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.

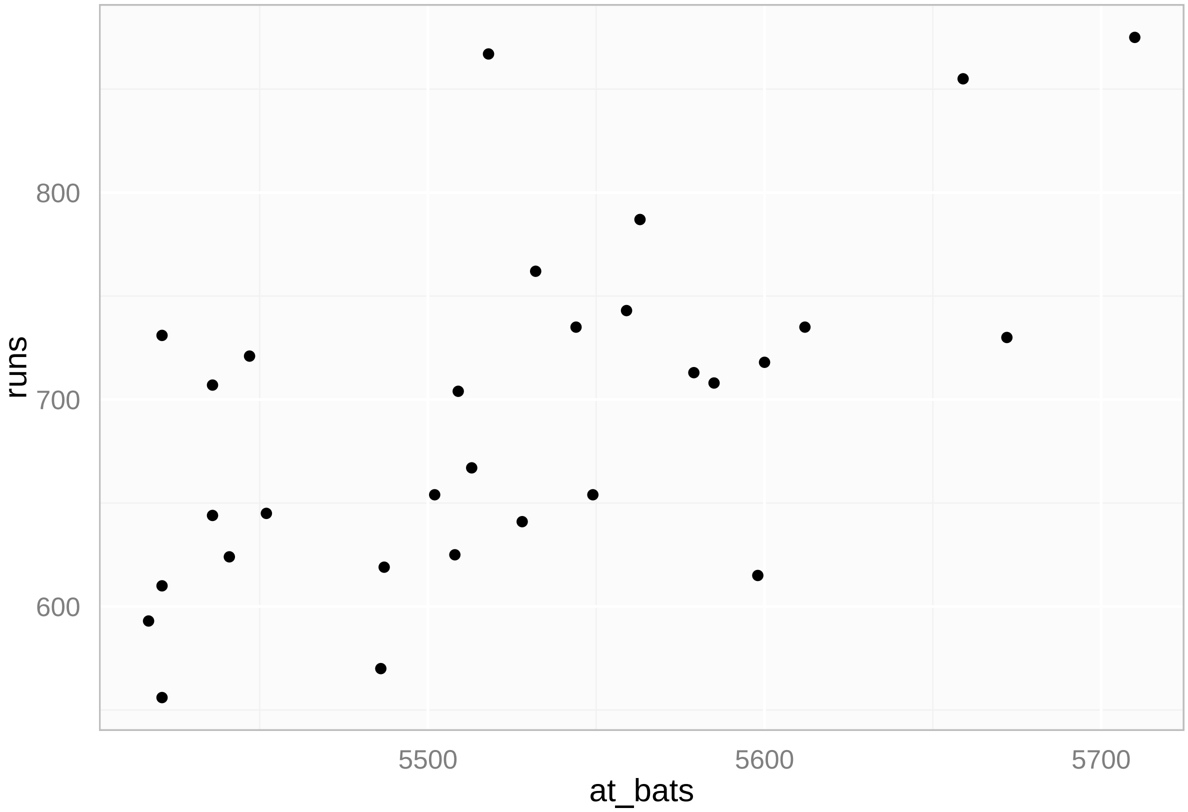
```
load("more/mlb11.RData")
```

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?

   Based on the visualization below, there does seem to be a positive relationship between `at_bats` and `runs`. The spread is fairly wide, so a prediction might not be very useful alone, but it appears strong enough to get a flavor.

```
g1 <- ggplot(data=mlb11, aes(x=at_bats, y=runs)) + geom_point() + myTheme
g1
```



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

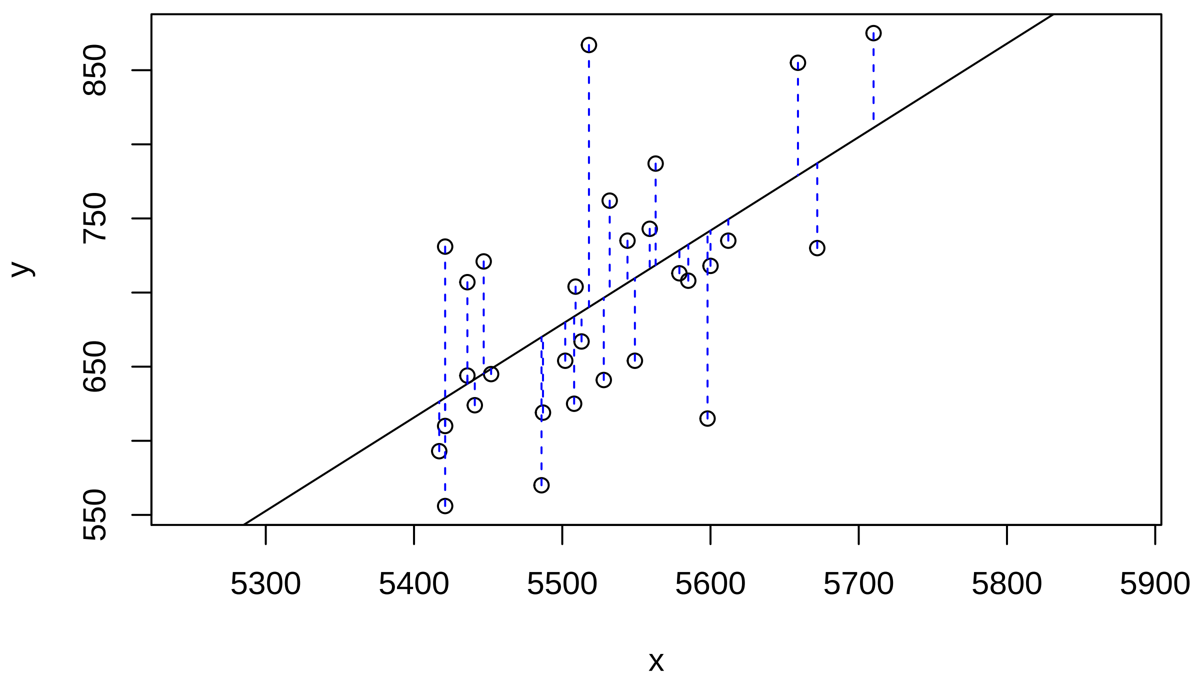
Think back to the way that we described the distribution of a single variable. Recall that we discussed characteristics such as center, spread, and shape. It's also useful to be able to describe the relationship of two numerical variables, such as `runs` and `at_bats` above.

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.

   There is a positive relationship. As `at_bats` increase, `runs` tend to also. This is not an absolute relationship, nor a tight extremely linear one. None the less, there is a trend in the data from lower left to upper right.

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
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`.

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

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?

   Smallest sum of squares was 145516.1. Largest was 183082.8

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?

```
m2 <- lm(runs ~ homeruns, data = mlb11)
summary(m2)
```

```
##
## 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
```

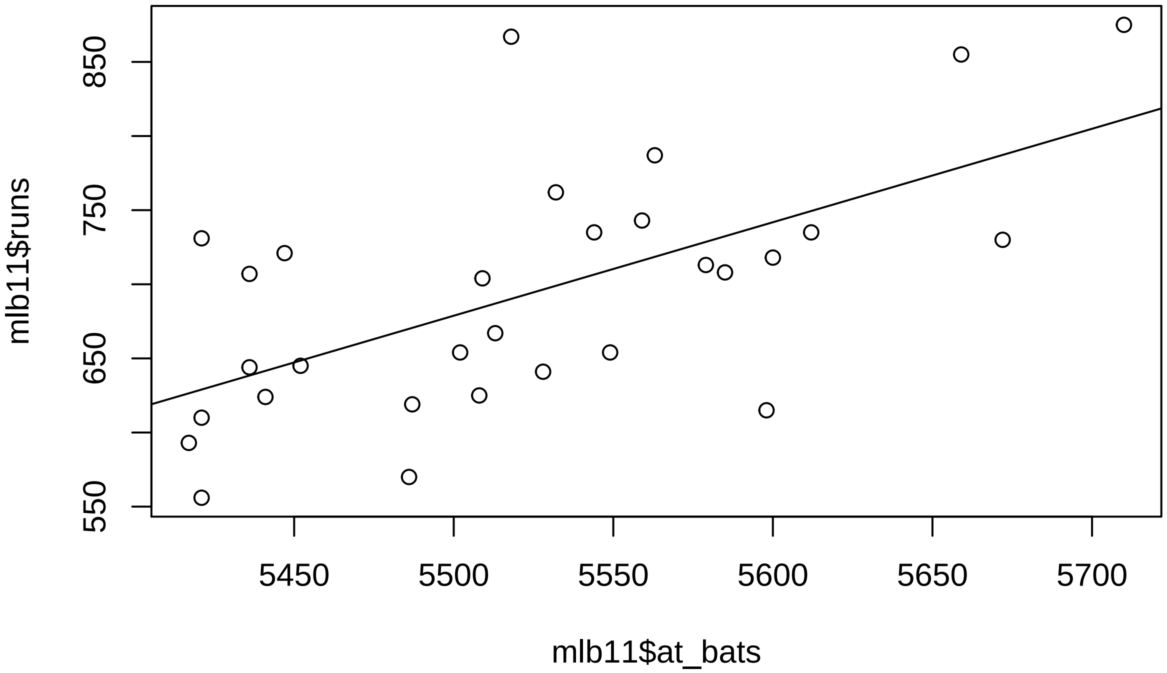
$$\hat{y} = 415.2388849 + 1.8345416 * homeruns$$

The slope of 1.8345416 tells us there is a 1.8345416 run increase for every 1 homerun increase. As compare to the `at_bats` slope, the homeruns slope is steeper.

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?

```
atbats <- 5578
prRuns <- -2789.2429 + 0.6305 * atbats
prRuns
```

```
## [1] 727.6861
```

```
mlb11[mlb11$atbats == atbats,]
```

```
##  [1] team         runs         at_bats      hits         homeruns    
##  [6] bat_avg      strikeouts   stolen_bases wins         new_onbase  
## [11] new_slug     new_obs     
## <0 rows> (or 0-length row.names)
```

Due to the prediction being based on and possibly exactly on the regression line, the residual would zero or close to it since the residual is the different between a data point and the

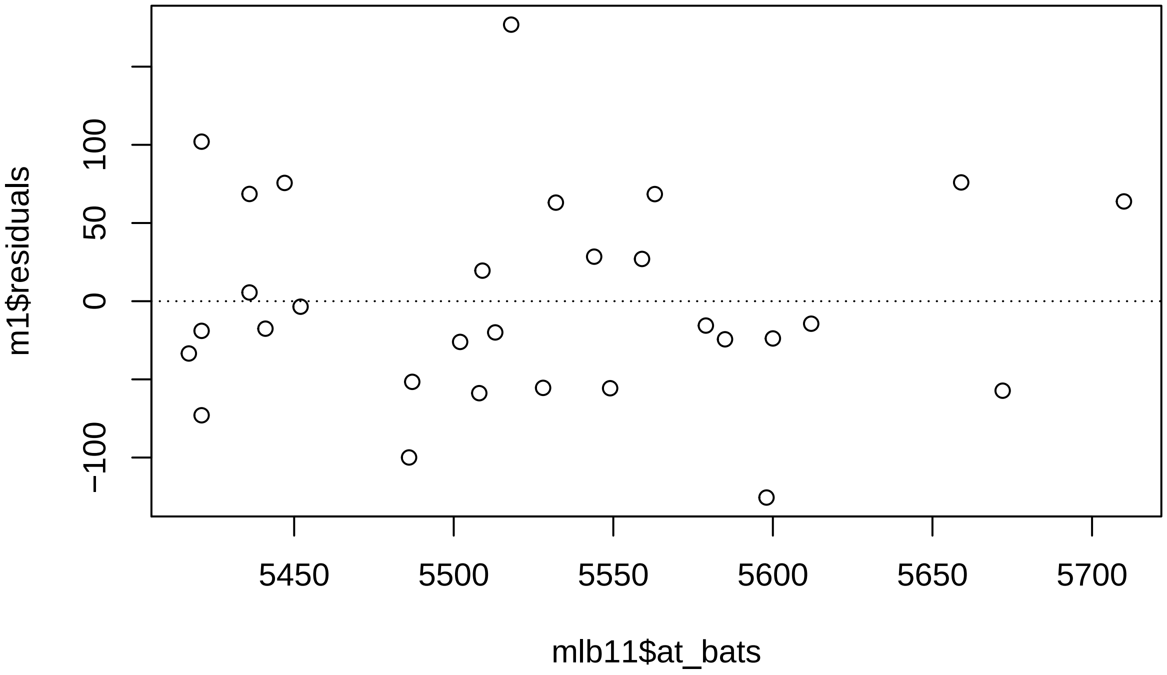
regression line. In terms of over or underestimating, the regression line is a little high and a little low in this range. depending on which side of the 5,578 point we want to use to measure the estimate.

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
```

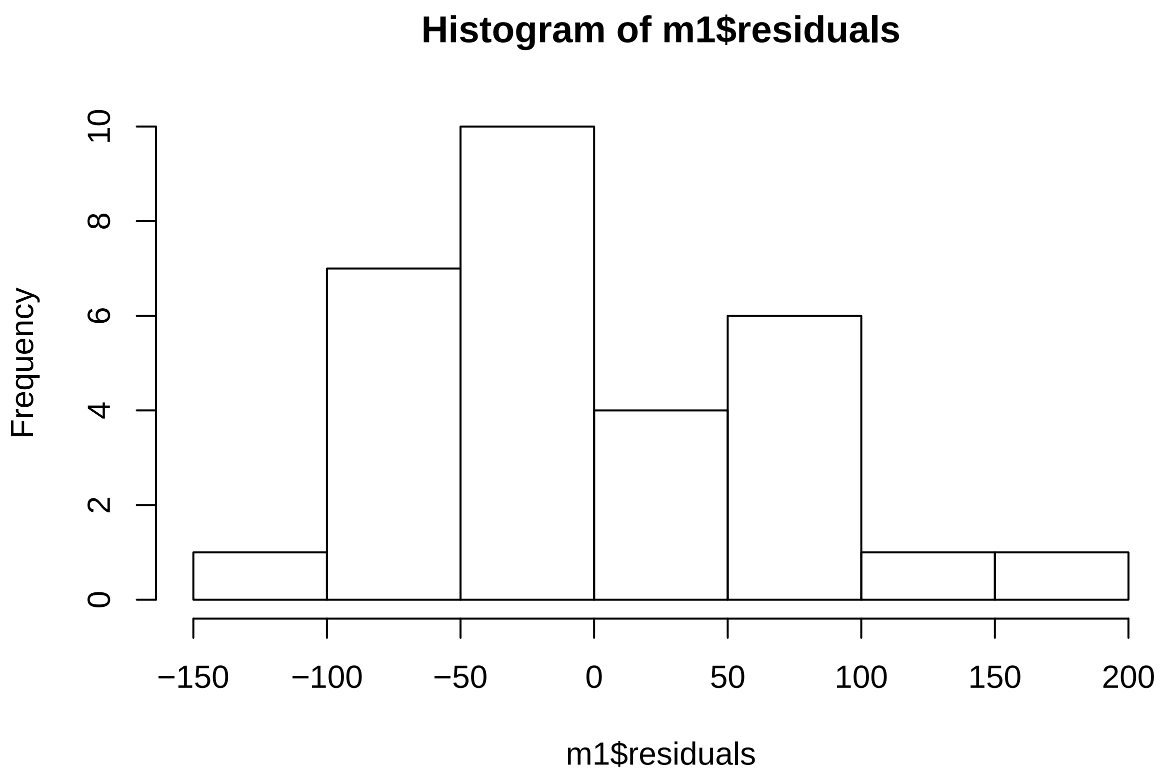


6. Is there any apparent pattern in the residuals plot? What does this indicate about the linearity of the relationship between runs and at-bats?

   There seems to be more points on the left versus the right side, but generally constant variability with a similar amount of point above and below the line. This suggests a somewhat linear though wide relationship.

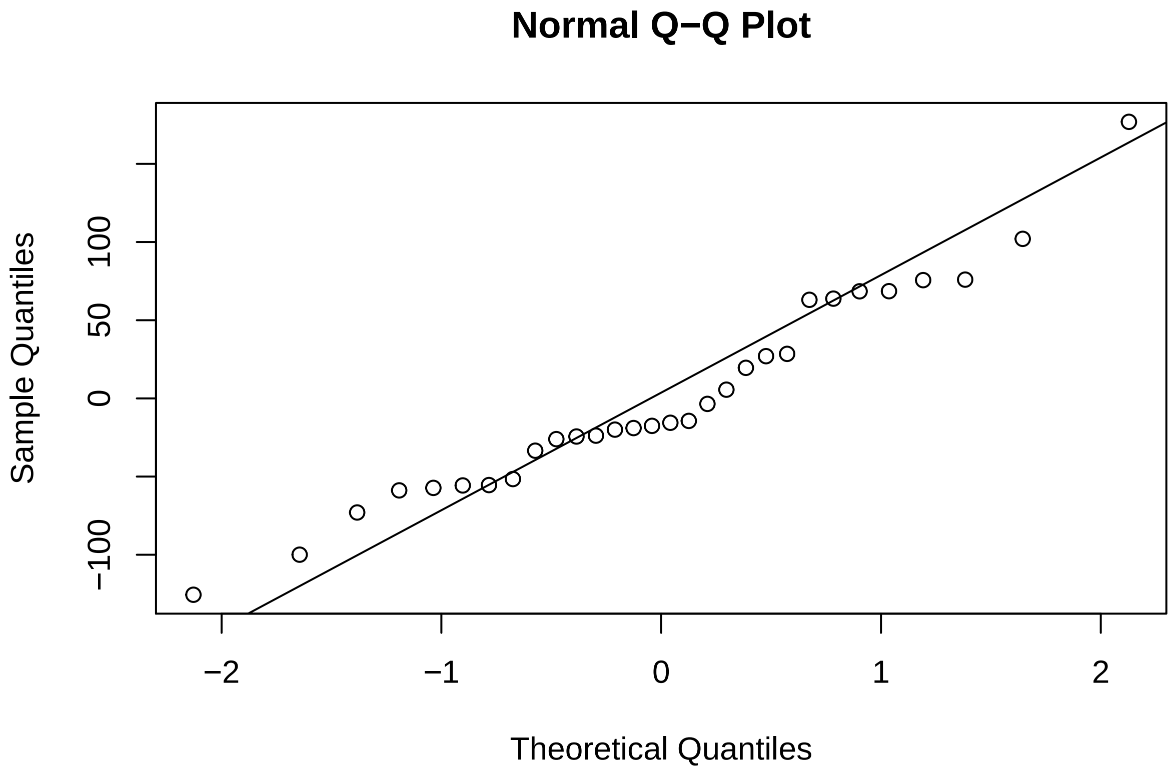
Nearly normal residuals: To check this condition, we can look at a histogram

```r
hist(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
```



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

   I would say the distribution is not nearly as normal as we would like, though there is a weak normal shape.

Constant variability:

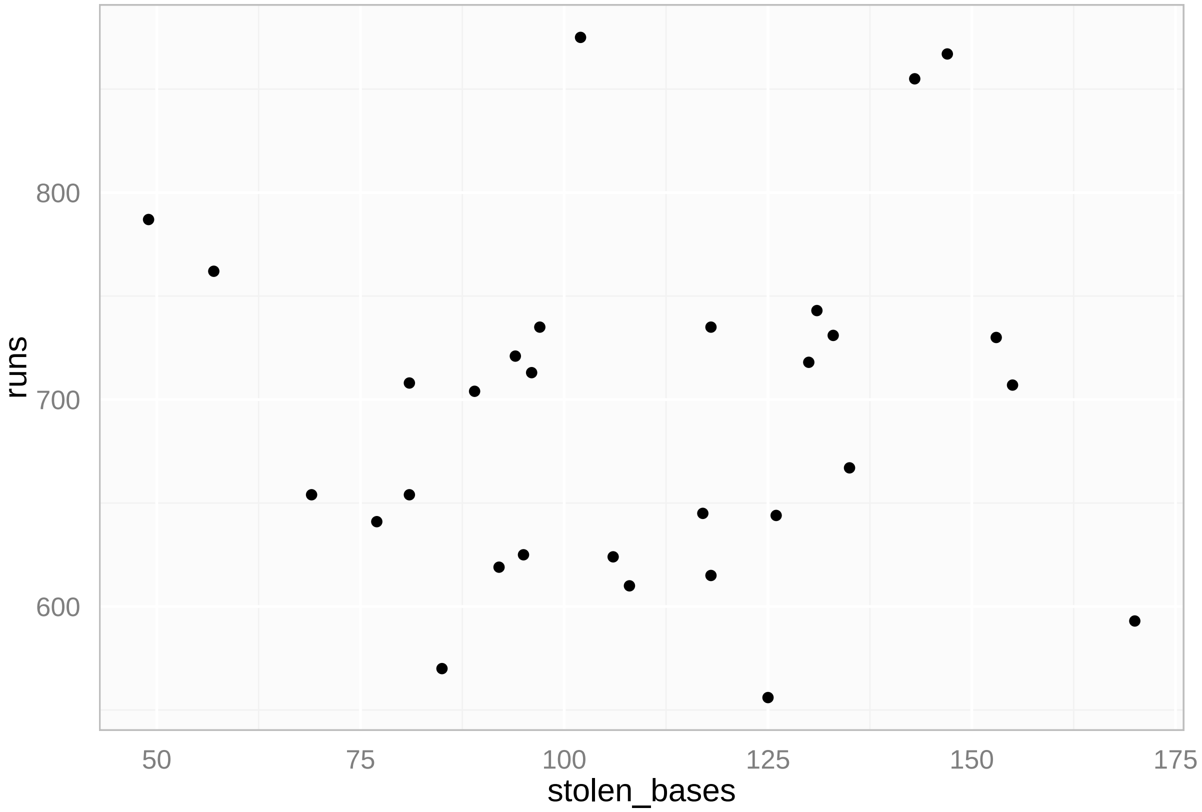
8. Based on the plot in (1), does the constant variability condition appear to be met?

   It appears the constant variability condition would be satisfied.

On Your Own

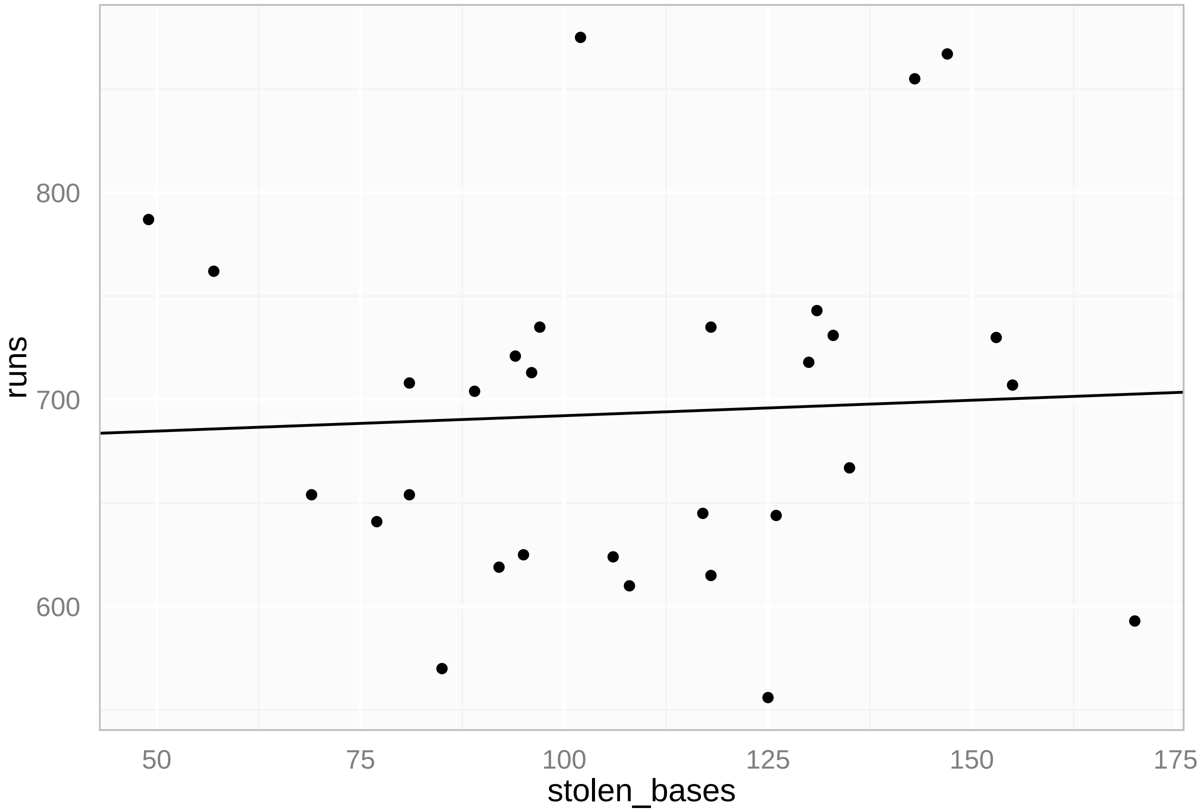
- Choose another traditional variable from `mlb11` that you think might be a good predictor of `runs`. Produce a scatterplot of the two variables and fit a linear model. At a glance, does there seem to be a linear relationship?

```
g3 <- ggplot(data=mlb11, aes(x=stolen_bases, y=runs)) + geom_point() + myTheme
g3
```



There does *not* seem to be a linear relationship between `stolen_bases` and `runs`.

```
m3 <- lm(runs ~ stolen_bases, data=mlb11)
g3 + geom_abline(intercept=m3$coefficients[1], slope=m3$coefficients)
```



```r
summary(m3)
```

```
##
## Call:
## lm(formula = runs ~ stolen_bases, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -139.94  -62.87   10.01   38.54  182.49
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  677.3074    58.9751  11.485 4.17e-12 ***
## stolen_bases   0.1491     0.5211   0.286    0.777
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 83.82 on 28 degrees of freedom
## Multiple R-squared:  0.002914,   Adjusted R-squared:  -0.0327
## F-statistic: 0.08183 on 1 and 28 DF,  p-value: 0.7769
```

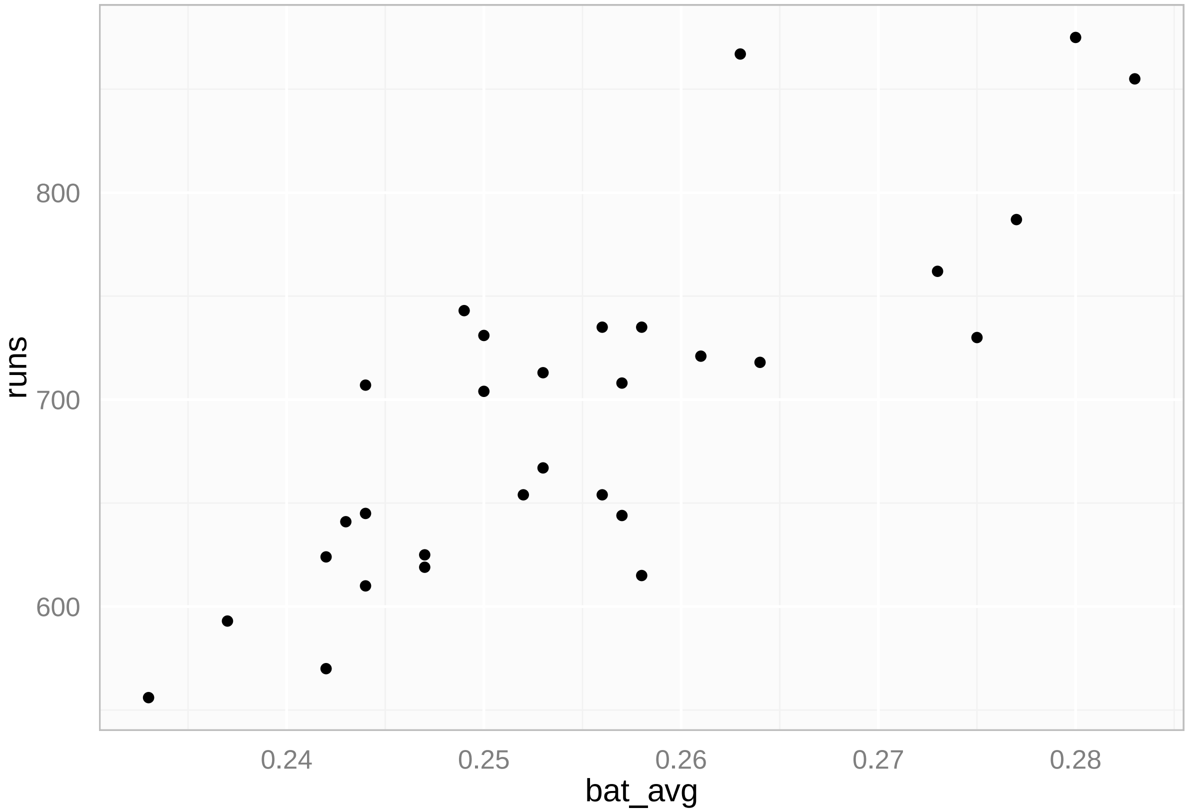
- How does this relationship compare to the relationship between `runs` and `at_bats`? Use the R^2 values from the two model summaries to compare. Does your variable seem to predict `runs` better than `at_bats`? How can you tell?

The `stolen_bases` $R^2 = 0.002914$ which is much lower than the `at_bats` $R^2 = 0.3729$. This indicates that the `at_bats` model explains significantly more of the variability of `runs` than `stolen_bases` does.

- 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).

  Based on a correlation analysis, `bat_avg` appears to have the best positive relationship with `runs` at ~0.8099. The scatterplot below shows a general linear positive trend in the data.

```
g4 <- ggplot(data=mlb11, aes(x=bat_avg, y=runs)) + geom_point() + myTheme
g4
```



```
m4 <- lm(runs ~ bat_avg, data=mlb11)
summary(m4)
```

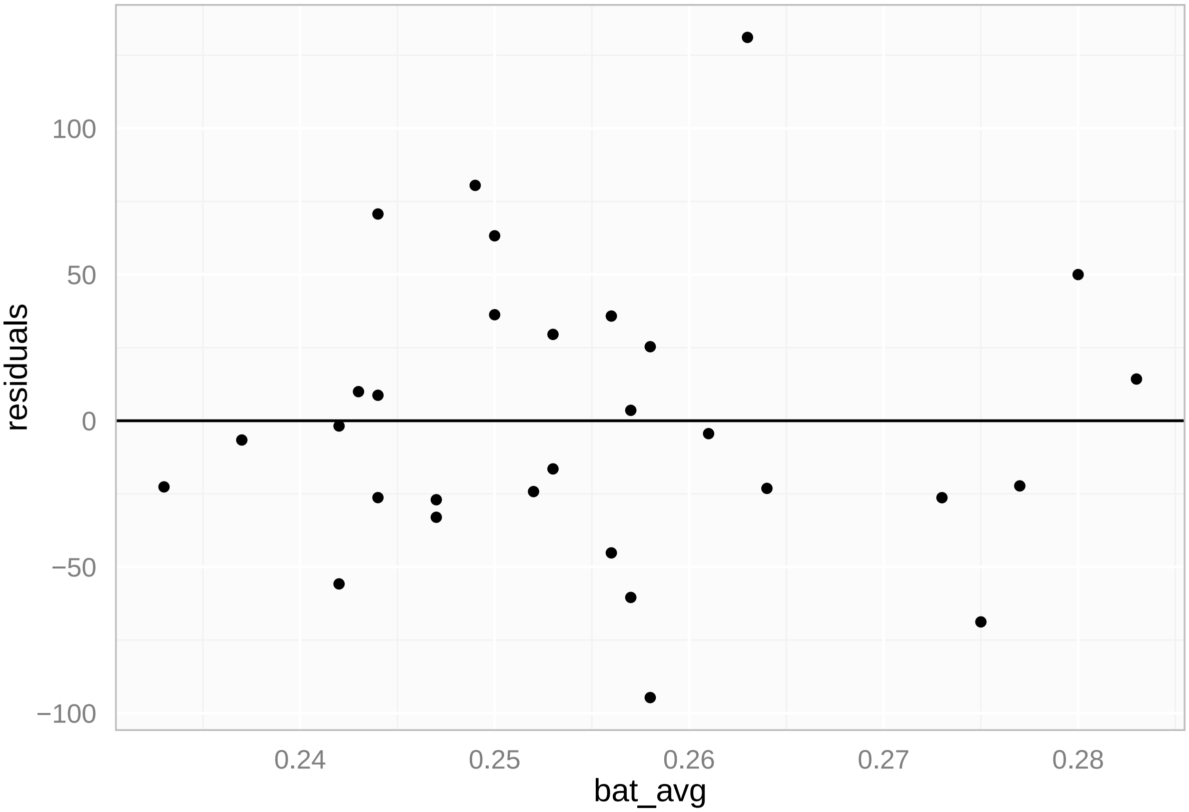
```
##
## Call:
## lm(formula = runs ~ bat_avg, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -94.676 -26.303  -5.496  28.482 131.113
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -642.8      183.1  -3.511  0.00153 **
## bat_avg       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
```

The $R^2 = 0.6561$, and the model would be as follows:

$$\hat{y} = -642.8189333 + 5242.2290794 * bat_avg$$

The following visualization shows the residuals. No obvious pattern is apparent.

```
resData <- data.frame(residuals=m4$residuals, bat_avg=mlb11$bat_avg)
g5 <- ggplot(data=resData, aes(x=bat_avg, y=residuals)) +
  geom_point() +
  geom_hline(yintercept=0) +
  myTheme
g5
```



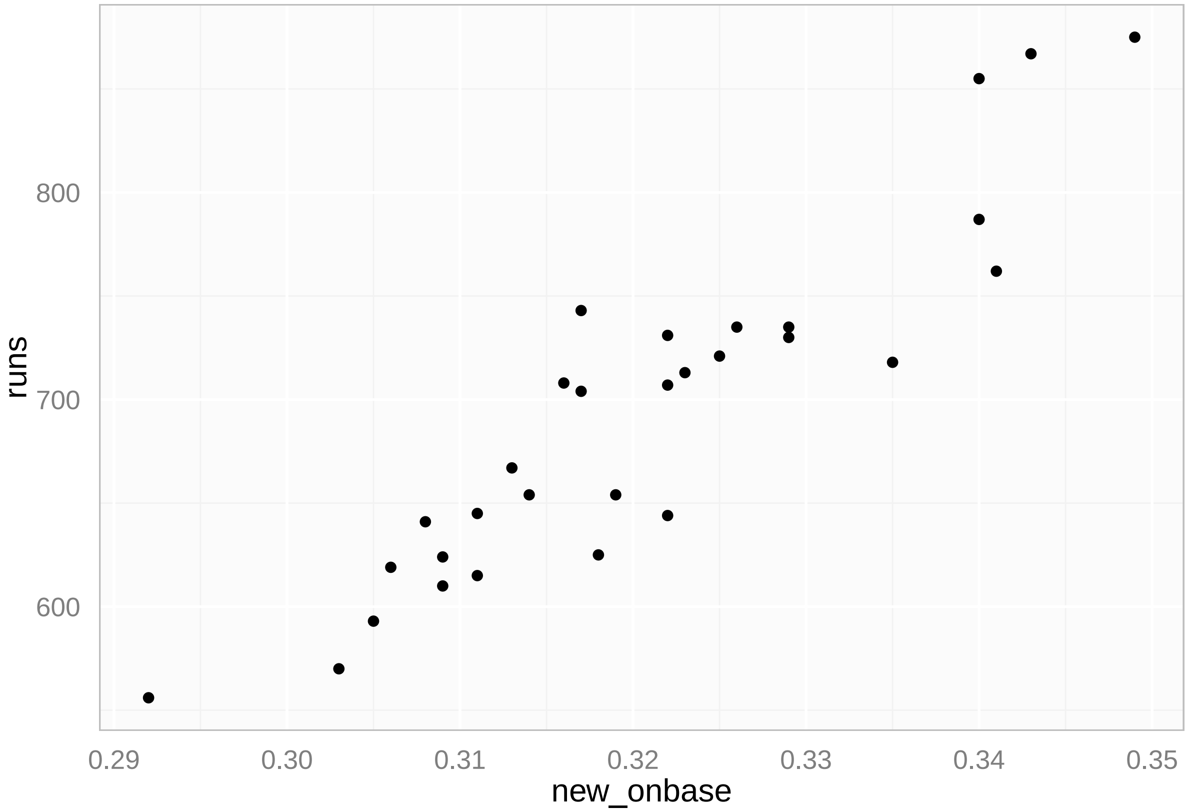
- Now examine the three newer variables. These are the statistics used by the author of *Moneyball* to predict a teams success. In general, are they more or less effective at predicting runs that the old variables? Explain using appropriate graphical and numerical evidence. Of all ten variables we've 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?

  As shown below, the correlation coefficents are higher, in the low to mid 90s, versus the low 80s with the best standard statistic `bat_avg`. The visualizations that follow show a much stronger linear relationship with a narrower range across the data. This supports a better predictive outcome of a linear regression model.

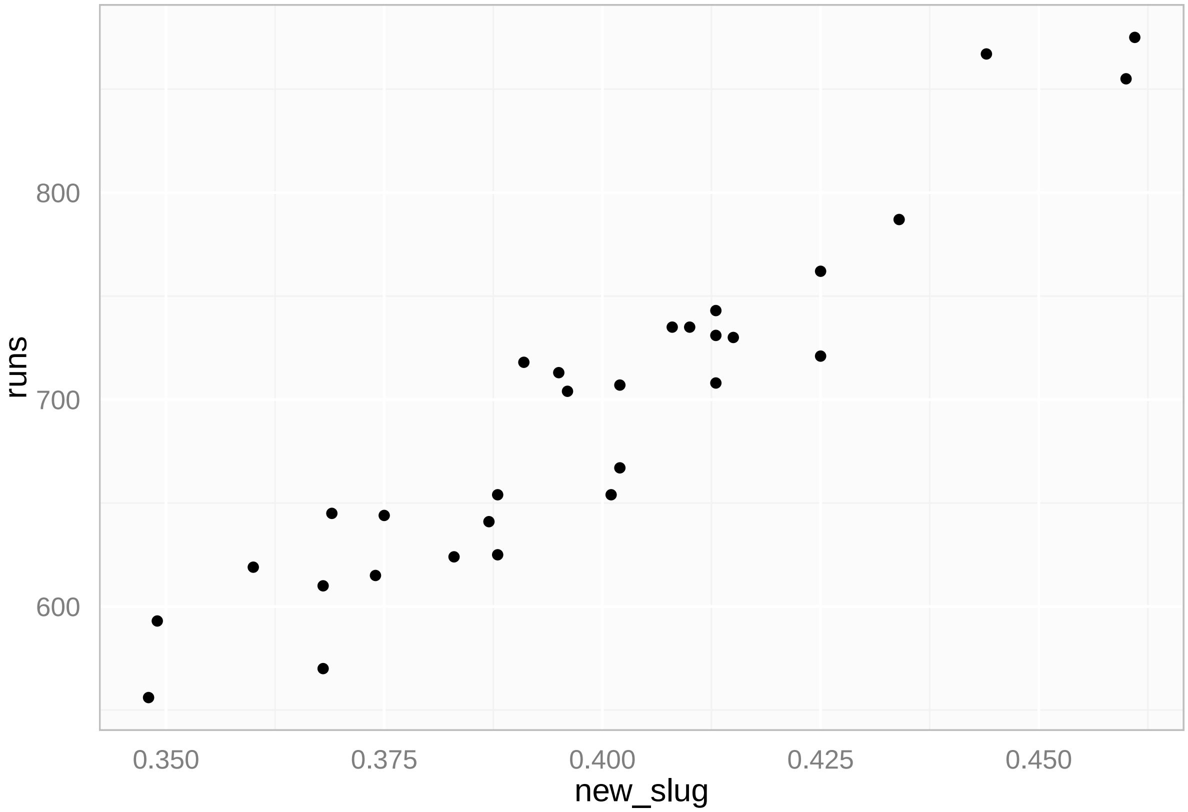
```
cor(mlb11[,c(2,10,11,12)])
```

```
##                 runs new_onbase  new_slug   new_obs
## runs       1.0000000  0.9214691 0.9470324 0.9669163
## new_onbase 0.9214691  1.0000000 0.8718645 0.9372834
## new_slug   0.9470324  0.8718645 1.0000000 0.9877645
## new_obs    0.9669163  0.9372834 0.9877645 1.0000000
```

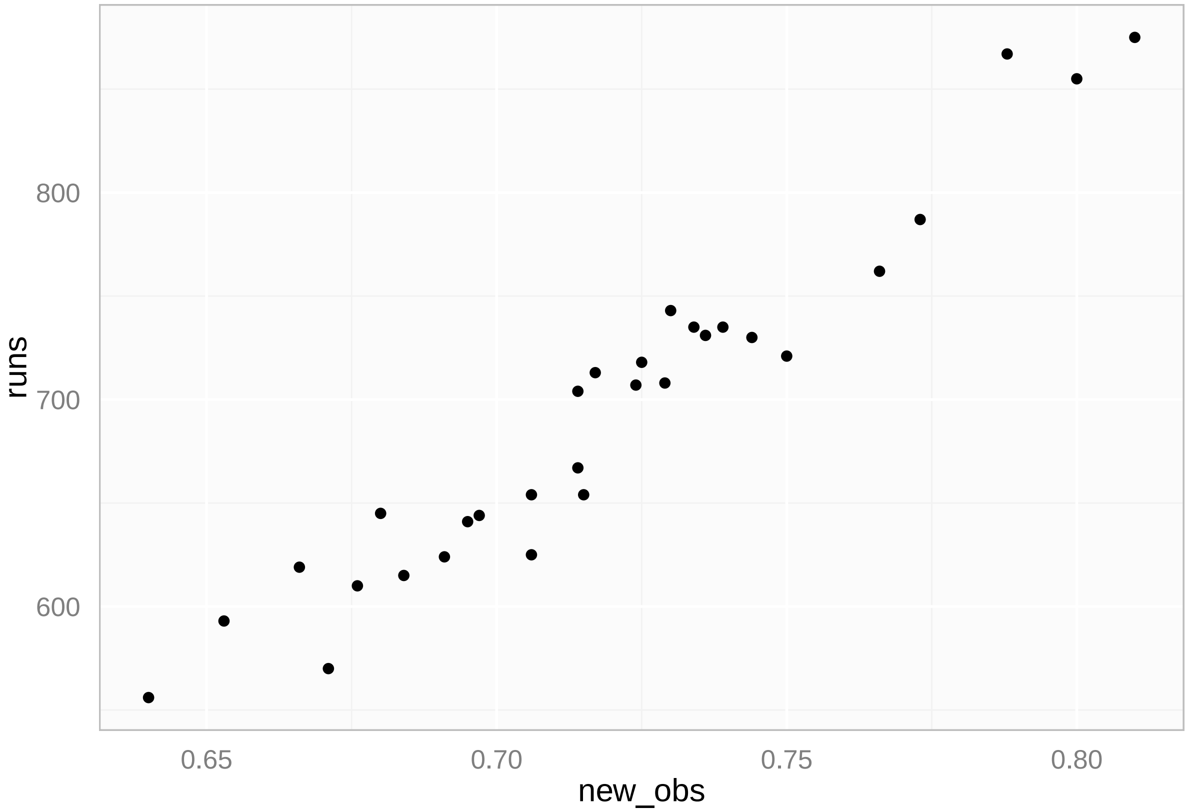
```
g6 <- ggplot(data=mlb11, aes(x=new_onbase, y=runs)) + geom_point() + myTheme
g6
```



```
g6 <- ggplot(data=mlb11, aes(x=new_slug, y=runs)) + geom_point() + myTheme
g6
```



```
g6 <- ggplot(data=mlb11, aes(x=new_obs, y=runs)) + geom_point() + myTheme
g6
```



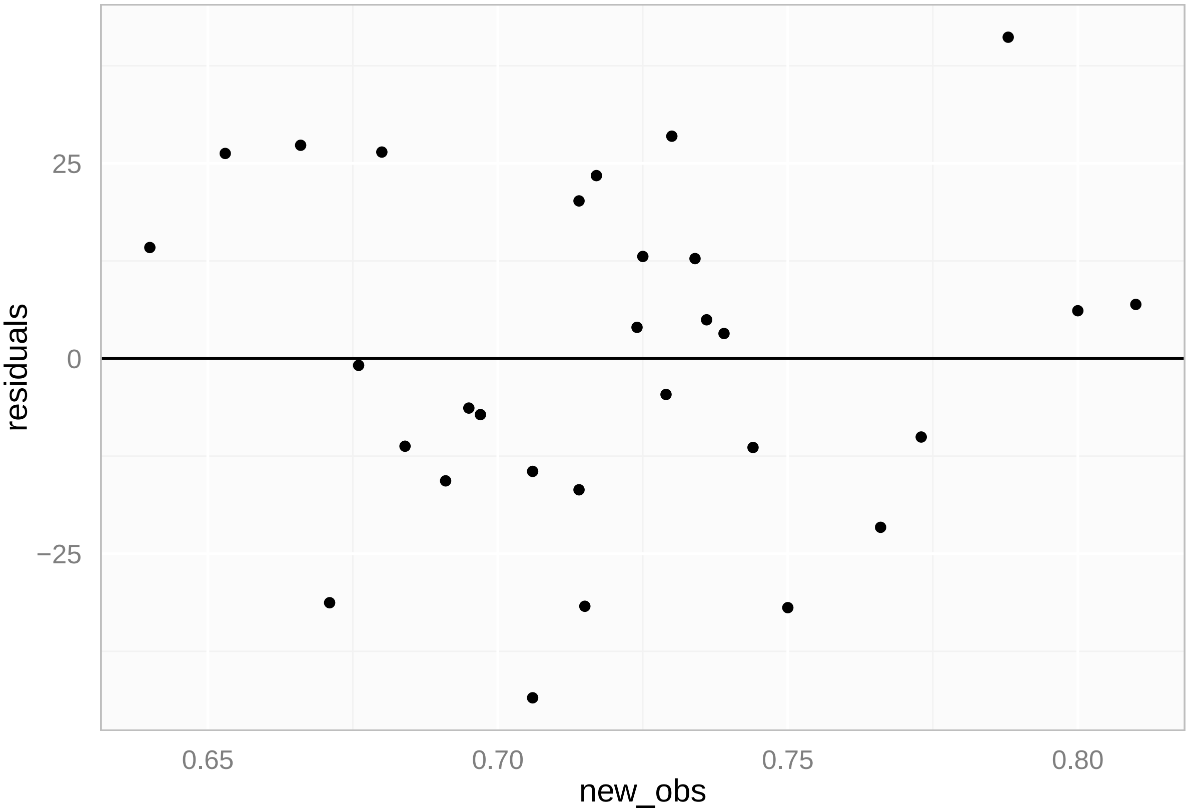
Moving forward with `new_obs`, let's check the linear model. We see from the summary output that the $R^2 = 0.9349$. I'm not sure what the `new_obs` represents, but it seems to be a good predictor. P-values are well below 0.05.

```
m6 <- lm(runs ~ new_obs, data=mlb11)
summary(m6)
```

```
##
## Call:
## lm(formula = runs ~ new_obs, 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 ***
## new_obs      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 residual plot below looks good. No obvious trends, and constant variability.

```
resData <- data.frame(residuals=m6$residuals, new_obs=mlb11$new_obs)
g5 <- ggplot(data=resData, aes(x=new_obs, y=residuals)) +
  geom_point() +
  geom_hline(yintercept=0) +
  myTheme
g5
```



- Check the model diagnostics for the regression model with the variable you decided was the best predictor for runs.

  P-values are very low, well below 0.05, and R^2 is high, at 0.9349, so `new_obs` seems like a good predictor of runs.

This is a product of OpenIntro that is released under a Creative Commons Attribution-ShareAlike 3.0 Unported. This lab was adapted for OpenIntro by Andrew Bray and Mine Çetinkaya-Rundel from a lab written by the faculty and TAs of UCLA Statistics.