Chapter 11
Stuff for Students

11.1 R

R is available from the **CRAN** website (https://cran.r-project.org/). As of January 2020, the author's personal computer has Version 3.3.1 (June 21, 2016) of *R*. *R* is similar to *Splus*, but is free. *R* is very versatile since many people have contributed useful code, often as packages.

Downloading the book's files into R
Many of the homework problems use *R* functions contained in the book's website (http://parker.ad.siu.edu/Olive/slearnbk.htm) under the file name *slpack.txt*. The following two *R* commands can be copied and pasted into *R* from near the top of the file (http://parker.ad.siu.edu/Olive/slrhw.txt).

Downloading the book's R functions *slpack.txt* and data files *lregdata.txt* into *R*: the commands

```
source("http://parker.ad.siu.edu/Olive/slpack.txt")
source("http://parker.ad.siu.edu/Olive/sldata.txt")
```

can be used to download the *R* functions and data sets into *R*. Type *ls()*. Nearly 70 *R* functions from *slpack.txt* should appear. In *R*, enter the command *q()*. A window asking "*Save workspace image?*" will appear. Click on *No* to remove the functions from the computer (clicking on *Yes* saves the functions in *R*, but the functions and data are easily obtained with the source commands).

Citing packages
We will use *R* packages often in this book. The following *R* command is useful for citing the Mevik et al. (2015) `pls` package.

```
citation("pls")
```

Other packages cited in this book include `MASS` and `class`: both from Venables and Ripley (2010), `glmnet`: Friedman et al. (2015), and `leaps`: Lumley (2009).

This section gives tips on using *R*, but is no replacement for books such as Becker et al. (1988), Crawley (2005, 2013), Fox and Weisberg (2010), or Venables and Ripley (2010). Also see Mathsoft (1999ab) and use the website (www.google.com) to search for useful websites. For example enter the search words *R documentation.*

The command *q()* gets you out of *R.*

Least squares regression can be done with the function *lsfit* or *lm.*

The commands *help(fn)* and *args(fn)* give information about the function fn, e.g. if fn = lsfit.

Type the following commands.

```
x <- matrix(rnorm(300),nrow=100,ncol=3)
y <- x%*%1:3 + rnorm(100)
out<- lsfit(x,y)
out$coef
ls.print(out)
```

The first line makes a 100 by 3 matrix x with N(0,1) entries. The second line makes $y[i] = 0+1*x[i,1]+2*x[i,2]+3*x[i,2]+e$ where e is N(0,1). The term 1:3 creates the vector $(1,2,3)^T$ and the matrix multiplication operator is $\%*\%$. The function `lsfit` will automatically add the constant to the model. Typing "out" will give you a lot of irrelevant information, but *out$coef* and *out$resid* give the OLS coefficients and residuals respectively.

To make a residual plot, type the following commands.

```
fit <- y - out$resid
plot(fit,out$resid)
title("residual plot")
```

The first term in the plot command is always the horizontal axis while the second is on the vertical axis.

To put a graph in *Word,* hold down the *Ctrl* and *c* buttons simultaneously. Then select "Paste" from the *Word* menu, or hit *Ctrl* and *v* at the same time.

To enter data, open a data set in *Notepad* or *Word.* You need to know the number of rows and the number of columns. Assume that each case is entered in a row. For example, assuming that the file *cyp.lsp* has been saved on your flash drive from the webpage for this book, open *cyp.lsp* in *Word.* It has 76 rows and 8 columns. In *R* , write the following command.

```
cyp <- matrix(scan(),nrow=76,ncol=8,byrow=T)
```

A data frame is a two-dimensional array in which the values of different variables are stored in different named columns.

Then copy the data lines from *Word* and paste them in *R.* If a cursor does not appear, hit *enter.* The command *dim(cyp)* will show if you have entered the data correctly.

Enter the following commands

```
cypy <- cyp[,2]
cypx<- cyp[,-c(1,2)]
lsfit(cypx,cypy)$coef
```

to produce the output below.

```
   Intercept           X1           X2           X3
205.40825985   0.94653718   0.17514405   0.23415181
          X4           X5           X6
  0.75927197  -0.05318671  -0.30944144
```

Making functions in R is easy.

For example, type the following commands.

```
mysquare <- function(x){
# this function squares x
r <- x^2
r }
```

The second line in the function shows how to put comments into functions.

Modifying your function is easy.

Store a function as text file, modify the function in *Notepad*, and copy and paste the function into *R*.

To save data or a function in *R*, when you exit, click on *Yes* when the "*Save worksheet image?*" window appears. When you reenter *R*, type *ls()*. This will show you what is saved. You should rarely need to save anything for this book. To remove unwanted items from the worksheet, e.g. *x*, type *rm(x)*,
pairs(x) makes a scatterplot matrix of the columns of *x*,
hist(y) makes a histogram of *y*,
boxplot(y) makes a boxplot of *y*,
stem(y) makes a stem and leaf plot of y,
scan(), source(), and *sink()* can be are useful.
To type a simple list, use *y <− c(1,2,3.5).*
The commands *mean(y), median(y), var(y)* are self explanatory.

The following commands are useful for a scatterplot created by the command *plot(x,y).*
lines(x,y), lines(lowess(x,y,f=.2))
identify(x,y)
abline(out$coef), abline(0,1)

The usual arithmetic operators are $2 + 4$, $3 - 7$, $8 * 4$, $8/4$, and

```
2^{10}.
```

The *i*th element of vector y is $y[i]$ while the ij element of matrix x is $x[i, j]$. The second row of x is $x[2,]$ while the 4th column of x is $x[, 4]$. The transpose of x is *t(x)*.

The command *apply(x,1,fn)* will compute the row means if fn = mean. The command *apply(x,2,fn)* will compute the column variances if fn = var. The commands *cbind* and *rbind* combine column vectors or row vectors with an existing matrix or vector of the appropriate dimension.

Getting information about a library in R
In *R*, a *library* is an add–on package of *R* code. The command *library()* lists all available libraries, and information about a specific library, such as `leaps` for variable selection, can be found, e.g., with the command
library(help=leaps).

Downloading a library into R
Many researchers have contributed a *library* or *package* of *R* code that can be downloaded for use. To see what is available, go to the website
(http://cran.us.r-project.org/) and click on the Packages icon.

Following Crawley (2013, p. 8), you may need to “Run as administrator” before you can install packages (right click on the *R* icon to find this). Then use the following command to install the *glmnet* package.

```
install.packages("glmnet")
```

Open *R* and type the following command.
library(glmnet)
Next type *help(glmnet)* to make sure that the library is available for use.

Warning: *R* is free but not fool proof. If you have an old version of *R* and want to download a library, you may need to update your version of *R*. The libraries for robust statistics may be useful for outlier detection, but the methods have not been shown to be consistent or high breakdown. All software has some bugs. For example, Version 1.1.1 (August 15, 2000) of *R* had a random generator for the Poisson distribution that produced variates with too small of a mean θ for $\theta \geq 10$. Hence simulated 95% confidence intervals might contain θ 0% of the time. This bug seems to have been fixed in Versions 2.4.1 and later. Also, some functions in *lregpack* may no longer work in new versions of *R*.

11.2 Hints for Selected Problems

1.9. See Example 1.7.

3.7 Note that $\boldsymbol{Z}_A^T \boldsymbol{Z}_A = \boldsymbol{Z}^T \boldsymbol{Z}$,

$$\boldsymbol{G}_A \ \boldsymbol{\eta}_A = \begin{pmatrix} \boldsymbol{G\eta} \\ \sqrt{\lambda_2^*} \ \ \boldsymbol{\eta} \end{pmatrix},$$

and $\boldsymbol{Z}_A^T \boldsymbol{G}_A \boldsymbol{\eta}_A = \boldsymbol{Z}^T \boldsymbol{G\eta}$. Then

$$RSS(\boldsymbol{\eta}_A) = \|\boldsymbol{Z}_A - \boldsymbol{G}_A\boldsymbol{\eta}_A\|_2^2 = (\boldsymbol{Z}_A - \boldsymbol{G}_A\boldsymbol{\eta}_A)^T(\boldsymbol{Z}_A - \boldsymbol{G}_A\boldsymbol{\eta}_A) =$$

$$\boldsymbol{Z}_A^T\boldsymbol{Z}_A - \boldsymbol{Z}_A^T\boldsymbol{G}_A\boldsymbol{\eta}_A - \boldsymbol{\eta}_A^T\boldsymbol{G}_A^T\boldsymbol{Z}_A + \boldsymbol{\eta}_A^T\boldsymbol{G}_A^T\boldsymbol{G}_A\boldsymbol{\eta}_A =$$

$$\boldsymbol{Z}^T\boldsymbol{Z} - \boldsymbol{Z}^T\boldsymbol{G\eta} - \boldsymbol{\eta}^T\boldsymbol{G}^T\boldsymbol{Z} + \begin{pmatrix} \boldsymbol{\eta}^T\boldsymbol{G}^T & \sqrt{\lambda_2} & \boldsymbol{\eta}^T \end{pmatrix} \begin{pmatrix} \boldsymbol{G\eta} \\ \sqrt{\lambda_2^*} \ \ \boldsymbol{\eta} \end{pmatrix}.$$

Thus

$$Q_N(\boldsymbol{\eta}_A) = \boldsymbol{Z}^T\boldsymbol{Z} - \boldsymbol{Z}^T\boldsymbol{G\eta} - \boldsymbol{\eta}^T\boldsymbol{G}^T\boldsymbol{Z} + \boldsymbol{\eta}^T\boldsymbol{G}^T\boldsymbol{G\eta} + \lambda_2^*\boldsymbol{\eta}^T\boldsymbol{\eta} + \gamma\|\boldsymbol{\eta}_A\|_1 =$$

$$\|\boldsymbol{Z} - \boldsymbol{G\eta}\|_2^2 + \lambda_2^*\|\boldsymbol{\eta}\|_2^2 + \frac{\lambda_1^*}{\sqrt{1+\lambda_2^*}}\|\boldsymbol{\eta}_A\|_1 =$$

$$RSS(\boldsymbol{\eta}) + \lambda_2^*\|\boldsymbol{\eta}\|_2^2 + \lambda_1^*\|\boldsymbol{\eta}\|_1 = Q(\boldsymbol{\eta}). \ \ \square$$

11.3 Projects

Straightforward Projects

1) Bootstrap OLS and forward selection with C_p as in Table 2.2, but use more values of n, p, k, ψ, and error distributions. See some *R* code for Problem 3.12.

2) Bootstrap OLS and forward selection with BIC in a maaner similar to bootstrapping OLS and forward selection with C_p as in Table 2.2, but use more values of n, p, k, ψ, and error distributions. The *slpack* functions `bicboot` and `bicbootsim` are useful.

3) For a support vector machine (SVM), $Y = 1$ or $Y = -1$. Let $Z = 1$ if $Y = 1$ and $Z = 0$ if $Y = -1$. Let $f(\boldsymbol{x}) = \hat{\boldsymbol{\beta}}_0 + \sum_{i=1}^n \hat{\alpha}_i K(\boldsymbol{x}, \boldsymbol{x}_i) = ESP$. Plot ESP versus Z and add lowess as a visual aid. This treats $Z \| \boldsymbol{x}$ as a binary regression where $\rho(ESP)$ is not specified. Use the prediction region method to bootstrap $\boldsymbol{\beta}$.

4) Analyze a data set with one or more statistical learning methods. The UC Irvine Machine Learning Repository website has interesting data sets. See (http://archive.ics.uci.edu/ml/index.php) and (http://mlearn.ics.uci.edu/MLRepository.html).

Harder Projects

1) Compare the Bickel and Ren (2001) bootstrap confidence region (2.21) with the prediction region method bootstrap confidence region (2.22) on a problem. For example for OLS or forward selection testing $H_0 : \boldsymbol{\beta}_0 = \boldsymbol{0}$.

2) A regression tree can be made for the model $Y = m(\boldsymbol{x}) + e$. Develop a prediction interval for Y_f using (2.7) with $d =$ number of terminal nodes.

3) For multiple linear regression, shrinkage estimators often shrink $\hat{\boldsymbol{\beta}}$ and the ESP too much. See Figure 1.9b for ridge regression. Suppose $Y = \beta_1 + \beta_2 x_2 + \cdots + \beta_{101} x_{101} + e = x_2 + e$ with $n = 100$ and $p = 101$. This model is sparse and lasso performs well, similar to Figure 1.9a. Ridge regression shrinks too much, but $\hat{\boldsymbol{Y}}$ is highly correlated with Y. This suggests regressing $\boldsymbol{Y}$ on $\hat{\boldsymbol{Y}}$ to get $Y = a + b\hat{Y} + \epsilon$. Then $\hat{\boldsymbol{Y}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}_2$ where $\hat{\beta}_{i2} = \hat{b}\hat{\beta}_{iM}$ for $i = 2, ..., p$ and $\hat{\beta}_{i1} = \hat{a} + \hat{b}\hat{\beta}_{iM}$ and M is the shrinkage method such as ridge regression. If $\hat{b} \approx 1$ or if the response plot using shrinkage method M looks good (the plotted points are linear and cover the identity line), then the improvement is not needed.

This technique greatly improves the appearance of the response plot and the prediction intervals on the training data. Investigate whether the technique improves the prediction intervals on test data. Consider automating the procedure by using the improvement if $H_0 : b = 1$ versus $H_1 : b \neq 1$ is rejected, e.g. if 1 is not in the CI $\hat{b} \pm 2SE(\hat{b})$. Some *R* code is shown below.

(It may be possible to improve shrinkage estimators for regression models such as Poisson regression. For Poisson regression, we would want $\exp(\hat{a} + \hat{b}\hat{\boldsymbol{\beta}}_M^T \boldsymbol{x})$ to track Y well.)

```
#Possible way to correct shrinkage estimator
#underfitting.
#The response plot looks much better, but is the idea
#useful for prediction? Usually x1 was x2 in
#the formula Y = 0 + x1 + e.
#The corrected version has ``x1" coef approx 0.48.

library(glmnet)
set.seed(13)
par(mfrow=c(2,1))
x <- matrix(rnorm(10000),nrow=100,ncol=100)
Y <- x[,1] + rnorm(100,sd=0.1)
#sparse model, iid predictors
out <- cv.glmnet(x,Y,alpha=1) #lasso
lam <- out$lambda.min
fit <- predict(out,s=lam,newx=x)
res<- Y-fit
#PI bands used d = 1
AERplot2(yhat=fit,y=Y,res=res)
title("lasso")
cor(fit,Y) #about 0.997
tem <- lsfit(fit,Y)
tem$coef   #changes even if set.seed is used
#   Intercept            1
```

```
#0.0009741988 1.0132965955
out <- cv.glmnet(x,Y,alpha=0) #ridge regression
lam <- out$lambda.min
fit <- predict(out,s=lam,newx=x)
res<- Y-fit
#PI bands used d = 1
AERplot2(yhat=fit,y=Y,res=res)
#$respi
#[1] -1.276461  1.693856  #PI length about  2.97
title("ridge regression")
par(mfrow=c(1,1))
#ridge regression shrank betahat and ESP too much
cor(fit,Y) #about 0.91
tem <- lsfit(fit,Y)
tem$coef
# Intercept          1
#0.3523725   5.8094443   #Fig. 1.9 has -0.7008187  5.7954084
fit2 <- Y-tem$resid
#Y = yhat + r, fit2 = yhat for scaled RR estimator
plot(fit2,Y)  #response plot is much better
abline(0,1)
rrcoef <- predict(out,type="coefficients",s=lam)
plot(rrcoef)
bhat <- tem$coef[2]*rrcoef
bhat[1] <- bhat[1] + tem$coef[1]
#bhat is the betahat for the new ESP fit2
fit3 <- x%*%bhat[-1] + bhat[1]
plot(fit2,fit3)
max(abs(fit2-fit3))
#[1] 1.110223e-15
plot(rrcoef)
plot(bhat)
res2 <- Y - fit2
AERplot2(yhat=fit2,y=Y,res=res2)
$respi
[1] -0.7857706  0.6794579  #PI length about 1.47
title("Response Plot for Scaled Ridge Regression Estimator")
```

Research Ideas That Have Confounded the Author

1) We want clearer and weaker sufficient conditions for when the bootstrap methods work. In particular, we want to weaken sufficient conditions for when the shorth CI and prediction region method confidence region work. See Remark 2.9, Section 2.3.4, Equation (2.2), and the Warning before Example 2.8. Some heuristics for why these bootstrap methods may work for MLR forward selection are given in Sections 2.3.5 and 3.11.

11.4 Tables

Tabled values are F(k,d, 0.95) where $P(F < F(k, d, 0.95)) = 0.95$.
00 stands for ∞. Entries were produced with the `qf(.95,k,d)` command in *R*. The numerator degrees of freedom are k while the denominator degrees of freedom are d.

k	1	2	3	4	5	6	7	8	9	00
d										
1	161	200	216	225	230	234	237	239	241	254
2	18.5	19.0	19.2	19.3	19.3	19.3	19.4	19.4	19.4	19.5
3	10.1	9.55	9.28	9.12	9.01	8.94	8.89	8.85	8.81	8.53
4	7.71	6.94	6.59	6.39	6.26	6.16	6.09	6.04	6.00	5.63
5	6.61	5.79	5.41	5.19	5.05	4.95	4.88	4.82	4.77	4.37
6	5.99	5.14	4.76	4.53	4.39	4.28	4.21	4.15	4.10	3.67
7	5.59	4.74	4.35	4.12	3.97	3.87	3.79	3.73	3.68	3.23
8	5.32	4.46	4.07	3.84	3.69	3.58	3.50	3.44	3.39	2.93
9	5.12	4.26	3.86	3.63	3.48	3.37	3.29	3.23	3.18	2.71
10	4.96	4.10	3.71	3.48	3.33	3.22	3.14	3.07	3.02	2.54
11	4.84	3.98	3.59	3.36	3.20	3.09	3.01	2.95	2.90	2.41
12	4.75	3.89	3.49	3.26	3.11	3.00	2.91	2.85	2.80	2.30
13	4.67	3.81	3.41	3.18	3.03	2.92	2.83	2.77	2.71	2.21
14	4.60	3.74	3.34	3.11	2.96	2.85	2.76	2.70	2.65	2.13
15	4.54	3.68	3.29	3.06	2.90	2.79	2.71	2.64	2.59	2.07
16	4.49	3.63	3.24	3.01	2.85	2.74	2.66	2.59	2.54	2.01
17	4.45	3.59	3.20	2.96	2.81	2.70	2.61	2.55	2.49	1.96
18	4.41	3.55	3.16	2.93	2.77	2.66	2.58	2.51	2.46	1.92
19	4.38	3.52	3.13	2.90	2.74	2.63	2.54	2.48	2.42	1.88
20	4.35	3.49	3.10	2.87	2.71	2.60	2.51	2.45	2.39	1.84
25	4.24	3.39	2.99	2.76	2.60	2.49	2.40	2.34	2.28	1.71
30	4.17	3.32	2.92	2.69	2.53	2.42	2.33	2.27	2.21	1.62
00	3.84	3.00	2.61	2.37	2.21	2.10	2.01	1.94	1.88	1.00

Tabled values are $t_{\alpha,d}$ where $P(t < t_{\alpha,d}) = \alpha$ where t has a t distribution with d degrees of freedom. If $d > 29$ use the $N(0,1)$ cutoffs $d = Z = \infty$.

					alpha					pvalue
d	0.005	0.01	0.025	0.05	0.5	0.95	0.975	0.99	0.995	left tail
1	-63.66	-31.82	-12.71	-6.314	0	6.314	12.71	31.82	63.66	
2	-9.925	-6.965	-4.303	-2.920	0	2.920	4.303	6.965	9.925	
3	-5.841	-4.541	-3.182	-2.353	0	2.353	3.182	4.541	5.841	
4	-4.604	-3.747	-2.776	-2.132	0	2.132	2.776	3.747	4.604	
5	-4.032	-3.365	-2.571	-2.015	0	2.015	2.571	3.365	4.032	
6	-3.707	-3.143	-2.447	-1.943	0	1.943	2.447	3.143	3.707	
7	-3.499	-2.998	-2.365	-1.895	0	1.895	2.365	2.998	3.499	
8	-3.355	-2.896	-2.306	-1.860	0	1.860	2.306	2.896	3.355	
9	-3.250	-2.821	-2.262	-1.833	0	1.833	2.262	2.821	3.250	
10	-3.169	-2.764	-2.228	-1.812	0	1.812	2.228	2.764	3.169	
11	-3.106	-2.718	-2.201	-1.796	0	1.796	2.201	2.718	3.106	
12	-3.055	-2.681	-2.179	-1.782	0	1.782	2.179	2.681	3.055	
13	-3.012	-2.650	-2.160	-1.771	0	1.771	2.160	2.650	3.012	
14	-2.977	-2.624	-2.145	-1.761	0	1.761	2.145	2.624	2.977	
15	-2.947	-2.602	-2.131	-1.753	0	1.753	2.131	2.602	2.947	
16	-2.921	-2.583	-2.120	-1.746	0	1.746	2.120	2.583	2.921	
17	-2.898	-2.567	-2.110	-1.740	0	1.740	2.110	2.567	2.898	
18	-2.878	-2.552	-2.101	-1.734	0	1.734	2.101	2.552	2.878	
19	-2.861	-2.539	-2.093	-1.729	0	1.729	2.093	2.539	2.861	
20	-2.845	-2.528	-2.086	-1.725	0	1.725	2.086	2.528	2.845	
21	-2.831	-2.518	-2.080	-1.721	0	1.721	2.080	2.518	2.831	
22	-2.819	-2.508	-2.074	-1.717	0	1.717	2.074	2.508	2.819	
23	-2.807	-2.500	-2.069	-1.714	0	1.714	2.069	2.500	2.807	
24	-2.797	-2.492	-2.064	-1.711	0	1.711	2.064	2.492	2.797	
25	-2.787	-2.485	-2.060	-1.708	0	1.708	2.060	2.485	2.787	
26	-2.779	-2.479	-2.056	-1.706	0	1.706	2.056	2.479	2.779	
27	-2.771	-2.473	-2.052	-1.703	0	1.703	2.052	2.473	2.771	
28	-2.763	-2.467	-2.048	-1.701	0	1.701	2.048	2.467	2.763	
29	-2.756	-2.462	-2.045	-1.699	0	1.699	2.045	2.462	2.756	
Z	-2.576	-2.326	-1.960	-1.645	0	1.645	1.960	2.326	2.576	
CI						90%	95%		99%	
	0.995	0.99	0.975	0.95	0.5	0.05	0.025	0.01	0.005	right tail
	0.01	0.02	0.05	0.10	1	0.10	0.05	0.02	0.01	two tail