

Data Science Linear Regression

The textbook for the Data Science course series is [freely available online](#).

Learning Objectives

- How linear regression was originally developed by Galton
- What confounding is and how to detect it
- How to examine the relationships between variables by implementing linear regression in R

Course Overview

There are three major sections in this course: introduction to linear regression, linear models, and confounding.

Introduction to Linear Regression

In this section, you'll learn the basics of linear regression through this course's motivating example, the data-driven approach used to construct baseball teams. You'll also learn about correlation, the correlation coefficient, stratification, and the variance explained.

Linear Models

In this section, you'll learn about linear models. You'll learn about least squares estimates, multivariate regression, and several useful features of R, such as `tibbles`, `lm`, `do`, and `broom`. You'll learn how to apply regression to baseball to build a better offensive metric.

Confounding

In the final section of the course, you'll learn about confounding and several reasons that correlation is not the same as causation, such as spurious correlation, outliers, reversing cause and effect, and confounders. You'll also learn about Simpson's Paradox.

Section 1 - Introduction to Regression Overview

In the **Introduction to Regression** section, you will learn the basics of linear regression.

After completing this section, you will be able to:

- Understand how Galton developed **linear regression**.
- Calculate and interpret the **sample correlation**.
- **Stratify** a dataset when appropriate.
- Understand what a **bivariate normal distribution** is.
- Explain what the term **variance explained** means.
- Interpret the two **regression lines**.

This section has three parts: **Baseball as a Motivating Example**, **Correlation**, and **Stratification and Variance Explained**.

Motivating Example: Moneyball

The corresponding section of the textbook is the [case study on Moneyball](#)

Key points

Bill James was the originator of the **sabermetrics**, the approach of using data to predict what outcomes best predicted if a team would win.

Baseball basics

The corresponding section of the textbook is the [section on baseball basics](#)

Key points

- The goal of a baseball game is to score more runs (points) than the other team.
- Each team has 9 batters who have an opportunity to hit a ball with a bat in a predetermined order.
- Each time a batter has an opportunity to bat, we call it a plate appearance (PA).
- The PA ends with a binary outcome: the batter either makes an out (failure) and returns to the bench or the batter doesn't (success) and can run around the bases, and potentially score a run (reach all 4 bases).
- We are simplifying a bit, but there are five ways a batter can succeed (not make an out):
 1. Bases on balls (BB): the pitcher fails to throw the ball through a predefined area considered to be hittable (the strike zone), so the batter is permitted to go to first base.
 2. Single: the batter hits the ball and gets to first base.
 3. Double (2B): the batter hits the ball and gets to second base.
 4. Triple (3B): the batter hits the ball and gets to third base.
 5. Home Run (HR): the batter hits the ball and goes all the way home and scores a run.
- Historically, the batting average has been considered the most important offensive statistic. To define this average, we define a hit (H) and an at bat (AB). Singles, doubles, triples and home runs are hits. The fifth way to be successful, a walk (BB), is not a hit. An AB is the number of times you either get a hit or make an out; BBs are excluded. The batting average is simply H/AB and is considered the main measure of a success rate.

Bases on Balls or Stolen Bases?

The corresponding section of the textbook is the [base on balls or stolen bases textbook section](#)

Key points

The visualization of choice when exploring the relationship between two variables like home runs and runs is a scatterplot.

Code: Scatterplot of the relationship between HRs and runs

```
if(!require(Lahman)) install.packages("Lahman")

## Loading required package: Lahman

if(!require(tidyverse)) install.packages("tidyverse")

## Loading required package: tidyverse

## -- Attaching packages ----- tidyverse 1.3.0 --

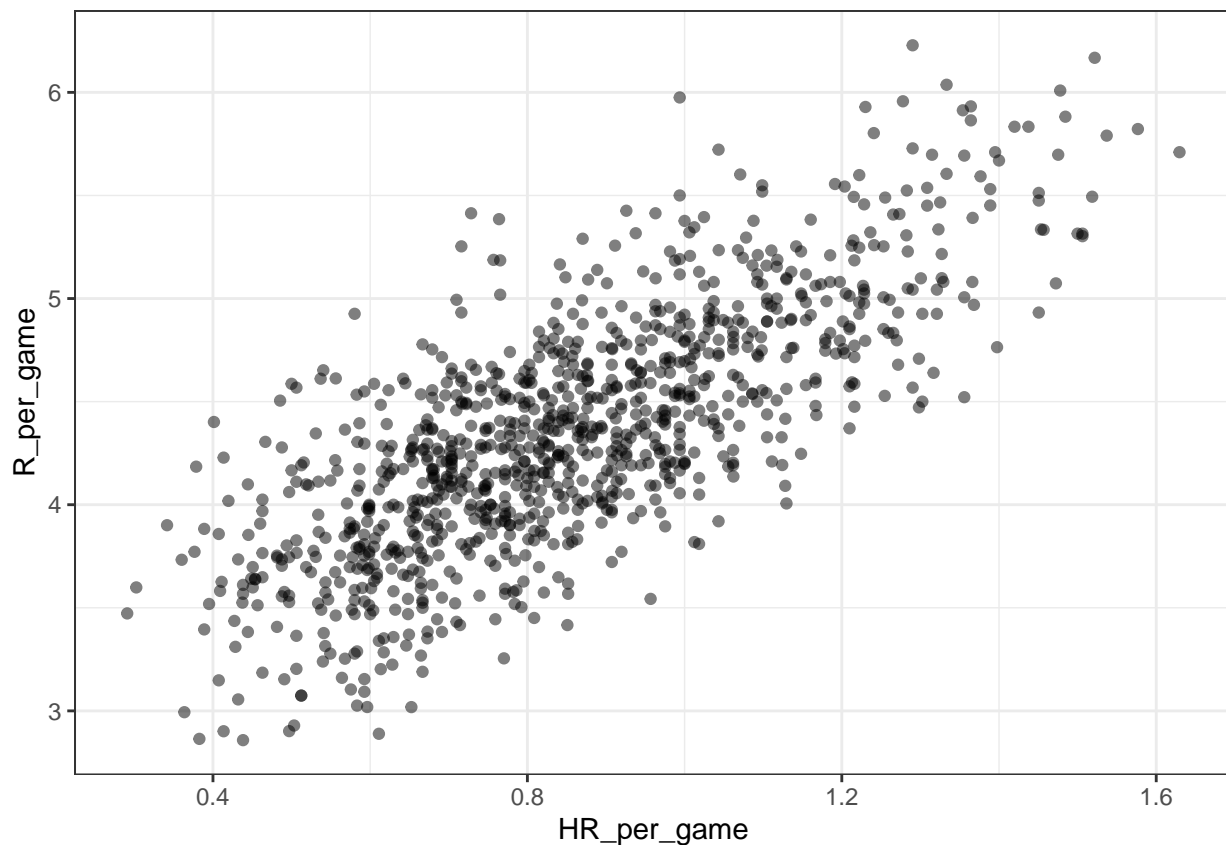
## v ggplot2 3.3.2      v purrr   0.3.4
## v tibble  3.0.3      v dplyr   1.0.2
## v tidyr   1.1.2      v stringr 1.4.0
## v readr   1.3.1      v forcats 0.5.0
```

```
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag() masks stats::lag()
if(!require(dslabs)) install.packages("dslabs")

## Loading required package: dslabs

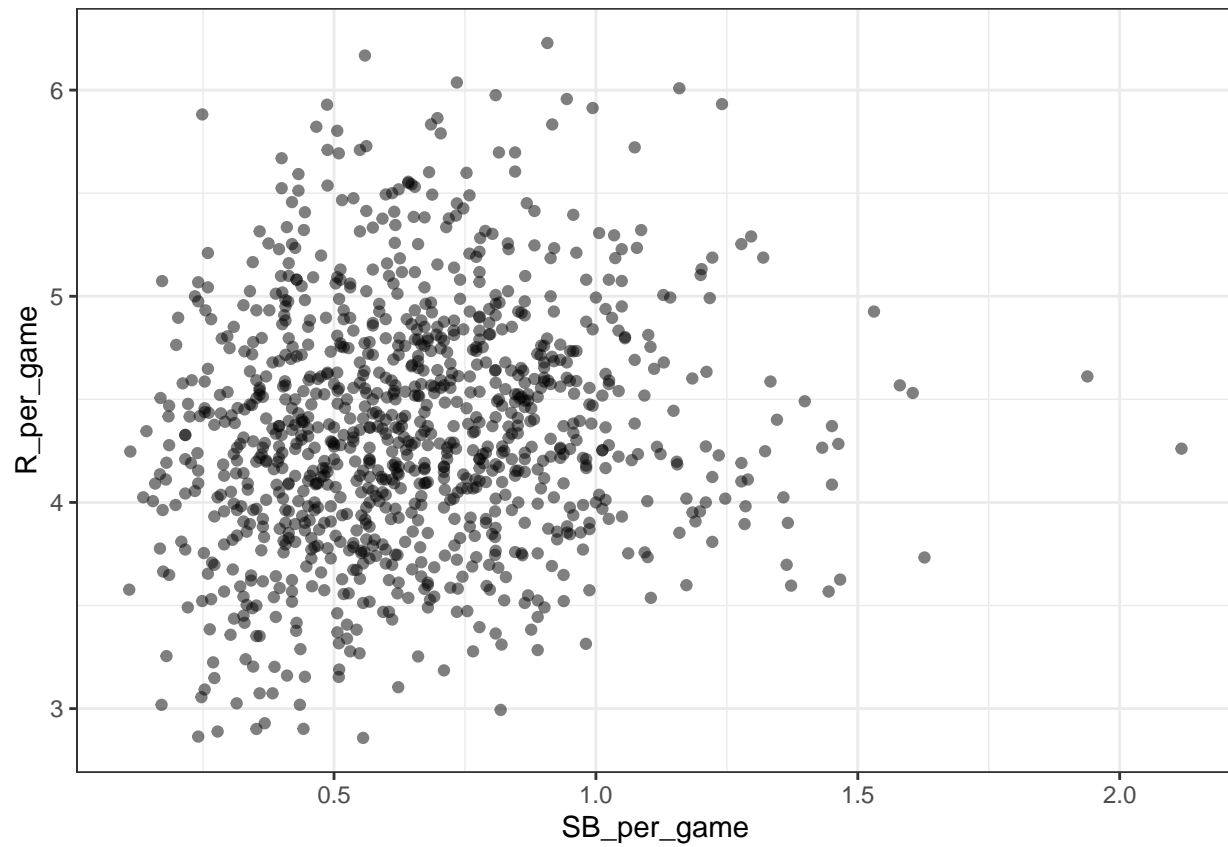
library(Lahman)
library(tidyverse)
library(dslabs)
ds_theme_set()

Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(HR_per_game = HR / G, R_per_game = R / G) %>%
  ggplot(aes(HR_per_game, R_per_game)) +
  geom_point(alpha = 0.5)
```



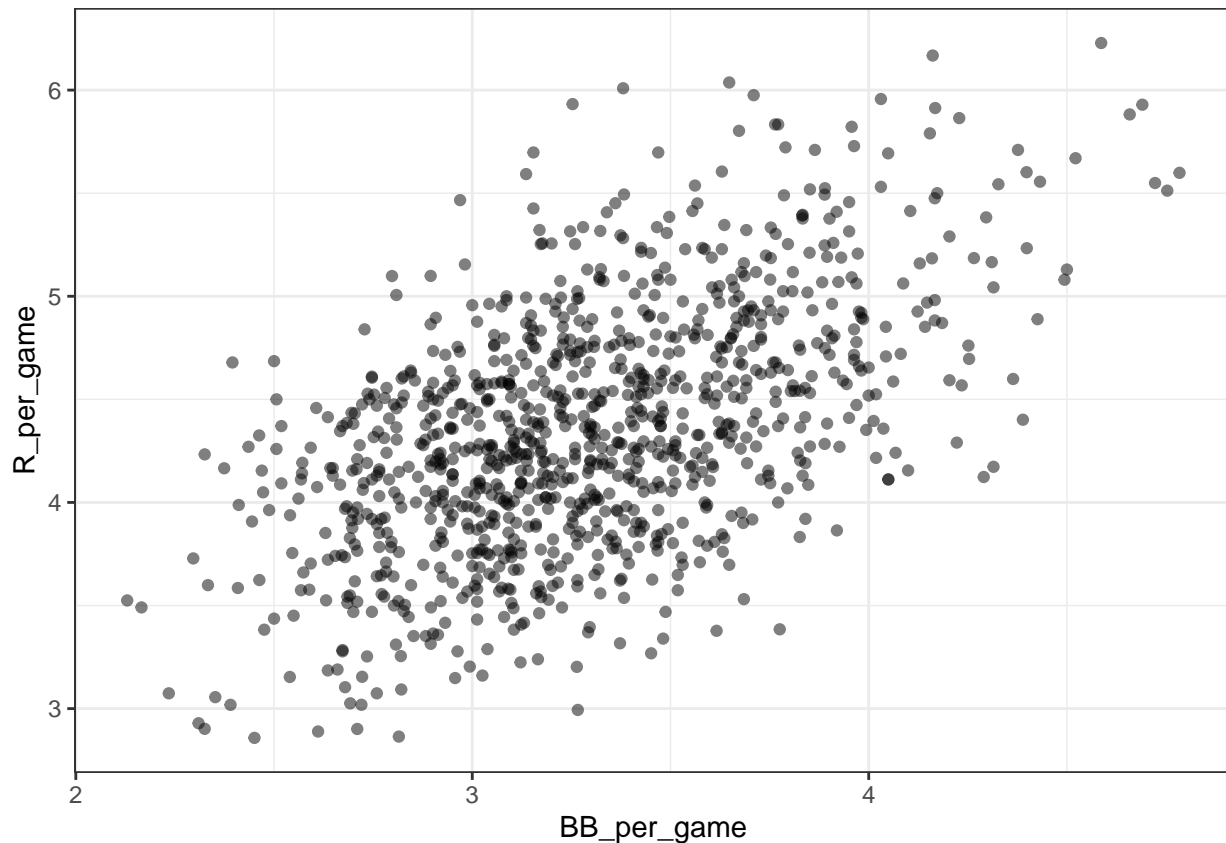
Code: Scatterplot of the relationship between stolen bases and runs

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(SB_per_game = SB / G, R_per_game = R / G) %>%
  ggplot(aes(SB_per_game, R_per_game)) +
  geom_point(alpha = 0.5)
```



Code: Scatterplot of the relationship between bases on balls and runs

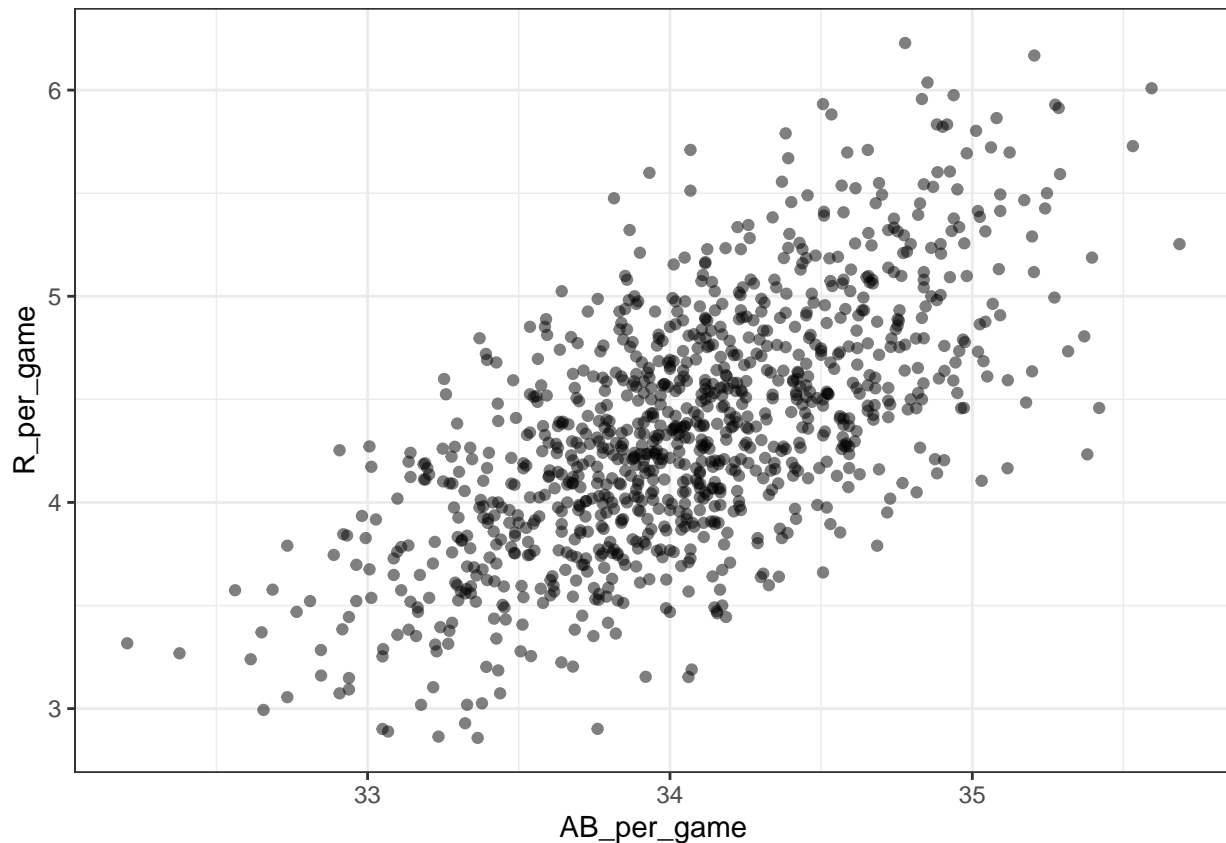
```
Teams %>% filter(yearID %in% 1961:2001) %>%  
  mutate(BB_per_game = BB / G, R_per_game = R / G) %>%  
  ggplot(aes(BB_per_game, R_per_game)) +  
  geom_point(alpha = 0.5)
```



Assessment - Baseball as a Motivating Example

1. What is the application of statistics and data science to baseball called?
 - ☐ A. Moneyball
 - ☒ B. Sabermetrics
 - ☐ C. The “Oakland A’s Approach”
 - ☐ D. There is no specific name for this; it’s just data science.
2. Which of the following outcomes is not included in the batting average?
 - ☐ A. A home run
 - ☒ B. A base on balls
 - ☐ C. An out
 - ☐ D. A single
3. Why do we consider team statistics as well as individual player statistics?
 - ☒ A. The success of any individual player also depends on the strength of their team.
 - ☐ B. Team statistics can be easier to calculate.
 - ☐ C. The ultimate goal of sabermetrics is to rank teams, not players.
4. You want to know whether teams with more at-bats per game have more runs per game. What R code below correctly makes a scatter plot for this relationship?

```
Teams %>% filter(yearID %in% 1961:2001 ) %>%
  mutate(AB_per_game = AB/G, R_per_game = R/G) %>%
  ggplot(aes(AB_per_game, R_per_game)) +
  geom_point(alpha = 0.5)
```



- [] A.

```
Teams %>% filter(yearID %in% 1961:2001 ) %>%
  ggplot(aes(AB, R)) +
  geom_point(alpha = 0.5)
```

- [X] B.

```
Teams %>% filter(yearID %in% 1961:2001 ) %>%
  mutate(AB_per_game = AB/G, R_per_game = R/G) %>%
  ggplot(aes(AB_per_game, R_per_game)) +
  geom_point(alpha = 0.5)
```

- [] C.

```
Teams %>% filter(yearID %in% 1961:2001 ) %>%
  mutate(AB_per_game = AB/G, R_per_game = R/G) %>%
  ggplot(aes(AB_per_game, R_per_game)) +
  geom_line()
```

- [] D.

```
Teams %>% filter(yearID %in% 1961:2001 ) %>%
  mutate(AB_per_game = AB/G, R_per_game = R/G) %>%
  ggplot(aes(R_per_game, AB_per_game)) +
  geom_point()
```

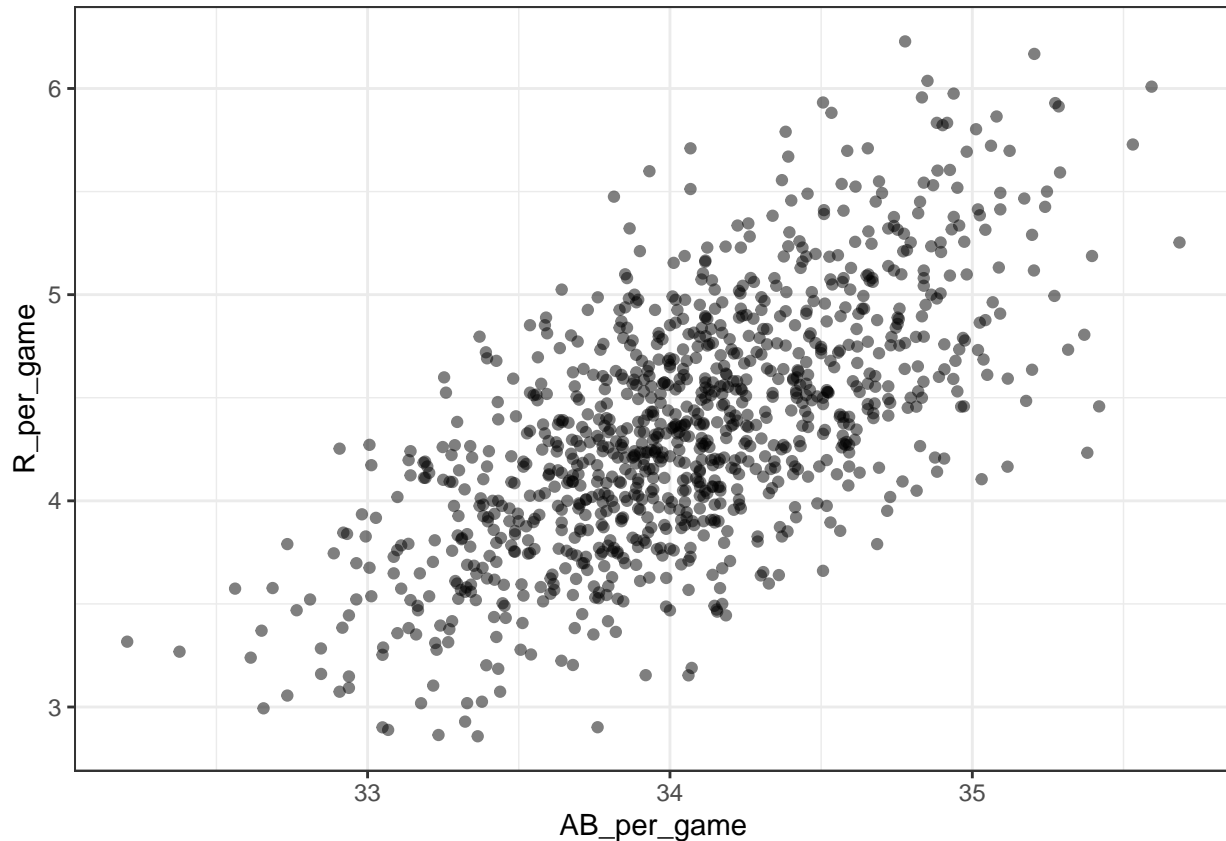
5. What does the variable “SOA” stand for in the Teams table?

Hint: make sure to use the help file (?Teams).

- ☐ A. sacrifice out
- ☐ B. slides or attempts
- ☒ C. strikeouts by pitchers
- ☐ D. accumulated singles

6. Load the **Lahman** library. Filter the **Teams** data frame to include years from 1961 to 2001. Make a scatterplot of runs per game versus at bats (AB) per game.

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(AB_per_game = AB / G, R_per_game = R / G) %>%
  ggplot(aes(AB_per_game, R_per_game)) +
  geom_point(alpha = 0.5)
```

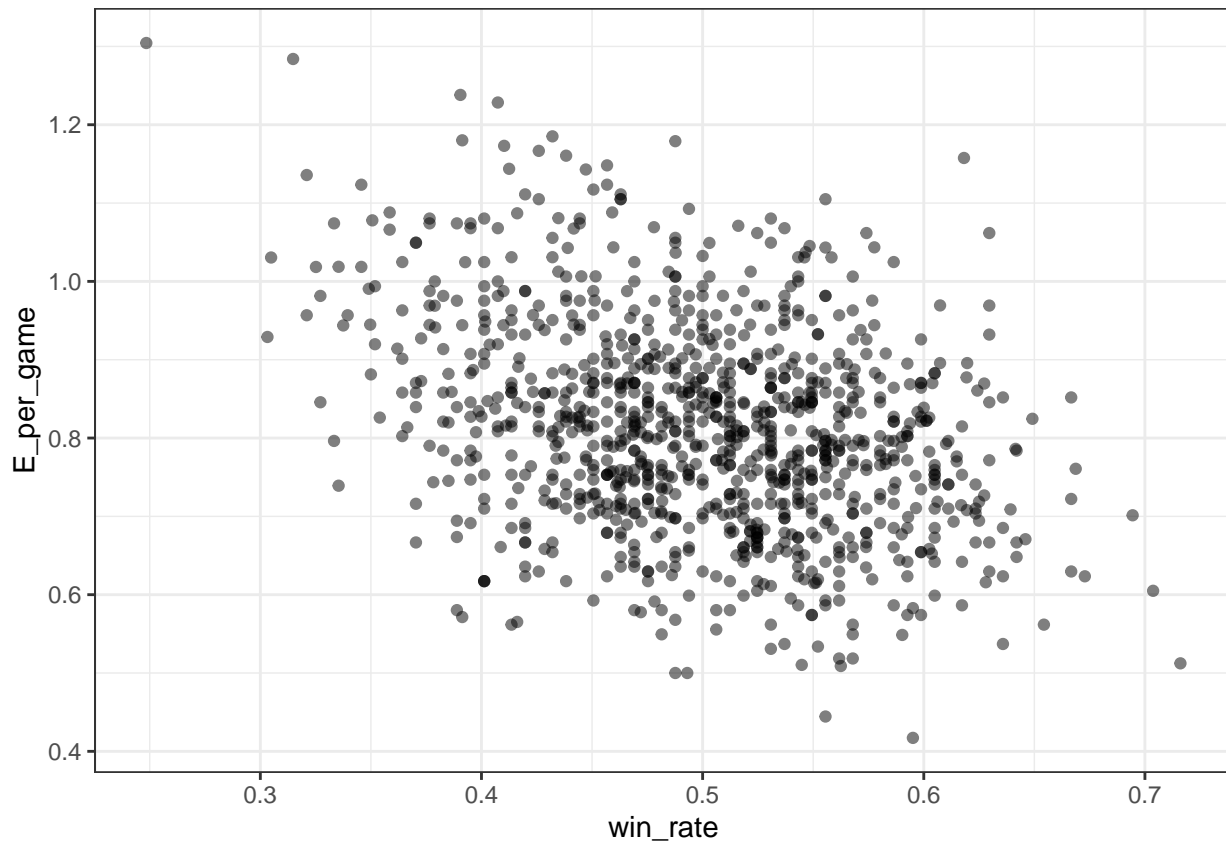


Which of the following is true?

- ☐ A. There is no clear relationship between runs and at bats per game.
- ☒ B. As the number of at bats per game increases, the number of runs per game tends to increase.
- ☐ C. As the number of at bats per game increases, the number of runs per game tends to decrease.

7. Use the filtered **Teams** data frame from Question 6. Make a scatterplot of win rate (number of wins per game) versus number of fielding errors (E) per game.

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(win_rate = W / G, E_per_game = E / G) %>%
  ggplot(aes(win_rate, E_per_game)) +
  geom_point(alpha = 0.5)
```

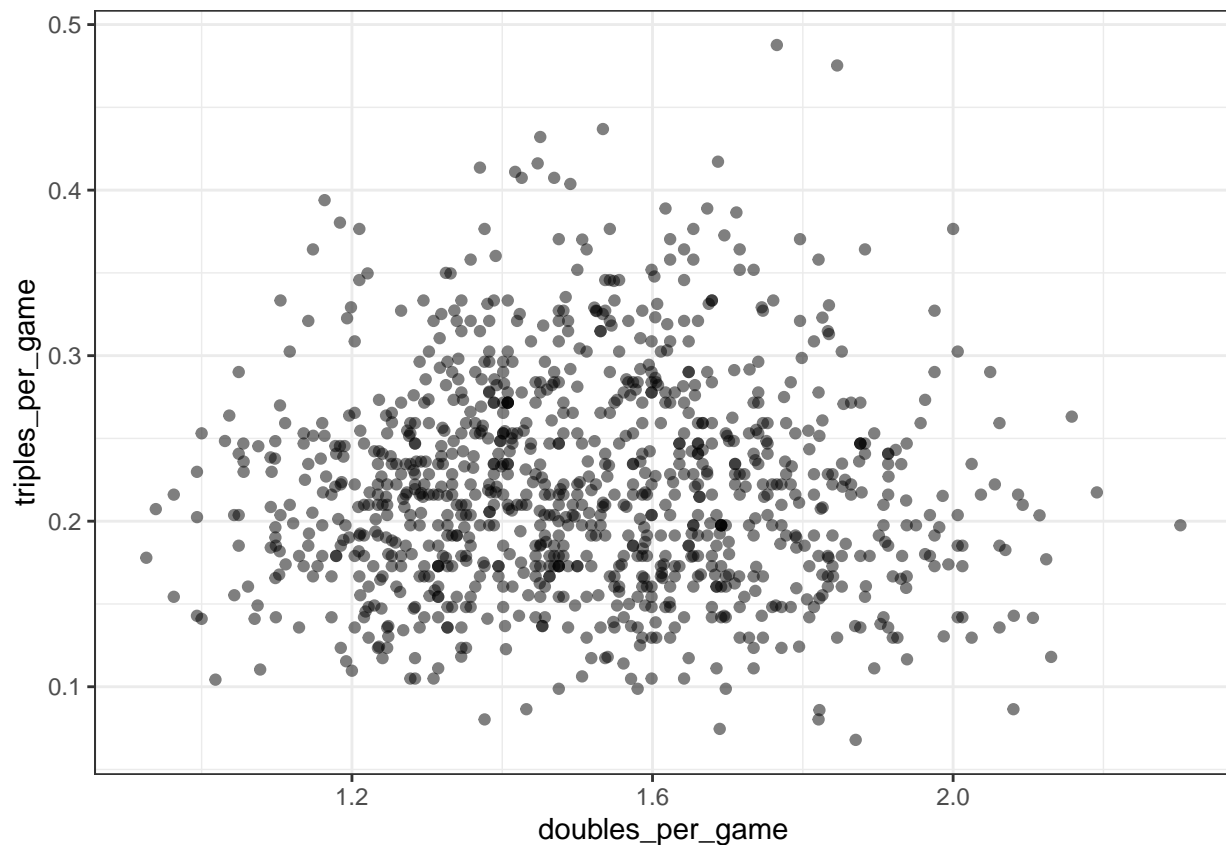


Which of the following is true?

- [] A. There is no relationship between win rate and errors per game.
- [] B. As the number of errors per game increases, the win rate tends to increase.
- [X] C. As the number of errors per game increases, the win rate tends to decrease.

8. Use the filtered `Teams` data frame from Question 6. Make a scatterplot of triples (`X3B`) per game versus doubles (`X2B`) per game.

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(doubles_per_game = X2B / G, triples_per_game = X3B / G) %>%
  ggplot(aes(doubles_per_game, triples_per_game)) +
  geom_point(alpha = 0.5)
```

Which of the following is true?

- ☒ A. There is no clear relationship between doubles per game and triples per game.
- ☐ B. As the number of doubles per game increases, the number of triples per game tends to increase.
- ☐ C. As the number of doubles per game increases, the number of triples per game tends to decrease.

Correlation

The corresponding textbook section is [Case Study: is height hereditary?](#)

Key points

- Galton tried to predict sons' heights based on fathers' heights.
- The mean and standard errors are insufficient for describing an important characteristic of the data: the trend that the taller the father, the taller the son.
- The correlation coefficient is an informative summary of how two variables move together that can be used to predict one variable using the other.

Code

```
# create the dataset
if(!require(HistData)) install.packages("HistData")

## Loading required package: HistData
library(HistData)
data("GaltonFamilies")
set.seed(1983)
galton_heights <- GaltonFamilies %>%
  filter(gender == "male") %>%
```

```

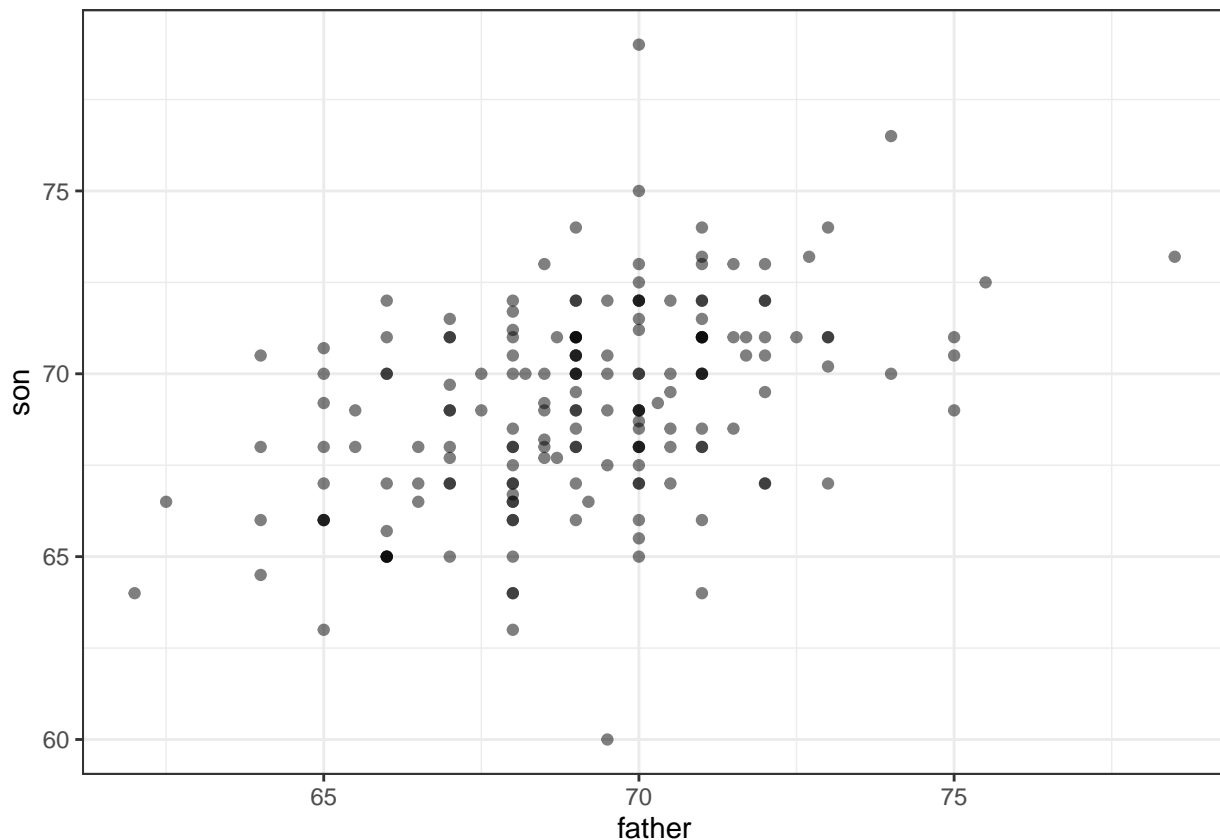
group_by(family) %>%
sample_n(1) %>%
ungroup() %>%
select(father, childHeight) %>%
rename(son = childHeight)

# means and standard deviations
galton_heights %>%
  summarize(mean(father), sd(father), mean(son), sd(son))

## # A tibble: 1 x 4
##   `mean(father)` `sd(father)` `mean(son)` `sd(son)`
##         <dbl>         <dbl>         <dbl>         <dbl>
## 1          69.1           2.55          69.2           2.71

# scatterplot of father and son heights
galton_heights %>%
  ggplot(aes(father, son)) +
  geom_point(alpha = 0.5)

```



Correlation Coefficient

The corresponding textbook section is [the correlation coefficient](#)

Key points

- The correlation coefficient is defined for a list of pairs $(x_1, y_1), \dots, (x_n, y_n)$ as the product of the standardized values: $(\frac{x_i - \mu_x}{\sigma_x})(\frac{y_i - \mu_y}{\sigma_y})$.

- The correlation coefficient essentially conveys how two variables move together.
- The correlation coefficient is always between -1 and 1.

Code

```
rho <- mean(scale(x)*scale(y))
```

```
galton_heights <- GaltonFamilies %>% filter(childNum == 1 & gender == "male") %>% select(father, childH
galton_heights %>% summarize(r = cor(father, son)) %>% pull(r)
```

```
## [1] 0.5007248
```

Sample Correlation is a Random Variable

The corresponding textbook section is [Sample correlation is a random variable](#)

Key points

- The correlation that we compute and use as a summary is a random variable.
- When interpreting correlations, it is important to remember that correlations derived from samples are estimates containing uncertainty.
- Because the sample correlation is an average of independent draws, the central limit theorem applies.

Code

```
# compute sample correlation
```

```
R <- sample_n(galton_heights, 25, replace = TRUE) %>%
  summarize(r = cor(father, son))
```

```
R
```

```
##           r
## 1 0.4787613
```

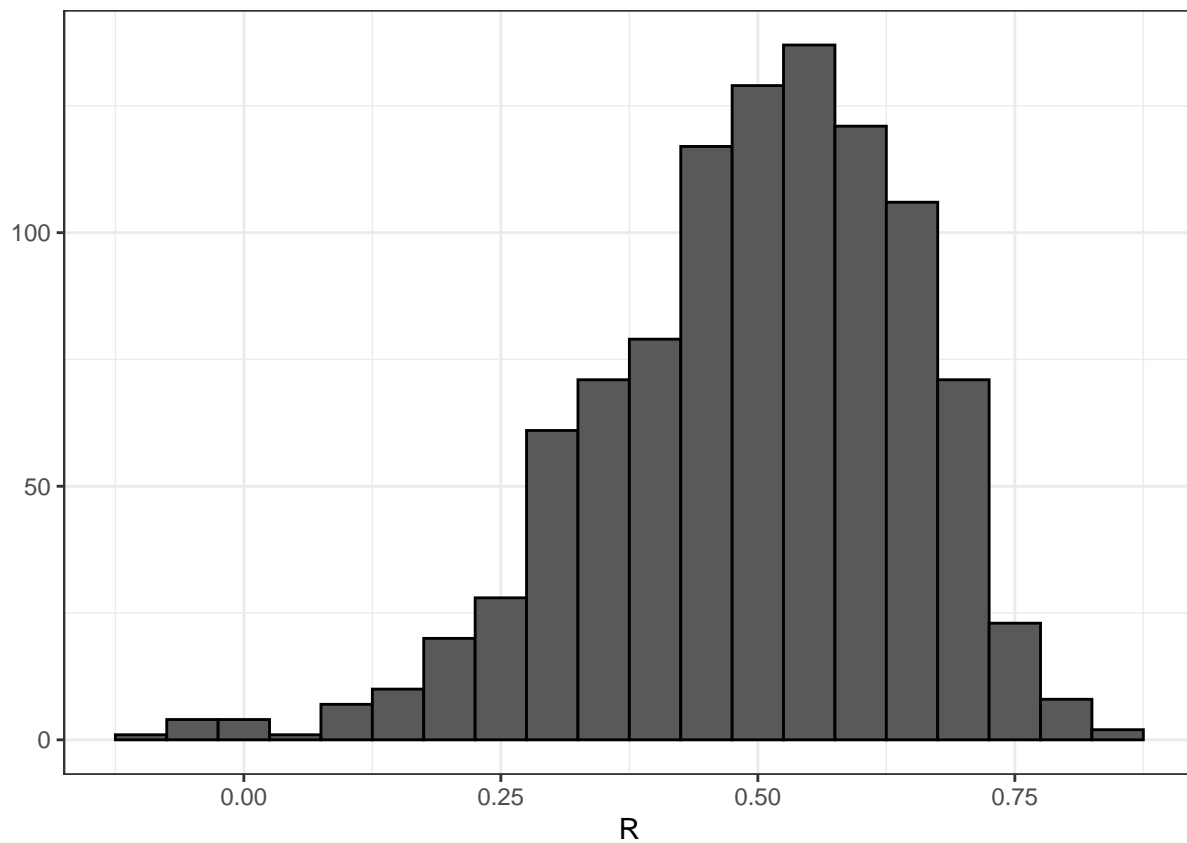
```
# Monte Carlo simulation to show distribution of sample correlation
```

```
B <- 1000
```

```
N <- 25
```

```
R <- replicate(B, {
  sample_n(galton_heights, N, replace = TRUE) %>%
    summarize(r = cor(father, son)) %>%
    pull(r)
})
```

```
qplot(R, geom = "histogram", binwidth = 0.05, color = I("black"))
```



```
# expected value and standard error
```

```
mean(R)
```

```
## [1] 0.4970997
```

```
sd(R)
```

```
## [1] 0.1512451
```

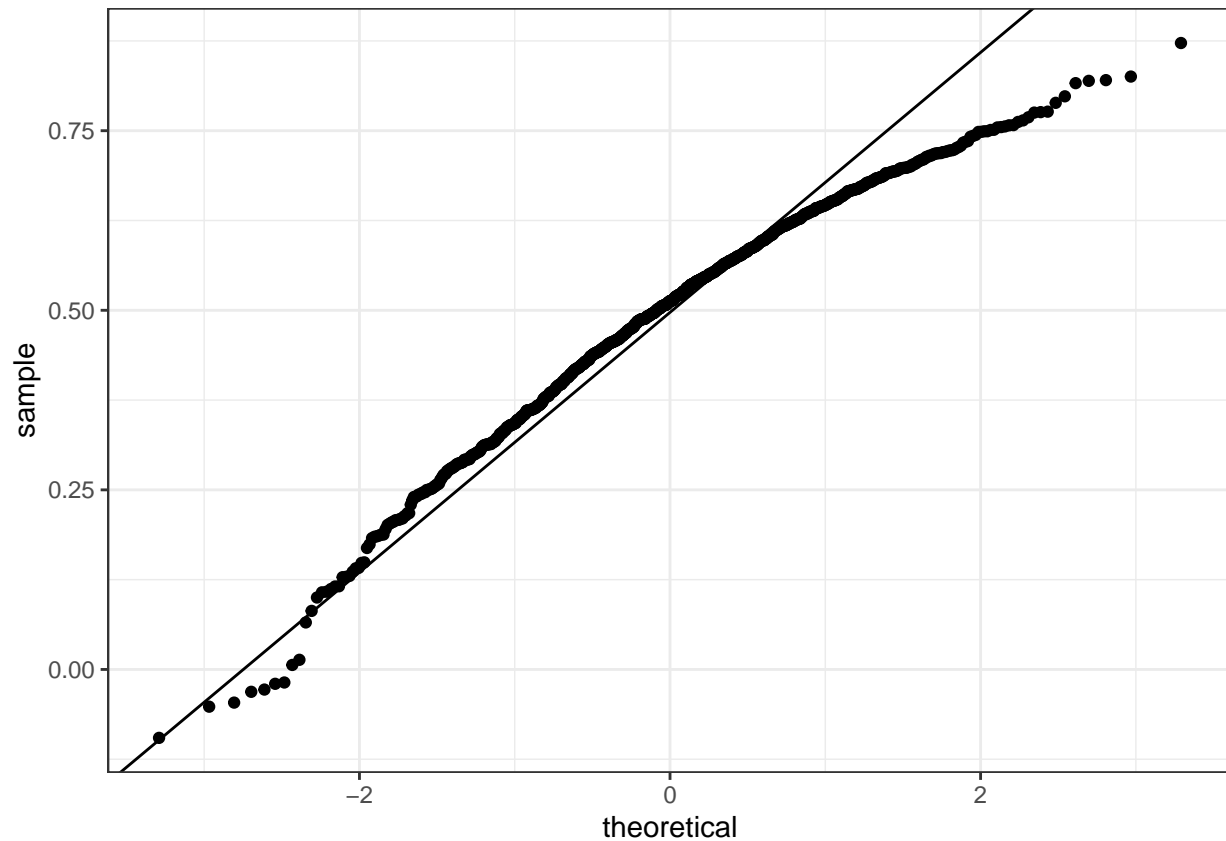
```
# QQ-plot to evaluate whether N is large enough
```

```
data.frame(R) %>%
```

```
  ggplot(aes(sample = R)) +
```

```
  stat_qq() +
```

```
  geom_abline(intercept = mean(R), slope = sqrt((1-mean(R)^2)/(N-2)))
```



Assessment - Correlation

1. While studying heredity, Francis Galton developed what important statistical concept?
 - ☐ A. Standard deviation
 - ☐ B. Normal distribution
 - ☒ C. Correlation
 - ☐ D. Probability
2. The correlation coefficient is a summary of what?
 - ☒ A. The trend between two variables
 - ☐ B. The dispersion of a variable
 - ☐ C. The central tendency of a variable
 - ☐ D. The distribution of a variable

3. Below is a scatter plot showing the relationship between two variables, x and y.

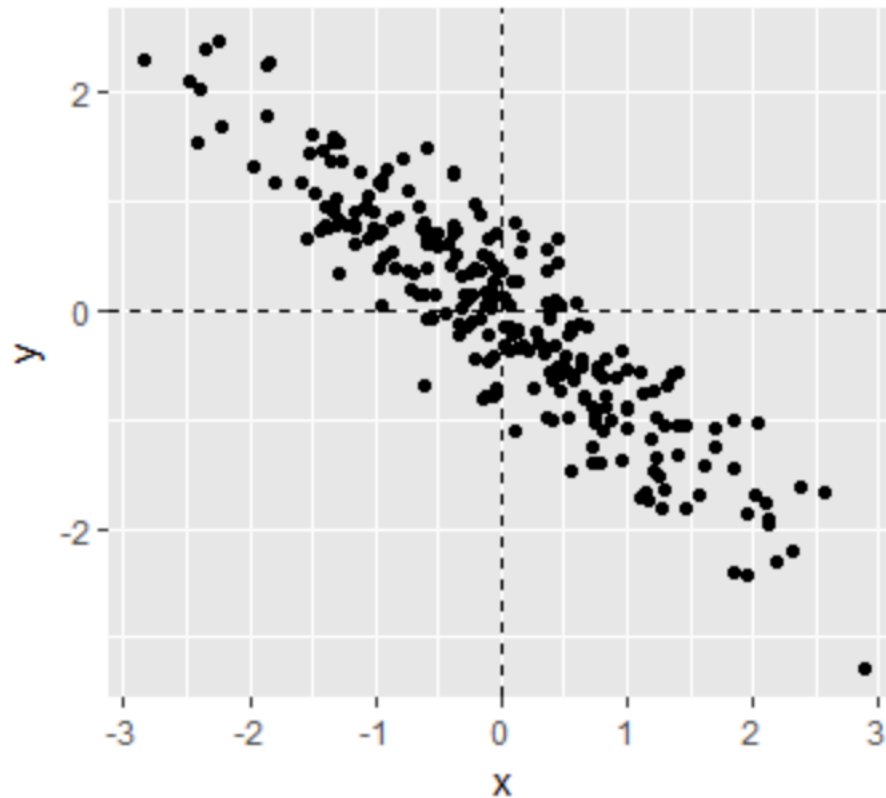


Figure 1: Scatter plot relationship x and y

From this figure, the correlation between x and y appears to be about:

- ☒ A. -0.9
- ☐ B. -0.2
- ☐ C. 0.9
- ☐ D. 2

4. Instead of running a Monte Carlo simulation with a sample size of 25 from our 179 father-son pairs, we now run our simulation with a sample size of 50.

Would you expect the **mean** of our sample correlation to increase, decrease, or stay approximately the same?

- ☐ A. Increase
- ☐ B. Decrease
- ☒ C. Stay approximately the same

5. Instead of running a Monte Carlo simulation with a sample size of 25 from our 179 father-son pairs, we now run our simulation with a sample size of 50.

Would you expect the **standard deviation** of our sample correlation to increase, decrease, or stay approximately the same?

- ☐ A. Increase
- ☒ B. Decrease
- ☐ C. Stay approximately the same

6. If X and Y are completely independent, what do you expect the value of the correlation coefficient to be?

- ☐ A. -1
- ☐ B. -0.5
- ☒ C. 0
- ☐ D. 0.5
- ☐ E. 1
- ☐ F. Not enough information to answer the question

7. Load the **Lahman** library. Filter the **Teams** data frame to include years from 1961 to 2001.

What is the correlation coefficient between number of runs per game and number of at bats per game?

```
Teams_small <- Teams %>% filter(yearID %in% 1961:2001)
cor(Teams_small$R/Teams_small$G, Teams_small$AB/Teams_small$G)
```

```
## [1] 0.6580976
```

8. Use the filtered **Teams** data frame from Question 7.

What is the correlation coefficient between win rate (number of wins per game) and number of errors per game?

```
cor(Teams_small$W/Teams_small$G, Teams_small$E/Teams_small$G)
```

```
## [1] -0.3396947
```

9. Use the filtered **Teams** data frame from Question 7.

What is the correlation coefficient between doubles (X2B) per game and triples (X3B) per game?

```
cor(Teams_small$X2B/Teams_small$G, Teams_small$X3B/Teams_small$G)
```

```
## [1] -0.01157404
```

Anscombe's Quartet/Stratification

There are three links to relevant sections of the textbook:

- [Correlation is not always a useful summary](#)
- [Conditional expectation](#)
- [The regression line](#)

Key points

- Correlation is not always a good summary of the relationship between two variables.
- The general idea of conditional expectation is that we stratify a population into groups and compute summaries in each group.
- A practical way to improve the estimates of the conditional expectations is to define strata of with similar values of x.
- If there is perfect correlation, the regression line predicts an increase that is the same number of SDs for both variables. If there is 0 correlation, then we don't use x at all for the prediction and simply predict the average μ_y . For values between 0 and 1, the prediction is somewhere in between. If the correlation is negative, we predict a reduction instead of an increase.

Code

```
# number of fathers with height 72 or 72.5 inches
sum(galton_heights$father == 72)
```

```
## [1] 8
```

```
sum(galton_heights$father == 72.5)
```

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

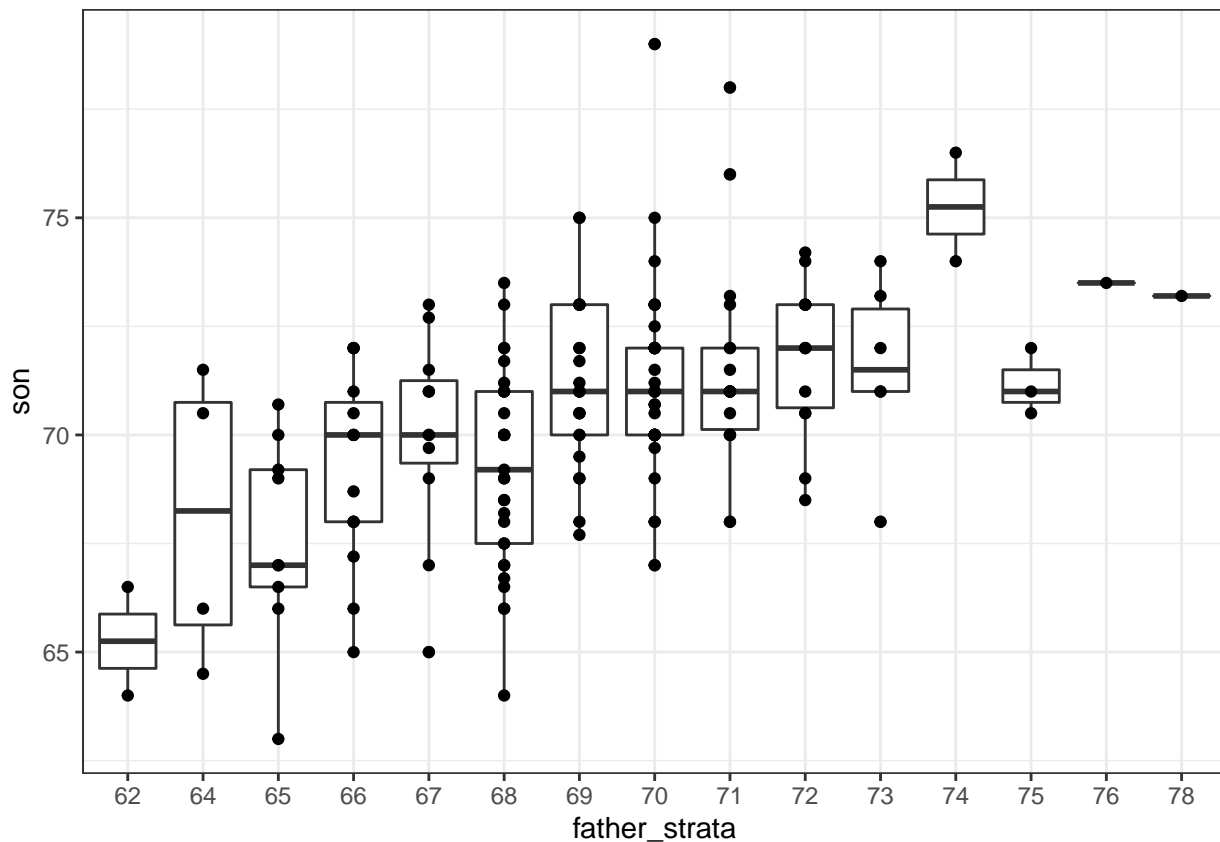
```
# predicted height of a son with a 72 inch tall father
```

```
conditional_avg <- galton_heights %>%  
  filter(round(father) == 72) %>%  
  summarize(avg = mean(son)) %>%  
  pull(avg)  
conditional_avg
```

```
## [1] 71.83571
```

```
# stratify fathers' heights to make a boxplot of son heights
```

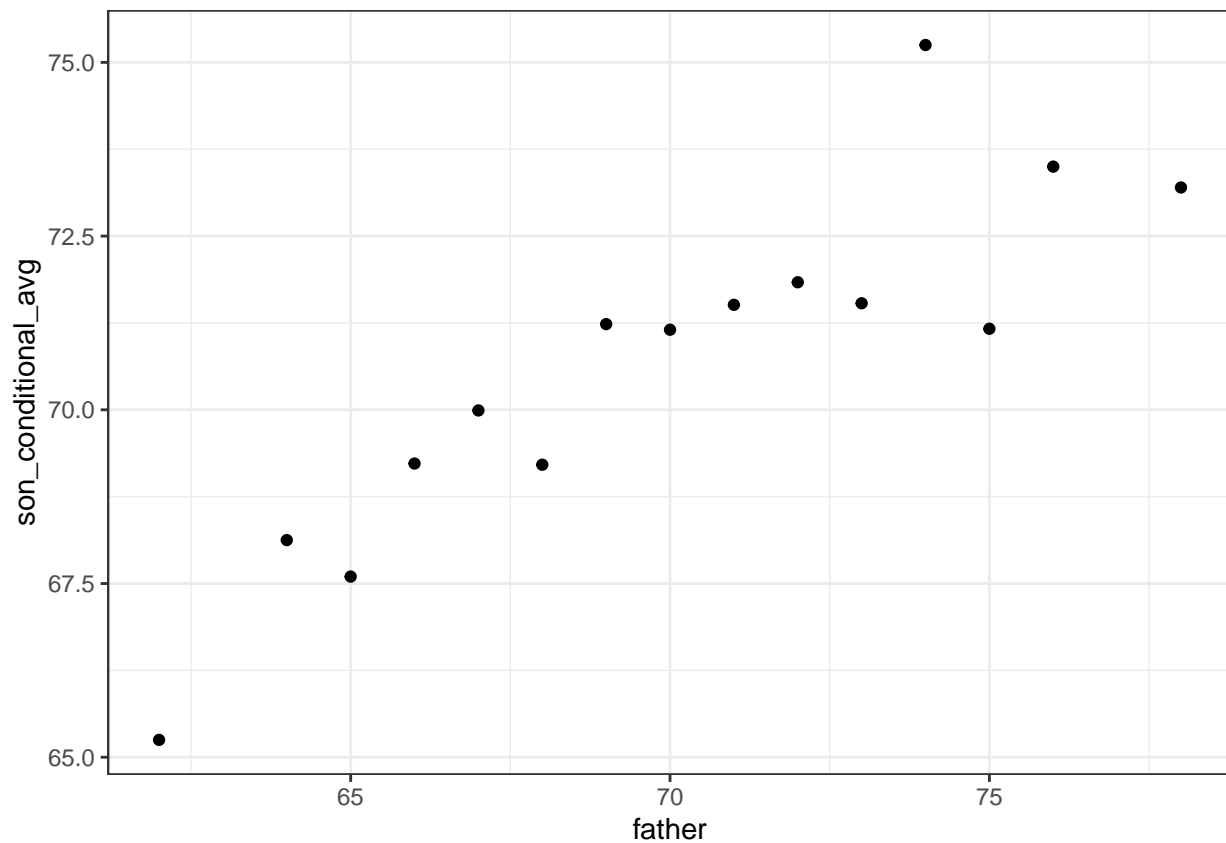
```
galton_heights %>% mutate(father_strata = factor(round(father))) %>%  
  ggplot(aes(father_strata, son)) +  
  geom_boxplot() +  
  geom_point()
```



```
# center of each boxplot
```

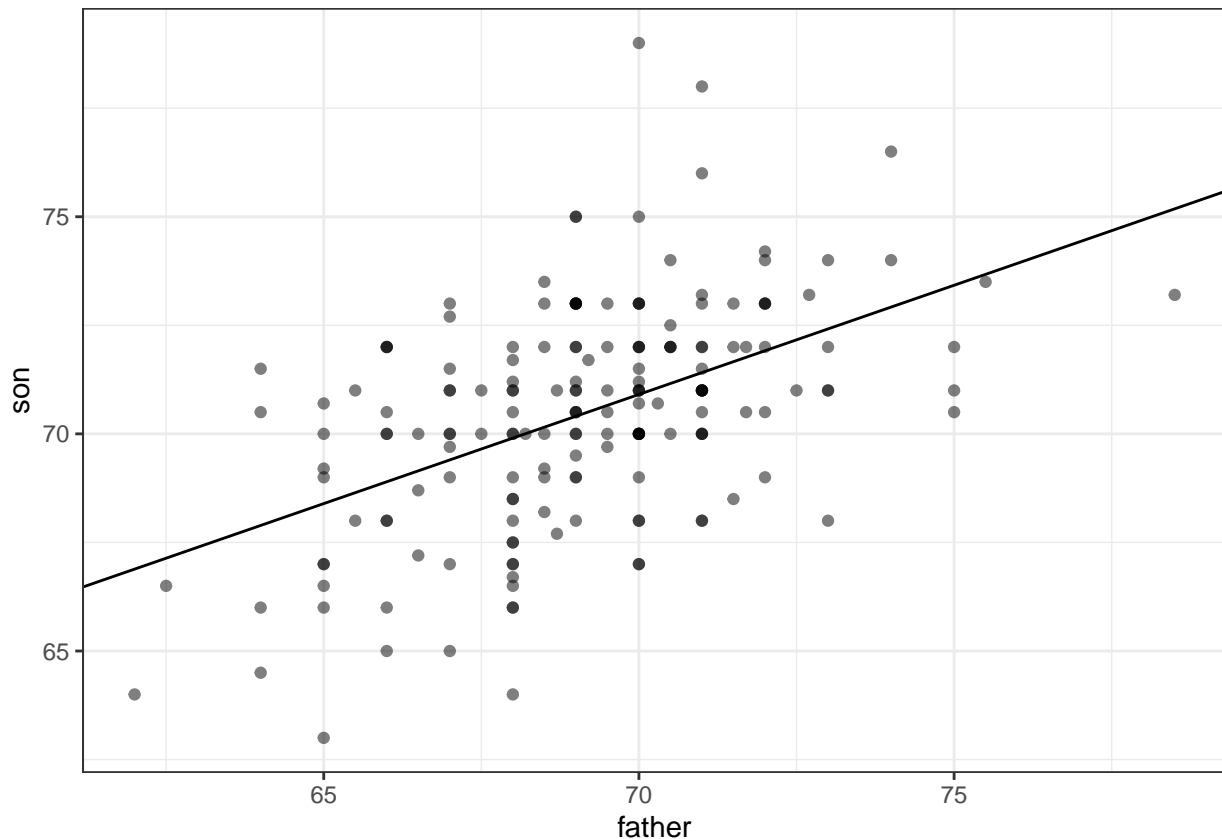
```
galton_heights %>%  
  mutate(father = round(father)) %>%  
  group_by(father) %>%  
  summarize(son_conditional_avg = mean(son)) %>%  
  ggplot(aes(father, son_conditional_avg)) +  
  geom_point()
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
# calculate values to plot regression line on original data
mu_x <- mean(galton_heights$father)
mu_y <- mean(galton_heights$son)
s_x <- sd(galton_heights$father)
s_y <- sd(galton_heights$son)
r <- cor(galton_heights$father, galton_heights$son)
m <- r * s_y/s_x
b <- mu_y - m*mu_x

# add regression line to plot
galton_heights %>%
  ggplot(aes(father, son)) +
  geom_point(alpha = 0.5) +
  geom_abline(intercept = b, slope = m)
```



Bivariate Normal Distribution

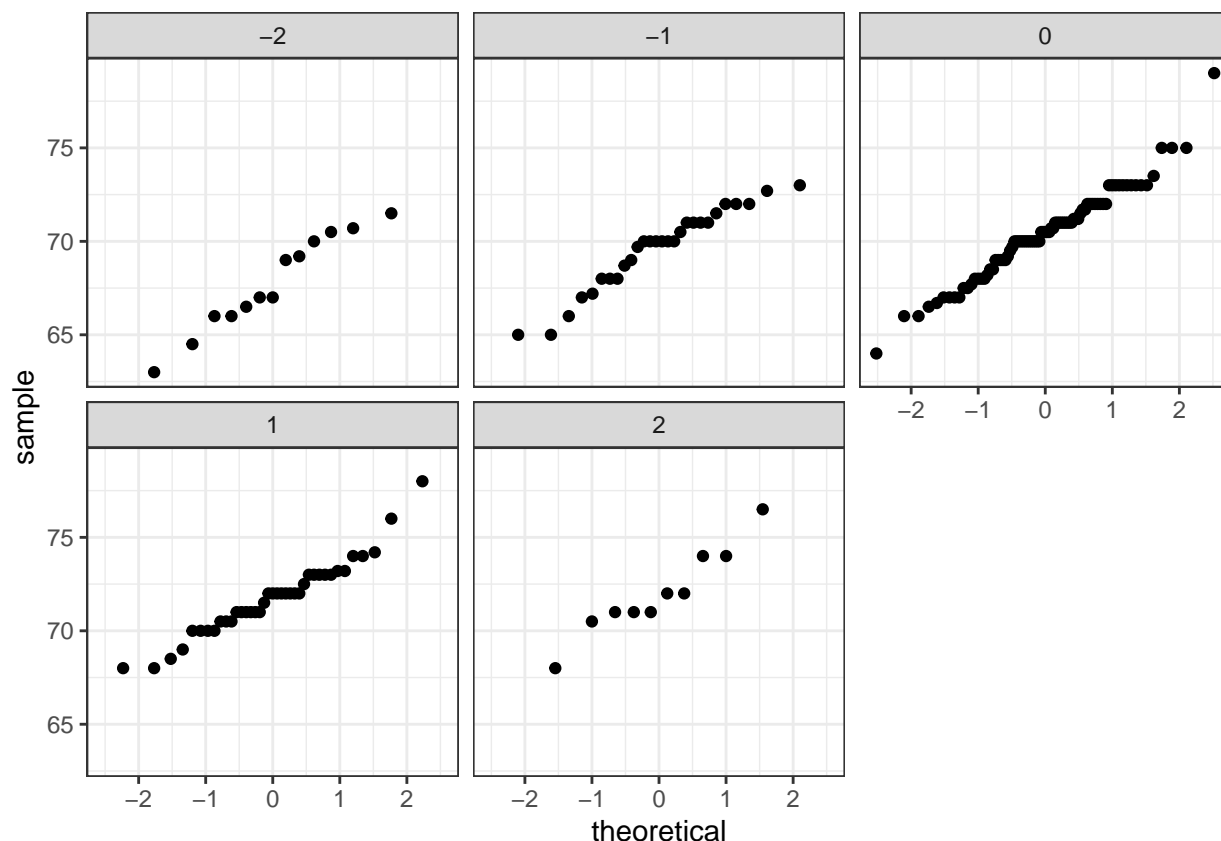
There is a link to the relevant section of the textbook: [Bivariate normal distribution \(advanced\)](#)

Key points

- When a pair of random variables are approximated by the bivariate normal distribution, scatterplots look like ovals. They can be thin (high correlation) or circle-shaped (no correlation).
- When two variables follow a bivariate normal distribution, computing the regression line is equivalent to computing conditional expectations.
- We can obtain a much more stable estimate of the conditional expectation by finding the regression line and using it to make predictions.

Code

```
galton_heights %>%
  mutate(z_father = round((father - mean(father)) / sd(father))) %>%
  filter(z_father %in% -2:2) %>%
  ggplot() +
  stat_qq(aes(sample = son)) +
  facet_wrap(~ z_father)
```



Variance Explained

There is a link to the relevant section of the textbook: [Variance explained](#)

Key points

- Conditioning on a random variable X can help to reduce variance of response variable Y .
- The standard deviation of the conditional distribution is $SD(Y | X = x) = \sigma_y \sqrt{1 - \rho^2}$, which is smaller than the standard deviation without conditioning σ_y .
- Because variance is the standard deviation squared, the variance of the conditional distribution is $\sigma_y^2(1 - \rho^2)$.
- In the statement “ X explains such and such percent of the variability,” the percent value refers to the variance. The variance decreases by ρ^2 percent.
- The “variance explained” statement only makes sense when the data is approximated by a bivariate normal distribution.

There are Two Regression Lines

There is a link to the relevant section of the textbook: [Warning: there are two regression lines](#)

Key point

There are two different regression lines depending on whether we are taking the expectation of Y given X or taking the expectation of X given Y .

Code

```
# compute a regression line to predict the son's height from the father's height
mu_x <- mean(galton_heights$father)
mu_y <- mean(galton_heights$son)
```

```
s_x <- sd(galton_heights$father)
s_y <- sd(galton_heights$son)
r <- cor(galton_heights$father, galton_heights$son)
m_1 <- r * s_y / s_x
b_1 <- mu_y - m_1*mu_x
m_1 # slope 1
```

```
## [1] 0.5027904
```

```
b_1 # intercept 1
```

```
## [1] 35.71249
```

```
# compute a regression line to predict the father's height from the son's height
```

```
m_2 <- r * s_x / s_y
b_2 <- mu_x - m_2*mu_y
m_2 # slope 2
```

```
## [1] 0.4986676
```

```
b_2 # intercept 2
```

```
## [1] 33.96539
```

Assessment - Stratification and Variance Explained, Part 1

1. Look at the figure below. The slope of the regression line in this figure is equal to what, in words?

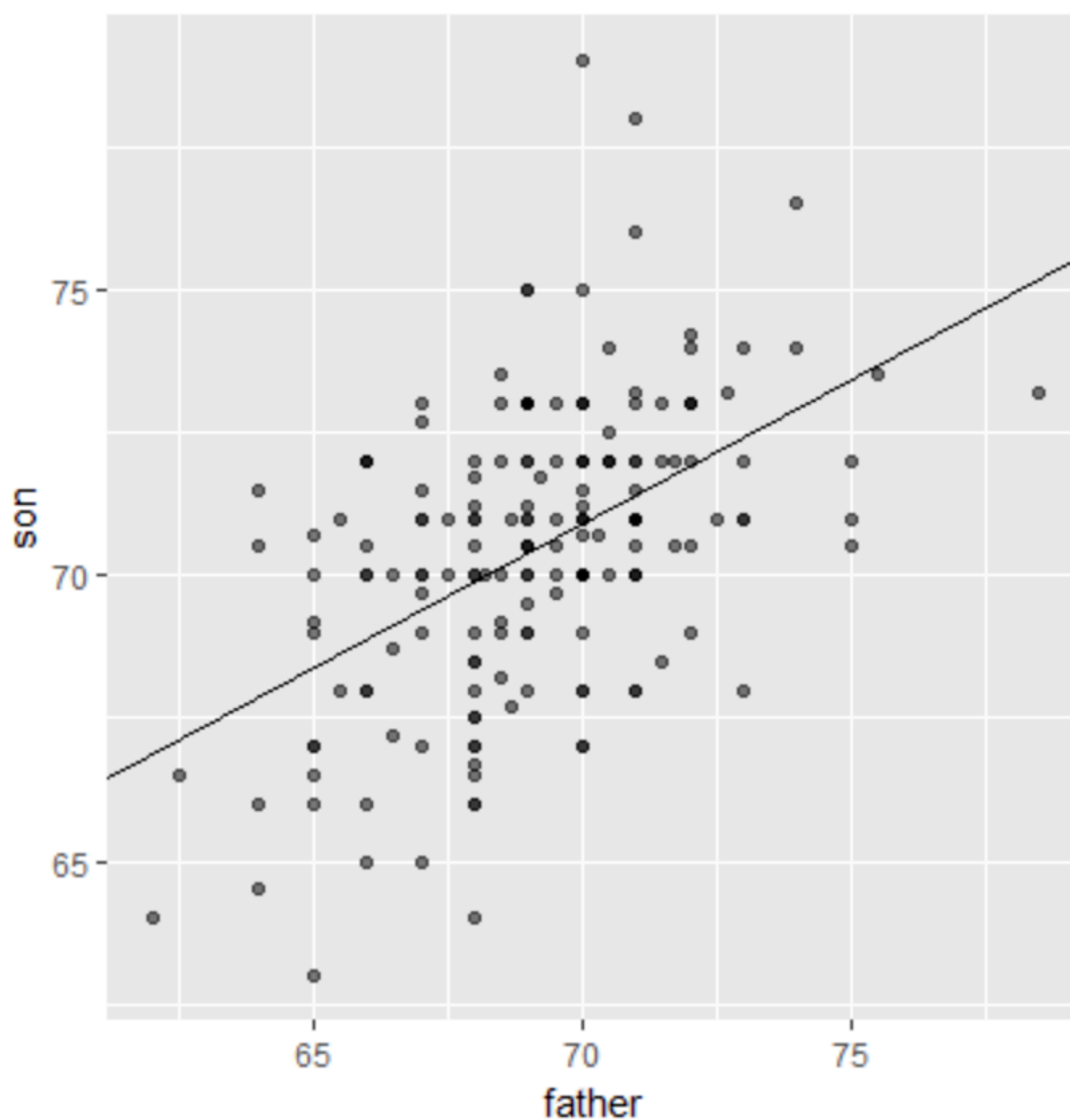
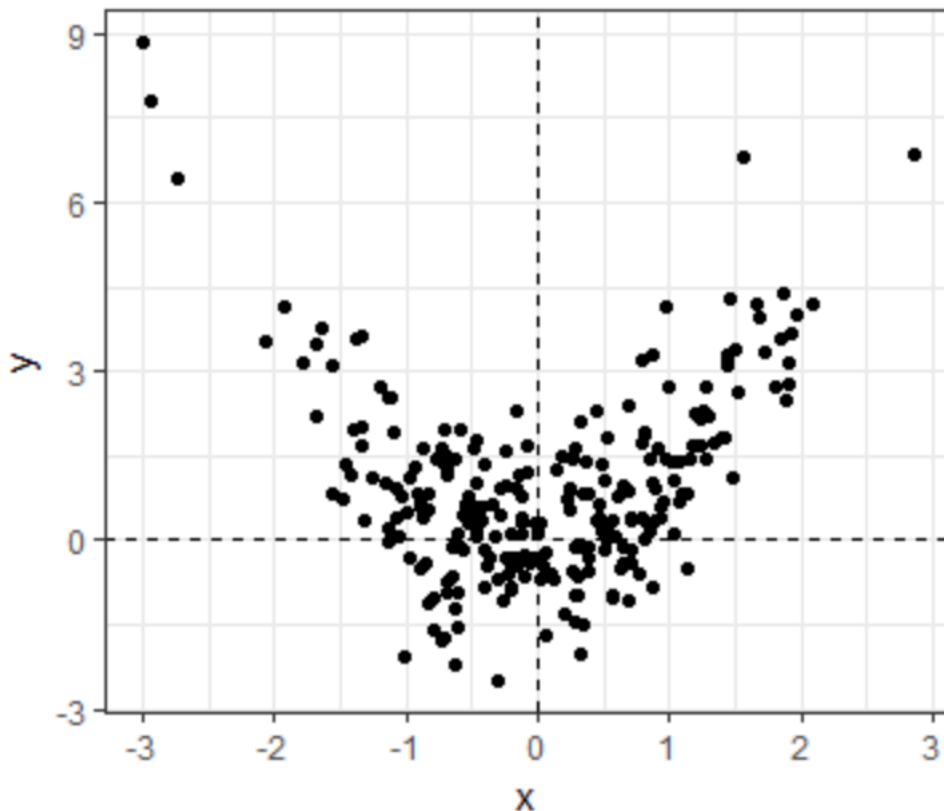


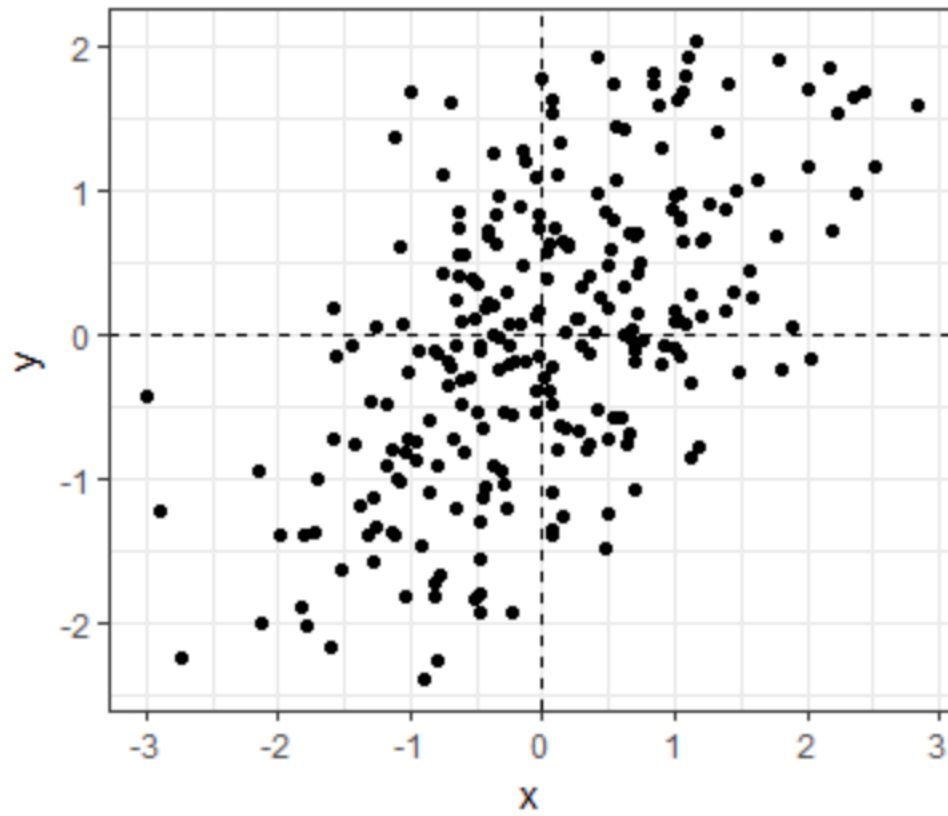
Figure 2: Scatter plot and regression line of son and father heights

- ☒ A. Slope = (correlation coefficient of son and father heights) * (standard deviation of sons' heights / standard deviation of fathers' heights)
 - ☐ B. Slope = (correlation coefficient of son and father heights) * (standard deviation of fathers' heights / standard deviation of sons' heights)
 - ☐ C. Slope = (correlation coefficient of son and father heights) / (standard deviation of sons' heights * standard deviation of fathers' heights)
 - ☐ D. Slope = (mean height of fathers) - (correlation coefficient of son and father heights * mean height of sons).
2. Why does the regression line simplify to a line with intercept zero and slope ρ when we standardize our x and y variables? Try the simplification on your own first!

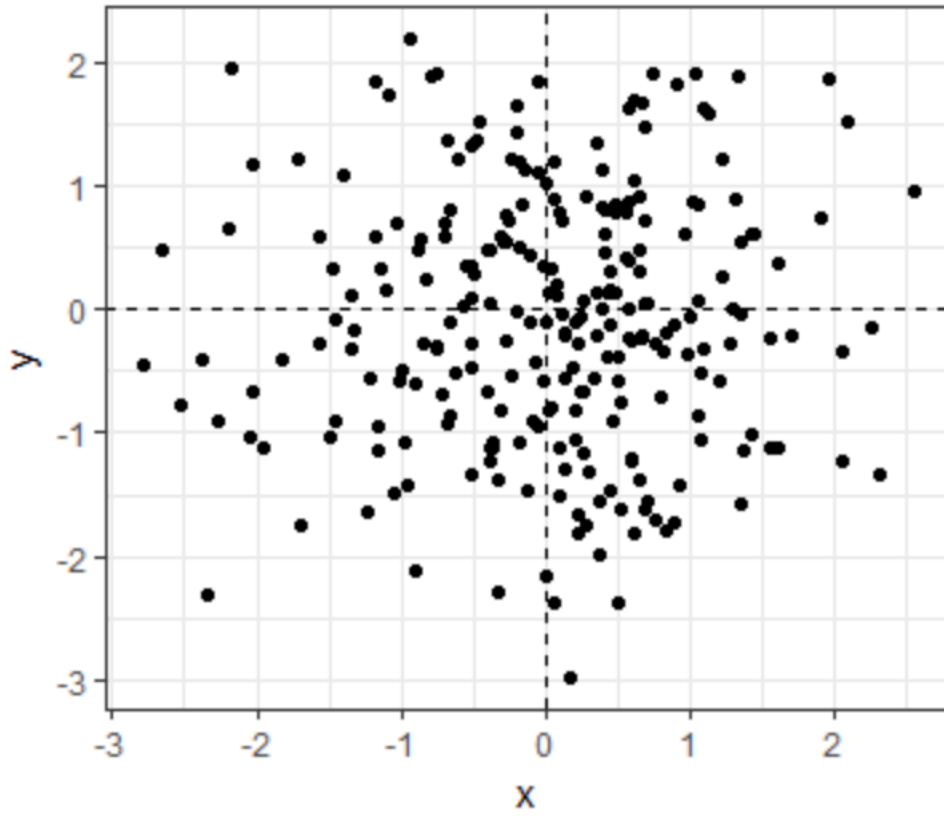
- ☐ A. When we standardize variables, both x and y will have a mean of one and a standard deviation of zero. When you substitute this into the formula for the regression line, the terms cancel out until we have the following equation: $y_i = \rho x_i$.
 - ☒ B. When we standardize variables, both x and y will have a mean of zero and a standard deviation of one. When you substitute this into the formula for the regression line, the terms cancel out until we have the following equation: $y_i = \rho x_i$.
 - ☐ C. When we standardize variables, both x and y will have a mean of zero and a standard deviation of one. When you substitute this into the formula for the regression line, the terms cancel out until we have the following equation: $y_i = \rho + x_i$.
3. What is a limitation of calculating conditional means?
- ☒ A. Each stratum we condition on (e.g., a specific father's height) may not have many data points.
 - ☒ B. Because there are limited data points for each stratum, our average values have large standard errors.
 - ☒ C. Conditional means are less stable than a regression line.
 - ☐ D. Conditional means are a useful theoretical tool but cannot be calculated.
4. A regression line is the best prediction of Y given we know the value of X when:
- ☒ A. X and Y follow a bivariate normal distribution.
 - ☐ B. Both X and Y are normally distributed.
 - ☐ C. Both X and Y have been standardized.
 - ☐ D. There are at least 25 X-Y pairs.
5. Which one of the following scatterplots depicts an x and y distribution that is NOT well-approximated by the bivariate normal distribution?
- ☒ A.



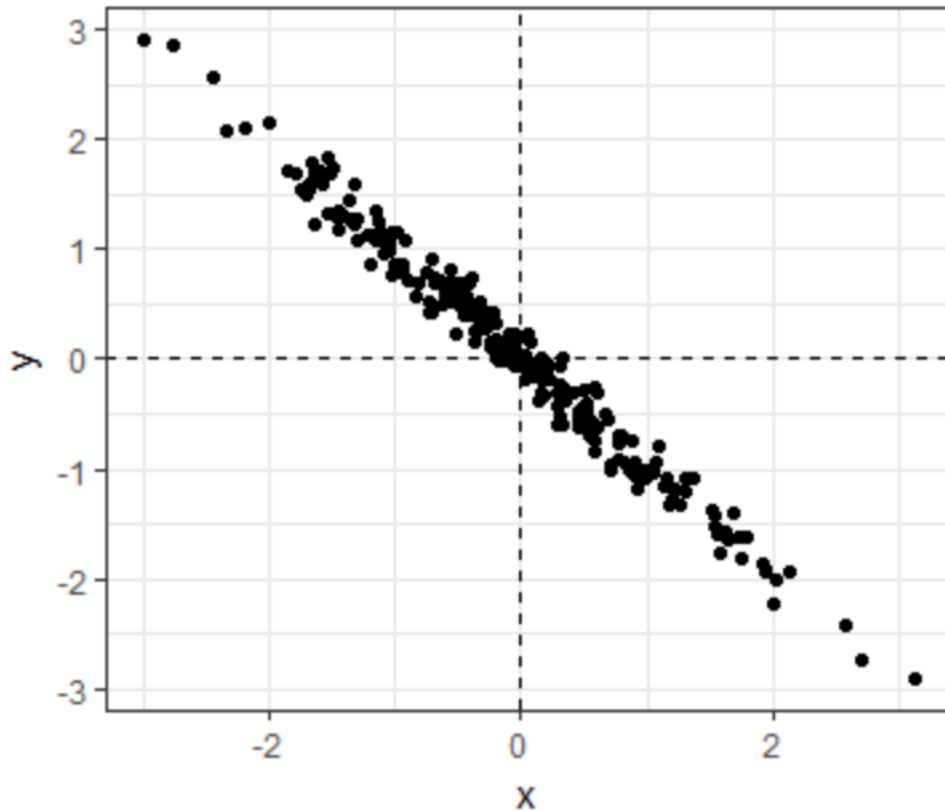
- [] B.



- [] C.



- [] D.



6. We previously calculated that the correlation coefficient between fathers' and sons' heights is 0.5.

Given this, what percent of the variation in sons' heights is explained by fathers' heights?

- ☐ A. 0%
- ☒ B. 25%
- ☐ C. 50%
- ☐ D. 75%

When two variables follow a bivariate normal distribution, the variation explained can be calculated as $\rho^2 \times 100$.

7. Suppose the correlation between father and son's height is 0.5, the standard deviation of fathers' heights is 2 inches, and the standard deviation of sons' heights is 3 inches.

Given a one inch increase in a father's height, what is the predicted change in the son's height?

- ☐ A. 0.333
- ☐ B. 0.5
- ☐ C. 0.667
- ☒ D. 0.75
- ☐ E. 1
- ☐ F. 1.5

The slope of the regression line is calculated by multiplying the correlation coefficient by the ratio of the standard deviation of son heights and standard deviation of father heights: $\sigma_{son}/\sigma_{father}$.

Assessment - Stratification and Variance Explained, Part 2

In the second part of this assessment, you'll analyze a set of mother and daughter heights, also from GaltonFamilies.

Define `female_heights`, a set of mother and daughter heights sampled from `GaltonFamilies`, as follows:

```
set.seed(1989, sample.kind="Rounding") #if you are using R 3.6 or later

## Warning in set.seed(1989, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used

female_heights <- GaltonFamilies%>%
  filter(gender == "female") %>%
  group_by(family) %>%
  sample_n(1) %>%
  ungroup() %>%
  select(mother, childHeight) %>%
  rename(daughter = childHeight)
```

8. Calculate the mean and standard deviation of mothers' heights, the mean and standard deviation of daughters' heights, and the correlation coefficient between mother and daughter heights.

Mean of mothers' heights

```
mean(female_heights$mother)
```

```
## [1] 64.125
```

Standard deviation of mothers' heights

```
sd(female_heights$mother)
```

```
## [1] 2.289292
```

Mean of daughters' heights

```
mean(female_heights$daughter)
```

```
## [1] 64.28011
```

Standard deviation of daughters' heights

```
sd(female_heights$daughter)
```

```
## [1] 2.39416
```

Correlation coefficient

```
cor(female_heights$mother, female_heights$daughter)
```

```
## [1] 0.3245199
```

9. Calculate the slope and intercept of the regression line predicting daughters' heights given mothers' heights. Given an increase in mother's height by 1 inch, how many inches is the daughter's height expected to change?

Slope of regression line predicting daughters' height from mothers' heights

```
r <- cor(female_heights$mother, female_heights$daughter)
s_y <- sd(female_heights$daughter)
s_x <- sd(female_heights$mother)
r * s_y/s_x
```

```
## [1] 0.3393856
```

Intercept of regression line predicting daughters' height from mothers' heights

```
mu_y <- mean(female_heights$daughter)
mu_x <- mean(female_heights$mother)
mu_y - (r * s_y/s_x)*mu_x
```

```
## [1] 42.51701
```

Change in daughter's height in inches given a 1 inch increase in the mother's height

```
r * s_y/s_x
```

```
## [1] 0.3393856
```

10. What percent of the variability in daughter heights is explained by the mother's height?

```
r^2*100
```

```
## [1] 10.53132
```

11. A mother has a height of 60 inches.

What is the conditional expected value of her daughter's height given the mother's height?

```
m = r * s_y/s_x
b = mu_y - (r * s_y/s_x)*mu_x
x = 60
m*x+b
```

```
## [1] 62.88015
```

Section 2 - Linear Models Overview

In the **Linear Models** section, you will learn how to do linear regression.

After completing this section, you will be able to:

- Use **multivariate regression** to adjust for confounders.
- Write **linear models** to describe the relationship between two or more variables.
- Calculate the **least squares estimates** for a regression model using the **lm** function.
- Understand the differences between **tibbles** and **data frames**.
- Use the **do()** function to bridge R functions and the tidyverse.
- Use the **tidy()**, **glance()**, and **augment()** functions from the **broom** package.
- Apply linear regression to **measurement error models**.

This section has four parts: **Introduction to Linear Models**, **Least Squares Estimates**, **Tibbles**, **do**, **and broom**, and **Regression and Baseball**.

Confounding: Are BBs More Predictive?

The textbook for this section is available [here](#)

Key points

- Association is not causation!
- Although it may appear that BB cause runs, it is actually the HR that cause most of these runs. We say that BB are **confounded** with HR.
- Regression can help us account for confounding.

Code

```

# find regression line for predicting runs from BBs
bb_slope <- Teams %>%
  filter(yearID %in% 1961:2001 ) %>%
  mutate(BB_per_game = BB/G, R_per_game = R/G) %>%
  lm(R_per_game ~ BB_per_game, data = .) %>%
  .$coef %>%
  .[2]
bb_slope

## BB_per_game
## 0.7353288

# compute regression line for predicting runs from singles
singles_slope <- Teams %>%
  filter(yearID %in% 1961:2001 ) %>%
  mutate(Singles_per_game = (H-HR-X2B-X3B)/G, R_per_game = R/G) %>%
  lm(R_per_game ~ Singles_per_game, data = .) %>%
  .$coef %>%
  .[2]
singles_slope

## Singles_per_game
## 0.4494253

# calculate correlation between HR, BB and singles
Teams %>%
  filter(yearID %in% 1961:2001 ) %>%
  mutate(Singles = (H-HR-X2B-X3B)/G, BB = BB/G, HR = HR/G) %>%
  summarize(cor(BB, HR), cor(Singles, HR), cor(BB,Singles))

## cor(BB, HR) cor(Singles, HR) cor(BB, Singles)
## 1 0.4039313 -0.1737435 -0.05603822

```

Stratification and Multivariate Regression

The textbook for this section is available [here](#)

Key points

- A first approach to check confounding is to keep HRs fixed at a certain value and then examine the relationship between BB and runs.
- The slopes of BB after stratifying on HR are reduced, but they are not 0, which indicates that BB are helpful for producing runs, just not as much as previously thought.

Code

```

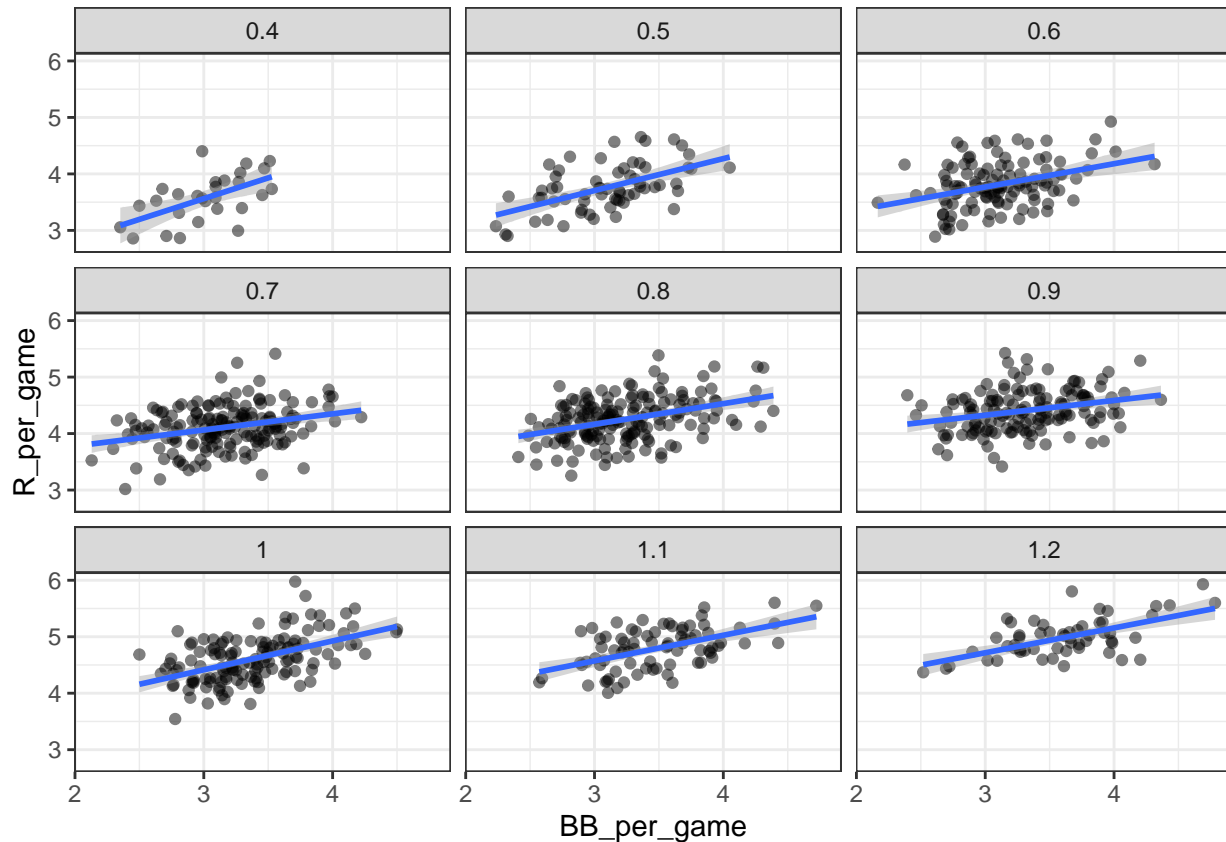
# stratify HR per game to nearest 10, filter out strata with few points
dat <- Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(HR_strata = round(HR/G, 1),
         BB_per_game = BB / G,
         R_per_game = R / G) %>%
  filter(HR_strata >= 0.4 & HR_strata <=1.2)

# scatterplot for each HR stratum
dat %>%
  ggplot(aes(BB_per_game, R_per_game)) +
  geom_point(alpha = 0.5) +

```

```
geom_smooth(method = "lm") +  
facet_wrap( ~ HR_strata)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
# calculate slope of regression line after stratifying by HR  
dat %>%  
  group_by(HR_strata) %>%  
  summarize(slope = cor(BB_per_game, R_per_game)*sd(R_per_game)/sd(BB_per_game))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 9 x 2  
##   HR_strata slope  
##   <dbl> <dbl>  
## 1      0.4 0.734  
## 2      0.5 0.566  
## 3      0.6 0.412  
## 4      0.7 0.285  
## 5      0.8 0.365  
## 6      0.9 0.261  
## 7       1  0.511  
## 8      1.1 0.454  
## 9      1.2 0.440
```

```
# stratify by BB  
dat <- Teams %>% filter(yearID %in% 1961:2001) %>%  
  mutate(BB_strata = round(BB/G, 1),
```

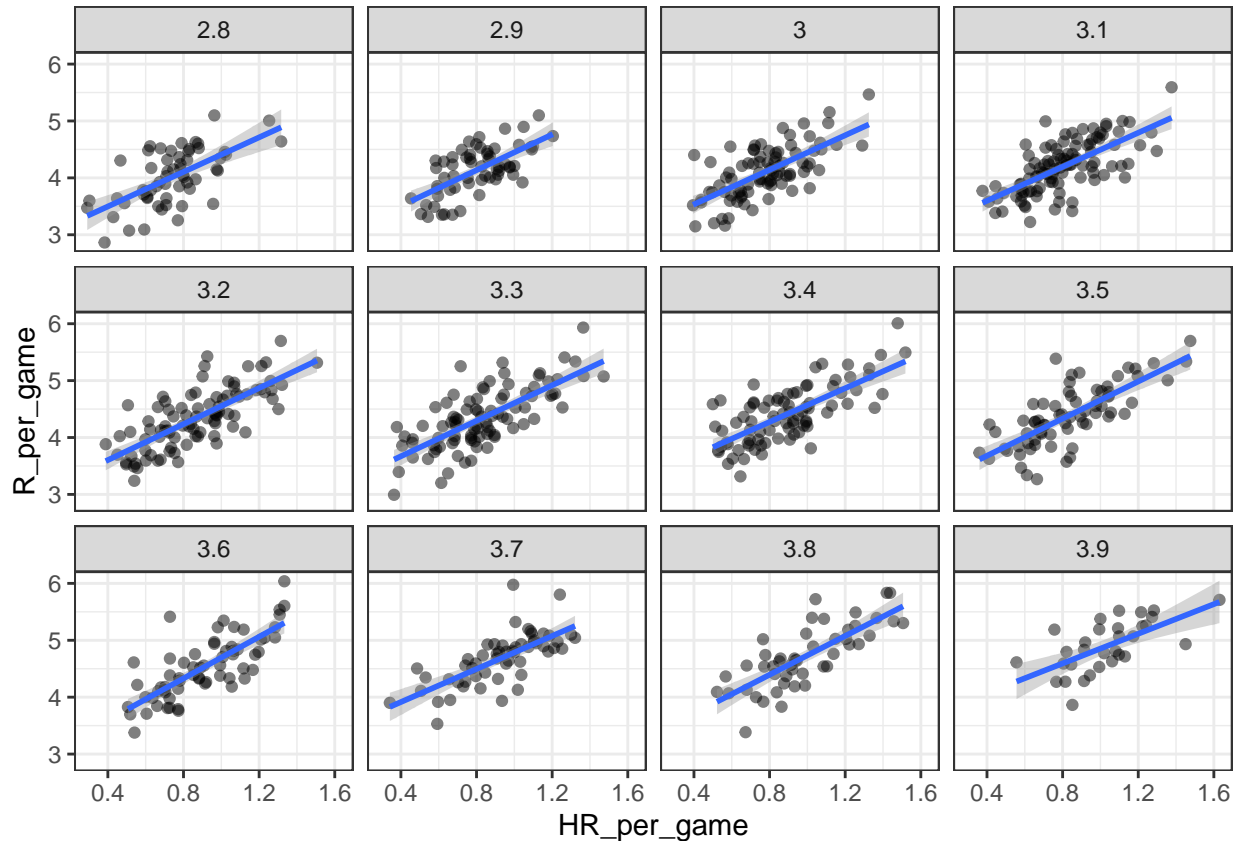
```

HR_per_game = HR / G,
R_per_game = R / G) %>%
filter(BB_strata >= 2.8 & BB_strata <=3.9)

# scatterplot for each BB stratum
dat %>% ggplot(aes(HR_per_game, R_per_game)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm") +
  facet_wrap( ~ BB_strata)

```

```
## `geom_smooth()` using formula 'y ~ x'
```



```

# slope of regression line after stratifying by BB
dat %>%
  group_by(BB_strata) %>%
  summarize(slope = cor(HR_per_game, R_per_game)*sd(R_per_game)/sd(HR_per_game))

```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```

## # A tibble: 12 x 2
##   BB_strata slope
##   <dbl> <dbl>
## 1     2.8  1.52
## 2     2.9  1.57
## 3      3   1.52
## 4     3.1  1.49
## 5     3.2  1.58
## 6     3.3  1.56

```

##	7	3.4	1.48
##	8	3.5	1.63
##	9	3.6	1.83
##	10	3.7	1.45
##	11	3.8	1.70
##	12	3.9	1.30

Linear Models

The textbook for this section is available [here](#)

Key points

- “Linear” here does not refer to lines, but rather to the fact that the conditional expectation is a linear combination of known quantities.
- In Galton’s model, we assume Y (son’s height) is a linear combination of a constant and X (father’s height) plus random noise. We further assume that ϵ_i are independent from each other, have expected value 0 and the standard deviation σ which does not depend on i .
- Note that if we further assume that ϵ is normally distributed, then the model is exactly the same one we derived earlier by assuming bivariate normal data.
- We can subtract the mean from X to make β_0 more interpretable.

Assessment: Introduction to Linear Models

1. When we stratified our regression lines for runs per game vs. bases on balls by the number of home runs, what happened?
 - [X] A. The slope of runs per game vs. bases on balls within each stratum was reduced because we removed confounding by home runs.
 - [] B. The slope of runs per game vs. bases on balls within each stratum was reduced because there were fewer data points.
 - [] C. The slope of runs per game vs. bases on balls within each stratum increased after we removed confounding by home runs.
 - [] D. The slope of runs per game vs. bases on balls within each stratum stayed about the same as the original slope.
2. We run a linear model for sons’ heights vs. fathers’ heights using the Galton height data, and get the following results:

```
> lm(son ~ father, data = galton_heights)

Call:
lm(formula = son ~ father, data = galton_heights)

Coefficients:
(Intercept)    father
      35.71         0.50
```

Interpret the numeric coefficient for “father.”

- [] A. For every inch we increase the son’s height, the predicted father’s height increases by 0.5 inches.
 - [X] B. For every inch we increase the father’s height, the predicted son’s height grows by 0.5 inches.
 - [] C. For every inch we increase the father’s height, the predicted son’s height is 0.5 times greater.
3. We want the intercept term for our model to be more interpretable, so we run the same model as before but now we subtract the mean of fathers’ heights from each individual father’s height to create a new variable centered at zero.

```
galton_heights <- galton_heights %>%
  mutate(father_centered=father - mean(father))
```

We run a linear model using this centered fathers' height variable.

```
> lm(son ~ father_centered, data = galton_heights)

Call:
lm(formula = son ~ father_centered, data = galton_heights)

Coefficients:
(Intercept)      father_centered
      70.45           0.50
```

Interpret the numeric coefficient for the intercept.

- [X] A. The height of a son of a father of average height is 70.45 inches.
- [] B. The height of a son when a father's height is zero is 70.45 inches.
- [] C. The height of an average father is 70.45 inches.

4. Suppose we fit a multivariate regression model for expected runs based on BB and HR:

$$E[R|BB = x_1, HR = x_2] = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Suppose we fix $BB = x_1$. Then we observe a linear relationship between runs and HR with intercept of:

- [] A. β_0
- [] B. $\beta_0 + \beta_2 x_2$
- [X] C. $\beta_0 + \beta_1 x_1$
- [] D. $\beta_0 + \beta_2 x_1$

5. Which of the following are assumptions for the errors ϵ_i in a linear regression model? Check ALL correct answers.

- [X] A. The ϵ_i are independent of each other
- [X] B. The ϵ_i have expected value 0
- [X] C. The variance of ϵ_i is a constant

Least Squares Estimates (LSE)

The textbook for this section is available [here](#)

Key points

- For regression, we aim to find the coefficient values that minimize the distance of the fitted model to the data.
- Residual sum of squares (RSS) measures the distance between the true value and the predicted value given by the regression line. The values that minimize the RSS are called the least squares estimates (LSE).
- We can use partial derivatives to get the values for β_0 and β_1 in Galton's data.

Code

```
# compute RSS for any pair of beta0 and beta1 in Galton's data
set.seed(1983)
galton_heights <- GaltonFamilies %>%
  filter(gender == "male") %>%
  group_by(family) %>%
  sample_n(1) %>%
  ungroup() %>%
```

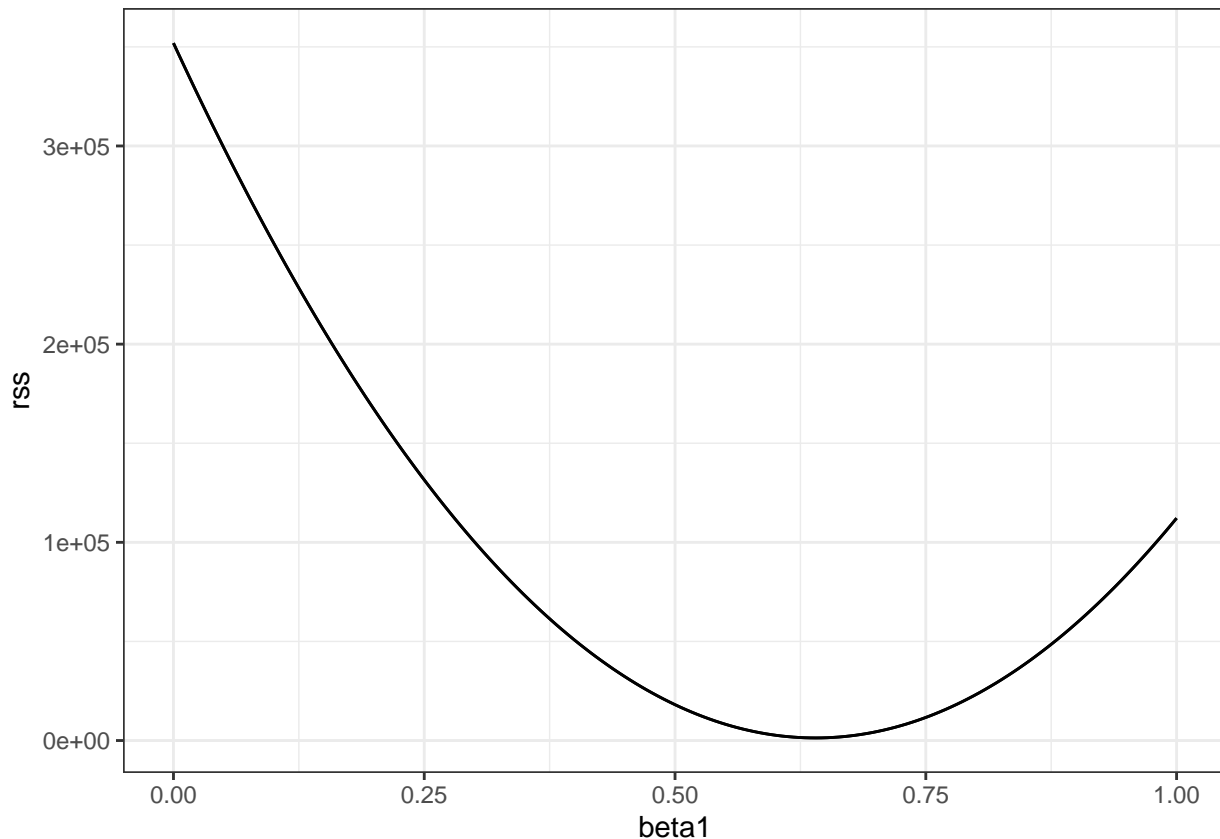


```

select(father, childHeight) %>%
  rename(son = childHeight)
rss <- function(beta0, beta1, data){
  resid <- galton_heights$son - (beta0+beta1*galton_heights$father)
  return(sum(resid^2))
}

# plot RSS as a function of beta1 when beta0=25
beta1 = seq(0, 1, len=nrow(galton_heights))
results <- data.frame(beta1 = beta1,
                      rss = sapply(beta1, rss, beta0 = 25))
results %>% ggplot(aes(beta1, rss)) + geom_line() +
  geom_line(aes(beta1, rss))

```



The lm Function

The textbook for this section is available [here](#)

Key points

- When calling the `lm()` function, the variable that we want to predict is put to the left of the `~` symbol, and the variables that we use to predict is put to the right of the `~` symbol. The intercept is added automatically.
- LSEs are random variables.

Code

```

# fit regression line to predict son's height from father's height
fit <- lm(son ~ father, data = galton_heights)

```

```
fit

##
## Call:
## lm(formula = son ~ father, data = galton_heights)
##
## Coefficients:
## (Intercept)      father
##      38.7646      0.4411

# summary statistics
summary(fit)

##
## Call:
## lm(formula = son ~ father, data = galton_heights)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -9.4228 -1.7022  0.0333  1.5670  9.3567
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 38.76457     5.41093   7.164 2.03e-11 ***
## father       0.44112     0.07825   5.637 6.72e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.659 on 177 degrees of freedom
## Multiple R-squared:  0.1522, Adjusted R-squared:  0.1474
## F-statistic: 31.78 on 1 and 177 DF,  p-value: 6.719e-08
```

LSE are Random Variables

The textbook for this section is available [here](#)

Key points

- Because they are derived from the samples, LSE are random variables.
- β_0 and β_1 appear to be normally distributed because the central limit theorem plays a role.
- The t-statistic depends on the assumption that ϵ follows a normal distribution.

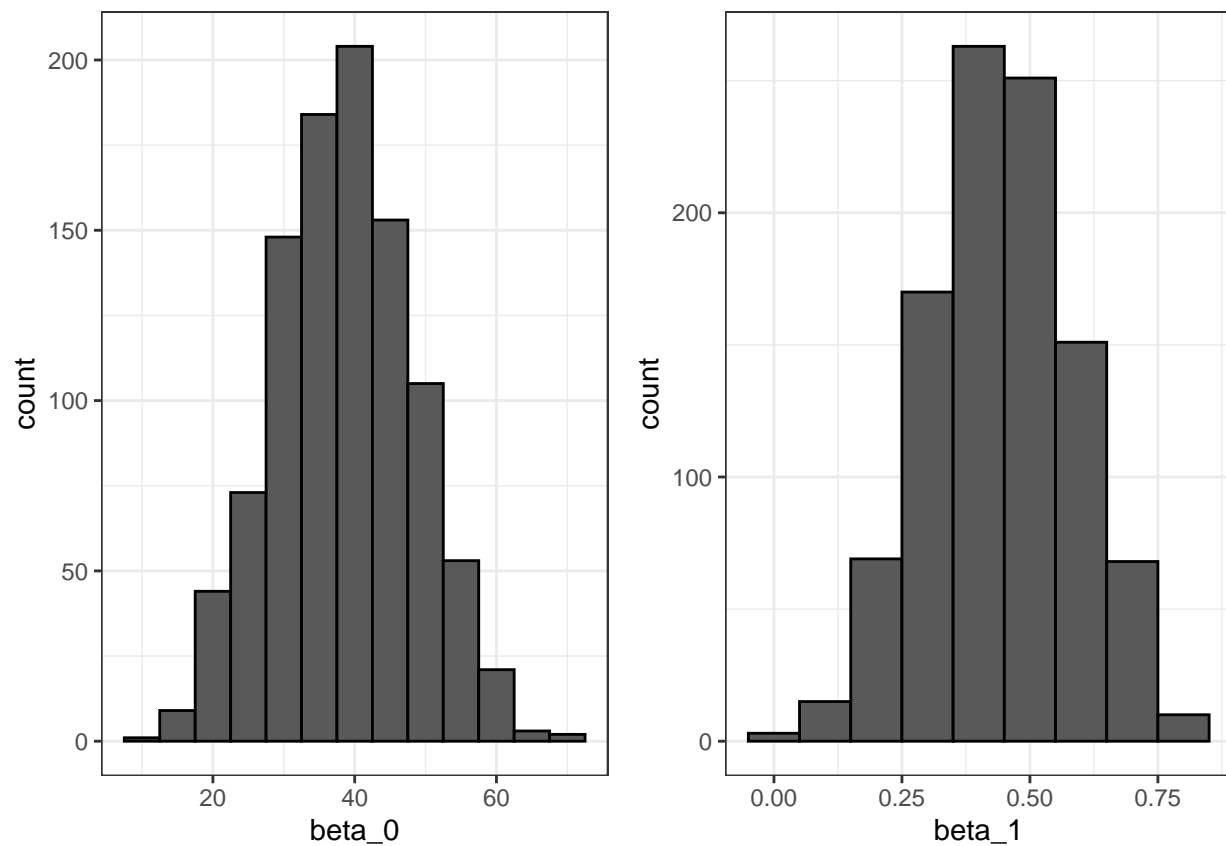
Code

```
# Monte Carlo simulation
B <- 1000
N <- 50
lse <- replicate(B, {
  sample_n(galton_heights, N, replace = TRUE) %>%
    lm(son ~ father, data = .) %>%
    .$coef
})
lse <- data.frame(beta_0 = lse[1,], beta_1 = lse[2,])

# Plot the distribution of beta_0 and beta_1
if(!require(gridExtra)) install.packages("gridExtra")
```

```
## Loading required package: gridExtra
##
## Attaching package: 'gridExtra'
## The following object is masked from 'package:dplyr':
##
##      combine
```

```
library(gridExtra)
p1 <- lse %>% ggplot(aes(beta_0)) + geom_histogram(binwidth = 5, color = "black")
p2 <- lse %>% ggplot(aes(beta_1)) + geom_histogram(binwidth = 0.1, color = "black")
grid.arrange(p1, p2, ncol = 2)
```



```
# summary statistics
sample_n(galton_heights, N, replace = TRUE) %>%
  lm(son ~ father, data = .) %>%
  summary %>%
  .$coef
```

```
##              Estimate Std. Error  t value    Pr(>|t|)
## (Intercept) 34.4729422  8.6021831  4.007464 0.0002129225
## father      0.4990193  0.1240572  4.022493 0.0002030210
```

```
lse %>% summarize(se_0 = sd(beta_0), se_1 = sd(beta_1))
```

```
##      se_0      se_1
## 1 9.683973 0.1411404
```

Advanced Note on LSE

Although interpretation is not straight-forward, it is also useful to know that the LSE can be strongly correlated, which can be seen using this code:

```
lse %>% summarize(cor(beta_0, beta_1))
```

```
##   cor(beta_0, beta_1)
## 1                -0.9993386
```

However, the correlation depends on how the predictors are defined or transformed.

Here we standardize the father heights, which changes x_i to $x_i - \bar{x}$.

```
B <- 1000
N <- 50
lse <- replicate(B, {
  sample_n(galton_heights, N, replace = TRUE) %>%
  mutate(father = father - mean(father)) %>%
  lm(son ~ father, data = .) %>% .$.coef
})
```

Observe what happens to the correlation in this case:

```
cor(lse[1,], lse[2,])
```

```
## [1] 0.1100929
```

Predicted Variables are Random Variables

The textbook for this section is available [here](#)

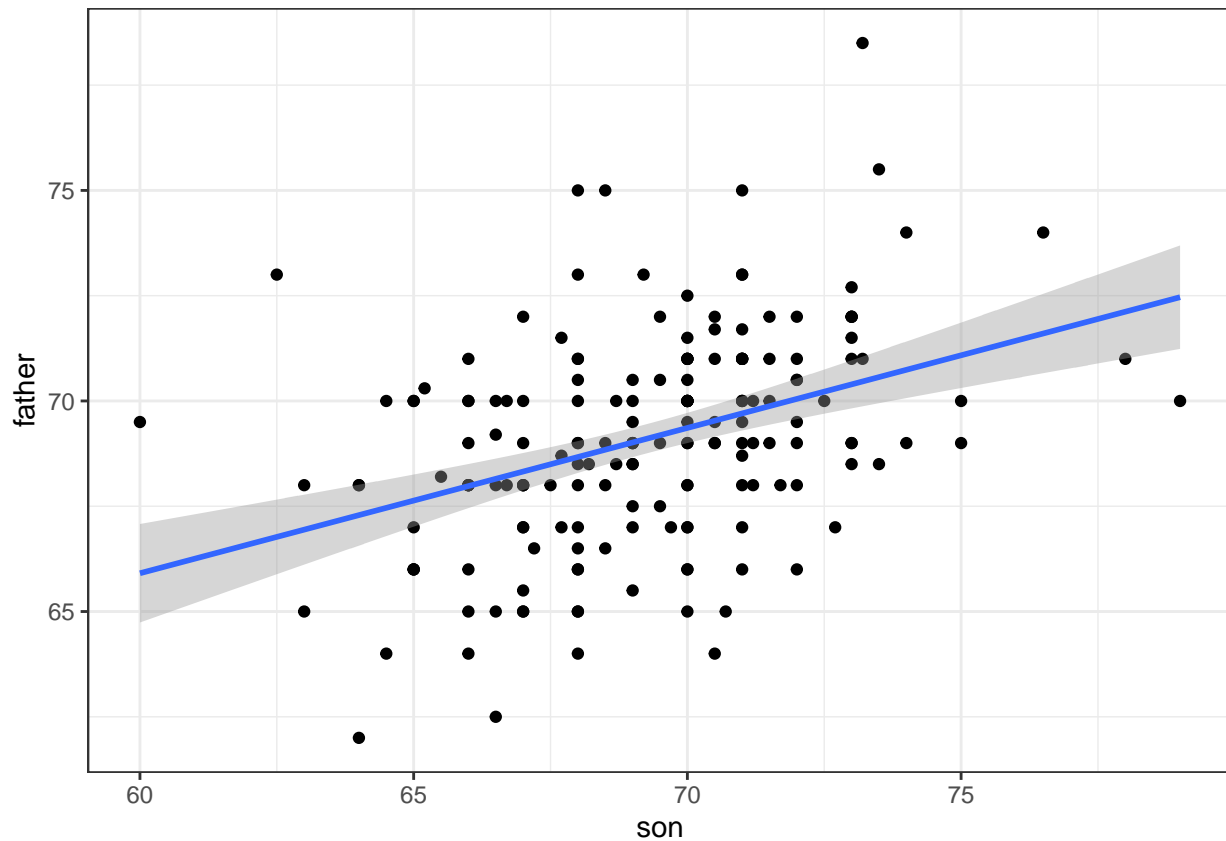
Key points

- The predicted value is often denoted as \hat{Y} , which is a random variable. Mathematical theory tells us what the standard error of the predicted value is.
- The `predict()` function in R can give us predictions directly.

Code

```
# plot predictions and confidence intervals
galton_heights %>% ggplot(aes(son, father)) +
  geom_point() +
  geom_smooth(method = "lm")
```

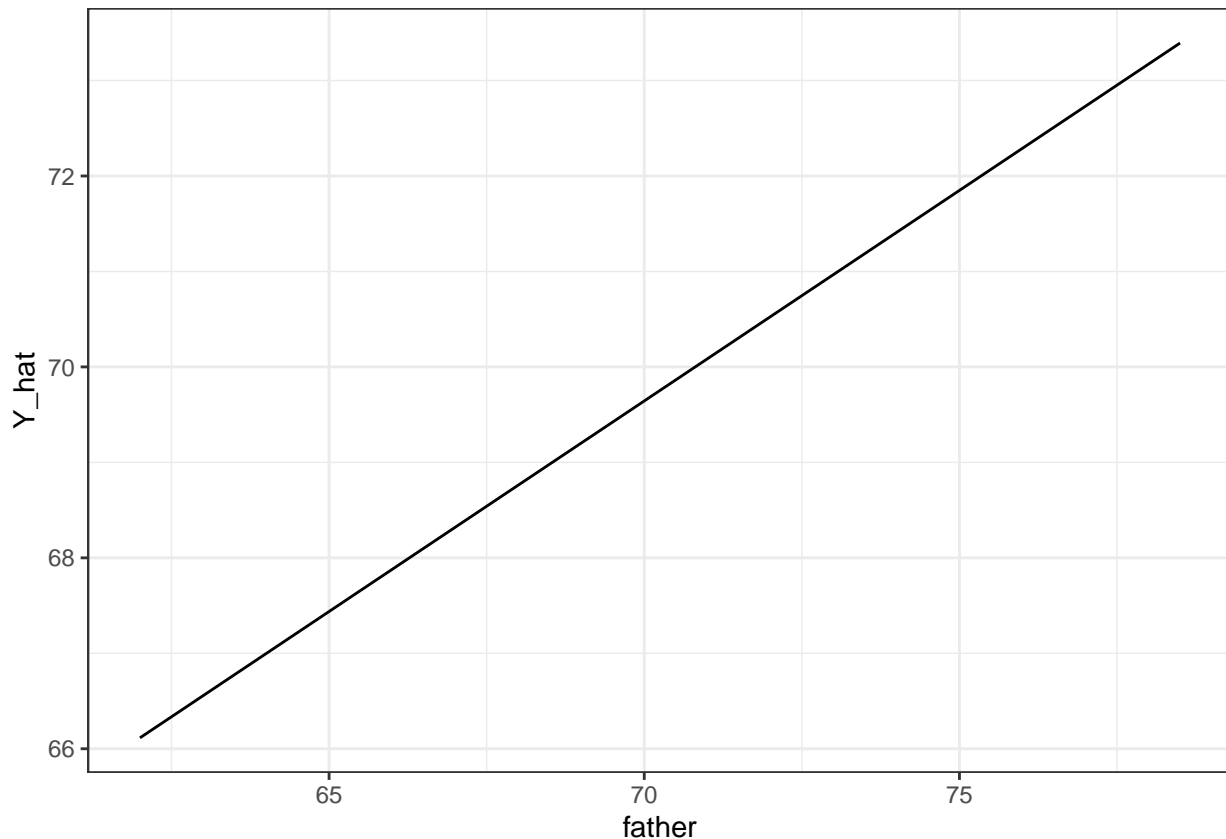
```
## `geom_smooth()` using formula 'y ~ x'
```



```
# predict Y directly
fit <- galton_heights %>% lm(son ~ father, data = .)
Y_hat <- predict(fit, se.fit = TRUE)
names(Y_hat)
```

```
## [1] "fit"          "se.fit"       "df"           "residual.scale"
```

```
# plot best fit line
galton_heights %>%
  mutate(Y_hat = predict(lm(son ~ father, data=))) %>%
  ggplot(aes(father, Y_hat))+
  geom_line()
```



Assessment - Least Squares Estimates (LSE), part 1

1. The following code was used in the video to plot RSS with $\beta_0 = 25$.

```
beta1 = seq(0, 1, len=nrow(galton_heights))
results <- data.frame(beta1 = beta1,
                      rss = sapply(beta1, rss, beta0 = 25))
results %>% ggplot(aes(beta1, rss)) + geom_line() +
  geom_line(aes(beta1, rss), col=2)
```

In a model for sons' heights vs fathers' heights, what is the least squares estimate (LSE) for β_1 if we assume $\hat{\beta}_0$ is 36? Hint: modify the code above to do your analysis.

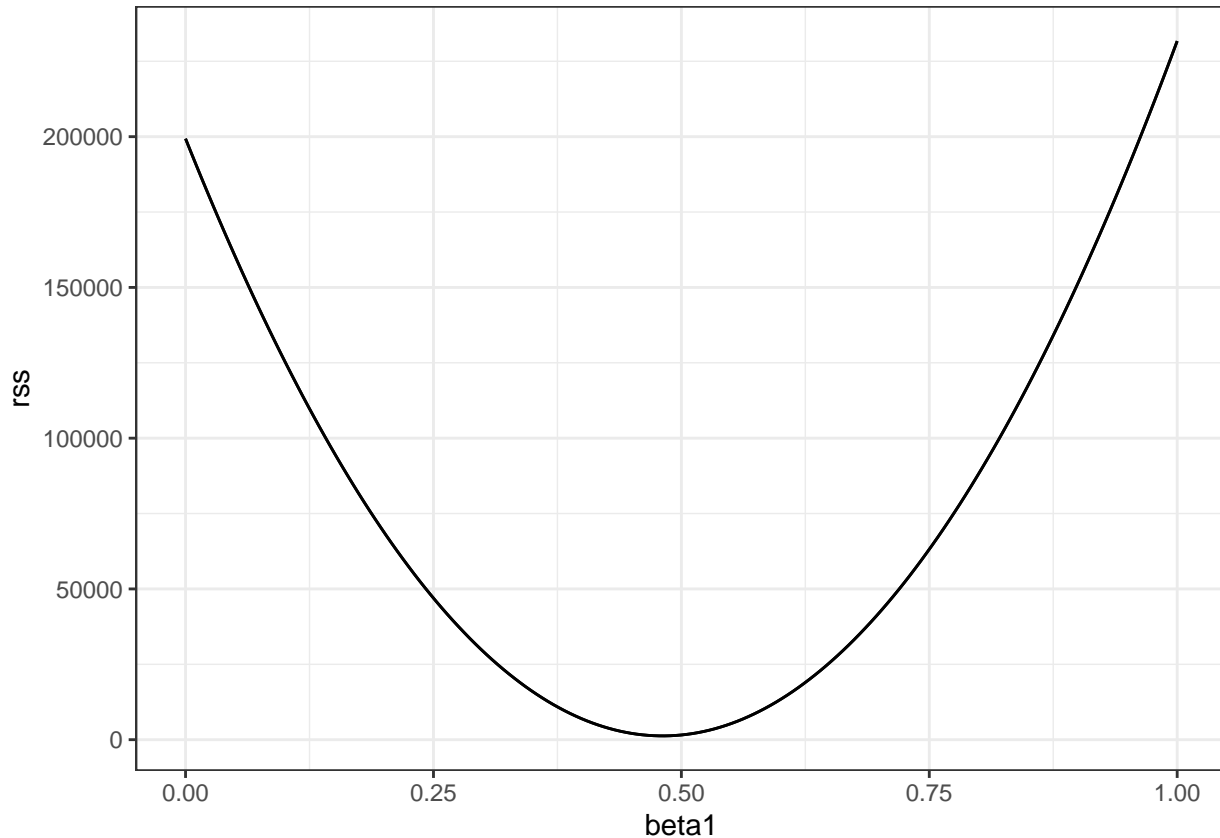
```
# compute RSS for any pair of beta0 and beta1 in Galton's data
set.seed(1983)
galton_heights <- GaltonFamilies %>%
  filter(gender == "male") %>%
  group_by(family) %>%
  sample_n(1) %>%
  ungroup() %>%
  select(father, childHeight) %>%
  rename(son = childHeight)
rss <- function(beta0, beta1, data){
  resid <- galton_heights$son - (beta0+beta1*galton_heights$father)
  return(sum(resid^2))
}

# plot RSS as a function of beta1 when beta0=36
```

```

beta1 = seq(0, 1, len=nrow(galton_heights))
results <- data.frame(beta1 = beta1,
                      rss = sapply(beta1, rss, beta0 = 36))
results %>% ggplot(aes(beta1, rss)) + geom_line() +
  geom_line(aes(beta1, rss))

```



- [] A. 0.65
- [X] B. 0.5
- [] C. 0.2
- [] D. 12

2. The least squares estimates for the parameters $\beta_0, \beta_1, \dots, \beta_n$ **minimize** the residual sum of squares.

3. Load the **Lahman** library and filter the **Teams** data frame to the years 1961-2001.

Run a linear model in R predicting the number of runs per game based on the number of bases on balls and the number of home runs. Remember to first limit your data to 1961-2001.

What is the coefficient for bases on balls?

```

fit <- Teams %>% filter(yearID %in% 1961:2001) %>%
mutate(R_per_game = R / G, BB_per_game = BB / G, HR_per_game = HR / G) %>%
lm(R_per_game ~ BB_per_game + HR_per_game, data = .)

summary(fit)

```

```

##
## Call:
## lm(formula = R_per_game ~ BB_per_game + HR_per_game, data = .)
##

```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.87325 -0.24507 -0.01449  0.23866  1.24218
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.74430    0.08236   21.18  <2e-16 ***
## BB_per_game   0.38742    0.02701   14.34  <2e-16 ***
## HR_per_game   1.56117    0.04896   31.89  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3484 on 1023 degrees of freedom
## Multiple R-squared:  0.6503, Adjusted R-squared:  0.6496
## F-statistic: 951.2 on 2 and 1023 DF,  p-value: < 2.2e-16
```

- [X] A. 0.39
- [] B. 1.56
- [] C. 1.74
- [] D. 0.027

The coefficient for bases on balls is 0.39; the coefficient for home runs is 1.56; the intercept is 1.74; the standard error for the BB coefficient is 0.027.

4. We run a Monte Carlo simulation where we repeatedly take samples of $N = 100$ from the Galton heights data and compute the regression slope coefficients for each sample:

```
B <- 1000
N <- 100
lse <- replicate(B, {
  sample_n(galton_heights, N, replace = TRUE) %>%
  lm(son ~ father, data = .) %>% .$coef
})

lse <- data.frame(beta_0 = lse[1,], beta_1 = lse[2,])
```

What does the central limit theorem tell us about the variables `beta_0` and `beta_1`?

- [X] A. They are approximately normally distributed.
- [X] B. The expected value of each is the true value of β_0 and β_1 (assuming the Galton heights data is a complete population).
- [] C. The central limit theorem does not apply in this situation.
- [] D. It allows us to test the hypothesis that $\beta_0 = 0$ and $\beta_0 = 1$

5. In an earlier video, we ran the following linear model and looked at a summary of the results.

```
summary(mod)

Call:
lm(formula = son ~ father, data = galton_heights)

Residuals:
    Min       1Q   Median       3Q      Max
-5.902  -1.405   0.092   1.342   8.092

Coefficients:
```


	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	35.7125	4.5174	7.91	2.8e-13 ***
father	0.5028	0.0653	7.70	9.5e-13 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
 β_0

What null hypothesis is the second p-value (the one in the father row) testing?

- ☐ A. $\beta_1 = 1$, where β_1 is the coefficient for the variable “father.”
- ☐ B. $\beta_1 = 0.503$, where β_1 is the coefficient for the variable “father.”
- ☒ C. $\beta_1 = 0$, where β_1 is the coefficient for the variable “father.”

6. Which R code(s) below would properly plot the predictions and confidence intervals for our linear model of sons' heights?

Select ALL that apply.

- ☐ A.

```
galton_heights %>% ggplot(aes(father, son)) +
  geom_point() +
  geom_smooth()
```

- ☒ B.

```
galton_heights %>% ggplot(aes(father, son)) +
  geom_point() +
  geom_smooth(method = "lm")
```

- ☒ C.

```
model <- lm(son ~ father, data = galton_heights)
predictions <- predict(model, interval = c("confidence"), level = 0.95)
data <- as.tibble(predictions) %>% bind_cols(father = galton_heights$father)

ggplot(data, aes(x = father, y = fit)) +
  geom_line(color = "blue", size = 1) +
  geom_ribbon(aes(ymin=lwr, ymax=upr), alpha=0.2) +
  geom_point(data = galton_heights, aes(x = father, y = son))
```

- ☐ D.

```
model <- lm(son ~ father, data = galton_heights)
predictions <- predict(model)
data <- as.tibble(predictions) %>% bind_cols(father = galton_heights$father)

ggplot(data, aes(x = father, y = fit)) +
  geom_line(color = "blue", size = 1) +
  geom_point(data = galton_heights, aes(x = father, y = son))
```

Assessment - Least Squares Estimates, part 2

In Questions 7 and 8, you'll look again at female heights from GaltonFamilies.

Define `female_heights`, a set of mother and daughter heights sampled from `GaltonFamilies`, as follows:

```
set.seed(1989, sample.kind="Rounding") #if you are using R 3.6 or later
```

```
## Warning in set.seed(1989, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
options(digits = 3)      # report 3 significant digits
```

```
female_heights <- GaltonFamilies %>%
  filter(gender == "female") %>%
  group_by(family) %>%
  sample_n(1) %>%
  ungroup() %>%
  select(mother, childHeight) %>%
  rename(daughter = childHeight)
```

7. Fit a linear regression model predicting the mothers' heights using daughters' heights.

What is the slope of the model?

```
fit <- lm(mother ~ daughter, data = female_heights)
fit$coef[2]
```

```
## daughter
##      0.31
```

What the intercept of the model?

```
fit$coef[1]
```

```
## (Intercept)
##      44.2
```

8. Predict mothers' heights using the model.

What is the predicted height of the first mother in the dataset?

```
predict(fit)[1]
```

```
##      1
## 65.6
```

What is the actual height of the first mother in the dataset?

```
female_heights$mother[1]
```

```
## [1] 67
```

We have shown how BB and singles have similar predictive power for scoring runs. Another way to compare the usefulness of these baseball metrics is by assessing how stable they are across the years. Because we have to pick players based on their previous performances, we will prefer metrics that are more stable. In these exercises, we will compare the stability of singles and BBs.

Before we get started, we want to generate two tables: one for 2002 and another for the average of 1999-2001 seasons. We want to define per plate appearance statistics, keeping only players with more than 100 plate appearances. Here is how we create the 2002 table:

```
bat_02 <- Batting %>% filter(yearID == 2002) %>%
  mutate(pa = AB + BB, singles = (H - X2B - X3B - HR)/pa, bb = BB/pa) %>%
  filter(pa >= 100) %>%
  select(playerID, singles, bb)
```

9. Now compute a similar table but with rates computed over 1999-2001. Keep only rows from 1999-2001 where players have 100 or more plate appearances, calculate each player's single rate and BB rate

per season, then calculate the average single rate (`mean_singles`) and average BB rate (`mean_bb`) per player over those three seasons.

How many players had a single rate `mean_singles` of greater than 0.2 per plate appearance over 1999-2001?

```
bat_99_01 <- Batting %>% filter(yearID %in% 1999:2001) %>%  
  mutate(pa = AB + BB, singles = (H - X2B - X3B - HR)/pa, bb = BB/pa) %>%  
  filter(pa >= 100) %>%  
  group_by(playerID) %>%  
  summarize(mean_singles = mean(singles), mean_bb = mean(bb))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
sum(bat_99_01$mean_singles > 0.2)
```

```
## [1] 46
```

How many players had a BB rate `mean_bb` of greater than 0.2 per plate appearance over 1999-2001?

```
sum(bat_99_01$mean_bb > 0.2)
```

```
## [1] 3
```

10. Use `inner_join()` to combine the `bat_02` table with the table of 1999-2001 rate averages you created in the previous question.

What is the correlation between 2002 singles rates and 1999-2001 average singles rates?

```
dat <- inner_join(bat_02, bat_99_01)
```

```
## Joining, by = "playerID"
```

```
cor(dat$singles, dat$mean_singles)
```

```
## [1] 0.551
```

What is the correlation between 2002 BB rates and 1999-2001 average BB rates?

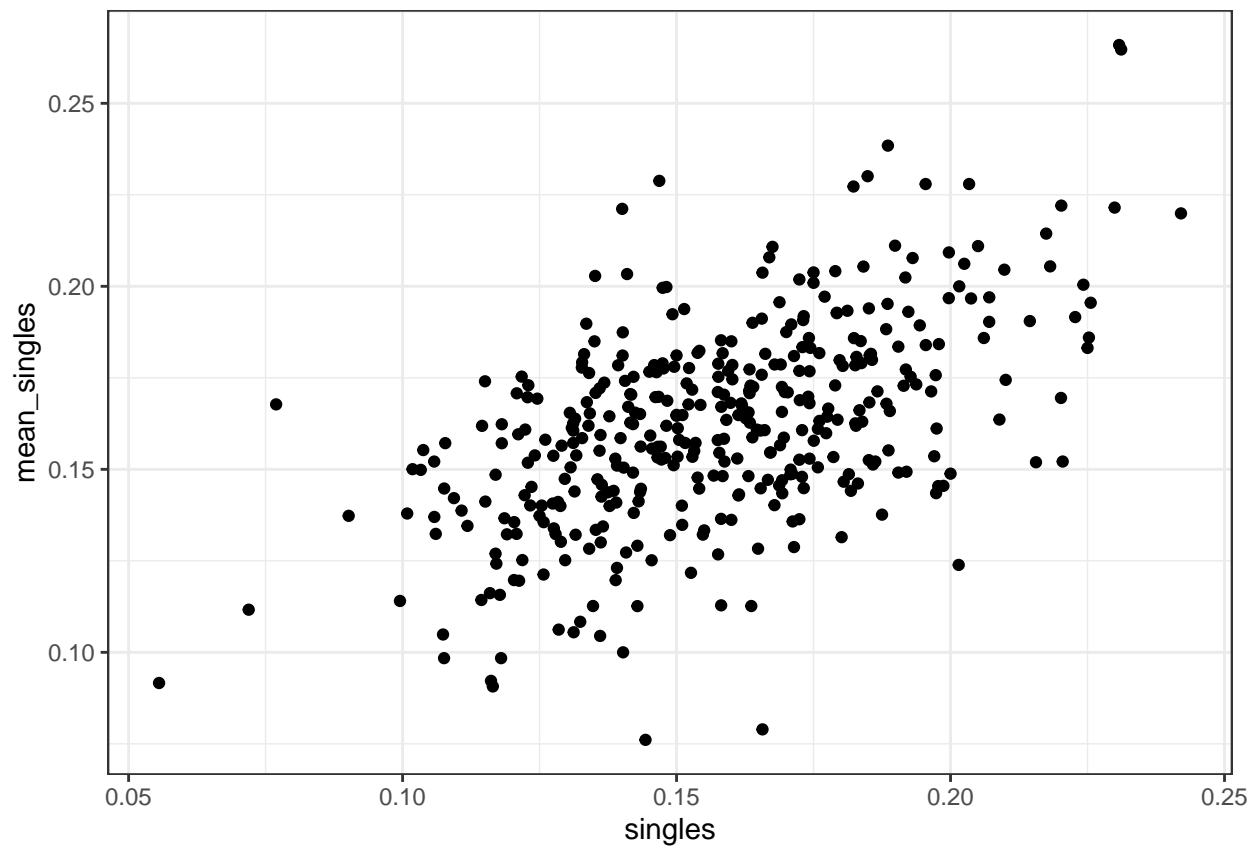
```
cor(dat$bb, dat$mean_bb)
```

```
## [1] 0.717
```

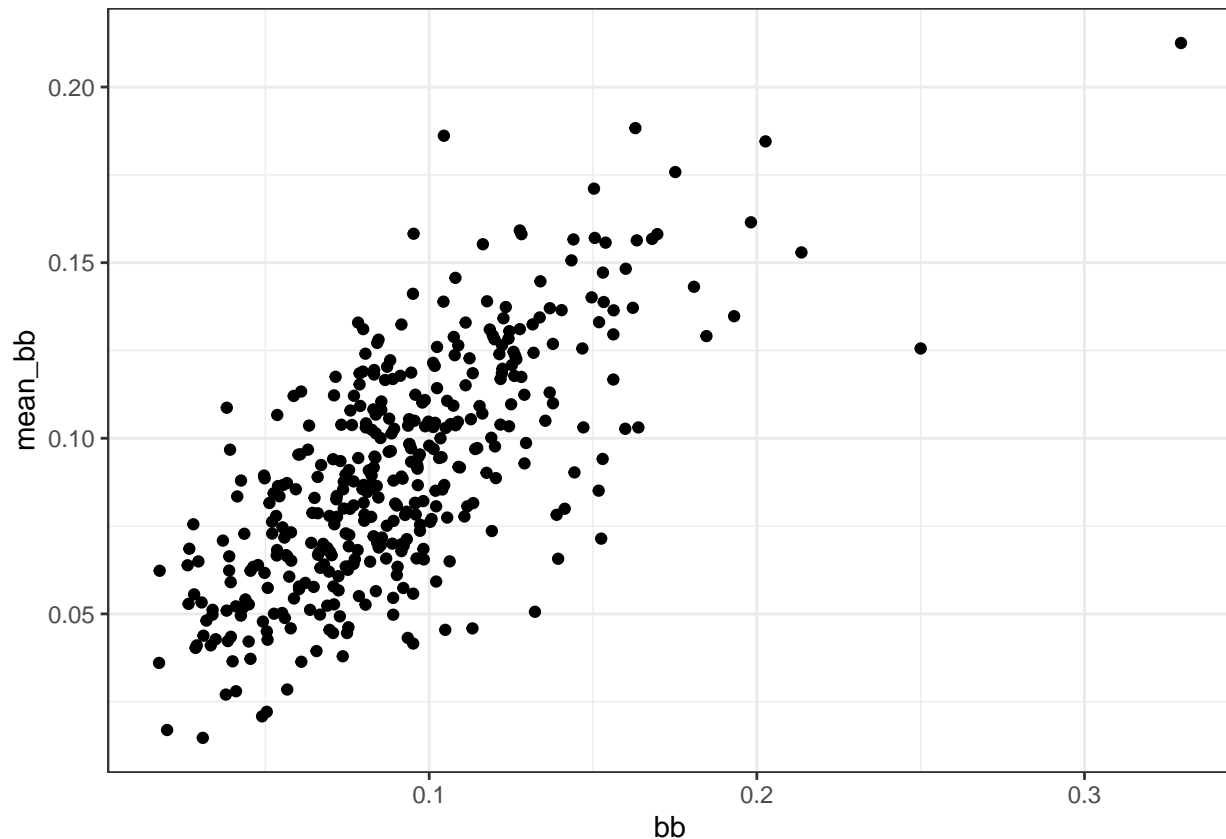
11. Make scatterplots of `mean_singles` versus `singles` and `mean_bb` versus `bb`.

Are either of these distributions bivariate normal?

```
dat %>%  
  ggplot(aes(singles, mean_singles)) +  
  geom_point()
```



```
dat %>%  
  ggplot(aes(bb, mean_bb)) +  
  geom_point()
```



- [] A. Neither distribution is bivariate normal.
- [] B. `singles` and `mean_singles` are bivariate normal, but `bb` and `mean_bb` are not.
- [] C. `bb` and `mean_bb` are bivariate normal, but `singles` and `mean_singles` are not.
- [X] D. Both distributions are bivariate normal.

12. Fit a linear model to predict 2002 `singles` given 1999-2001 `mean_singles`.

What is the coefficient of `mean_singles`, the slope of the fit?

```
fit_singles <- lm(singles ~ mean_singles, data = dat)
fit_singles$coef[2]
```

```
## mean_singles
##      0.588
```

Fit a linear model to predict 2002 `bb` given 1999-2001 `mean_bb`.

What is the coefficient of `mean_bb`, the slope of the fit?

```
fit_bb <- lm(bb ~ mean_bb, data = dat)
fit_bb$coef[2]
```

```
## mean_bb
##      0.829
```

Advanced dplyr: Tibbles

The textbook for this section is available [here](#)

Key points

- Tibbles can be regarded as a modern version of data frames and are the default data structure in the tidyverse.
- Some functions that do not work properly with data frames do work with tibbles.

Code

```
# stratify by HR
dat <- Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(HR = round(HR/G, 1),
         BB = BB/G,
         R = R/G) %>%
  select(HR, BB, R) %>%
  filter(HR >= 0.4 & HR<=1.2)

# calculate slope of regression lines to predict runs by BB in different HR strata
dat %>%
  group_by(HR) %>%
  summarize(slope = cor(BB,R)*sd(R)/sd(BB))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 9 x 2
##   HR slope
##   <dbl> <dbl>
## 1  0.4 0.734
## 2  0.5 0.566
## 3  0.6 0.412
## 4  0.7 0.285
## 5  0.8 0.365
## 6  0.9 0.261
## 7  1   0.511
## 8  1.1 0.454
## 9  1.2 0.440
```

```
# use lm to get estimated slopes - lm does not work with grouped tibbles
dat %>%
  group_by(HR) %>%
  lm(R ~ BB, data = .) %>%
  .$coef
```

```
## (Intercept)      BB
##      2.198      0.638
```

```
# inspect a grouped tibble
dat %>% group_by(HR) %>% head()
```

```
## # A tibble: 6 x 3
## # Groups:   HR [5]
##   HR    BB    R
##   <dbl> <dbl> <dbl>
## 1  0.9  3.56  4.24
## 2  0.7  3.97  4.47
## 3  0.8  3.37  4.69
## 4  1.1  3.46  4.42
## 5  1    2.75  4.61
## 6  0.9  3.06  4.58
```

```
dat %>% group_by(HR) %>% class()
```

```
## [1] "grouped_df" "tbl_df"      "tbl"        "data.frame"
```

Tibbles: Differences from Data Frames

Key points

- Tibbles are more readable than data frames.
- If you subset a data frame, you may not get a data frame. If you subset a tibble, you always get a tibble.
- Tibbles can hold more complex objects such as lists or functions.
- Tibbles can be grouped.

Code

```
# inspect data frame and tibble
head(Teams)
```

```
##   yearID lgID teamID franchID divID Rank  G  Ghome  W  L DivWin WCWin LgWin
## 1  1871   NA   BS1      BNA  <NA>    3 31    NA 20 10  <NA>  <NA>    N
## 2  1871   NA   CH1      CNA  <NA>    2 28    NA 19  9  <NA>  <NA>    N
## 3  1871   NA   CL1      CFC  <NA>    8 29    NA 10 19  <NA>  <NA>    N
## 4  1871   NA   FW1      KEK  <NA>    7 19    NA  7 12  <NA>  <NA>    N
## 5  1871   NA   NY2      NNA  <NA>    5 33    NA 16 17  <NA>  <NA>    N
## 6  1871   NA   PH1      PNA  <NA>    1 28    NA 21  7  <NA>  <NA>    Y
##   WSwIn  R  AB  H X2B X3B HR BB SO SB CS HBP SF  RA  ER  ERA CG SHO SV
## 1  <NA> 401 1372 426  70  37  3 60 19 73 16  NA NA 303 109 3.55 22  1  3
## 2  <NA> 302 1196 323  52  21 10 60 22 69 21  NA NA 241  77 2.76 25  0  1
## 3  <NA> 249 1186 328  35  40  7 26 25 18  8  NA NA 341 116 4.11 23  0  0
## 4  <NA> 137  746 178  19  8  2 33  9 16  4  NA NA 243  97 5.17 19  1  0
## 5  <NA> 302 1404 403  43  21  1 33 15 46 15  NA NA 313 121 3.72 32  1  0
## 6  <NA> 376 1281 410  66  27  9 46 23 56 12  NA NA 266 137 4.95 27  0  0
##   IPouts  HA  HRA  BBA  SOA  E DP  FP  name
## 1    828 367   2  42  23 243 24 0.834  Boston Red Stockings
## 2    753 308   6  28  22 229 16 0.829  Chicago White Stockings
## 3    762 346  13  53  34 234 15 0.818  Cleveland Forest Citys
## 4    507 261   5  21  17 163  8 0.803  Fort Wayne Kekiongas
## 5    879 373   7  42  22 235 14 0.840  New York Mutuals
## 6    747 329   3  53  16 194 13 0.845  Philadelphia Athletics
##   park attendance BPF PPF teamIDBR teamIDlahman45
## 1      South End Grounds I      NA 103  98      BOS      BS1
## 2      Union Base-Ball Grounds      NA 104 102      CHI      CH1
## 3 National Association Grounds      NA  96 100      CLE      CL1
## 4      Hamilton Field      NA 101 107      KEK      FW1
## 5      Union Grounds (Brooklyn)      NA  90  88      NYU      NY2
## 6      Jefferson Street Grounds      NA 102  98      ATH      PH1
##   teamIDretro
## 1      BS1
## 2      CH1
## 3      CL1
## 4      FW1
## 5      NY2
## 6      PH1
```

```
as_tibble(Teams)
```

```
## # A tibble: 2,925 x 48
##   yearID lgID teamID franchID divID Rank   G Ghome   W   L DivWin WCWin
##   <int> <fct> <fct>   <fct>   <chr> <int> <int> <int> <int> <chr> <chr>
## 1  1871 NA   BS1     BNA     <NA>   3    31   NA   20   10 <NA> <NA>
## 2  1871 NA   CH1     CNA     <NA>   2    28   NA   19    9 <NA> <NA>
## 3  1871 NA   CL1     CFC     <NA>   8    29   NA   10   19 <NA> <NA>
## 4  1871 NA   FW1     KEK     <NA>   7    19   NA    7   12 <NA> <NA>
## 5  1871 NA   NY2     NNA     <NA>   5    33   NA   16   17 <NA> <NA>
## 6  1871 NA   PH1     PNA     <NA>   1    28   NA   21    7 <NA> <NA>
## 7  1871 NA   RC1     ROK     <NA>   9    25   NA    4   21 <NA> <NA>
## 8  1871 NA   TRO     TRO     <NA>   6    29   NA   13   15 <NA> <NA>
## 9  1871 NA   WS3     OLY     <NA>   4    32   NA   15   15 <NA> <NA>
## 10 1872 NA   BL1     BLC     <NA>   2    58   NA   35   19 <NA> <NA>
## # ... with 2,915 more rows, and 36 more variables: LgWin <chr>, WSWin <chr>,
## #   R <int>, AB <int>, H <int>, X2B <int>, X3B <int>, HR <int>, BB <int>,
## #   SO <int>, SB <int>, CS <int>, HBP <int>, SF <int>, RA <int>, ER <int>,
## #   ERA <dbl>, CG <int>, SHO <int>, SV <int>, IPouts <int>, HA <int>,
## #   HRA <int>, BBA <int>, SOA <int>, E <int>, DP <int>, FP <dbl>, name <chr>,
## #   park <chr>, attendance <int>, BPF <int>, PPF <int>, teamIDBR <chr>,
## #   teamIDlahman45 <chr>, teamIDretro <chr>
```

```
# Note that the function was formerly called as.tibble()
```

```
# subsetting a data frame sometimes generates vectors
```

```
class(Teams[,20])
```

```
## [1] "integer"
```

```
# subsetting a tibble always generates tibbles
```

```
class(as_tibble(Teams[,20]))
```

```
## [1] "tbl_df"      "tbl"        "data.frame"
```

```
# pulling a vector out of a tibble
```

```
class(as_tibble(Teams)$HR)
```

```
## [1] "integer"
```

```
# access a non-existing column in a data frame or a tibble
```

```
Teams$hr
```

```
## NULL
```

```
as_tibble(Teams)$HR
```

```
##   [1]  3 10  7  2  1  9  3  6  6 14  0  1  7  0  2  4  4  5
##  [19]  0  0  9  0  6 13  0  5  4  8  2  1  1 17  4  2  7  6
##  [37]  2  2 15  0  2  0  2  7  7  5  0  0  0  0  9  8  4  2
##  [55]  6  2  7  2  4  0  6  4  9  1  2  3  5  3  2  8  2 20
##  [73]  3  4  8 12  5  4  3 20  4  7  7  8  5  8 12  5 12  7
##  [91] 17 11  5  7 18  4 15 15 20  5 19  9  5 11 18 11 12 16
## [109]  8  5 34 13  8 15 34 13 14 24  6 20  3 21 13  7  2 39
## [127] 32 17 16 36 19 142 10 16 40 36 26 31 20  6 17  0 22 22
## [145] 26 14  7 21  2  7 11 32  0  8  2  6  4 23 17 14 22 54
## [163] 26 25 19 16 21 30 20  6  5 17  8  8 16 24 53 45 53 19
## [181] 20 21 18 21 26 16 20 30 23 31 25 53 80 14 37 55 33 27
```



```

## [199] 48 21 29 47 20 39 47 19 25 56 77 12 32 51 34 19 14 55
## [217] 31 16 14 36 30 20 47 42 79 25 36 52 62 18 22 52 43 44
## [235] 42 58 25 20 2 13 43 34 31 58 67 31 27 21 16 27 15 25
## [253] 66 24 23 49 20 35 31 48 14 24 30 23 52 53 60 40 22 20
## [271] 28 17 13 46 55 21 29 57 19 30 30 34 26 44 26 18 39 50
## [289] 38 45 37 27 45 65 32 29 32 19 61 80 37 10 23 33 42 103
## [307] 65 61 37 42 45 40 48 54 59 25 41 54 55 36 29 34 32 61
## [325] 27 39 55 23 28 36 34 20 28 37 40 49 27 37 45 19 24 45
## [343] 38 22 16 40 31 40 25 32 36 12 17 53 18 19 18 32 34 33
## [361] 14 13 36 17 27 39 27 13 12 40 23 31 29 47 47 26 48 33
## [379] 33 23 29 26 36 24 37 32 28 32 18 38 12 29 26 19 35 24
## [397] 29 39 33 33 42 19 14 14 6 18 33 22 6 38 5 18 29 10
## [415] 47 48 15 25 14 9 28 31 12 20 18 32 12 34 12 8 17 26
## [433] 15 24 14 22 21 27 11 31 27 31 23 15 10 24 10 29 29 17
## [451] 11 12 27 18 13 39 23 24 16 22 16 20 22 13 25 16 7 20
## [469] 16 12 10 14 17 32 12 12 20 9 26 18 18 22 5 13 15 11
## [487] 11 23 15 22 12 19 10 18 12 14 28 17 3 19 14 18 19 20
## [505] 13 21 11 25 20 17 8 20 16 14 4 20 22 10 19 26 16 21
## [523] 12 25 10 15 9 43 25 31 7 34 23 9 28 31 20 19 22 33
## [541] 12 15 9 35 28 37 20 54 21 20 30 41 25 35 60 49 17 26
## [559] 16 29 32 35 17 43 21 12 19 47 18 22 43 39 19 27 20 17
## [577] 39 32 24 59 27 16 24 30 8 33 73 35 18 15 19 32 18 42
## [595] 31 35 38 19 52 42 16 10 25 33 39 30 12 29 62 18 34 17
## [613] 26 33 18 36 14 36 14 17 40 25 50 53 15 20 23 28 17 24
## [631] 31 16 58 24 20 19 23 20 12 14 28 22 17 46 14 16 17 42
## [649] 35 19 42 20 14 25 12 14 25 22 18 17 26 13 25 39 27 17
## [667] 38 9 15 26 4 15 10 13 8 21 15 9 13 13 20 22 25 15
## [685] 5 27 4 33 25 24 25 21 20 24 23 40 45 35 42 17 31 18
## [703] 24 22 28 23 37 34 18 35 30 46 115 44 64 16 50 32 36 17
## [721] 59 61 35 37 20 42 58 75 134 82 88 37 67 83 42 45 56 32
## [739] 45 42 45 32 54 80 95 111 116 52 98 107 45 34 62 32 42 90
## [757] 45 59 41 85 105 53 112 49 82 63 26 30 72 25 41 66 36 41
## [775] 35 95 98 63 94 44 67 67 22 41 64 41 38 86 44 52 50 114
## [793] 110 76 100 78 110 109 56 32 40 16 32 66 35 27 36 73 121 61
## [811] 75 44 72 90 43 28 39 37 36 74 29 26 51 109 158 56 57 54
## [829] 55 84 29 38 66 52 24 92 32 34 62 118 133 89 85 52 63 113
## [847] 40 28 99 33 37 139 34 62 110 136 142 122 153 60 46 100 48 47
## [865] 122 66 63 171 74 72 82 143 152 125 126 86 75 104 57 37 71 34
## [883] 27 84 21 71 43 101 155 118 81 41 76 60 49 53 110 63 36 69
## [901] 47 78 80 116 160 172 122 48 67 76 61 50 62 54 43 72 34 50
## [919] 57 82 144 139 60 39 64 57 60 51 79 83 71 101 55 100 74 126
## [937] 135 144 56 52 62 104 51 69 59 75 74 88 73 93 106 123 104 112
## [955] 92 66 73 86 32 86 33 67 60 76 82 123 94 97 182 72 103 60
## [973] 79 88 62 100 37 63 67 96 73 103 150 111 174 94 103 47 71 94
## [991] 47 98 61 54 67 65 110 113 137 125 174 98 40 65 92 91 85 124
## [1009] 78 56 64 91 98 85 124 116 166 98 49 63 91 98 44 145 93 59
## [1027] 73 86 89 101 134 91 155 105 75 76 118 119 52 124 101 48 47 99
## [1045] 64 103 81 95 151 85 64 56 91 70 52 103 62 68 25 75 66 50
## [1063] 76 109 108 33 44 54 98 60 40 57 39 39 33 52 43 55 77 81
## [1081] 100 26 66 42 78 70 47 69 56 79 23 71 51 70 60 93 96 36
## [1099] 55 70 72 100 33 50 57 101 22 57 56 65 77 114 93 33 56 72
## [1117] 63 64 27 109 55 44 37 56 65 79 108 121 136 40 80 60 84 81
## [1135] 60 103 83 85 53 71 95 112 103 221 115 61 60 156 90 115 42 121
## [1153] 91 95 55 87 104 155 78 164 139 68 91 108 63 105 31 131 152 103

```

```

## [1171] 43 97 86 112 88 147 115 82 122 126 117 102 81 161 194 148 93 161
## [1189] 99 164 114 133 159 100 125 138 106 102 76 127 184 130 86 103 88 140
## [1207] 104 179 140 102 108 137 86 95 54 113 153 110 80 107 104 148 103 151
## [1225] 129 89 93 92 82 97 50 101 208 74 137 166 160 108 156 176 139 116
## [1243] 115 99 112 140 69 52 123 186 94 159 147 156 90 139 186 133 94 102
## [1261] 76 119 81 54 137 201 116 164 181 148 130 121 182 169 175 132 91 143
## [1279] 80 91 139 179 128 142 221 153 150 112 177 145 190 121 110 124 112 87
## [1297] 153 147 106 147 187 140 116 166 199 157 145 117 92 132 111 108 155 101
## [1315] 182 123 161 109 138 172 167 164 124 134 170 111 121 109 125 97 163 161
## [1333] 167 160 117 148 177 153 113 112 167 118 163 123 124 112 119 140 127 150
## [1351] 110 126 170 193 99 120 130 138 147 149 112 138 176 158 150 180 90 189
## [1369] 157 167 188 240 103 128 183 103 119 156 146 92 126 167 180 209 105 116
## [1387] 137 140 185 181 199 139 142 108 204 137 132 146 171 114 127 122 169 148
## [1405] 62 95 95 110 225 139 188 96 126 108 197 128 138 162 186 106 145 130
## [1423] 164 157 70 166 102 79 221 159 162 103 130 121 165 109 125 125 165 92
## [1441] 125 134 183 156 162 97 110 78 150 196 149 107 144 111 159 109 136 207
## [1459] 175 145 122 87 140 149 155 179 112 70 108 144 162 98 117 158 181 108
## [1477] 126 158 138 158 114 89 128 109 131 152 93 69 82 131 100 83 103 91
## [1495] 140 115 115 80 133 125 83 71 130 106 75 185 66 67 105 109 81 94
## [1513] 100 80 108 73 124 141 175 197 88 112 142 171 119 182 104 98 97 163
## [1531] 125 94 109 148 137 119 99 125 136 90 148 160 179 203 114 123 179 191
## [1549] 183 148 129 97 87 153 126 136 111 120 171 101 130 172 165 113 138 153
## [1567] 158 161 96 138 128 138 109 179 71 80 95 116 104 88 97 98 160 123
## [1585] 154 96 140 95 86 144 100 124 78 108 133 124 91 122 134 78 98 93
## [1603] 88 91 103 105 134 98 110 102 150 70 56 206 119 147 93 111 117 137
## [1621] 158 157 134 114 110 120 145 125 131 85 147 134 154 112 161 75 110 120
## [1639] 116 109 95 135 110 135 131 131 110 89 139 111 120 86 101 96 132 95
## [1657] 114 99 93 83 99 107 124 134 55 94 95 124 153 125 84 118 118 121
## [1675] 146 98 110 101 151 125 138 78 84 81 134 82 119 134 63 73 105 141
## [1693] 85 101 66 65 91 81 88 94 120 102 113 110 110 64 85 63 80 139
## [1711] 148 213 131 192 111 181 100 166 114 146 191 123 125 138 184 88 117 186
## [1729] 133 120 133 134 96 135 100 123 154 172 108 106 72 136 106 129 70 98
## [1747] 149 82 173 121 125 86 100 133 115 75 97 117 79 132 98 126 181 194
## [1765] 164 127 135 132 138 164 49 116 183 112 185 143 150 74 108 119 148 93
## [1783] 132 125 100 140 95 144 156 162 106 91 107 113 89 143 75 115 148 99
## [1801] 203 114 189 61 137 117 116 67 104 80 101 124 126 64 88 90 97 76
## [1819] 57 64 39 65 45 61 82 47 96 81 100 57 104 69 55 32 89 63
## [1837] 50 49 61 146 179 136 186 136 102 82 109 177 74 132 138 148 216 133
## [1855] 161 97 149 112 134 81 130 133 67 115 106 130 168 142 154 157 140 107
## [1873] 86 156 97 109 146 141 132 102 153 112 121 125 121 93 111 142 83 106
## [1891] 167 111 160 181 150 172 136 106 123 187 79 117 102 114 96 96 130 107
## [1909] 158 147 98 109 129 112 75 120 143 126 214 162 153 146 150 114 116 202
## [1927] 121 154 129 141 101 118 176 134 155 141 80 109 171 115 87 129 158 138
## [1945] 169 144 167 121 155 144 157 198 125 137 130 196 127 110 188 148 163 154
## [1963] 111 136 158 114 58 184 181 152 211 174 172 173 209 192 187 225 122 168
## [1981] 125 196 163 120 196 192 199 169 131 113 161 205 94 194 215 96 137 124
## [1999] 124 132 113 122 134 143 96 121 99 151 113 107 148 152 156 106 110 94
## [2017] 148 113 71 112 158 128 129 108 145 94 124 128 127 116 97 101 89 117
## [2035] 126 100 130 147 127 123 95 120 134 141 73 122 142 162 132 106 147 106
## [2053] 136 125 110 172 94 100 129 100 128 114 147 172 164 103 138 123 107 152
## [2071] 73 110 167 141 170 126 115 139 159 164 79 209 79 117 108 140 116 95
## [2089] 147 117 159 111 126 121 126 141 68 177 133 138 148 84 88 110 104 99
## [2107] 127 182 96 75 72 104 82 102 163 93 142 118 106 135 149 105 94 159
## [2125] 163 169 157 114 114 162 161 137 141 142 178 94 138 125 130 121 125 122

```

```
## [2143] 178 158 158 156 110 153 161 168 118 181 159 137 139 120 120 121 109 124
## [2161] 167 125 161 94 120 100 115 103 99 108 139 117 113 80 80 92 153 123
## [2179] 108 124 115 168 173 175 186 146 158 161 207 200 159 144 109 119 140 120
## [2197] 128 118 122 125 169 94 125 116 182 152 107 138 140 197 257 209 192 195
## [2215] 175 191 218 221 204 150 129 123 150 118 178 148 162 147 243 132 138 147
## [2233] 245 153 142 221 177 161 174 196 185 158 127 142 220 239 176 136 133 158
## [2251] 174 132 135 172 161 153 197 116 129 152 264 172 144 187 147 147 159 215
## [2269] 214 205 198 212 138 198 183 165 114 166 134 159 152 115 147 207 136 149
## [2287] 126 107 167 234 161 223 111 201 221 158 216 197 203 176 162 189 209 209
## [2305] 223 212 128 168 151 187 165 105 163 193 181 235 161 171 153 244 188 194
## [2323] 145 230 212 236 179 179 184 167 216 183 200 221 161 177 160 249 150 211
## [2341] 177 116 178 205 198 239 144 168 157 198 226 235 162 173 244 158 208 174
## [2359] 136 198 214 194 176 212 213 139 166 208 152 206 209 164 131 203 147 199
## [2377] 164 161 161 169 235 199 121 246 195 152 165 164 165 177 217 200 169 192
## [2395] 152 124 146 167 140 155 139 167 162 223 160 205 165 142 136 152 198 175
## [2413] 133 230 187 150 152 235 152 238 220 172 182 158 198 153 157 191 162 124
## [2431] 196 155 144 230 124 176 166 163 128 139 180 196 137 239 190 162 135 178
## [2449] 169 222 242 235 194 184 202 201 148 187 150 203 135 191 151 242 185 189
## [2467] 215 142 139 136 183 214 145 227 145 191 184 189 199 200 194 222 207 150
## [2485] 168 128 161 126 147 149 175 134 229 175 155 167 139 130 130 128 170 157
## [2503] 260 136 117 160 222 164 192 236 166 217 196 157 203 182 174 124 159 153
## [2521] 180 143 210 200 175 216 141 161 172 163 184 190 183 199 164 171 176 142
## [2539] 166 190 151 204 178 171 177 201 167 102 123 129 231 118 201 177 171 213
## [2557] 148 171 153 131 141 187 179 165 123 159 130 172 173 235 184 187 171 160
## [2575] 200 208 167 120 159 137 198 111 180 172 125 214 153 154 124 94 174 180
## [2593] 194 126 117 173 149 160 212 184 161 158 161 190 183 159 142 144 173 145
## [2611] 182 172 244 95 135 224 125 141 160 122 160 199 224 209 156 180 139 133
## [2629] 211 177 149 188 128 173 152 152 108 121 155 120 182 142 201 128 109 166
## [2647] 126 132 101 162 150 160 162 257 149 172 173 191 203 154 148 183 154 163
## [2665] 169 149 95 129 155 117 185 103 222 108 114 153 107 91 109 121 162 172
## [2683] 210 186 154 165 149 214 165 211 137 172 136 166 163 146 131 187 116 137
## [2701] 202 131 245 139 195 158 170 121 149 103 159 175 200 198 194 130 181 212
## [2719] 178 148 172 155 171 159 176 148 112 164 138 95 157 151 144 130 186 140
## [2737] 161 146 188 107 125 165 176 185 161 118 123 211 123 155 157 131 142 186
## [2755] 155 163 95 155 134 122 150 128 147 125 146 125 156 109 136 132 105 117
## [2773] 111 177 152 154 100 217 161 136 171 167 141 186 151 230 139 176 187 120
## [2791] 145 156 212 177 146 130 140 148 198 136 137 167 172 232 177 190 122 253
## [2809] 208 168 199 164 185 204 211 198 147 156 189 128 194 200 183 218 169 161
## [2827] 153 177 223 130 225 216 215 221 203 220 165 232 168 186 223 219 212 192
## [2845] 187 238 193 186 221 194 224 206 241 224 234 174 151 189 200 128 196 228
## [2863] 237 222 215 176 175 188 208 182 167 172 216 210 135 205 155 214 235 128
## [2881] 218 166 267 170 227 186 157 162 176 133 205 150 194 217 191 220 249 213
## [2899] 245 182 256 227 223 224 149 288 162 220 279 146 250 307 306 242 257 215
## [2917] 163 219 239 167 210 217 223 247 231
```

```
# create a tibble with complex objects
tibble(id = c(1, 2, 3), func = c(mean, median, sd))
```

```
## # A tibble: 3 x 2
##   id func
##   <dbl> <list>
## 1     1 <fn>
## 2     2 <fn>
## 3     3 <fn>
```

do

The textbook for this section is available [here](#)

Key points

- The `do()` function serves as a bridge between R functions, such as `lm()`, and the tidyverse.
- We have to specify a column when using the `do()` function, otherwise we will get an error.
- If the data frame being returned has more than one row, the rows will be concatenated appropriately.

Code

```
# use do to fit a regression line to each HR stratum
dat %>%
  group_by(HR) %>%
  do(fit = lm(R ~ BB, data = .))
```

```
## # A tibble: 9 x 2
## # Rowwise:
##       HR fit
##   <dbl> <list>
## 1  0.4 <lm>
## 2  0.5 <lm>
## 3  0.6 <lm>
## 4  0.7 <lm>
## 5  0.8 <lm>
## 6  0.9 <lm>
## 7  1    <lm>
## 8  1.1 <lm>
## 9  1.2 <lm>
```

```
# using do without a column name gives an error
dat %>%
  group_by(HR) %>%
  do(lm(R ~ BB, data = .))
```

```
# define a function to extract slope from lm
get_slope <- function(data){
  fit <- lm(R ~ BB, data = data)
  data.frame(slope = fit$coefficients[2],
             se = summary(fit)$coefficient[2,2])
}
```

```
# return the desired data frame
dat %>%
  group_by(HR) %>%
  do(get_slope(.))
```

```
## # A tibble: 9 x 3
## # Groups:   HR [9]
##       HR slope    se
##   <dbl> <dbl> <dbl>
## 1  0.4 0.734 0.208
## 2  0.5 0.566 0.110
## 3  0.6 0.412 0.0974
## 4  0.7 0.285 0.0705
## 5  0.8 0.365 0.0653
## 6  0.9 0.261 0.0754
```

```
## 7 1 0.511 0.0749
## 8 1.1 0.454 0.0855
## 9 1.2 0.440 0.0801
```

not the desired output: a column containing data frames

```
dat %>%
  group_by(HR) %>%
  do(slope = get_slope(.))
```

```
## # A tibble: 9 x 2
## # Rowwise:
##   HR slope
##   <dbl> <list>
## 1 0.4 <df[,2] [1 x 2]>
## 2 0.5 <df[,2] [1 x 2]>
## 3 0.6 <df[,2] [1 x 2]>
## 4 0.7 <df[,2] [1 x 2]>
## 5 0.8 <df[,2] [1 x 2]>
## 6 0.9 <df[,2] [1 x 2]>
## 7 1 <df[,2] [1 x 2]>
## 8 1.1 <df[,2] [1 x 2]>
## 9 1.2 <df[,2] [1 x 2]>
```

data frames with multiple rows will be concatenated appropriately

```
get_lse <- function(data){
  fit <- lm(R ~ BB, data = data)
  data.frame(term = names(fit$coefficients),
             estimate = fit$coefficients,
             se = summary(fit)$coefficient[,2])
}
```

```
dat %>%
  group_by(HR) %>%
  do(get_lse(.))
```

```
## # A tibble: 18 x 4
## # Groups:   HR [9]
##   HR term estimate se
##   <dbl> <fct>   <dbl> <dbl>
## 1 0.4 (Intercept) 1.36 0.631
## 2 0.4 BB 0.734 0.208
## 3 0.5 (Intercept) 2.01 0.344
## 4 0.5 BB 0.566 0.110
## 5 0.6 (Intercept) 2.53 0.305
## 6 0.6 BB 0.412 0.0974
## 7 0.7 (Intercept) 3.21 0.225
## 8 0.7 BB 0.285 0.0705
## 9 0.8 (Intercept) 3.07 0.213
## 10 0.8 BB 0.365 0.0653
## 11 0.9 (Intercept) 3.54 0.252
## 12 0.9 BB 0.261 0.0754
## 13 1 (Intercept) 2.88 0.255
## 14 1 BB 0.511 0.0749
## 15 1.1 (Intercept) 3.21 0.300
## 16 1.1 BB 0.454 0.0855
```

```
## 17 1.2 (Intercept) 3.40 0.291
## 18 1.2 BB          0.440 0.0801
```

broom

The textbook for this section is available [here](#)

Key points

- The **broom** package has three main functions, all of which extract information from the object returned by **lm** and return it in a **tidyverse** friendly data frame.
- The `tidy()` function returns estimates and related information as a data frame.
- The functions `glance()` and `augment()` relate to model specific and observation specific outcomes respectively.

Code

```
# use tidy to return lm estimates and related information as a data frame
if(!require(broom)) install.packages("broom")
```

```
## Loading required package: broom
```

```
library(broom)
fit <- lm(R ~ BB, data = dat)
tidy(fit)
```

```
## # A tibble: 2 x 5
##   term      estimate std.error statistic  p.value
##   <chr>      <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept) 2.20      0.113     19.4 1.12e-70
## 2 BB          0.638     0.0344     18.5 1.35e-65
```

```
# add confidence intervals with tidy
tidy(fit, conf.int = TRUE)
```

```
## # A tibble: 2 x 7
##   term      estimate std.error statistic  p.value conf.low conf.high
##   <chr>      <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept) 2.20      0.113     19.4 1.12e-70     1.98     2.42
## 2 BB          0.638     0.0344     18.5 1.35e-65     0.570     0.705
```

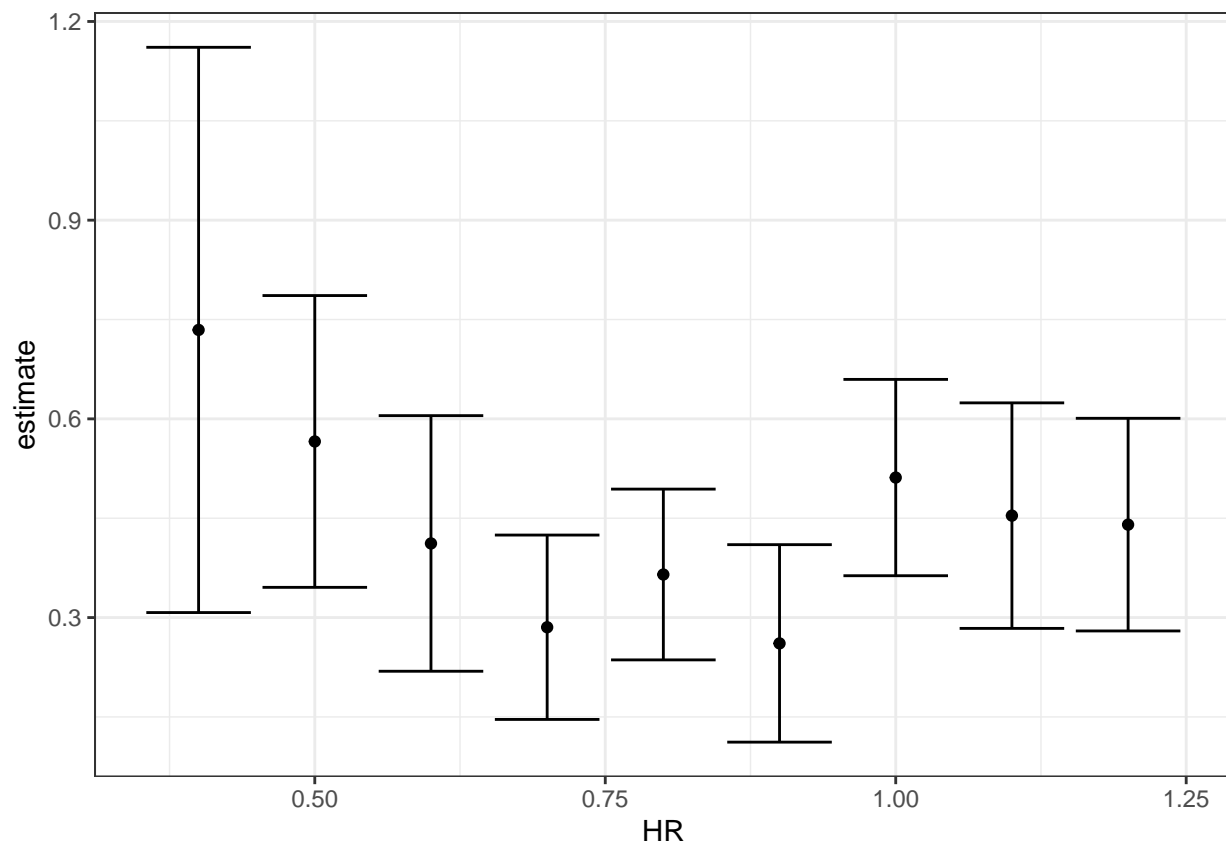
```
# pipeline with lm, do, tidy
dat %>%
  group_by(HR) %>%
  do(tidy(lm(R ~ BB, data = .), conf.int = TRUE)) %>%
  filter(term == "BB") %>%
  select(HR, estimate, conf.low, conf.high)
```

```
## # A tibble: 9 x 4
## # Groups:   HR [9]
##   HR estimate conf.low conf.high
##   <dbl>    <dbl>    <dbl>    <dbl>
## 1 0.4 0.734 0.308 1.16
## 2 0.5 0.566 0.346 0.786
## 3 0.6 0.412 0.219 0.605
## 4 0.7 0.285 0.146 0.425
## 5 0.8 0.365 0.236 0.494
## 6 0.9 0.261 0.112 0.410
## 7 1 0.511 0.363 0.660
```

```
## 8 1.1 0.454 0.284 0.624
## 9 1.2 0.440 0.280 0.601
```

```
# make ggplots
```

```
dat %>%
  group_by(HR) %>%
  do(tidy(lm(R ~ BB, data = .), conf.int = TRUE)) %>%
  filter(term == "BB") %>%
  select(HR, estimate, conf.low, conf.high) %>%
  ggplot(aes(HR, y = estimate, ymin = conf.low, ymax = conf.high)) +
  geom_errorbar() +
  geom_point()
```



```
# inspect with glance
```

```
glance(fit)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df logLik   AIC   BIC
##   <dbl>      <dbl> <dbl>      <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl>
## 1    0.266      0.265 0.454      343. 1.35e-65     1 -596. 1199. 1214.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Assessment - Tibbles, do, and broom, part 1

- As seen in the videos, what problem do we encounter when we try to run a linear model on our baseball data, grouping by home runs?
 - [] A. There is not enough data in some levels to run the model.
 - [X] B. The `lm()` function does not know how to handle grouped tibbles.

- ☐ C. The results of the `lm()` function cannot be put into a tidy format.
2. Tibbles are similar to what other class in R?
- ☐ A. Vectors
 - ☐ B. Matrices
 - ☒ C. Data frames
 - ☐ D. Lists
3. What are some advantages of tibbles compared to data frames?
- ☒ A. Tibbles display better.
 - ☒ B. If you subset a tibble, you always get back a tibble.
 - ☒ C. Tibbles can have complex entries.
 - ☒ D. Tibbles can be grouped.
4. What are two advantages of the `do` command, when applied to the tidyverse?
- ☐ A. It is faster than normal functions.
 - ☐ B. It returns useful error messages.
 - ☒ C. It understands grouped tibbles.
 - ☒ D. It always returns a `data.frame`.
5. You want to take the tibble `dat`, which we used in the video on the `do()` function, and run the linear model $R \sim BB$ for each strata of `HR`. Then you want to add three new columns to your grouped tibble: the coefficient, standard error, and p-value for the `BB` term in the model.

You've already written the function `get_slope()`, shown below.

```
get_slope <- function(data) {
  fit <- lm(R ~ BB, data = data)
  sum.fit <- summary(fit)

  data.frame(slope = sum.fit$coefficients[2, "Estimate"],
             se = sum.fit$coefficients[2, "Std. Error"],
             pvalue = sum.fit$coefficients[2, "Pr(>|t|)"])
}
```

What additional code could you write to accomplish your goal?

```
dat %>%
  group_by(HR) %>%
  do(get_slope(.))
```

```
## # A tibble: 9 x 4
## # Groups:   HR [9]
##   HR slope      se  pvalue
##   <dbl> <dbl> <dbl>   <dbl>
## 1  0.4 0.734 0.208 1.54e- 3
## 2  0.5 0.566 0.110 3.02e- 6
## 3  0.6 0.412 0.0974 4.80e- 5
## 4  0.7 0.285 0.0705 7.93e- 5
## 5  0.8 0.365 0.0653 9.13e- 8
## 6  0.9 0.261 0.0754 7.12e- 4
## 7  1    0.511 0.0749 3.00e-10
## 8  1.1 0.454 0.0855 1.03e- 6
## 9  1.2 0.440 0.0801 1.07e- 6
```

- ☐ A.


```
dat %>%
  group_by(HR) %>%
  do(get_slope)
```

- [X] B.

```
dat %>%
  group_by(HR) %>%
  do(get_slope(.))
```

- [] C.

```
dat %>%
  group_by(HR) %>%
  do(slope = get_slope(.))
```

- [] D.

```
dat %>%
  do(get_slope(.))
```

6. The output of a broom function is always what?

- [X] A. A data.frame
- [] B. A list
- [] C. A vector

7. You want to know whether the relationship between home runs and runs per game varies by baseball league.

You create the following dataset:

```
dat <- Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(HR = HR/G,
         R = R/G) %>%
  select(lgID, HR, BB, R)
```

What code would help you quickly answer this question?

```
dat %>%
  group_by(lgID) %>%
  do(tidy(lm(R ~ HR, data = .), conf.int = T)) %>%
  filter(term == "HR")
```

```
## # A tibble: 2 x 8
## # Groups:   lgID [2]
##   lgID term estimate std.error statistic p.value conf.low conf.high
##   <fct> <chr>      <dbl>      <dbl>      <dbl>   <dbl>   <dbl>      <dbl>
## 1 AL   HR          1.90     0.0734     25.9 1.29e-95    1.75     2.04
## 2 NL   HR          1.76     0.0671     26.2 1.16e-95    1.62     1.89
```

- [X] A.

```
dat %>%
  group_by(lgID) %>%
  do(tidy(lm(R ~ HR, data = .), conf.int = T)) %>%
  filter(term == "HR")
```

- [] B.

```
dat %>%
  group_by(lgID) %>%
  do(glance(lm(R ~ HR, data = .)))
```

- [] C.

```
dat %>%
  do(tidy(lm(R ~ HR, data = .), conf.int = T)) %>%
  filter(term == "HR")
```

- [] D.

```
dat %>%
  group_by(lgID) %>%
  do(mod = lm(R ~ HR, data = .))
```

Assessment - Tibbles, do, and broom, part 2

We have investigated the relationship between fathers' heights and sons' heights. But what about other parent-child relationships? Does one parent's height have a stronger association with child height? How does the child's gender affect this relationship in heights? Are any differences that we observe statistically significant?

The `galton` dataset is a sample of one male and one female child from each family in the `GaltonFamilies` dataset. The `pair` column denotes whether the pair is father and daughter, father and son, mother and daughter, or mother and son.

Create the `galton` dataset using the code below:

```
data("GaltonFamilies")
set.seed(1) # if you are using R 3.5 or earlier
set.seed(1, sample.kind = "Rounding") # if you are using R 3.6 or later
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
galton <- GaltonFamilies %>%
  group_by(family, gender) %>%
  sample_n(1) %>%
  ungroup() %>%
  gather(parent, parentHeight, father:mother) %>%
  mutate(child = ifelse(gender == "female", "daughter", "son")) %>%
  unite(pair, c("parent", "child"))
```

```
galton
```

```
## # A tibble: 710 x 8
##   family midparentHeight children childNum gender childHeight pair
##   <fct>          <dbl>      <int>   <int> <fct>          <dbl> <chr>
## 1 001             75.4         4       2 female         69.2 fath~
## 2 001             75.4         4       1 male          73.2 fath~
## 3 002             73.7         4       4 female         65.5 fath~
## 4 002             73.7         4       2 male          72.5 fath~
## 5 003             72.1         2       2 female         68   fath~
## 6 003             72.1         2       1 male          71   fath~
## 7 004             72.1         5       5 female         63   fath~
## 8 004             72.1         5       2 male          68.5 fath~
```

```
## 9 005          69.1      6      5 female      62.5 fath~
## 10 005         69.1      6      1 male       72  fath~
## # ... with 700 more rows, and 1 more variable: parentHeight <dbl>
```

8. Group by `pair` and summarize the number of observations in each group.

How many father-daughter pairs are in the dataset?

How many mother-son pairs are in the dataset?

```
galton %>%
  group_by(pair) %>%
  summarize(n = n())

## `summarise()` ungrouping output (override with `.groups` argument)
## # A tibble: 4 x 2
##   pair      n
##   <chr>  <int>
## 1 father_daughter 176
## 2 father_son      179
## 3 mother_daughter 176
## 4 mother_son      179
```

9. Calculate the correlation coefficients for fathers and daughters, fathers and sons, mothers and daughters and mothers and sons.

Which pair has the **strongest** correlation in heights?

```
galton %>%
  group_by(pair) %>%
  summarize(cor = cor(parentHeight, childHeight)) %>%
  filter(cor == max(cor))

## `summarise()` ungrouping output (override with `.groups` argument)
## # A tibble: 1 x 2
##   pair      cor
##   <chr>  <dbl>
## 1 father_son 0.430
```

Which pair has the **weakest** correlation in heights?

```
galton %>%
  group_by(pair) %>%
  summarize(cor = cor(parentHeight, childHeight)) %>%
  filter(cor == min(cor))

## `summarise()` ungrouping output (override with `.groups` argument)
## # A tibble: 1 x 2
##   pair      cor
##   <chr>  <dbl>
## 1 mother_son 0.343
```

Question 10 has two parts. The information here applies to both parts.

Use `lm()` and the **broom** package to fit regression lines for each parent-child pair type. Compute the least squares estimates, standard errors, confidence intervals and p-values for the `parentHeight` coefficient for each pair.

10a. What is the estimate of the father-daughter coefficient?

```
galton %>%
  group_by(pair) %>%
  do(tidy(lm(childHeight ~ parentHeight, data = .), conf.int = TRUE)) %>%
  filter(term == "parentHeight", pair == "father_daughter") %>%
  pull(estimate)
```

```
## [1] 0.345
```

For every 1-inch increase in mother's height, how many inches does the typical son's height increase?

```
galton %>%
  group_by(pair) %>%
  do(tidy(lm(childHeight ~ parentHeight, data = .), conf.int = TRUE)) %>%
  filter(term == "parentHeight", pair == "mother_son") %>%
  pull(estimate)
```

```
## [1] 0.381
```

10b. Which sets of parent-child heights are significantly correlated at a p-value cut off of .05?

```
galton %>%
  group_by(pair) %>%
  do(tidy(lm(childHeight ~ parentHeight, data = .), conf.int = TRUE)) %>%
  filter(term == "parentHeight" & p.value < .05)
```

```
## # A tibble: 4 x 8
## # Groups:   pair [4]
##   pair      term      estimate std.error statistic  p.value conf.low conf.high
##   <chr>    <chr>      <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 father_da~ parentHe~  0.345    0.0599     5.77  3.56e-8    0.227    0.464
## 2 father_son parentHe~  0.443    0.0700     6.33  1.94e-9    0.305    0.581
## 3 mother_da~ parentHe~  0.394    0.0720     5.47  1.56e-7    0.252    0.536
## 4 mother_son parentHe~  0.381    0.0784     4.86  2.59e-6    0.226    0.535
```

- ☒ A. father-daughter
- ☒ B. father-son
- ☒ C. mother-daughter
- ☒ D. mother-son

Which of the following statements are true?

- ☒ A. All of the confidence intervals overlap each other.
- ☐ B. At least one confidence interval covers zero.
- ☒ C. The confidence intervals involving mothers' heights are larger than the confidence intervals involving fathers' heights.
- ☐ D. The confidence intervals involving daughters' heights are larger than the confidence intervals involving sons' heights.
- ☒ E. The data are consistent with inheritance of height being independent of the child's gender.
- ☒ F. The data are consistent with inheritance of height being independent of the parent's gender.

Building a Better Offensive Metric for Baseball

The textbook for this section is available [here](#)

Code

```
# linear regression with two variables
fit <- Teams %>%
  filter(yearID %in% 1961:2001) %>%
```

```

mutate(BB = BB/G, HR = HR/G, R = R/G) %>%
lm(R ~ BB + HR, data = .)
tidy(fit, conf.int = TRUE)

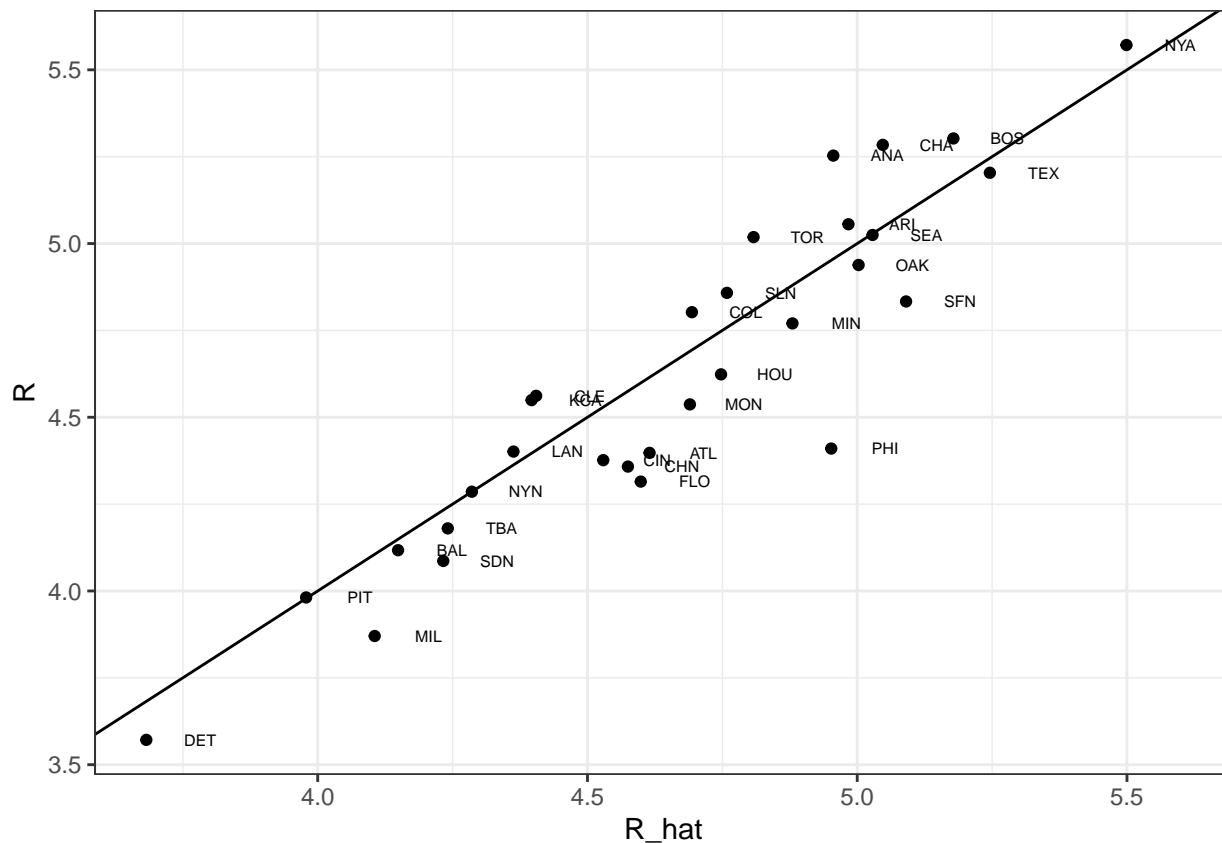
## # A tibble: 3 x 7
##   term          estimate std.error statistic    p.value conf.low conf.high
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)    1.74      0.0824     21.2 7.62e- 83    1.58      1.91
## 2 BB             0.387     0.0270     14.3 1.20e- 42    0.334     0.440
## 3 HR             1.56      0.0490     31.9 1.78e-155    1.47      1.66

# regression with BB, singles, doubles, triples, HR
fit <- Teams %>%
  filter(yearID %in% 1961:2001) %>%
  mutate(BB = BB / G,
         singles = (H - X2B - X3B - HR) / G,
         doubles = X2B / G,
         triples = X3B / G,
         HR = HR / G,
         R = R / G) %>%
  lm(R ~ BB + singles + doubles + triples + HR, data = .)
coefs <- tidy(fit, conf.int = TRUE)
coefs

## # A tibble: 6 x 7
##   term          estimate std.error statistic    p.value conf.low conf.high
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)   -2.77      0.0862    -32.1 4.76e-157   -2.94     -2.60
## 2 BB             0.371     0.0117     31.6 1.87e-153    0.348     0.394
## 3 singles        0.519     0.0127     40.8 8.67e-217    0.494     0.544
## 4 doubles        0.771     0.0226     34.1 8.44e-171    0.727     0.816
## 5 triples        1.24      0.0768     16.1 2.12e- 52    1.09      1.39
## 6 HR             1.44      0.0243     59.3 0.          1.40      1.49

# predict number of runs for each team in 2002 and plot
Teams %>%
  filter(yearID %in% 2002) %>%
  mutate(BB = BB/G,
         singles = (H-X2B-X3B-HR)/G,
         doubles = X2B/G,
         triples =X3B/G,
         HR=HR/G,
         R=R/G) %>%
  mutate(R_hat = predict(fit, newdata = .)) %>%
  ggplot(aes(R_hat, R, label = teamID)) +
  geom_point() +
  geom_text(nudge_x=0.1, cex = 2) +
  geom_abline()

```



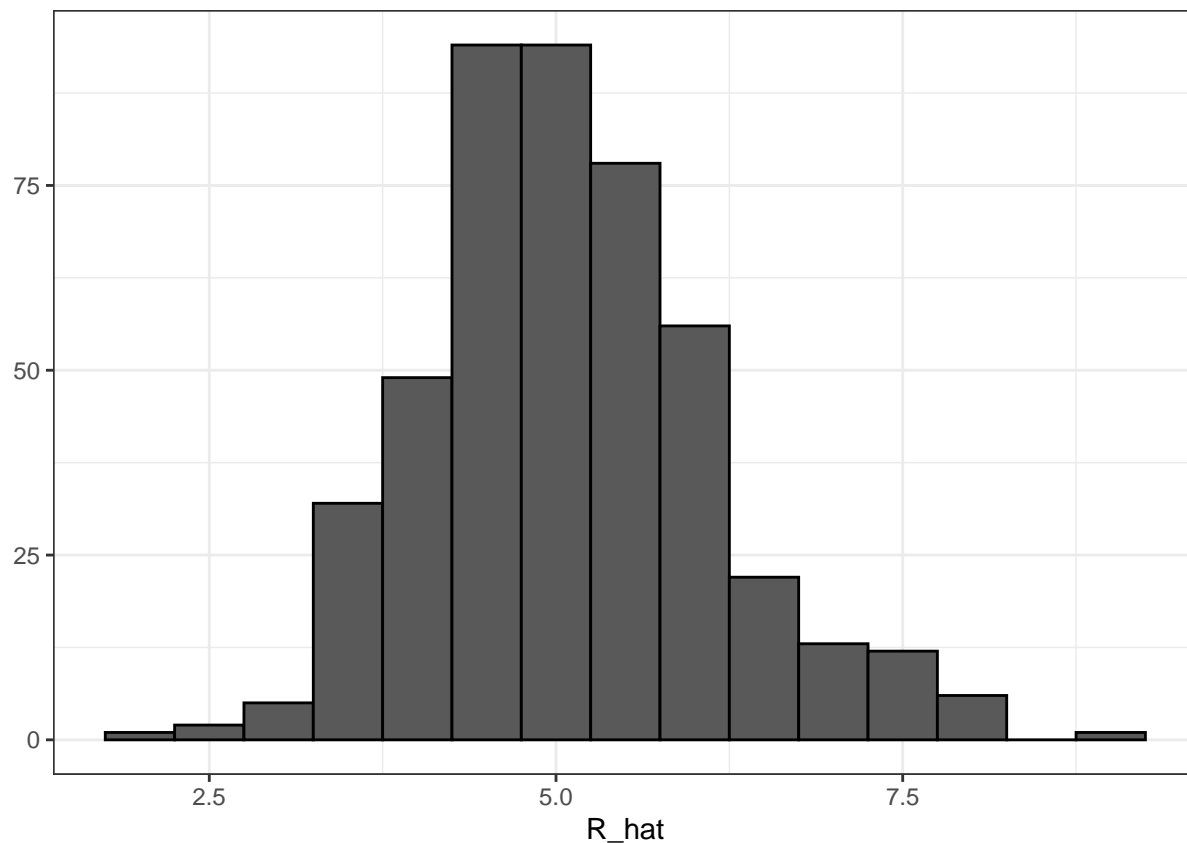
```
# average number of team plate appearances per game
pa_per_game <- Batting %>% filter(yearID == 2002) %>%
  group_by(teamID) %>%
  summarize(pa_per_game = sum(AB+BB)/max(G)) %>%
  pull(pa_per_game) %>%
  mean
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
# compute per-plate-appearance rates for players available in 2002 using previous data
players <- Batting %>% filter(yearID %in% 1999:2001) %>%
  group_by(playerID) %>%
  mutate(PA = BB + AB) %>%
  summarize(G = sum(PA)/pa_per_game,
    BB = sum(BB)/G,
    singles = sum(H-X2B-X3B-HR)/G,
    doubles = sum(X2B)/G,
    triples = sum(X3B)/G,
    HR = sum(HR)/G,
    AVG = sum(H)/sum(AB),
    PA = sum(PA)) %>%
  filter(PA >= 300) %>%
  select(-G) %>%
  mutate(R_hat = predict(fit, newdata = .))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
# plot player-specific predicted runs
qplot(R_hat, data = players, geom = "histogram", binwidth = 0.5, color = I("black"))
```



```
# add 2002 salary of each player
players <- Salaries %>%
  filter(yearID == 2002) %>%
  select(playerID, salary) %>%
  right_join(players, by="playerID")

# add defensive position
position_names <- c("G_p", "G_c", "G_1b", "G_2b", "G_3b", "G_ss", "G_lf", "G_cf", "G_rf")
tmp_tab <- Appearances %>%
  filter(yearID == 2002) %>%
  group_by(playerID) %>%
  summarize_at(position_names, sum) %>%
  ungroup()
pos <- tmp_tab %>%
  select(position_names) %>%
  apply(., 1, which.max)

## Note: Using an external vector in selections is ambiguous.
## i Use `all_of(position_names)` instead of `position_names` to silence this message.
## i See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.

players <- data_frame(playerID = tmp_tab$playerID, POS = position_names[pos]) %>%
  mutate(POS = str_to_upper(str_remove(POS, "G_"))) %>%
  filter(POS != "P") %>%
  right_join(players, by="playerID") %>%
  filter(!is.na(POS) & !is.na(salary))
```

```
## Warning: `data_frame()` is deprecated as of tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

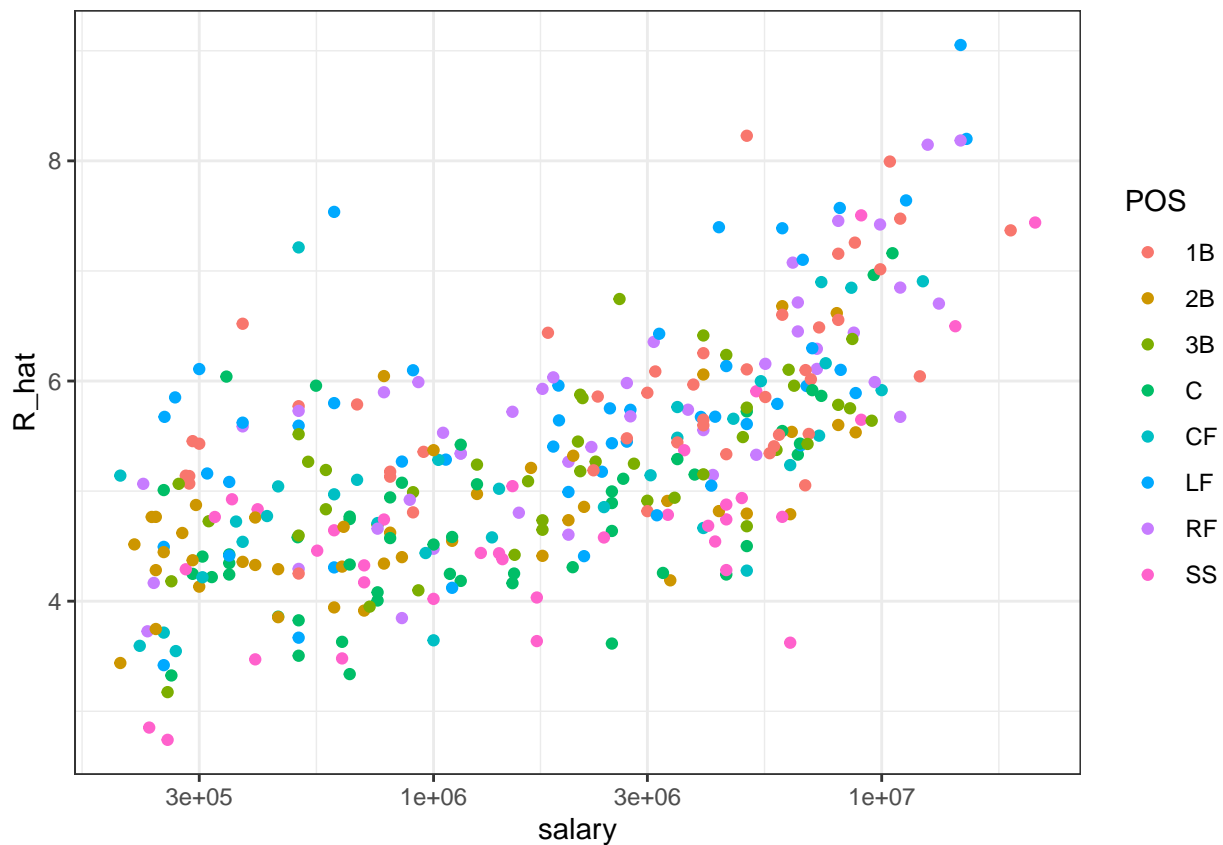
```
# add players' first and last names
players <- Master %>%
  select(playerID, nameFirst, nameLast, debut) %>%
  mutate(debut = as.Date(debut)) %>%
  right_join(players, by="playerID")

# top 10 players
players %>% select(nameFirst, nameLast, POS, salary, R_hat) %>%
  arrange(desc(R_hat)) %>%
  top_n(10)
```

```
## Selecting by R_hat
```

```
##   nameFirst nameLast POS  salary R_hat
## 1    Barry    Bonds  LF 15000000  9.05
## 2    Todd    Helton  1B  5000000  8.23
## 3    Manny   Ramirez  LF 15462727  8.20
## 4    Sammy    Sosa   RF 15000000  8.19
## 5    Larry   Walker  RF 12666667  8.15
## 6    Jason    Giambi  1B 10428571  7.99
## 7   Chipper    Jones  LF 11333333  7.64
## 8    Brian    Giles  LF  8063003  7.57
## 9    Albert   Pujols  LF   600000  7.54
## 10   Nomar Garciaparra SS  9000000  7.51
```

```
# players with a higher metric have higher salaries
players %>% ggplot(aes(salary, R_hat, color = POS)) +
  geom_point() +
  scale_x_log10()
```

```
# remake plot without players that debuted after 1998
if(!require(lubridate)) install.packages("lubridate")
```

```
## Loading required package: lubridate
```

```
##
```

```
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
##      date, intersect, setdiff, union
```

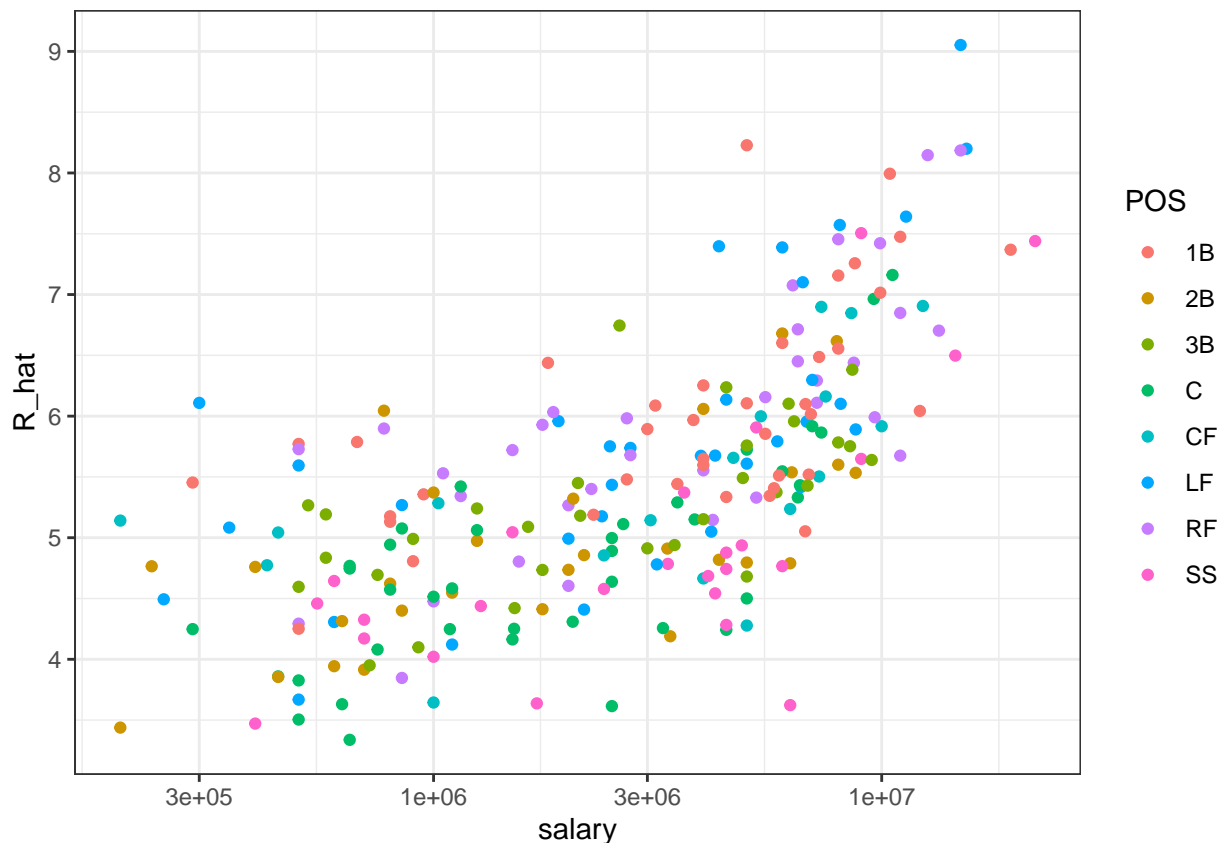
```
library(lubridate)
```

```
players %>% filter(year(debut) < 1998) %>%
```

```
ggplot(aes(salary, R_hat, color = POS)) +
```

```
  geom_point() +
```

```
  scale_x_log10()
```



Building a Better Offensive Metric for Baseball: Linear Programming

A way to actually pick the players for the team can be done using what computer scientists call linear programming. Although we don't go into this topic in detail in this course, we include the code anyway:

```
if(!require(reshape2)) install.packages("reshape2")
```

```
## Loading required package: reshape2
```

```
##
```

```
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
```

```
##
```

```
## smiths
```

```
if(!require(lpSolve)) install.packages("lpSolve")
```

```
## Loading required package: lpSolve
```

```
library(reshape2)
```

```
library(lpSolve)
```

```
players <- players %>% filter(debut <= "1997-01-01" & debut > "1988-01-01")
```

```
constraint_matrix <- acast(players, POS ~ playerID, fun.aggregate = length)
```

```
## Using R_hat as value column: use value.var to override.
```

```
npos <- nrow(constraint_matrix)
```

```
constraint_matrix <- rbind(constraint_matrix, salary = players$salary)
```

```

constraint_dir <- c(rep("==", npos), "<=")
constraint_limit <- c(rep(1, npos), 50*10^6)
lp_solution <- lp("max", players$R_hat,
                  constraint_matrix, constraint_dir, constraint_limit,
                  all.int = TRUE)

```

This algorithm chooses these 9 players:

```

our_team <- players %>%
  filter(lp_solution$solution == 1) %>%
  arrange(desc(R_hat))
our_team %>% select(nameFirst, nameLast, POS, salary, R_hat)

```

##	nameFirst	nameLast	POS	salary	R_hat
## 1	Larry	Walker	RF	12666667	8.15
## 2	Nomar	Garcia	SS	9000000	7.51
## 3	Luis	Gonzalez	LF	4333333	7.40
## 4	Mike	Piazza	C	10571429	7.16
## 5	Jim	Edmonds	CF	7333333	6.90
## 6	Phil	Nevin	3B	2600000	6.75
## 7	Greg	Colbrunn	1B	1800000	6.44
## 8	Terry	Shumpert	2B	775000	6.04

We note that these players all have above average BB and HR rates while the same is not true for singles.

```

my_scale <- function(x) (x - median(x))/mad(x)
players %>% mutate(BB = my_scale(BB),
                  singles = my_scale(singles),
                  doubles = my_scale(doubles),
                  triples = my_scale(triples),
                  HR = my_scale(HR),
                  AVG = my_scale(AVG),
                  R_hat = my_scale(R_hat)) %>%
  filter(playerID %in% our_team$playerID) %>%
  select(nameFirst, nameLast, BB, singles, doubles, triples, HR, AVG, R_hat) %>%
  arrange(desc(R_hat))

```

##	nameFirst	nameLast	BB	singles	doubles	triples	HR	AVG	R_hat
## 1	Larry	Walker	1.0605	0.6554	0.922	1.562	1.566	2.835	2.904
## 2	Nomar	Garcia	0.0274	1.6371	3.118	0.336	0.625	3.197	2.244
## 3	Luis	Gonzalez	0.7046	0.0000	1.413	0.537	1.355	1.829	2.133
## 4	Mike	Piazza	0.3129	-0.0547	-0.242	-1.274	2.035	1.252	1.891
## 5	Jim	Edmonds	1.8074	-1.1409	0.674	-0.674	1.264	0.579	1.621
## 6	Phil	Nevin	0.4909	-0.6479	0.764	-1.098	1.548	0.728	1.463
## 7	Greg	Colbrunn	0.2703	0.6546	0.784	0.585	0.475	1.375	1.148
## 8	Terry	Shumpert	-0.1576	0.1221	1.326	3.908	-0.123	0.859	0.744

On Base Plus Slugging (OPS)

Key point

The on-base-percentage plus slugging percentage (OPS) metric is:

$$\frac{BB}{PA} + \frac{(Singles + 2Doubles + 3Triples + 4HR)}{AB}$$

Regression Fallacy

The textbook for this section is available [here](#)

Key points

- Regression can bring about errors in reasoning, especially when interpreting individual observations.
- The example showed in the video demonstrates that the “**sophomore slump**” observed in the data is caused by regressing to the mean.

Code

The code to create a table with player ID, their names, and their most played position:

```
playerInfo <- Fielding %>%
  group_by(playerID) %>%
  arrange(desc(G)) %>%
  slice(1) %>%
  ungroup %>%
  left_join(Master, by="playerID") %>%
  select(playerID, nameFirst, nameLast, POS)
```

The code to create a table with only the ROY award winners and add their batting statistics:

```
ROY <- AwardsPlayers %>%
  filter(awardID == "Rookie of the Year") %>%
  left_join(playerInfo, by="playerID") %>%
  rename(rookie_year = yearID) %>%
  right_join(Batting, by="playerID") %>%
  mutate(AVG = H/AB) %>%
  filter(POS != "P")
```

The code to keep only the rookie and sophomore seasons and remove players who did not play sophomore seasons:

```
ROY <- ROY %>%
  filter(yearID == rookie_year | yearID == rookie_year+1) %>%
  group_by(playerID) %>%
  mutate(rookie = ifelse(yearID == min(yearID), "rookie", "sophomore")) %>%
  filter(n() == 2) %>%
  ungroup %>%
  select(playerID, rookie_year, rookie, nameFirst, nameLast, AVG)
```

The code to use the spread function to have one column for the rookie and sophomore years batting averages:

```
ROY <- ROY %>% spread(rookie, AVG) %>% arrange(desc(rookie))
ROY
```

```
## # A tibble: 102 x 6
##   playerID  rookie_year nameFirst nameLast  rookie  sophomore
##   <chr>         <int> <chr>      <chr>    <dbl>    <dbl>
## 1 mccovwi01     1959 Willie   McCovey  0.354    0.238
## 2 suzukic01     2001 Ichiro   Suzuki   0.350    0.321
## 3 bumbral01     1973 Al      Bumbray  0.337    0.233
## 4 lynnfr01      1975 Fred    Lynn     0.331    0.314
## 5 pujola01      2001 Albert  Pujols   0.329    0.314
## 6 troutmi01     2012 Mike    Trout    0.326    0.323
## 7 braunry02     2007 Ryan    Braun    0.324    0.285
## 8 olivato01     1964 Tony    Oliva    0.323    0.321
## 9 hargrmi01     1974 Mike    Hargrove 0.323    0.303
```

```
## 10 darkal01      1948 A1      Dark      0.322      0.276
## # ... with 92 more rows
```

The code to calculate the proportion of players who have a lower batting average their sophomore year:

```
mean(ROY$sophomore - ROY$rookie <= 0)
```

```
## [1] 0.686
```

The code to do the similar analysis on all players that played the 2013 and 2014 seasons and batted more than 130 times (minimum to win Rookie of the Year):

```
two_years <- Batting %>%
  filter(yearID %in% 2013:2014) %>%
  group_by(playerID, yearID) %>%
  filter(sum(AB) >= 130) %>%
  summarize(AVG = sum(H)/sum(AB)) %>%
  ungroup %>%
  spread(yearID, AVG) %>%
  filter(!is.na(`2013`) & !is.na(`2014`)) %>%
  left_join(playerInfo, by="playerID") %>%
  filter(POS!="P") %>%
  select(-POS) %>%
  arrange(desc(`2013`)) %>%
  select(nameFirst, nameLast, `2013`, `2014`)
```

```
## `summarise()` regrouping output by 'playerID' (override with `.groups` argument)
```

```
two_years
```

```
## # A tibble: 312 x 4
##   nameFirst nameLast `2013` `2014`
##   <chr>      <chr>    <dbl> <dbl>
## 1 Miguel    Cabrera    0.348 0.313
## 2 Hanley    Ramirez    0.345 0.283
## 3 Michael   Cuddyer    0.331 0.332
## 4 Scooter   Gennett    0.324 0.289
## 5 Joe       Mauer      0.324 0.277
## 6 Mike      Trout      0.323 0.287
## 7 Chris     Johnson    0.321 0.263
## 8 Freddie  Freeman    0.319 0.288
## 9 Yasiel    Puig       0.319 0.296
## 10 Yadier    Molina     0.319 0.282
## # ... with 302 more rows
```

The code to see what happens to the worst performers of 2013:

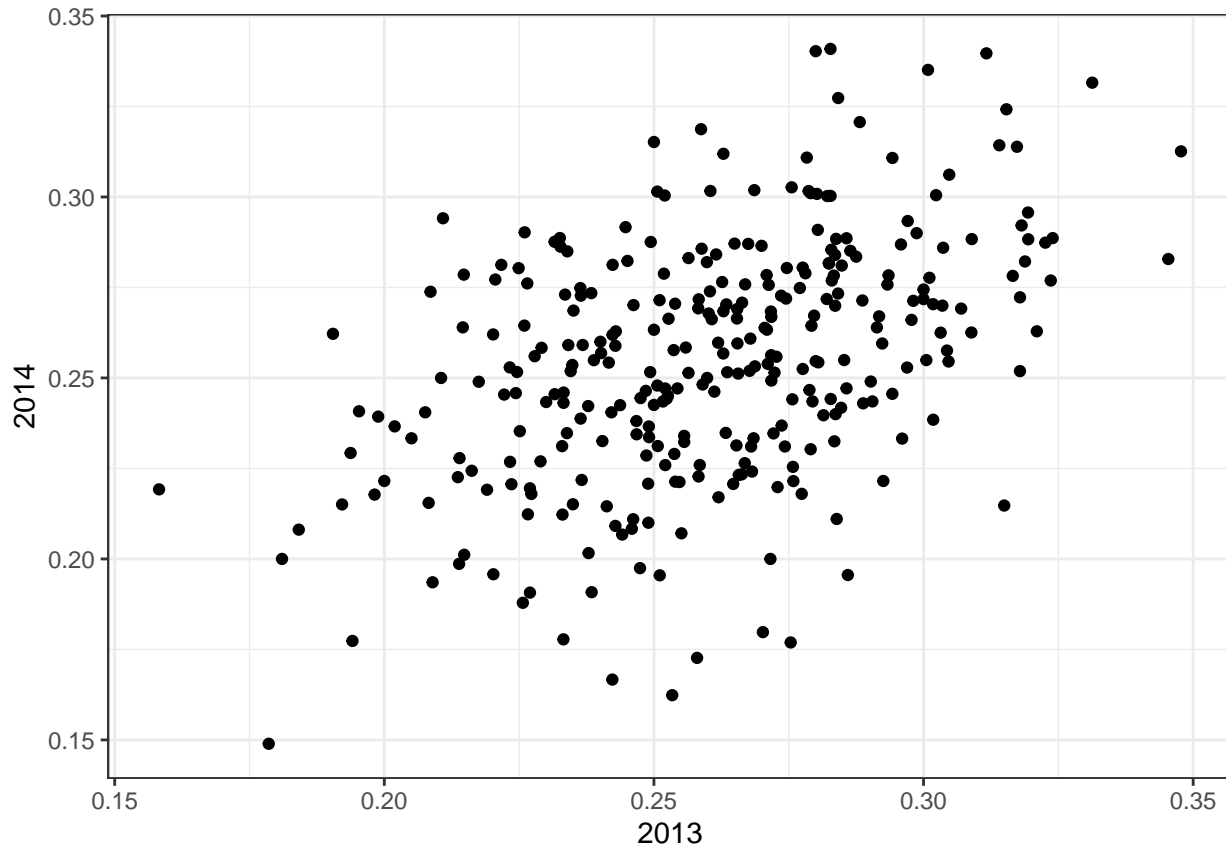
```
arrange(two_years, `2013`)
```

```
## # A tibble: 312 x 4
##   nameFirst nameLast `2013` `2014`
##   <chr>      <chr>    <dbl> <dbl>
## 1 Danny     Espinosa    0.158 0.219
## 2 Dan       Uggla      0.179 0.149
## 3 Jeff      Mathis     0.181 0.2
## 4 B. J.     Upton      0.184 0.208
## 5 Adam      Rosales    0.190 0.262
## 6 Aaron     Hicks      0.192 0.215
```

```
## 7 Chris      Colabello  0.194  0.229
## 8 J. P.      Arencibia  0.194  0.177
## 9 Tyler      Flowers    0.195  0.241
## 10 Ryan      Hanigan    0.198  0.218
## # ... with 302 more rows
```

The code to see the correlation for performance in two separate years:

```
qplot(`2013`, `2014`, data = two_years)
```



```
summarize(two_years, cor(`2013`, `2014`))
```

```
## # A tibble: 1 x 1
##   `cor(\`2013\`, \`2014\`)`
##               <dbl>
## 1               0.460
```

Measurement Error Models

The textbook for this section is available [here](#)

Key points

- Up to now, all our linear regression examples have been applied to two or more random variables. We assume the pairs are bivariate normal and use this to motivate a linear model.
- Another use for linear regression is with **measurement error models**, where it is common to have a non-random covariate (such as time). Randomness is introduced from measurement error rather than sampling or natural variability.

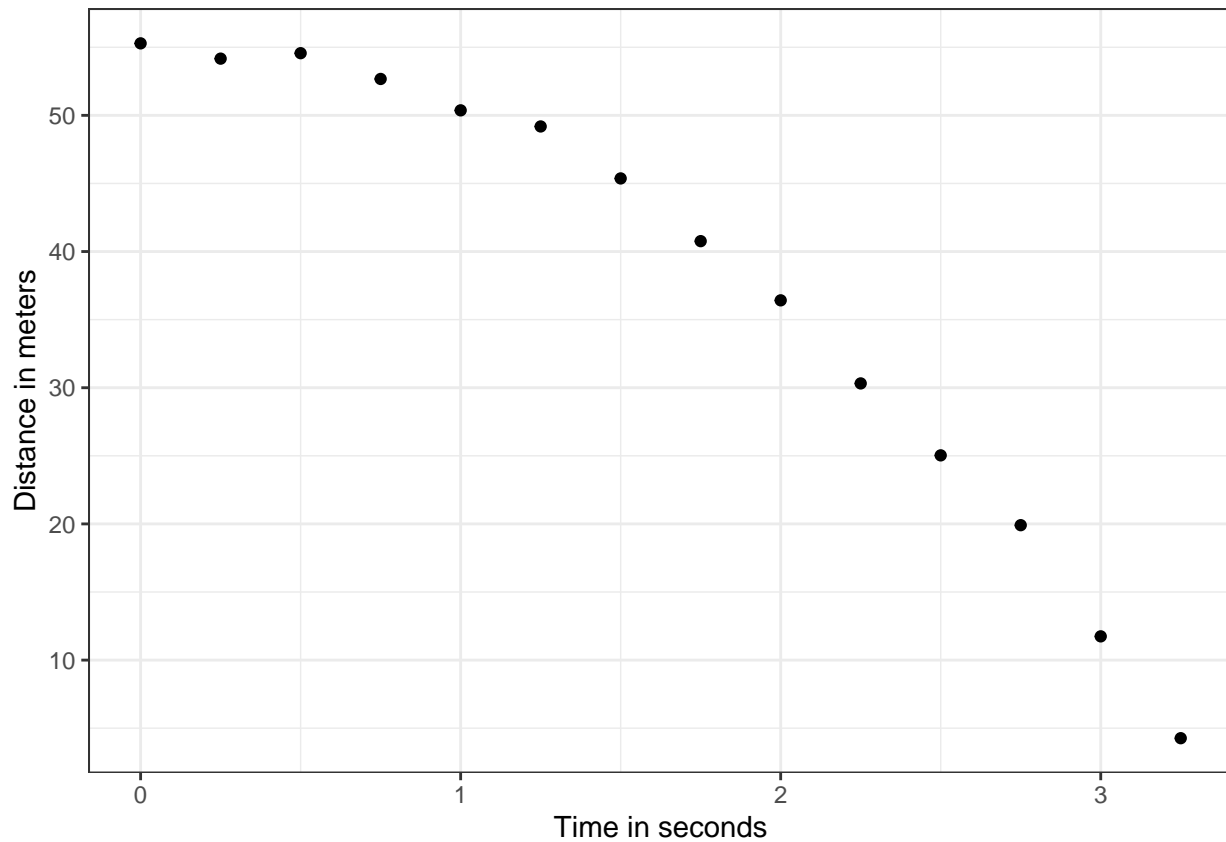
Code

The code to use **dslabs** function **rfalling_object** to generate simulations of dropping balls:

```
falling_object <- rfalling_object()
```

The code to draw the trajectory of the ball:

```
falling_object %>%  
  ggplot(aes(time, observed_distance)) +  
  geom_point() +  
  ylab("Distance in meters") +  
  xlab("Time in seconds")
```



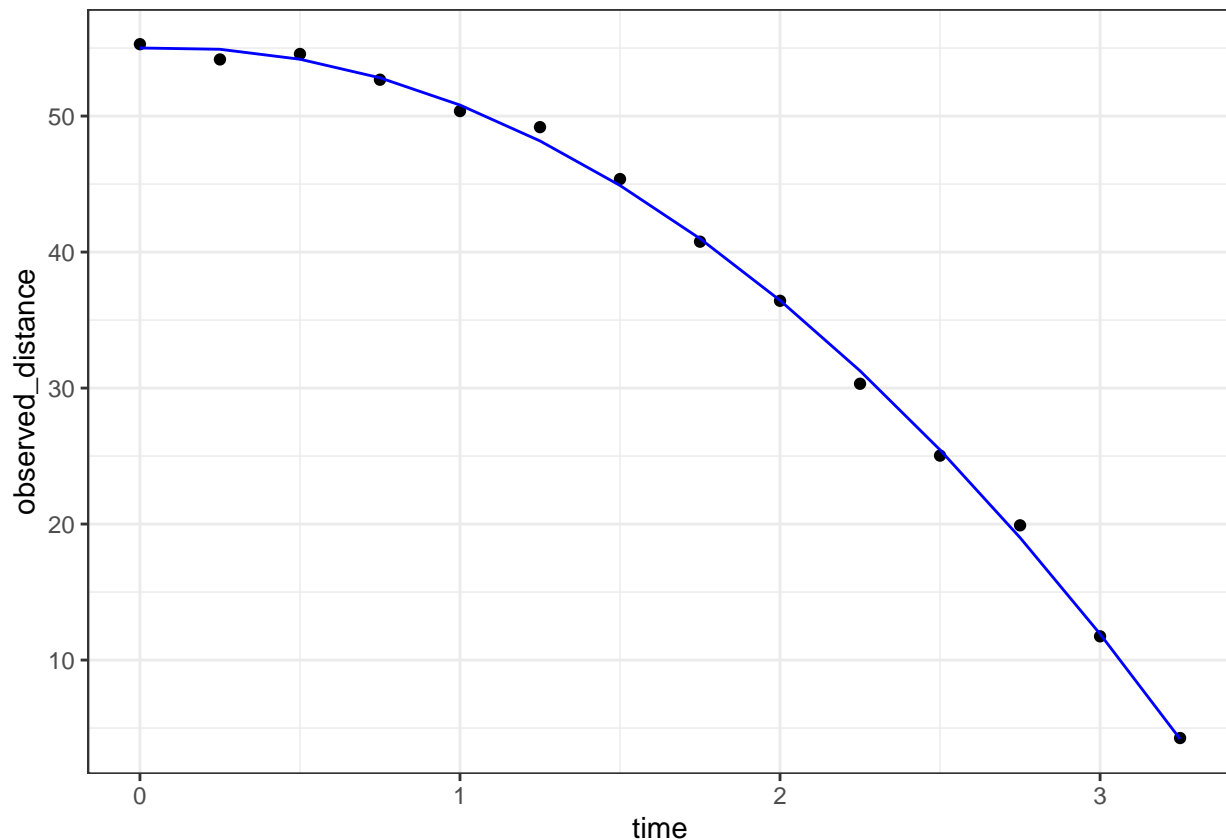
The code to use the **lm()** function to estimate the coefficients:

```
fit <- falling_object %>%  
  mutate(time_sq = time^2) %>%  
  lm(observed_distance~time+time_sq, data=.)  
  
tidy(fit)
```

```
## # A tibble: 3 x 5  
##   term          estimate std.error statistic  p.value  
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>  
## 1 (Intercept)    55.0      0.434    127. 9.15e-19  
## 2 time           0.888      0.620     1.43 1.80e- 1  
## 3 time_sq       -5.08      0.184    -27.7 1.61e-11
```

The code to check if the estimated parabola fits the data:

```
augment(fit) %>%
  ggplot() +
  geom_point(aes(time, observed_distance)) +
  geom_line(aes(time, .fitted), col = "blue")
```



The code to see the summary statistic of the regression:

```
tidy(fit, conf.int = TRUE)
```

```
## # A tibble: 3 x 7
##   term          estimate std.error statistic  p.value conf.low conf.high
##   <chr>         <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)   55.0      0.434   127. 9.15e-19  54.1     56.0
## 2 time          0.888      0.620    1.43 1.80e- 1  -0.476    2.25
## 3 time_sq      -5.08      0.184   -27.7 1.61e-11  -5.49   -4.68
```

Assessment - Regression and Baseball, part 1

1. What is the final linear model we use to predict runs scored per game?

- ☐ A. `lm(R ~ BB + HR)`
- ☐ B. `lm(HR ~ BB + singles + doubles + triples)`
- ☒ C. `lm(R ~ BB + singles + doubles + triples + HR)`
- ☐ D. `lm(R ~ singles + doubles + triples + HR)`

2. We want to estimate runs per game scored by individual players, not just by teams. What summary metric do we calculate to help estimate this?

Look at the code from the video for a hint:


```
pa_per_game <- Batting %>%
  filter(yearID == 2002) %>%
  group_by(teamID) %>%
  summarize(pa_per_game = sum(AB+BB)/max(G)) %>%
  .$pa_per_game %>%
  mean
```

- ☐ A. pa_per_game: the mean number of plate appearances per team per game for each team
 - ☐ B. pa_per_game: the mean number of plate appearances per game for each player
 - ☒ C. pa_per_game: the number of plate appearances per team per game, averaged across all teams
3. Imagine you have two teams. Team A is comprised of batters who, on average, get two bases on balls, four singles, one double, and one home run. Team B is comprised of batters who, on average, get one base on balls, six singles, two doubles, and one triple.

Which team scores more runs, as predicted by our model?

- ☐ A. Team A
 - ☒ B. Team B
 - ☐ C. Tie
 - ☐ D. Impossible to know
4. The on-base-percentage plus slugging percentage (OPS) metric gives the most weight to:
- ☐ A. Singles
 - ☐ B. Doubles
 - ☐ C. Triples
 - ☒ D. Home Runs
5. What statistical concept properly explains the “sophomore slump”?
- ☒ A. Regression to the mean
 - ☐ B. Law of averages
 - ☐ C. Normal distribution
6. In our model of time vs. observed_distance, the randomness of our data was due to:
- ☐ A. sampling
 - ☐ B. natural variability
 - ☒ C. measurement error
7. Which of the following are important assumptions about the measurement errors in this experiment?
- ☒ A. The measurement error is random
 - ☒ B. The measurement error is independent
 - ☒ C. The measurement error has the same distribution for each time i
8. Which of the following scenarios would violate an assumption of our measurement error model?
- ☐ A. The experiment was conducted on the moon.
 - ☒ B. There was one position where it was particularly difficult to see the dropped ball.
 - ☐ C. The experiment was only repeated 10 times, not 100 times.

Assessment - Regression and Baseball, part 2

Section 3 - Confounding Overview

In the Confounding section, you will learn what is perhaps the most important lesson of statistics: that correlation is not causation.

After completing this section, you will be able to:

- Identify examples of spurious correlation and explain how data dredging can lead to spurious correlation.
- Explain how outliers can drive correlation and learn to adjust for outliers using Spearman correlation.
- Explain how reversing cause and effect can lead to associations being confused with causation.
- Understand how confounders can lead to the misinterpretation of associations.
- Explain and give examples of Simpson's Paradox.

This section has one part: Correlation is Not Causation.

The textbook for this section is available [here](#)

Assessment 1 - Correlation is Not Causation: Spurious Correlation

1. In the video, we ran one million tests of correlation for two random variables, X and Y.

How many of these correlations would you expect to have a significant p-value ($p > 0.05$), just by chance?

- ☐ A. 5,000
- ☒ B. 50,000
- ☐ C. 100,000
- ☐ D. It's impossible to know

2. Which of the following are examples of p-hacking?

- ☒ A. Looking for associations between an outcome and several exposures and only reporting the one that is significant.
- ☒ B. Trying several different models and selecting the one that yields the smallest p-value.
- ☒ C. Repeating an experiment multiple times and only reporting the one with the smallest p-value.
- ☐ D. Using a Monte Carlo simulations in an analysis.

Assessment 2 - Correlation is Not Causation: Outliers

1. The Spearman correlation coefficient is robust to outliers because:

- ☐ A. It drops outliers before calculating correlation.
- ☐ B. It is the correlation of standardized values.
- ☒ C. It calculates correlation between ranks, not values.

Assessment 3 - Correlation is Not Causation: Reversing Cause and Effect

1. Which of the following may be examples of reversed cause and effect?

- ☒ A. Past smokers who have quit smoking may be more likely to die from lung cancer.
- ☐ B. Tall fathers are more likely to have tall sons.
- ☒ C. People with high blood pressure tend to have a healthier diet.
- ☒ D. Individuals in a low social status have a higher risk of schizophrenia.

Assessment 4 - Correlation is Not Causation: Confounders

1. What can you do to determine if you are misinterpreting results because of a confounder?

- ☐ A. Nothing, if the p-value says the result is significant, then it is.
- ☒ B. More closely examine the results by stratifying and plotting the data.
- ☐ C. Always assume that you are misinterpreting the results.
- ☐ D. Use linear models to tease out a confounder.

2. Look again at the admissions data using ?admissions. What important characteristic of the table variables do you need to know to understand the calculations used in this video? Select the best answer.

- ☐ A. The data is from 1973.
 - ☐ B. The columns “major” and “gender” are of class character, while “admitted” and “applicants” are numeric.
 - ☐ C. The data is from the “dslabs” package.
 - ☒ D. The column “admitted” is the percent of student admitted, while the column “applicants” is the total number of applicants.
3. In the example in the video, major selectivity confounds the relationship between UC Berkley admission rates and gender because:
- ☐ A. It was harder for women to be admitted to UC Berkeley.
 - ☒ B. Major selectivity is associated with both admission rates and with gender, as women tended to apply to more selective majors.
 - ☐ C. Some majors are more selective than others
 - ☐ D. Major selectivity is not a confounder.

Assessment 5 - Simpson’s Paradox

1. Admission rates at UC Berkeley are an example of Simpson’s Paradox because:
- ☒ A. It appears that men have higher a higher admission rate than women, however, after we stratify by major, we see that on average women have a higher admission rate than men.
 - ☐ B. It was a paradox that women were being admitted at a lower rate than men.
 - ☐ C. The relationship between admissions and gender is confounded by major selectivity.