```

In both cases the interaction is not statistical, so we may proceed with the ANCOVA analysis.

Create a regression model:

```
larsonhall.m1=aov(rlwscore~erlyexp*sex+aptscore+totalhrs, data=larsonhall2008)
summary(larsonhall.m1)
```

The summary finds that only “erlyexp” is statistical, so remove the two-way interaction:

```
larsonhall.m2=update(larsonhall.m1,~.-erlyexp:sex)
anova(larsonhall.m1, larsonhall.m2)
```

There is no statistical difference between models, so keep the simpler Model 2. A look at the summary for Model 2 shows that nothing has changed—only the main effect of early experience is statistical. We don’t want to remove the factors of aptitude score or total hours, though, because we still want to factor their effects out, even if the terms themselves aren’t statistical. The only other term we want to consider removing for the minimal adequate model is “sex,” which has a *p*-value of *p*=.09. Let’s remove this term:

```
larsonhall.m3=update(larsonhall.m2,~.- sex)
anova(larsonhall.m2, larsonhall.m3)
```

The ANOVA shows no difference, so keep the simpler model. Thus, we can say that the best model of the RLW score is done with the variable of early experience ($F_{1, 196}$=6.7, *p*=.01), with the variables of aptitude score and total hours of input factored out.