Lab 2 - Model Building, Model Checking, and Baseball Salaries

---

Lab Goals

In this lab we will look at the relationship between the performance statistics and contract status of baseball player and his annual salary using multiple linear regression (MLR). This analysis will explore the following in the context of a MLR framework.

1. model construction
2. model checking
3. hypothesis testing

Baseball Data

The data file *bball.csv* can be found in the folder Lab 2. A subset of the variables in this data set can be classified as either **Performance Statistics** or **Contractual Statistics**. We will consider 9 of these variables as possible *independent variables* or *covariates* in a MLR model. The data set also includes the dependent variable for the model, `salary`. All variables except for the contractural statistics are continuous. All contractural statistics are binary. Here is a list of the ten variables we will consider in this data set with a description of each variable.

Variable	Description
`salary`	annual salary in thousands of dollars
`bat.av`	batting average
`on.base`	on base percentage
`runs`	number of runs
`home.runs`	number of home runs
`rbi`	number of runs batted in
`free.elig`	free agent eligibility, 1 = Yes, 0 = No
`free.agent`	free agent?, 1 = Yes, 0 = No
`arb.elig`	arbitration eligibility, 1 = Yes, 0 = No
`arb`	arbitration?, 1 = Yes, 0 = No

Load this data set both into the console below and in this R Markdown document using the code chunk provided for you.

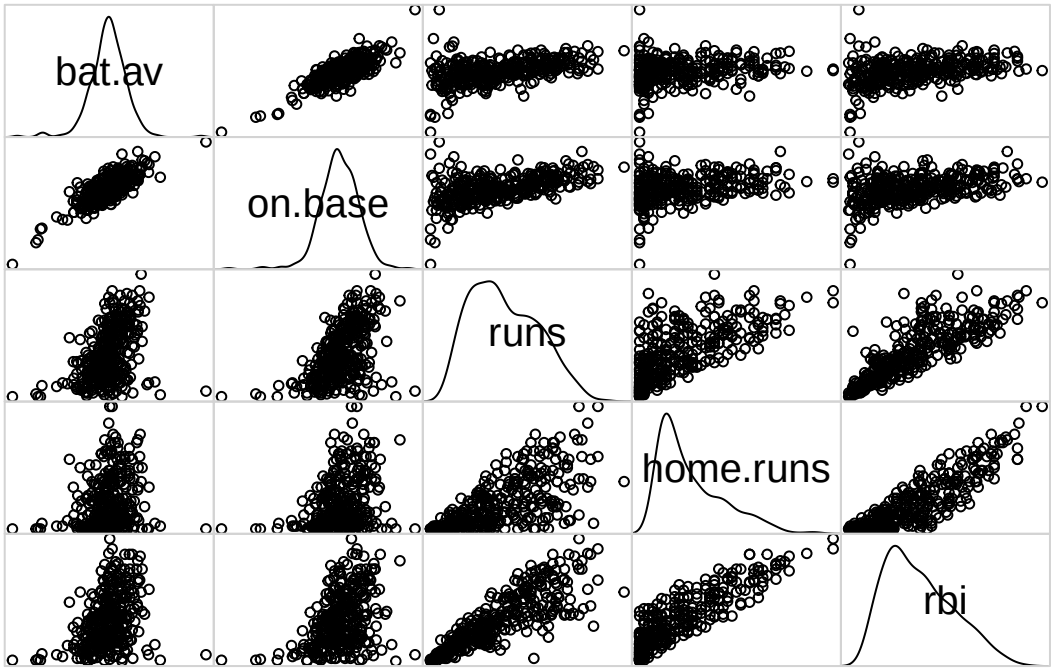
```
bball <- read.csv("~/Documents/Classes/BTRY6020/Labs/Lab2/bball.csv")
```

This document includes R code that will help us analyze the baseball data. Knit this document now to get a pdf of this document with all of the R code evaluated.

Examining Distributions - Independent Variables

First, we want to examine the joint distribution of the continuous explanatory variables. We will use the `corrgram()` function found in the **corrgram** package to create a matrix of scatterplots of all possible pairs of these covariates.

```
library(corrgram)
# Vector of variable names
corrdat = c("bat.av", "on.base", "runs", "home.runs", "rbi")
# Creates scatterplots of variables in corrdat
corrgram(bball[, corrdat], panel = "panel.pts", diag.panel = "panel.density")
```

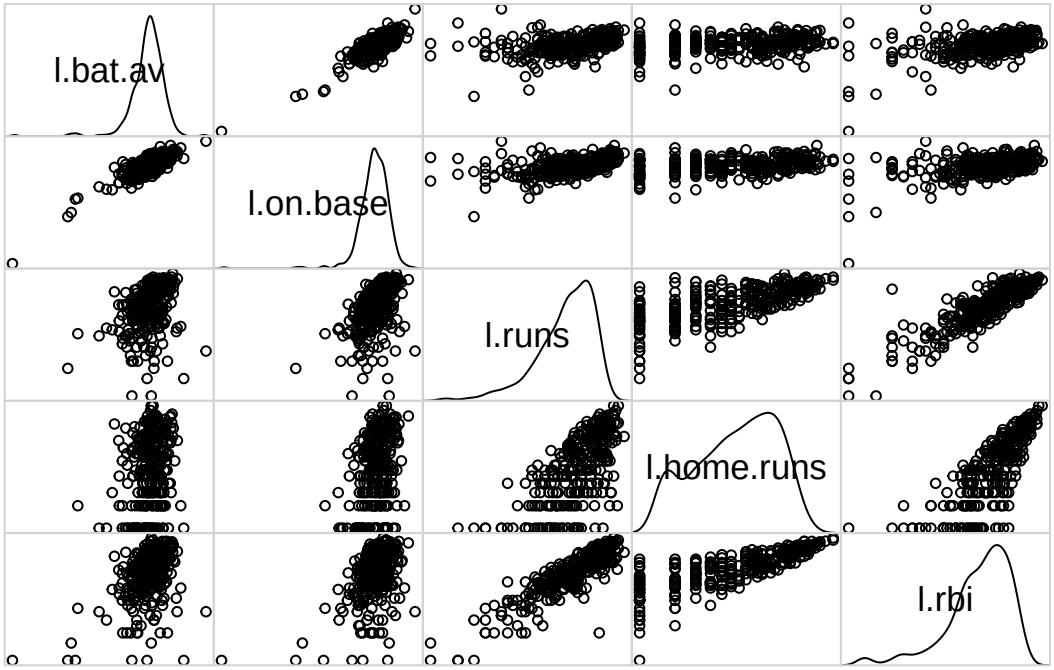


1. The diagonal of this plot includes approximate probability density functions for each continuous independent variable. Do you see any distributional problems with these data?

You can try experimenting with changing the scale of the axes from linear to log. The following code will look at these distributions on the log scale.

```
# Creating log variables in the dataset
bball[, 19:23] = c(log(bball$bat.av), log(bball$on.base), log(bball$runs), log(bball$home.runs),
    log(bball$rbi))
# Assigning names to these variables
names(bball)[19:23] = c("l.bat.av", "l.on.base", "l.runs", "l.home.runs", "l.rbi")
# Vector of variable names
```

```
corrdat2 = c("l.bat.av", "l.on.base", "l.runs", "l.home.runs", "l.rbi")
# Creates Scatterplots
corrgram(bball[, corrdat2], panel = "panel.pts", diag.panel = "panel.density")
```



2. Does taking the log of the original variables result in less skewness in these distributions?

Zero values in the data are causing some compression in these distributions. This can be improved by creating new columns using, for example, log(rbi+1) instead of `rbi`. However, since the skewness in the original data appears to be fairly mild we will leave the variables as is.

The `cor` function in R will create a correlation matrix for these variables.

```
cor(bball[,corrdat])
```

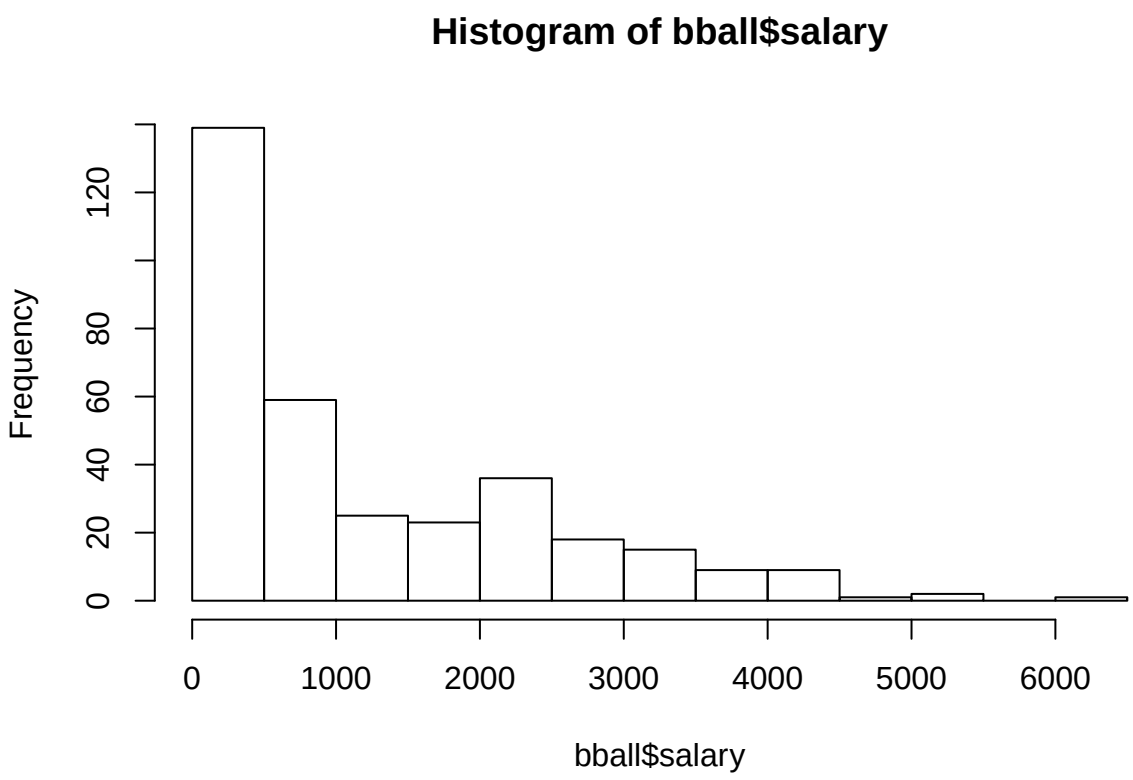
```
##           bat.av on.base   runs home.runs    rbi
## bat.av    1.0000  0.8060 0.4367    0.2127 0.3695
## on.base   0.8060  1.0000 0.5136    0.3112 0.3994
## runs      0.4367  0.5136 1.0000    0.6811 0.8335
## home.runs 0.2127  0.3112 0.6811    1.0000 0.8774
## rbi       0.3695  0.3994 0.8335    0.8774 1.0000
```

3. Which three pairs of variables are most correlated?
4. How does this correlation affect the standard errors of the estimated regression coefficients in the MLR model?

Examining Distributions - Response

We now turn to the response, `salary`. Using the `hist()` function in R, include a code chunk below that creates a histogram of `salary`. Run the code in the console below.

```r
hist(bball$salary)
```



1. How would you describe the distribution of `salary`? Why might this be a problem?
2. How might we transform `salary`?

Initial Model

We will fit an initial model with all the variables without any transformations. The `lm()` function in R will fit a multiple linear regression model. It is a good idea to use the `as.factor()` function to denote all of the categorical predictors as factors before running a linear model. The following code chunk first denotes the categorical predictors as factors and then runs a MLR model with `salary` as the response and the 9 independent variables described above as covariates. Run this model both in this document by setting `eval=TRUE` and in the R console below.

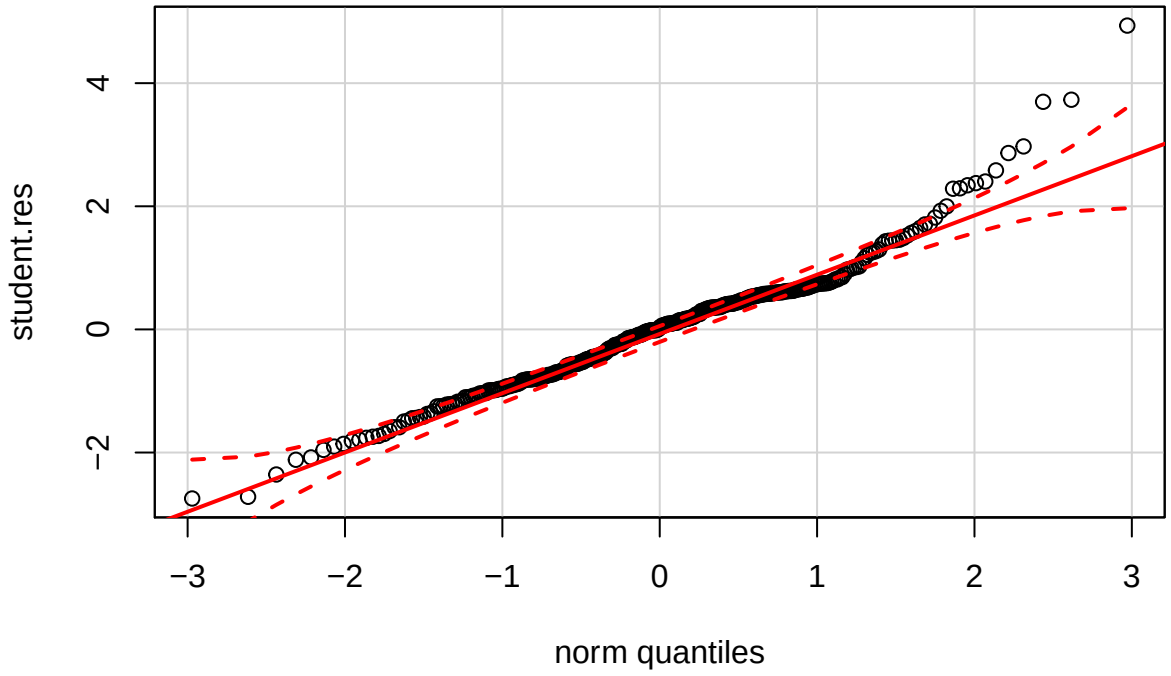
```r
bball$free.elig<-as.factor(bball$free.elig)
bball$free.agent<-as.factor(bball$free.agent)
bball$arb.elig<-as.factor(bball$arb.elig)
bball$arb<-as.factor(bball$arb)
bball.lm <- lm(salary~free.elig+free.agent+arb.elig+arb+bat.av+on.base+runs+home.runs+rbi,data=bball)
```

We were concerned about how the distribution of `salary` might affect the model assumptions. Here we will check the following assumptions of this model:

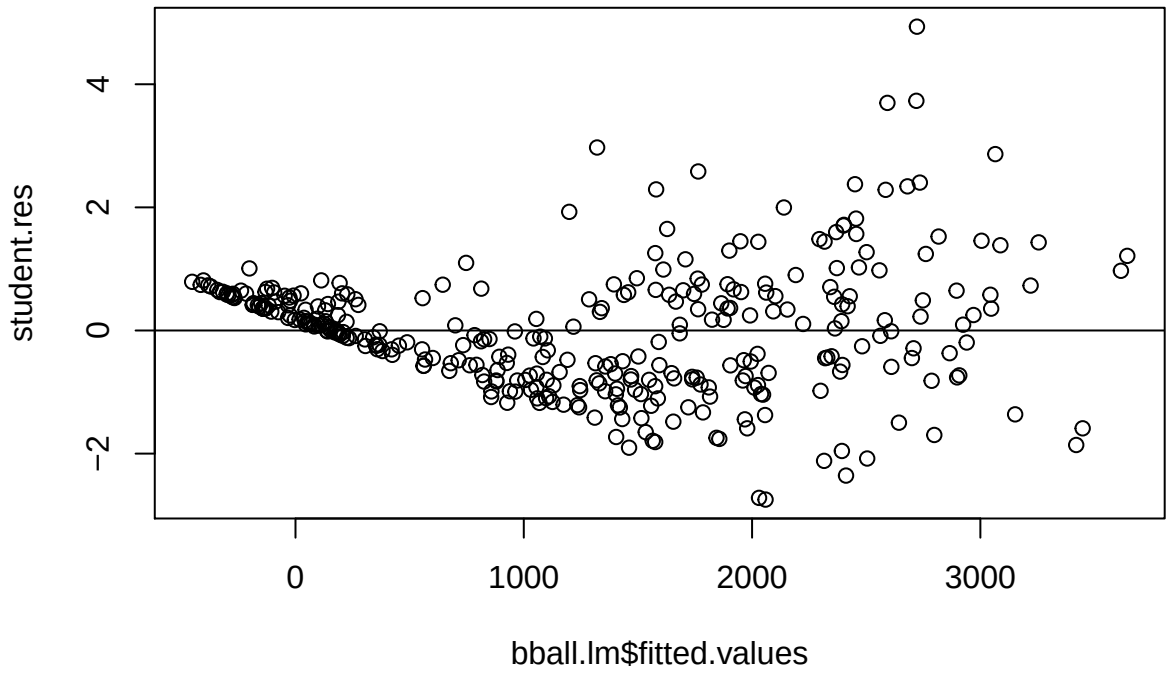
1. normality of error distribution
2. homogeneous variance of the error distribution

The studentized residuals of the model can be obtained using the `studres()` function with the name of the linear model as the argument. In the code chunk below, we first obtain the studentized residuals. Then we will use both a Q-Q plot and a plot of the residuals against the predicted values to check assumptions (1) and (2) above. Note: the `studres()` function is found in the `MASS` package and the `qqPlot()` function is found in the `car` package. These packages must be accessed through the `library()` function.

```
library(MASS)
student.res <- studres(bball.lm)
library(car)
qqPlot(student.res)
```



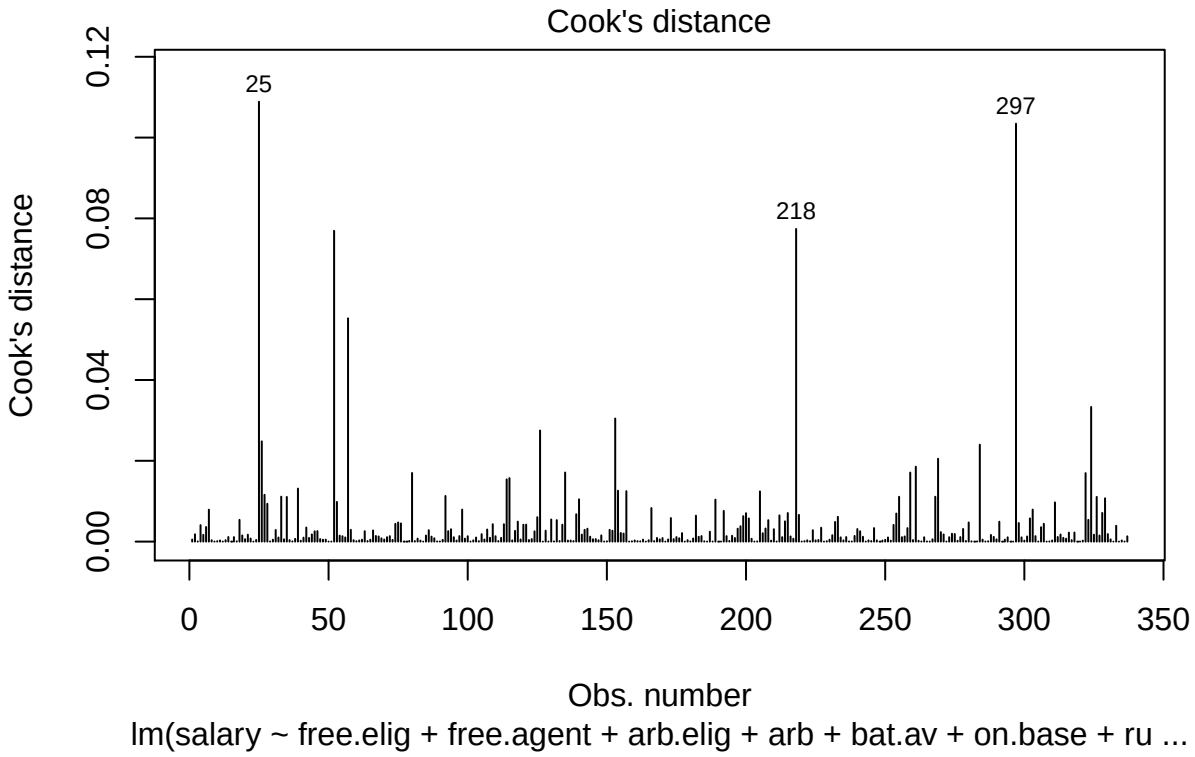

```
plot(bball.lm$fitted.values,student.res)
abline(0,0)
```



1. Looking first at the Q-Q plot, does there seem to be a problem with the normality assumption?
2. Looking at the second plot, does the homogeneous variance assumption seem to be met?

Another concern we might have is whether there are any influential points in the data. The following code stores the Cook's distance for every observation and then creates a plot of the Cook's distances. Are there any observations that are significantly affecting the model?

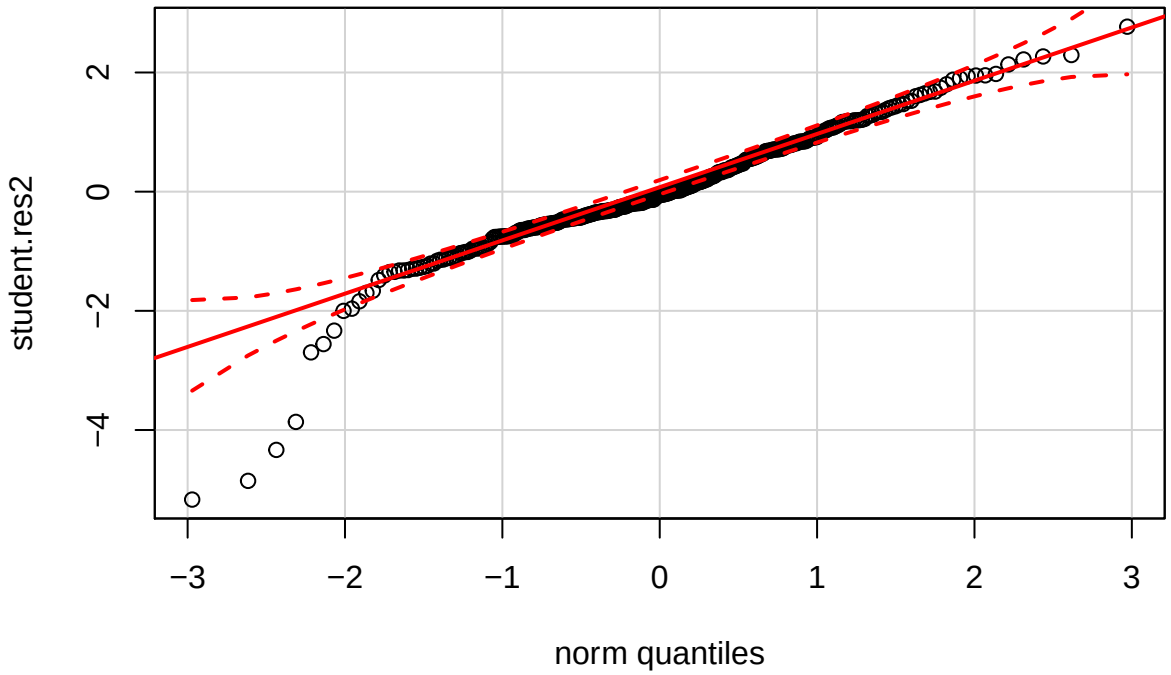
```
#Stores Cook's distances in a vector
cooks_d = cooks.distance(bball.lm)
#Creates plot of Cook's distances
cutoff=1
plot(bball.lm, which=4, cook.levels=cutoff)
```



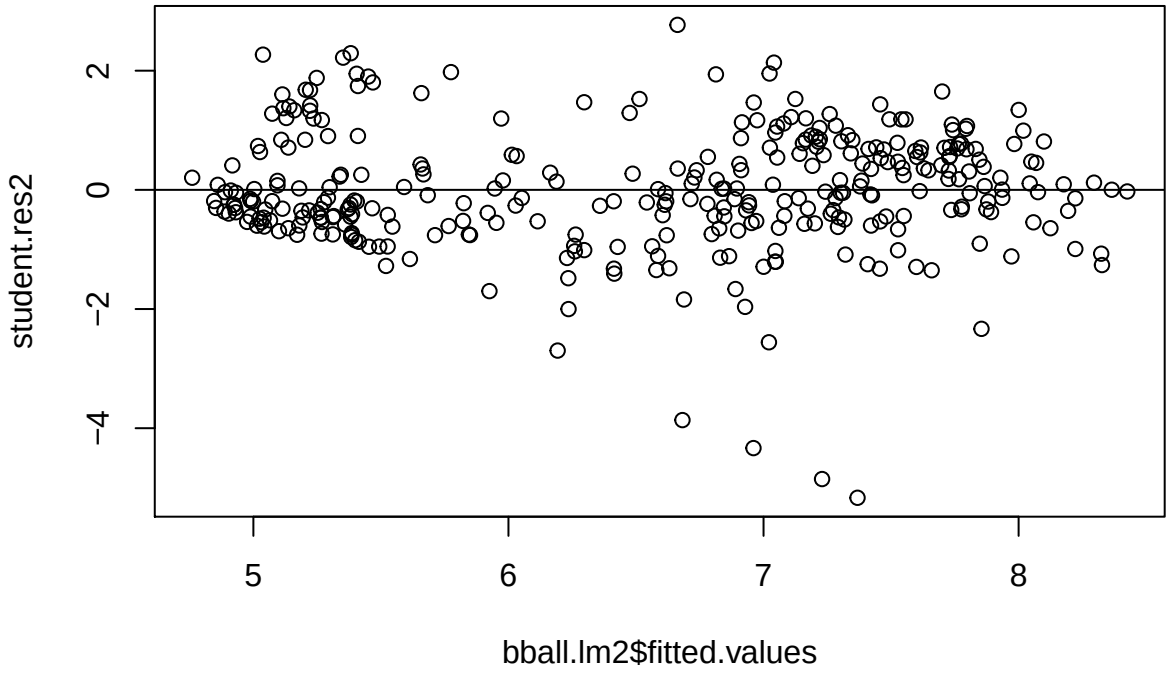
New Model

Using the log transformation on the response, `salary`, may fix the violation of assumptions seen above for the MLR model. Here we will re-run our model using log(salary) as the response instead of `salary`. The following code will also create a Q-Q plot and a studentized residual by predicted plot for the new model.

```
bball.lm2 <- lm(log(salary)~free.elig+free.agent+arb.elig+arb+bat.av+on.base+runs+home.runs+rbi,data=bba
library(MASS)
student.res2 <-studres(bball.lm2)
library(car)
qqPlot(student.res2)
```




```
plot(bball.lm2$fitted.values,student.res2)
abline(0,0)
```



1. Any studentized residuals that are greater than 3 in absolute value are an indicative of observations that are outliers. Looking at the studentized residuals by predicted values plot, are there any outliers?
2. How are the outliers impacting the Q-Q plot and the studentized residuals versus predicted values plot?

These outliers should be examined to determine whether these are data errors that could be corrected or if they should be excluded from the analysis. We can use the `which()` function to determine which observations are outliers. We will also take a look at the salaries of the outlying observations.

```
#Determines which observations have large residuals
outliers=which(abs(student.res2)>2)
outliers  #Both rows are the observation numbers of the outliers
```

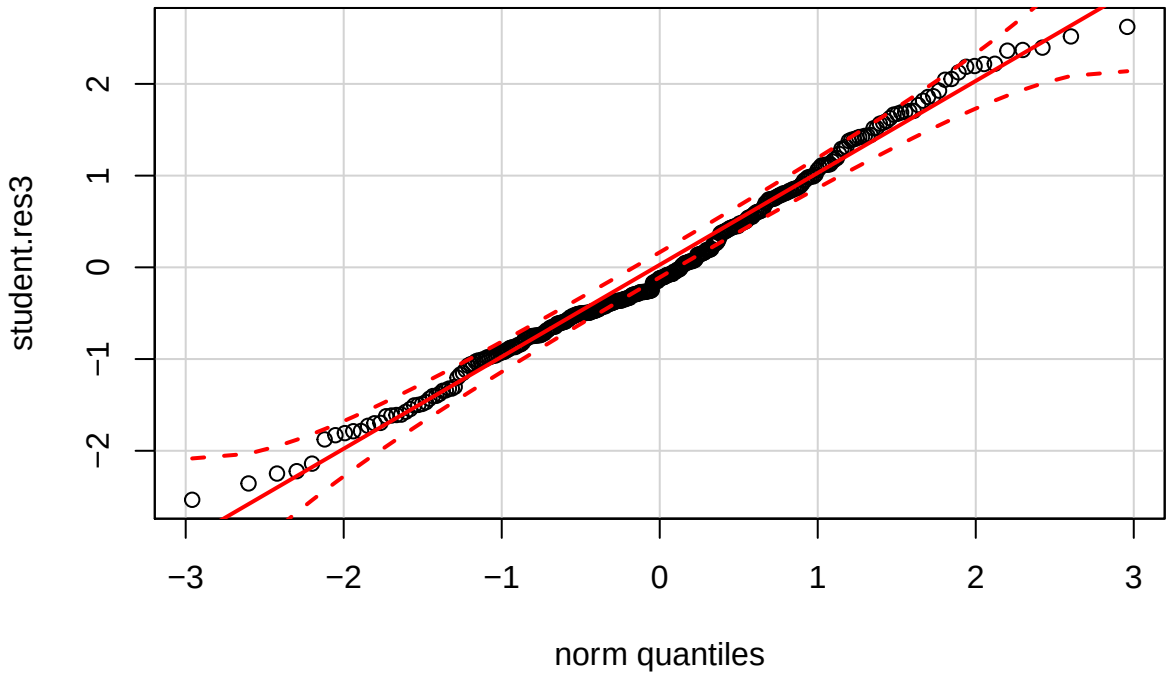
```
##  99 114 135 153 181 183 189 205 215 259 268 284 322
##  99 114 135 153 181 183 189 205 215 259 268 284 322
```

```
bball$salary[outliers] #Salary of outliers
```

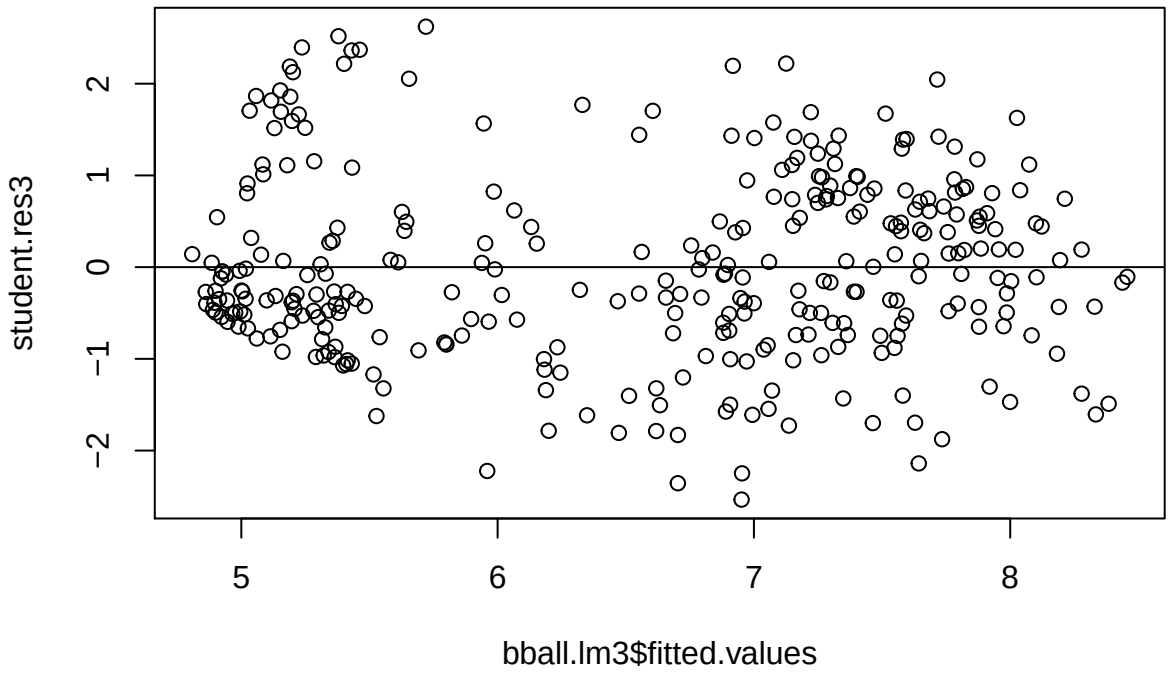
```
##  [1]  750 3600  125 3415  700  525  175  109  284  740  109  109  109
```

The four large outliers are 205, 268, 284 and 322. When we examine them, they all have salary $109,000: the lowest in the league. We'll exclude them from the analysis. Below we will re-run the MLR model with log(salary) as the response without the outliers.

```
# Determines observations we want to include
not_outlier = which(abs(student.res2) <= 2)

# Subset option allows only a subset of the data to be analyzed
bball.lm3 <- lm(log(salary) ~ free.elig + free.agent + arb.elig + arb + bat.av +
    on.base + runs + home.runs + rbi, data = bball, subset = not_outlier)

library(MASS)
student.res3 <- studres(bball.lm3)
library(car)
qqPlot(student.res3)
```




```
plot(bball.lm3$fitted.values, student.res3)
abline(0, 0)
```



3. Are the normality and homogeneous variance assumptions now met?

Examining the model

Now we will examine the model run without the outliers using the response log(salary). A summary of the model can be obtained using the `summary()` function with the name of the linear model as the argument.

```
summary(bball.lm3)
```

```
##
## Call:
## lm(formula = log(salary) ~ free.elig + free.agent + arb.elig +
##     arb + bat.av + on.base + runs + home.runs + rbi, data = bball,
##     subset = not_outlier)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0797 -0.2781 -0.0514  0.3012  1.1202
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.88306    0.18201   26.83  < 2e-16 ***
## free.elig1   1.74729    0.06697   26.09  < 2e-16 ***
## free.agent1 -0.31033    0.08776   -3.54  0.00047 ***
## arb.elig1    1.44463    0.07412   19.49  < 2e-16 ***
## arb1        -0.01365    0.15892   -0.09  0.93163
## bat.av       0.46505    1.09987    0.42  0.67271
## on.base     -0.46803    0.95076   -0.49  0.62288
## runs         0.00806    0.00170    4.74  3.2e-06 ***
## home.runs   -0.00934    0.00587   -1.59  0.11236
```

```
## rbi          0.01095    0.00246    4.46  1.1e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.435 on 314 degrees of freedom
## Multiple R-squared:  0.865,  Adjusted R-squared:  0.861
## F-statistic:  223 on 9 and 314 DF,  p-value: <2e-16
```

Looking at the p-values associated with each independent variable, we see that many of the regression coefficients are not determined to be significantly different from zero. This may be due to the strong correlation we saw above between several of the independent variables. The following code uses the `vif()` function to determine the *Variance Inflation Factor* for each independent variable. Again the argument of this function is the name of your linear model.

```
vif(bball.lm3)
```

```
##  free.elig free.agent   arb.elig        arb     bat.av    on.base
##      1.831      1.331      1.453      1.166      3.291      3.503
##       runs  home.runs        rbi
##      4.171      5.067      8.939
```

Some of these VIFs are quite large, although the largest is associated with `rbi` which is already strongly significant. Notably, however, `home.runs` is not significant, but does have a high variance inflation factor. We might also consider `bat.av` and `on.base` since we know them to be correlated.

To further examine which variables might be indistinguishable, we look at the correlation matrix of the parameter estimates. This can be obtained by adding the option `correlation = TRUE` to the summary function for this model.

```
summary(bball.lm3, correlation=TRUE)
```

```
##
## Call:
## lm(formula = log(salary) ~ free.elig + free.agent + arb.elig +
##     arb + bat.av + on.base + runs + home.runs + rbi, data = bball,
##     subset = not_outlier)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0797 -0.2781 -0.0514  0.3012  1.1202
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.88306    0.18201   26.83  < 2e-16 ***
## free.elig1   1.74729    0.06697   26.09  < 2e-16 ***
## free.agent1 -0.31033    0.08776   -3.54  0.00047 ***
## arb.elig1    1.44463    0.07412   19.49  < 2e-16 ***
## arb1        -0.01365    0.15892   -0.09  0.93163
## bat.av       0.46505    1.09987    0.42  0.67271
## on.base     -0.46803    0.95076   -0.49  0.62288
## runs         0.00806    0.00170    4.74  3.2e-06 ***
## home.runs   -0.00934    0.00587   -1.59  0.11236
## rbi          0.01095    0.00246    4.46  1.1e-05 ***
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.435 on 314 degrees of freedom
## Multiple R-squared:  0.865,  Adjusted R-squared:  0.861
## F-statistic:  223 on 9 and 314 DF,  p-value: <2e-16
##
## Correlation of Coefficients:
##             (Intercept) free.elig1 free.agent1 arb.elig1 arb1  bat.av
## free.elig1  -0.06
## free.agent1 -0.06       -0.45
## arb.elig1   -0.11        0.39      -0.02
## arb1         0.00        0.05      -0.02       -0.26
## bat.av      -0.21        0.09      -0.02       -0.05     -0.03
## on.base     -0.42       -0.07       0.04        0.08      0.04 -0.78
## runs         0.24       -0.15       0.12       -0.07     -0.11  0.07
## home.runs   -0.03       -0.04       0.16        0.09      0.03  0.27
## rbi         -0.02       -0.05      -0.12       -0.13      0.00 -0.25
##             on.base runs  home.runs
## free.elig1
## free.agent1
## arb.elig1
## arb1
## bat.av
## on.base
## runs        -0.27
## home.runs   -0.18    0.19
## rbi          0.20   -0.61 -0.78
```

This outputs a correlation matrix giving how strongly related two coefficients are. We observe that the coefficient for `home.runs` is strongly correlated with that for `rbi` and that the coefficients for `bat.av` and `on.base` are similarly correlated.

`home.runs` could be part of the cause of the high VIF for `rbi` so we'll try removing it. Neither `bat.av` nor `on.base` are significant, but this may be because they are correlated. We'll try removing `bat.av` since it has the highest p-value.

As a final note, we want to keep track of sums of squares so that we can compare our reduced model to this one with an F test to see if we have been too aggressive in removing observations. The `anova()` function with the name of the linear model as its argument will provide us with the break down of the sequential sum of squares for the model and the sum of squares for error. Note that the independent variables are listed in the order in which they were put in the model.

```
anova(bball.lm3)
```

```
## Analysis of Variance Table
##
## Response: log(salary)
##             Df Sum Sq Mean Sq F value  Pr(>F)
## free.elig    1  175.4   175.4  924.94 < 2e-16 ***
## free.agent   1    7.7     7.7   40.45 7.1e-10 ***
## arb.elig     1  143.7   143.7  757.76 < 2e-16 ***
## arb          1    0.8     0.8    4.47  0.0353 *
## bat.av       1    9.1     9.1   48.23 2.2e-11 ***
```

```
## on.base      1    1.4     1.4    7.13  0.0080 **
## runs         1   37.3    37.3  196.48 < 2e-16 ***
## home.runs    1    1.7     1.7    9.14  0.0027 **
## rbi          1    3.8     3.8   19.88 1.1e-05 ***
## Residuals  314   59.5     0.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Reduced model

When we remove `home.runs` and `bat.av` from the model we find that `on.base` is not significant with a much lower VIF. We therefore also remove it (`arb` is also not significant, but we will consider the contractual variables later). Having removed these from the model, most of the VIFs are fairly reasonable: only `rbi` and `runs` are large, and these variables are significant, anyway.

```
# Determines observations we want to include
not_outlier = which(abs(student.res2) <= 3)
# Reduced model
bball.lm4 <- lm(log(salary) ~ free.elig + free.agent + arb.elig + arb + runs +
    rbi, data = bball, subset = not_outlier)
summary(bball.lm4)
```

```
##
## Call:
## lm(formula = log(salary) ~ free.elig + free.agent + arb.elig +
##     arb + runs + rbi, data = bball, subset = not_outlier)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.436 -0.289 -0.074  0.330  1.311
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.90404    0.05245   93.51  < 2e-16 ***
## free.elig1   1.71788    0.07164   23.98  < 2e-16 ***
## free.agent1 -0.24547    0.09370   -2.62   0.0092 **
## arb.elig1    1.42205    0.07990   17.80  < 2e-16 ***
## arb1        -0.13423    0.16561   -0.81   0.4182
## runs         0.00880    0.00170    5.18  3.9e-07 ***
## rbi          0.00739    0.00165    4.46  1.1e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.478 on 326 degrees of freedom
## Multiple R-squared:  0.835,  Adjusted R-squared:  0.832
## F-statistic:  274 on 6 and 326 DF,  p-value: <2e-16
```

```
vif(bball.lm4)
```

```
##  free.elig free.agent   arb.elig        arb       runs        rbi
##      1.782      1.292      1.442      1.163      3.551      3.483
```

If we examine the ANOVA table:

```
anova(bball.lm4)
```

```
## Analysis of Variance Table
##
## Response: log(salary)
##             Df Sum Sq Mean Sq F value  Pr(>F)
## free.elig    1  178.9   178.9   781.7 < 2e-16 ***
## free.agent   1    6.7     6.7    29.1 1.3e-07 ***
## arb.elig     1  137.2   137.2   599.7 < 2e-16 ***
## arb          1    0.0     0.0     0.2    0.65
## runs         1   49.3    49.3   215.5 < 2e-16 ***
## rbi          1    4.6     4.6    19.9 1.1e-05 ***
## Residuals  326   74.6     0.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We can test whether there is evidence that this model fits less well than the first. Recall that if Model 2 is nested in Model 1 then the appropriate F statistic is

$F^* = \frac{SSE(Model2) - SSE(Model1)/(p2-p1)}{SSE(Model2)/(n-(p2+1)} = \frac{(74.606-73.959)/3}{73.959/323} = 0.94$

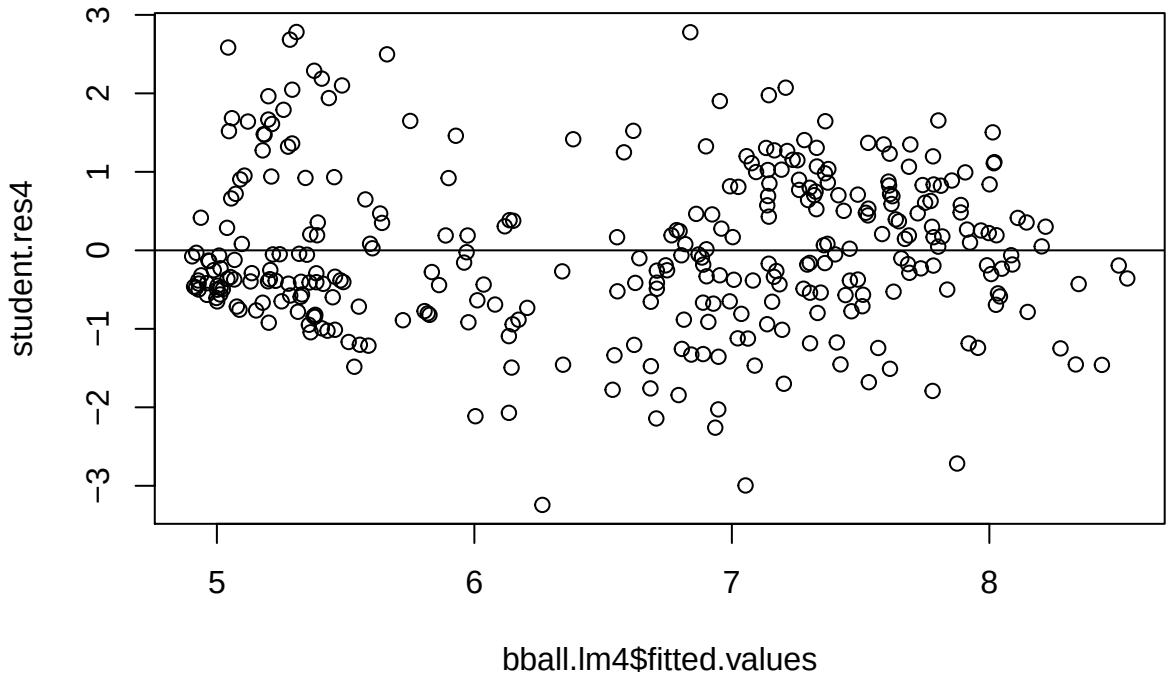
```
Fstar = ((74.606-73.959)/3)/(73.959/323)
Fstar
```

```
## [1] 0.9419
```

The null hypothesis of this test is that all regression coefficients associated with terms in Model 1 that are not in Model 2 are equal to zero. This statistic can be compared to an F distribution with 3 and 323 degrees of freedom. Here p1 is the number of model df for Model 1 and p2 is the number of model df for Model 2. The 0.05 critical value for the null distribution is 2.63, so there is no evidence that the reduced model fits substantially worse (we fail to reject the null hypothesis).

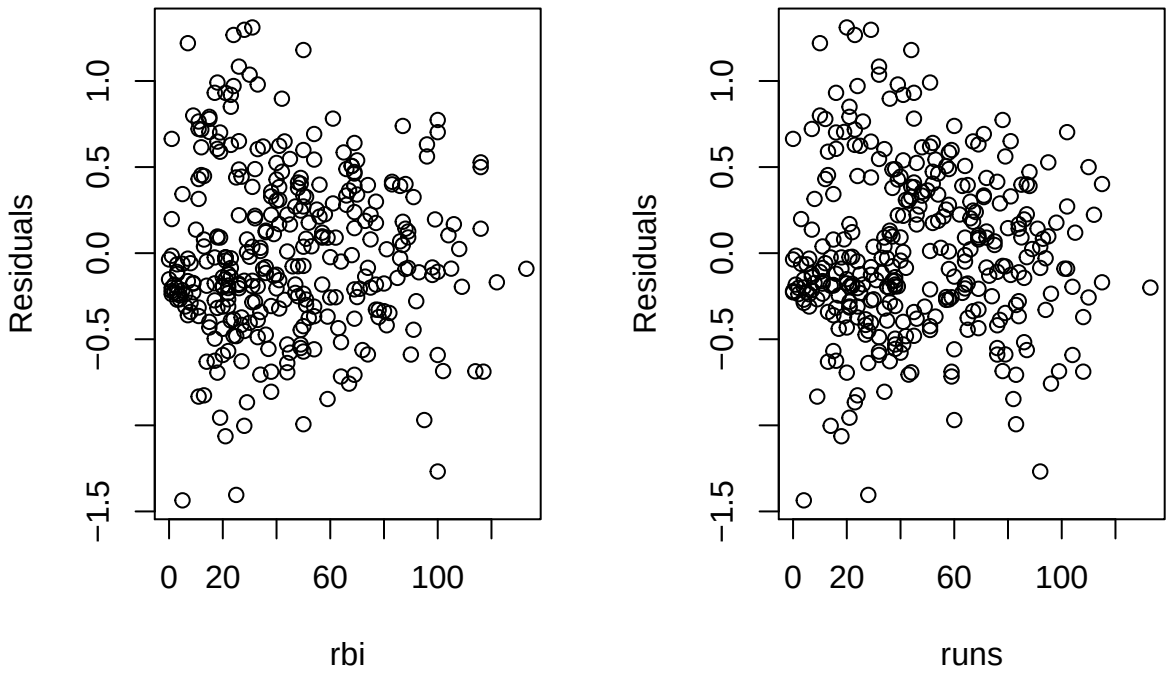
We can now examine the residuals. The studentized residual by predicted plot does not produce any obvious problems.

```
library(MASS)
student.res4 <-studres(bball.lm4)
plot(bball.lm4$fitted.values,student.res4)
abline(0,0)
```



We also want to look for curvature in the model. We therefore create a new scatterplots of the residuals plotted against each of the continuous explanatory variables . The residuals are output from `lm()` function and can be accessed by using the command `name.lm$residuals` where `name.lm` is the name of your linear model.

```
res=bball.lm4$residuals #Extracting the residuals
par(mfrow=c(1,2)) # Includes two plots in one figure
plot(bball$rbi[not_outlier],res, xlab='rbi',ylab='Residuals')
plot(bball$runs[not_outlier],res, xlab='runs',ylab='Residuals')
```



These plots do not appear to show any strong nonlinear effects.

Bonus Section: Interactions with Contractual Variables

It is important to understand how to include and interpret interaction terms in a MLR model. Please read through the following material on your own if it is not covered in lab. Questions on this material can be addressed during office hours.

Free agent eligibility, being a free agent, arbitration eligibility and going into arbitration are all variables that describe the contractual status of the player. All except arbitration itself appear to play an important role in determining salary. This is unsurprising since this is a time when players can re-negotiate their contracts. An interesting question is whether or not the players' performance affects their negotiation ability.

As a first way to examine this, we will re-do the residual by predicted plot, color-coding by contract status. This plot can be created in a couple of ways. Here we will consider creating a vector that indicates the contractural status of each player. We will call this vector `cont.stat`.

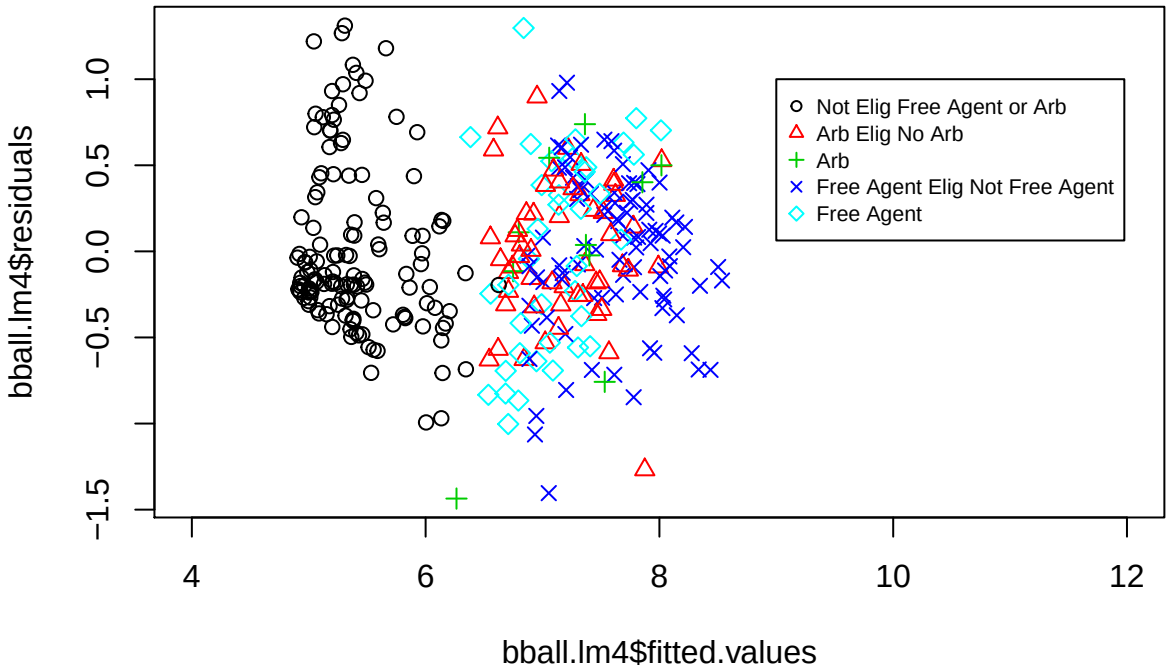
```
#Note the length of this vector is set to the number of observations in the original dataset
cont.stat = vector(mode="character",length=dim(bball)[1])

for ( i in 1:length(cont.stat)) {
  if ((bball$free.elig[i]=="0")&(bball$arb.elig[i]=="0")) {
    cont.stat[i] = "1"
  }
 if ((bball$arb.elig[i]=="1")&(bball$arb[i]=="0")) {
    cont.stat[i] = "2"
  }
 if ((bball$arb.elig[i]=="1")&(bball$arb[i]=="1")) {
    cont.stat[i] = "3"
  }
 if ((bball$free.elig[i]=="1")&(bball$free.agent[i]=="0")) {
    cont.stat[i] = "4"
  }
 if ((bball$free.elig[i]=="1")&(bball$free.agent[i]=="1")) {
    cont.stat[i] = "5"
  }
}
```

Description of `cont.stat` (Contractual Status)

Level	Description
1	Not eligible to be a free agent or for arbitration
2	Eligible for arbitration, no arbitration
3	Arbitration
4	Eligible to be a free agent, not a free agent
5	Free Agent

We will now create a plot that color codes by `cont.stat`. Notice that we created a level of `cont.stat` for every player in the original data set. So here we need to make sure to only use the `cont.stat` values for the observations that were not considered outliers. Also notice that as long as we use numbers for the levels of the categorical variable (as we did for `cont.stat`), the color (option `col`) and character (option `pch`) of each point can be specified by these numbers.

```r
plot(bball.lm4$fitted.values, bball.lm4$residuals, col = cont.stat[not_outlier],
    pch = as.numeric(cont.stat[not_outlier]), xlim = c(4, 12))
legend(9, 1, c("Not Elig Free Agent or Arb", "Arb Elig No Arb", "Arb", "Free Agent Elig Not Free Agent",
    "Free Agent"), cex = 0.7, col = c(1, 2, 3, 4, 5), pch = c(1, 2, 3, 4, 5))
```



Looking at this plot, we can see that there appear to be some trends with predicted salary, notably in the red triangles. This is a good indication that some interactions might be important.

To further investigate this possibility, we will include interactions between `rbi` and `runs` with all the contract variables. Interactions can be added to our model by including terms like `var1*var2` in the model formula to include an interaction between `var1` and `var2`.

```r
# Model with interactions
bball.lm5 <- lm(log(salary) ~ free.elig * runs + free.agent * runs + arb.elig *
    runs + arb * runs + free.elig * rbi + free.agent * rbi + arb.elig * rbi +
    arb * rbi, data = bball, subset = not_outlier)
summary(bball.lm5)
```

```
##
## Call:
## lm(formula = log(salary) ~ free.elig * runs + free.agent * runs +
##     arb.elig * runs + arb * runs + free.elig * rbi + free.agent *
##     rbi + arb.elig * rbi + arb * rbi, data = bball, subset = not_outlier)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4150 -0.3113 -0.0596  0.2832  1.4821
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       5.022350   0.065032   77.23  < 2e-16 ***
## free.elig1        1.619126   0.138609   11.68  < 2e-16 ***
```

```
## runs              0.008667   0.003150    2.75  0.00626 **
## free.agent1      -0.754031   0.207593   -3.63  0.00033 ***
## arb.elig1         1.298285   0.163152    7.96  3.1e-14 ***
## arb1             -0.937313   0.371593   -2.52  0.01214 *
## rbi               0.003534   0.003181    1.11  0.26744
## free.elig1:runs  -0.000169   0.004068   -0.04  0.96698
## runs:free.agent1  0.004413   0.005882    0.75  0.45365
## runs:arb.elig1    0.003109   0.005350    0.58  0.56152
## runs:arb1        -0.009555   0.008423   -1.13  0.25748
## free.elig1:rbi    0.003846   0.004034    0.95  0.34113
## free.agent1:rbi   0.006618   0.005279    1.25  0.21089
## arb.elig1:rbi     0.000892   0.005105    0.17  0.86141
## arb1:rbi          0.023893   0.009407    2.54  0.01156 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.464 on 318 degrees of freedom
## Multiple R-squared:  0.848,  Adjusted R-squared:  0.841
## F-statistic:  127 on 14 and 318 DF,  p-value: <2e-16
```

Of the resulting interactions, only `rbi-arb` is significant. But since we know that `runs` and `rbi` are correlated, we might want to investigate removing one or other from the model. To test this idea, we'll try a series of sequential tests, since the `runs` interactions come before the `rbi` interactions we expect that they will be significant before the `rbi` interactions are included. The `anova()` function in R will perform sequential tests for each predictor.

```
anova(bball.lm5)
```

```
## Analysis of Variance Table
##
## Response: log(salary)
##                  Df Sum Sq Mean Sq F value  Pr(>F)
## free.elig         1  178.9   178.9  829.28 < 2e-16 ***
## runs              1  111.1   111.1  514.84 < 2e-16 ***
## free.agent        1    0.6     0.6    2.63  0.1057
## arb.elig          1   81.4    81.4  377.51 < 2e-16 ***
## arb               1    0.2     0.2    1.03  0.3101
## rbi               1    4.6     4.6   21.15 6.1e-06 ***
## free.elig:runs    1    0.6     0.6    2.87  0.0914 .
## runs:free.agent   1    1.7     1.7    7.86  0.0054 **
## runs:arb.elig     1    1.0     1.0    4.70  0.0309 *
## runs:arb          1    0.5     0.5    2.36  0.1258
## free.elig:rbi     1    0.2     0.2    0.88  0.3490
## free.agent:rbi    1    0.3     0.3    1.57  0.2109
## arb.elig:rbi      1    0.3     0.3    1.16  0.2815
## arb:rbi           1    1.4     1.4    6.45  0.0116 *
## Residuals       318   68.6     0.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It turns out that our intuition is correct. When the variables are considered sequentially, the `runs-free.agent` and `runs-arb.elig` interactions are significant. For the sake of simplicity we might try removing some interactions from the model. Since `rbi*arb` is always significant, we'll keep the interactions between contract variables and `rbi` in the model and remove the interactions with `runs`.

A Final Model

We have now arrived at a model with `runs`, `rbi`, `free.elig`, `free.agent`, `arb.elig`, `arb`, and interactions between `rbi` and the contract variables. Let's have a look at some of our results.

```
# Model with only rbi interactions
bball.lm6 <- lm(log(salary) ~ runs + free.elig * rbi + free.agent * rbi + arb.elig *
    rbi + arb * rbi, data = bball, subset = not_outlier)

summary(bball.lm6)
```

```
##
## Call:
## lm(formula = log(salary) ~ runs + free.elig * rbi + free.agent *
##     rbi + arb.elig * rbi + arb * rbi, data = bball, subset = not_outlier)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4086 -0.3128 -0.0635  0.2845  1.4661
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      5.01932    0.06251   80.30  < 2e-16 ***
## runs             0.00914    0.00166    5.50  7.7e-08 ***
## free.elig1       1.60921    0.12845   12.53  < 2e-16 ***
## rbi              0.00312    0.00216    1.45  0.14883
## free.agent1     -0.68910    0.18695   -3.69  0.00027 ***
## arb.elig1        1.33990    0.14916    8.98  < 2e-16 ***
## arb1            -1.05026    0.35671   -2.94  0.00347 **
## free.elig1:rbi   0.00380    0.00233    1.63  0.10358
## rbi:free.agent1  0.00970    0.00331    2.93  0.00362 **
## rbi:arb.elig1    0.00327    0.00285    1.15  0.25180
## rbi:arb1         0.01528    0.00546    2.80  0.00541 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.463 on 322 degrees of freedom
## Multiple R-squared:  0.847,  Adjusted R-squared:  0.842
## F-statistic:  178 on 10 and 322 DF,  p-value: <2e-16
```

First, the R-squared for the regression is 0.84, meaning that 84% of the variability has been explained. This seems pretty good.

Examining the ANOVA table:

```
anova(bball.lm6)
```

```
## Analysis of Variance Table
##
## Response: log(salary)
##                 Df Sum Sq Mean Sq F value  Pr(>F)
## runs             1  199.1   199.1  929.05 < 2e-16 ***
## free.elig        1   90.9    90.9  423.98 < 2e-16 ***
```

```
## rbi              1    8.6     8.6   40.06 8.3e-10 ***
## free.agent       1    0.6     0.6    2.80  0.0951 .
## arb.elig         1   77.5    77.5  361.42 < 2e-16 ***
## arb              1    0.2     0.2    0.70  0.4029
## free.elig:rbi    1    0.9     0.9    4.39  0.0370 *
## rbi:free.agent   1    1.8     1.8    8.59  0.0036 **
## rbi:arb.elig     1    1.1     1.1    5.32  0.0217 *
## rbi:arb          1    1.7     1.7    7.84  0.0054 **
## Residuals      322   69.0     0.2
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We could again test that this is doing a better job than just the main effects model. However, since we have some significant interactions, we'll skip this step (you should work it out on your own).

Looking at the parameter estimates we see that the interactions with eligibility variables are not significant, but we will leave this for the moment. There do not appear to be further problems with the model (explore if you wish).

Interpreting the Final Model

Our parameter estimates are

```
summary(bball.lm6)
```

```
##
## Call:
## lm(formula = log(salary) ~ runs + free.elig * rbi + free.agent *
##     rbi + arb.elig * rbi + arb * rbi, data = bball, subset = not_outlier)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4086 -0.3128 -0.0635  0.2845  1.4661
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      5.01932    0.06251   80.30  < 2e-16 ***
## runs             0.00914    0.00166    5.50  7.7e-08 ***
## free.elig1       1.60921    0.12845   12.53  < 2e-16 ***
## rbi              0.00312    0.00216    1.45  0.14883
## free.agent1     -0.68910    0.18695   -3.69  0.00027 ***
## arb.elig1        1.33990    0.14916    8.98  < 2e-16 ***
## arb1            -1.05026    0.35671   -2.94  0.00347 **
## free.elig1:rbi   0.00380    0.00233    1.63  0.10358
## rbi:free.agent1  0.00970    0.00331    2.93  0.00362 **
## rbi:arb.elig1    0.00327    0.00285    1.15  0.25180
## rbi:arb1         0.01528    0.00546    2.80  0.00541 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.463 on 322 degrees of freedom
## Multiple R-squared:  0.847,  Adjusted R-squared:  0.842
## F-statistic:  178 on 10 and 322 DF,  p-value: <2e-16
```

Some interesting points on this model:

- `runs` and `rbi` are both positive, indicating that doing better increases your salary.
- Here the reference level for all contract variables is `0`. So for example, the positive coefficient for `free.elig` indicates that being eligible to be a free agent (`free.elig = 1`) increases your log salary by 1.6.
- Similarly, being eligible for arbitration increases your salary. Interestingly, actually being a free agent or taking arbitration decreases your salary.
- Neither eligibility interacted strongly with `rbi`, the p-values are not significant and the coefficients are relatively small. So being better doesn't change the boost you get from eligibility status.
- However, the interaction between performance and taking arbitration or becoming a free agent is significant. Since the effect is for taking these statuses, the positive coefficients suggest that if you do go into arbitration or become a free agent, you salary is improved by doing better.

Finally, remember that we have modeled log(`salary`). In order to get an idea of the size of these effects in dollar terms we recall that the model suggests

$Y = e^{\mathbf{X}\boldsymbol{\beta}+\epsilon} = e^{\beta_0} e^{\beta_1 X_1} e^{\beta_2 X_2} \dots e^{\epsilon}$

So that the effect of increasing X_1 by one unit is to multiply Y by e^{β_1} . To get a handle on what these changes are like, we need to exponentiate our coefficients.

```
exp(bball.lm6$coefficients)
```

```
##     (Intercept)            runs      free.elig1             rbi
##        151.3088          1.0092          4.9989          1.0031
##     free.agent1       arb.elig1            arb1  free.elig1:rbi
##          0.5020          3.8187          0.3498          1.0038
## rbi:free.agent1   rbi:arb.elig1        rbi:arb1
##          1.0097          1.0033          1.0154
```

By far the largest effects are associated with the contract variables.

- If you are eligible to be a free agent, the increase in log(`salary`) is 1.6. This means eligibility to be a free agent increases `salary` by a factor of almost 5, a 400% increase! A similar calculation shows that being eligible for arbitration results in almost a 300% increase in `salary`.
- Actually taking up being a free agent reduces your salary by a factor of .5 , and by .35 by taking up arbitration.
- The effect of each `rbi` is significantly higher if you take up arbitration. While each additional `rbi` only increases `salary` by .3% without arbitration, the increase with arbitration is approximately 1.8%, 6 times as much!

Note the Effect of our Choices

The choices we have made (leaving out variables etc) all affect the model that we end up with. Try making different choices and see where the model leads you. The great thing about R Markdown is that we now have a record of each choice we made (and why we made it) that can be reproduced in the future.

A Note on the Ordering of Categorical Values

By default R orders levels of categorical variables alphabetically and chooses the first to be the "reference" level. If you would like to change the reference level, you can re-order the levels of the categorical variable

using the `relevel()` function. Here is an example where we set the reference level for `free.elig` to `1` instead of `0`.

```
bball$free.elig=relevel(bball$free.elig,ref="1")
```

Now we can re-run the model to see what effect this has on the summary output.

```
bball.lm7 <- lm(log(salary) ~ runs + free.elig * rbi + free.agent * rbi + arb.elig *
    rbi + arb * rbi, data = bball, subset = not_outlier)  #Model with only rbi interactions
summary(bball.lm7)
```

```
##
## Call:
## lm(formula = log(salary) ~ runs + free.elig * rbi + free.agent *
##     rbi + arb.elig * rbi + arb * rbi, data = bball, subset = not_outlier)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4086 -0.3128 -0.0635  0.2845  1.4661
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      6.62854    0.11538   57.45  < 2e-16 ***
## runs             0.00914    0.00166    5.50  7.7e-08 ***
## free.elig0      -1.60921    0.12845  -12.53  < 2e-16 ***
## rbi              0.00693    0.00204    3.39  0.00079 ***
## free.agent1     -0.68910    0.18695   -3.69  0.00027 ***
## arb.elig1        1.33990    0.14916    8.98  < 2e-16 ***
## arb1            -1.05026    0.35671   -2.94  0.00347 **
## free.elig0:rbi  -0.00380    0.00233   -1.63  0.10358
## rbi:free.agent1  0.00970    0.00331    2.93  0.00362 **
## rbi:arb.elig1    0.00327    0.00285    1.15  0.25180
## rbi:arb1         0.01528    0.00546    2.80  0.00541 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.463 on 322 degrees of freedom
## Multiple R-squared:  0.847,  Adjusted R-squared:  0.842
## F-statistic:  178 on 10 and 322 DF,  p-value: <2e-16
```

Note, that all of the variables associated with `free.elig` now have a 0 instead of a 1 after them. This indicates that the missing level (1) is now the reference level for `free.elig`.