Introduction to linear regression

Batter up

The movie Moneyball focuses on the "quest for the secret of success in baseball". It follows a low-budget team, the Oakland Athletics, who believed that underused statistics, such as a player's ability to get on base, betterpredict the ability to score runs than typical statistics like home runs, RBIs (runs batted in), and batting average. Obtaining players who excelled in these underused statistics turned out to be much more affordable for the team.

In this lab we'll be looking at data from all 30 Major League Baseball teams and examining the linear relationship between runs scored in a season and a number of other player statistics. Our aim will be to summarize these relationships both graphically and numerically in order to find which variable, if any, helps us best predict a team's runs scored in a season.

The data

Let's load up the data for the 2011 season.

```
load("more/mlb11.RData")
```

In addition to runs scored, there are seven traditionally used variables in the data set: at-bats, hits, home runs, batting average, strikeouts, stolen bases, and wins. There are also three newer variables: on-base percentage, slugging percentage, and on-base plus slugging. For the first portion of the analysis we'll consider the seven traditional variables. At the end of the lab, you'll work with the newer variables on your own.

1. What type of plot would you use to display the relationship between `runs` and one of the other numerical variables? Plot this relationship using the variable `at_bats` as the predictor. Does the relationship look linear? If you knew a team's `at_bats`, would you be comfortable using a linear model to predict the number of runs?

At first, I would look at a scatter plot. This looks reasonably linear, but a more careful look is warranted.

```
library(tidyverse)
```

```
## Registered S3 method overwritten by 'rvest':
##    method            from
##    read_xml.response xml2
```

```
## -- Attaching packages --------------------------------------------------------

## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts -----------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```
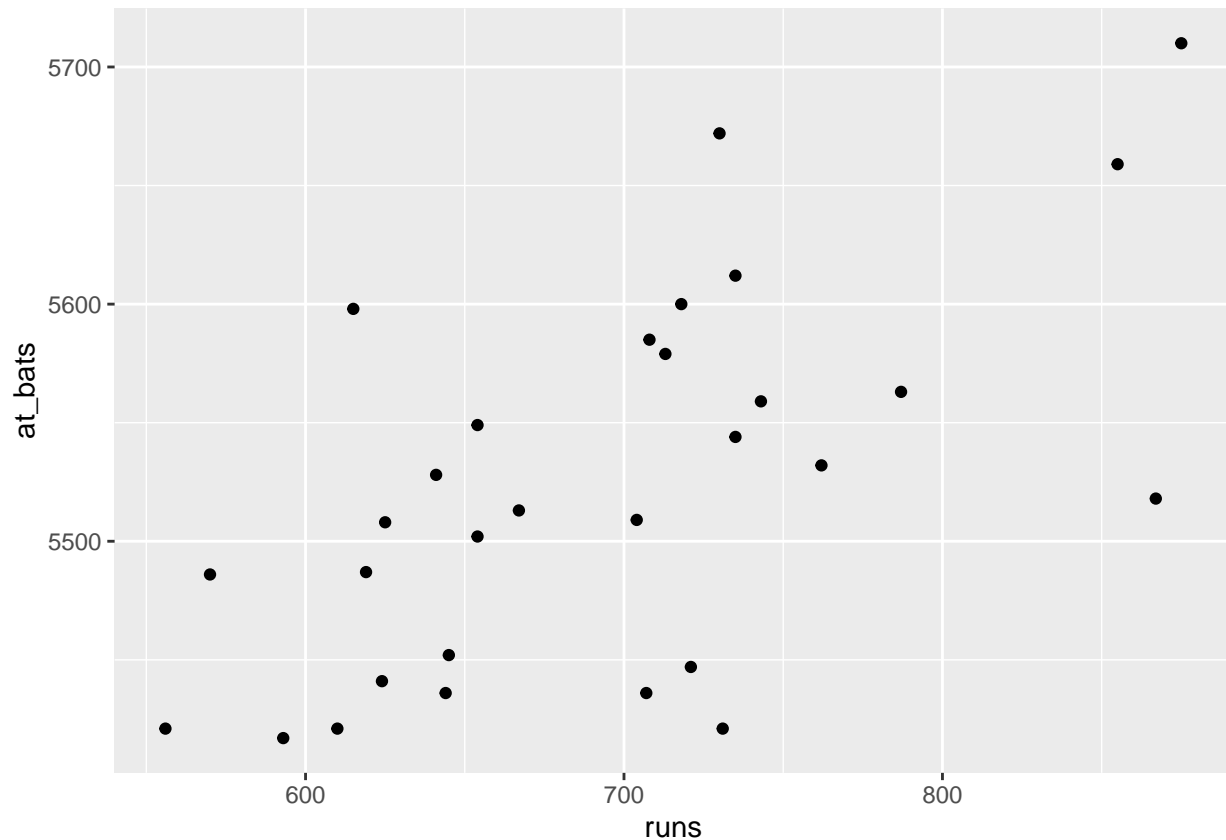
```
mlb11 %>%
  select(runs, at_bats) %>%
```

```
ggplot(aes(x=runs, y=at_bats)) +
geom_point()
```


If the relationship looks linear, we can quantify the strength of the relationship with the correlation coefficient.

```
cor(mlb11$runs, mlb11$at_bats)
```

```
## [1] 0.610627
```

Sum of squared residuals

Think back to the way that we described the distribution of a single variable. Recall that we discussed characteristics such as center, spread, and shape. It's also useful to be able to describe the relationship of two numerical variables, such as `runs` and `at_bats` above.

2. Looking at your plot from the previous exercise, describe the relationship between these two variables. Make sure to discuss the form, direction, and strength of the relationship as well as any unusual observations.

This data appears to be linear with a positive trend. It has a clear direction though the form may point to issues with constant variance. There is at least one outlier for a large number of runs with few at bats. A correlation coefficient of 0.67 is not bad given the type of data, so it merits a closer look.

Just as we used the mean and standard deviation to summarize a single variable, we can summarize the relationship between these two variables by finding the line that best follows their association. Use the following interactive function to select the line that you think does the best job of going through the cloud of points.

```r
names(mlb11)
```

```
##  [1] "team"         "runs"         "at_bats"      "hits"         "homeruns"
##  [6] "bat_avg"      "strikeouts"   "stolen_bases" "wins"         "new_onbase"
## [11] "new_slug"     "new_obs"
```

```r
plot_ss(x = mlb11$at_bats, y = mlb11$runs)
```
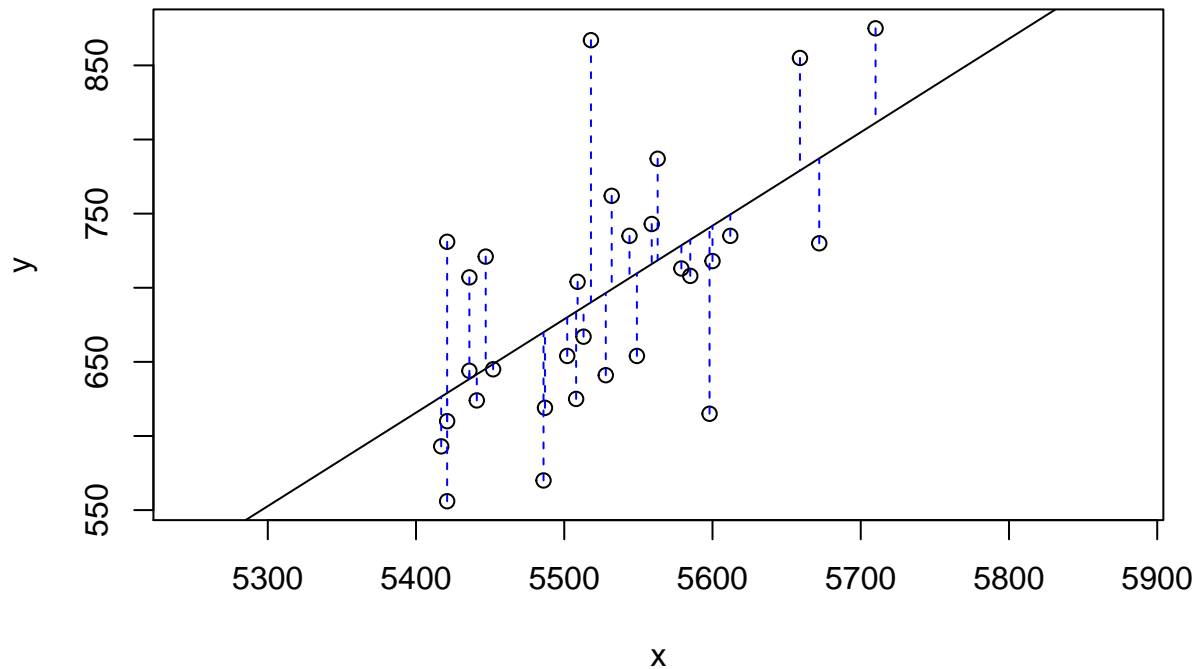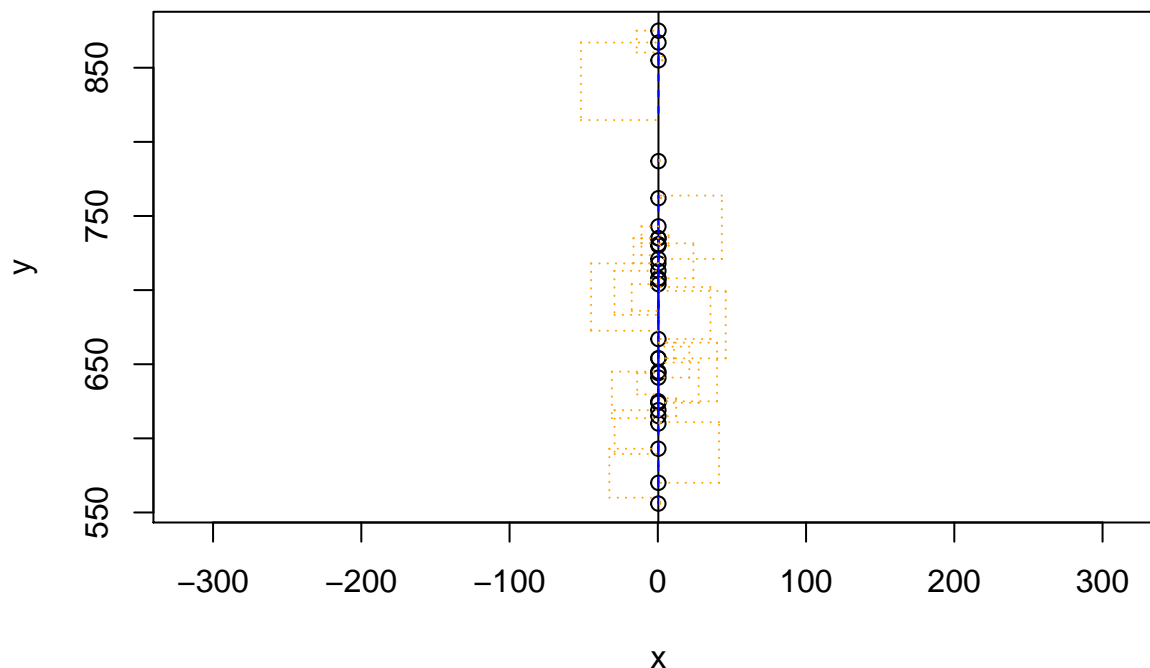


```
## Click two points to make a line.
## Call:
## lm(formula = y ~ x, data = pts)
##
## Coefficients:
## (Intercept)            x
##  -2789.2429       0.6305
##
## Sum of Squares:  123721.9
```

After running this command, you'll be prompted to click two points on the plot to define a line. Once you've done that, the line you specified will be shown in black and the residuals in blue. Note that there are 30 residuals, one for each of the 30 observations. Recall that the residuals are the difference between the observed values and the values predicted by the line:

$$e_i = y_i - \hat{y}_i$$

The most common way to do linear regression is to select the line that minimizes the sum of squared residuals. To visualize the squared residuals, you can rerun the plot command and add the argument `showSquares = TRUE`.

```r
plot_ss(x = mlb11$new_slug, y = mlb11$runs, showSquares = TRUE)
```

```
## Click two points to make a line.
## Call:
## lm(formula = y ~ x, data = pts)
##
## Coefficients:
## (Intercept)            x
##      -375.8       2681.3
##
## Sum of Squares:  20345.54
```

Note that the output from the **plot_ss** function provides you with the slope and intercept of your line as well as the sum of squares.

3. Using **plot_ss**, choose a line that does a good job of minimizing the sum of squares. Run the function several times. What was the smallest sum of squares that you got? How does it compare to your neighbors?

The variables `new_slug` and `new_onbase` are fairly low in SSE compared to `at_bats`. `bat_avg` is also pretty low comparatively.

The linear model

It is rather cumbersome to try to get the correct least squares line, i.e. the line that minimizes the sum of squared residuals, through trial and error. Instead we can use the **lm** function in R to fit the linear model (a.k.a. regression line).

```
m1 <- lm(runs~at_bats, data = mlb11)
```

The first argument in the function **lm** is a formula that takes the form y ~ x. Here it can be read that we want to make a linear model of **runs** as a function of **at_bats**. The second argument specifies that R should look in the **mlb11** data frame to find the **runs** and **at_bats** variables.

The output of **lm** is an object that contains all of the information we need about the linear model that was just fit. We can access this information using the summary function.

```
summary(m1)
```

```
##
## Call:
## lm(formula = runs ~ at_bats, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -125.58  -47.05  -16.59   54.40  176.87
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2789.2429   853.6957  -3.267 0.002871 **
## at_bats         0.6305     0.1545   4.080 0.000339 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 66.47 on 28 degrees of freedom
## Multiple R-squared:  0.3729, Adjusted R-squared:  0.3505
## F-statistic: 16.65 on 1 and 28 DF,  p-value: 0.0003388
```

Let's consider this output piece by piece. First, the formula used to describe the model is shown at the top.
After the formula you find the five-number summary of the residuals. The "Coefficients" table shown next is
key; its first column displays the linear model's y-intercept and the coefficient of `at_bats`. With this table,
we can write down the least squares regression line for the linear model:

$$\hat{y} = -2789.2429 + 0.6305 * atbats$$

One last piece of information we will discuss from the summary output is the Multiple R-squared, or more
simply, R^2. The R^2 value represents the proportion of variability in the response variable that is explained
by the explanatory variable. For this model, 37.3% of the variability in runs is explained by at-bats.

4. Fit a new model that uses `homeruns` to predict `runs`. Using the estimates from the R output, write the
   equation of the regression line. What does the slope tell us in the context of the relationship between
   success of a team and its home runs?

```
homers<-lm(runs ~homeruns,data=mlb11)
summary(homers)
```

```
##
## Call:
## lm(formula = runs ~ homeruns, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -91.615 -33.410   3.231  24.292 104.631
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 415.2389    41.6779   9.963 1.04e-10 ***
## homeruns      1.8345     0.2677   6.854 1.90e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 51.29 on 28 degrees of freedom
```

```
## Multiple R-squared:  0.6266, Adjusted R-squared:  0.6132
## F-statistic: 46.98 on 1 and 28 DF,  p-value: 1.9e-07
```

Equation

```
paste0("runs = ",homers$coefficients[2],'*homeruns + ',homers$coefficients[1])
```
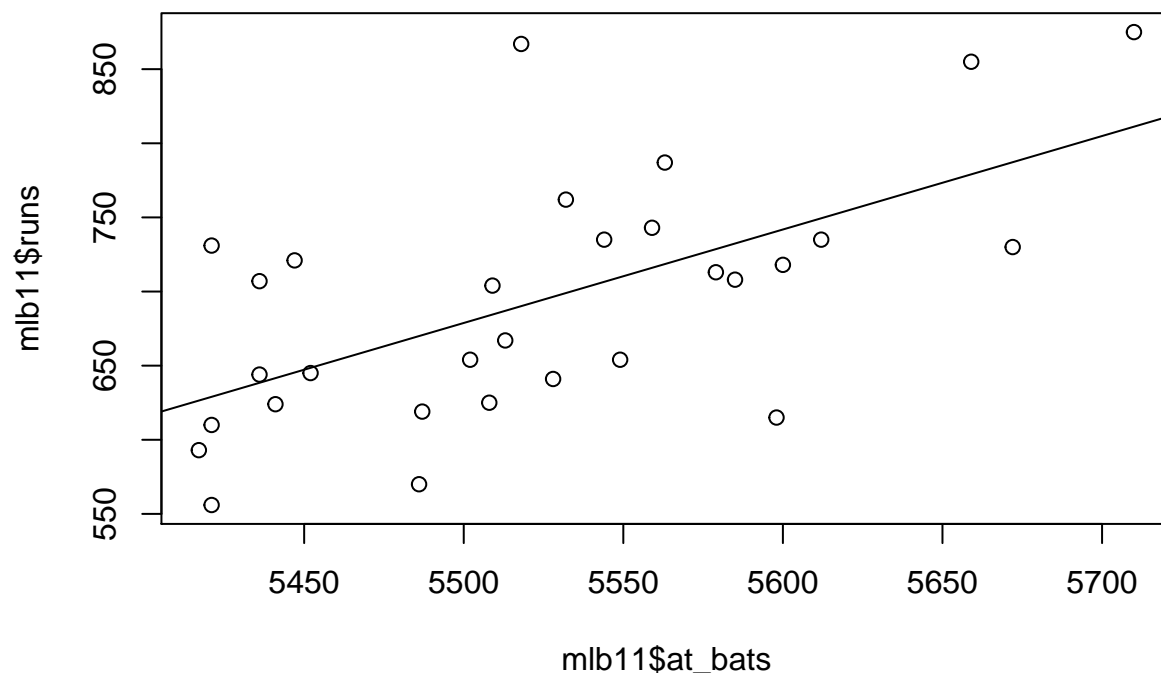
```
## [1] "runs = 1.83454161962397*homeruns + 415.238884915723"
```

This is cool, the slope tells us that for every homerun there is nearly a 2 run increase.

Prediction and prediction errors

Let's create a scatterplot with the least squares line laid on top.

```
plot(mlb11$runs ~ mlb11$at_bats)
abline(m1)
```


The function `abline` plots a line based on its slope and intercept. Here, we used a shortcut by providing the model `m1`, which contains both parameter estimates. This line can be used to predict y at any value of x. When predictions are made for values of x that are beyond the range of the observed data, it is referred to as *extrapolation* and is not usually recommended. However, predictions made within the range of the data are more reliable. They're also used to compute the residuals.

5. If a team manager saw the least squares regression line and not the actual data, how many runs would he or she predict for a team with 5,578 at-bats? Is this an overestimate or an underestimate, and by how much? In other words, what is the residual for this prediction?

```
pred<-m1$coefficients[2]*5578 + m1$coefficients[1]
paste("predicted runs", pred)
```

```
## [1] "predicted runs 727.964974609686"
```

```
mlb11 %>% select(at_bats, runs) %>%filter(at_bats >= 5570) %>% arrange(at_bats)
```

```
##   at_bats runs
## 1    5579  713
```

```
## 2      5585   708
## 3      5598   615
## 4      5600   718
## 5      5612   735
## 6      5659   855
## 7      5672   730
## 8      5710   875
```

```
res <- 713 - pred
paste0("closest residual= ", res)
```

```
[1] "closest residual= -14.9649746096861"
```

**This model would return an overestimate of the numbers of runs to expect given the at bats.**

### Model diagnostics

To assess whether the linear model is reliable, we need to check for (1) linearity, (2) nearly normal residuals, and (3) constant variability.

*Linearity*: You already checked if the relationship between runs and at-bats is linear using a scatterplot. We should also verify this condition with a plot of the residuals vs. at-bats. Recall that any code following a # is intended to be a comment that helps understand the code but is ignored by R.

```
plot(m1$residuals ~ mlb11$at_bats)
abline(h = 0, lty = 3) # adds a horizontal dashed line at y = 0
```
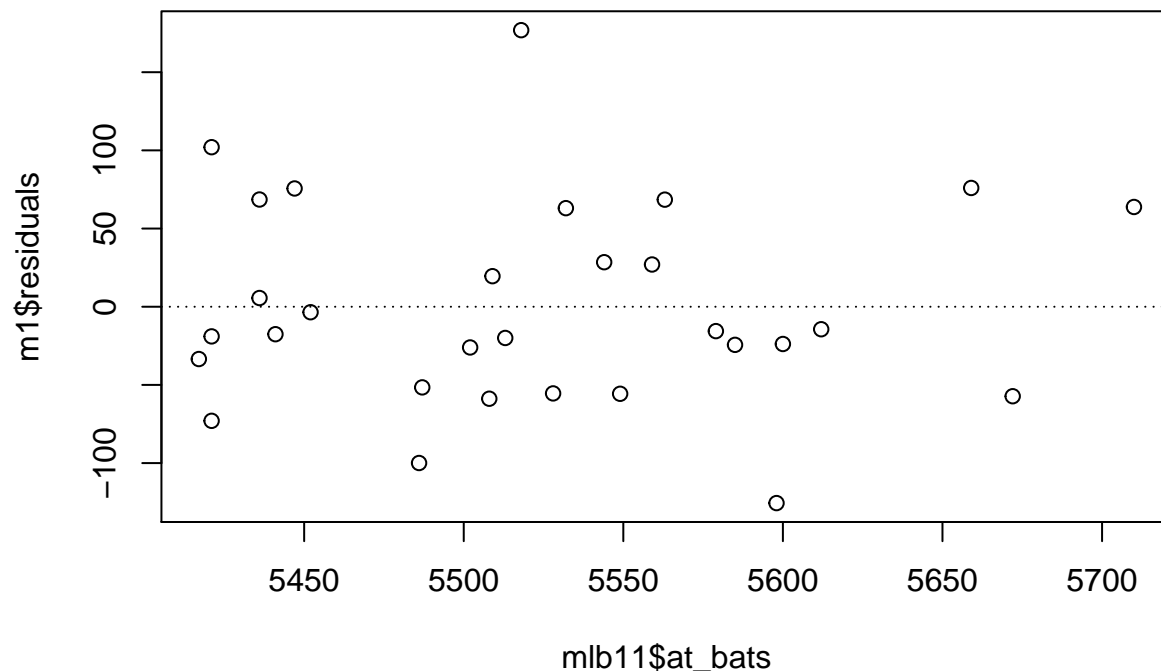


6. Is there any apparent pattern in the residuals plot? What does this indicate about the linearity of the relationship between runs and at-bats?

**This residual plots looks pretty good in that it is distributed around zero fairly well. There is an outlier at around 5520 at bats. This plots also seems to have a constant variance.**

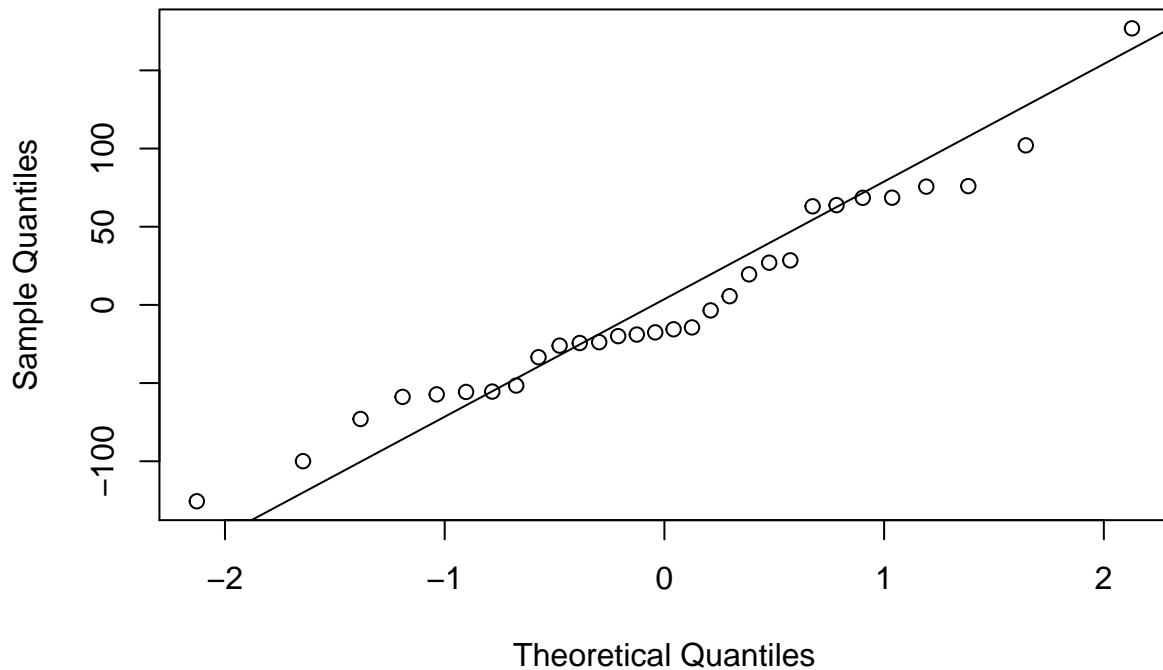*Nearly normal residuals*: To check this condition, we can look at a histogram

```r
hist(m1$residuals)
```

Histogram of m1$residuals
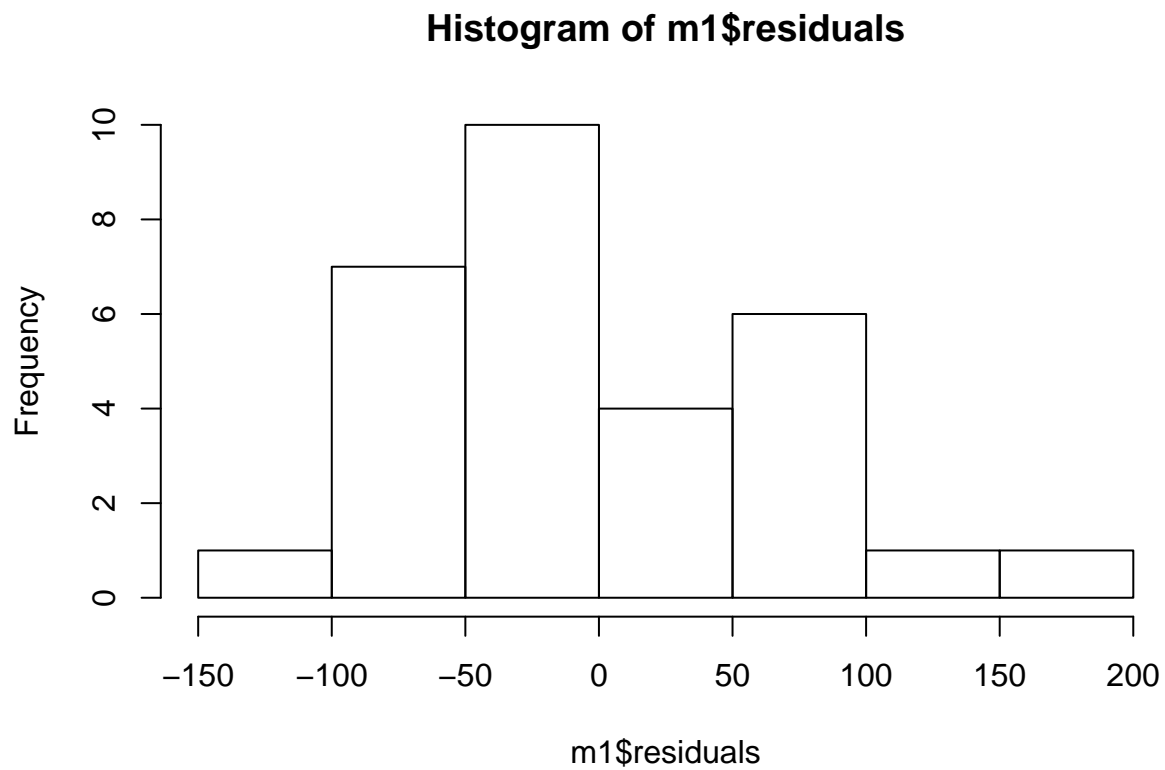
or a normal probability plot of the residuals.

```r
qqnorm(m1$residuals)
qqline(m1$residuals)  # adds diagonal line to the normal prob plot
```

Normal Q–Q Plot

7. Based on the histogram and the normal probability plot, does the nearly normal residuals condition appear to be met?

depending on how "nearly" we are okay with, this could meet that standard. The histogram seems pretty normal but qqplot is less great. Generally, the nearly normal assumption seems pretty generous, so we can keep it for now.

Constant variability:

8. Based on the plot in (1), does the constant variability condition appear to be met?

Other than the outlier, yes.

On Your Own

- Choose another traditional variable from `mlb11` that you think might be a good predictor of `runs`. Produce a scatterplot of the two variables and fit a linear model. At a glance, does there seem to be a linear relationship?

I'll choose the old metric of batting average since, I hope, there is some value to it.

```
lm_bat<- lm(runs ~ bat_avg, data=mlb11)
summary(lm_bat)

##
## Call:
## lm(formula = runs ~ bat_avg, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -94.676 -26.303  -5.496  28.482 131.113
```

```
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -642.8      183.1  -3.511  0.00153 **
## bat_avg        5242.2      717.3   7.308 5.88e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 49.23 on 28 degrees of freedom
## Multiple R-squared:  0.6561, Adjusted R-squared:  0.6438
## F-statistic: 53.41 on 1 and 28 DF,  p-value: 5.877e-08
```

```r
mlb11 %>%
  ggplot(aes(x=bat_avg, y=runs)) +
  geom_point(color="black") +
  geom_abline(slope = lm_bat$coefficients[2],
              intercept = lm_bat$coefficients[1],
              color="red")
```
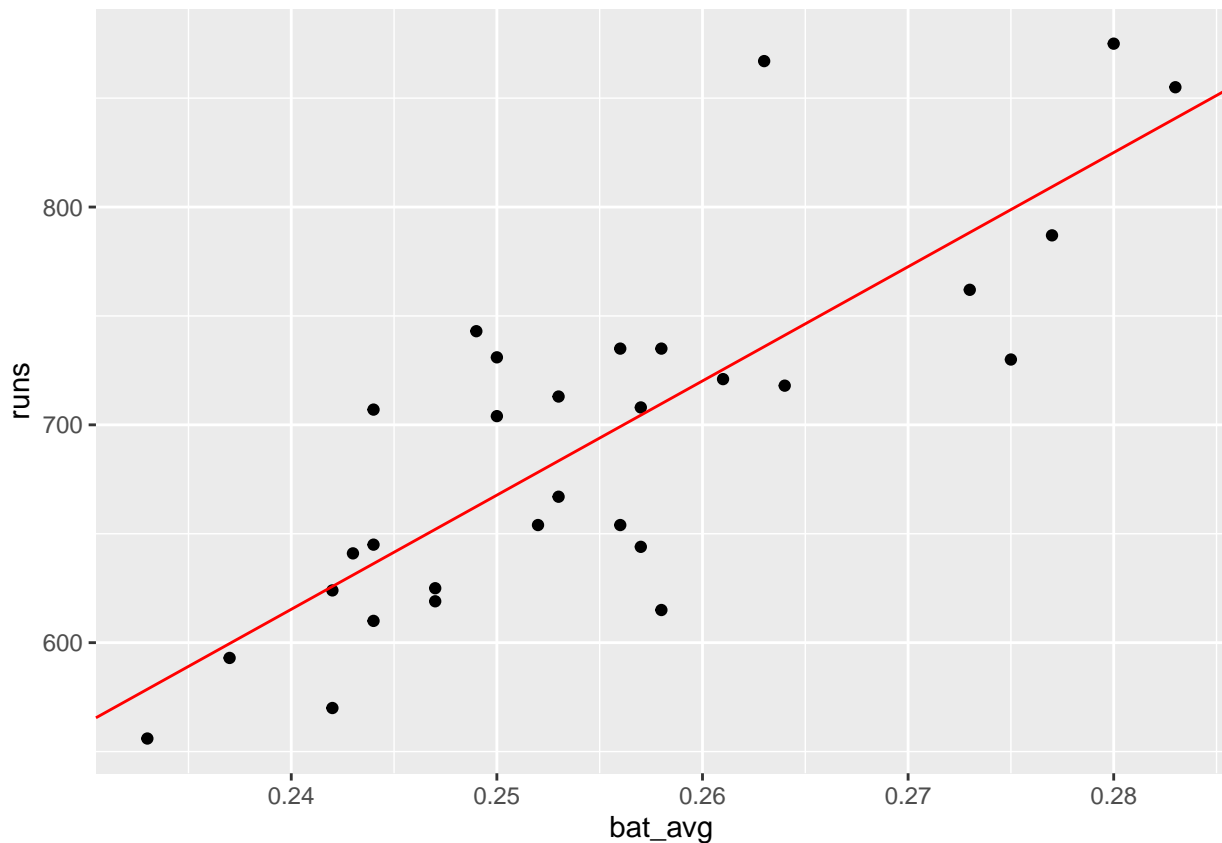

Looks pretty good. A few ood points above and below.

- How does this relationship compare to the relationship between `runs` and `at_bats`? Use the R^2 values from the two model summaries to compare. Does your variable seem to predict `runs` better than `at_bats`? How can you tell?

Batting average is very comparable to at_bats in terms of R squared, residuals, and standard error coefficient ratio. If at_bats is a good predictor then batting average is as well.

- Now that you can summarize the linear relationship between two variables, investigate the relationships

between `runs` and each of the other five traditional variables. Which variable best predicts `runs`? Support your conclusion using the graphical and numerical methods we've discussed (for the sake of conciseness, only include output for the best variable, not all five).

```
names(mlb11)
```

```
##  [1] "team"         "runs"         "at_bats"      "hits"         "homeruns"
##  [6] "bat_avg"      "strikeouts"   "stolen_bases" "wins"         "new_onbase"
## [11] "new_slug"     "new_obs"
```

```
# regress and print all metrics
# metric_list<-c("hits", "strikeouts", "stolen_bases", "wins","homeruns")
# for (i in metric_list){
#    print(noquote(i))
#    linreg<-lm(runs ~ mlb11[,i], data=mlb11)
#    print(summary(linreg))
# }
# Homeruns was the best, which I already have from above.
summary(homers)
```

```
##
## Call:
## lm(formula = runs ~ homeruns, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -91.615 -33.410   3.231  24.292 104.631
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 415.2389    41.6779   9.963 1.04e-10 ***
## homeruns      1.8345     0.2677   6.854 1.90e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 51.29 on 28 degrees of freedom
## Multiple R-squared:  0.6266, Adjusted R-squared:  0.6132
## F-statistic: 46.98 on 1 and 28 DF,  p-value: 1.9e-07
```

```
mlb11 %>%
  ggplot(aes(x=homeruns, y=runs)) +
  geom_point(color="black") +
  geom_abline(slope = homers$coefficients[2],
              intercept = homers$coefficients[1],
              color="red")
```
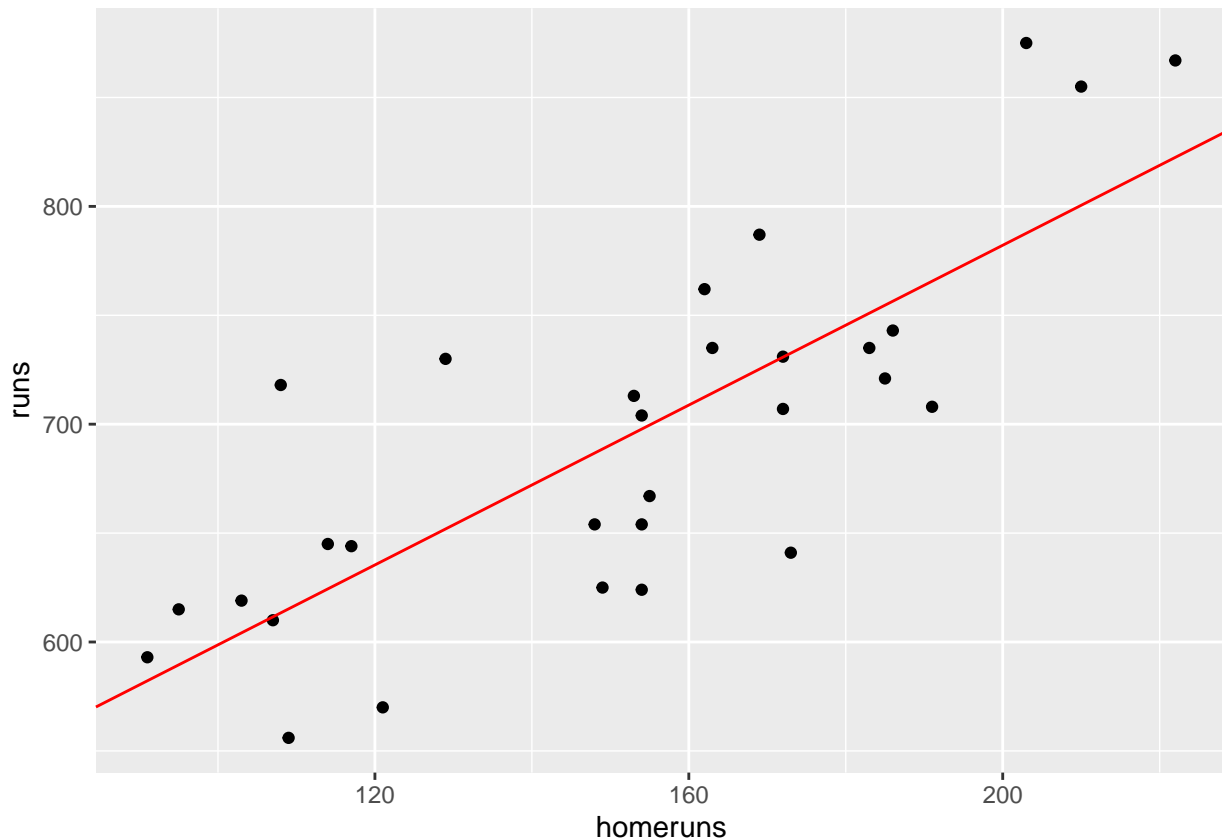
- Now examine the three newer variables. These are the statistics used by the author of *Moneyball* to predict a teams success. In general, are they more or less effective at predicting runs that the old variables? Explain using appropriate graphical and numerical evidence. Of all ten variables we've analyzed, which seems to be the best predictor of **runs**? Using the limited (or not so limited) information you know about these baseball statistics, does your result make sense?

```r
#"new_onbase"    "new_slug"      "new_obs"
metric_list<-c("new_onbase", "new_slug","new_obs")
for (i in metric_list){
  print(i)
  linreg<-lm(runs ~ mlb11[,i], data=mlb11)
  print(summary(linreg))
}
```

```
## [1] "new_onbase"
##
## Call:
## lm(formula = runs ~ mlb11[, i], data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -58.270 -18.335   3.249  19.520  69.002
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1118.4      144.5  -7.741 1.97e-08 ***
## mlb11[, i]    5654.3      450.5  12.552 5.12e-13 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32.61 on 28 degrees of freedom
## Multiple R-squared:  0.8491, Adjusted R-squared:  0.8437
## F-statistic: 157.6 on 1 and 28 DF,  p-value: 5.116e-13
##
## [1] "new_slug"
##
## Call:
## lm(formula = runs ~ mlb11[, i], data = mlb11)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -45.41 -18.66  -0.91  16.29  52.29
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -375.80      68.71   -5.47 7.70e-06 ***
## mlb11[, i]   2681.33     171.83   15.61 2.42e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26.96 on 28 degrees of freedom
## Multiple R-squared:  0.8969, Adjusted R-squared:  0.8932
## F-statistic: 243.5 on 1 and 28 DF,  p-value: 2.42e-15
##
## [1] "new_obs"
##
## Call:
## lm(formula = runs ~ mlb11[, i], data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -43.456 -13.690   1.165  13.935  41.156
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -686.61      68.93  -9.962 1.05e-10 ***
## mlb11[, i]   1919.36      95.70  20.057  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.41 on 28 degrees of freedom
## Multiple R-squared:  0.9349, Adjusted R-squared:  0.9326
## F-statistic: 402.3 on 1 and 28 DF,  p-value: < 2.2e-16
```

The three new metrics; new_onbase, new_slug, new_obs are huge improvements of the previous predictors. R, R squared, residual distribution are all significantly better than the top metrics from the past (homeruns, at bats). So the top predictors are the all three new one plus homeruns and at bats where homeruns and at bats are the worst of those five.
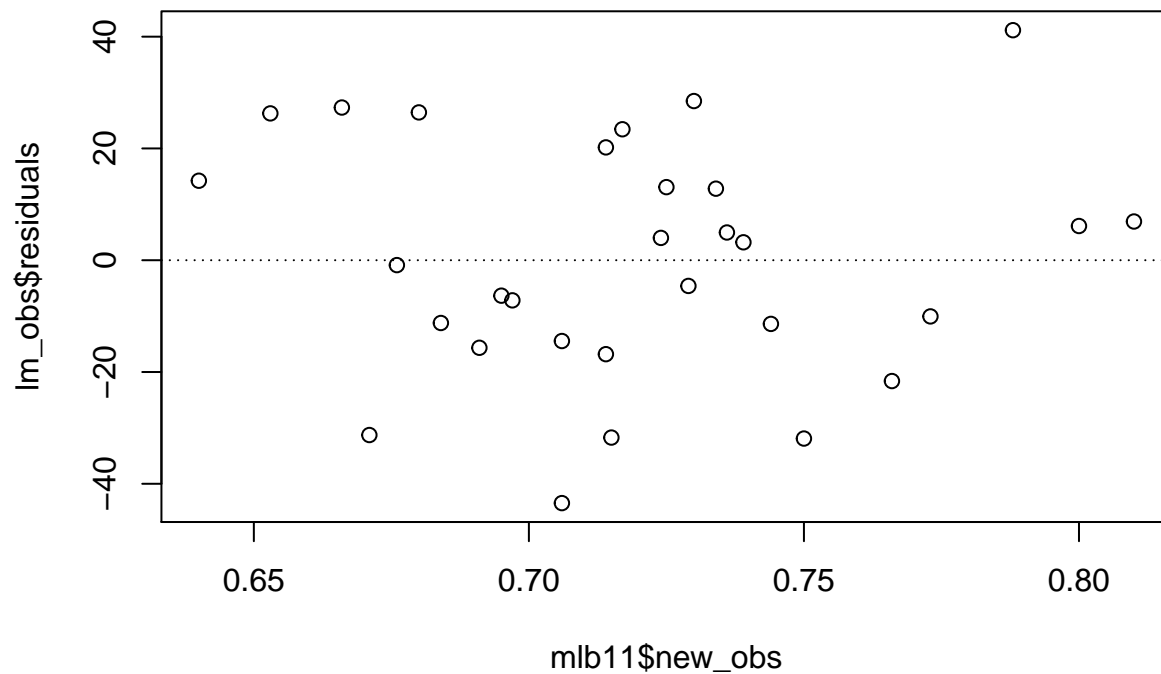
- Check the model diagnostics for the regression model with the variable you decided was the best predictor for runs.

```
lm_obs<-lm(runs ~ new_obs, data=mlb11)

hist(lm_obs$residuals)
```

Histogram of lm_obs$residuals
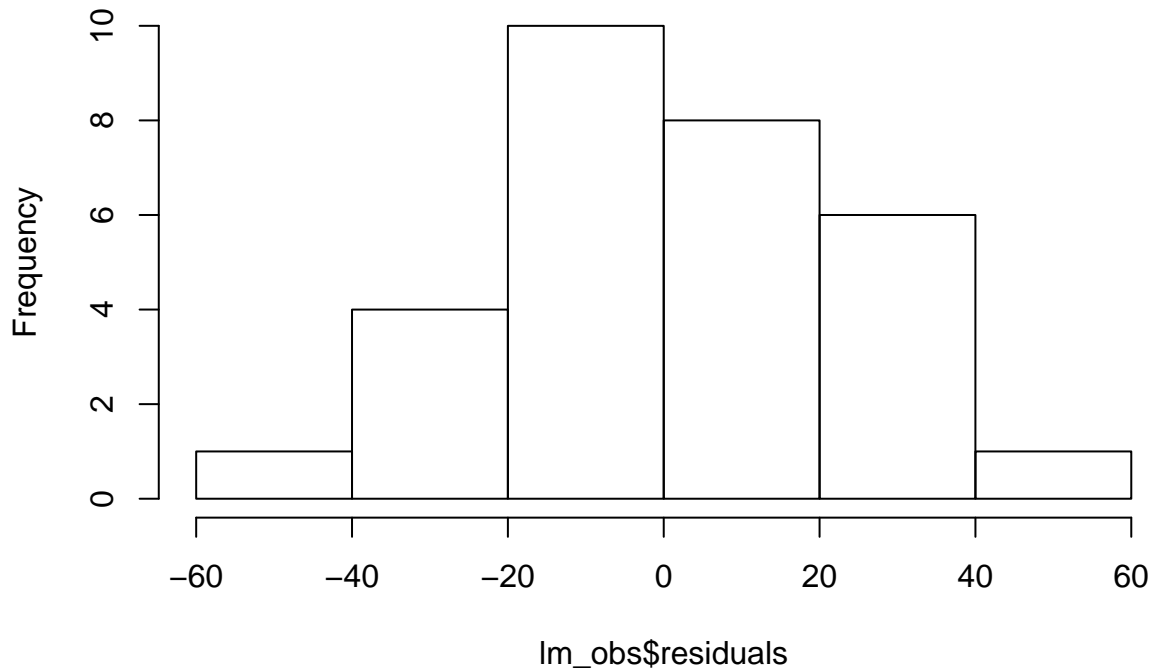

```
plot(lm_obs$residuals ~ mlb11$new_obs)
abline(h = 0, lty = 3)
```


New On Base and Sluging is the top metric, and upon further inquery, it continues to look

very good.