Lesson 6 R Activity

Rick Davila

5/4/2020

Lesson 6 - Install packages

Install necessary packages using library()

```
knitr::opts_chunk$set(echo = TRUE)

library(e1071)
library(xtable)
library("xlsx") # Needed to read data
```

Perform data housekeeping - upload, name columns, display to make sure it reads properly, etc.

```
## Warning: package 'xlsx' was built under R version 4.0.3
```

```
library(car) # Needed for alternative scatterplot matrix to default
```

```
## Loading required package: carData
```

```
library(scatterplot3d) # Needed for 3D scatterplot
```

```
## Warning: package 'scatterplot3d' was built under R version 4.0.3
```

```
library(matlib) # Needed for Invers() function
```

```
## Warning: package 'matlib' was built under R version 4.0.4
```

```
library(MASS) # Needed for ginv() function
library(standardize) # Needed for unit normal scaling in Example 3.14
```

```
## Warning: package 'standardize' was built under R version 4.0.4
```

```
## Registered S3 methods overwritten by 'lme4':
##   method                          from
##   cooks.distance.influence.merMod car
##   influence.merMod                car
##   dfbeta.influence.merMod         car
##   dfbetas.influence.merMod        car
```

```
##
## *************************************************************
##           Loading standardize package version 0.2.2
##      Call standardize.news() to see new features/changes
## *************************************************************
```

```
rm(list = ls())
```

```
ex3_1 <- read.xlsx("data-ex-3-1.xlsx",
                   sheetIndex = 1,
                   colIndex = c(2,3,4),
                   as.data.frame = TRUE,
                   header = TRUE)
```

Read data file (data-ex-3-1.xlsx)

```
names(ex3_1) <- c("Delivery_Time", "Num_Cases", "Distance")
attach(ex3_1)
```

Assign labels to data columns using names() and attach() commands

```
out <- as.data.frame(c(ex3_1))
colnames(out) <- c("Delivery Time, $y$ (min)", "Number of Cases, $x_i$","Distance, $x_2$ (ft)")
tab <- (xtable(out, digits=c(0,2,0,0)))
print(tab, type="html")
```

Output data to make sure it reads properly Delivery Time, y (min)

Number of Cases, x_i

Distance, x_2 (ft)

1

16.68

7

560

2

11.50

3

220

3

12.03

3

340

4

14.88

4

80

5

13.75

6

150

6

18.11

7

330

7

8.00

2

110

8

17.83

7

210

9

79.24

30

1460

10

21.50

5

605

11

40.33

16

688

12

21.00

10

215

13

13.50

4

255

14

19.75

6

462

15

24.00

9

448

16

29.00

10

776

17

15.35

6

200

18

19.00

7

132

19

9.50

3

36

20

35.10

17

770

21

17.90

10

140

22

52.32

26

810

23

18.75

9

450

24

19.83

8

635

25

10.75

4

150

```
# Output data structure and dimensions
str(ex3_1)
```

'data.frame': 25 obs. of 3 variables: $ Delivery_Time: num 16.7 11.5 12 14.9 13.8 … $ Num_Cases : num 7 3 3 4 6 7 2 7 30 5 … $ Distance : num 560 220 340 80 150 330 110 210 1460 605 …

```
dim(ex3_1)
```

[1] 25 3

Lesson 6 - Example 4.1 (p.135-136)

```


# Define X matrix of regressor observations
X <- cbind(matrix(1,length(Distance),1),as.matrix(Num_Cases),as.matrix(Distance))
y <- as.matrix(Delivery_Time)

# X'X matrix
xTx <- t(X) %*% X

# Calculate least-squares estimator of beta_coeffs
beta_hat <- ginv(xTx) %*% t(X) %*% y
beta_0 <- beta_hat[1,1]
beta_1 <- beta_hat[2,1]
beta_2 <- beta_hat[3,1]
```

Calculate residuals and compare to Column (1) in Table 4.1, p. 137. The least squares fit is

$$\hat{y} = (2.3412311) + (1.6159072)x_1 + (0.0143848)x_2$$

```
# least squares fit equation
y_hat <- beta_0 + beta_1*Num_Cases + beta_2*Distance
e_i=data.frame(Delivery_Time - y_hat) # e_i are the residuals
names(e_i) <- c("Res")
attach(e_i)

out <- as.data.frame(c(e_i))
colnames(out) <- c("Residuals")
tab <- (xtable(out, digits=c(0,6)))
print(tab, type="html")
```

Residuals

1

-5.028084

2

1.146385

3

-0.049794

4

4.924354

5

-0.444398

6

-0.289574

7

0.844624

8

1.156605

9

7.419706

10

2.376413

11

2.237493

12

-0.593041

13

1.027009

14

1.067536

15

0.671202

16

-0.662928

17

0.436360

18

3.448621

19

1.793193

20

-5.787970

21

-2.614179

22

-3.686528

23

-4.607568

24

-4.572854

25

-0.212584

```
n <- length(Res)       # number of residuals
ei_bar <- sum(Res)/n   # avg of residuals
k <- 2                 # number of beta regressor coefficients
p <- k + 1             # number of parameters

MS_Res <- sum((Res - ei_bar)^2)/(n-p)

# standardized residuals (eqn 4.2)
d_i <- data.frame(Res/(sqrt(MS_Res)))
names(d_i) <- c("Std_Res")
attach(d_i)

out <- as.data.frame(c(d_i))
```

```r
colnames(out) <- c("Standardized Residuals")
tab <- (xtable(out, digits=c(0,6)))
print(tab, type="html")
```

Calculate standardized residuals using Equation 4.2;compare to Column (2) in Table 4.1, p. 137. Standardized Residuals

1

-1.542606

2

0.351709

3

-0.015277

4

1.510782

5

-0.136341

6

-0.088841

7

0.259129

8

0.354844

9

2.276351

10

0.729079

11

0.686458

12

-0.181944

13

0.315084

14

0.327518

15

0.205923

16

-0.203385

17

0.133874

18

1.058030

19

0.550148

20

-1.775738

21

-0.802025

22

-1.131019

23

-1.413593

24

-1.402942

25

-0.065220

```
# calculate the hat matrix
H <- X %*% ginv(xTx) %*% t(X)

# to get the diagonal elements of H, use diag(H) function

# studentized residuals (eqn 4.8)
r_i <- data.frame(Res/(sqrt(MS_Res*(1-diag(H)))))
names(r_i) <- c("Stud_Res")
attach(r_i)

out <- as.data.frame(c(r_i))
colnames(out) <- c("Studentized Residuals")
tab <- (xtable(out, digits=c(0,6)))
print(tab, type="html")
```

Calculate studentized residuals using Equation 4.8; compare to Column (3) in Table 4.1, p. 137.
Studentized Residuals

1

-1.627680

2

0.364843

3

-0.016092

4

1.579720

5

-0.141761

6

-0.090808

7

0.270425

8

0.366721

9

3.213763

10

0.813254

11

0.718080

12

-0.193257

13

0.325179

14

0.341135

15

0.210291

16

-0.222700

17

0.138039

18

1.112952

19

0.578766

20

-1.873546

21

-0.877843

22

-1.449995

23

-1.443690

24

-1.496059

25

-0.067509

```r
# PRESS residuals (eqn 4.11)
e_pip <- data.frame(Res/(1-diag(H)))
names(e_pip) <- c("PRESS_Res")
attach(e_pip)

out <- as.data.frame(c(e_pip))
colnames(out) <- c("PRESS residuals")
tab <- (xtable(out, digits=c(0,6)))
print(tab, type="html")
```

Calculate PRESS residuals using Equation 4.11; compare to Column (5) in Table 4.1, p. 137.
PRESS residuals

1

-5.597967

2

1.233603

3

-0.055249

4

5.384013

5

-0.480436

6

-0.302543

7

0.919867

8

1.235327

9

14.788898

10

2.956826

11

2.448378

12

-0.669086

13

1.093872

14

1.158154

15

0.699978

16

-0.794821

17

0.463933

18

3.815946

19

1.984606

20

-6.443140

21

-3.131792

22

-6.059135

23

-4.805858

24

-5.200019

25

-0.227763

```r
# using eqn 4.12
Si_sqr = ((n-p)*MS_Res-(Res^2/(1-diag(H))))/(n-p-1)

# R-student (eqn 4.13)
t_i <- data.frame(Res/(sqrt(Si_sqr*(1-diag(H)))))
names(t_i) <- c("R_Stud")
attach(t_i)

out <- as.data.frame(c(t_i))
colnames(out) <- c("R-Student Residuals")
tab <- (xtable(out, digits=c(0,6)))
print(tab, type="html")
```

Calculate R-Student residuals using Equations 4.12 and 4.13; compare to Column (6) in Table 4.1, p. 137. Looks good. R-Student Residuals

1

-1.695629

2

0.357538

3

-0.015722

4

1.639165

5

-0.138565

6

-0.088737

7

0.264648

8

0.359390

9

4.310780

10

0.806776

11

0.709939

12

-0.188975

13

0.318469

14

0.334177

15

0.205663

16

-0.217826

17

0.134924

18

1.119331

19

0.569814

20

-1.996677

21

-0.873087

22

-1.489625

23

-1.482467

24

-1.542215

25

-0.065963

```
out <- as.data.frame(c(e_i, d_i, r_i, e_pip, t_i))
colnames(out) <- c(" Residuals, $e_i$ "," Standardized Residuals, $d_i$ "," Studentized Residuals, $r_i
tab <- (xtable(out, digits=c(0, 6, 6, 6, 6, 6)))
print(tab, type="html")
```

Reproduce Table 4.1 on p. 137 - Consolidate above vectors into a dataframe using data.frame(). Give column names using colnames() and c(). Residuals, e_i

Standardized Residuals, d_i

Studentized Residuals, r_i

PRESS Residuals, $e_{(i)}$

R-student, t_i

1

-5.028084

-1.542606

-1.627680

-5.597967

-1.695629

2

1.146385

0.351709

0.364843

1.233603

0.357538

3

-0.049794

-0.015277

-0.016092

-0.055249

-0.015722

4

4.924354

1.510782

1.579720

5.384013

1.639165

5

-0.444398

-0.136341

-0.141761

-0.480436

-0.138565

6

-0.289574

-0.088841

-0.090808

-0.302543

-0.088737

7

0.844624

0.259129

0.270425

0.919867

0.264648

8

1.156605

0.354844

0.366721

1.235327

0.359390

9

7.419706

2.276351

3.213763

14.788898

4.310780

10

2.376413

0.729079

0.813254

2.956826

0.806776

11

2.237493

0.686458

0.718080

2.448378

0.709939

12

-0.593041

-0.181944

-0.193257

-0.669086

-0.188975

13

1.027009

0.315084

0.325179

1.093872

0.318469

14

1.067536

0.327518

0.341135

1.158154

0.334177

15

0.671202

0.205923

0.210291

0.699978

0.205663

16

-0.662928

-0.203385

-0.222700

-0.794821

-0.217826

17

0.436360

0.133874

0.138039

0.463933

0.134924

18

3.448621

1.058030

1.112952

3.815946

1.119331

19

1.793193

0.550148

0.578766

1.984606

0.569814

20

-5.787970

-1.775738

-1.873546

-6.443140

-1.996677

21

-2.614179

-0.802025

-0.877843

-3.131792

-0.873087

22

-3.686528

-1.131019

-1.449995

-6.059135

-1.489625

23

-4.607568

-1.413593

-1.443690

-4.805858

-1.482467

24

-4.572854

-1.402942

-1.496059

-5.200019

-1.542215

25

-0.212584

-0.065220

-0.067509

-0.227763

-0.065963

```r
# test rstandard() function
test.lm <- lm(Delivery_Time~ Num_Cases + Distance)
test.rstandard <- data.frame(rstandard(test.lm))
test.rstudent <- data.frame(rstudent(test.lm))

out <- as.data.frame(c(e_i, d_i, r_i, e_pip, t_i, test.rstandard, test.rstudent))
colnames(out) <- c(" Residuals, $e_i$ "," Standardized Residuals, $d_i$ "," Studentized Residuals, $r_i$
tab <- (xtable(out, digits=c(0, 6, 6, 6, 6, 6, 6, 6)))
print(tab, type="html")
```

Note: R has functions called rstandard() and rstudent(). Note that rstandard() outputs what the textbook calls Studentized Residuals and rstudent() outputs what the textbook calls R-Student residuals or 'externally studentized' residuals. Don't get confused. Compare your manually-computer vectors above with the rstandard() and rstudent() functions. The rounded difference should be a vector of zeros. Residuals, e_i

Standardized Residuals, d_i

Studentized Residuals, r_i

PRESS Residuals, $e_{(i)}$

R-student, t_i

rstandard()

rstudent()

1

-5.028084

-1.542606

-1.627680

-5.597967

-1.695629

-1.627680

-1.695629

2

1.146385

0.351709

0.364843

1.233603

0.357538

0.364843

0.357538

3

-0.049794

-0.015277

-0.016092

-0.055249

-0.015722

-0.016092

-0.015722

4

4.924354

1.510782

1.579720

5.384013

1.639165

1.579720

1.639165

5

-0.444398

-0.136341

-0.141761

-0.480436

-0.138565

-0.141761

-0.138565

6

-0.289574

-0.088841

-0.090808

-0.302543

-0.088737

-0.090808

-0.088737

7

0.844624

0.259129

0.270425

0.919867

0.264648

0.270425

0.264648

8

1.156605

0.354844

0.366721

1.235327

0.359390

0.366721

0.359390

9

7.419706

2.276351

3.213763

14.788898

4.310780

3.213763

4.310780

10

2.376413

0.729079

0.813254

2.956826

0.806776

0.813254

0.806776

11

2.237493

0.686458

0.718080

2.448378

0.709939

0.718080

0.709939

12

-0.593041

-0.181944

-0.193257

-0.669086

-0.188975

-0.193257

-0.188975

13

1.027009

0.315084

0.325179

1.093872

0.318469

0.325179

0.318469

14

1.067536

0.327518

0.341135

1.158154

0.334177

0.341135

0.334177

15

0.671202

0.205923

0.210291

0.699978

0.205663

0.210291

0.205663

16

-0.662928

-0.203385

-0.222700

-0.794821

-0.217826

-0.222700

-0.217826

17

0.436360

0.133874

0.138039

0.463933

0.134924

0.138039

0.134924

18

3.448621

1.058030

1.112952

3.815946

1.119331

1.112952

1.119331

19

1.793193

0.550148

0.578766

1.984606

0.569814

0.578766

0.569814

20

-5.787970

-1.775738

-1.873546

-6.443140

-1.996677

-1.873546

-1.996677

21

-2.614179

-0.802025

-0.877843

-3.131792

-0.873087

-0.877843

-0.873087

22

-3.686528

-1.131019

-1.449995

-6.059135

-1.489625

-1.449995

-1.489625

23

-4.607568

-1.413593

-1.443690

-4.805858

-1.482467

-1.443690

-1.482467

24

-4.572854

-1.402942

-1.496059

-5.200019

-1.542215

-1.496059

-1.542215

25

-0.212584

-0.065220

-0.067509

-0.227763

-0.065963

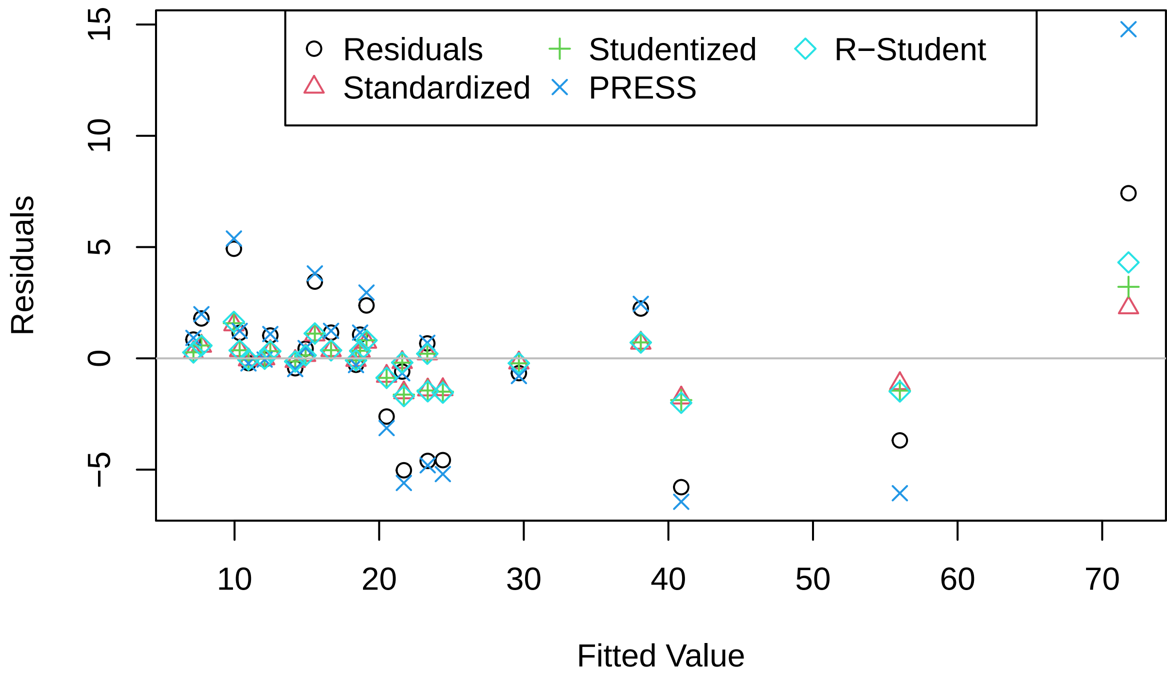
-0.067509

-0.065963

Lesson 6 - Example 4.2 (p. 139-140)

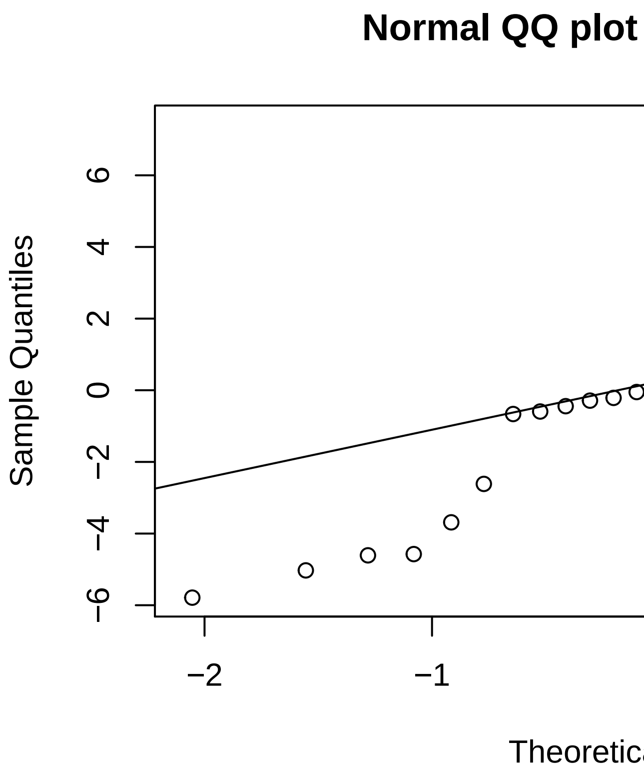
```
# before plotting, get colors
cols <- palette()

# plot the first curve by calling plot() function
plot(y_hat, Res, ylab = "Residuals", xlab = "Fitted Value", col=cols[1], pch=1, ylim=range(Res, Std_Res
points(y_hat, Std_Res, col=cols[2], pch=2)
points(y_hat, Stud_Res, col=cols[3], pch=3)
points(y_hat, PRESS_Res, col=cols[4], pch=4)
points(y_hat, R_Stud, col=cols[5], pch=5)

abline(h=0, col="gray")

legend("top",legend=c("Residuals","Standardized","Studentized","PRESS","R-Student"), col=c(cols[1],cols
```

Create plot of residuals against fitted y__hat values for each type of residual calculated in Exam-



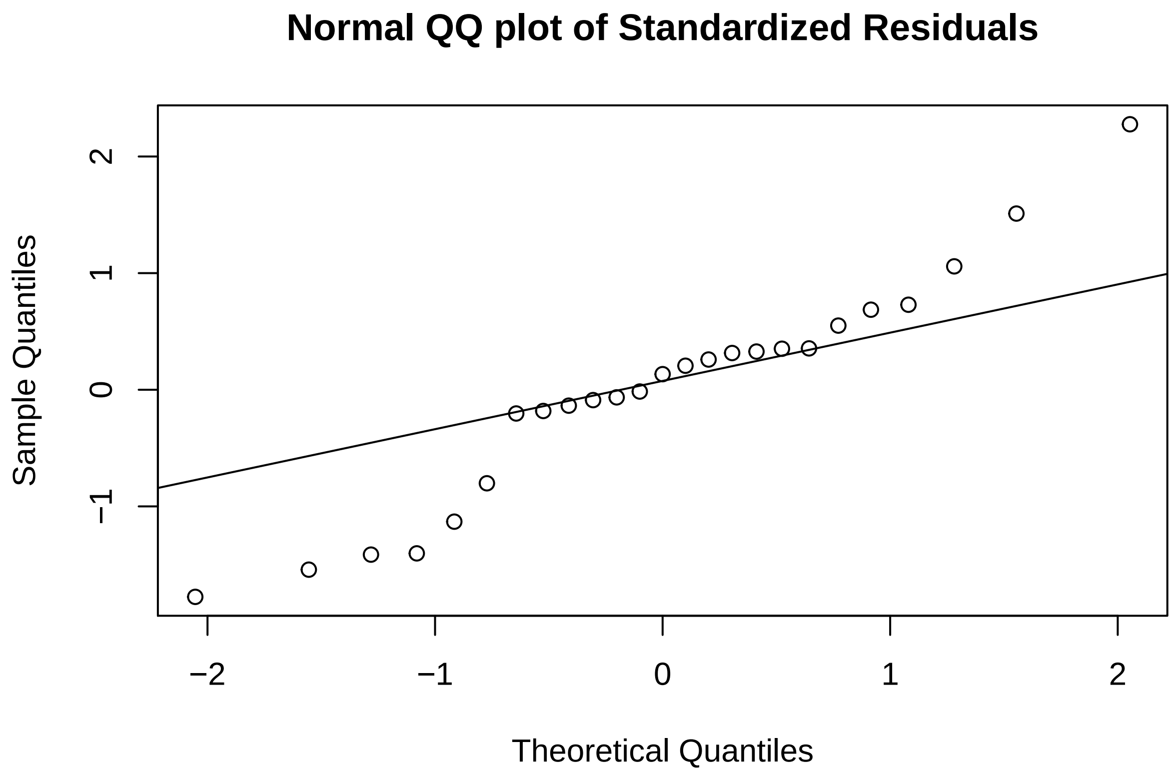
ple 4.1 using plot()

```
# the normal probability plot is a graphical technique for assessing whether or not a data set is appro

qqnorm(Res,main="Normal QQ plot of Residuals (e_i)")
qqline(Res)
```

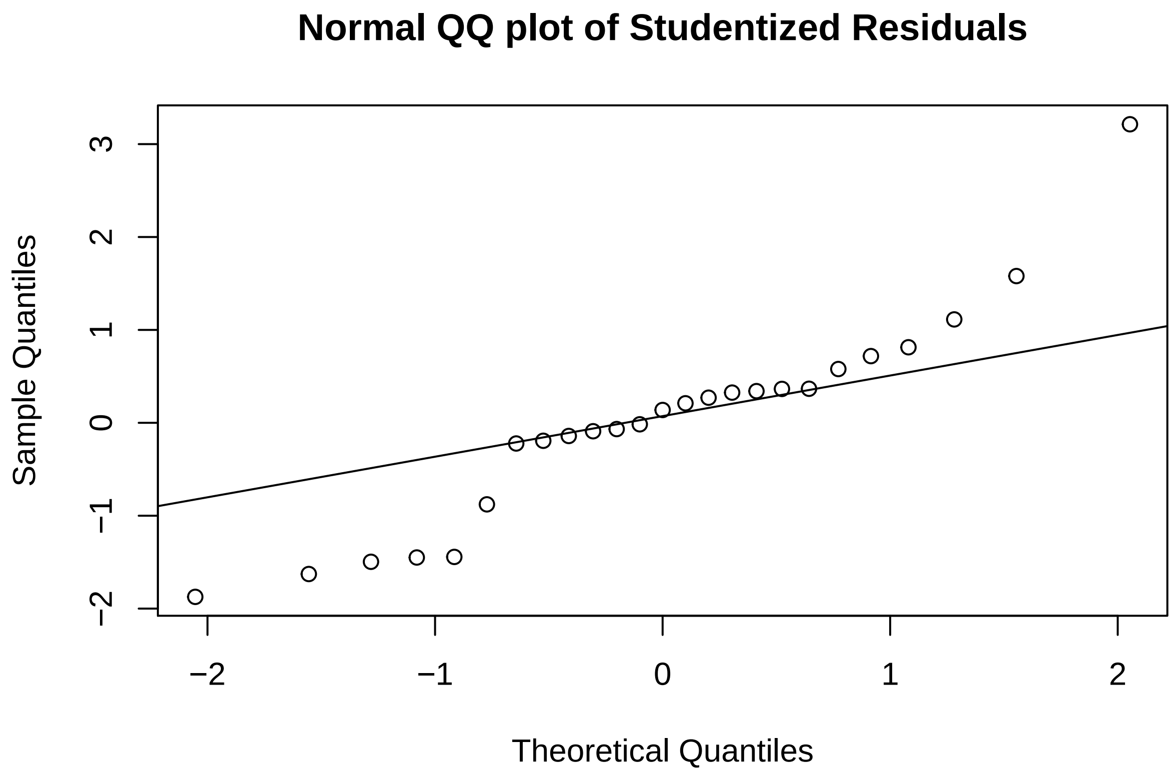


Create normal probability plot of residuals using qqnorm()

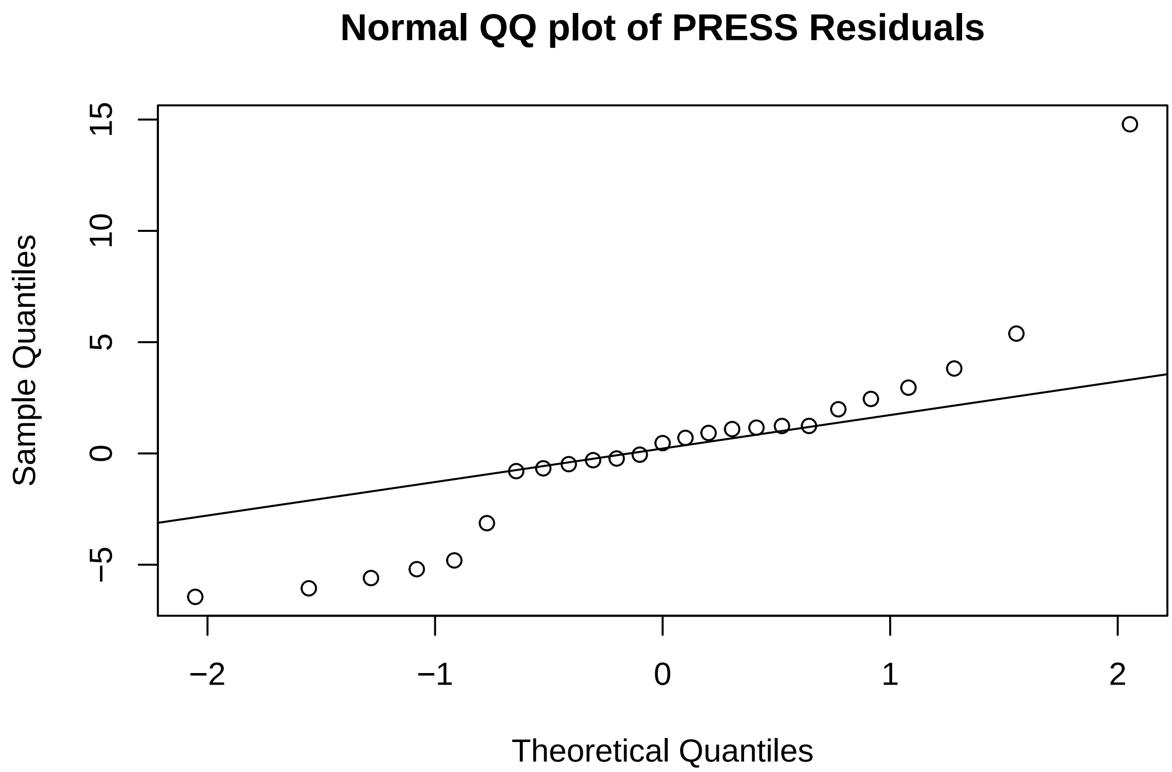
```
qqnorm(Std_Res,main="Normal QQ plot of Standardized Residuals")
qqline(Std_Res)
```



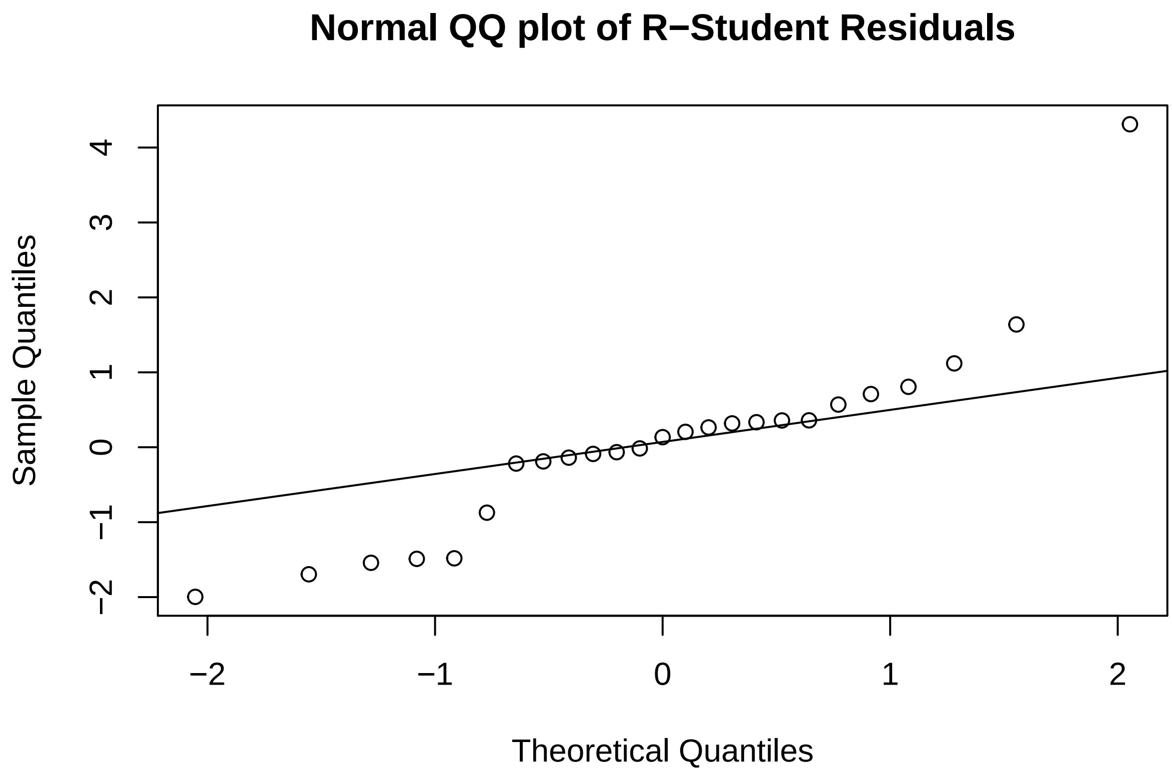

```
qqnorm(Stud_Res,main="Normal QQ plot of Studentized Residuals")
qqline(Stud_Res)
```




```
qqnorm(PRESS_Res,main="Normal QQ plot of PRESS Residuals")
qqline(PRESS_Res)
```



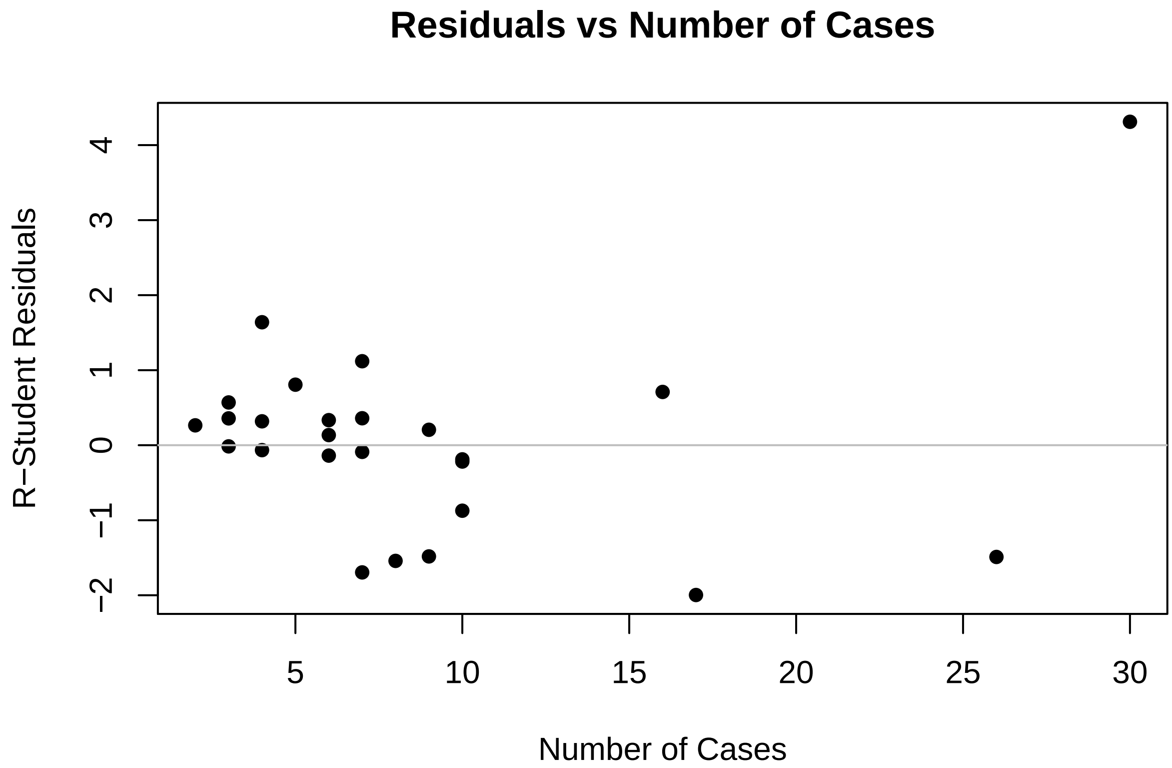

```
qqnorm(R_Stud,main="Normal QQ plot of R-Student Residuals")
qqline(R_Stud)
```



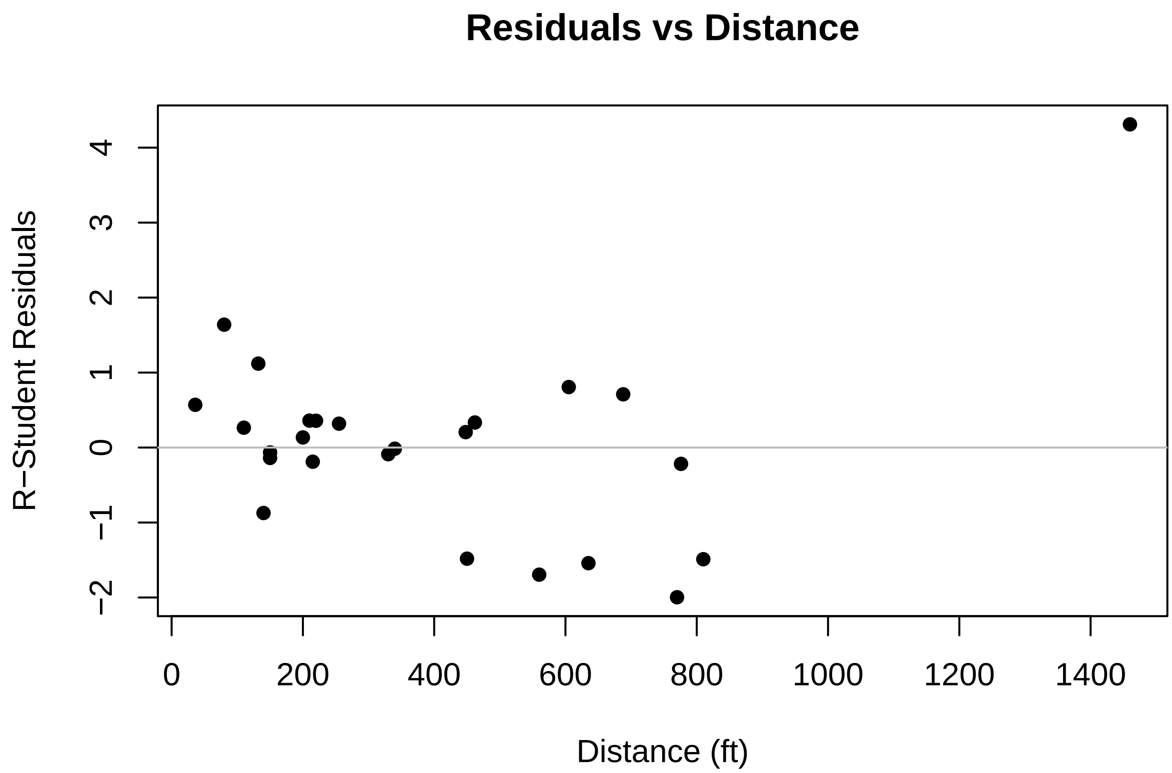
Lesson 6 - Example 4.3 (p. 141)

```
# plot number of cases versus R-student residuals
plot(Num_Cases, R_Stud, ylab = "R-Student Residuals", xlab = "Number of Cases", main = "Residuals vs Nu
abline(h=0, col="gray")
```

Create a plot of residuals against the each of the regressors (Use R-Student residuals)



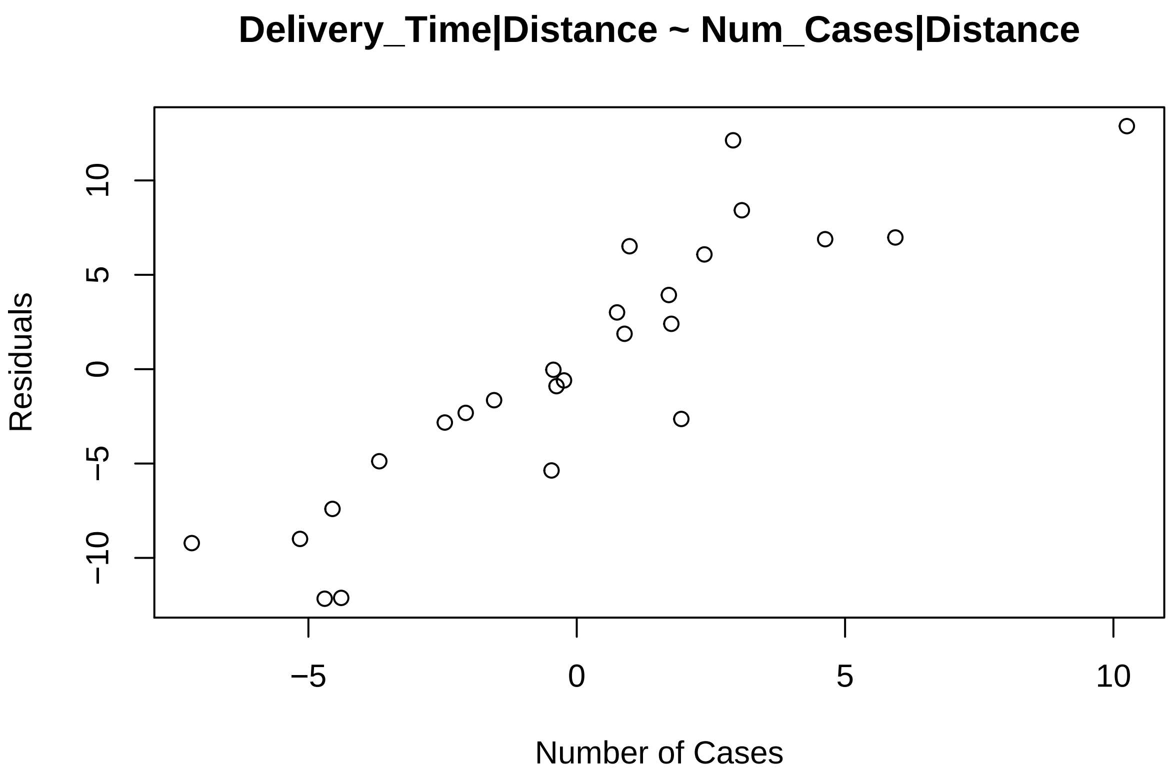

```
# plot Distance versus R-student residuals
plot(Distance, R_Stud, ylab = "R-Student Residuals", xlab = "Distance (ft)", main = "Residuals vs Distan
abline(h=0, col="gray")
```



Lesson 6 - Example 4.5 (p. 144)

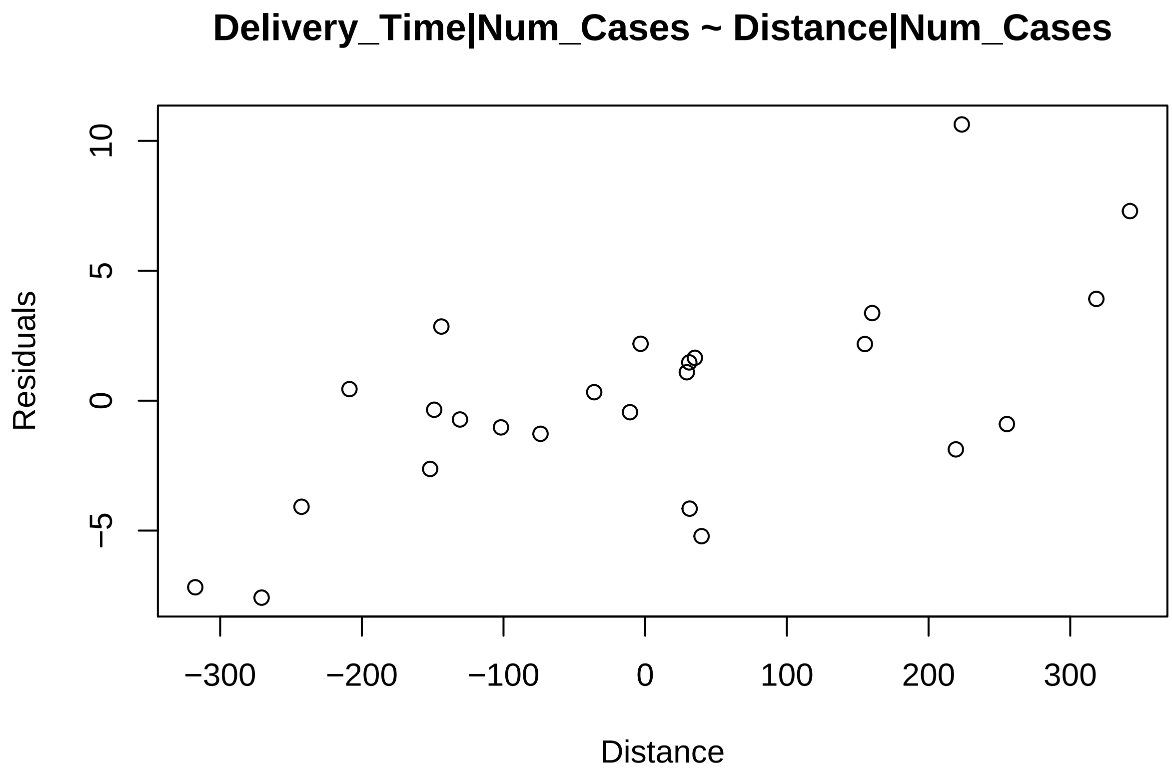
```
# Delivery_Time, given Distance
model1 <- lm(Delivery_Time ~ Distance )
residuals.1 <- model1$residuals

# Num_Cases, given Distance
model2 <- lm(Num_Cases ~ Distance)
residuals.2 <- model2$residuals

# Delivery_Time, given x1
model3 <- lm(Delivery_Time ~ Num_Cases)
residuals.3 <- model3$residuals

# Distance, given Num_Cases
model4 <- lm(Distance ~ Num_Cases)
residuals.4 <- model4$residuals

plot(residuals.2, residuals.1, main='Delivery_Time|Distance ~ Num_Cases|Distance', xlab = "Number of Ca
```

Create partial regression plots for each of the two regressors. Use rstudent() to calculate resid-



uals.

```
plot(residuals.4, residuals.3, main='Delivery_Time|Num_Cases ~ Distance|Num_Cases', xlab = "Distance", y
```



Lesson 6 - Example 4.6 (p.151-152)

```
# PRESS statistic using eqn 4.17
PRESS_statistic <- sum((Res/(1-diag(H)))^2)

# sum of squares total
SS_T <- t(Delivery_Time)%*%Delivery_Time-(sum(Delivery_Time))^2/length(Delivery_Time)

# R-Squared PRESS using eqn 4.18
R_Squared_PRESS <- 1 - PRESS_statistic/SS_T
```

Calculate PRESS Statistic and R-Squared_PRESS (by hand) The PRESS Statistic is 459.0393147 and the R^2 for Prediction Based on PRESS is $R^2_{prediction} = 0.9206438$