Lesson 12: Simple Regression Analysis and Correlation

References

- Black, Chapter 12 Simple Regression Analysis and Correlation (pp. 424-448)
- Kabakoff, Chapter 8 Regression (pp. 167-175)
- Davies, Chapter 20 Simple Linear Regression (pp. 451-460)

Data set: newspapers.csv

Description: The data are from the Gale Directory of Publications, 1994. A sample of 34 newspapers are listed along with their Daily and Sunday circulations (in thousands).

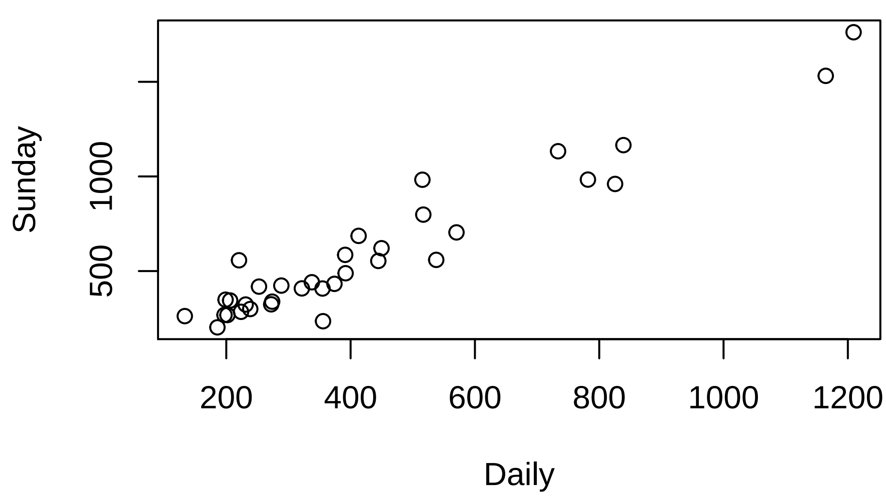
Exercises:

1) Plot Sunday circulation versus Daily circulation. Does the scatter plot suggest a linear relationship between the two variables? Calculate the Pearson Product Moment Correlation Coefficient between Sunday and Daily circulation.

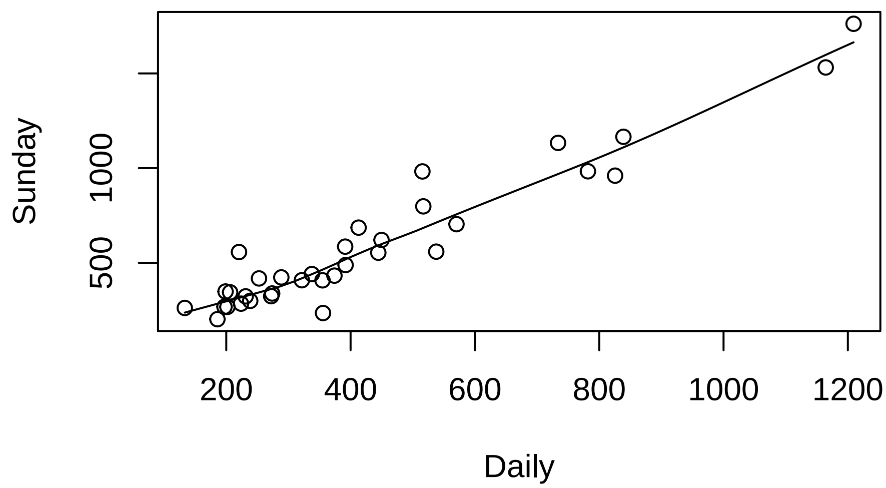
```
# Read the comma-delimited text file creating a data frame object in R,
# then examine its structure:

newspapers <- read.csv("newspapers.csv")
str(newspapers)
```

```
## 'data.frame':    34 obs. of  3 variables:
##  $ Newspaper: Factor w/ 34 levels "Baltimore Sun",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Daily    : num  392 517 356 239 538 ...
##  $ Sunday   : num  489 798 235 299 559 ...
```

```
with(newspapers, plot(Daily, Sunday))
```



```r
with(newspapers, scatter.smooth(Daily, Sunday))  # smooth line looks straight
```




```r
with(newspapers, print(cor(Daily, Sunday)))  # strong positive correlation
```

```
## [1] 0.9581543
```

2) Fit a regression line with Sunday circulation as the dependent variable. Plot the regression line with the circulation data. (Use Lander pages 212 and 213 for reference.) Comment on the quality of the fit. What percent of the total variation in Sunday circulation is accounted for by the regression line?

```r
my_model <- lm(Sunday ~ Daily, data = newspapers)
my_model
```

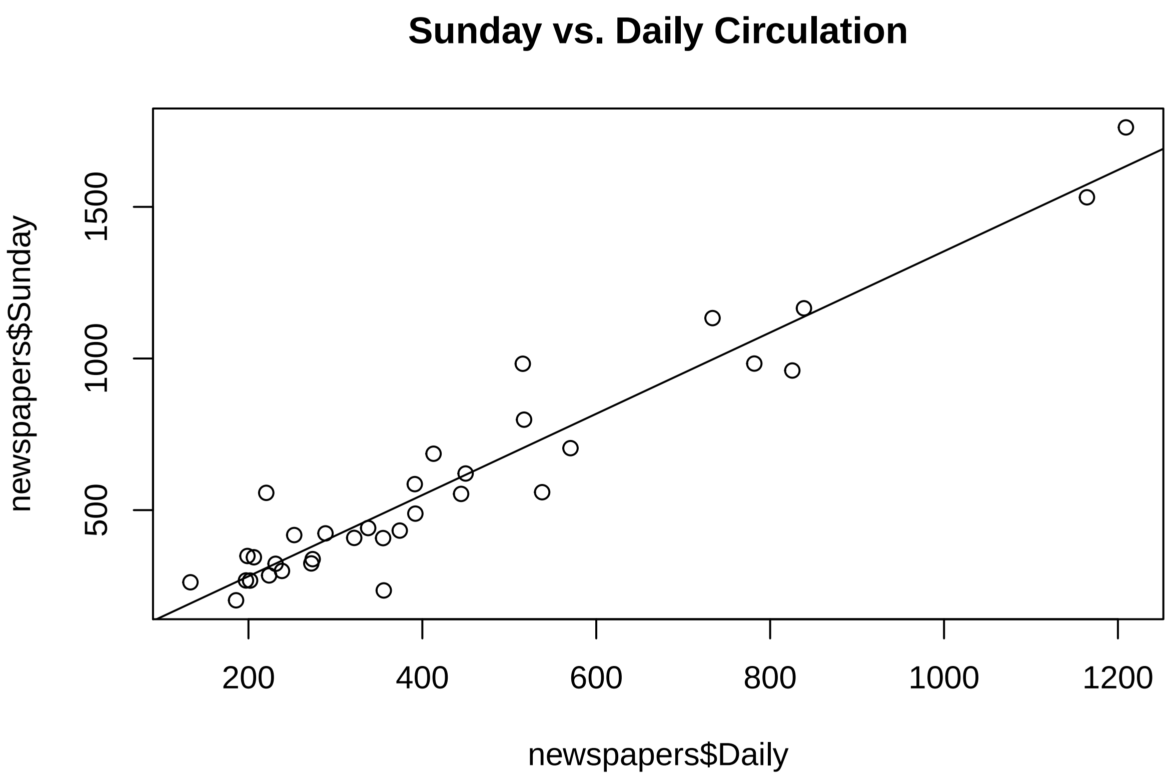
```
##
## Call:
## lm(formula = Sunday ~ Daily, data = newspapers)
##
## Coefficients:
## (Intercept)        Daily
##       13.84         1.34
```

```r
summary(my_model)
```

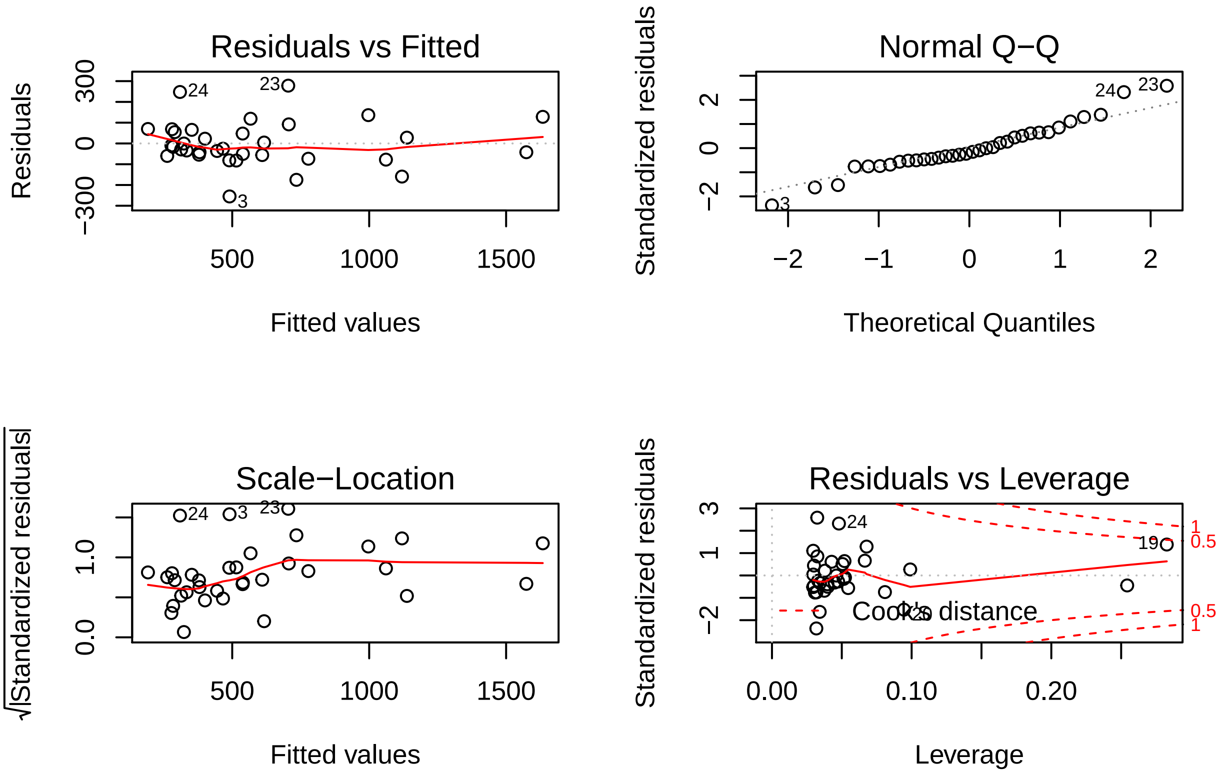
```
##
## Call:
## lm(formula = Sunday ~ Daily, data = newspapers)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -255.19  -55.57  -20.89   62.73  278.17
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 13.83563   35.80401   0.386    0.702
```

```
## Daily        1.33971    0.07075  18.935   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 109.4 on 32 degrees of freedom
## Multiple R-squared:  0.9181, Adjusted R-squared:  0.9155
## F-statistic: 358.5 on 1 and 32 DF,  p-value: < 2.2e-16
```

```r
plot(newspapers$Daily, newspapers$Sunday, main = "Sunday vs. Daily Circulation")
abline(my_model)
```




```r
# In addition to plotting our fitted line, we should observe the behavior of the residuals.
par(mfrow=c(2,2))
plot(my_model)
```




```r
par(mfrow=c(1,1))

# A linear model fits these data very well.
# R-squared:  0.9181 = proportion of Sunday circulation variance accounted for.
```

3) Obtain 95% confidence intervals for the coefficients in the regression model. Use confint().

```r
confint(my_model, level = 0.95)
```

```
##                  2.5 %    97.5 %
## (Intercept) -59.094743 86.766003
## Daily         1.195594  1.483836
```

4) Determine a 95% prediction interval to predict Sunday circulation for all available values of Daily circulation. Use predict(model, interval="prediction", level=0.95). Then, define a new data frame using Daily = 500 and Sunday = NA. Predict an interval for Sunday circulation.

```r
# Use predict(model, interval = "prediction", level = 0.95):
predict(my_model, interval = "prediction", level = 0.95)
```

```
## Warning in predict.lm(my_model, interval = "prediction", level = 0.95): predictions on current data
```

```
##         fit       lwr       upr
## 1  538.9395 312.73212  765.1469
## 2  706.4427 479.96564  932.9198
## 3  490.2757 263.87771  716.6737
## 4  333.4313 105.59993  561.2626
## 5  734.3074 507.64652  960.9683
## 6  996.8848 766.57467 1227.1950
```

```
## 7   280.2138   51.61501  508.8126
## 8   352.2797  124.68627  579.8732
## 9   290.0902   61.64446  518.5359
## 10  323.5469   95.58365  551.5101
## 11  616.3790  390.22531  842.5328
## 12  400.4385  173.37170  627.5052
## 13  262.6689   33.78627  491.5515
## 14 1573.7834 1324.16150 1823.4053
## 15  609.4474  383.30133  835.5934
## 16  566.9650  340.81246  793.1175
## 17  378.6132  151.32219  605.9041
## 18 1061.2193  829.49801 1292.9405
## 19 1633.8522 1381.42609 1886.2783
## 20 1119.7862  886.60913 1352.9633
## 21  313.5941   85.49321  541.6950
## 22  489.2240  262.82058  715.6275
## 23  704.4894  478.02374  930.9551
## 24  309.1958   81.03249  537.3592
## 25  466.2198  239.68294  692.7566
## 26  277.9202   49.28518  506.5552
## 27  192.3379  -37.83442  422.5102
## 28  514.9010  288.61458  741.1874
## 29  380.7085  153.44007  607.9769
## 30  777.9607  550.93248 1004.9889
## 31  538.0473  311.83746  764.2571
## 32  284.2705   55.73513  512.8058
## 33  444.7227  218.03687  671.4086
## 34 1137.7250  904.06982 1371.3802
```

```
# This gives the prediction intervals for all observations.

# Then, define a new data frame using Daily = 500 and Sunday = NA. Predict an interval
# for Sunday circulation.

# To get the prediction interval for a new observation with daily circulation
# of 500 thousand, say, we could set up a new data frame with that value.

Daily <- 500
Sunday <- NA
new_data_frame <- data.frame(Daily, Sunday)
predict(my_model, newdata=new_data_frame, interval="prediction", level=0.95)
```

```
##       fit      lwr      upr
## 1 683.693 457.3367 910.0493
```

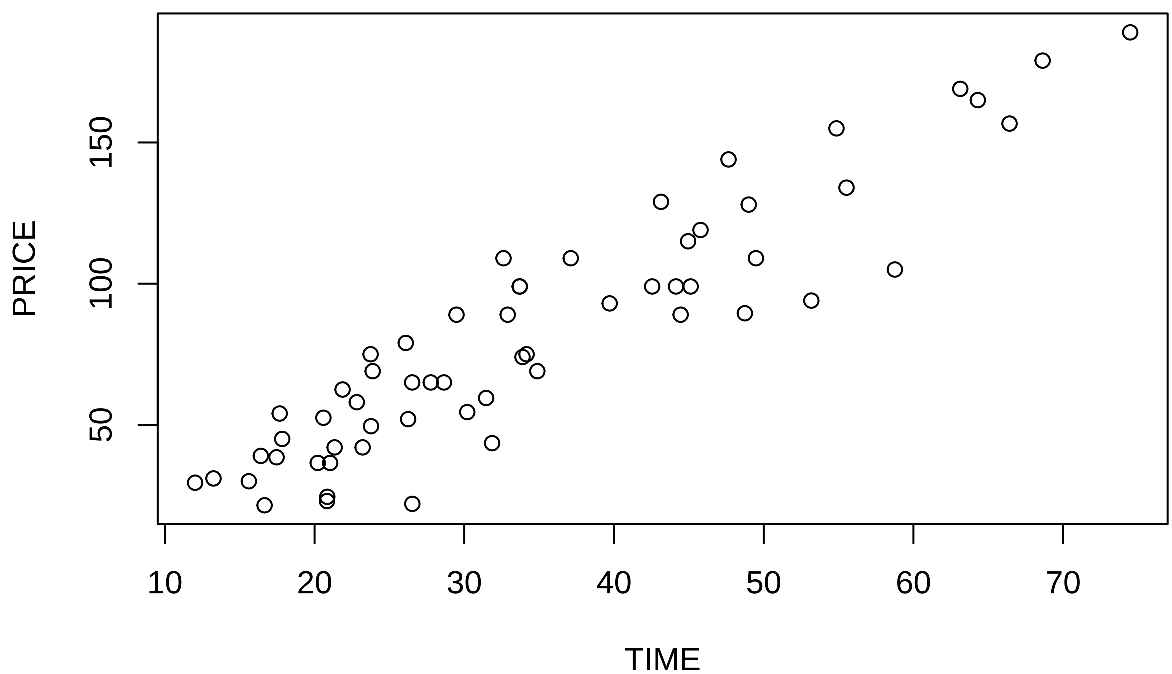
5) Use the tableware.csv data. Regress PRICE as a dependent variable against TIME. Comment on the quality of the fit. Is a simple linear regression model adequate or is something more needed?

```
# Read the comma-delimited text file creating a data frame object in R,
# then examine its structure:

tableware <- read.csv("tableware.csv")
str(tableware)
```

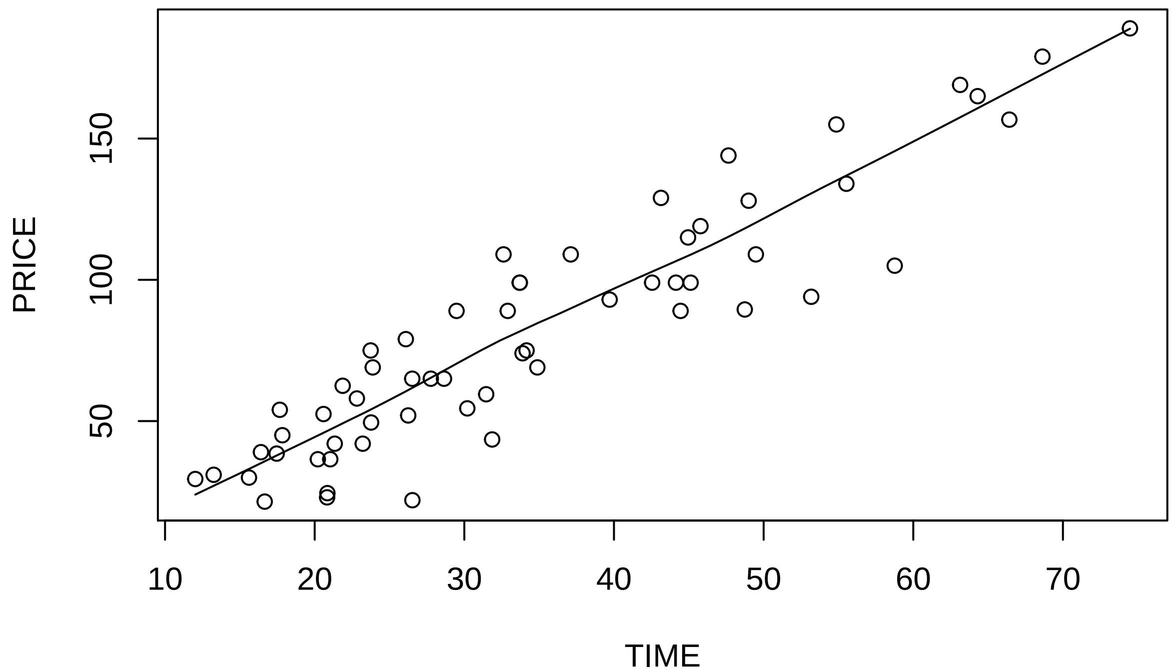
```
## 'data.frame':    59 obs. of  9 variables:
##  $ TYPE : Factor w/ 5 levels "bowl","cass",..: 2 2 2 1 3 2 5 5 3 3 ...
```

```
##  $ BOWL : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ CASS : int  1 1 1 0 0 1 0 0 0 0 ...
##  $ DISH : int  0 0 0 0 1 0 0 0 1 1 ...
##  $ TRAY : int  0 0 0 0 0 0 1 1 0 0 ...
##  $ DIAM : num  10.7 14 9 8 10 10.5 16 15 6.5 5 ...
##  $ TIME : num  47.6 63.1 58.8 34.9 55.5 ...
##  $ PRICE: num  144 169 105 69 134 129 155 99 38.5 36.5 ...
##  $ RATE : num  3.02 2.68 1.79 1.98 2.41 2.99 2.83 2.24 2.21 1.73 ...
```

```r
# Let's start with a couple plots.
with(tableware, plot(TIME, PRICE))
```




```r
with(tableware, scatter.smooth(TIME, PRICE))  # smooth line looks straight
```




```r
with(tableware, print(cor(TIME, PRICE)))  # strong positive correlation
```

```
## [1] 0.9224022
```

```r
my_model <- lm(PRICE ~ TIME, data = tableware)
my_model
```

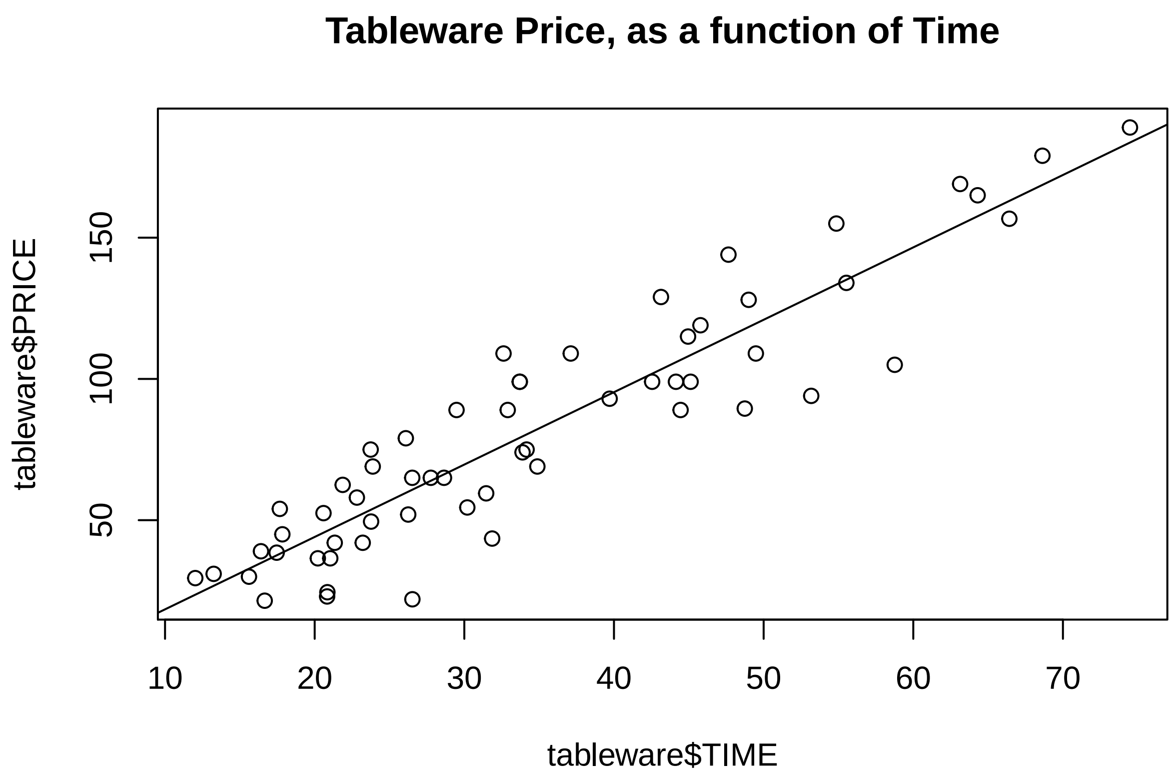
```
##
## Call:
## lm(formula = PRICE ~ TIME, data = tableware)
##
## Coefficients:
## (Intercept)         TIME
##      -7.189        2.562
```

```r
summary(my_model)
```

```
##
## Call:
## lm(formula = PRICE ~ TIME, data = tableware)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.794  -9.828   0.948  11.104  32.601
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -7.1891     5.4053   -1.33    0.189
```

```
## TIME          2.5625     0.1421   18.03   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.77 on 57 degrees of freedom
## Multiple R-squared:  0.8508, Adjusted R-squared:  0.8482
## F-statistic: 325.1 on 1 and 57 DF,  p-value: < 2.2e-16
```

```r
plot(tableware$TIME, tableware$PRICE, main = "Tableware Price, as a function of Time")
abline(my_model)
```




```r
# A linear model fits these data very well.
# R-squared:  0.8508 = proportion of PRICE variance accounted for by the model.
# We may be able to do better by adding explanatory variables,
# but this is a good start.
```

6) Use the tableware.csv data. ANOVA can be accomplished using a regression model. Regress PRICE against the variables BOWL, CASS, DISH and TRAY as they are presented in the data file. What do the coefficients represent in this regression model? How is the effect of plate accounted for?

```r
model_with_binary_indicators <- {PRICE ~ BOWL + CASS + DISH + TRAY}
model_with_binary_indicators_fit <- lm(model_with_binary_indicators, data = tableware)
print(model_with_binary_indicators_fit)
```

```
##
## Call:
## lm(formula = model_with_binary_indicators, data = tableware)
##
```

```
## Coefficients:
## (Intercept)         BOWL         CASS         DISH         TRAY
##       51.83        15.56        75.09        28.31        47.12
```

```
print(summary(model_with_binary_indicators_fit))
```

```
##
## Call:
## lm(formula = model_with_binary_indicators, data = tableware)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -76.950 -26.362  -2.333  26.109  90.050
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    51.83      12.11   4.281 7.68e-05 ***
## BOWL           15.56      14.28   1.089  0.28086
## CASS           75.09      16.69   4.499 3.67e-05 ***
## DISH           28.31      18.31   1.546  0.12785
## TRAY           47.12      16.69   2.823  0.00665 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36.33 on 54 degrees of freedom
## Multiple R-squared:  0.3367, Adjusted R-squared:  0.2876
## F-statistic: 6.853 on 4 and 54 DF,  p-value: 0.0001548
```

```
# The estimated coefficients represent incremental costs associated with the
# types of tableware. The type plate is represented by all zeroes for the
# indicator variables included in the model with binary indicators.

index <- tableware$TYPE == "plate"
mean(tableware[index,8])
```

```
## [1] 51.83333
```

```
# But there is a better way to fit a model of this form using R
# because the tableware data frame has the factor variable type.
# This factor variable can be used to create contrasts.
# If we like binary indicator contrasts, we can ask for treatment contrasts.
options(contrasts = c("contr.treatment", "contr.poly"))
my_factor_model <- {PRICE ~ TYPE}
my_factor_model_fit <- lm(my_factor_model, data = tableware)
print(my_factor_model_fit)
```

```
##
## Call:
## lm(formula = my_factor_model, data = tableware)
##
## Coefficients:
## (Intercept)     TYPEcass     TYPEdish    TYPEplate     TYPEtray
##       67.39        59.53        12.75       -15.56        31.56
```

```
print(summary(my_factor_model_fit))
```

```
##
```

```
## Call:
## lm(formula = my_factor_model, data = tableware)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -76.950 -26.362  -2.333  26.109  90.050
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   67.391      7.575   8.897 3.62e-12 ***
## TYPEcass      59.529     13.760   4.326 6.59e-05 ***
## TYPEdish      12.752     15.681   0.813   0.4197
## TYPEplate    -15.558     14.283  -1.089   0.2809
## TYPEtray      31.559     13.760   2.294   0.0257 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36.33 on 54 degrees of freedom
## Multiple R-squared:  0.3367, Adjusted R-squared:  0.2876
## F-statistic: 6.853 on 4 and 54 DF,  p-value: 0.0001548
```

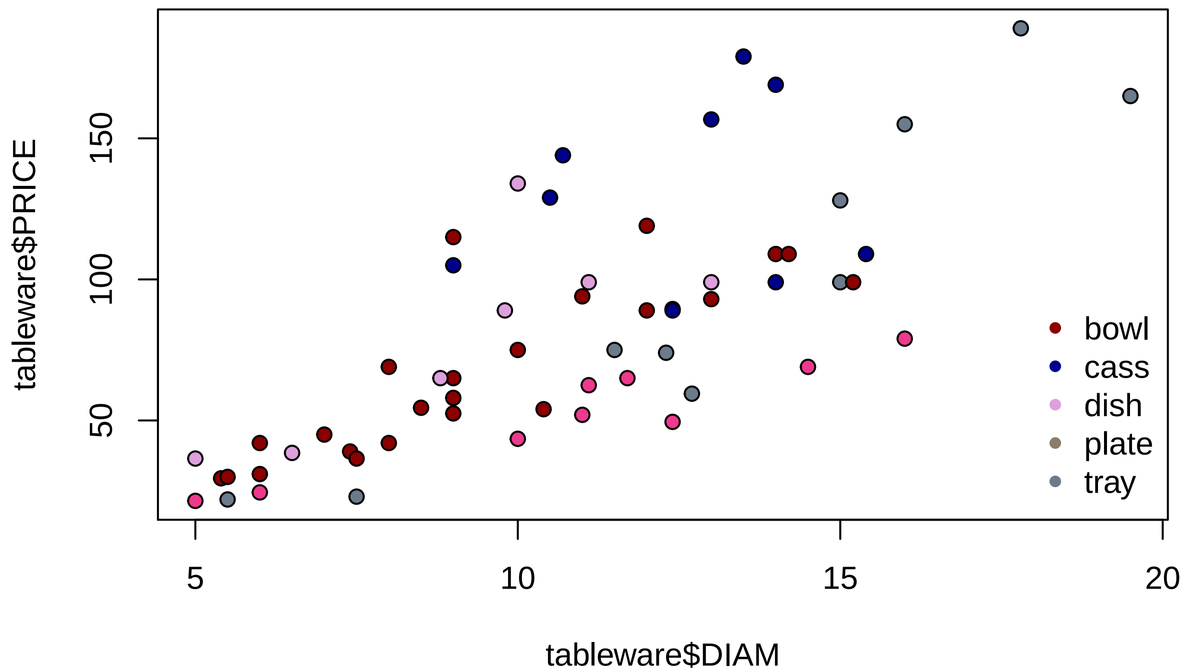
```r
print(anova(my_factor_model_fit))  # type is statistically significant
```

```
## Analysis of Variance Table
##
## Response: PRICE
##           Df Sum Sq Mean Sq F value    Pr(>F)
## TYPE       4  36174  9043.5  6.8532 0.0001548 ***
## Residuals 54  71258  1319.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Note that the R-squared value from this model is identical to
# that obtained with the binary indicators.
```

7) Use the tableware.csv data. Plot PRICE versus DIAM and calculate the Pearson product moment correlation coefficient. Include DIAM in the regression model in (6). Compare results between the two models. DIAM is referred to as a covariate. Does its inclusion improve upon the fit of the first model without DIAM?

```r
plot(tableware$DIAM, tableware$PRICE, pch = 21, bg = c("darkred","blue4","plum",
    "violetred2","slategray4")[unclass(tableware$TYPE)])
legend("bottomright", legend = c("bowl","cass","dish","plate","tray"),
    col = c("darkred","blue4","plum","bisque4","slategray4"), pch = 20, bty = "n")
```




```
with(tableware, print(cor(DIAM, PRICE)))
```

```
## [1] 0.7552496
```

```
# First fit PRICE as a function of TYPE.
Price_Type <- {PRICE ~ TYPE}
Price_Type_fit <- lm(Price_Type, data = tableware)
summary(Price_Type_fit)
```

```
##
## Call:
## lm(formula = Price_Type, data = tableware)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -76.950 -26.362  -2.333  26.109  90.050
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   67.391      7.575   8.897 3.62e-12 ***
## TYPEcass      59.529     13.760   4.326 6.59e-05 ***
## TYPEdish      12.752     15.681   0.813   0.4197
## TYPEplate    -15.558     14.283  -1.089   0.2809
## TYPEtray      31.559     13.760   2.294   0.0257 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 36.33 on 54 degrees of freedom
## Multiple R-squared:  0.3367, Adjusted R-squared:  0.2876
## F-statistic: 6.853 on 4 and 54 DF,  p-value: 0.0001548
```

```r
anova(Price_Type_fit)
```

```
## Analysis of Variance Table
##
## Response: PRICE
##           Df Sum Sq Mean Sq F value    Pr(>F)
## TYPE       4  36174  9043.5  6.8532 0.0001548 ***
## Residuals 54  71258  1319.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

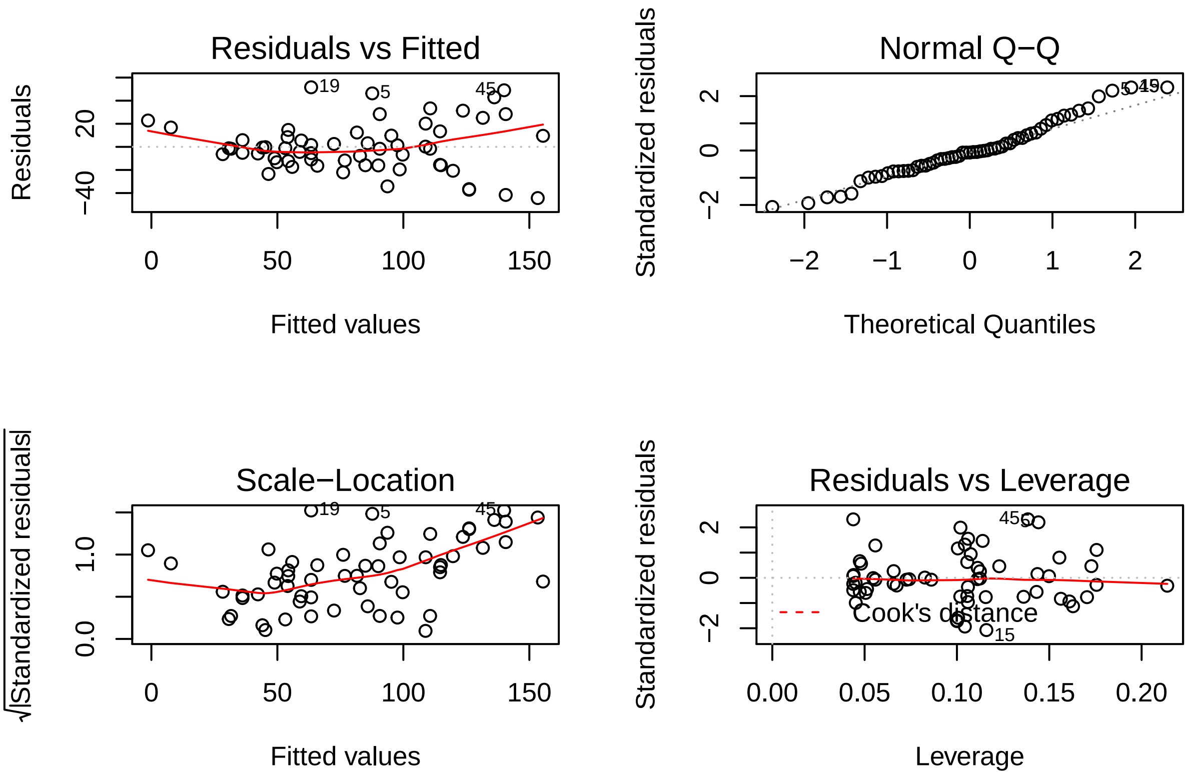
```r
# Then, expand the model to include DIAM.
bigger_model <- {PRICE ~ DIAM + TYPE}
bigger_model_fit <- lm(bigger_model, data = tableware)
summary(bigger_model_fit)
```

```
##
## Call:
## lm(formula = bigger_model, data = tableware)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -44.341 -14.426  -1.617  11.102  51.596
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -18.3107    10.4568  -1.751 0.085719 .
## DIAM          9.0794     0.9872   9.197 1.44e-12 ***
## TYPEcass     31.8285     9.1312   3.486 0.000994 ***
## TYPEdish     15.1821     9.8272   1.545 0.128318
## TYPEplate   -28.4183     9.0563  -3.138 0.002778 **
## TYPEtray     -3.3142     9.4172  -0.352 0.726284
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.76 on 53 degrees of freedom
## Multiple R-squared:  0.7445, Adjusted R-squared:  0.7204
## F-statistic: 30.89 on 5 and 53 DF,  p-value: 1.422e-14
```

```r
anova(bigger_model_fit)  # both variables are significant
```

```
## Analysis of Variance Table
##
## Response: PRICE
##           Df Sum Sq Mean Sq  F value    Pr(>F)
## DIAM       1  61280   61280 118.3229 4.091e-15 ***
## TYPE       4  18704    4676   9.0287 1.228e-05 ***
## Residuals 53  27449     518
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Additional graphics can be examined for the fitted model itself.
# These are diagnostic graphics.
par(mfrow=c(2,2))
plot(bigger_model_fit)
```




```r
par(mfrow=c(1,1))

# Note how everything we do in R involves objects and functions.
# Fitted models are objects. And by giving these fitted models names we
# make it easy to use all kinds of functions with these models.
# By naming a fitted linear model "my_biggest_model_fit," for example,
# we can easily obtain a summary table with regression coefficients,
# an analysis of variance for effects, and diagnostic graphics.
# We can also obtain confidence intervals, predictions, and
# prediction intervals. R is an object-oriented programming
# environment that helps us to do data science.
```

On the value of iterative model development and transformation

The below does not include questions or code prompts. It is meant to show the value of iterative model development and the use of a transformation to address heteroscedasticity and improve model specification.

```r
tableware <- read.csv("tableware.csv", sep = ",")
str(tableware)
```

```
## 'data.frame':    59 obs. of  9 variables:
##  $ TYPE : Factor w/ 5 levels "bowl","cass",..: 2 2 2 1 3 2 5 5 3 3 ...
##  $ BOWL : int  0 0 0 1 0 0 0 0 0 0 ...
##  $ CASS : int  1 1 1 0 0 1 0 0 0 0 ...
##  $ DISH : int  0 0 0 0 1 0 0 0 1 1 ...
##  $ TRAY : int  0 0 0 0 0 0 1 1 0 0 ...
##  $ DIAM : num  10.7 14 9 8 10 10.5 16 15 6.5 5 ...
##  $ TIME : num  47.6 63.1 58.8 34.9 55.5 ...
##  $ PRICE: num  144 169 105 69 134 129 155 99 38.5 36.5 ...
##  $ RATE : num  3.02 2.68 1.79 1.98 2.41 2.99 2.83 2.24 2.21 1.73 ...
```

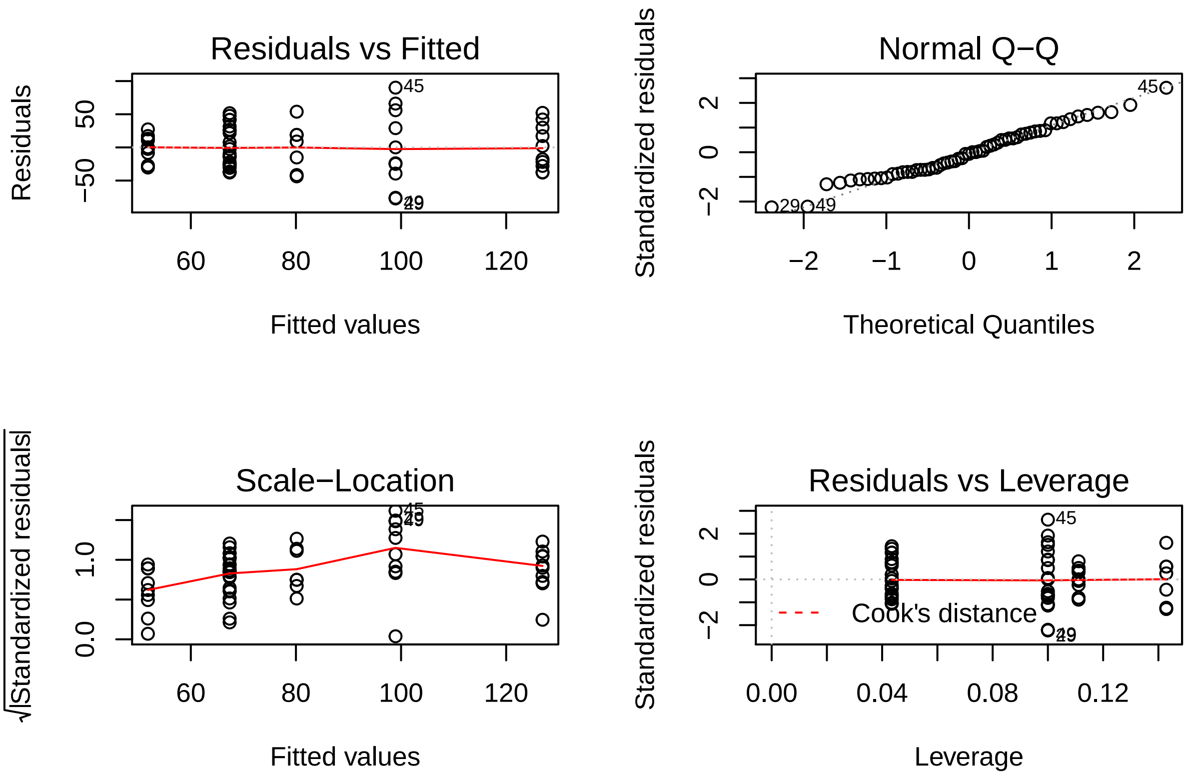
```r
# First fit PRICE as a function of TYPE.
Price_Type <- {PRICE ~ TYPE}
Price_Type_fit <- lm(Price_Type, data = tableware)
summary(Price_Type_fit)
```

```
##
## Call:
## lm(formula = Price_Type, data = tableware)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -76.950 -26.362  -2.333  26.109  90.050
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   67.391      7.575   8.897 3.62e-12 ***
## TYPEcass      59.529     13.760   4.326 6.59e-05 ***
## TYPEdish      12.752     15.681   0.813   0.4197
## TYPEplate    -15.558     14.283  -1.089   0.2809
## TYPEtray      31.559     13.760   2.294   0.0257 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36.33 on 54 degrees of freedom
## Multiple R-squared:  0.3367, Adjusted R-squared:  0.2876
## F-statistic: 6.853 on 4 and 54 DF,  p-value: 0.0001548
```

```r
anova(Price_Type_fit)
```

```
## Analysis of Variance Table
##
## Response: PRICE
##           Df Sum Sq Mean Sq F value    Pr(>F)
## TYPE       4  36174  9043.5  6.8532 0.0001548 ***
## Residuals 54  71258  1319.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
par(mfrow=c(2,2))
plot(Price_Type_fit)
```




```r
par(mfrow=c(1,1))

# Expand the model to include DIAM.

bigger_model <- {(PRICE) ~ (DIAM) + TYPE}
bigger_model_fit <- lm(bigger_model, data = tableware)
summary(bigger_model_fit)
```

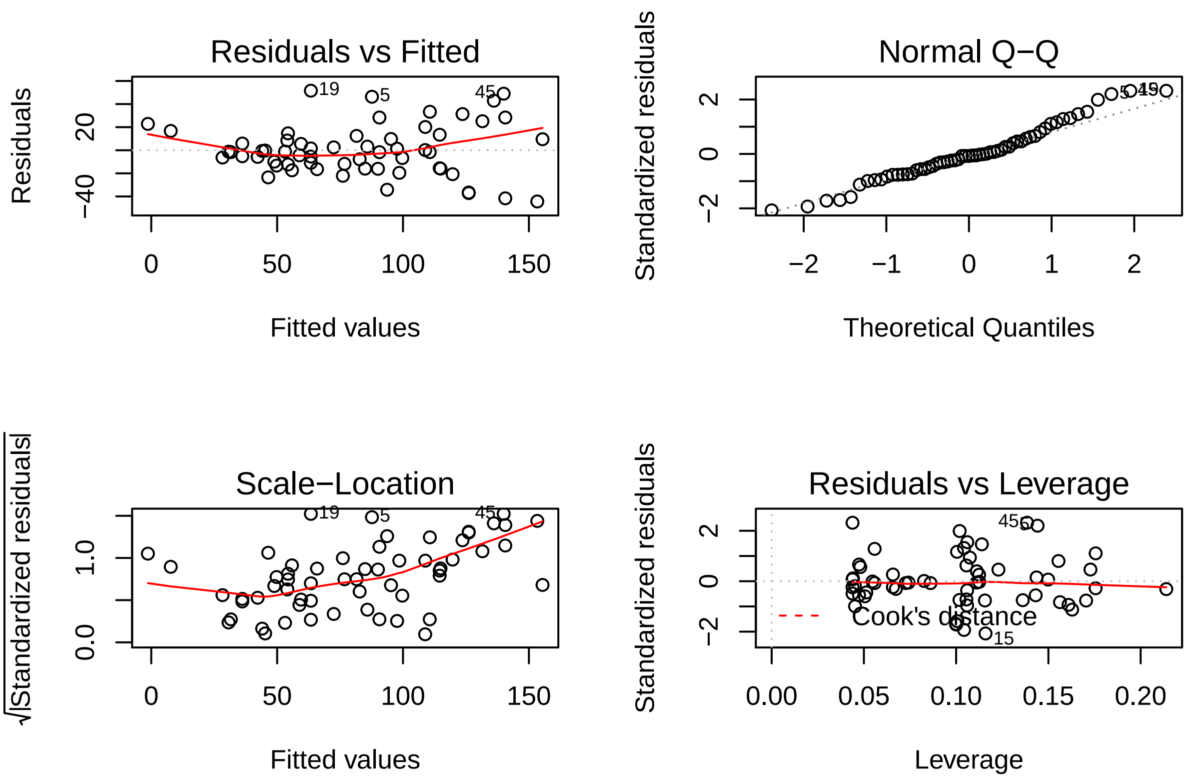
```
##
## Call:
## lm(formula = bigger_model, data = tableware)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -44.341 -14.426  -1.617  11.102  51.596
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -18.3107    10.4568  -1.751 0.085719 .
## DIAM          9.0794     0.9872   9.197 1.44e-12 ***
## TYPEcass     31.8285     9.1312   3.486 0.000994 ***
## TYPEdish     15.1821     9.8272   1.545 0.128318
## TYPEplate   -28.4183     9.0563  -3.138 0.002778 **
## TYPEtray     -3.3142     9.4172  -0.352 0.726284
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 22.76 on 53 degrees of freedom
## Multiple R-squared:  0.7445, Adjusted R-squared:  0.7204
## F-statistic: 30.89 on 5 and 53 DF,  p-value: 1.422e-14
```

```
anova(bigger_model_fit)  # both variables are significant
```

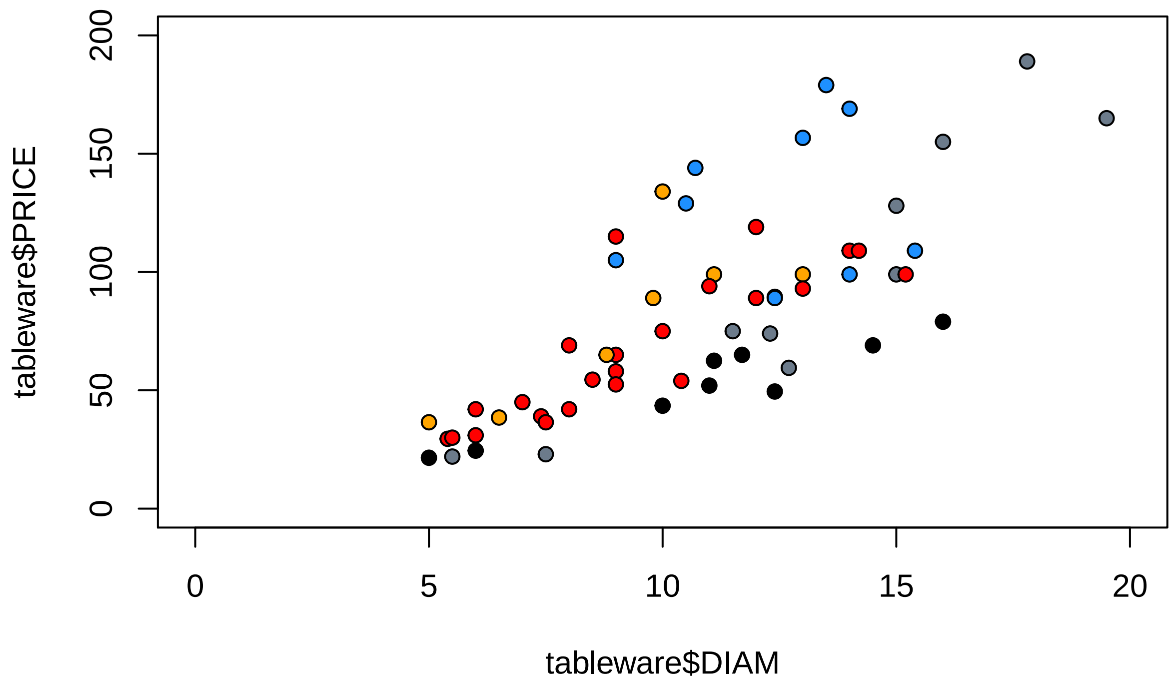
```
## Analysis of Variance Table
##
## Response: (PRICE)
##           Df Sum Sq Mean Sq  F value    Pr(>F)
## DIAM       1  61280   61280 118.3229 4.091e-15 ***
## TYPE       4  18704    4676   9.0287 1.228e-05 ***
## Residuals 53  27449     518
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
par(mfrow=c(2,2))
plot(bigger_model_fit)
```

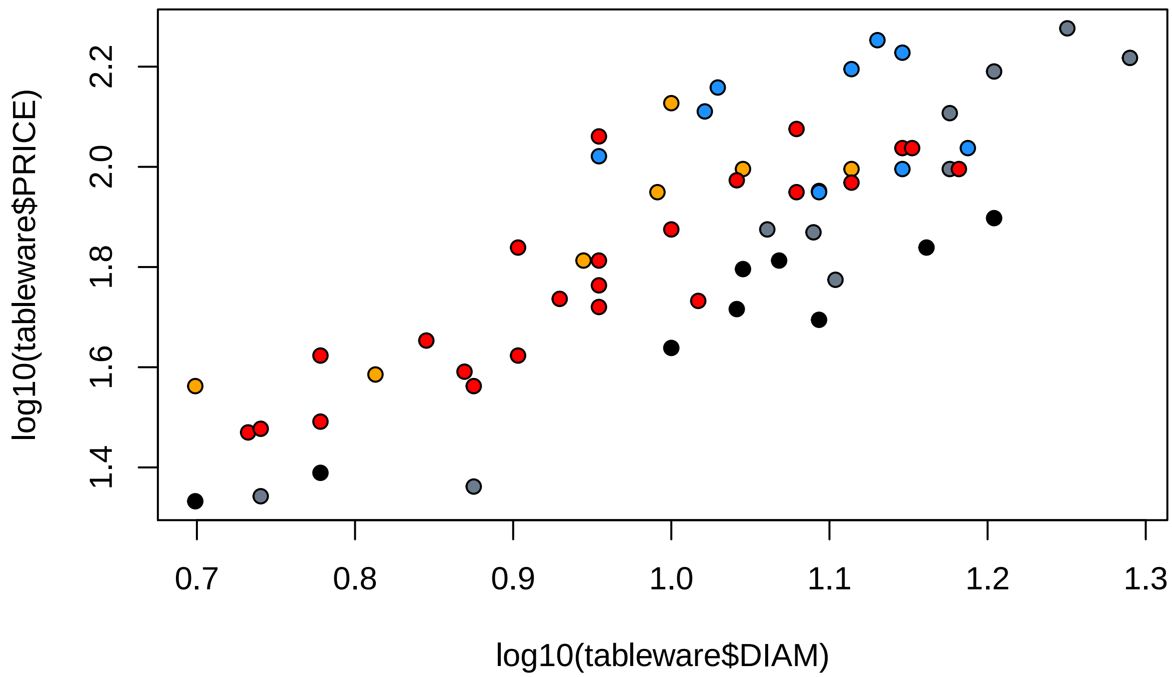



```
par(mfrow=c(1,1))

# This model needs improvement.  The residuals vs fitted plot shows variance
# heteroscedasticity.  The QQ plot shows systematic departure from normality.

# First evaluate the relationship between PRICE and DIAM with a couple plots.

plot(tableware$DIAM, tableware$PRICE, pch = 21, bg = c("red","dodgerblue","orange",
```

```
    "black","slategray4")[unclass(tableware$TYPE)], xlim = c(0,20), ylim = c(0,200))
```




```
# When each level of TYPE is considered, there is a suggestion of a curved wedge shape in the
# plot of PRICE vs DIAM. TYPE also appears to shift a number of points higher on the figure.
# On the log10 scale, the variability is more homogeneous.  A regression may be attempted.
# We allow for different intercepts for each TYPE.  Note the following plot.

plot(log10(tableware$DIAM), log10(tableware$PRICE), pch = 21, bg = c("red","dodgerblue","orange",
     "black","slategray4")[unclass(tableware$TYPE)])
```



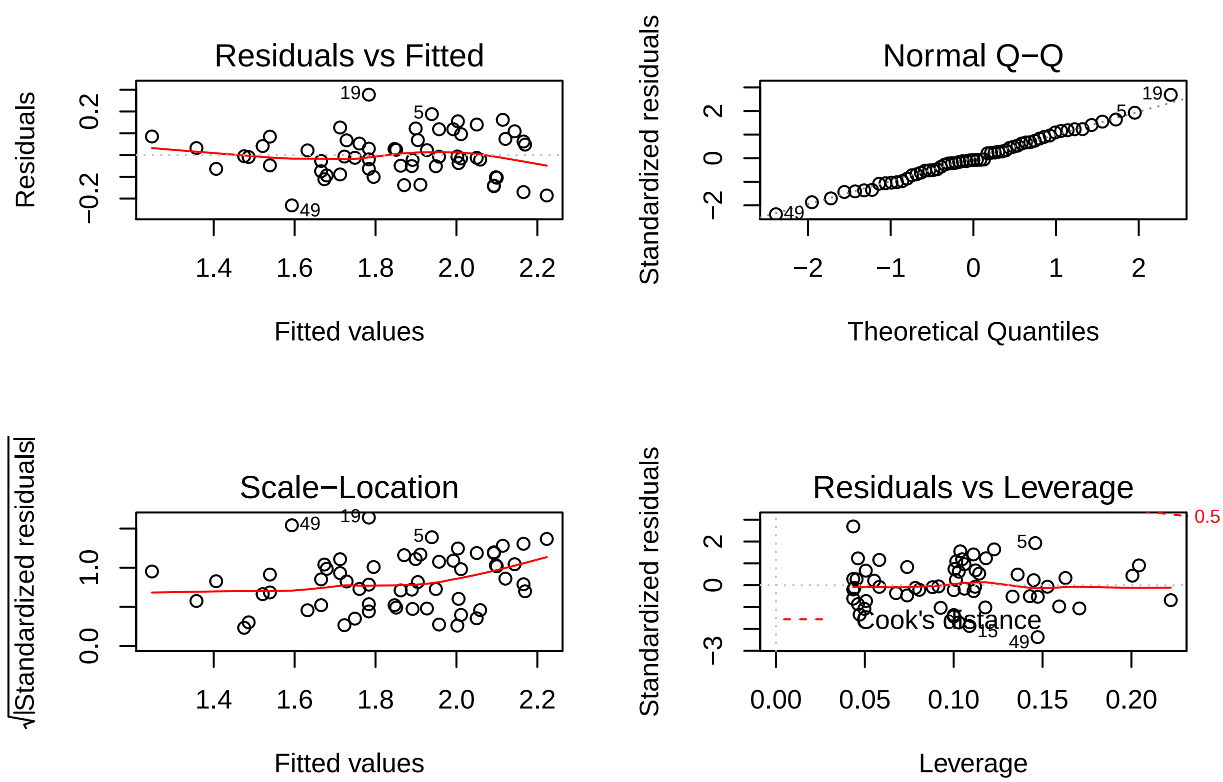

```r
bigger_model <- {log10(PRICE) ~ log10(DIAM) + TYPE }
bigger_model_fit <- lm(bigger_model, data = tableware)
summary(bigger_model_fit)
```

```
##
## Call:
## lm(formula = bigger_model, data = tableware)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.231395 -0.063445 -0.007697  0.067969  0.277199
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.45768    0.10125   4.520 3.50e-05 ***
## log10(DIAM)  1.38939    0.10341  13.435  < 2e-16 ***
## TYPEcass     0.11572    0.04233   2.734   0.0085 **
## TYPEdish     0.09216    0.04553   2.024   0.0480 *
## TYPEplate   -0.18156    0.04183  -4.340 6.44e-05 ***
## TYPEtray    -0.08036    0.04251  -1.890   0.0642 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1054 on 53 degrees of freedom
## Multiple R-squared:  0.8343, Adjusted R-squared:  0.8187
## F-statistic: 53.37 on 5 and 53 DF,  p-value: < 2.2e-16
```

```r
anova(bigger_model_fit)
```

```
## Analysis of Variance Table
##
## Response: log10(PRICE)
##              Df  Sum Sq Mean Sq F value    Pr(>F)
## log10(DIAM)   1 2.41883 2.41883 217.606 < 2.2e-16 ***
## TYPE          4 0.54752 0.13688  12.314 3.761e-07 ***
## Residuals    53 0.58913 0.01112
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Both variables are statistically significant, although some levels of TYPE do not differ
# from the intercept representing "bowl".

par(mfrow=c(2,2))
plot(bigger_model_fit)
```




```r
par(mfrow=c(1,1))

# The fitted model with the log10 transformation seems better.  Using Bartlett's
# test we can reassure ourselves regarding variance homogeneity. The more extreme
# residuals (15, 19 and 49) do not have sufficient influence (leverage) in the
# regression to be of concern.

class.intervals <- cut(bigger_model_fit$fitted.values, breaks = 5)
tableware <- data.frame(tableware, class.intervals)
```

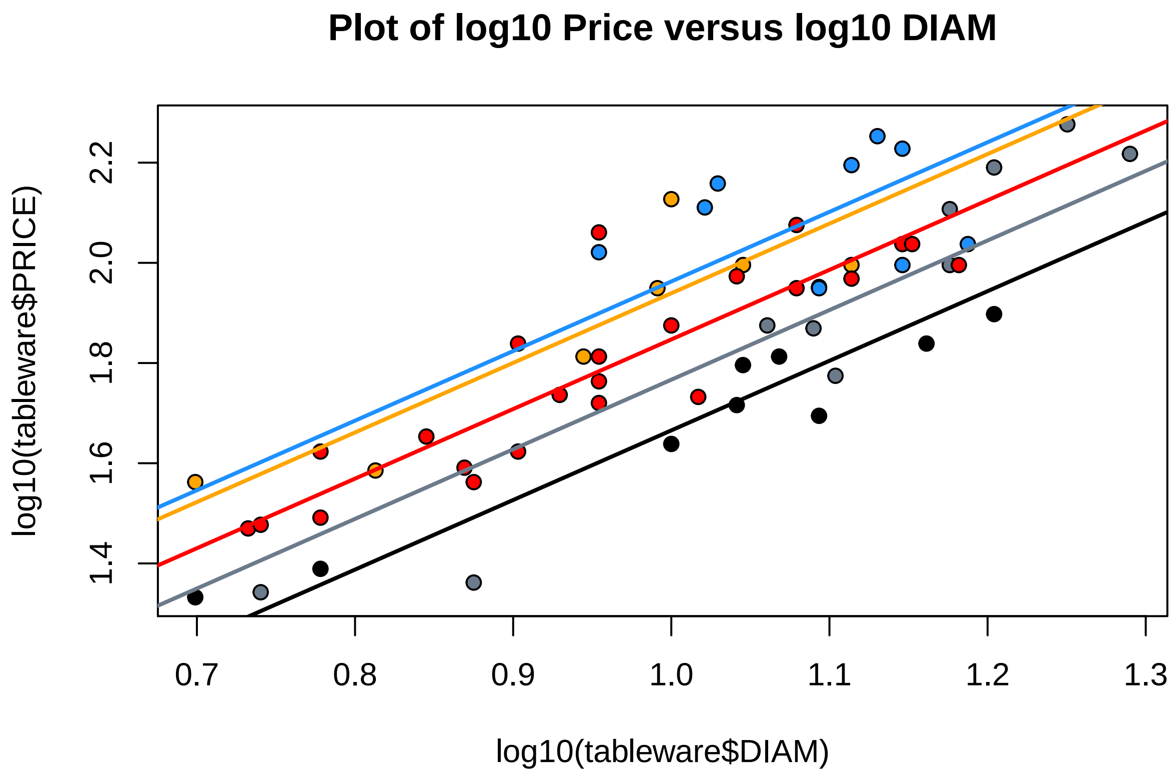
```r
bartlett.test(bigger_model_fit$residuals ~ class.intervals, data = tableware)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  bigger_model_fit$residuals by class.intervals
## Bartlett's K-squared = 1.4999, df = 4, p-value = 0.8267
```

```r
# One interpretation of this is that we have PRICE = constant*DIAM^power, where
# the constant is a function of TYPE, and the error term has a variance that is
# proportional to DIAM.  On the log10 scale, this relationship is addressed
# resulting in a more homogeneous variance for regression.

# One way to verify results is to use a robust regression package and compare
# coefficients between the two models.

log10.coeff <- round(bigger_model_fit$coefficients, digits = 2)

library(MASS)
object <- rlm(log10(PRICE) ~ log10(DIAM) + TYPE, tableware)
robust.coeff <- round(object$coefficients, digits = 2)

cbind(log10.coeff,robust.coeff)
```

```
##             log10.coeff robust.coeff
## (Intercept)        0.46         0.47
## log10(DIAM)        1.39         1.37
## TYPEcass           0.12         0.14
## TYPEdish           0.09         0.09
## TYPEplate         -0.18        -0.17
## TYPEtray          -0.08        -0.06
```

```r
# Considering the standard error for the coefficients is around 0.1, the results are
# very comparable and further reinforce the notion the outliers have little influence.

# The results can be presented with the data and fitted regression lines.

bowl.intercept <- bigger_model_fit$coefficients[1]
cass.intercept <- bowl.intercept + bigger_model_fit$coefficients[3]
dish.intercept <- bowl.intercept + bigger_model_fit$coefficients[4]
plate.intercept <- bowl.intercept + bigger_model_fit$coefficients[5]
tray.intercept <- bowl.intercept + bigger_model_fit$coefficients[6]
slope <- bigger_model_fit$coefficients[2]

plot(log10(tableware$DIAM), log10(tableware$PRICE), pch = 21, main =
       c("Plot of log10 Price versus log10 DIAM"), bg = c("red","dodgerblue",
      "orange", "black","slategray4")[unclass(tableware$TYPE)])
abline(bowl.intercept, slope, col = "red", lwd = "2")
abline(cass.intercept, slope, col = "dodgerblue", lwd = "2")
abline(dish.intercept, slope, col = "orange", lwd = "2")
abline(plate.intercept, slope, col = "black", lwd = "2")
abline(tray.intercept, slope, col = "slategray4", lwd = "2")
```


Plot of log10 Price versus log10 DIAM
log10(tableware$PRICE)
1.4
1.6
1.8
2.0
2.2
0.7
0.8
0.9
1.0
1.1
1.2
1.3
log10(tableware$DIAM)