Data Science for Biological, Medical and Health Research: Notes for 431

Thomas E. Love, Ph.D.

Version: 2017-10-22

Contents

Introduction

These Notes provide a series of examples using R to work through issues that are likely to come up in PQHS/CRSP/MPHP 431.

While these Notes share some of the features of a textbook, they are neither comprehensive nor completely original. The main purpose is to give 431 students a set of common materials on which to draw during the course. In class, we will sometimes:

- reiterate points made in this document,
- amplify what is here,
- simplify the presentation of things done here,
- use new examples to show some of the same techniques,
- refer to issues not mentioned in this document,

but what we don't do is follow these notes very precisely. We assume instead that you will read the materials and try to learn from them, just as you will attend classes and try to learn from them. We welcome feedback of all kinds on this document or anything else. Just email us at 431-help at case dot edu, or submit a pull request.

What you will mostly find are brief explanations of a key idea or summary, accompanied (most of the time) by R code and a demonstration of the results of applying that code.

Everything you see here is available to you as HTML or PDF. You will also have access to the R Markdown files, which contain the code which generates everything in the document, including all of the R results. We will demonstrate the use of R Markdown (this document is generated with the additional help of an R package called `bookdown`) and R Studio (the "program" which we use to interface with the R language) in class.

To download the data and R code related to these notes, visit https://github.com/THOMASELOVE/431data

Structure

The Notes, like the 431 course, fall in three main parts.

Part A is about **visualizing data and exploratory data analyses**. These Notes focus on using R to work through issues that arise in the process of exploring data, managing (cleaning and manipulating) data into a tidy format to facilitate useful work downstream, and describing those data effectively with visualizations, numerical summaries, and some simple models.

Part B is about **making comparisons** with data. The Notes discuss the use of R to address comparisons of means and of rates/proportions, primarily. The main ideas include confidence intervals, the bootstrap and parametric and non-parametric tests of hypotheses. Key ideas from Part A that have an impact here include visualizations to check the assumptions behind our inferences, and cleaning/manipulating data to facilitate our comparisons.

Part C is about **building models** with data. The Notes are primarily concerned (in 431) with linear regression models for continuous quantitative outcomes, using one or more predictors. We'll see how to use models to accomplish many of the comparisons discussed in Part B, and make heavy use of visualization and data management tools developed in Part A to assess our models.

Course Philosophy

In developing this course, we adopt a modern approach that places data at the center of our work. Our goal is to teach you how to do truly reproducible research with modern tools. We want you to be able to answer real questions using data and equip you with the tools you need in order to answer those questions well (Çetinkaya-Rundel (2017) has more on a related teaching philosophy.)

The curriculum includes more on several topics than you might expect from a standard graduate introduction to statistics.

- data gathering
- data wrangling
- exploratory data analysis and visualization
- multivariate modeling
- communication

It also nearly completely avoids formalism and is extremely applied - this is most definitely **not** a course in theoretical or mathematical statistics.

The 431 course is about **getting things done**. It's not a statistics course, nor is it a computer science course. It is instead a course in **data science**.

Chapter 1

Data Science

The definition of **data science** can be a little slippery. One current view of data science, is exemplified by Steven Geringer's 2014 Venn diagram.

- The field encompasses ideas from mathematics and statistics and from computer science, but with a heavy reliance on subject-matter knowledge. In our case, this includes clinical, health-related, medical or biological knowledge.
- As Gelman and Nolan (2017) suggest, the experience and intuition necessary for good statistical practice are hard to obtain, and teaching data science provides an excellent opportunity to reinforce statistical thinking skills across the full cycle of a data analysis project.
- The principal form in which computer science (coding/programming) play a role in this course is to provide a form of communication. You'll need to learn how to express your ideas not just orally and in writing, but also through your code.

1.1 Why a unicorn?

Data Science is a **team** activity. Everyone working in data science brings some part of the necessary skillset, but no one person can cover all three areas alone for excellent projects.

> [The individual who is truly expert in all three key areas (mathematics/statistics, computer science and subject-matter knowledge) is] a mythical beast with magical powers who's rumored to exist but is never actually seen in the wild.
>
> http://www.kdnuggets.com/2016/10/battle-data-science-venn-diagrams.html

1.2 Data Science Project Cycle

A typical data science project can be modeled as follows, which comes from the introduction to the amazing book **R for Data Science**, by Garrett Grolemund and Hadley Wickham, which is a key text for this course (Grolemund and Wickham 2017).

This diagram is sometimes referred to as the Krebs Cycle of Data Science. For more on the steps of a data science project, we encourage you to read the Introduction of Grolemund and Wickham (2017).

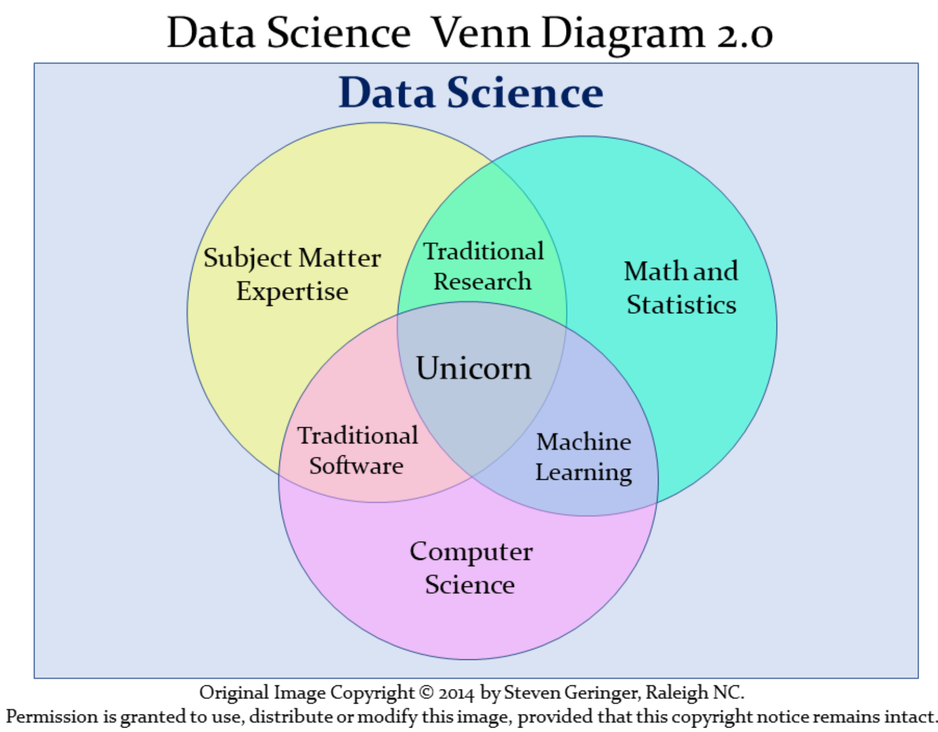


Figure 1.1: Data Science Venn Diagram from Steven Geringer

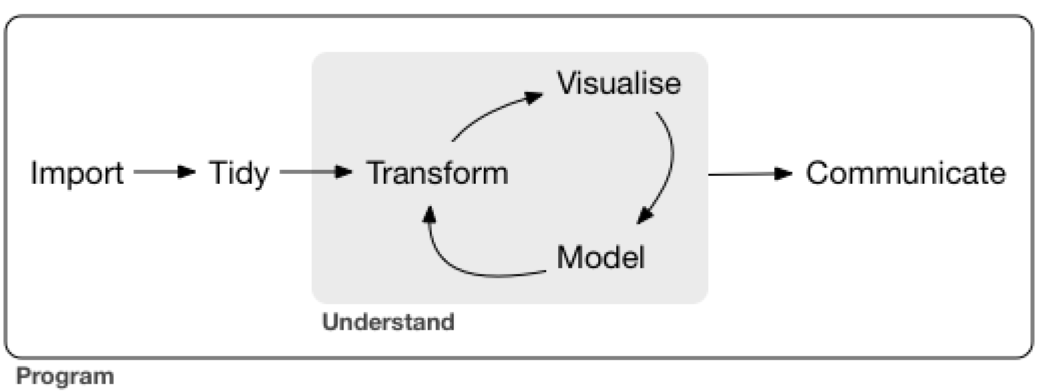


Figure 1.2: Source: R for Data Science: Introduction

1.3 What Will We Discuss in 431?

We'll discuss each of these elements in the 431 course, focusing at the start on understanding our data through transformation, modeling and (especially in the early stages) visualization. In 431, we learn how to get things done.

- We get people working with R and R Studio and R Markdown, even if they are completely new to coding. A gentle introduction is provided at Ismay and Kim (2017)
- We learn how to use the `tidyverse` (http://www.tidyverse.org/), an array of tools in R (mostly developed by Hadley Wickham and his colleagues at R Studio) which share an underlying philosophy to make data science faster, easier, more reproducible and more fun. A critical text for understanding the tidyverse is Grolemund and Wickham (2017). Tidyverse tools facilitate:
  - **importing** data into R, which can be the source of intense pain for some things, but is really quite easy 95% of the time with the right tool.
  - **tidying** data, that is, storing it in a format that includes one row per observation and one column per variable. This is harder, and more important, than you might think.
  - **transforming** data, perhaps by identifying specific subgroups of interest, creating new variables based on existing ones, or calculating summaries.
  - **visualizing** data to generate actual knowledge and identify questions about the data - this is an area where R really shines, and we'll start with it in class.
  - **modeling** data, taking the approach that modeling is complementary to visualization, and allows us to answer questions that visualization helps us identify.
  - and last, but definitely not least, **communicating** results, models and visualizations to others, in a way that is reproducible and effective.
- Some programming/coding is an inevitable requirement to accomplish all of these aims. If you are leery of coding, you'll need to get past that, with the help of this course and our stellar teaching assistants. Getting started is always the most challenging part, but our experience is that most of the pain of developing these new skills evaporates by early October.
- Having completed some fundamental work in Part A of the course, we then learn how to use a variety of R packages and statistical methods to accomplish specific inferential tasks (in Part B, mostly) and modeling tasks (in Part C, mostly.)

Chapter 2

Setting Up R

These Notes make extensive use of

- the statistical software language R, and
- the development environment R Studio,

both of which are free, and you'll need to install them on your machine. Instructions for doing so are in found in the course syllabus.

If you need an even gentler introduction, or if you're just new to R and RStudio and need to learn about them, we encourage you to take a look at http://moderndive.com/, which provides an introduction to statistical and data sciences via R at Ismay and Kim (2017).

2.1 R Markdown

These notes were written using R Markdown. R Markdown, like R and R Studio, is free and open source.

R Markdown is described as an *authoring framework* for data science, which lets you

- save and execute R code
- generate high-quality reports that can be shared with an audience

This description comes from http://rmarkdown.rstudio.com/lesson-1.html which you can visit to get an overview and quick tour of what's possible with R Markdown.

Another excellent resource to learn more about R Markdown tools is the Communicate section (especially the R Markdown chapter) of Grolemund and Wickham (2017).

2.2 R Packages

To start, I'll present a series of commands I run at the beginning of these Notes. These particular commands set up the output so it will look nice as either an HTML or PDF file, and also set up R to use several packages (libraries) of functions that expand its capabilities. A chunk of code like this will occur near the top of any R Markdown work.

```
knitr::opts_chunk$set(comment = NA)

library(boot); library(devtools); library(forcats)
library(grid); library(knitr); library(pander)
```

```r
library(pwr); library(viridis); library(NHANES)
library(tidyverse)

source("data/Love-boost.R")
```

I have deliberately set up this list of loaded packages/libraries to be relatively small, and will add some other packages later, as needed. You only need to install a package once, but you need to reload it every time you start a new session.

2.3 Other Packages

I will also make use of functions in the following packages/libraries, but when I do so, I will explicitly specify the package name, using a command like `Hmisc::describe(x)`, rather than just `describe(x)`, so as to specify that I want the Hmisc package's version of `describe` applied to whatever `x` is. Those packages are:

- `aplpack` which provides `stem.leaf` and `stem.leaf.backback` for building fancier stem-and-leaf displays
- `arm` which provides a set of functions for model building and checking that are used in Gelman and Hill (2007)
- `broom` which turns the results lots of different analyses in R into more useful tidy data frames (tibbles.)
- `car` which provides some tools for building scatterplot matrices, but also many other functions described in Fox and Weisberg (2011)
- `Epi` for 2x2 table analyses and materials for classical epidemiology: http://BendixCarstensen.com/Epi/
- `GGally` for scatterplot and correlation matrix visualizations: http://ggobi.github.io/ggally/
- `ggridges` which is used to make ridgeline plots
- `gridExtra` which includes a variety of functions for manipulating graphs: https://github.com/baptiste/gridextra
- `Hmisc` from Frank Harrell at Vanderbilt U., for its version of `describe` and for many regression modeling functions we'll use in 432. Details on Hmisc are at http://biostat.mc.vanderbilt.edu/wiki/Main/Hmisc. Frank has written several books - the most useful of which for 431 students is probably Harrell and Slaughter (2017)
- `mice`, which we'll use (a little) in 431 for multiple imputation to deal with missing data: http://www.stefvanbuuren.nl/mi/
- `mosaic`, mostly for its `favstats` summary, but Project MOSAIC is a community of educators you might be interested in: http://mosaic-web.org/
- `psych` for its own version of `describe`, but other features are described at http://personality-project.org/r/psych/

We also will use a package called `xda` for two functions called `numSummary` and `charSummary`, but that package gets loaded via `devtools` and GitHub by the code in these Notes.

Several other packages are included below, even though they are not used in these Notes, because they will be used in class sessions or in 432.

When compiling the Notes from the original code files, these packages will need to be installed (but not loaded) in R, or an error will be thrown when compiling this document. To install all of the packages used within these Notes, type in (or copy and paste) the following commands and run them in the R Console. Again, you only need to install a package once, but you need to reload it every time you start a new session.

```r
pkgs <- c("aplpack", "arm", "babynames", "boot", "broom", "car", "devtools", "Epi",
          "faraway", "forcats", "foreign", "gapminder", "GGally", "ggridges",
          "gridExtra", "Hmisc", "knitr", "lme4", "markdown", "MASS",
          "mice", "mosaic", "multcomp", "NHANES", "pander", "psych",
          "pwr", "qcc", "rmarkdown", "rms", "sandwich", "survival",
          "tableone", "tidyverse", "vcd", "viridis")
```

```
install.packages(pkgs)
```

Part A. Exploring Data

Chapter 3

Visualizing Data

Part A of these Notes is designed to ease your transition into working effectively with data, so that you can better understand it. We'll start by visualizing some data from the US National Health and Nutrition Examination Survey, or NHANES. We'll display R code as we go, but we'll return to all of the key coding ideas involved later in the Notes.

3.1 The NHANES data: Collecting a Sample

To begin, we'll gather a random sample of 1,000 subjects participating in NHANES, and then identify several variables of interest about those subjects[1]. The motivation for this example came from a Figure in Baumer, Kaplan, and Horton (2017).

```
# library(NHANES) # already loaded NHANES package/library of functions, data

set.seed(431001)
# use set.seed to ensure that we all get the same random sample
# of 1,000 NHANES subjects in our nh_data collection

nh_data <- sample_n(NHANES, size = 1000) %>%
    select(ID, Gender, Age, Height, Weight, BMI, Pulse, Race1, HealthGen, Diabetes)

nh_data
```

```
# A tibble: 1,000 x 10
      ID Gender   Age Height Weight   BMI Pulse    Race1 HealthGen
   <int> <fctr> <int>  <dbl>  <dbl> <dbl> <int>   <fctr>    <fctr>
 1 59640   male    54    176  129.0  41.8    74    White      Good
 2 59826 female    67    156   50.2  20.5    66    White     Vgood
 3 56340   male     9    128   23.3  14.2    86    Black      <NA>
 4 56747   male    33    194  105.1  27.9    68    White     Vgood
 5 51754 female    58    167  106.0  37.9    70    White      <NA>
 6 52712   male     6    109   16.9  14.3    NA    White      <NA>
 7 63908   male    55    169   90.6  31.9    62  Mexican     Vgood
 8 60865 female    25    156   55.0  22.8    58    Other     Vgood
 9 66642   male    41    178   89.3  28.2    72    White     Vgood
10 59880 female    45    163   98.3  36.9    80 Hispanic      Good
```

[1] For more on the NHANES data available in the NHANES package, type ?NHANES in the Console in R Studio.

```
# ... with 990 more rows, and 1 more variables: Diabetes <fctr>
```

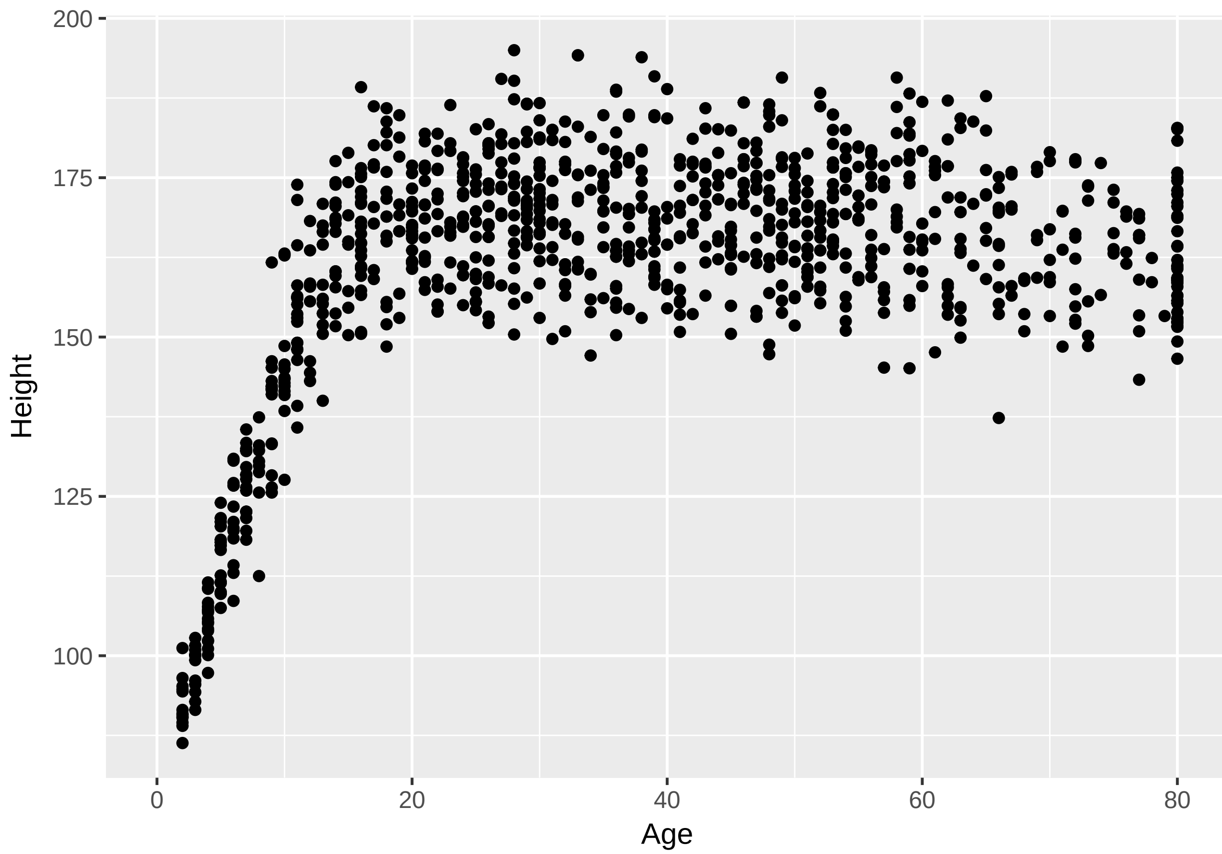
We have 1000 rows (observations) and 10 columns (variables) that describe the subjects listed in the rows.

3.2 Age and Height

Suppose we want to visualize the relationship of Height and Age in our 1,000 NHANES observations. The best choice is likely to be a scatterplot.

```
ggplot(data = nh_data, aes(x = Age, y = Height)) +
    geom_point()
```

```
Warning: Removed 25 rows containing missing values (geom_point).
```



We note several interesting results here.

1. As a warning, R tells us that it has "Removed 25 rows containing missing values (geom_point)." Only 975 subjects plotted here, because the remaining 25 people have missing (NA) values for either Height, Age or both.
2. Unsurprisingly, the measured Heights of subjects grow from Age 0 to Age 20 or so, and we see that a typical Height increases rapidly across these Ages. The middle of the distribution at later Ages is pretty consistent at at a Height somewhere between 150 and 175. The units aren't specified, but we expect they must be centimeters. The Ages are clearly reported in Years.
3. No Age is reported over 80, and it appears that there is a large cluster of Ages at 80. This may be due to a requirement that Ages 80 and above be reported at 80 so as to help mask the identity of those

individuals.[2]

As in this case, we're going to build most of our visualizations using tools from the `ggplot2` package, which is part of the `tidyverse` series of packages. You'll see similar coding structures throughout this Chapter, most of which are covered as well in Chapter 3 of Grolemund and Wickham (2017).

3.3 Subset of Subjects with Known Age and Height

Before we move on, let's manipulate the data set a bit, to focus on only those subjects who have complete data on both Age and Height. This will help us avoid that warning message.

```
nh_dat2 <- nh_data %>%
    filter(complete.cases(Age, Height))

summary(nh_dat2)
```

```
       ID            Gender         Age            Height     
 Min.   :51654   female:498   Min.   : 2.0   Min.   : 86.3  
 1st Qu.:56752   male  :477   1st Qu.:20.0   1st Qu.:156.4  
 Median :61453                Median :36.0   Median :165.8  
 Mean   :61602                Mean   :37.3   Mean   :161.7  
 3rd Qu.:66484                3rd Qu.:53.0   3rd Qu.:174.1  
 Max.   :71826                Max.   :80.0   Max.   :195.0  
                                                            
     Weight            BMI            Pulse              Race1    
 Min.   : 12.5   Min.   :13.2   Min.   : 42.0   Black   :112  
 1st Qu.: 57.6   1st Qu.:21.6   1st Qu.: 66.0   Hispanic: 69  
 Median : 73.4   Median :26.1   Median : 72.0   Mexican :104  
 Mean   : 73.4   Mean   :27.0   Mean   : 73.7   White   :607  
 3rd Qu.: 90.2   3rd Qu.:31.1   3rd Qu.: 82.0   Other   : 83  
 Max.   :198.7   Max.   :80.6   Max.   :124.0                 
 NA's   :2       NA's   :2      NA's   :120                   
     HealthGen   Diabetes  
 Excellent: 87   No  :910  
 Vgood    :276   Yes : 64  
 Good     :276   NA's:  1  
 Fair     :103             
 Poor     : 15             
 NA's     :218             
```

Note that the units and explanations for these variables are contained in the NHANES help file, available via ?NHANES in the Console of R Studio.

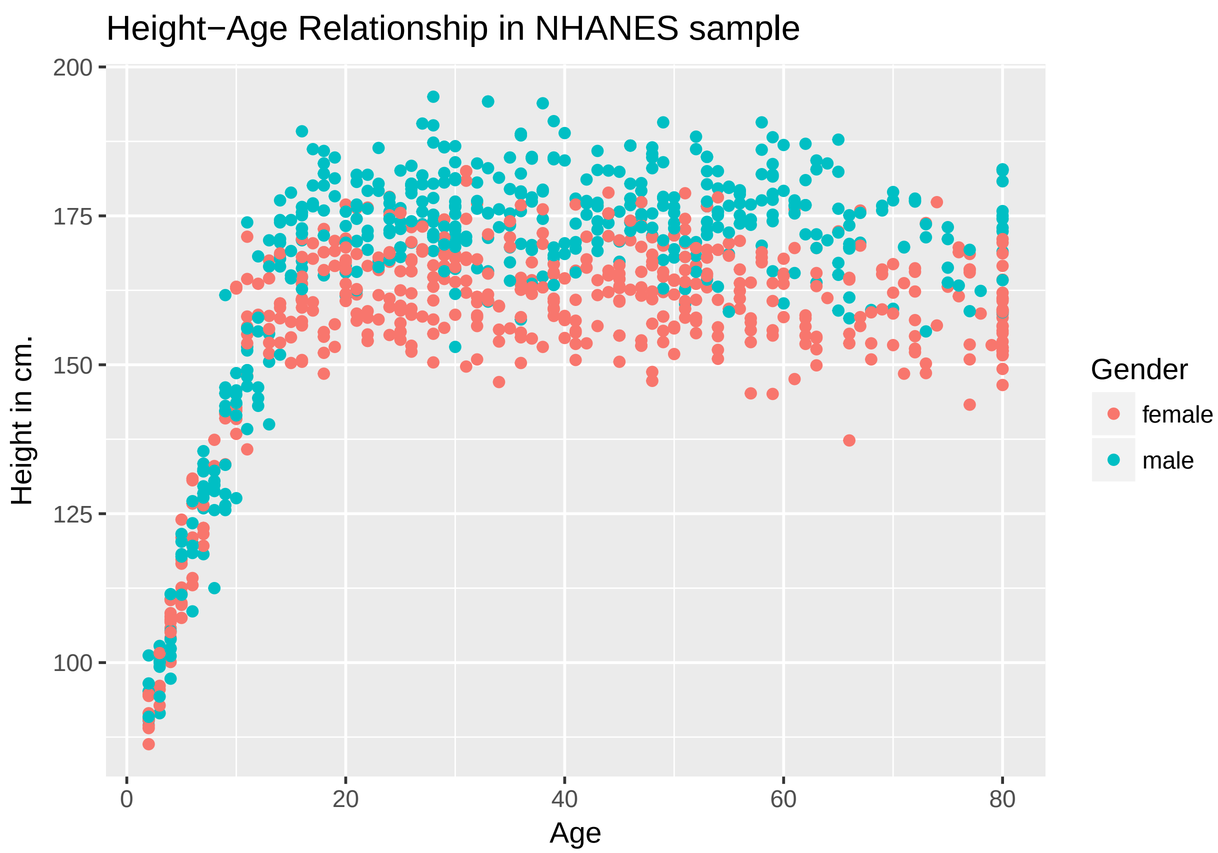
3.4 Age-Height and Gender?

Let's add Gender to the plot using color, and also adjust the y axis label to incorporate the units of measurement.

```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Gender)) +
    geom_point() +
```

[2] If you visit the NHANES help file with ?NHANES, you will see that subjects 80 years or older were indeed recorded as 80.

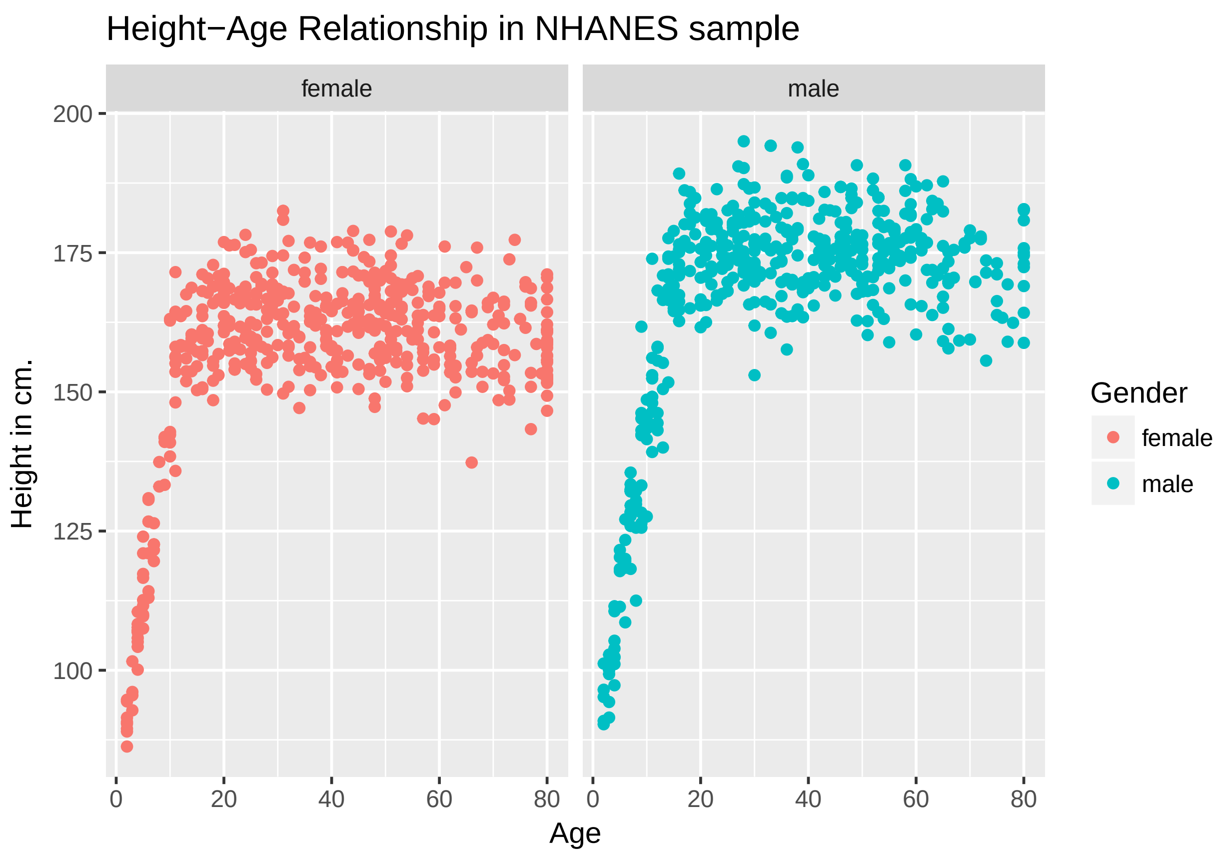
```
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.")
```



3.4.1 Can we show the Female and Male relationships in separate panels?

Sure.

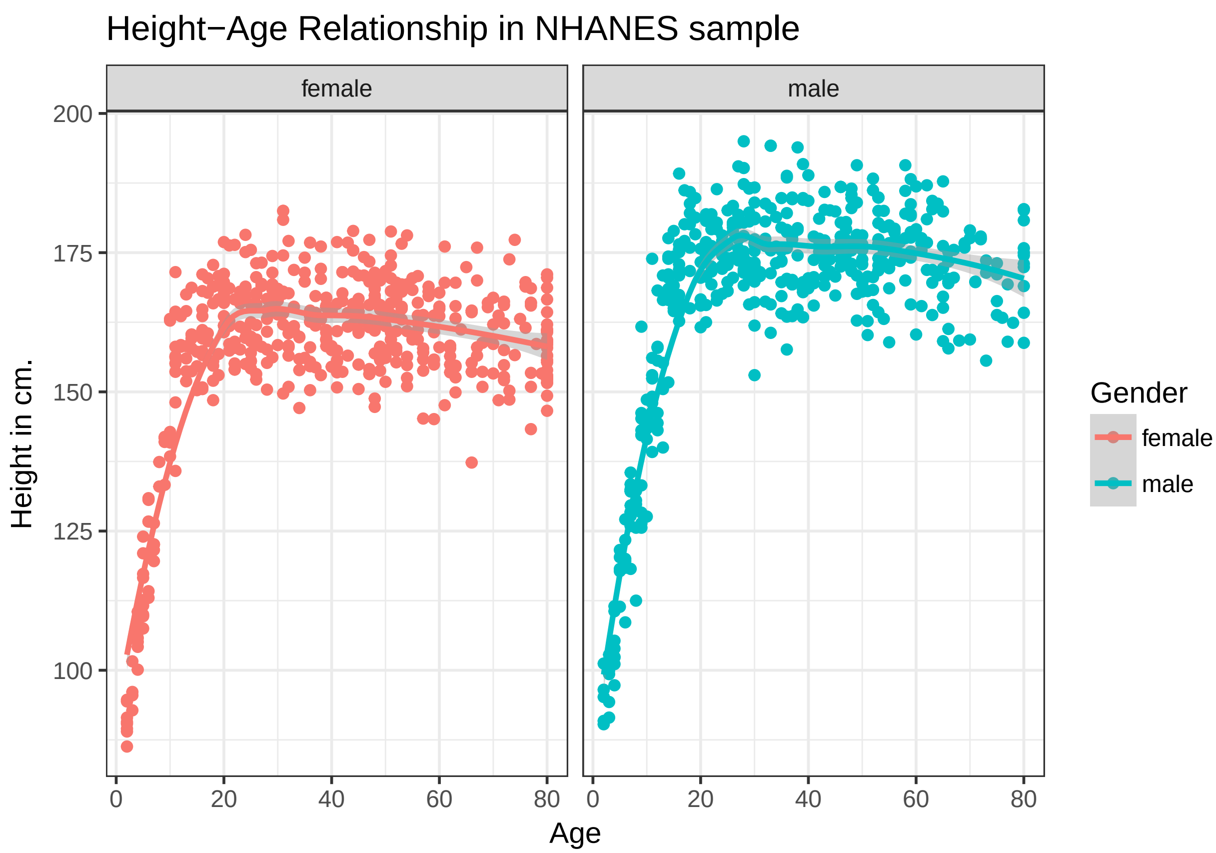
```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Gender)) +
    geom_point() +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    facet_wrap(~ Gender)
```



3.4.2 Can we add a smooth curve to show the relationship in each plot?

Yep, and let's change the theme of the graph to remove the gray background, too.

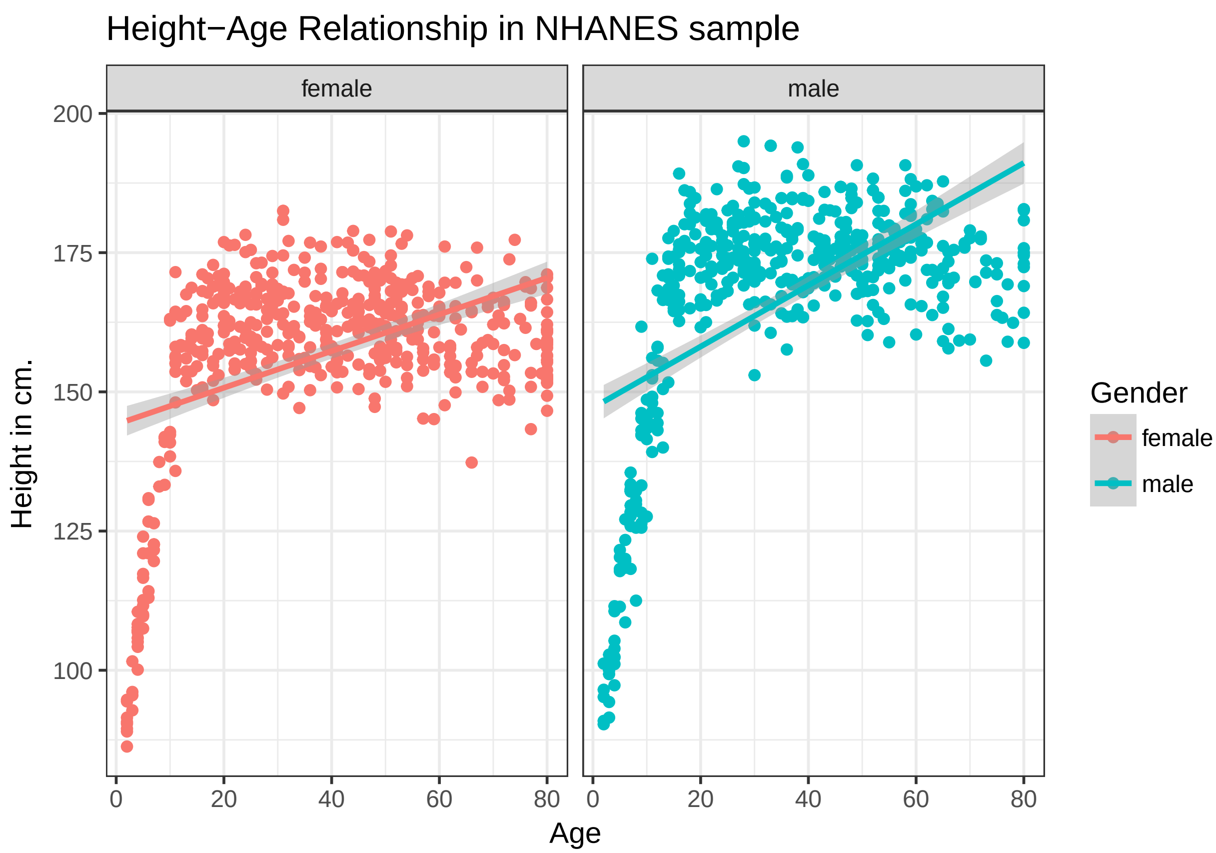
```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Gender)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    theme_bw() +
    facet_wrap(~ Gender)
```



3.4.3 What if we want to assume straight line relationships?

We could look at a linear model in the plot. Does this make sense here?

```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Gender)) +
    geom_point() +
    geom_smooth(method = "lm") +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    theme_bw() +
    facet_wrap(~ Gender)
```



3.5 A Subset: Ages 21-79

Suppose we wanted to look at a subset of our sample - those observations (subjects) whose Age is at least 21 and at most 79. We'll create that sample below, and also subset the variables to include nine of particular interest, and remove any observations with any missingness on *any* of the nine variables we're including here.

```
nh_data_2179 <- nh_data %>%
    filter(Age > 20 & Age < 80) %>%
    select(ID, Gender, Age, Height, Weight, BMI, Pulse, Race1, HealthGen, Diabetes) %>%
    na.omit

nh_data_2179
```

```
# A tibble: 594 x 10
      ID Gender   Age Height Weight   BMI Pulse    Race1 HealthGen
   <int> <fctr> <int>  <dbl>  <dbl> <dbl> <int>   <fctr>    <fctr>
 1 59640   male    54    176  129.0  41.8    74    White      Good
 2 59826 female    67    156   50.2  20.5    66    White     Vgood
 3 56747   male    33    194  105.1  27.9    68    White     Vgood
 4 63908   male    55    169   90.6  31.9    62  Mexican     Vgood
 5 60865 female    25    156   55.0  22.8    58    Other     Vgood
 6 66642   male    41    178   89.3  28.2    72    White     Vgood
 7 59880 female    45    163   98.3  36.9    80 Hispanic      Good
 8 71784 female    24    161   50.2  19.3    72    White     Vgood
 9 67616   male    63    184   70.0  20.6    82    White     Vgood
```

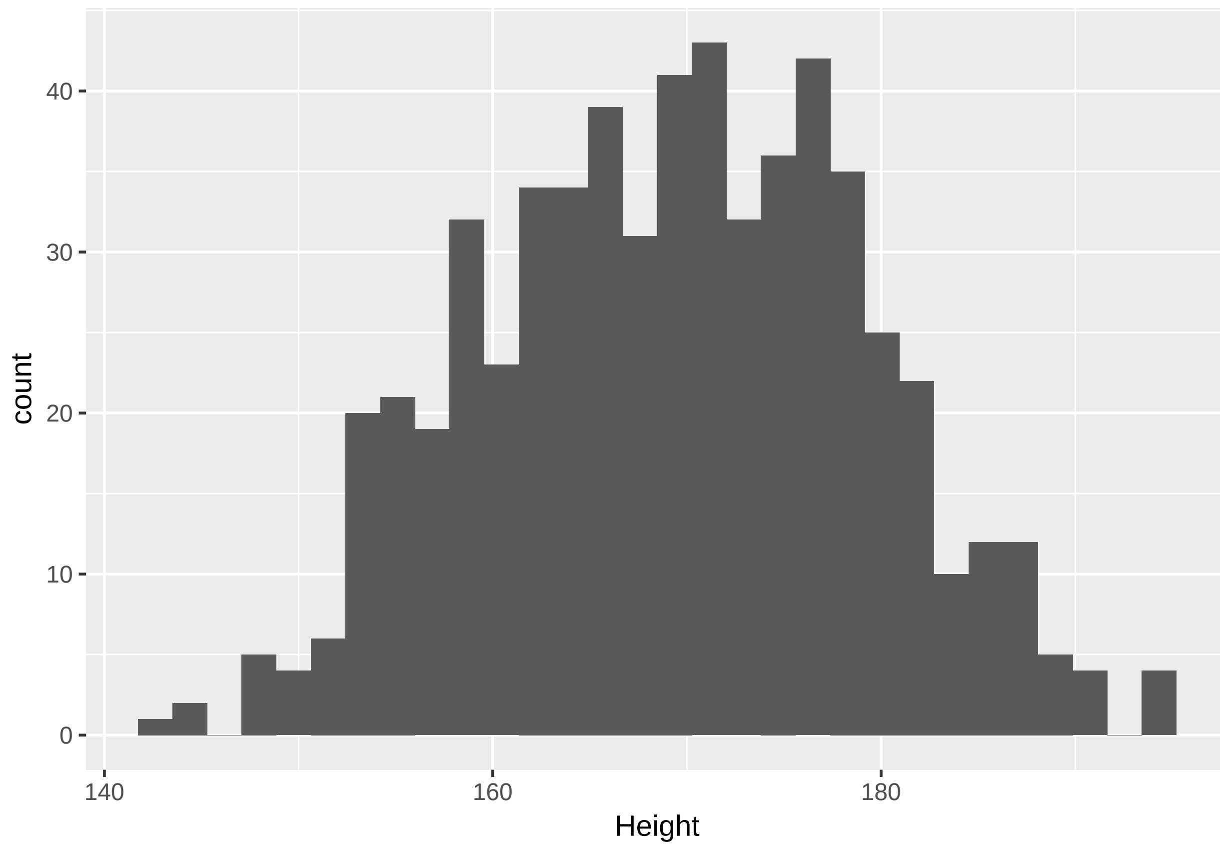
```
10 55391 female    32    161   69.2  26.6   114    Other      Good
# ... with 584 more rows, and 1 more variables: Diabetes <fctr>
```

3.6 Distribution of Heights

What is the distribution of height in this new sample?

```
ggplot(data = nh_data_2179, aes(x = Height)) +
    geom_histogram()
```

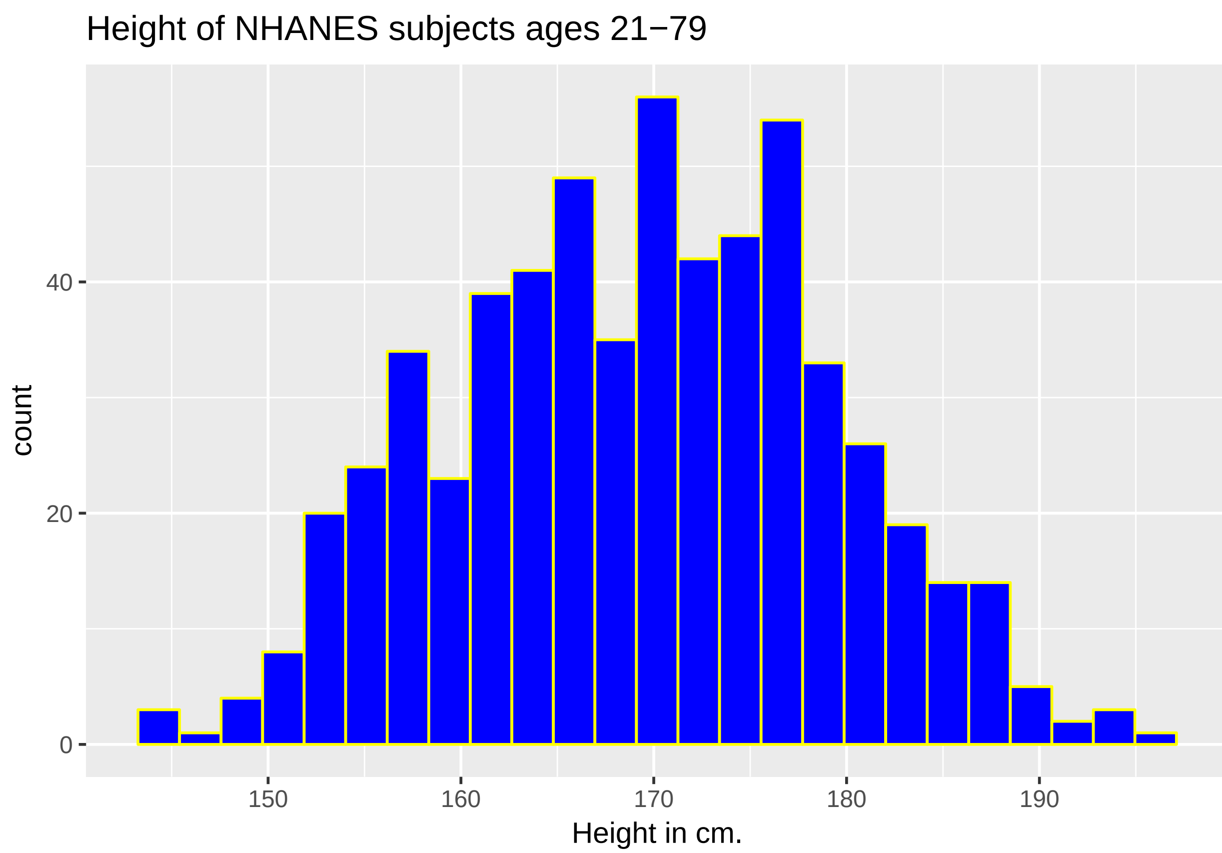
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We can do several things to clean this up.

1. We'll change the color of the lines for each bar of the histogram.
2. We'll change the fill inside each bar to make them stand out a bit more.
3. We'll add a title and relabel the horizontal (x) axis to include the units of measurement.
4. We'll avoid the warning by selecting a number of bins (we'll use 25 here) into which we'll group the heights before drawing the histogram.

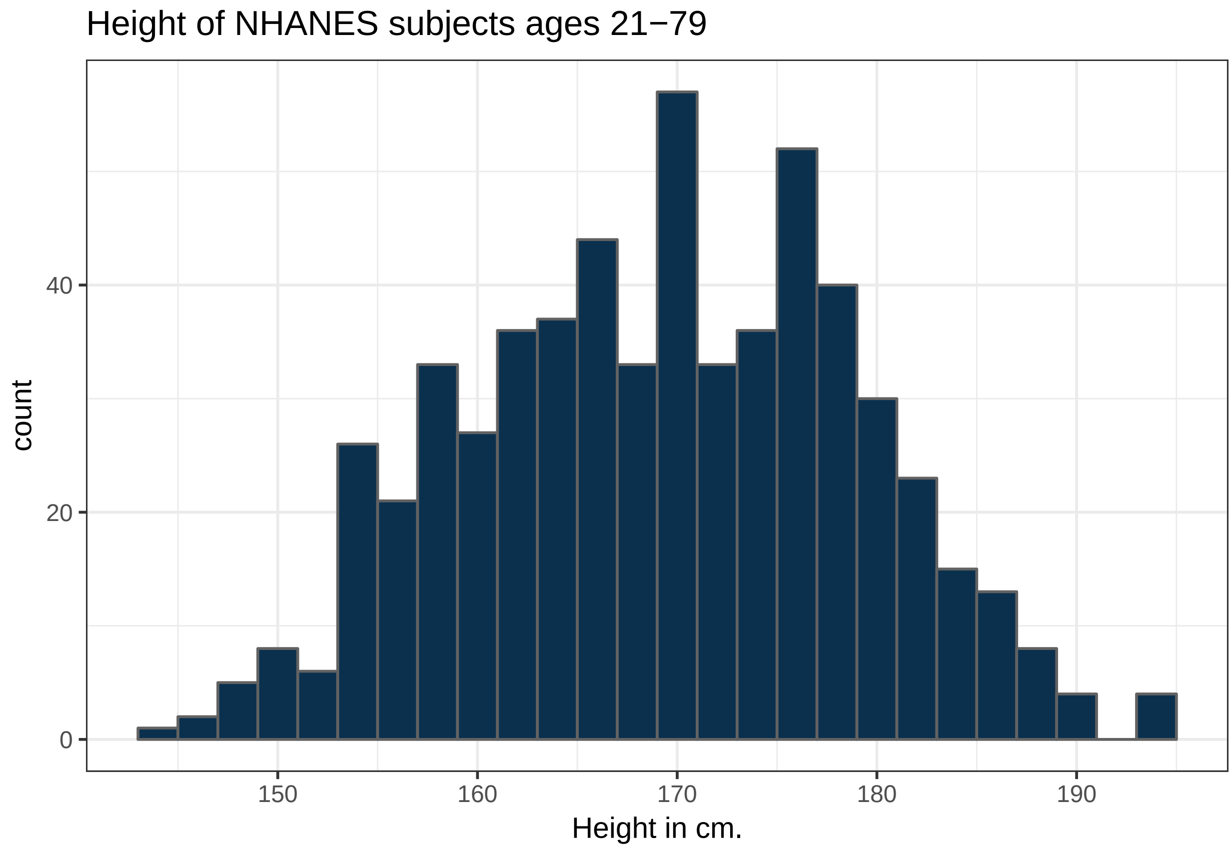
```
ggplot(data = nh_data_2179, aes(x = Height)) +
    geom_histogram(bins = 25, col = "yellow", fill = "blue") +
    labs(title = "Height of NHANES subjects ages 21-79",
         x = "Height in cm.")
```



3.6.1 Changing a Histogram's Fill and Color

The CWRU color guide (https://case.edu/umc/our-brand/visual-guidelines/) lists the HTML color schemes for CWRU blue and CWRU gray. Let's match that color scheme.

```
cwru.blue <- '#0a304e'
cwru.gray <- '#626262'

ggplot(data = nh_data_2179, aes(x = Height)) +
    geom_histogram(binwidth = 2, col = cwru.gray, fill = cwru.blue) +
    labs(title = "Height of NHANES subjects ages 21-79",
         x = "Height in cm.") +
    theme_bw()
```

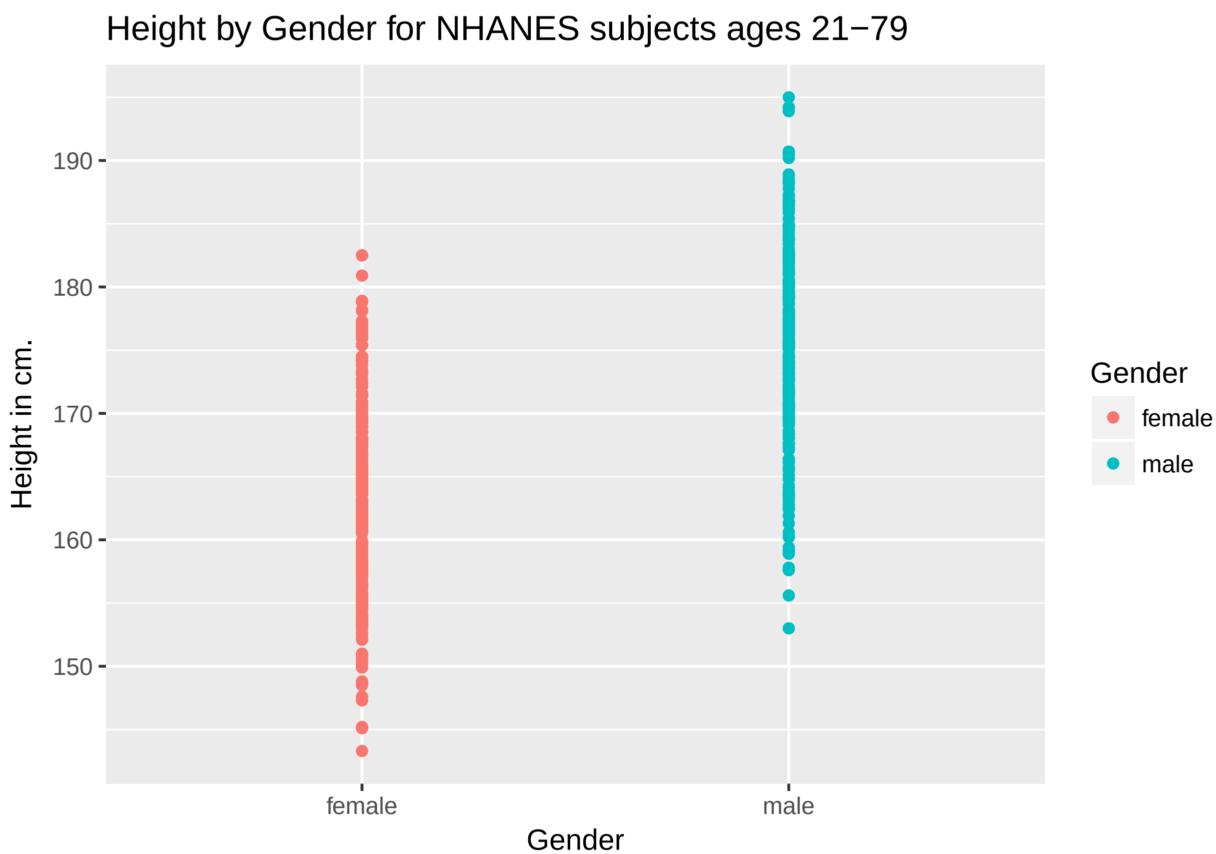


Note the other changes to the graph above.

1. We changed the theme to replace the gray background.
2. We changed the bins for the histogram, to gather observations into groups of 2 cm. each.

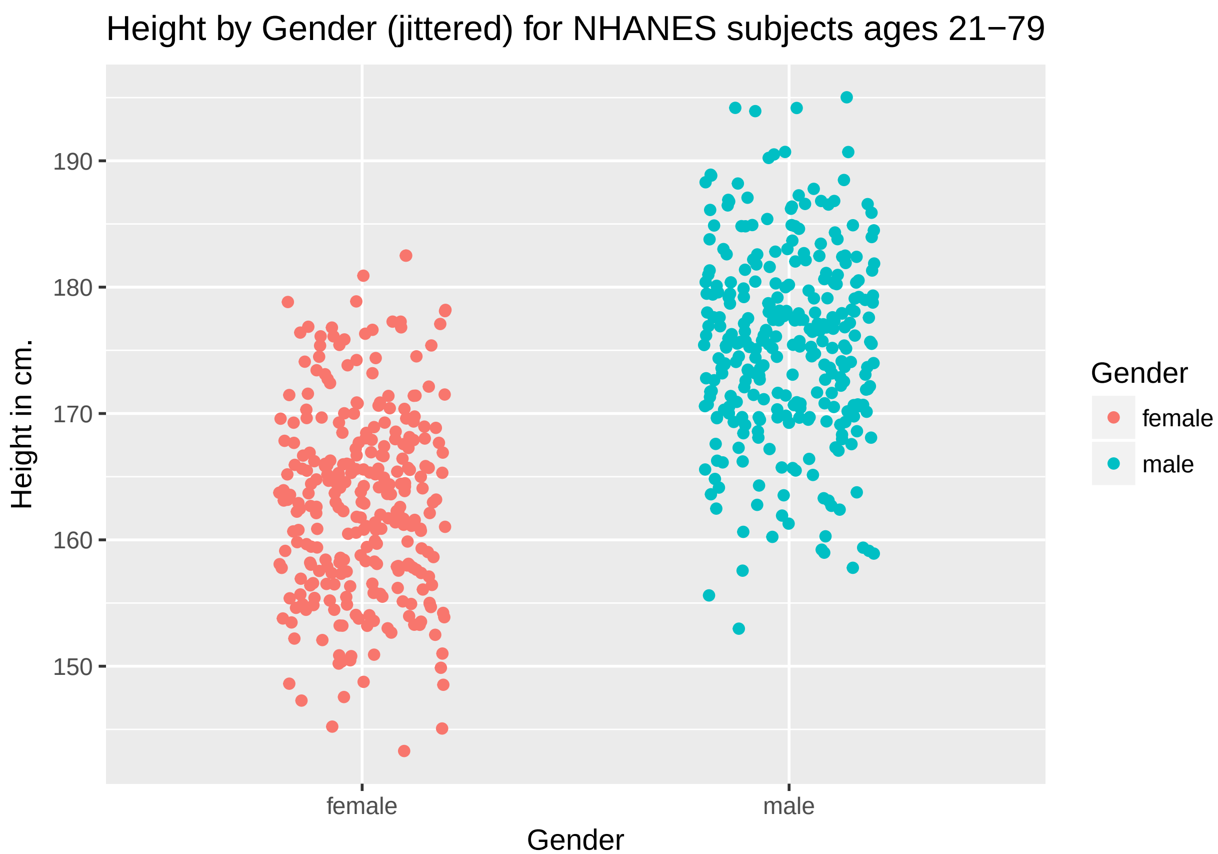
3.7 Height and Gender

```
ggplot(data = nh_data_2179, aes(x = Gender, y = Height, color = Gender)) +
    geom_point() +
    labs(title = "Height by Gender for NHANES subjects ages 21-79",
         y = "Height in cm.")
```



This plot isn't so useful. We can improve things a little by jittering the points horizontally, so that the overlap is reduced.

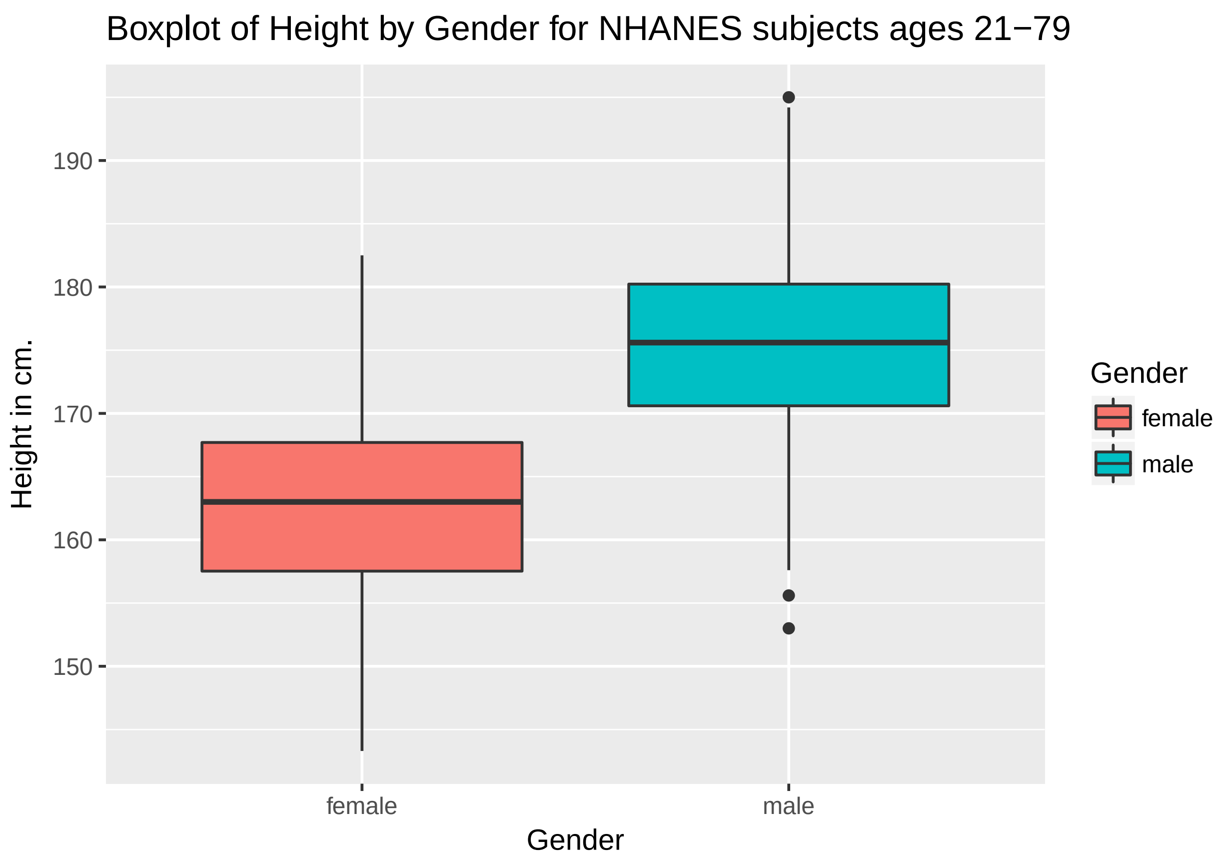
```
ggplot(data = nh_data_2179, aes(x = Gender, y = Height, color = Gender)) +
    geom_jitter(width = 0.2) +
    labs(title = "Height by Gender (jittered) for NHANES subjects ages 21-79",
         y = "Height in cm.")
```



Perhaps it might be better to summarise the distribution in a different way. We might consider a boxplot of the data.

3.7.1 A Boxplot of Height by Gender

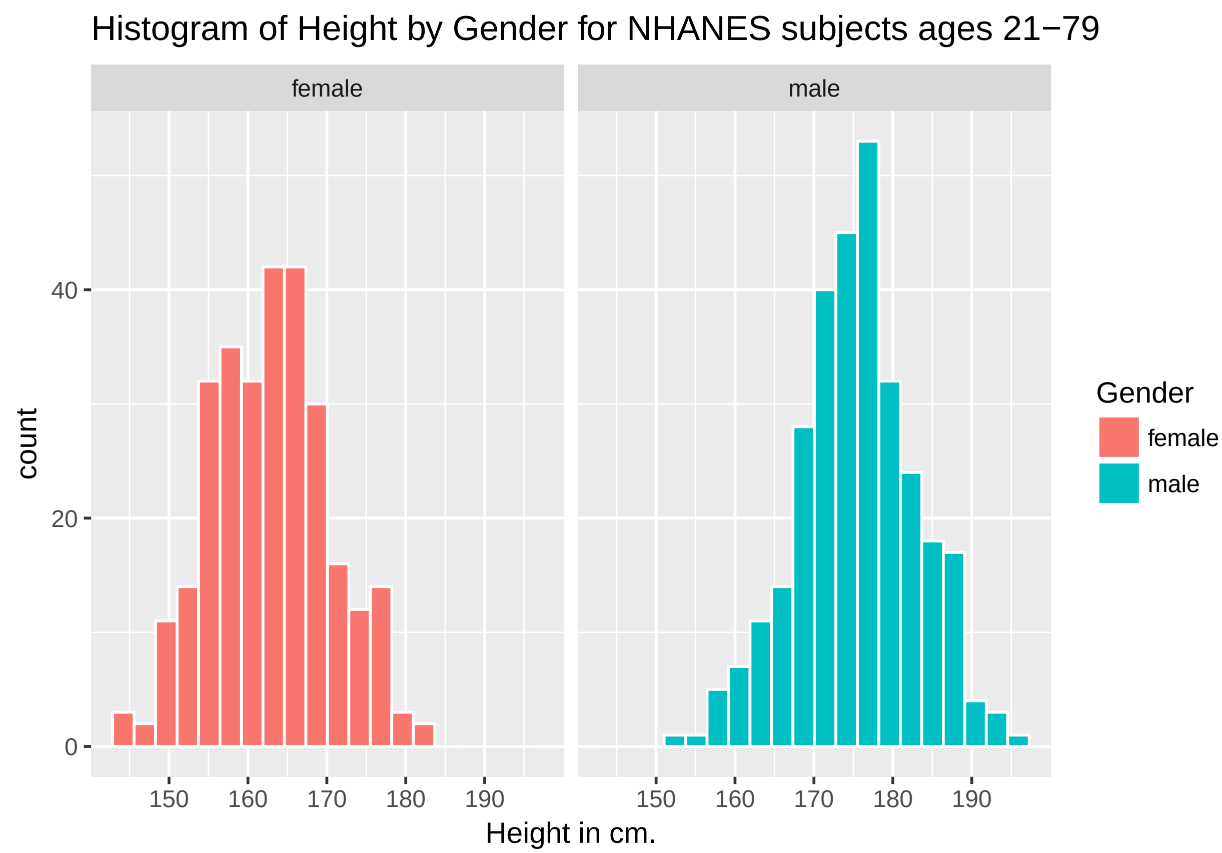
```
ggplot(data = nh_data_2179, aes(x = Gender, y = Height, fill = Gender)) +
    geom_boxplot() +
    labs(title = "Boxplot of Height by Gender for NHANES subjects ages 21-79",
         y = "Height in cm.")
```



Or perhaps we'd like to see a pair of histograms?

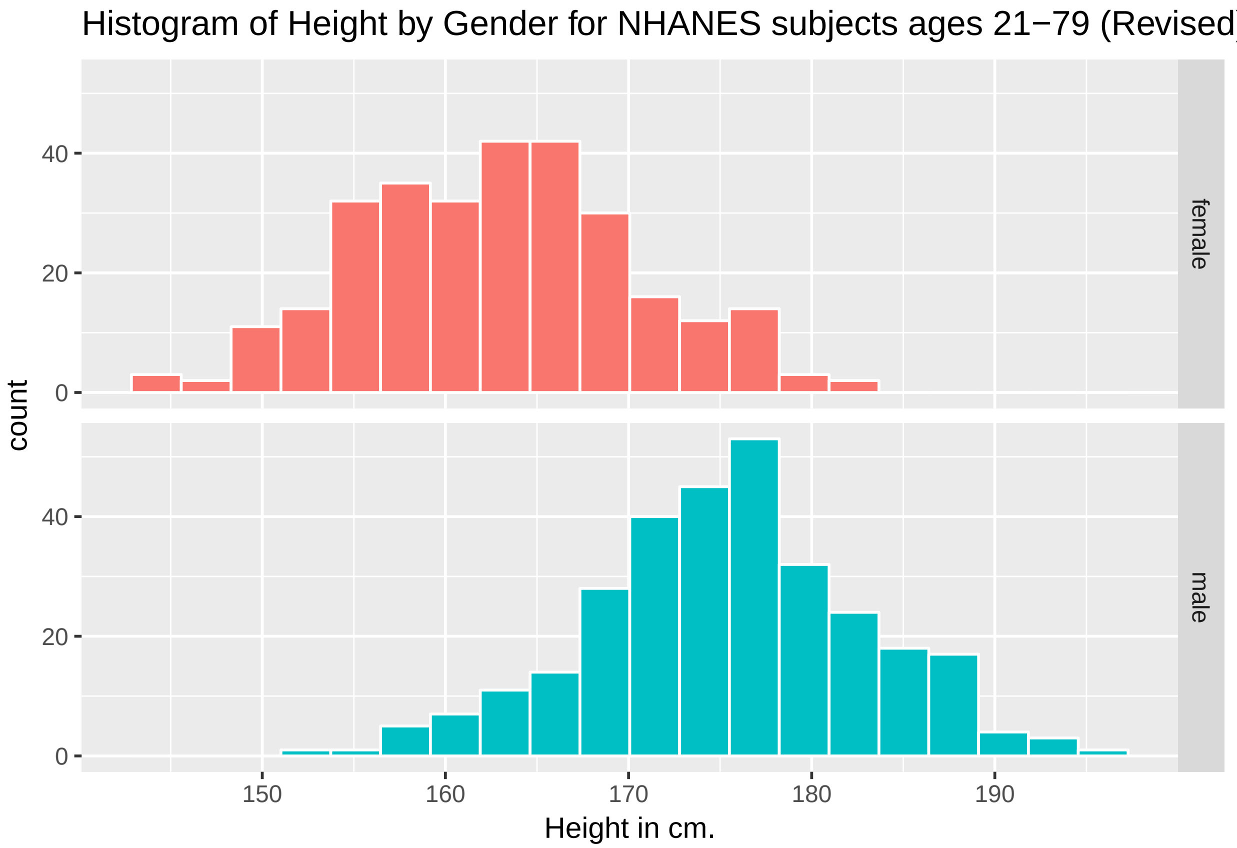
3.7.2 Histograms of Height by Gender

```
ggplot(data = nh_data_2179, aes(x = Height, fill = Gender)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of Height by Gender for NHANES subjects ages 21-79",
         x = "Height in cm.") +
    facet_wrap(~ Gender)
```



Can we redraw these histograms so that they are a little more comparable, and to get rid of the unnecessary legend?

```
ggplot(data = nh_data_2179, aes(x = Height, fill = Gender)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of Height by Gender for NHANES subjects ages 21-79 (Revised)",
         x = "Height in cm.") +
    guides(fill = FALSE) +
    facet_grid(Gender ~ .)
```



3.8 A Look at Body-Mass Index

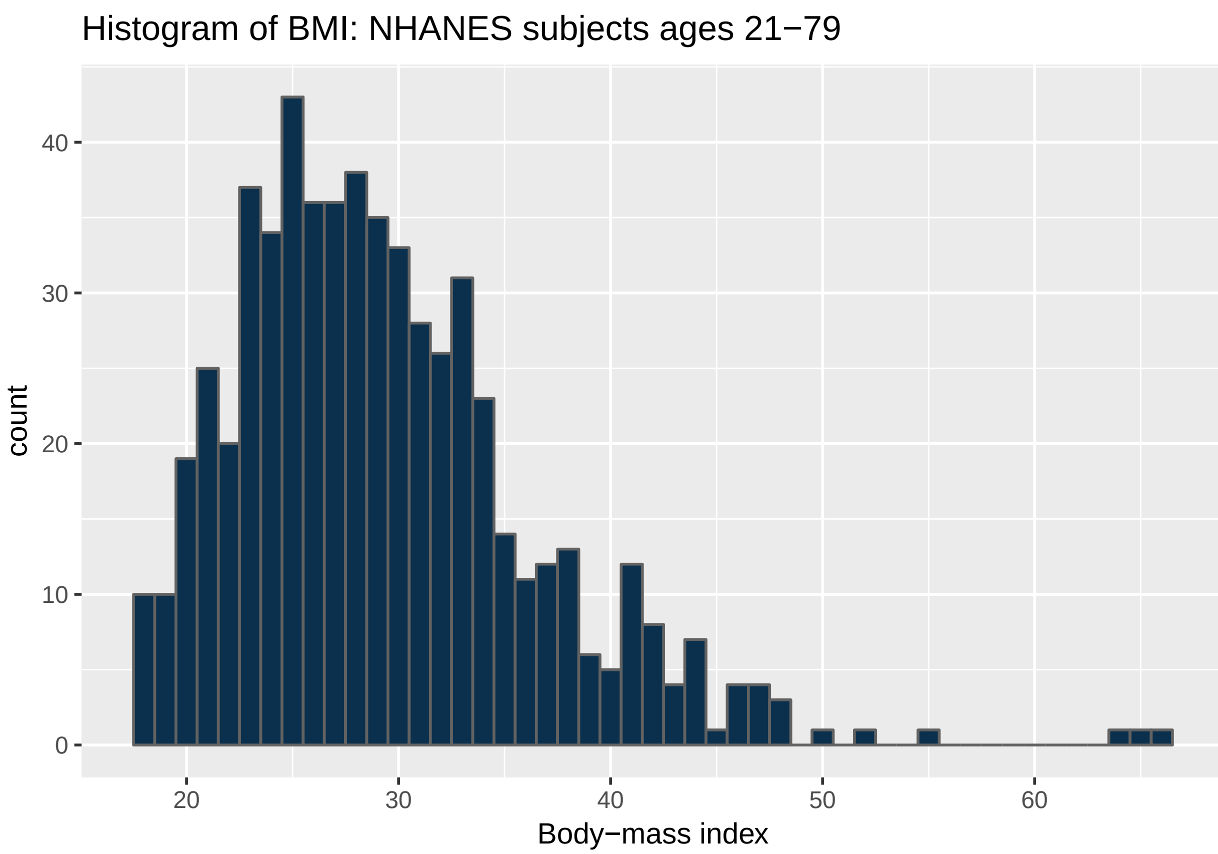
Let's look at a different outcome, the *body-mass index*, or BMI. The definition of BMI for adult subjects (which is expressed in units of kg/m^2) is:

$$\text{Body Mass Index} = \frac{\text{weight in kg}}{(\text{height in meters})^2} = 703 \times \frac{\text{weight in pounds}}{(\text{height in inches})^2}$$

> [BMI is essentially] ... a measure of a person's *thinness* or *thickness*... BMI was designed for use as a simple means of classifying average sedentary (physically inactive) populations, with an average body composition. For these individuals, the current value recommendations are as follow: a BMI from 18.5 up to 25 may indicate optimal weight, a BMI lower than 18.5 suggests the person is underweight, a number from 25 up to 30 may indicate the person is overweight, and a number from 30 upwards suggests the person is obese.
>
> Wikipedia, https://en.wikipedia.org/wiki/Body_mass_index

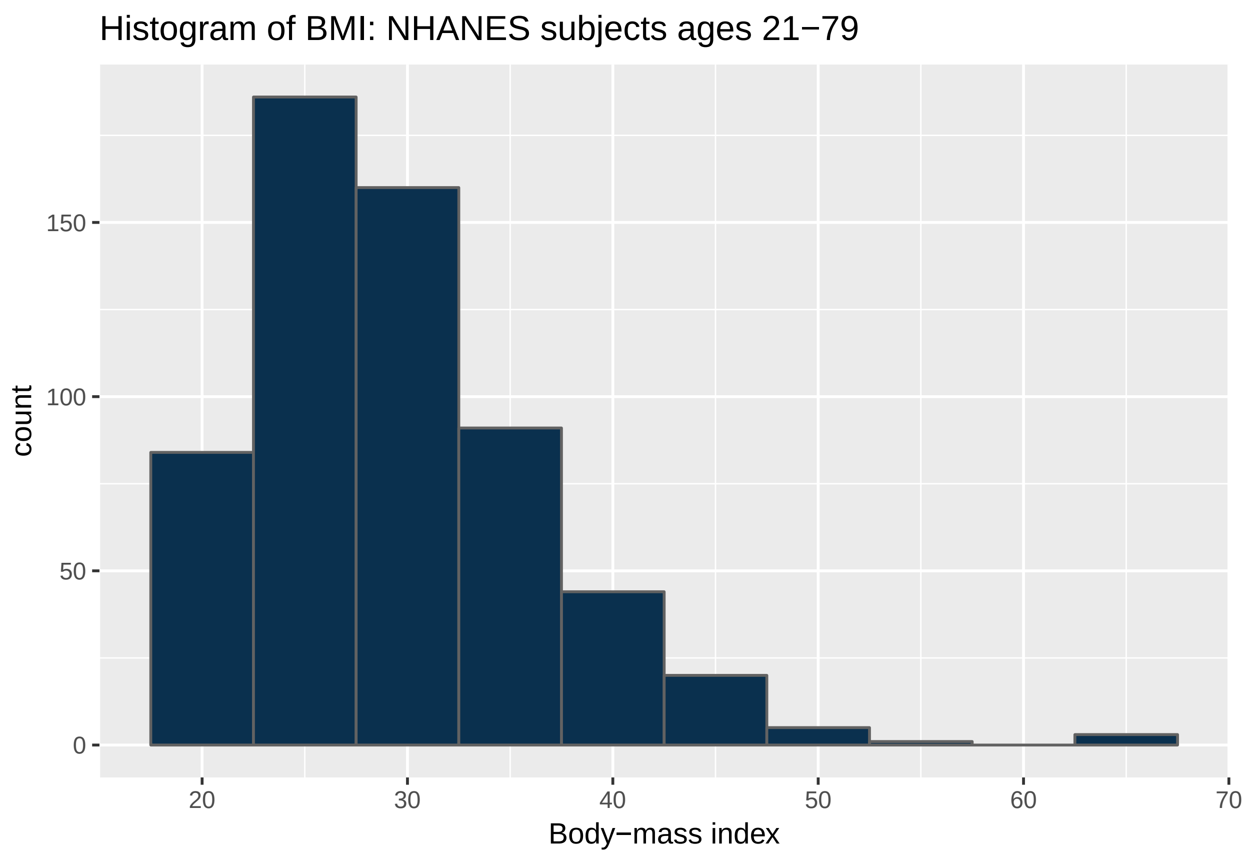
Here's a histogram, again with CWRU colors, for the BMI data.

```
ggplot(data = nh_data_2179, aes(x = BMI)) +
    geom_histogram(binwidth = 1, fill = cwru.blue, col = cwru.gray) +
    labs(title = "Histogram of BMI: NHANES subjects ages 21-79",
         x = "Body-mass index")
```



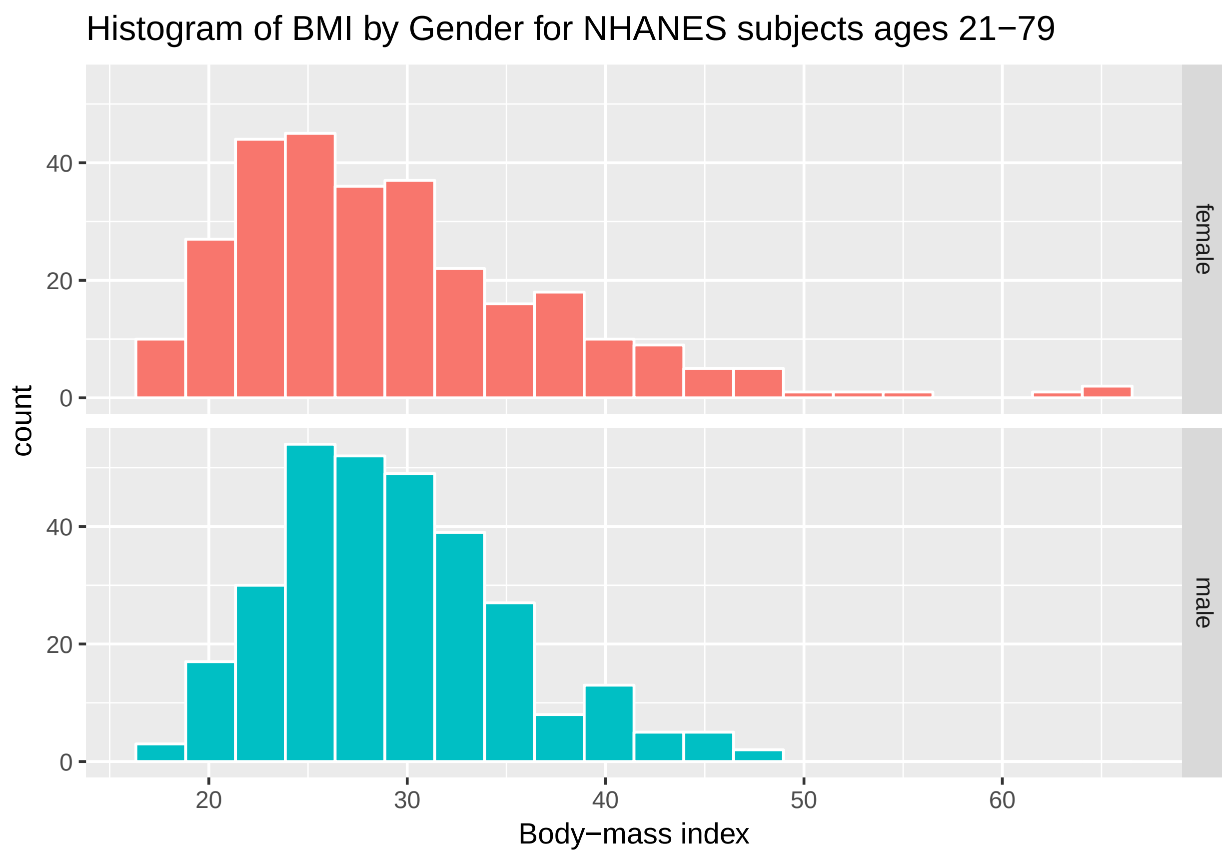
Note how different this picture looks if instead we bin up groups of 5 kg/m^2 at a time. Which is the more useful representation will depend a lot on what questions you're trying to answer.

```
ggplot(data = nh_data_2179, aes(x = BMI)) +
    geom_histogram(binwidth = 5, fill = cwru.blue, col = cwru.gray) +
    labs(title = "Histogram of BMI: NHANES subjects ages 21-79",
         x = "Body-mass index")
```



3.8.1 BMI by Gender

```
ggplot(data = nh_data_2179, aes(x = BMI, fill = Gender)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of BMI by Gender for NHANES subjects ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE) +
    facet_grid(Gender ~ .)
```



As an accompanying numerical summary, we might ask how many people fall into each of these Gender categories, and what is their "average" BMI.

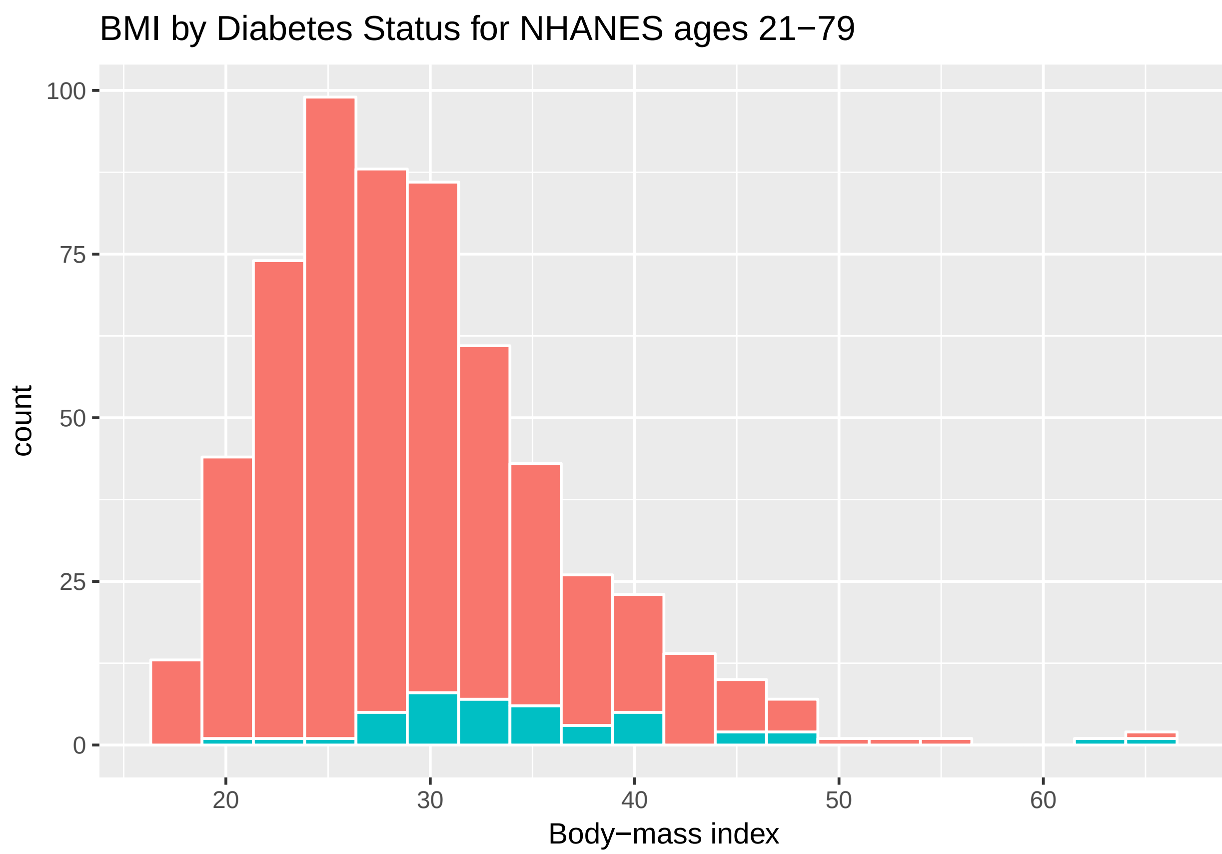
```
nh_data_2179 %>%
    group_by(Gender) %>%
    summarise(count = n(), mean(BMI), median(BMI)) %>%
    knitr::kable()
```

Gender	count	mean(BMI)	median(BMI)
female	290	29.4	27.4
male	304	29.4	28.7

3.8.2 BMI and Diabetes

We can split up our histogram into groups based on whether the subjects have been told they have diabetes.

```
ggplot(data = nh_data_2179, aes(x = BMI, fill = Diabetes)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "BMI by Diabetes Status for NHANES ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE)
```



How many people fall into each of these Diabetes categories, and what is their "average" BMI?

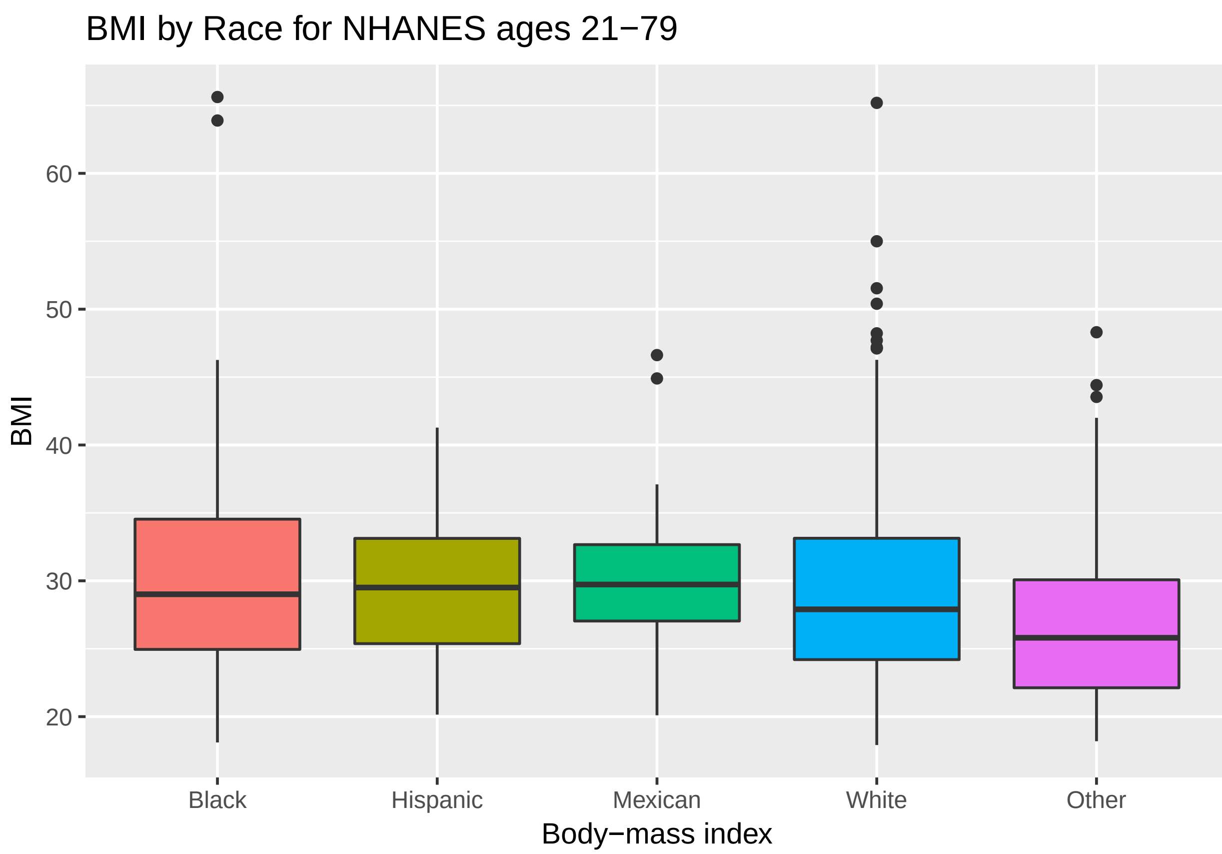
```
nh_data_2179 %>%
    group_by(Diabetes) %>%
    summarise(count = n(), mean(BMI), median(BMI)) %>%
    knitr::kable()
```

Diabetes	count	mean(BMI)	median(BMI)
No	551	28.9	27.9
Yes	43	35.3	33.4

3.8.3 BMI and Race

We can compare the distribution of BMI across Race groups, as well.

```
ggplot(data = nh_data_2179, aes(x = Race1, y = BMI, fill = Race1)) +
    geom_boxplot() +
    labs(title = "BMI by Race for NHANES ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE)
```



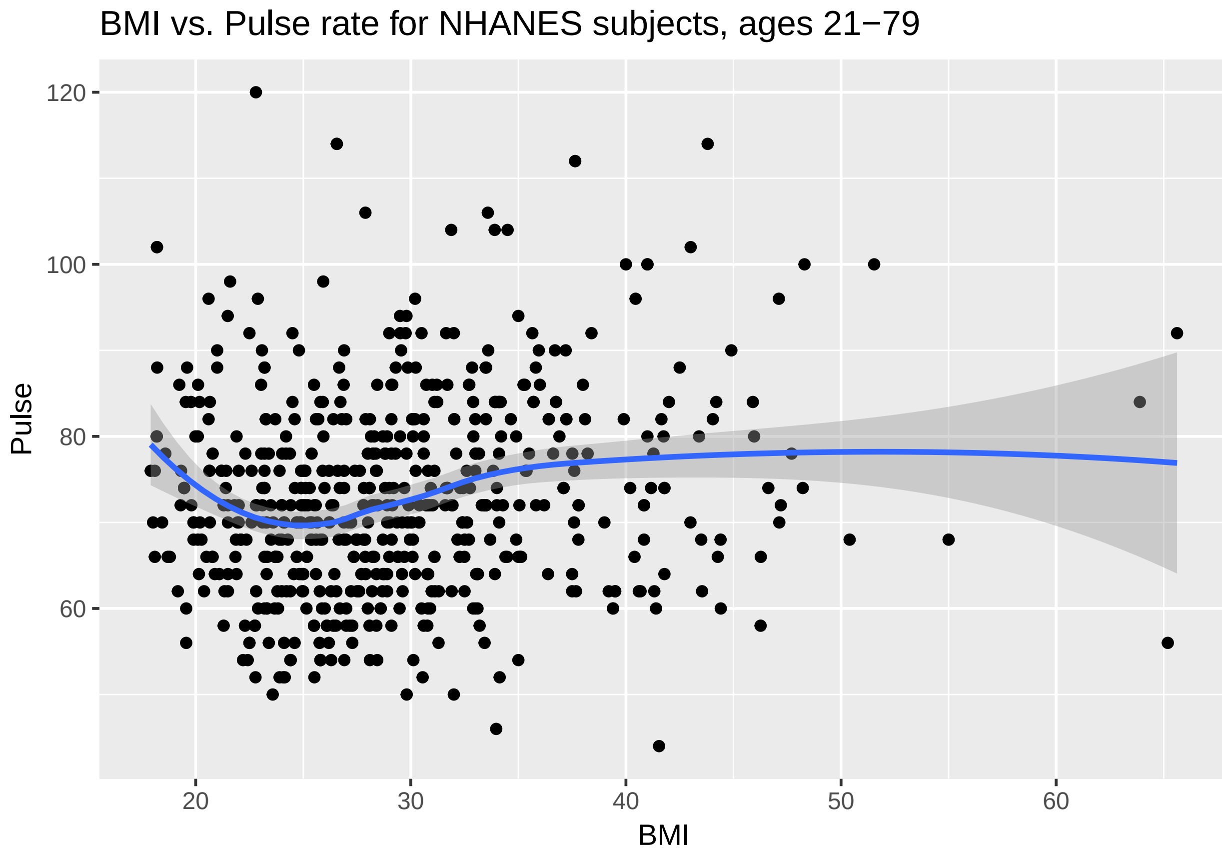
How many people fall into each of these Race1 categories, and what is their "average" BMI?

```
library(tidyverse)
nh_data_2179 %>%
    group_by(Race1) %>%
    summarise(count = n(), mean(BMI), median(BMI)) %>%
    knitr::kable()
```

Race1	count	mean(BMI)	median(BMI)
Black	63	31.0	29.0
Hispanic	44	29.4	29.5
Mexican	50	30.0	29.7
White	387	29.3	27.9
Other	50	27.3	25.8

3.8.4 BMI and Pulse Rate

```
ggplot(data = nh_data_2179, aes(x = BMI, y = Pulse)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "BMI vs. Pulse rate for NHANES subjects, ages 21-79")
```

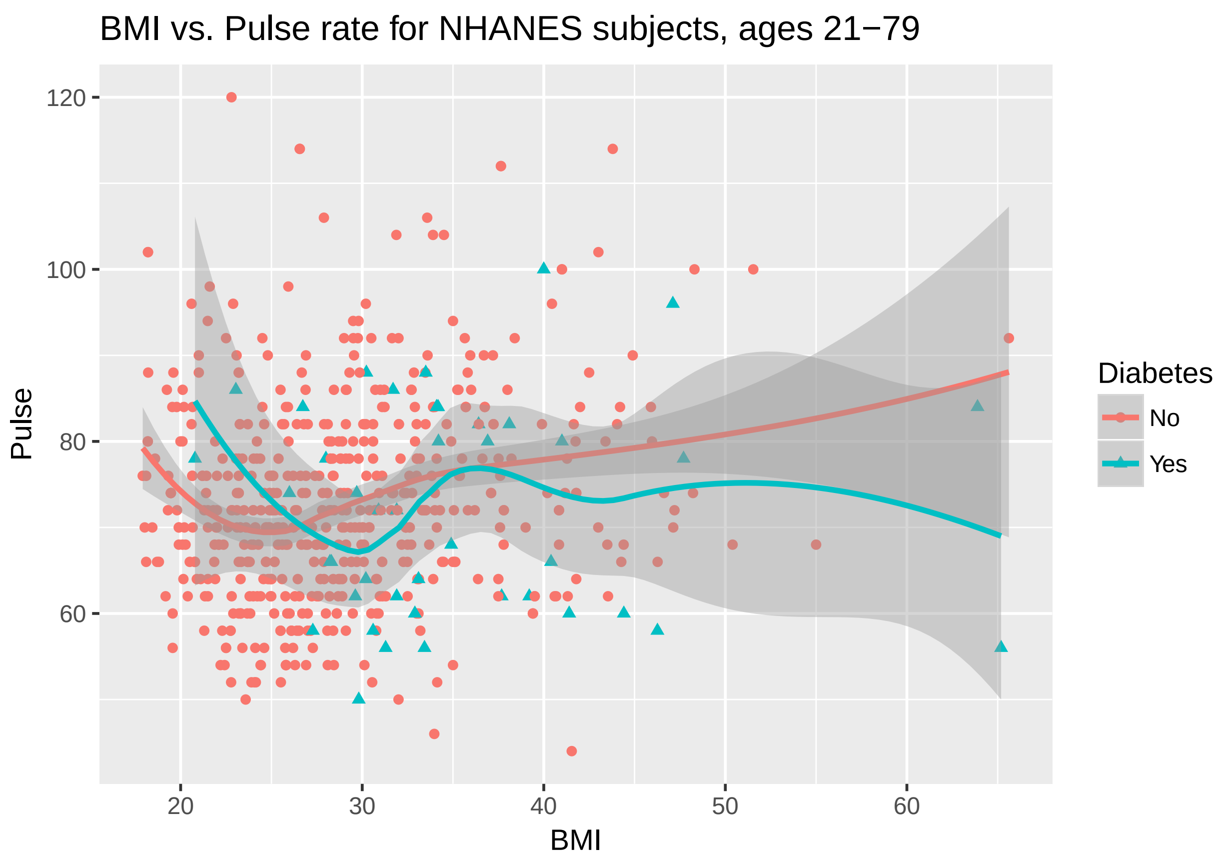


3.8.5 Diabetes vs. No Diabetes

Could we see whether subjects who have been told they have diabetes show different BMI-pulse rate patterns than the subjects who haven't?

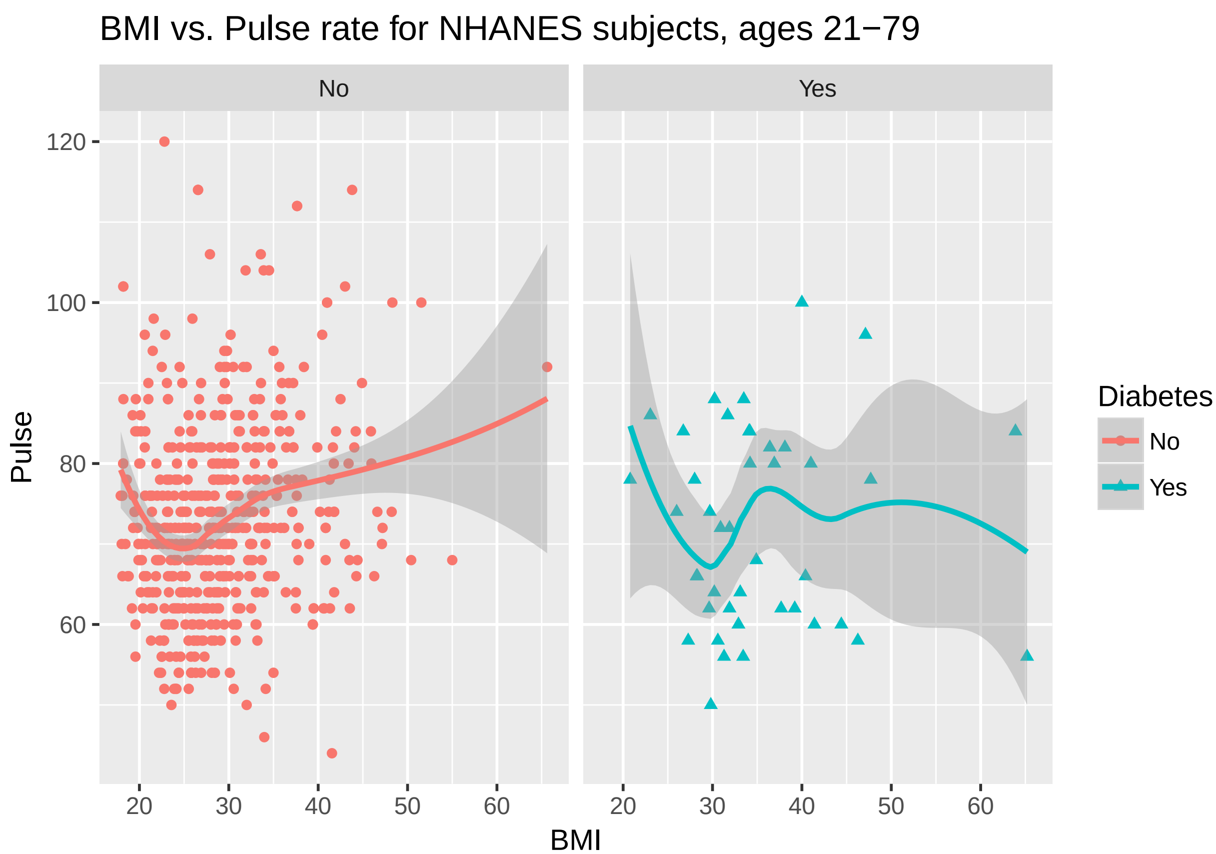
- Let's try doing this by changing the shape *and* the color of the points based on diabetes status.

```
ggplot(data = nh_data_2179,
       aes(x = BMI, y = Pulse,
           color = Diabetes, shape = Diabetes)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "BMI vs. Pulse rate for NHANES subjects, ages 21-79")
```



This plot might be easier to interpret if we faceted by Diabetes status, as well.

```
ggplot(data = nh_data_2179,
       aes(x = BMI, y = Pulse,
           color = Diabetes, shape = Diabetes)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "BMI vs. Pulse rate for NHANES subjects, ages 21-79") +
    facet_wrap(~ Diabetes)
```



3.9 General Health Status

Here's a Table of the General Health Status results. This is a self-reported rating of each subject's health on a five point scale (Excellent, Very Good, Good, Fair, Poor.)

```
nh_data_2179 %>%
    select(HealthGen) %>%
    table()
```

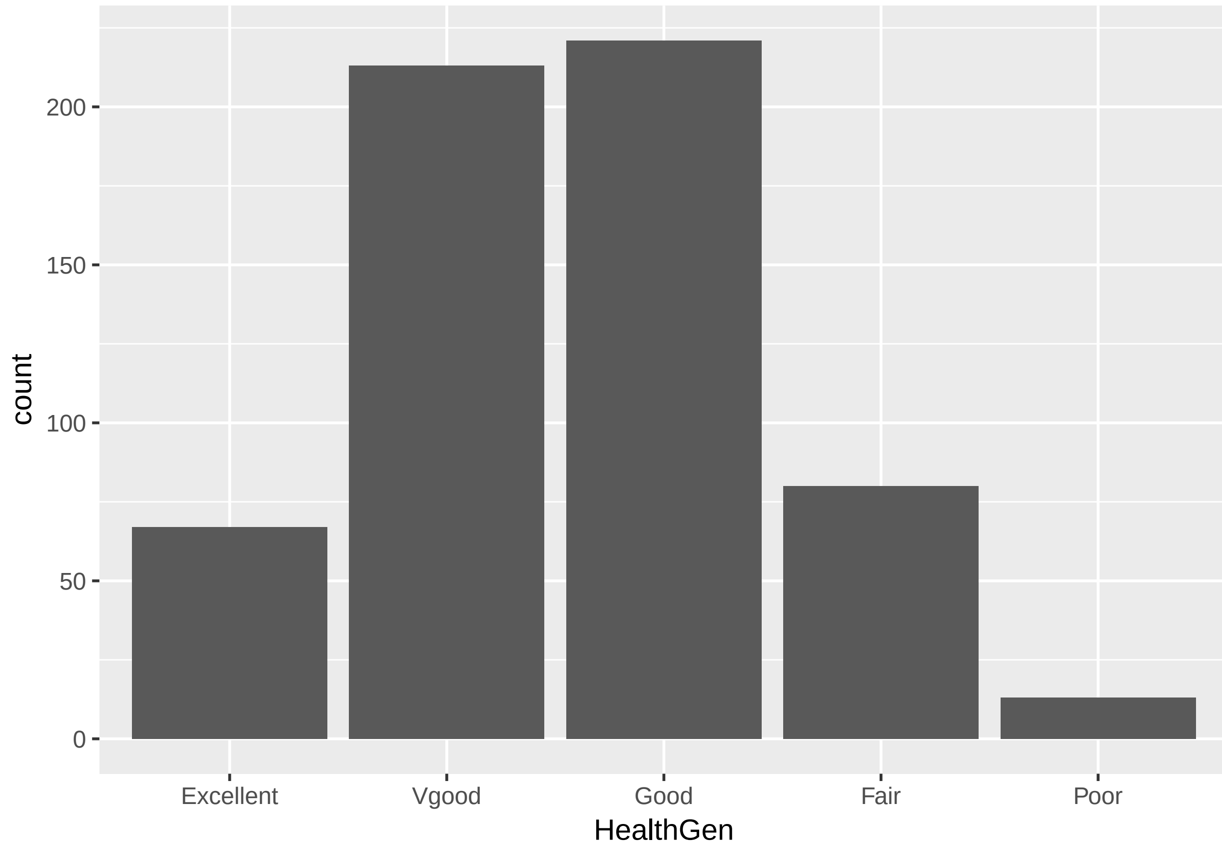
```
.
Excellent     Vgood      Good      Fair      Poor
       67       213       221        80        13
```

The HealthGen data are categorical, which means that summarizing them with averages isn't as appealing as looking at percentages, proportions and rates.

3.9.1 Bar Chart for Categorical Data

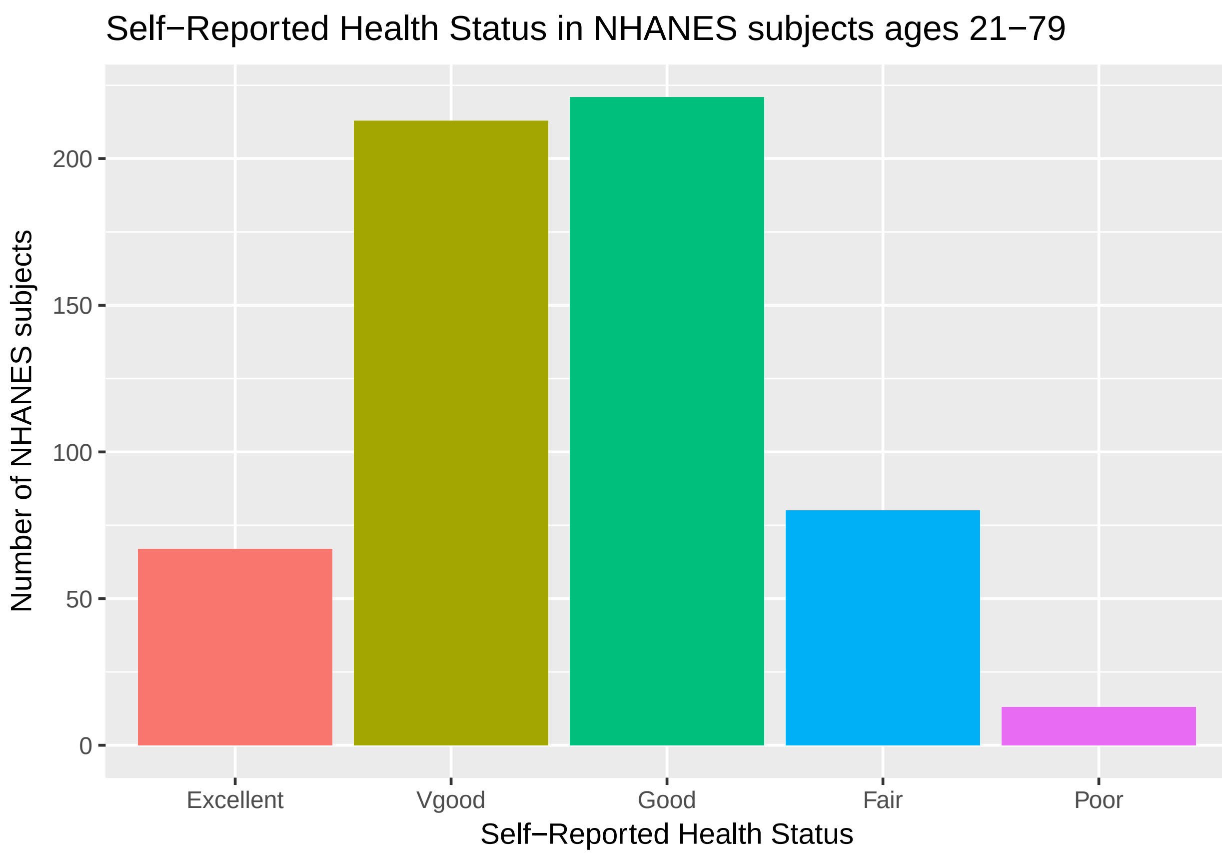
Usually, a **bar chart** is the best choice for a graphing a variable made up of categories.

```
ggplot(data = nh_data_2179, aes(x = HealthGen)) +
    geom_bar()
```



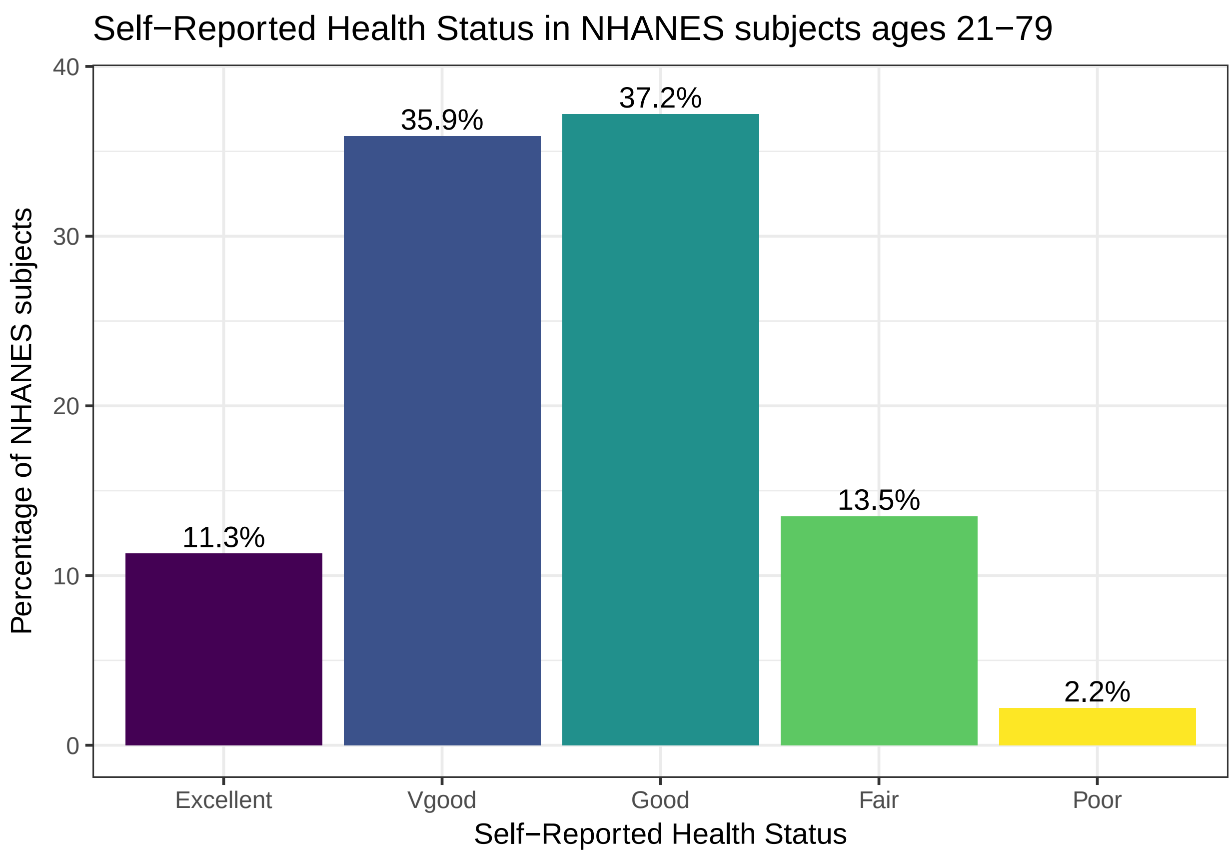
There are lots of things we can do to make this plot fancier.

```
ggplot(data = nh_data_2179, aes(x = HealthGen, fill = HealthGen)) +
    geom_bar() +
    guides(fill = FALSE) +
    labs(x = "Self-Reported Health Status",
         y = "Number of NHANES subjects",
         title = "Self-Reported Health Status in NHANES subjects ages 21-79")
```



Or, we can really go crazy...

```
nh_data_2179 %>%
    count(HealthGen) %>%
    ungroup() %>%
    mutate(pct = round(prop.table(n) * 100, 1)) %>%
    ggplot(aes(x = HealthGen, y = pct, fill = HealthGen)) +
    geom_bar(stat = "identity", position = "dodge") +
    scale_fill_viridis(discrete = TRUE) +
    guides(fill = FALSE) +
    geom_text(aes(y = pct + 1,     # nudge above top of bar
                  label = paste0(pct, '%')),  # prettify
              position = position_dodge(width = .9),
              size = 4) +
    labs(x = "Self-Reported Health Status",
         y = "Percentage of NHANES subjects",
         title = "Self-Reported Health Status in NHANES subjects ages 21-79") +
    theme_bw()
```



3.9.2 Working with Tables

We can add a marginal total, and compare subjects by Gender, as follows...

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    addmargins()
```

```
        HealthGen
Gender   Excellent Vgood Good Fair Poor Sum
  female        34   107  107   34    8 290
  male          33   106  114   46    5 304
  Sum           67   213  221   80   13 594
```

If we like, we can make this look a little more polished with the `knitr::kable` function...

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    addmargins() %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor	Sum
female	34	107	107	34	8	290
male	33	106	114	46	5	304
Sum	67	213	221	80	13	594

If we want the proportions of patients within each Gender that fall in each HealthGen category (the row percentages), we can get them, too.

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    prop.table(.,1) %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor
female	0.117	0.369	0.369	0.117	0.028
male	0.109	0.349	0.375	0.151	0.016

To make this a little easier to use, we might consider rounding.

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    prop.table(.,1) %>%
    round(.,2) %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor
female	0.12	0.37	0.37	0.12	0.03
male	0.11	0.35	0.38	0.15	0.02

Another possibility would be to show the percentages, rather than the proportions (which requires multiplying the proportion by 100.) Note the strange “*" function, which is needed to convince R to multiply each entry by 100 here.

```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    prop.table(.,1) %>%
    "*"(100) %>%
    round(.,2)  %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor
female	11.7	36.9	36.9	11.7	2.76
male	10.9	34.9	37.5	15.1	1.64

And, if we wanted the column percentages, to determine which gender had the higher rate of each HealthGen status level, we can get that by changing the prop.table to calculate 2 (column) proportions, rather than 1 (rows.)

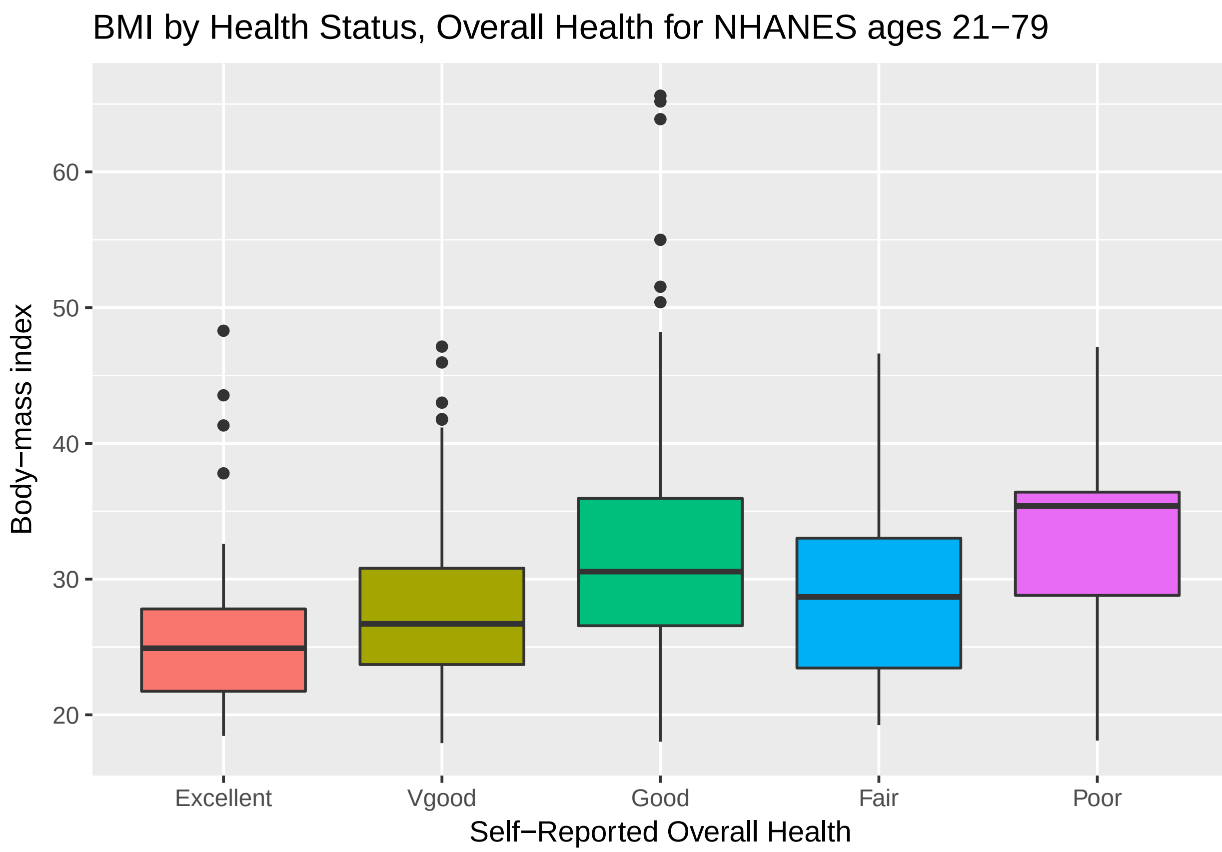
```
nh_data_2179 %>%
    select(Gender, HealthGen) %>%
    table() %>%
    prop.table(.,2) %>%
    "*"(100) %>%
    round(.,2)  %>%
    knitr::kable()
```

	Excellent	Vgood	Good	Fair	Poor
female	50.8	50.2	48.4	42.5	61.5
male	49.2	49.8	51.6	57.5	38.5

3.9.3 BMI by General Health Status

Let's consider now the relationship between self-reported overall health and body-mass index.

```r
ggplot(data = nh_data_2179, aes(x = HealthGen, y = BMI, fill = HealthGen)) +
    geom_boxplot() +
    labs(title = "BMI by Health Status, Overall Health for NHANES ages 21-79",
         y = "Body-mass index", x = "Self-Reported Overall Health") +
    guides(fill = FALSE)
```



We can see that not too many people self-identify with the "Poor" health category.

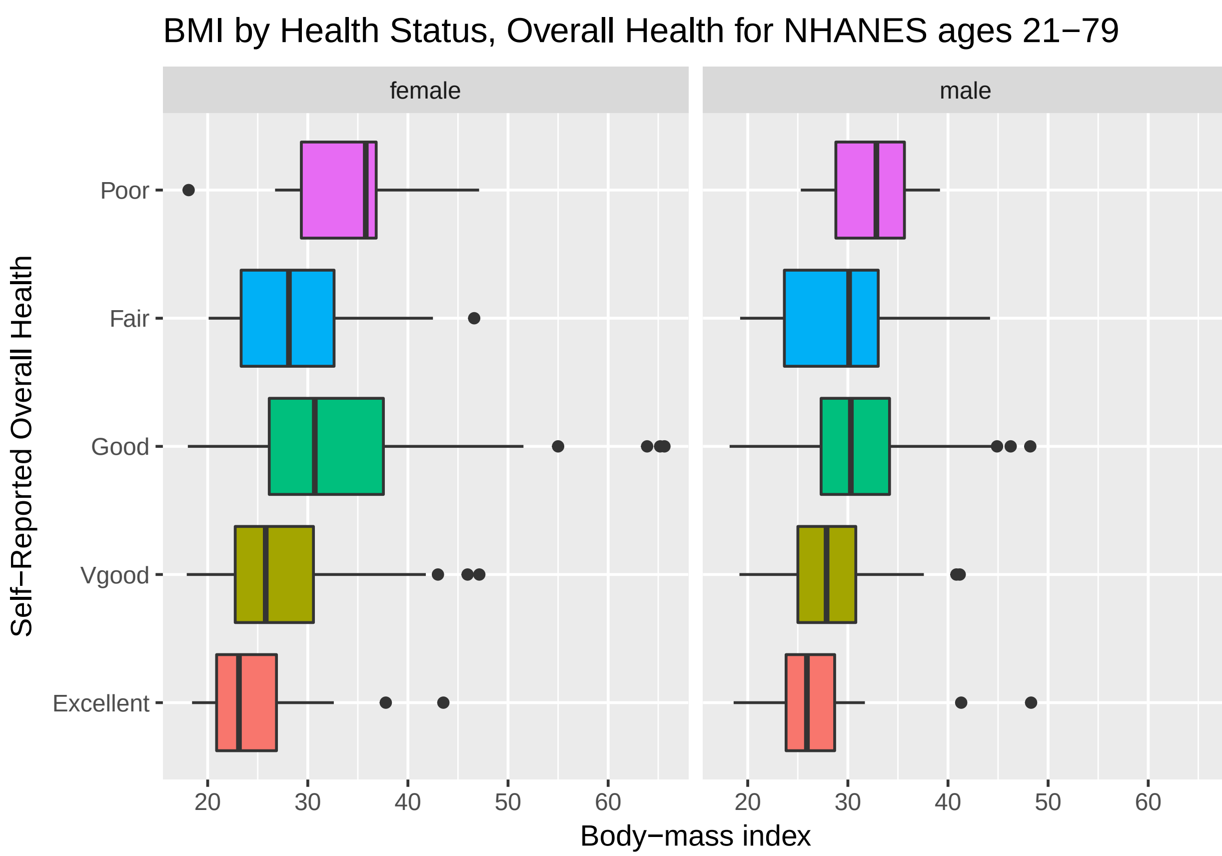
```r
nh_data_2179 %>%
    group_by(HealthGen) %>%
    summarise(count = n(), mean(BMI), median(BMI)) %>%
    knitr::kable()
```

HealthGen	count	mean(BMI)	median(BMI)
Excellent	67	25.7	24.9
Vgood	213	27.6	26.7
Good	221	32.0	30.6
Fair	80	29.3	28.7
Poor	13	33.1	35.4

3.9.4 BMI by Gender and General Health Status

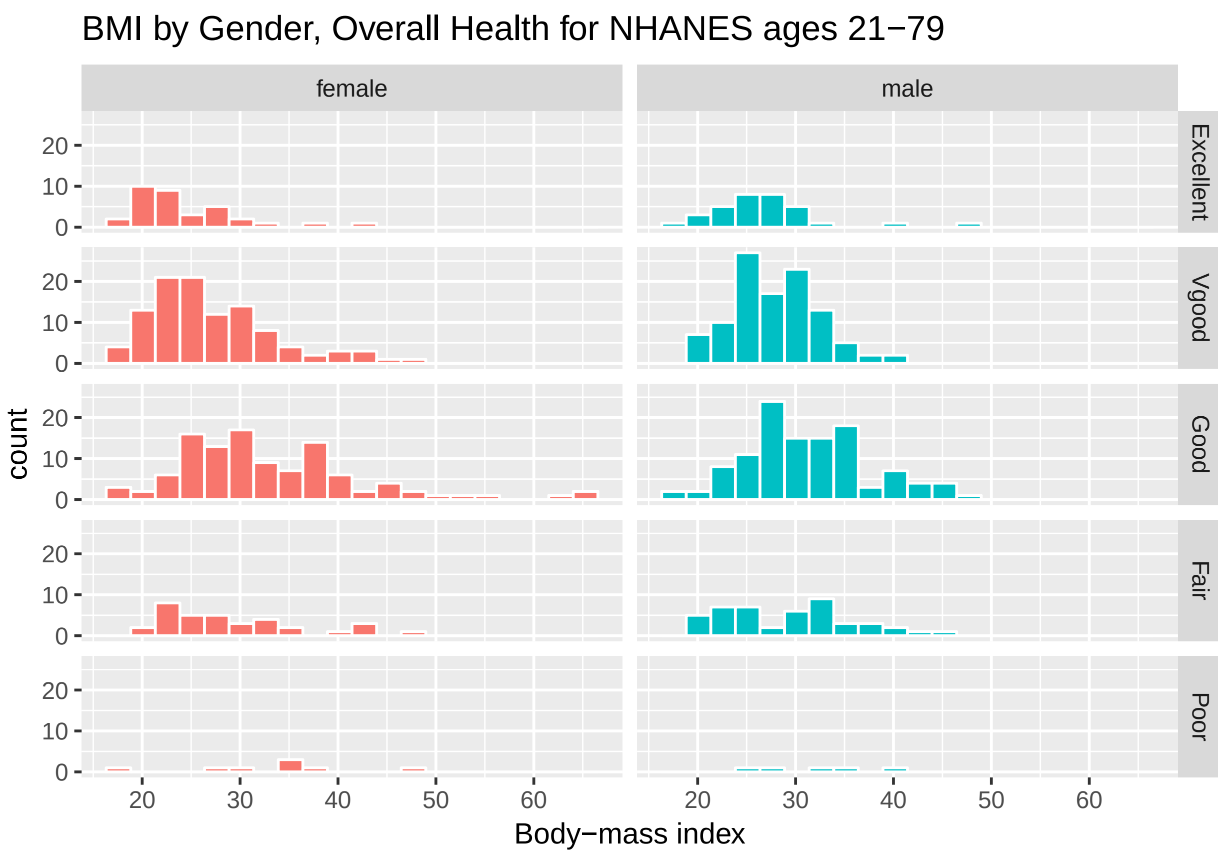
We'll start with two panels of boxplots to try to understand the relationships between BMI, General Health Status and Gender. Note the use of `coord_flip` to rotate the graph 90 degrees.

```
ggplot(data = nh_data_2179, aes(x = HealthGen, y = BMI, fill = HealthGen)) +
    geom_boxplot() +
    labs(title = "BMI by Health Status, Overall Health for NHANES ages 21-79",
         y = "Body-mass index", x = "Self-Reported Overall Health") +
    guides(fill = FALSE) +
    facet_wrap(~ Gender) +
    coord_flip()
```



Here's a plot of faceted histograms, which might be used to address similar questions.

```
ggplot(data = nh_data_2179, aes(x = BMI, fill = Gender)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "BMI by Gender, Overall Health for NHANES ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE) +
    facet_grid(HealthGen ~ Gender)
```



3.10 Conclusions

This is just a small piece of the toolbox for visualizations that we'll create in this class. Many additional tools are on the way, but the main idea won't change. Using the `ggplot2` package, we can accomplish several critical tasks in creating a visualization, including:

- Identifying (and labeling) the axes and titles
- Identifying a type of `geom` to use, like a point, bar or histogram
- Changing fill, color, shape, size to facilitate comparisons
- Building "small multiples" of plots with faceting

Good data visualizations make it easy to see the data, and `ggplot2`'s tools make it relatively difficult to make a really bad graph.

Chapter 4

Data Structures and Types of Variables

4.1 Data require structure and context

Descriptive statistics are concerned with the presentation, organization and summary of data, as suggested in Norman and Streiner (2014). This includes various methods of organizing and graphing data to get an idea of what those data can tell us.

As Vittinghoff et al. (2012) suggest, the nature of the measurement determines how best to describe it statistically, and the main distinction is between **numerical** and **categorical** variables. Even this is a little tricky - plenty of data can have values that look like numerical values, but are just numerals serving as labels.

As Bock, Velleman, and De Veaux (2004) point out, the truly critical notion, of course, is that data values, no matter what kind, are useless without their contexts. The Five W's (Who, What [and in what units], When, Where, Why, and often How) are just as useful for establishing the context of data as they are in journalism. If you can't answer Who and What, in particular, you don't have any useful information.

In general, each row of a data frame corresponds to an individual (respondent, experimental unit, record, or observation) about whom some characteristics are gathered in columns (and these characteristics may be called variables, factors or data elements.) Every column / variable should have a name that indicates *what* it is measuring, and every row / observation should have a name that indicates *who* is being measured.

4.2 A New NHANES Adult Sample

In previous work, we spent some time with a sample from the National Health and Nutrition Examination. Now, by changing the value of the `set.seed` function which determines the starting place for the random sampling, and changing some other specifications, we'll generate a new sample describing 500 adult subjects who completed the 2011-12 version of the survey when they were between the ages of 21 and 64.

Note also that what is listed in the NHANES data frame as `Gender` should be more correctly referred to as `sex`. `Sex` is a biological feature of an individual, while `Gender` is a social construct. This is an important distinction, so I'll change the name of the variable. I'm also changing the names of three other variables, to create `Race`, `SBP` and `DBP`.

```
# library(NHANES) # NHANES package/library of functions, data

nh_temp <- NHANES %>%
```

```
    filter(SurveyYr == "2011_12") %>%
    filter(Age >= 21 & Age < 65) %>%
    mutate(Sex = Gender, Race = Race3, SBP = BPSysAve, DBP = BPDiaAve) %>%
    select(ID, Sex, Age, Race, Education, BMI, SBP, DBP, Pulse, PhysActive, Smoke100, SleepTrouble, Hea

set.seed(431002)
# use set.seed to ensure that we all get the same random sample

nh_adults <- sample_n(nh_temp, size = 500)

nh_adults
```

```
# A tibble: 500 x 13
      ID    Sex   Age  Race    Education   BMI   SBP   DBP Pulse
   <int> <fctr> <int> <fctr>      <fctr> <dbl> <int> <int> <int>
 1 64427   male    37  White College Grad  36.5   111    72    56
 2 63788 female    40  White  High School  18.2   115    74   102
 3 66874 female    31  White Some College  27.2    95    52    98
 4 69734   male    26  White College Grad  20.6   137    75    74
 5 70409   male    44  White  High School  29.2   112    71    62
 6 68961 female    64  White College Grad  24.2   123    70    80
 7 62616 female    37  Asian    8th Grade  19.3   109    73    82
 8 70130   male    42  Black  High School  31.2   119    71    62
 9 71218   male    33  White College Grad  27.7   110    67    68
10 69181 female    37  White    8th Grade  25.0   114    74    82
# ... with 490 more rows, and 4 more variables: PhysActive <fctr>,
#   Smoke100 <fctr>, SleepTrouble <fctr>, HealthGen <fctr>
```

The data consist of 500 rows (observations) on 13 variables (columns). Essentially, we have 13 pieces of information on each of 500 adult NHANES subjects who were included in the 2011-12 panel.

4.2.1 Summarizing the Data's Structure

We can identify the number of rows and columns in a data frame or tibble with the `dim` function.

```
dim(nh_adults)
```

```
[1] 500  13
```

The `str` function provides a lot of information about the structure of a data frame or tibble.

```
str(nh_adults)
```

```
Classes 'tbl_df', 'tbl' and 'data.frame':   500 obs. of  13 variables:
 $ ID          : int  64427 63788 66874 69734 70409 68961 62616 70130 71218 69181 ...
 $ Sex         : Factor w/ 2 levels "female","male": 2 1 1 2 2 1 1 2 2 1 ...
 $ Age         : int  37 40 31 26 44 64 37 42 33 37 ...
 $ Race        : Factor w/ 6 levels "Asian","Black",..: 5 5 5 5 5 5 1 2 5 5 ...
 $ Education   : Factor w/ 5 levels "8th Grade","9 - 11th Grade",..: 5 3 4 5 3 5 1 3 5 1 ...
 $ BMI         : num  36.5 18.2 27.2 20.6 29.2 24.2 19.3 31.2 27.7 25 ...
 $ SBP         : int  111 115 95 137 112 123 109 119 110 114 ...
 $ DBP         : int  72 74 52 75 71 70 73 71 67 74 ...
 $ Pulse       : int  56 102 98 74 62 80 82 62 68 82 ...
 $ PhysActive  : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 1 1 2 2 ...
 $ Smoke100    : Factor w/ 2 levels "No","Yes": 1 2 1 1 2 2 1 1 1 2 ...
```

```
 $ SleepTrouble: Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 2 ...
 $ HealthGen   : Factor w/ 5 levels "Excellent","Vgood",..: 2 3 3 1 3 2 3 3 3 2 ...
```

To see the first few observations, use **head**, and to see the last few, try **tail**...

```
tail(nh_adults, 5) # shows the last five observations in the data set
```

```
# A tibble: 5 x 13
     ID    Sex   Age    Race      Education   BMI   SBP   DBP Pulse
  <int> <fctr> <int>  <fctr>         <fctr> <dbl> <int> <int> <int>
1 69692   male    50   Black 9 - 11th Grade  22.7   132    82    60
2 66472   male    61   White   Some College  41.3   141    77    62
3 71456   male    21 Mexican 9 - 11th Grade  26.7   113    66    78
4 71420 female    54 Mexican 9 - 11th Grade  32.5   126    69    68
5 63617   male    29   White   College Grad  23.2   105    72    76
# ... with 4 more variables: PhysActive <fctr>, Smoke100 <fctr>,
#   SleepTrouble <fctr>, HealthGen <fctr>
```

4.2.2 What are the variables?

The variables we have collected are described in the brief table below[1].

Variable	Description	Sample Values
ID	a numerical code identifying the subject	64427, 63788
Sex	sex of subject (2 levels)	male, female
Age	age (years) at screening of subject	37, 40
Race	reported race of subject (6 levels)	White, Asian
Education	educational level of subject (5 levels)	College Grad, High School
BMI	body-mass index, in kg/m^2	36.5, 18.2
SBP	systolic blood pressure in mm Hg	111, 115
DBP	diastolic blood pressure in mm Hg	72, 74
Pulse	60 second pulse rate in beats per minute	56, 102
PhysActive	Moderate or vigorous-intensity sports?	Yes, No
Smoke100	Smoked at least 100 cigarettes lifetime?	Yes, No
SleepTrouble	Told a doctor they have trouble sleeping?	Yes, No
HealthGen	Self-report general health rating (5 lev.)	Vgood, Good

The levels for the multi-categorical variables are:

- **Race**: Mexican, Hispanic, White, Black, Asian, or Other.
- **Education**: 8th Grade, 9 - 11th Grade, High School, Some College, or College Grad.
- **HealthGen**: Excellent, Vgood, Good, Fair or Poor.

4.3 Types of Variables

4.3.1 Quantitative Variables

Variables recorded in numbers that we use as numbers are called **quantitative**. Familiar examples include incomes, heights, weights, ages, distances, times, and counts. All quantitative variables have measurement

[1] Descriptions are adapted from the ?NHANES help file. Remember that what NHANES lists as Gender is captured here as Sex, and similarly Race3, BPSysAve and BPDiaAve from NHANES are here listed as Race, SBP and DBP.

units, which tell you how the quantitative variable was measured. Without units (like miles per hour, angstroms, yen or degrees Celsius) the values of a quantitative variable have no meaning.

- It does little good to be promised a salary of 80,000 a year if you don't know whether it will be paid in Euros, dollars, yen or Estonian kroon.
- You might be surprised to see someone whose age is 72 listed in a database on childhood diseases until you find out that age is measured in months.
- Often just seeking the units can reveal a variable whose definition is challenging - just how do we measure "friendliness", or "success," for example.
- Quantitative variables may also be classified by whether they are **continuous** or can only take on a **discrete** set of values. Continuous data may take on any value, within a defined range. Suppose we are measuring height. While height is really continuous, our measuring stick usually only lets us measure with a certain degree of precision. If our measurements are only trustworthy to the nearest centimeter with the ruler we have, we might describe them as discrete measures. But we could always get a more precise ruler. The measurement divisions we make in moving from a continuous concept to a discrete measurement are usually fairly arbitrary. Another way to think of this, if you enjoy music, is that, as suggested in Norman and Streiner (2014), a piano is a *discrete* instrument, but a violin is a *continuous* one, enabling finer distinctions between notes than the piano is capable of making. Sometimes the distinction between continuous and discrete is important, but usually, it's not.
  - The `nh_adults` data includes several quantitative variables, specifically Age, BMI, SBP, DBP and Pulse.
  - We know these are quantitative because they have units: Age in years, BMI in kg/m^2, the BP measurements in mm Hg, and Pulse in beats per minute.
  - Depending on the context, we would likely treat most of these as *discrete* given that are measurements are fairly crude (this is certainly true for Age, measured in years) although BMI is probably *continuous* in most settings, even though it is a function of two other measures (Height and Weight) which are rounded off to integer numbers of centimeters and kilograms, respectively.
- It is also possible to separate out quantitative variables into **ratio** variables or **interval** variables. An interval variable has equal distances between values, but the zero point is arbitrary. A ratio variable has equal intervals between values, and a meaningful zero point. For example, weight is an example of a ratio variable, while IQ is an example of an interval variable. We all know what zero weight is. An intelligence score like IQ is a different matter. We say that the average IQ is 100, but that's only by convention. We could just as easily have decided to add 400 to every IQ value and make the average 500 instead. Because IQ's intervals are equal, the difference between and IQ of 70 and an IQ of 80 is the same as the difference between 120 and 130. However, an IQ of 100 is not twice as high as an IQ of 50. The point is that if the zero point is artificial and moveable, then the differences between numbers are meaningful but the ratios between them are not. On the other hand, most lab test values are ratio variables, as are physical characteristics like height and weight. A person who weighs 100 kg is twice as heavy as one who weighs 50 kg; even when we convert kg to pounds, this is still true. For the most part, we can treat and analyze interval or ratio variables the same way.
  - Each of the quantitative variables in our `nh_adults` data can be thought of as ratio variables.
- Quantitative variables lend themselves to many of the summaries we will discuss, like means, quantiles, and our various measures of spread, like the standard deviation or inter-quartile range. They also have at least a chance to follow the Normal distribution.

4.3.2 Qualitative (Categorical) Variables

Qualitative or categorical variables consist of names of categories. These names may be numerical, but the numbers (or names) are simply codes to identify the groups or categories into which the individuals are divided. Categorical variables with two categories, like yes or no, up or down, or, more generally, 1 and 0, are called **binary** variables. Those with more than two-categories are sometimes called **multi-categorical** variables.

- When the categories included in a variable are merely names, and come in no particular order, we sometimes call them **nominal** variables. The most important summary of such a variable is usually a table of frequencies, and the mode becomes an important single summary, while the mean and median are essentially useless.
  - In the nh_adults data, Race is clearly a nominal variable with multiple unordered categories.
- The alternative categorical variable (where order matters) is called **ordinal**, and includes variables that are sometimes thought of as falling right in between quantitative and qualitative variables.
  - Examples of ordinal multi-categorical variables in the `nh_adults` data include the Education and HealthGen variables.
  - Answers to questions like "How is your overall physical health?" with available responses Excellent, Very Good, Good, Fair or Poor, which are often coded as 1-5, certainly provide a perceived *order*, but a group of people with average health status 4 (Very Good) is not necessarily twice as healthy as a group with average health status of 2 (Fair).
- Sometimes we treat the values from ordinal variables as sufficiently scaled to permit us to use quantitative approaches like means, quantiles, and standard deviations to summarize and model the results, and at other times, we'll treat ordinal variables as if they were nominal, with tables and percentages our primary tools.
- Note that all binary variables may be treated as ordinal, or nominal.
  - Binary variables in the `nh_adults` data include Sex, PhysActive, Smoke100, SleepTrouble. Each can be thought of as either ordinal or nominal.

Lots of variables may be treated as either quantitative or qualitative, depending on how we use them. For instance, we usually think of age as a quantitative variable, but if we simply use age to make the distinction between "child" and "adult" then we are using it to describe categorical information. Just because your variable's values are numbers, don't assume that the information provided is quantitative.

Chapter 5

Summarizing Quantitative Variables

Most numerical summaries that might be new to you are applied most appropriately to quantitative variables. The measures that will interest us relate to:

- the **center** of our distribution,
- the **spread** of our distribution, and
- the **shape** of our distribution.

5.1 The `summary` function for Quantitative data

R provides a small sampling of numerical summaries with the `summary` function, for instance.

```
nh_adults %>%
  select(Age, BMI, SBP, DBP, Pulse) %>%
  summary()
```

```
      Age             BMI             SBP            DBP             Pulse
 Min.   :21.0   Min.   :17.8   Min.   : 84   Min.   : 19.0   Min.   : 46
 1st Qu.:31.0   1st Qu.:24.2   1st Qu.:109   1st Qu.: 65.0   1st Qu.: 64
 Median :42.0   Median :27.7   Median :118   Median : 72.0   Median : 72
 Mean   :42.1   Mean   :28.7   Mean   :119   Mean   : 72.2   Mean   : 73
 3rd Qu.:53.0   3rd Qu.:32.1   3rd Qu.:127   3rd Qu.: 79.0   3rd Qu.: 80
 Max.   :64.0   Max.   :69.0   Max.   :202   Max.   :105.0   Max.   :120
                NA's   :3      NA's   :15    NA's   :15      NA's   :15
```

This basic summary includes a set of five **quantiles**[1], plus the sample's **mean**.

- `Min.` = the **minimum** value for each variable, so, for example, the youngest subject's Age was 21.
- `1st Qu.` = the **first quartile** (25^{th} percentile) for each variable - for example, 25% of the subjects were Age 31 or younger.
- `Median` = the **median** (50^{th} percentile) - half of the subjects were Age 42 or younger.
- `Mean` = the **mean**, usually what one means by an *average* - the sum of the Ages divided by 500 is 42.1,
- `3rd Qu.` = the **third quartile** (75^{th} percentile) - 25% of the subjects were Age 53 or older.
- `Max.` = the **maximum** value for each variable, so the oldest subject was Age 64.

The summary also specifies the number of missing values for each variable. Here, we are missing 3 of the BMI values, for example.

[1] The quantiles (sometimes referred to as percentiles) can also be summarised with a boxplot.

5.2 Measuring the Center of a Distribution

5.2.1 The Mean and The Median

The **mean** and **median** are the most commonly used measures of the center of a distribution for a quantitative variable. The median is the more generally useful value, as it is relevant even if the data have a shape that is not symmetric. We might also collect the **sum** of the observations, and the **count** of the number of observations, usually symbolized with n.

For variables without missing values, like `Age`, this is pretty straightforward.

```
nh_adults %>%
    summarise(n = n(), Mean = mean(Age), Median = median(Age), Sum = sum(Age))
```

```
# A tibble: 1 x 4
      n  Mean Median   Sum
  <int> <dbl>  <dbl> <int>
1   500  42.1     42 21051
```

And again, the Mean is just the Sum (21051), divided by the number of non-missing values of Age (500), or 42.102.

The Median is the middle value when the data are sorted in order. When we have an odd number of values, this is sufficient. When we have an even number, as in this case, we take the mean of the two middle values. We could sort and list all 500 Ages, if we wanted to do so.

```
nh_adults %>% select(Age) %>%
    arrange(Age)
```

```
# A tibble: 500 x 1
     Age
   <int>
 1    21
 2    21
 3    21
 4    21
 5    21
 6    21
 7    21
 8    21
 9    21
10    21
# ... with 490 more rows
```

But this data set figures we don't want to output more than 10 observations to a table like this.

If we really want to see all of the data, we can use `View(nh_adults)` to get a spreadsheet-style presentation, or use the `sort` command…

```
sort(nh_adults$Age)
```

```
  [1] 21 21 21 21 21 21 21 21 21 21 21 21 22 22 22 22 22 22 22 22 22 22 23
 [24] 23 23 23 23 23 23 23 23 23 23 23 24 24 24 24 24 24 24 24 24 24 24 24
 [47] 24 25 25 25 25 25 25 25 25 25 25 25 25 26 26 26 26 26 26 26 26 26 26
 [70] 26 26 27 27 27 27 27 27 27 27 27 27 27 27 27 28 28 28 28 28 28 28 28
 [93] 28 28 28 28 28 28 28 28 29 29 29 29 29 29 29 29 29 29 29 30 30 30 30
[116] 30 30 30 30 30 30 30 30 31 31 31 31 31 31 31 31 31 31 31 31 32 32 32
[139] 32 32 32 32 32 32 32 32 33 33 33 33 33 33 33 33 33 33 33 34 34 34 34
```

```
[162] 34 34 34 34 35 35 35 35 36 36 36 36 36 36 36 36 36 37 37 37 37 37 37
[185] 37 37 37 37 37 37 37 37 37 37 38 38 38 38 38 38 38 38 38 38 39 39 39
[208] 39 39 39 39 39 39 39 39 39 39 40 40 40 40 40 40 40 40 40 40 41 41 41
[231] 41 41 41 41 41 41 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 42 43
[254] 43 43 43 43 43 43 43 43 43 43 43 44 44 44 44 44 44 44 44 44 44 44 45
[277] 45 45 45 45 45 45 45 46 46 46 46 46 46 46 46 46 46 46 46 46 47 47 47
[300] 47 47 47 47 47 47 48 48 48 48 48 48 48 48 48 48 48 49 49 49 49 49 49
[323] 49 49 49 49 49 49 49 49 49 50 50 50 50 50 50 50 50 50 50 50 50 50 50
[346] 50 50 50 50 51 51 51 51 51 51 51 51 51 51 51 51 52 52 52 52 52 52 52
[369] 52 52 52 53 53 53 53 53 53 53 53 53 53 53 53 53 54 54 54 54 54 54 54
[392] 54 54 54 54 55 55 55 55 55 55 56 56 56 56 56 56 56 56 56 56 56 56 56
[415] 56 56 56 56 56 56 57 57 57 57 57 57 57 57 57 57 57 57 57 58 58 58 58
[438] 58 58 58 58 58 58 58 59 59 59 59 59 59 59 59 59 59 59 60 60 60 60 60
[461] 60 60 60 60 60 60 60 61 61 61 61 61 61 61 61 61 61 61 61 62 62 62 62
[484] 62 62 62 63 63 63 63 63 64 64 64 64 64 64 64 64 64
```

Again, to find the median, we would take the mean of the middle two observations in this sorted data set. That would be the 250th and 251st largest Ages.

```
sort(nh_adults$Age)[250:251]
```

```
[1] 42 42
```

5.2.2 Dealing with Missingness

When calculating a mean, you may be tempted to try something like this...

```
nh_adults %>%
    summarise(mean(Pulse), median(Pulse))
```

```
# A tibble: 1 x 2
  `mean(Pulse)` `median(Pulse)`
          <dbl>           <int>
1            NA              NA
```

This fails because we have some missing values in the Pulse data. We can address this by either omitting the data with missing values before we run the summarise function, or tell the mean and median summary functions to remove missing values[2].

```
nh_adults %>%
    filter(complete.cases(Pulse)) %>%
    summarise(count = n(), mean(Pulse), median(Pulse))
```

```
# A tibble: 1 x 3
  count `mean(Pulse)` `median(Pulse)`
  <int>         <dbl>           <int>
1   485            73              72
```

Or, we could tell the summary functions themselves to remove NA values.

```
nh_adults %>%
    summarise(mean(Pulse, na.rm=TRUE), median(Pulse, na.rm=TRUE))
```

```
# A tibble: 1 x 2
  `mean(Pulse, na.rm = TRUE)` `median(Pulse, na.rm = TRUE)`
                        <dbl>                         <int>
```

[2] We could also use `!is.na` in place of `complete.cases` to accomplish the same thing.

```
1                             73                             72
```

While we eventually discuss the importance of **imputation** when dealing with missing data, this doesn't apply to providing descriptive summaries of actual, observed values.

5.2.3 The Mode of a Quantitative Variable

One other less common measure of the center of a quantitative variable's distribution is its most frequently observed value, referred to as the **mode**. This measure is only appropriate for discrete variables, be they quantitative or categorical. To find the mode, we usually tabulate the data, and then sort by the counts of the numbers of observations.

```
nh_adults %>%
    group_by(Age) %>%
    summarise(count = n()) %>%
    arrange(desc(count))
```

```
# A tibble: 44 x 2
     Age count
   <int> <int>
 1    56    19
 2    50    18
 3    28    16
 4    37    16
 5    42    16
 6    49    15
 7    24    13
 8    27    13
 9    39    13
10    46    13
# ... with 34 more rows
```

Note the use of three different "verbs" in our function there - for more explanation of this strategy, visit Grolemund and Wickham (2017).

As an alternative, the `modeest` package's `mfv` function calculates the sample mode (or most frequent value).[3]

5.3 Measuring the Spread of a Distribution

Statistics is all about variation, so spread or dispersion is an important fundamental concept in statistics. Measures of spread like the inter-quartile range and range (maximum - minimum) can help us understand and compare data sets. If the values in the data are close to the center, the spread will be small. If many of the values in the data are scattered far away from the center, the spread will be large.

5.3.1 The Range and the Interquartile Range (IQR)

The **range** of a quantitative variable is sometimes interpreted as the difference between the maximum and the minimum, even though R presents the actual minimum and maximum values when you ask for a range...

```
nh_adults %>%
    select(Age) %>%
    range()
```

[3] See the documentation for the `modeest` package's `mlv` function to look at other definitions of the mode.

```
[1] 21 64
```

And, for a variable with missing values, we can use...

```
nh_adults %>%
    select(BMI) %>%
    range(., na.rm=TRUE)
```

```
[1] 17.8 69.0
```

A more interesting and useful statistic is the **inter-quartile range**, or IQR, which is the range of the middle half of the distribution, calculated by subtracting the 25th percentile value from the 75th percentile value.

```
nh_adults %>%
    summarise(IQR(Age), quantile(Age, 0.25), quantile(Age, 0.75))
```

```
# A tibble: 1 x 3
  `IQR(Age)` `quantile(Age, 0.25)` `quantile(Age, 0.75)`
       <dbl>                 <dbl>                 <dbl>
1         22                    31                    53
```

We can calculate the range and IQR nicely from the summary information on quantiles, of course:

```
nh_adults %>%
    select(Age, BMI, SBP, DBP, Pulse) %>%
    summary()
```

```
      Age             BMI             SBP            DBP              Pulse
 Min.   :21.0   Min.   :17.8   Min.   : 84   Min.   : 19.0   Min.   : 46
 1st Qu.:31.0   1st Qu.:24.2   1st Qu.:109   1st Qu.: 65.0   1st Qu.: 64
 Median :42.0   Median :27.7   Median :118   Median : 72.0   Median : 72
 Mean   :42.1   Mean   :28.7   Mean   :119   Mean   : 72.2   Mean   : 73
 3rd Qu.:53.0   3rd Qu.:32.1   3rd Qu.:127   3rd Qu.: 79.0   3rd Qu.: 80
 Max.   :64.0   Max.   :69.0   Max.   :202   Max.   :105.0   Max.   :120
                NA's   :3      NA's   :15    NA's   :15      NA's   :15
```

5.3.2 The Variance and the Standard Deviation

The IQR is always a reasonable summary of spread, just as the median is always a reasonable summary of the center of a distribution. Yet, most people are inclined to summarise a batch of data using two numbers: the **mean** and the **standard deviation**. This is really only a sensible thing to do if you are willing to assume the data follow a Normal distribution: a bell-shaped, symmetric distribution without substantial outliers.

But **most data do not (even approximately) follow a Normal distribution**. Summarizing by the median and quartiles (25th and 75th percentiles) is much more robust, explaining R's emphasis on them.

5.3.3 Obtaining the Variance and Standard Deviation in R

Here are the variances of the quantitative variables in the `nh_adults` data. Note the need to include `na.rm = TRUE` to deal with the missing values in some variables.

```
nh_adults %>%
    select(Age, BMI, SBP, DBP, Pulse) %>%
    summarise_all(var, na.rm = TRUE)
```

```
# A tibble: 1 x 5
    Age   BMI   SBP   DBP Pulse
```

```
  <dbl> <dbl> <dbl> <dbl> <dbl>
1   157  42.1   234   117   132
```

And here are the standard deviations of those same variables.

```
nh_adults %>%
    select(Age, BMI, SBP, DBP, Pulse) %>%
    summarise_all(sd, na.rm = TRUE)
```

```
# A tibble: 1 x 5
    Age   BMI   SBP   DBP Pulse
  <dbl> <dbl> <dbl> <dbl> <dbl>
1  12.5  6.49  15.3  10.8  11.5
```

5.3.4 Defining the Variance and Standard Deviation

Bock, Velleman, and De Veaux (2004) have lots of useful thoughts here, which are lightly edited here.

In thinking about spread, we might consider how far each data value is from the mean. Such a difference is called a *deviation*. We could just average the deviations, but the positive and negative differences always cancel out, leaving an average deviation of zero, so that's not helpful. Instead, we *square* each deviation to obtain non-negative values, and to emphasize larger differences. When we add up these squared deviations and find their mean (almost), this yields the **variance**.

$$\text{Variance} = s^2 = \frac{\Sigma(y - \bar{y})^2}{n - 1}$$

Why almost? It would be the mean of the squared deviations only if we divided the sum by n, but instead we divide by $n - 1$ because doing so produces an estimate of the true (population) variance that is *unbiased*[4]. If you're looking for a more intuitive explanation, this Stack Exchange link awaits your attention.

- To return to the original units of measurement, we take the square root of s^2, and instead work with s, the **standard deviation**.

$$\text{Standard Deviation} = s = \sqrt{\frac{\Sigma(y - \bar{y})^2}{n - 1}}$$

5.3.5 Empirical Rule Interpretation of the Standard Deviation

For a set of measurements that follow a Normal distribution, the interval:

- Mean $\pm$ Standard Deviation contains approximately 68% of the measurements;
- Mean $\pm$ 2(Standard Deviation) contains approximately 95% of the measurements;
- Mean $\pm$ 3(Standard Deviation) contains approximately all (99.7%) of the measurements.

We often refer to the population or process mean of a distribution with μ and the standard deviation with σ, leading to the Figure below.

But if the data are not from an approximately Normal distribution, then this Empirical Rule is less helpful.

[4] When we divide by n-1 as we calculate the sample variance, the average of the sample variances for all possible samples is equal to the population variance. If we instead divided by n, the average sample variance across all possible samples would be a little smaller than the population variance.

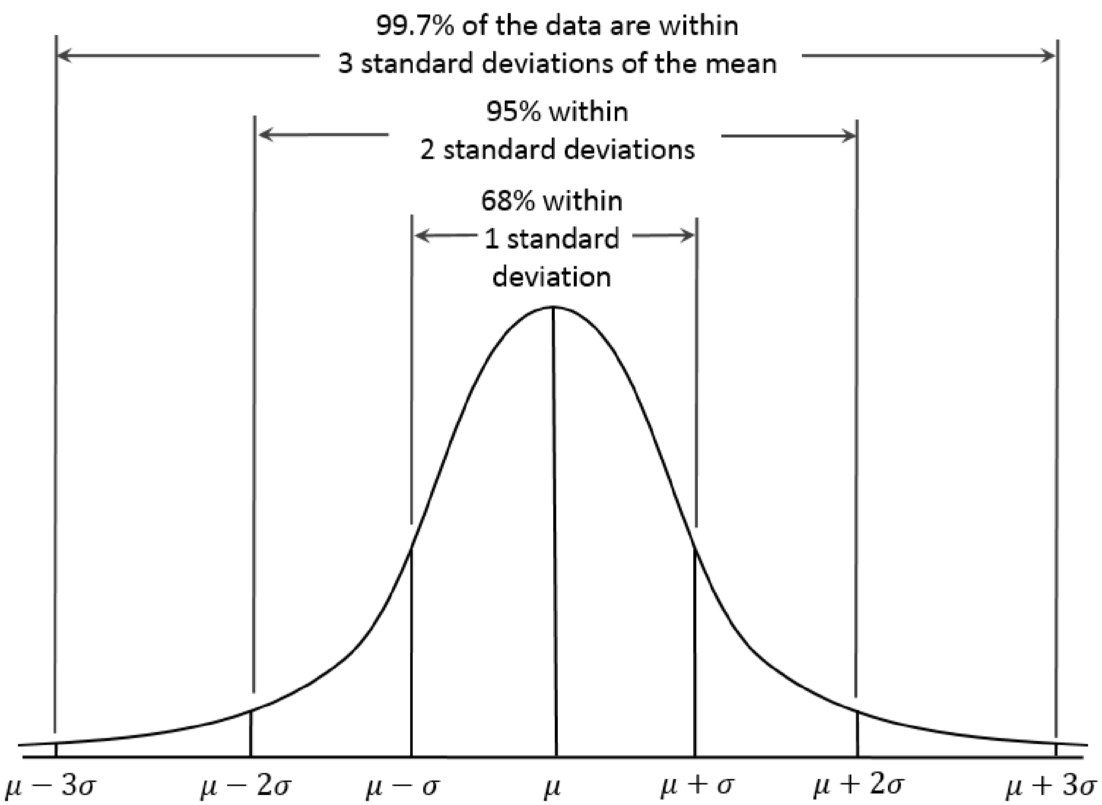


Figure 5.1: The Normal Distribution and the Empirical Rule

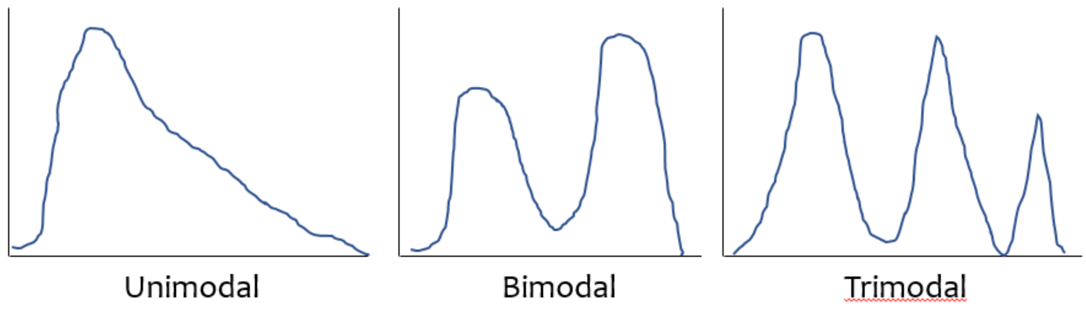


Figure 5.2: Unimodal and Multimodal Sketches

5.3.6 Chebyshev's Inequality: One Interpretation of the Standard Deviation

Chebyshev's Inequality tells us that for any distribution, regardless of its relationship to a Normal distribution, no more than $1/k^2$ of the distribution's values can lie more than k standard deviations from the mean. This implies, for instance, that for **any** distribution, at least 75% of the values must lie within two standard deviations of the mean, and at least 89% must lie within three standard deviations of the mean.

Again, most data sets do not follow a Normal distribution. We'll return to this notion soon. But first, let's try to draw some pictures that let us get a better understanding of the distribution of our data.

5.4 Measuring the Shape of a Distribution

When considering the shape of a distribution, one is often interested in three key points.

- The number of modes in the distribution, which I always assess through plotting the data.
- The **skewness**, or symmetry that is present, which I typically assess by looking at a plot of the distribution of the data, but if required to, will summarise with a non-parametric measure of **skewness**.
- The **kurtosis**, or heavy-tailedness (outlier-proneness) that is present, usually in comparison to a Normal distribution. Again, this is something I nearly inevitably assess graphically, but there are measures.

A Normal distribution has a single mode, is symmetric and, naturally, is neither heavy-tailed nor light-tailed as compared to a Normal distribution (we call this mesokurtic).

5.4.1 Multimodal vs. Unimodal distributions

A unimodal distribution, on some level, is straightforward. It is a distribution with a single mode, or "peak" in the distribution. Such a distribution may be skewed or symmetric, light-tailed or heavy-tailed. We usually describe as multimodal distributions like the two on the right below, which have multiple local maxima, even though they have just a single global maximum peak.

Truly multimodal distributions are usually described that way in terms of shape. For unimodal distributions, skewness and kurtosis become useful ideas.

5.4.2 Skew

Whether or not a distribution is approximately symmetric is an important consideration in describing its shape. Graphical assessments are always most useful in this setting, particularly for unimodal data. My favorite measure of skew, or skewness if the data have a single mode, is:

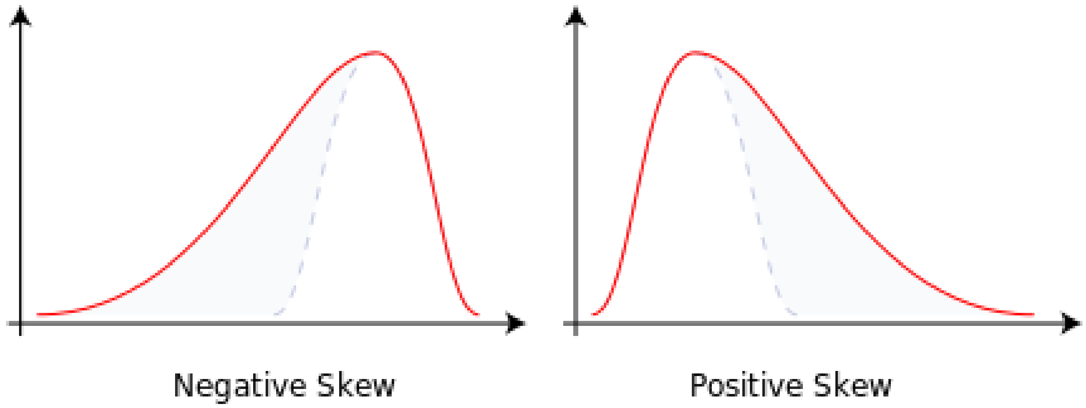


Figure 5.3: Negative (Left) Skew and Positive (Right) Skew

$$skew_1 = \frac{\text{mean} - \text{median}}{\text{standard deviation}}$$

- Symmetric distributions generally show values of $skew_1$ near zero. If the distribution is actually symmetric, the mean should be equal to the median.
- Distributions with $skew_1$ values above 0.2 in absolute value generally indicate meaningful skew.
- Positive skew (mean > median if the data are unimodal) is also referred to as *right skew*.
- Negative skew (mean < median if the data are unimodal) is referred to as *left skew*.

5.4.3 Kurtosis

When we have a unimodal distribution that is symmetric, we will often be interested in the behavior of the tails of the distribution, as compared to a Normal distribution with the same mean and standard deviation. High values of kurtosis measures (and there are several) indicate data which has extreme outliers, or is heavy-tailed.

- A mesokurtic distribution has similar tail behavior to what we would expect from a Normal distribution.
- A leptokurtic distribution is a thinner distribution, with lighter tails (fewer observations far from the center) than we'd expect from a Normal distribution.
- A platykurtic distribution is a flatter distribution, with heavier tails (more observations far from the center) than we'd expect from a Normal distribution.

Graphical tools are in most cases the best way to identify issues related to kurtosis.

5.5 More Detailed Numerical Summaries for Quantitative Variables

5.5.1 `favstats` in the `mosaic` package

The `favstats` function adds the standard deviation, and counts of overall and missing observations to our usual `summary` for a continuous variable. Let's look at systolic blood pressure, because we haven't yet.

```
mosaic::favstats(~ SBP, data = nh_adults)
```

```
 min  Q1 median  Q3 max mean   sd   n missing
  84 109    118 127 202  119 15.3 485      15
```

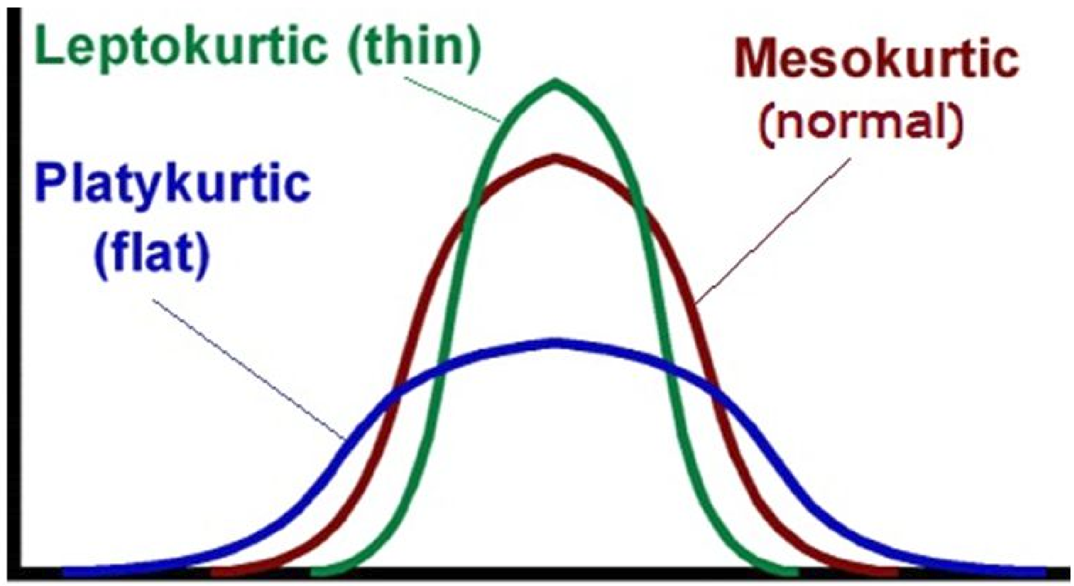


Figure 5.4: The Impact of Kurtosis

We could, of course, duplicate these results with a rather lengthy set of `summarise` pieces...

```
nh_adults %>%
    filter(complete.cases(SBP)) %>%
    summarise(min = min(SBP), Q1 = quantile(SBP, 0.25), median = median(SBP),
              Q3 = quantile(SBP, 0.75), max = max(SBP),
              mean = mean(SBP), sd = sd(SBP), n = n(), missing = sum(is.na(SBP)))
```

```
# A tibble: 1 x 9
    min    Q1 median    Q3   max  mean    sd     n missing
  <dbl> <dbl>  <int> <dbl> <dbl> <dbl> <dbl> <int>   <int>
1    84   109    118   127   202   119  15.3   485       0
```

The somewhat unusual structure of `favstats` (complete with an easy to forget `~`) is actually helpful. It allows you to look at some interesting grouping approaches, like this:

```
mosaic::favstats(SBP ~ Education, data = nh_adults)
```

```
       Education min  Q1 median  Q3 max mean   sd   n missing
1      8th Grade  95 109    122 126 147  119 14.1  21       3
2 9 - 11th Grade 100 111    115 126 152  118 12.0  57       0
3    High School  89 109    120 129 202  121 19.7  78       3
4   Some College  85 110    118 128 163  119 14.6 149       4
5   College Grad  84 108    116 124 172  117 14.7 180       5
```

Of course, we could accomplish the same comparison with `dplyr` commands, too, but the `favstats` approach has much to offer.

```
nh_adults %>%
    filter(complete.cases(SBP, Education)) %>%
    group_by(Education) %>%
    summarise(min = min(SBP), Q1 = quantile(SBP, 0.25), median = median(SBP),
              Q3 = quantile(SBP, 0.75), max = max(SBP),
```

```
              mean = mean(SBP), sd = sd(SBP), n = n(), missing = sum(is.na(SBP)))
```

```
# A tibble: 5 x 10
       Education   min    Q1 median    Q3   max  mean    sd     n missing
          <fctr> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <int>   <int>
1      8th Grade    95   109    122   126   147   119  14.1    21       0
2 9 - 11th Grade   100   111    115   126   152   118  12.0    57       0
3    High School    89   109    120   129   202   121  19.7    78       0
4   Some College    85   110    118   128   163   119  14.6   149       0
5   College Grad    84   108    116   124   172   117  14.7   180       0
```

5.5.2 describe in the psych package

The `psych` package has a more detailed list of numerical summaries for quantitative variables that lets us look at a group of observations at once.

```
psych::describe(nh_adults %>% select(Age, BMI, SBP, DBP, Pulse))
```

```
      vars   n  mean    sd median trimmed   mad  min max range  skew
Age      1 500  42.1 12.54   42.0    42.1 16.31 21.0  64  43.0 -0.03
BMI      2 497  28.7  6.49   27.7    28.1  5.78 17.8  69  51.2  1.33
SBP      3 485 118.6 15.30  118.0   117.8 13.34 84.0 202 118.0  1.00
DBP      4 485  72.2 10.83   72.0    72.1 10.38 19.0 105  86.0 -0.05
Pulse    5 485  73.0 11.47   72.0    72.5 11.86 46.0 120  74.0  0.46
      kurtosis   se
Age      -1.23 0.56
BMI       4.15 0.29
SBP       3.44 0.69
DBP       1.07 0.49
Pulse     0.45 0.52
```

The additional statistics presented here are:

- `trimmed` = a trimmed mean (by default in this function, this removes the top and bottom 10% from the data, then computes the mean of the remaining values - the middle 80% of the full data set.)
- `mad` = the median absolute deviation (from the median), which can be used in a manner similar to the standard deviation or IQR to measure spread.
  - If the data are $Y_1, Y_2, ..., Y_n$, then the `mad` is defined as $median(|Y_i - median(Y_i)|)$.
  - To find the `mad` for a set of numbers, find the median, subtract the median from each value and find the absolute value of that difference, and then find the median of those absolute differences.
  - For non-normal data with a skewed shape but tails well approximated by the Normal, the `mad` is likely to be a better (more robust) estimate of the spread than is the standard deviation.
- a measure of `skew`, which refers to how much asymmetry is present in the shape of the distribution. The measure is not the same as the *nonparametric skew* measure that we will usually prefer. The [Wikipedia page on skewness][https://en.wikipedia.org/wiki/Skewness] is very detailed.
- a measure of `kurtosis`, which refers to how outlier-prone, or heavy-tailed the shape of the distribution is, mainly as compared to a Normal distribution.
- `se` = the standard error of the sample mean, equal to the sample sd divided by the square root of the sample size.

5.5.3 describe in the Hmisc package

```
Hmisc::describe(nh_adults %>% select(Age, BMI, SBP, DBP, Pulse))
```

```
nh_adults %>% select(Age, BMI, SBP, DBP, Pulse)

 5  Variables      500  Observations
---------------------------------------------------------------------------
Age
       n  missing distinct     Info     Mean      Gmd      .05      .10
     500        0       44    0.999     42.1    14.48       23       25
     .25      .50      .75      .90      .95
      31       42       53       59       61

lowest : 21 22 23 24 25, highest: 60 61 62 63 64
---------------------------------------------------------------------------
BMI
       n  missing distinct     Info     Mean      Gmd      .05      .10
     497        3      203        1    28.73    6.947    19.90    22.00
     .25      .50      .75      .90      .95
   24.20    27.70    32.10    36.54    40.82

lowest : 17.8 18.0 18.1 18.2 18.4, highest: 47.6 48.6 48.8 62.8 69.0
---------------------------------------------------------------------------
SBP
       n  missing distinct     Info     Mean      Gmd      .05      .10
     485       15       71    0.999    118.6    16.51       96      101
     .25      .50      .75      .90      .95
     109      118      127      137      143

lowest :  84  85  86  89  91, highest: 163 167 168 172 202
---------------------------------------------------------------------------
DBP
       n  missing distinct     Info     Mean      Gmd      .05      .10
     485       15       57    0.999    72.25    12.04       56       59
     .25      .50      .75      .90      .95
      65       72       79       86       90

lowest :  19  41  45  47  49, highest: 100 101 102 103 105
---------------------------------------------------------------------------
Pulse
       n  missing distinct     Info     Mean      Gmd      .05      .10
     485       15       31    0.997    72.96    12.81       56       60
     .25      .50      .75      .90      .95
      64       72       80       88       92

lowest :  46  48  50  52  54, highest:  98 100 102 108 120
---------------------------------------------------------------------------
```

The `Hmisc` package's version of `describe` for a distribution of data presents three new ideas, in addition to a more comprehensive list of quartiles (the 5^{th}, 10^{th}, 25^{th}, 50^{th}, 75^{th}, 90^{th} and 95^th are shown) and the lowest and highest few observations. These are:

- `distinct` - the number of different values observed in the data.
- `Info` - a measure of how "continuous" the variable is, related to how many "ties" there are in the data, with Info taking a higher value (closer to its maximum of one) if the data are more continuous.
- `Gmd` - the Gini mean difference - a robust measure of spread that is calculated as the mean absolute difference between any pairs of observations. Larger values of Gmd indicate more spread-out distributions.

Chapter 6

Summarizing Categorical Variables

Summarizing categorical variables numerically is mostly about building tables, and calculating percentages or proportions. We'll save our discussion of modeling categorical data for later. Recall that in the `nh_adults` data set we built in Section (@ref(createnh_adults)), we had the following categorical variables. The number of levels indicates the number of possible categories for each categorical variable.

Variable	Description	Levels	Type
Sex	sex of subject	2	binary
Race	subject's race	6	nominal
Education	subject's educational level	5	ordinal
PhysActive	Participates in sports?	2	binary
Smoke100	Smoked 100+ cigarettes?	2	binary
SleepTrouble	Trouble sleeping?	2	binary
HealthGen	Self-report health	5	ordinal

6.1 The `summary` function for Categorical data

When R recognizes a variable as categorical, it stores it as a *factor*. Such variables get special treatment from the `summary` function, in particular a table of available values (so long as there aren't too many.)

```
nh_adults %>%
  select(Sex, Race, Education, PhysActive, Smoke100, SleepTrouble, HealthGen) %>%
  summary()
```

```
     Sex             Race              Education     PhysActive Smoke100
 female:253   Asian   : 29   8th Grade      : 24   No :225    No :289
 male  :247   Black   : 57   9 - 11th Grade: 57   Yes:275    Yes:211
              Hispanic: 39   High School    : 81
              Mexican : 43   Some College   :153
              White   :322   College Grad   :185
              Other   : 10
 SleepTrouble     HealthGen
 No :362      Excellent: 51
 Yes:138      Vgood    :153
              Good     :172
              Fair     : 71
              Poor     :  7
```

```
               NA's     : 46
```

6.2 Tables to describe One Categorical Variable

Suppose we build a table to describe the `HealthGen` distribution.

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "ifany")
```

```
.
Excellent     Vgood      Good      Fair      Poor      <NA>
       51       153       172        71         7        46
```

The main tools we have for augmenting tables are:

- adding in marginal totals, and
- working with proportions/percentages.

What if we want to add a total count?

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "ifany") %>%
    addmargins()
```

```
.
Excellent     Vgood      Good      Fair      Poor      <NA>       Sum
       51       153       172        71         7        46       500
```

What if we want to leave out the missing responses?

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "no") %>%
    addmargins()
```

```
.
Excellent     Vgood      Good      Fair      Poor       Sum
       51       153       172        71         7       454
```

Let's put the missing values back in, but now calculate proportions instead. Since the total will just be 1.0, we'll leave that out.

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "ifany") %>%
    prop.table()
```

```
.
Excellent     Vgood      Good      Fair      Poor      <NA>
    0.102     0.306     0.344     0.142     0.014     0.092
```

Now, we'll calculate percentages by multiplying the proportions by 100.

```
nh_adults %>%
    select(HealthGen) %>%
    table(., useNA = "ifany") %>%
```

```
    prop.table() %>%
    "*"(100)
```

```
.
Excellent     Vgood      Good      Fair      Poor      <NA>
     10.2      30.6      34.4      14.2       1.4       9.2
```

6.3 The Mode of a Categorical Variable

A common measure applied to a categorical variable is to identify the mode, the most frequently observed value. To find the mode for variables with lots of categories (so that the `summary` may not be sufficient), we usually tabulate the data, and then sort by the counts of the numbers of observations, as we did with discrete quantitative variables.

```
nh_adults %>%
    group_by(HealthGen) %>%
    summarise(count = n()) %>%
    arrange(desc(count))
```

```
# A tibble: 6 x 2
  HealthGen count
     <fctr> <int>
1      Good   172
2     Vgood   153
3      Fair    71
4 Excellent    51
5      <NA>    46
6      Poor     7
```

6.4 describe in the Hmisc package

```
Hmisc::describe(nh_adults %>%
                    select(Sex, Race, Education, PhysActive,
                           Smoke100, SleepTrouble, HealthGen))
```

```
nh_adults %>% select(Sex, Race, Education, PhysActive, Smoke100, SleepTrouble, HealthGen)

 7  Variables      500  Observations
---------------------------------------------------------------------------
Sex
       n  missing distinct
     500        0        2

Value      female   male
Frequency     253    247
Proportion  0.506  0.494
---------------------------------------------------------------------------
Race
       n  missing distinct
     500        0        6
```

```
Value         Asian    Black Hispanic  Mexican    White    Other
Frequency        29       57       39       43      322       10
Proportion    0.058    0.114    0.078    0.086    0.644    0.020
---------------------------------------------------------------------------
Education
       n  missing distinct
     500        0        5

Value           8th Grade 9 - 11th Grade    High School   Some College
Frequency              24             57             81            153
Proportion          0.048          0.114          0.162          0.306

Value        College Grad
Frequency             185
Proportion          0.370
---------------------------------------------------------------------------
PhysActive
       n  missing distinct
     500        0        2

Value        No  Yes
Frequency   225  275
Proportion 0.45 0.55
---------------------------------------------------------------------------
Smoke100
       n  missing distinct
     500        0        2

Value         No   Yes
Frequency    289   211
Proportion 0.578 0.422
---------------------------------------------------------------------------
SleepTrouble
       n  missing distinct
     500        0        2

Value         No   Yes
Frequency    362   138
Proportion 0.724 0.276
---------------------------------------------------------------------------
HealthGen
       n  missing distinct
     454       46        5

Value      Excellent     Vgood      Good      Fair      Poor
Frequency         51       153       172        71         7
Proportion     0.112     0.337     0.379     0.156     0.015
---------------------------------------------------------------------------
```

6.5 Cross-Tabulations

It is very common for us to want to describe the association of one categorical variable with another. For instance, is there a relationship between Education and SleepTrouble in these data?

```
nh_adults %>%
    select(Education, SleepTrouble) %>%
    table() %>%
    addmargins()
```

```
                SleepTrouble
Education         No Yes Sum
  8th Grade       15   9  24
  9 - 11th Grade  40  17  57
  High School     67  14  81
  Some College   107  46 153
  College Grad   133  52 185
  Sum            362 138 500
```

To get row percentages, we can use:

```
nh_adults %>%
    select(Education, SleepTrouble) %>%
    table() %>%
    prop.table(., 1) %>%
    "*"(100)
```

```
                SleepTrouble
Education          No  Yes
  8th Grade      62.5 37.5
  9 - 11th Grade 70.2 29.8
  High School    82.7 17.3
  Some College   69.9 30.1
  College Grad   71.9 28.1
```

For column percentages, we use 2 instead of 1 in the `prop.table` function. Here, we'll also round off to two decimal places:

```
nh_adults %>%
    select(Education, SleepTrouble) %>%
    table() %>%
    prop.table(., 2) %>%
    "*"(100) %>%
    round(.,2)
```

```
                SleepTrouble
Education           No   Yes
  8th Grade       4.14  6.52
  9 - 11th Grade 11.05 12.32
  High School    18.51 10.14
  Some College   29.56 33.33
  College Grad   36.74 37.68
```

Here's another approach, to look at the cross-classification of Race and HealthGen:

```
xtabs(~ Race + HealthGen, data = nh_adults)
```

```
          HealthGen
```

```
Race       Excellent Vgood Good Fair Poor
  Asian            4     7    9    2    1
  Black            7    11   16   11    2
  Hispanic         1     9   18    8    0
  Mexican          5     6   12   16    1
  White           34   115  115   32    3
  Other            0     5    2    2    0
```

6.5.1 Cross-Classifying Three Categorical Variables

Suppose we are interested in `Smoke100` and its relationship to `PhysActive` and `SleepTrouble`.

```
xtabs(~ Smoke100 + PhysActive + SleepTrouble, data = nh_adults)
```

```
, , SleepTrouble = No

        PhysActive
Smoke100  No Yes
     No   99 135
     Yes  62  66

, , SleepTrouble = Yes

        PhysActive
Smoke100  No Yes
     No   26  29
     Yes  38  45
```

We can also build a **flat** version of this table, as follows:

```
ftable(Smoke100 ~ PhysActive + SleepTrouble, data = nh_adults)
```

```
                        Smoke100  No Yes
PhysActive SleepTrouble
No         No                     99  62
           Yes                    26  38
Yes        No                    135  66
           Yes                    29  45
```

And we can do this with `dplyr` functions, as well, for example…

```
nh_adults %>%
    select(Smoke100, PhysActive, SleepTrouble) %>%
    table()
```

```
, , SleepTrouble = No

        PhysActive
Smoke100  No Yes
     No   99 135
     Yes  62  66

, , SleepTrouble = Yes

        PhysActive
Smoke100  No Yes
```

```
No   26  29
Yes  38  45
```

6.6 Constructing Tables Well

The prolific Howard Wainer is responsible for many interesting books on visualization and related issues, including Wainer (2005) and Wainer (2013). These rules come from Chapter 10 of Wainer (1997).

1. Order the rows and columns in a way that makes sense.
2. Round, a lot!
3. ALL is different and important

6.6.1 Alabama First!

Which of these Tables is more useful to you?

2013 Percent of Students in grades 9-12 who are obese

State	% Obese	95% CI	Sample Size
Alabama	17.1	(14.6 - 19.9)	1,499
Alaska	12.4	(10.5-14.6)	1,167
Arizona	10.7	(8.3-13.6)	1,520
Arkansas	17.8	(15.7-20.1)	1,470
Connecticut	12.3	(10.2-14.7)	2,270
Delaware	14.2	(12.9-15.6)	2,475
Florida	11.6	(10.5-12.8)	5,491
...			
Wisconsin	11.6	(9.7-13.9)	2,771
Wyoming	10.7	(9.4-12.2)	2,910

or ...

State	% Obese	95% CI	Sample Size
Kentucky	18.0	(15.7 - 20.6)	1,537
Arkansas	17.8	(15.7 - 20.1)	1,470
Alabama	17.1	(14.6 - 19.9)	1,499
Tennessee	16.9	(15.1 - 18.8)	1,831
Texas	15.7	(13.9 - 17.6)	3,039
...			
Massachusetts	10.2	(8.5 - 12.1)	2,547
Idaho	9.6	(8.2 - 11.1)	1,841
Montana	9.4	(8.4 - 10.5)	4,679
New Jersey	8.7	(6.8 - 11.2)	1,644
Utah	6.4	(4.8 - 8.5)	2,136

It is a rare event when Alabama first is the best choice.

6.6.2 Order rows and columns sensibly

- Alabama First!
  - Size places - put the largest first. We often look most carefully at the top.
- Order time from the past to the future to help the viewer.
- If there is a clear predictor-outcome relationship, put the predictors in the rows and the outcomes in the columns.

6.6.3 Round - a lot!

- Humans cannot understand more than two digits very easily.
- We almost never care about accuracy of more than two digits.
- We can almost never justify more than two digits of accuracy statistically.
- It's also helpful to remember that we are almost invariably publishing progress to date, rather than a truly final answer.

Suppose, for instance, we report a correlation coefficient of 0.25. How many observations do you think you would need to justify such a choice?

- To report 0.25 meaningfully, we want to be sure that the second digit isn't 4 or 6.
- That requires a standard error less than 0.005
- The *standard error* of any statistic is proportional to 1 over the square root of the sample size, n.

So $\frac{1}{\sqrt{n}} \sim 0.005$, but that means $\sqrt{n} = \frac{1}{0.005} = 200$. If $\sqrt{n} = 200$, then $n = (200)^2 = 40{,}000$.

Do we usually have 40,000 observations?

6.6.4 ALL is different and important

Summaries of rows and columns provide a measure of what is typical or usual. Sometimes a sum is helpful, at other times, consider presenting a median or other summary. The ALL category, as Wainer (1997) suggests, should be both visually different from the individual entries and set spatially apart.

On the whole, it's *far* easier to fall into a good graph in R (at least if you have some ggplot2 skills) than to produce a good table.

Chapter 7

The National Youth Fitness Survey (nyfs1)

The nyfs1.csv data file comes from the 2012 National Youth Fitness Survey.

> The NHANES National Youth Fitness Survey (NNYFS) was conducted in 2012 to collect data on physical activity and fitness levels in order to provide an evaluation of the health and fitness of children in the U.S. ages 3 to 15. The NNYFS collected data on physical activity and fitness levels of our youth through interviews and fitness tests.

In the nyfs1.csv data file, I'm only providing a tiny portion of the available information. More on the NNYFS (including information I'm not using) is available at the links below.

- Demographic Information including a complete description of all available variables.
- Body Measures, part of the general Examination data with complete variable descriptions

What I did was merge a few elements from the available demographic information with some elements from the body measures data, reformulated and simplified some variables, and restricted the sample to kids who had a complete set of body measure examinations.

7.1 Looking over the Data Set

To start with, I'll take a look at the nyfs1 data. One approach is to simply get the size of the set and the names of the available data elements.

```
## first, we'll import the data into the nyfs1 data frame
nyfs1 <- read.csv("data/nyfs1.csv")

## next we'll turn that data frame into a more useful tibble
nyfs1 <- tbl_df(nyfs1)

## size of the data frame
dim(nyfs1)
```

```
[1] 1416    7
```

There are 1416 rows (subjects) and 7 columns (variables), by which I mean that there are 1416 kids in the nyfs1 data frame, and we have 7 pieces of information on each subject.

So, what do we have, exactly?

```
nyfs1 # this is a tibble, has some nice features in a print-out like this
```

```
# A tibble: 1,416 x 7
   subject.id    sex age.exam   bmi         bmi.cat waist.circ
        <int> <fctr>    <int> <dbl>          <fctr>      <dbl>
 1      71918 Female        8  22.3         4 Obese       71.9
 2      71919 Female       14  19.8 2 Normal weight       79.4
 3      71921   Male        3  15.2 2 Normal weight       46.8
 4      71922   Male       12  25.9         4 Obese       90.0
 5      71923   Male       12  22.5    3 Overweight       72.3
 6      71924 Female        8  14.4 2 Normal weight       56.1
 7      71925   Male        7  15.9 2 Normal weight       54.5
 8      71926   Male        8  17.0 2 Normal weight       59.7
 9      71927   Male        3  15.8 2 Normal weight       49.9
10      71928 Female        9  16.0 2 Normal weight       59.9
# ... with 1,406 more rows, and 1 more variables: triceps.skinfold <dbl>
```

Tibbles are a modern reimagining of the main way in which people have stored data in R, called a data frame. Tibbles were developed to keep what time has proven to be effective, and throwing out what is not. We can obtain the structure of the tibble from the `str` function.

```
str(nyfs1)
```

```
Classes 'tbl_df', 'tbl' and 'data.frame':   1416 obs. of  7 variables:
 $ subject.id      : int  71918 71919 71921 71922 71923 71924 71925 71926 71927 71928 ...
 $ sex             : Factor w/ 2 levels "Female","Male": 1 1 2 2 2 1 2 2 2 1 ...
 $ age.exam        : int  8 14 3 12 12 8 7 8 3 9 ...
 $ bmi             : num  22.3 19.8 15.2 25.9 22.5 14.4 15.9 17 15.8 16 ...
 $ bmi.cat         : Factor w/ 4 levels "1 Underweight",..: 4 2 2 4 3 2 2 2 2 2 ...
 $ waist.circ      : num  71.9 79.4 46.8 90 72.3 56.1 54.5 59.7 49.9 59.9 ...
 $ triceps.skinfold: num  19.9 15 8.6 22.8 20.5 12.9 6.9 8.8 10.8 13.2 ...
```

7.1.1 subject.id

The first variable, `subject.id` is listed by R as an `int` variable, for integer, which means it consists of whole numbers. However, the information provided by this variable is minimal. This is just an identifying code attributable to a given subject of the survey. This is *nominal* data, which will be of little interest down the line. On some occasions, as in this case, the ID numbers are sequential, in the sense that subject 71919 was included in the data base after subject 71918, but this fact isn't particularly interesting here, because the protocol remained unchanged throughout the study.

7.1.2 sex

The second variable, `sex` is listed as a factor (R uses **factor** to refer to categorical, especially non-numeric information) with two levels, *Female* and *Male*. You'll note that what is stored in the structure is a series of 1 (referring to the first level - Female) and 2 (Male) values. If we want to know how many people fall in each category, we can build a little table.

```
dplyr::select(nyfs1, sex) %>%
  table()
```

```
.
Female   Male
   707    709
```

```
dplyr::select(nyfs1, sex) %>%
  table() %>%
  addmargins() ## add marginal totals
```

```
.
Female   Male    Sum
   707    709   1416
```

```
dplyr::select(nyfs1, sex) %>%
  table() %>%
  prop.table() ## look at the proportions instead
```

```
.
Female   Male
 0.499  0.501
```

Obviously, we don't actually need more than a couple of decimal places for any real purpose.

7.1.3 age.exam

The third variable, `age.exam` is the age of the child at the time of the examination, measured in years. Note that age is a continuous concept, but the measure used here (number of full years alive) is a common discrete approach to measurement. Age, of course, has a meaningful zero point, so this can be thought of as a ratio variable; a child who is 6 is half as old as one who is 12. We can get a table of the observed values.

```
dplyr::select(nyfs1, age.exam) %>%
  table() %>%
  addmargins()
```

```
.
   3    4    5    6    7    8    9   10   11   12   13   14   15   16  Sum
  97  111  119  129  123  120   90  109  102  108  113  104   85    6 1416
```

Note that some of the children apparently turned 16 between the time they were initially screened (when they were required to be between 3 and 15 years of age) and the time of the examination. The `sum` listed here is just the total count of all subjects. Since this is a meaningful quantitative variable, we may be interested in a more descriptive summary.

```
dplyr::select(nyfs1, age.exam) %>%
  summary()
```

```
    age.exam
 Min.   : 3.00
 1st Qu.: 6.00
 Median : 9.00
 Mean   : 8.86
 3rd Qu.:12.00
 Max.   :16.00
```

These six numbers provide a nice, if incomplete, look at the ages.

- `Min.` = the minimum, or youngest age at the examination was 3 years old.
- `1st Qu.` = the first quartile (25th percentile) of the ages was 6. This means that 25 percent of the subjects were age 6 or less.
- `Median` = the second quartile (50th percentile) of the ages was 9. This is often used to describe the center of the data. Half of the subjects were age 9 or less.
- `3rd Qu.` = the third quartile (75th percentile) of the ages was 12

- `Max.` = the maximum, or oldest age at the examination was 16 years.

7.1.4 bmi

The fourth variable, `bmi`, is the body-mass index of the child. The BMI is a person's weight in kilograms divided by his or her height in meters squared. Symbolically, BMI = weight in kg / (height in m)^2. This is a continuous concept, measured to as many decimal places as you like, and it has a meaningful zero point, so it's a ratio variable.

```
dplyr::select(nyfs1, bmi) %>%
  summary()
```

```
      bmi
 Min.   :11.9
 1st Qu.:15.8
 Median :17.7
 Mean   :18.8
 3rd Qu.:20.9
 Max.   :38.8
```

Why would a table of these BMI values not be a great idea, for these data? A hint is that R represents this variable as `num` or numeric in its depiction of the data structure, and this implies that R has some decimal values stored.

```
dplyr::select(nyfs1, bmi) %>%
  table()
```

```
.
11.9 12.6 12.7 12.9   13 13.1 13.2 13.3 13.4 13.5 13.6 13.7 13.8 13.9   14
   1    1    1    1    2    2    1    1    3    4    5    4    5   11    7
14.1 14.2 14.3 14.4 14.5 14.6 14.7 14.8 14.9   15 15.1 15.2 15.3 15.4 15.5
  12    9   11   11   11    9   17   20   23   13   14   18   27   24   32
15.6 15.7 15.8 15.9   16 16.1 16.2 16.3 16.4 16.5 16.6 16.7 16.8 16.9   17
  18   20   21   30   27   15   18   30   12   25   20   22   13   21   23
17.1 17.2 17.3 17.4 17.5 17.6 17.7 17.8 17.9   18 18.1 18.2 18.3 18.4 18.5
  14   20   14   10   19   13   17   18   14   17   13   10    9    8   15
18.6 18.7 18.8 18.9   19 19.1 19.2 19.3 19.4 19.5 19.6 19.7 19.8 19.9   20
  10   17   10   11    4   13   15   12    8   25    6    6   16    8   13
20.1 20.2 20.3 20.4 20.5 20.6 20.7 20.8 20.9   21 21.1 21.2 21.3 21.4 21.5
   9    7   12    7    3    9    5    6   11    7    5    6    8    9    8
21.6 21.7 21.8 21.9   22 22.1 22.2 22.3 22.4 22.5 22.6 22.7 22.8 22.9   23
   6    7   16    6   13    7    7    8    6    4    4    5    2   10    7
23.1 23.2 23.3 23.4 23.5 23.6 23.7 23.8 23.9   24 24.1 24.2 24.3 24.4 24.5
   3    8    3    5    4    3    2    4    4    5    1    4    3    5    5
24.6 24.7 24.8 24.9   25 25.1 25.2 25.3 25.4 25.5 25.6 25.7 25.8 25.9   26
   4    3    6    4    3    2    4    2    3    3    4    5    3    3    2
26.1 26.2 26.3 26.4 26.5 26.6 26.7 26.8   27 27.2 27.3 27.4 27.5 27.6 27.7
   1    4    2    1    2    1    2    1    2    1    2    2    1    2    1
27.9 28.1 28.2 28.4 28.5 28.6 28.7 28.8 28.9   29 29.2 29.5 29.7 29.8 30.1
   2    2    2    1    1    2    2    1    3    1    3    1    2    3    2
30.2 30.4 30.5 30.7 30.8 30.9 31.1 31.3 31.4 31.5 31.7 31.8   32 32.2 32.4
   4    1    2    1    1    1    1    1    2    1    1    2    2    1    1
32.6 32.9 33.2 33.5   34 34.4 34.6 34.7 35.9   37 38.8
   1    1    1    1    1    1    1    1    1    1    1
```

7.1.5 bmi.cat

Our next variable, `bmi.cat`, is a four-category ordinal variable, which divides the sample according to BMI into four groups. The BMI categories use sex-specific 2000 BMI-for-age (in months) growth charts prepared by the Centers for Disease Control for the US. We can get the breakdown from a table of the variable's values.

```
dplyr::select(nyfs1, bmi.cat) %>%
  table() %>%
  addmargins()
```

```
.
  1 Underweight 2 Normal weight    3 Overweight         4 Obese 
             42             926             237             211 
            Sum 
           1416 
```

In terms of percentiles by age and sex from the growth charts, the meanings of the categories are:

- Underweight (BMI $<$ 5th percentile)
- Normal weight (BMI 5th to $<$ 85th percentile)
- Overweight (BMI 85th to $<$ 95th percentile)
- Obese (BMI $\geq$ 95th percentile)

Note how I've used labels in the `bmi.cat` variable that include a number at the start so that the table results are sorted in a rational way. R sorts tables alphabetically, in general.

7.1.6 waist.circ

The sixth variable is `waist.circ`, which is the circumference of the child's waist, in centimeters. Again, this is a numeric variable, so perhaps we'll stick to the simple summary, rather than obtaining a table of observed values.

```
dplyr::select(nyfs1, waist.circ) %>%
summary()
```

```
   waist.circ   
 Min.   : 42.5  
 1st Qu.: 55.0  
 Median : 63.0  
 Mean   : 65.3  
 3rd Qu.: 72.9  
 Max.   :112.4  
```

7.1.7 triceps.skinfold

The seventh and final variable is `triceps.skinfold`, which is measured in millimeters. This is one of several common locations used for the assessment of body fat using skinfold calipers, and is a frequent part of growth assessments in children. Again, this is a numeric variable according to R.

```
dplyr::select(nyfs1, triceps.skinfold) %>%
  summary()
```

```
 triceps.skinfold
 Min.   : 4.0    
 1st Qu.: 9.0    
 Median :11.8    
```

```
 Mean   :13.4
 3rd Qu.:16.6
 Max.   :38.2
```

7.2 Summarizing the Data Set

The `summary` function can be applied to the whole tibble. For numerical and integer variables, this function produces the five number summary, plus the mean. For categorical (factor) variables, it lists the count for each category.

```
summary(nyfs1)
```

```
   subject.id        sex          age.exam          bmi
 Min.   :71918   Female:707   Min.   : 3.00   Min.   :11.9
 1st Qu.:72313   Male  :709   1st Qu.: 6.00   1st Qu.:15.8
 Median :72698                Median : 9.00   Median :17.7
 Mean   :72703                Mean   : 8.86   Mean   :18.8
 3rd Qu.:73096                3rd Qu.:12.00   3rd Qu.:20.9
 Max.   :73492                Max.   :16.00   Max.   :38.8
            bmi.cat      waist.circ    triceps.skinfold
 1 Underweight  : 42   Min.   : 42.5   Min.   : 4.0
 2 Normal weight:926   1st Qu.: 55.0   1st Qu.: 9.0
 3 Overweight   :237   Median : 63.0   Median :11.8
 4 Obese        :211   Mean   : 65.3   Mean   :13.4
                       3rd Qu.: 72.9   3rd Qu.:16.6
                       Max.   :112.4   Max.   :38.2
```

7.2.1 The Five Number Summary, Quantiles and IQR

The **five number summary** is most famous when used to form a box plot - it's the minimum, 25th percentile, median, 75th percentile and maximum. Our usual `summary` adds the mean.

```
nyfs1 %>%
    select(bmi) %>%
    summary()
```

```
      bmi
 Min.   :11.9
 1st Qu.:15.8
 Median :17.7
 Mean   :18.8
 3rd Qu.:20.9
 Max.   :38.8
```

As an alternative, we can use the `$` notation to indicate the variable we wish to study inside a data set, and we can use the `fivenum` function to get the five numbers used in developing a box plot.

```
fivenum(nyfs1$bmi)
```

```
[1] 11.9 15.8 17.7 20.9 38.8
```

- As mentioned in 5.3.1, the **inter-quartile range**, or IQR, is sometimes used as a competitor for the standard deviation. It's the difference between the 75th percentile and the 25th percentile. The 25th percentile, median, and 75th percentile are referred to as the quartiles of the data set, because, together, they split the data into quarters.

```
IQR(nyfs1$bmi)
```

```
[1] 5.1
```

We can obtain **quantiles** (percentiles) as we like - here, I'm asking for the 1st and 99th

```
quantile(nyfs1$bmi, probs=c(0.01, 0.99))
```

```
  1%  99%
13.5 32.0
```

7.3 Additional Summaries from `favstats`

If we're focusing on a single variable, the `favstats` function in the `mosaic` package can be very helpful. Rather than calling up the entire `mosaic` library here, I'll just specify the function within the library.

```
mosaic::favstats(nyfs1$bmi)
```

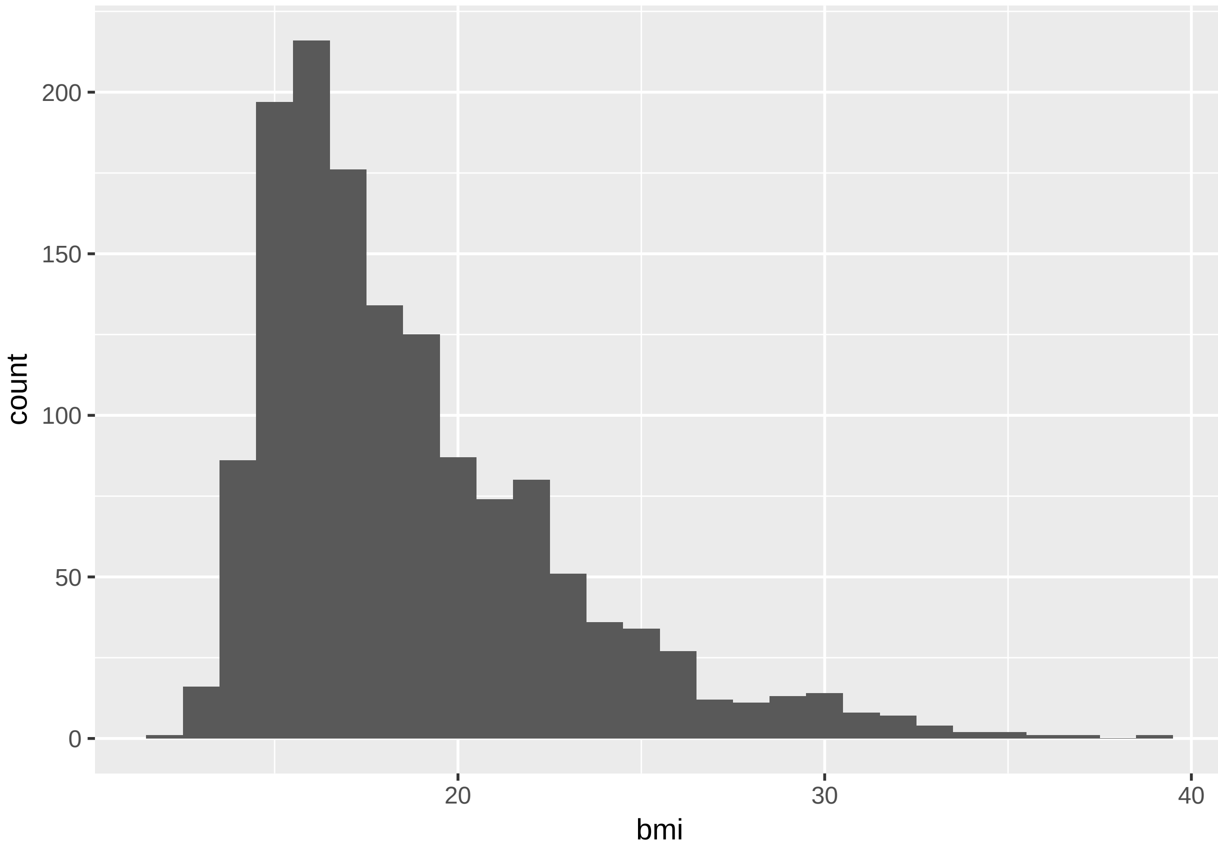
```
  min   Q1 median   Q3  max mean   sd    n missing
 11.9 15.8   17.7 20.9 38.8 18.8 4.08 1416       0
```

This adds three useful results to the base summary - the standard deviation, the sample size and the number of missing observations.

7.4 The Histogram

As we saw in 3, obtaining a basic **histogram** of, for example, the BMIs in the `nyfs1` data is pretty straightforward.

```
ggplot(data = nyfs1, aes(x = bmi)) +
    geom_histogram(binwidth = 1)
```



7.4.1 Freedman-Diaconis Rule to select bin width

If we like, we can suggest a particular number of cells for the histogram, instead of accepting the defaults. In this case, we have $n = 1416$ observations. The **Freedman-Diaconis rule** can be helpful here. That rule suggests that we set the bin-width to

$$h = \frac{2 * IQR}{n^{1/3}}$$

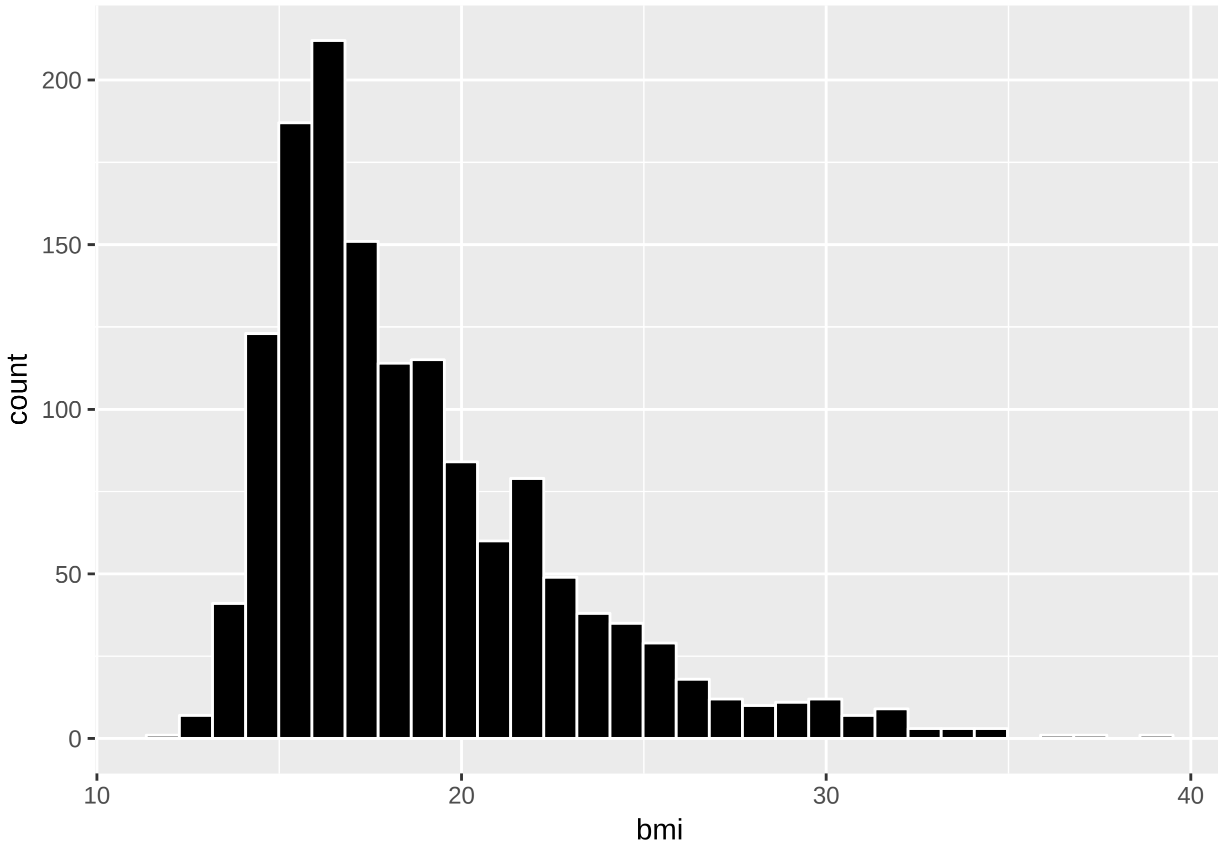
so that the number of bins is equal to the range of the data set (maximum - minimum) divided by h.

For the `bmi` data in the `nyfs1` tibble, we have

- IQR of 5.1, $n = 1416$ and range $= 26.9$
- Thus, by the Freedman-Diaconis rule, the optimal binwidth h is 0.908, or, realistically, 1.
- And so the number of bins would be 29.615, or, realistically 30.

Here, we'll draw the graph again, using the Freedman-Diaconis rule to identify the number of bins, and also play around a bit with the fill and color of the bars.

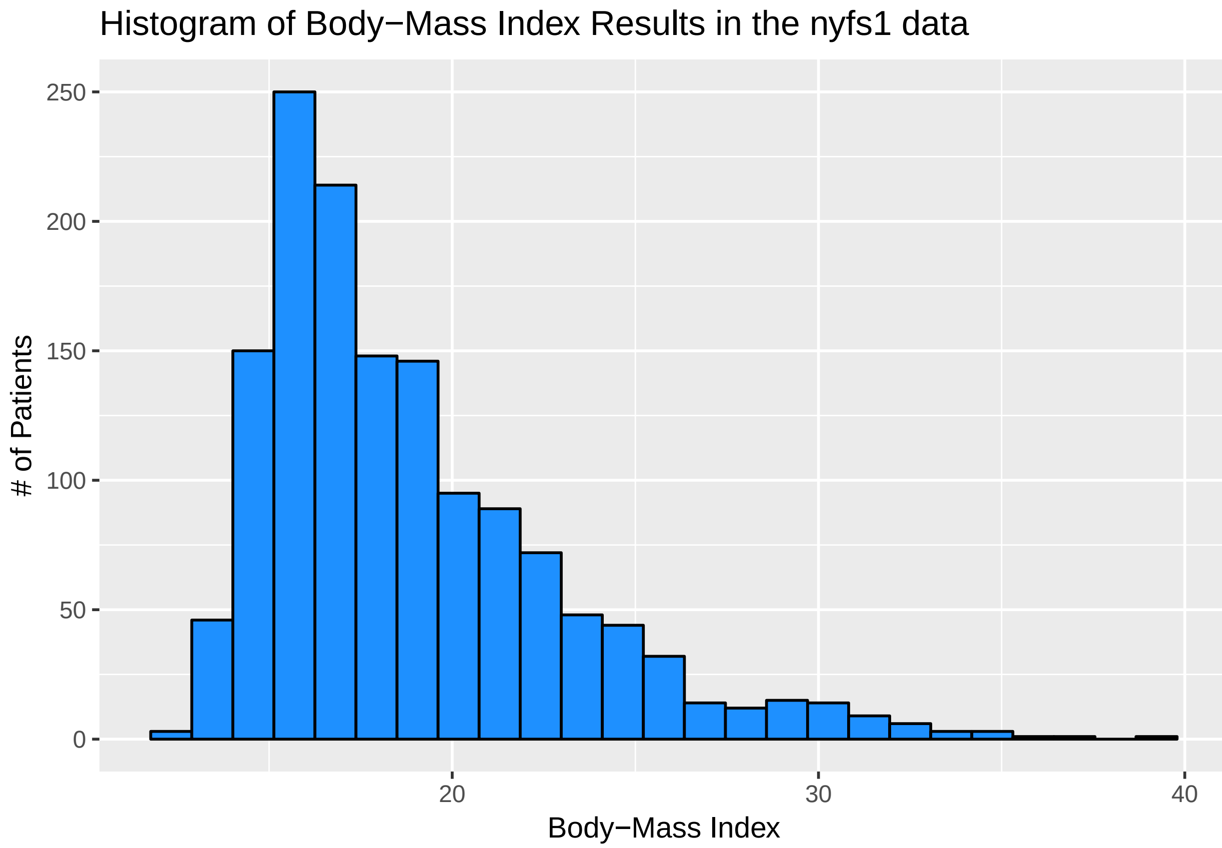
```
bw <- 2 * IQR(nyfs1$bmi) / length(nyfs1$bmi)^(1/3)
ggplot(data = nyfs1, aes(x = bmi)) +
    geom_histogram(binwidth=bw, color = "white", fill = "black")
```



This is a nice start, but it is by no means a finished graph.

Let's improve the axis labels, add a title, and fill in the bars with a distinctive blue and use a black outline around each bar. I'll just use 25 bars, because I like how that looks in this case, and optimizing the number of bins is rarely important.

```
ggplot(data = nyfs1, aes(x = bmi)) +
    geom_histogram(bins=25, color = "black", fill = "dodgerblue") +
    labs(title = "Histogram of Body-Mass Index Results in the nyfs1 data",
         x = "Body-Mass Index", y = "# of Patients")
```



7.5 A Note on Colors

The simplest way to specify a color is with its name, enclosed in parentheses. My favorite list of R colors is http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf. In a pinch, you can find it by googling **Colors in R**. You can also type `colors()` in the R console to obtain a list of the names of the same 657 colors.

When using colors to make comparisons, you may be interested in using a scale that has some nice properties. I suggest the `viridis` package to help with this work. The viridis package vignette describes four color scales (viridis, magma, plasma and inferno) that are designed to be colorful, robust to colorblindness and gray scale printing, and perceptually uniform, which means (as the package authors describe it) that values close to each other have similar-appearing colors and values far away from each other have more different-appearing colors, consistently across the range of values.

7.6 The Stem-and-Leaf

We might consider a **stem-and-leaf display** (a John Tukey invention) to show the actual data values while retaining the shape of a histogram. The `scale` parameter can help expand the size of the diagram, so you can see more of the values. Stem and leaf displays are usually used for relatively small samples, perhaps with 10-200 observations, so we'll first take a sample of 150 of the BMI values from the complete set gathered in the `nyfs1` tibble.

```
set.seed(431) # set a seed for the random sampling so we can replicate the results
```

```
sampleA <- sample_n(nyfs1, 150, replace = FALSE) # draw a sample of 150 unique rows from nyfs1

stem(sampleA$bmi) # build a stem-and-leaf for those 150 sampled BMI values
```

```

  The decimal point is at the |

  13 | 129
  14 | 001224455566778889
  15 | 02344455567789999
  16 | 00001122333345667779
  17 | 001225556677789
  18 | 0111346677888899
  19 | 111224555578889
  20 | 0113334456899
  21 | 014568
  22 | 11349
  23 | 012479
  24 | 478
  25 | 05669
  26 | 03
  27 | 05
  28 |
  29 |
  30 | 27
  31 |
  32 | 4
  33 |
  34 | 67
```

We can see that the minimum BMI value in this small sample is 13.1 and the maximum BMI value is 34.7.

Here's a summary of all variables for these 150 observations.

```
summary(sampleA)
```

```
   subject.id        sex         age.exam          bmi
 Min.   :71935   Female:68   Min.   : 3.00   Min.   :13.1
 1st Qu.:72302   Male  :82   1st Qu.: 6.00   1st Qu.:15.9
 Median :72688               Median :10.00   Median :18.1
 Mean   :72679               Mean   : 9.45   Mean   :19.0
 3rd Qu.:73080               3rd Qu.:13.00   3rd Qu.:20.6
 Max.   :73490               Max.   :15.00   Max.   :34.7
             bmi.cat      waist.circ     triceps.skinfold
 1 Underweight  :  4   Min.   : 45.6   Min.   : 5.6
 2 Normal weight:103   1st Qu.: 55.4   1st Qu.: 9.2
 3 Overweight   : 21   Median : 64.7   Median :12.2
 4 Obese        : 22   Mean   : 66.5   Mean   :13.6
                       3rd Qu.: 72.8   3rd Qu.:16.6
                       Max.   :108.4   Max.   :34.8
```

If we really wanted to, we could obtain a stem-and-leaf of all of the BMI values in the entire `nyfs1` data. The `scale` parameter lets us see some more of the values.

```
stem(nyfs1$bmi, scale = 2)
```

```
  The decimal point is at the |

  11 | 9
  12 | 679
  13 | 00112344455555666667777888889999999999
  14 | 0000000111111111112222222223333333333344444444444555555555566666+50
  15 | 0000000000000111111111111122222222222222222333333333333333333333+137
  16 | 0000000000000000000000000011111111111112222222222222222223333333+123
  17 | 0000000000000000000000111111111111222222222222222222233333333333+82
  18 | 0000000000000000011111111111222222222333333333344444445555555555+40
  19 | 0000111111111111222222222222223333333333333444444455555555555555+33
  20 | 0000000000000111111122222233333333333344444455566666666777777888+2
  21 | 0000000111122222233333333444444444555555556666667777778888888888
  22 | 00000000000001111112222223333333344444455566667777789999999999
  23 | 0000000111222222223334444455556667788889999
  24 | 00000122223334444455555666677788888889999
  25 | 00011222233444555666677777888999
  26 | 0012222334556778
  27 | 0023344566799
  28 | 11224566778999
  29 | 0222577888
  30 | 112222455789
  31 | 13445788
  32 | 002469
  33 | 25
  34 | 0467
  35 | 9
  36 |
  37 | 0
  38 | 8
```

Note that some of the rows extend far beyond what is displayed in the data (as indicated by the `+` sign, followed by a count of the number of unshown data values.)

7.6.1 A Fancier Stem-and-Leaf Display

We can use the `stem.leaf` function in the `aplpack` package to obtain a fancier version of the stem-and-leaf plot, that identifies outlying values. Below, we display this new version for the random sample of 150 BMI observations we developed earlier.

```r
aplpack::stem.leaf(sampleA$bmi)
```

```
1 | 2: represents 1.2
 leaf unit: 0.1
            n: 150
    3    13 | 129
   21    14 | 001224455566778889
   38    15 | 02344455567789999
   57    16 | 0000112233345667779
   72    17 | 001225556677789
  (16)   18 | 0111346677888899
   62    19 | 111224555578889
   47    20 | 0113334456899
   34    21 | 014568
```

```
   28    22 | 11349
   23    23 | 012479
   17    24 | 478
   14    25 | 05669
    9    26 | 03
    7    27 | 05
HI: 30.2 30.7 32.4 34.6 34.7
```

We can also produce back-to-back stem and leaf plots to compare, for instance, body-mass index by sex.

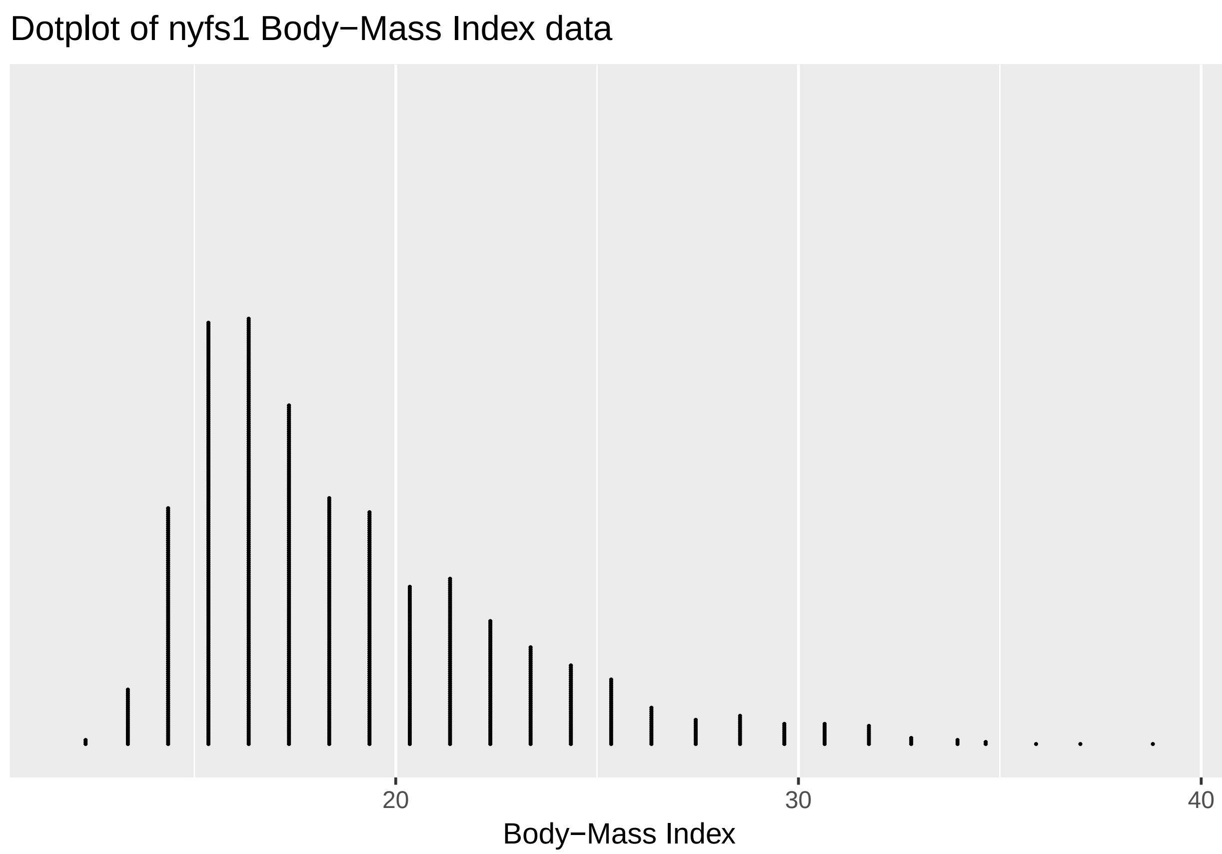
```
samp.F <- filter(sampleA, sex=="Female")
samp.M <- filter(sampleA, sex=="Male")

aplpack::stem.leaf.backback(samp.F$bmi, samp.M$bmi)
```

```
______________________________________________
  1 | 2: represents 1.2, leaf unit: 0.1
          samp.F$bmi      samp.M$bmi
______________________________________________
   3              921| 13 |
  16   9876654422100| 14 |55788           5
  21            98444| 15 |023555677999   17
  33     776653210000| 16 |1233479        24
  (2)              91| 17 |0022555667778  37
  33          9887410| 18 |113667889      (9)
  26       9888555411| 19 |12257          36
  16          9954310| 20 |133468         31
   9                0| 21 |14568          25
                     | 22 |11349          20
   8              910| 23 |247            15
   5                8| 24 |47             12
   4               95| 25 |066            10
                     | 26 |03              7
                     | 27 |05              5
                     | 28 |
______________________________________________
HI: 30.2 32.4             HI: 30.7 34.6 34.7
n:                 68      82
______________________________________________
```

7.7 The Dot Plot to display a distribution

We can plot the distribution of a single continuous variable using the `dotplot` geom:

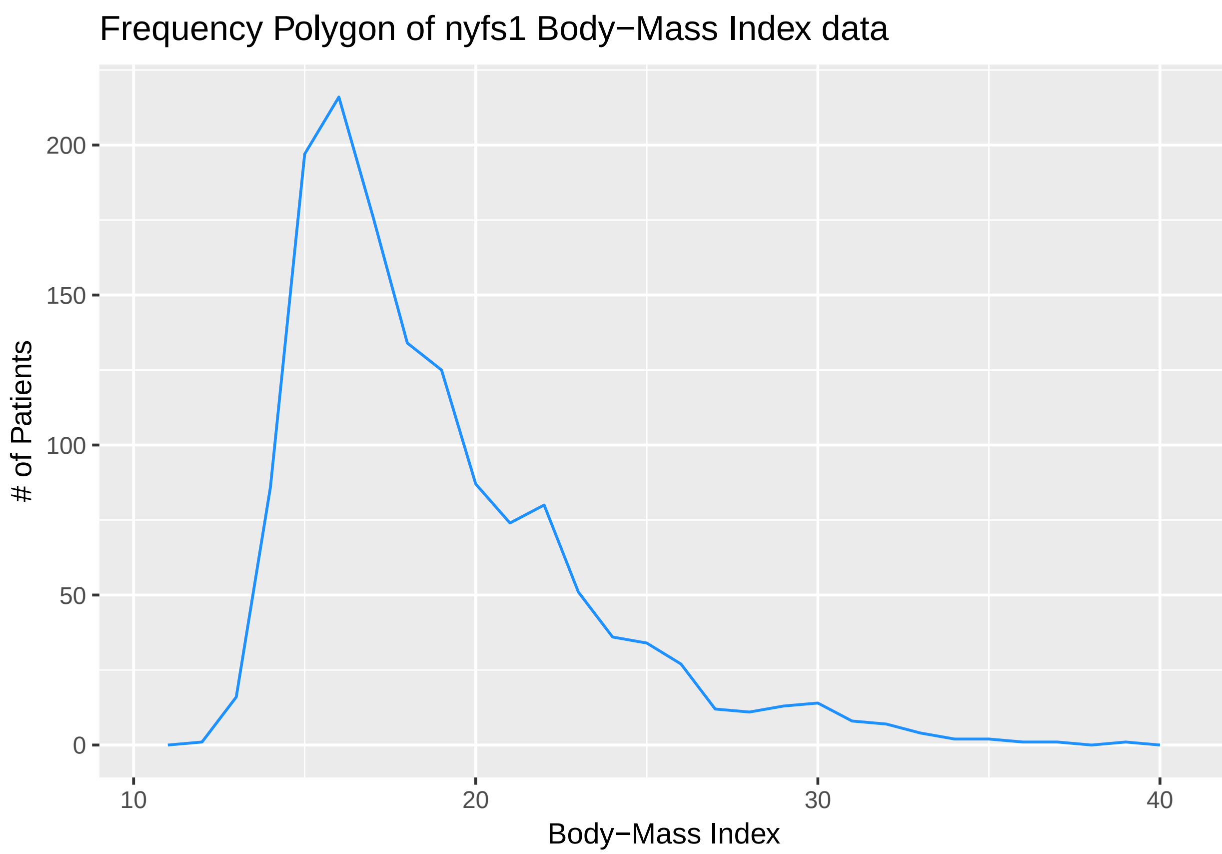
```
ggplot(data = nyfs1, aes(x = bmi)) +
    geom_dotplot(dotsize = 0.05, binwidth=1) +
    scale_y_continuous(NULL, breaks = NULL) + # hides y-axis since it is meaningless
    labs(title = "Dotplot of nyfs1 Body-Mass Index data",
         x = "Body-Mass Index")
```



7.8 The Frequency Polygon

We can plot the distribution of a single continuous variable using the `freqpoly` geom:

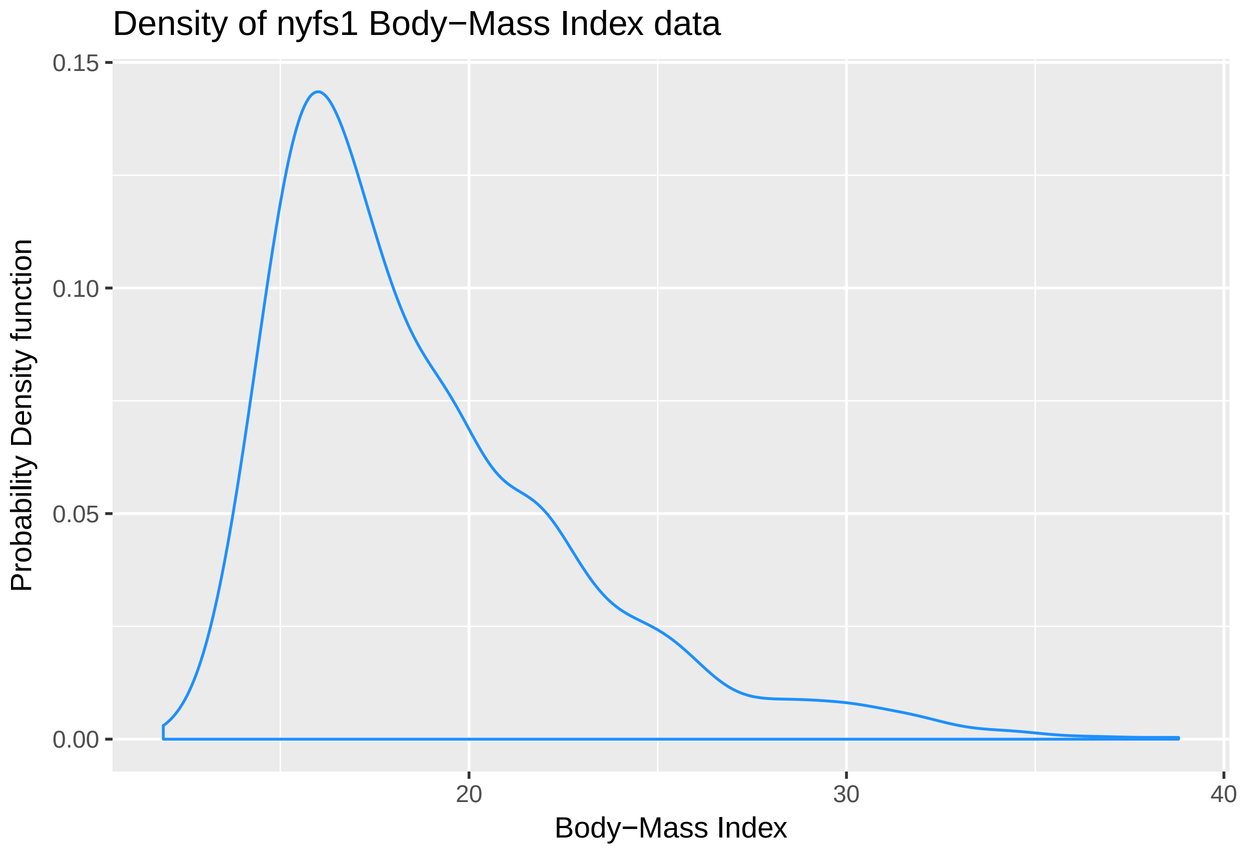
```
ggplot(data = nyfs1, aes(x = bmi)) +
    geom_freqpoly(binwidth = 1, color = "dodgerblue") +
    labs(title = "Frequency Polygon of nyfs1 Body-Mass Index data",
         x = "Body-Mass Index", y = "# of Patients")
```



7.9 Plotting the Probability Density Function

We can also produce a density function, which has the effect of smoothing out the bumps in a histogram or frequency polygon, while also changing what is plotted on the y-axis.

```
ggplot(data = nyfs1, aes(x = bmi)) +
    geom_density(kernel = "gaussian", color = "dodgerblue") +
    labs(title = "Density of nyfs1 Body-Mass Index data",
         x = "Body-Mass Index", y = "Probability Density function")
```



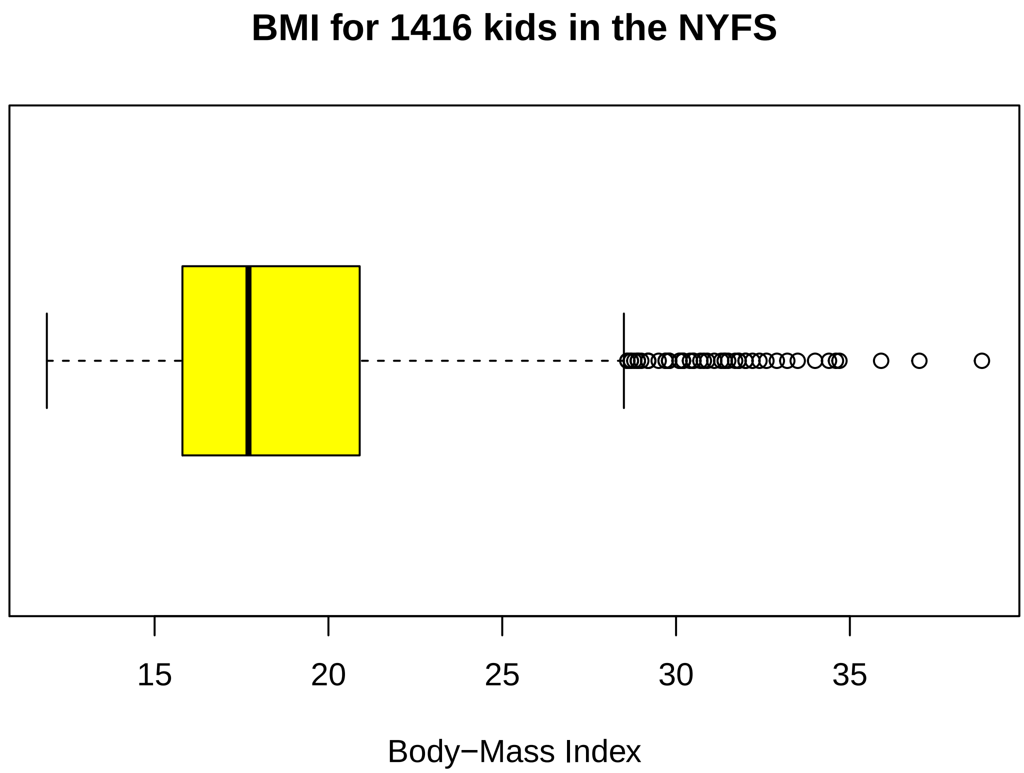
So, what's a density function?

- A probability density function is a function of a continuous variable, x, that represents the probability of x falling within a given range. Specifically, the integral over the interval (a,b) of the density function gives the probability that the value of x is within (a,b).
- If you're interested in exploring more on the notion of density functions for continuous (and discrete) random variables, some nice elementary material is available at Khan Academy.

7.10 The Boxplot

Sometimes, it's helpful to picture the five-number summary of the data in such a way as to get a general sense of the distribution. One approach is a **boxplot**, sometimes called a box-and-whisker plot.

```
boxplot(nyfs1$bmi, col="yellow", horizontal=T, xlab="Body-Mass Index",
        main="BMI for 1416 kids in the NYFS")
```



The boxplot is another John Tukey invention.

- R draws the box (here in yellow) so that its edges of the box fall at the 25th and 75th percentiles of the data, and the thick line inside the box falls at the median (50th percentile).
- The whiskers then extend out to the largest and smallest values that are not classified by the plot as candidate *outliers*.
- An outlier is an unusual point, far from the center of a distribution.
- Note that I've used the `horizontal` option to show this boxplot in this direction. Most comparison boxplots, as we'll see below, are oriented vertically.

The boxplot's **whiskers** that are drawn from the first and third quartiles (i.e. the 25th and 75th percentiles) out to the most extreme points in the data that do not meet the standard of "candidate outliers." An outlier is simply a point that is far away from the center of the data - which may be due to any number of reasons, and generally indicates a need for further investigation.

Most software, including R, uses a standard proposed by Tukey which describes a "candidate outlier" as any point above the ***upper fence*** or below the ***lower fence***. The definitions of the fences are based on the inter-quartile range (IQR).

If IQR = 75th percentile - 25th percentile, then the upper fence is 75th percentile + 1.5 IQR, and the lower fence is 25th percentile - 1.5 IQR.

So for these BMI data,

- the upper fence is located at 20.9 + 1.5(5.1) = 28.55
- the lower fence is located at 15.8 - 1.5(5.1) = 8.15

In this case, we see no points identified as outliers in the low part of the distribution, but quite a few identified that way on the high side. This tends to identify about 5% of the data as a candidate outlier, *if* the data follow a Normal distribution.

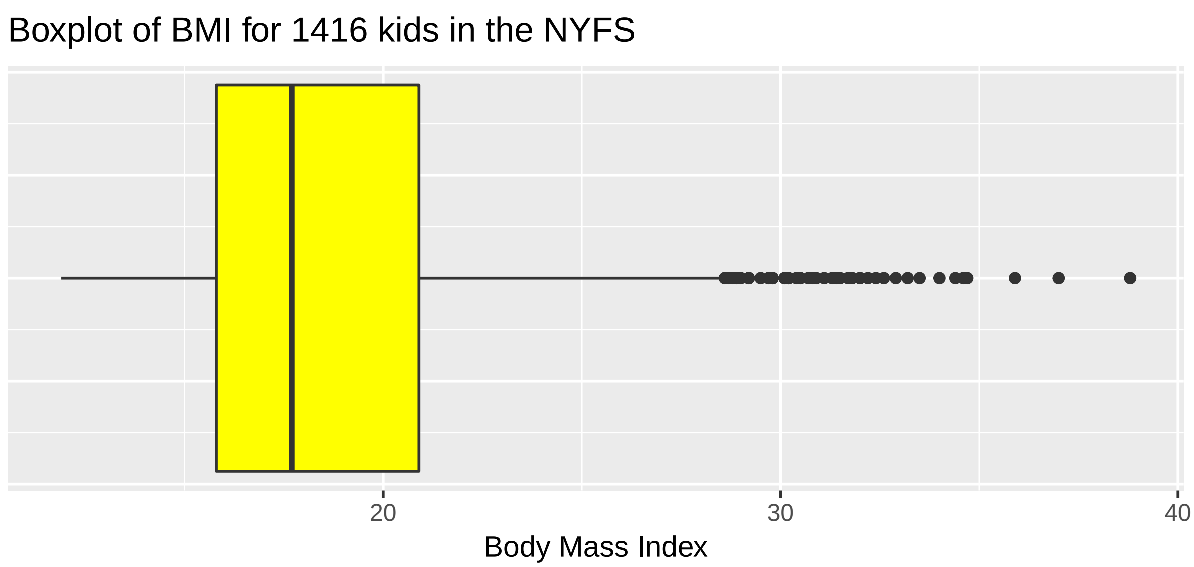
- This plot is indicating clearly that there is some asymmetry (skew) in the data, specifically right skew.
- The standard R uses is to indicate as outliers any points that are more than 1.5 inter-quartile ranges away from the edges of the box.

The horizontal orientation I've chosen here clarifies the relationship of direction of skew to the plot. A plot like this, with multiple outliers on the right side is indicative of a long right tail in the distribution, and hence, positive or right skew - with the mean being larger than the median. Other indications of skew include having one side of the box being substantially wider than the other, or one side of the whiskers being substantially longer than the other. More on skew later.

7.10.1 Drawing a Boxplot for One Variable in `ggplot2`

The `ggplot2` library easily handles comparison boxplots for multiple distributions, as we'll see in a moment. However, building a boxplot for a single distribution requires a little trickiness.

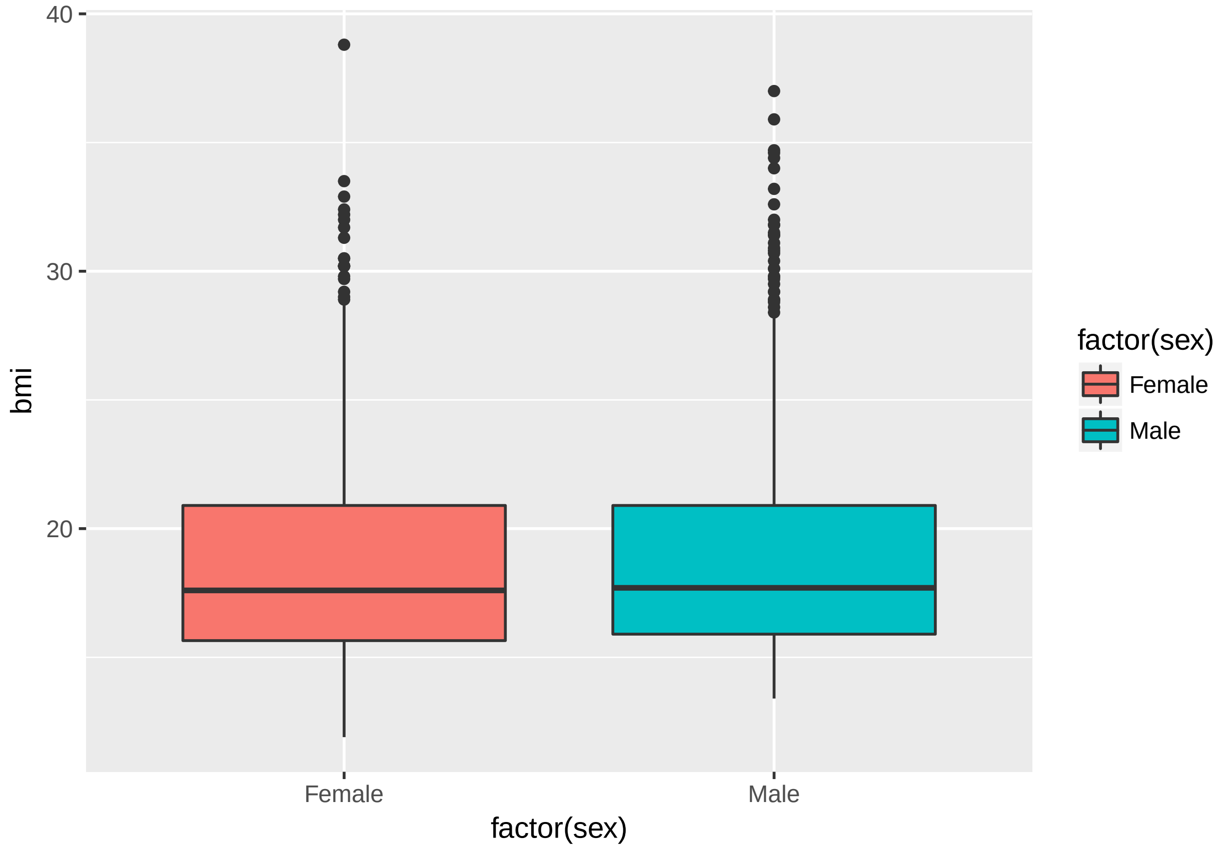
```
ggplot(nyfs1, aes(x = 1, y = bmi)) +
    geom_boxplot(fill = "yellow") +
    coord_flip() +
    labs(title = "Boxplot of BMI for 1416 kids in the NYFS",
         y = "Body Mass Index",
         x = "") +
    theme(axis.text.y = element_blank(),
          axis.ticks.y = element_blank())
```



7.11 A Simple Comparison Boxplot

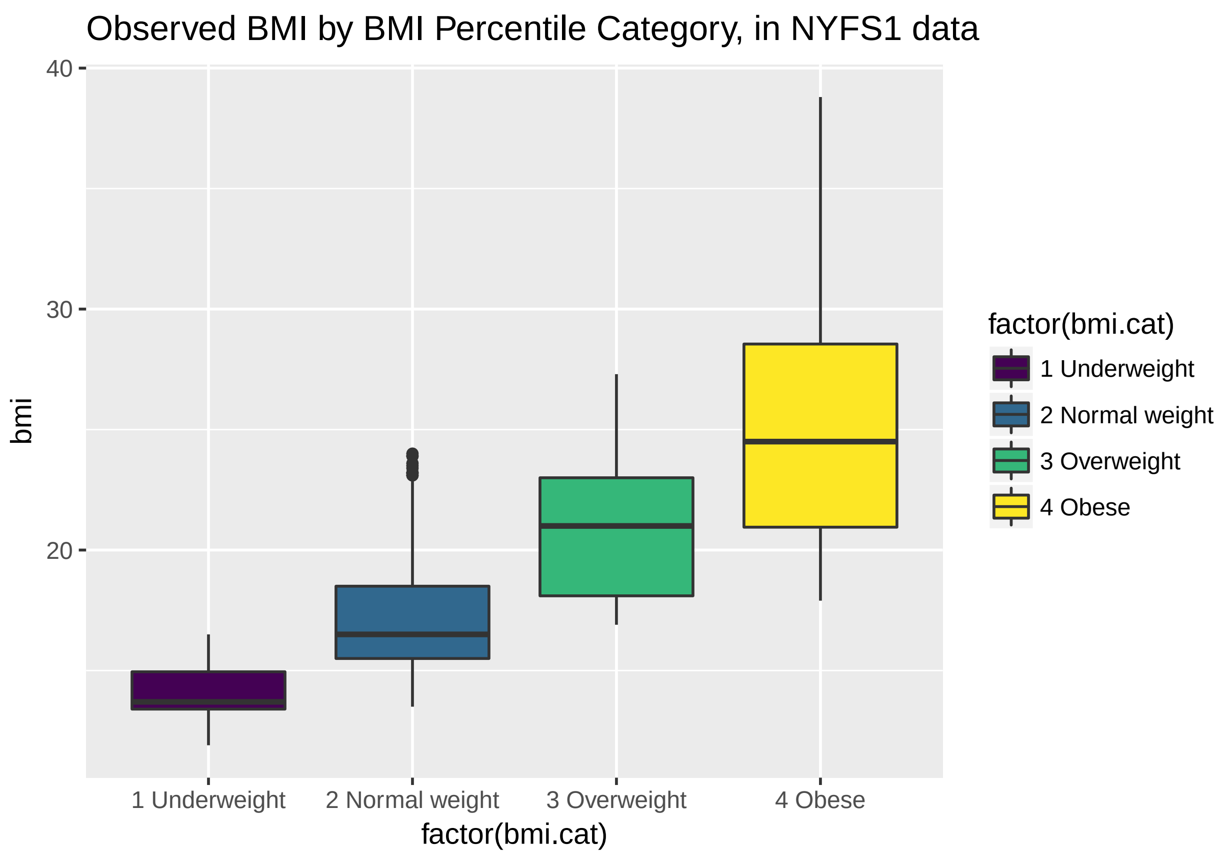
Boxplots are most often used for comparison. We can build boxplots using `ggplot2`, as well, and we'll discuss that in detail later. For now, here's a boxplot built to compare the `bmi` results by the child's sex.

```
ggplot(nyfs1, aes(x = factor(sex), y = bmi, fill=factor(sex))) +
    geom_boxplot()
```



Let's look at the comparison of observed BMI levels across the four categories in our `bmi.cat` variable, now making use of the `viridis` color scheme.

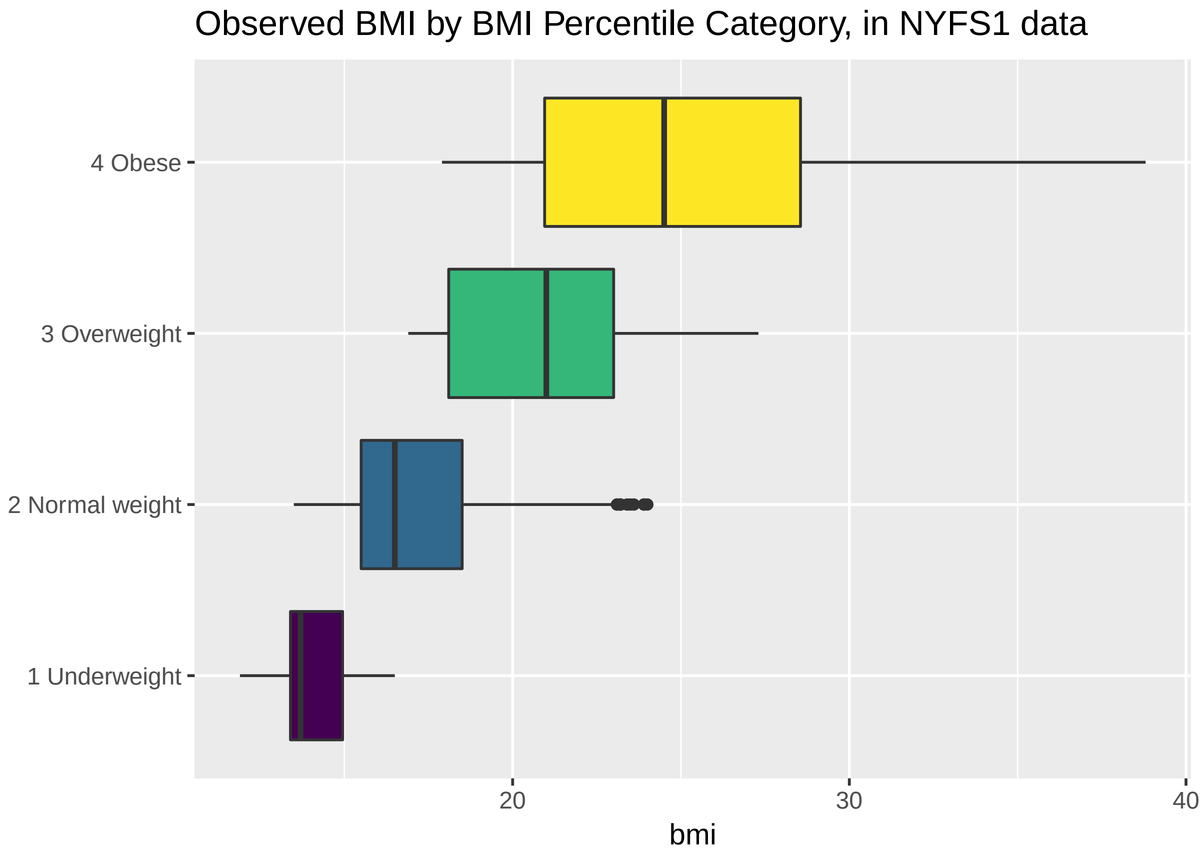
```
ggplot(nyfs1, aes(x = factor(bmi.cat), y = bmi, fill = factor(bmi.cat))) +
    geom_boxplot() +
    scale_fill_viridis(discrete=TRUE) +
    # above line uses viridis palette to identify color choices
    labs(title = "Observed BMI by BMI Percentile Category, in NYFS1 data")
```



Note that the BMI categories incorporate additional information (in particular the age and sex of the child) beyond the observed BMI, and so the observed BMI levels overlap quite a bit across the four categories. As a graph, that's not bad, but what if we want to improve it further?

Let's turn the boxes in the horizontal direction, and get rid of the perhaps unnecessary `bmi.cat` labels.

```
ggplot(nyfs1, aes(x = factor(bmi.cat), y = bmi, fill = factor(bmi.cat))) +
    geom_boxplot() +
    scale_fill_viridis(discrete=TRUE) +
    coord_flip() +
    guides(fill=FALSE) +
    labs(title = "Observed BMI by BMI Percentile Category, in NYFS1 data", x = "")
```



7.12 Using `describe` in the `psych` library

For additional numerical summaries, one option would be to consider using the `describe` function from the `psych` library.

```
psych::describe(nyfs1$bmi)
```

```
   vars    n mean   sd median trimmed  mad  min  max range skew kurtosis
X1    1 1416 18.8 4.08   17.7    18.2 3.26 11.9 38.8  26.9 1.35     1.97
     se
X1 0.11
```

This package provides, in order, the following...

- `n` = the sample size
- `mean` = the sample mean
- `sd` = the sample standard deviation
- `median` = the median, or 50th percentile
- `trimmed` = mean of the middle 80% of the data
- `mad` = median absolute deviation
- `min` = minimum value in the sample
- `max` = maximum value in the sample
- `range` = max - min
- `skew` = skewness measure, described below (indicates degree of asymmetry)
- `kurtosis` = kurtosis measure, described below (indicates heaviness of tails, degree of outlier-proneness)
- `se` = standard error of the sample mean = sd / square root of sample size, useful in inference

7.12.1 The Trimmed Mean

The **trimmed mean** trim value in R indicates proportion of observations to be trimmed from each end of the outcome distribution before the mean is calculated. The `trimmed` value provided by the `psych::describe` package describes what this particular package calls a 20% trimmed mean (bottom and top 10% of BMIs are removed before taking the mean - it's the mean of the middle 80% of the data.) I might call that a 10% trimmed mean in some settings, but that's just me.

```
mean(nyfs1$bmi, trim=.1)
```

```
[1] 18.2
```

7.12.2 The Median Absolute Deviation

An alternative to the IQR that is fancier, and a bit more robust, is the **median absolute deviation**, which, in large sample sizes, for data that follow a Normal distribution, will be (in expectation) equal to the standard deviation. The MAD is the median of the absolute deviations from the median, multiplied by a constant (1.4826) to yield asymptotically normal consistency.

```
mad(nyfs1$bmi)
```

```
[1] 3.26
```

7.13 Assessing Skew

A relatively common idea is to assess **skewness**, several measures of which (including the one below, sometimes called type 3 skewness, or Pearson's moment coefficient of skewness) are available. Many models assume a Normal distribution, where, among other things, the data are symmetric around the mean.

Skewness measures asymmetry in the distribution - left skew (mean < median) is indicated by negative skewness values, while right skew (mean > median) is indicated by positive values. The skew value will be near zero for data that follow a Normal distribution.

7.13.1 Non-parametric Skew via $\mathbf{skew_1}$

A simpler measure of skew, sometimes called the **nonparametric skew** and closely related to Pearson's notion of median skewness, falls between -1 and +1 for any distribution. It is just the difference between the mean and the median, divided by the standard deviation.

- Values greater than +0.2 are sometimes taken to indicate fairly substantial right skew, while values below -0.2 indicate fairly substantial left skew.

```
(mean(nyfs1$bmi) - median(nyfs1$bmi))/sd(nyfs1$bmi)
```

```
[1] 0.269
```

There is a function in the `Love-boost.R` script called `skew1` that can be used to do these calculations, so long as the variable has no missing data.

```
skew1(nyfs1$bmi)
```

```
[1] 0.269
```

The Wikipedia page on skewness, from which some of this material is derived, provides definitions for several other skewness measures.

7.14 Assessing Kurtosis (Heavy-Tailedness)

Another measure of a distribution's shape that can be found in the `psych` library is the **kurtosis**. Kurtosis is an indicator of whether the distribution is heavy-tailed or light-tailed as compared to a Normal distribution. Positive kurtosis means more of the variance is due to outliers - unusual points far away from the mean relative to what we might expect from a Normally distributed data set with the same standard deviation.

- A Normal distribution will have a kurtosis value near 0, a distribution with similar tail behavior to what we would expect from a Normal is said to be *mesokurtic*
- Higher kurtosis values (meaningfully higher than 0) indicate that, as compared to a Normal distribution, the observed variance is more the result of extreme outliers (i.e. heavy tails) as opposed to being the result of more modest sized deviations from the mean. These heavy-tailed, or outlier prone, distributions are sometimes called *leptokurtic*.
- Kurtosis values meaningfully lower than 0 indicate light-tailed data, with fewer outliers than we'd expect in a Normal distribution. Such distributions are sometimes referred to as *platykurtic*, and include distributions without outliers, like the Uniform distribution.

Here's a table:

Fewer outliers than a Normal	Approximately Normal	More outliers than a Normal
Light-tailed *platykurtic* (kurtosis $<$ 0)	"Normalish" *mesokurtic* (kurtosis $=$ 0)	Heavy-tailed *leptokurtic* (kurtosis $>$ 0)

```
psych::kurtosi(nyfs1$bmi)
```

```
[1] 1.97
```

7.14.1 The Standard Error of the Sample Mean

The **standard error** of the sample mean, which is the standard deviation divided by the square root of the sample size:

```
sd(nyfs1$bmi)/sqrt(length(nyfs1$bmi))
```

```
[1] 0.108
```

7.15 The `describe` function in the `Hmisc` library

The `Hmisc` library has lots of useful functions. It's named for its main developer, Frank Harrell. The `describe` function in `Hmisc` knows enough to separate numerical from categorical variables, and give you separate (and detailed) summaries for each.

- For a categorical variable, it provides counts of total observations (n), the number of missing values, and the number of unique categories, along with counts and percentages falling in each category.
- For a numerical variable, it provides:
- counts of total observations (n), the number of missing values, and the number of unique values
- an Info value for the data, which indicates how continuous the variable is (a score of 1 is generally indicative of a completely continuous variable with no ties, while scores near 0 indicate lots of ties, and very few unique values)
- the sample Mean
- many sample percentiles (quantiles) of the data, specifically (5, 10, 25, 50, 75, 90, 95, 99)

- either a complete table of all observed values, with counts and percentages (if there are a modest number of unique values), or
- a table of the five smallest and five largest values in the data set, which is useful for range checking

```
Hmisc::describe(nyfs1)
```

```
nyfs1

 7  Variables      1416  Observations
---------------------------------------------------------------------------
subject.id
       n  missing distinct     Info     Mean      Gmd      .05      .10
    1416        0     1416        1    72703    525.3    71994    72073
     .25      .50      .75      .90      .95
   72313    72698    73096    73331    73414

lowest : 71918 71919 71921 71922 71923, highest: 73488 73489 73490 73491 73492
---------------------------------------------------------------------------
sex
       n  missing distinct
    1416        0        2

Value      Female   Male
Frequency     707    709
Proportion  0.499  0.501
---------------------------------------------------------------------------
age.exam
       n  missing distinct     Info     Mean      Gmd      .05      .10
    1416        0       14    0.994    8.855    4.235        3        4
     .25      .50      .75      .90      .95
       6        9       12       14       15

Value          3     4     5     6     7     8     9    10    11    12
Frequency     97   111   119   129   123   120    90   109   102   108
Proportion 0.069 0.078 0.084 0.091 0.087 0.085 0.064 0.077 0.072 0.076

Value         13    14    15    16
Frequency    113   104    85     6
Proportion 0.080 0.073 0.060 0.004
---------------------------------------------------------------------------
bmi
       n  missing distinct     Info     Mean      Gmd      .05      .10
    1416        0      191        1     18.8    4.321    14.30    14.80
     .25      .50      .75      .90      .95
   15.80    17.70    20.90    24.45    27.00

lowest : 11.9 12.6 12.7 12.9 13.0, highest: 34.6 34.7 35.9 37.0 38.8
---------------------------------------------------------------------------
bmi.cat
       n  missing distinct
    1416        0        4

Value         1 Underweight 2 Normal weight    3 Overweight         4 Obese
Frequency                42             926             237             211
Proportion            0.030           0.654           0.167           0.149
```

```
--------------------------------------------------------------------------------
waist.circ
       n  missing distinct     Info     Mean      Gmd      .05      .10
    1416        0      462        1    65.29    14.23    49.30    51.10
     .25      .50      .75      .90      .95
   55.00    63.00    72.93    82.35    90.40

lowest :  42.5  43.4  44.1  44.4  44.7, highest: 108.4 108.5 110.4 111.0 112.4
--------------------------------------------------------------------------------
triceps.skinfold
       n  missing distinct     Info     Mean      Gmd      .05      .10
    1416        0      236        1    13.37    6.279    6.775    7.400
     .25      .50      .75      .90      .95
   9.000   11.800   16.600   21.750   25.600

lowest :  4.0  4.6  4.9  5.0  5.2, highest: 34.3 34.8 36.0 36.2 38.2
--------------------------------------------------------------------------------
```

More on the `Info` value in Hmisc::describe is available here

7.16 `xda` from GitHub for numerical summaries for exploratory data analysis

```r
## next two commands needed if xda is not already installed

library(devtools)
install_github("ujjwalkarn/xda")
```

```
Skipping install of 'xda' from a github remote, the SHA1 (fb68f0da) has not changed since last install.
  Use `force = TRUE` to force installation
```

```r
xda::numSummary(nyfs1)
```

```
                    n     mean     sd     max     min  range nunique
subject.id       1416 72702.70 454.75 73492.0 71918.0 1574.0    1416
age.exam         1416     8.86   3.68    16.0     3.0   13.0      14
bmi              1416    18.80   4.08    38.8    11.9   26.9     191
waist.circ       1416    65.29  12.85   112.4    42.5   69.9     462
triceps.skinfold 1416    13.37   5.83    38.2     4.0   34.2     236
                 nzeros   iqr lowerbound upperbound noutlier kurtosis
subject.id            0 784.0   71136.75    74272.2        0   -1.193
age.exam              0   6.0      -3.00       21.0        0   -1.198
bmi                   0   5.1       8.15       28.5       53    1.973
waist.circ            0  17.9      28.15       99.8       22    0.384
triceps.skinfold      0   7.6      -2.40       28.0       31    1.149
                 skewness    mode miss miss%      1%       5%      25%
subject.id        0.00815 71918.0    0     0 71933.1 71993.75 72312.8
age.exam          0.08202     6.0    0     0     3.0     3.00      6.0
bmi               1.34804    15.5    0     0    13.5    14.30     15.8
waist.circ        0.85106    55.4    0     0    46.1    49.30     55.0
triceps.skinfold  1.15791     8.0    0     0     5.6     6.77      9.0
                     50%     75%     95%   99%
subject.id       72697.5 73096.2 73414.2 73478
```

```
age.exam              9.0    12.0    15.0    15
bmi                  17.7    20.9    27.0    32
waist.circ           63.0    72.9    90.4   102
triceps.skinfold     11.8    16.6    25.6    31
```

Most of the elements of this `numSummary` should be familiar. Some new pieces include:

- `nunique` = number of unique values
- `nzeroes` = number of zeroes
- `noutlier` = number of outliers (using a standard that isn't entirely transparent to me)
- `miss` = number of rows with missing value
- `miss%` = percentage of total rows with missing values ((miss/n)*100)
- `5%` = 5th percentile value of that variable (value below which 5 percent of the observations may be found)

```
xda::charSummary(nyfs1)
```

```
           n miss miss% unique
sex     1416    0     0      2
bmi.cat 1416    0     0      4
                                                         top5levels:count
sex                                                  Male:709, Female:707
bmi.cat 2 Normal weight:926, 3 Overweight:237, 4 Obese:211, 1 Underweight:42
```

The `top5levels:count` provides the top 5 unique values for each variable, sorted by their counts.

7.17 What Summaries to Report

It is usually helpful to focus on the shape, center and spread of a distribution. Bock, Velleman and DeVeaux provide some useful advice:

- If the data are skewed, report the median and IQR (or the three middle quantiles). You may want to include the mean and standard deviation, but you should point out why the mean and median differ. The fact that the mean and median do not agree is a sign that the distribution may be skewed. A histogram will help you make that point.
- If the data are symmetric, report the mean and standard deviation, and possibly the median and IQR as well.
- If there are clear outliers and you are reporting the mean and standard deviation, report them with the outliers present and with the outliers removed. The differences may be revealing. The median and IQR are not likely to be seriously affected by outliers.

Chapter 8

Assessing Normality

Data are well approximated by a Normal distribution if the shape of the data's distribution is a good match for a Normal distribution with mean and standard deviation equal to the sample statistics.

- the data are symmetrically distributed about a single peak, located at the sample mean
- the spread of the distribution is well characterized by a Normal distribution with standard deviation equal to the sample standard deviation
- the data show outlying values (both in number of candidate outliers, and size of the distance between the outliers and the center of the distribution) that are similar to what would be predicted by a Normal model.

We have several tools for assessing Normality of a single batch of data, including:

- a histogram with superimposed Normal distribution
- histogram variants (like the boxplot) which provide information on the center, spread and shape of a distribution
- the Empirical Rule for interpretation of a standard deviation
- a specialized *normal Q-Q plot* (also called a normal probability plot or normal quantile-quantile plot) designed to reveal differences between a sample distribution and what we might expect from a normal distribution of a similar number of values with the same mean and standard deviation

8.1 Empirical Rule Interpretation of the Standard Deviation

For a set of measurements that follows a Normal distribution, the interval:

- Mean $\pm$ Standard Deviation contains approximately 68% of the measurements;
- Mean $\pm$ 2(Standard Deviation) contains approximately 95% of the measurements;
- Mean $\pm$ 3(Standard Deviation) contains approximately all (99.7%) of the measurements.

Again, most data sets do not follow a Normal distribution. We will occasionally think about transforming or re-expressing our data to obtain results which are better approximated by a Normal distribution, in part so that a standard deviation can be more meaningful.

For the BMI data we have been studying, here again are some summary statistics…

```
mosaic::favstats(nyfs1$bmi)
```

```
  min   Q1 median   Q3  max mean   sd    n missing
 11.9 15.8   17.7 20.9 38.8 18.8 4.08 1416       0
```

The mean is 18.8 and the standard deviation is 4.08, so if the data really were Normally distributed, we'd expect to see:

- About 68% of the data in the range (14.72, 22.88). In fact, 1074 of the 1416 BMI values are in this range, or 75.8%.
- About 95% of the data in the range (10.64, 26.96). In fact, 1344 of the 1416 BMI values are in this range, or 94.9%.
- About 99.7% of the data in the range (6.56, 31.04). In fact, 1393 of the 1416 BMI values are in this range, or 98.4%.

So, based on this Empirical Rule approximation, do the BMI data seem to be well approximated by a Normal distribution?

8.2 Describing Outlying Values with Z Scores

The maximum body-mass index value here is 38.8. One way to gauge how extreme this is (or how much of an outlier it is) uses that observation's **Z score**, the number of standard deviations away from the mean that the observation falls.

Here, the maximum value, 38.8 is 4.9 standard deviations above the mean, and thus has a Z score of 4.9.

A negative Z score would indicate a point below the mean, while a positive Z score indicates, as we've seen, a point above the mean. The minimum body-mass index, 11.9 is 1.69 standard deviations *below* the mean, so it has a Z score of -1.7.

Recall that the Empirical Rule suggests that if a variable follows a Normal distribution, it would have approximately 95% of its observations falling inside a Z score of (-2, 2), and 99.74% falling inside a Z score range of (-3, 3).

8.2.1 Fences and Z Scores

Note the relationship between the fences (Tukey's approach to identifying points which fall within the whiskers of a boxplot, as compared to candidate outliers) and the Z scores.

The upper inner fence in this case falls at 28.55, which indicates a Z score of 2.4, while the lower inner fence falls at 8.15, which indicates a Z score of -2.6. It is neither unusual nor inevitable for the inner fences to fall at Z scores near -2.0 and +2.0.

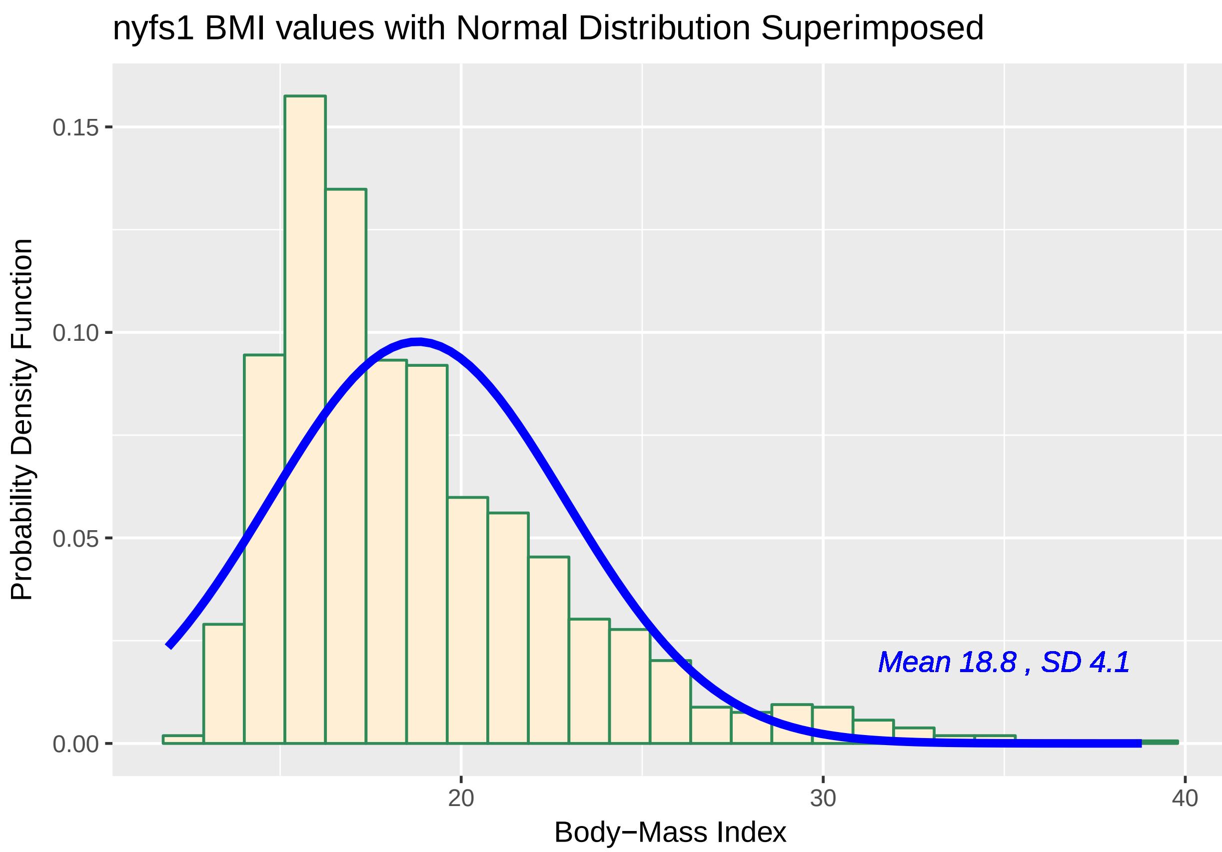
8.3 Comparing a Histogram to a Normal Distribution

Most of the time, when we want to understand whether our data are well approximated by a Normal distribution, we will use a graph to aid in the decision.

One option is to build a histogram with a Normal density function (with the same mean and standard deviation as our data) superimposed. This is one way to help visualize deviations between our data and what might be expected from a Normal distribution.

```
ggplot(nyfs1, aes(x=bmi)) +
    geom_histogram(aes(y = ..density..), bins=25, fill = "papayawhip", color = "seagreen") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(nyfs1$bmi), sd = sd(nyfs1$bmi)),
                  lwd = 1.5, col = "blue") +
    geom_text(aes(label = paste("Mean", round(mean(nyfs1$bmi),1),
                               ", SD", round(sd(nyfs1$bmi),1))),
```

```
           x = 35, y = 0.02, color="blue", fontface = "italic") +
    labs(title = "nyfs1 BMI values with Normal Distribution Superimposed",
         x = "Body-Mass Index", y = "Probability Density Function")
```



Does it seem as though the Normal model (as shown in the blue density curve) is an effective approximation to the observed distribution shown in the bars of the histogram?

We'll return shortly to the questions:

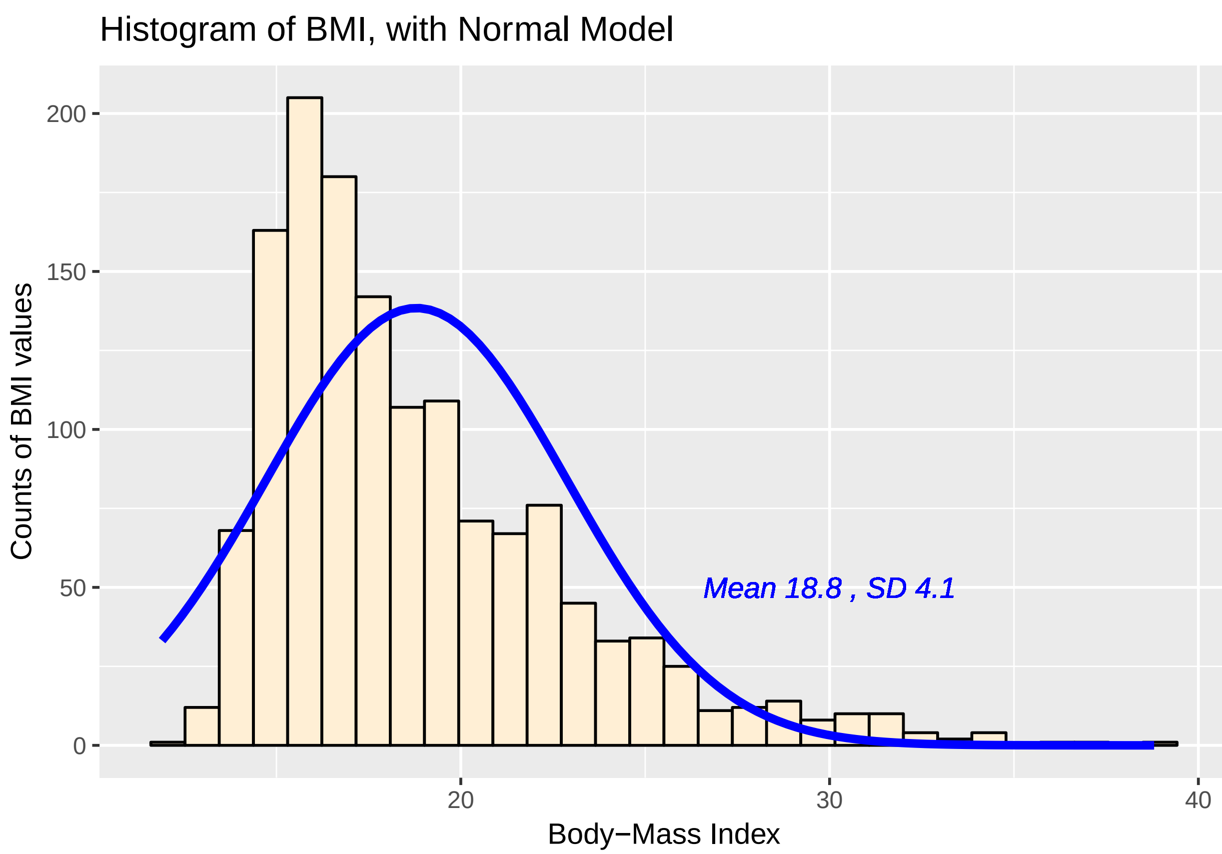
- Does a Normal distribution model fit our data well? *and*
- If the data aren't Normal, but we want to use a Normal model anyway, what should we do?

8.3.1 Histogram of BMI with Normal model (with Counts)

But first, we'll demonstrate an approach to building a histogram of counts (rather than a probability density) and then superimposing a Normal model.

```
## ggplot of counts of bmi with Normal model superimposed
## Source: https://stat.ethz.ch/pipermail/r-help//2009-September/403220.html

ggplot(nyfs1, aes(x = bmi)) +
    geom_histogram(bins = 30, fill = "papayawhip", color = "black") +
    stat_function(fun = function(x, mean, sd, n)
        n * dnorm(x = x, mean = mean, sd = sd),
        args = with(nyfs1,
                    c(mean = mean(bmi), sd = sd(bmi), n = length(bmi))),
        col = "blue", lwd = 1.5) +
```

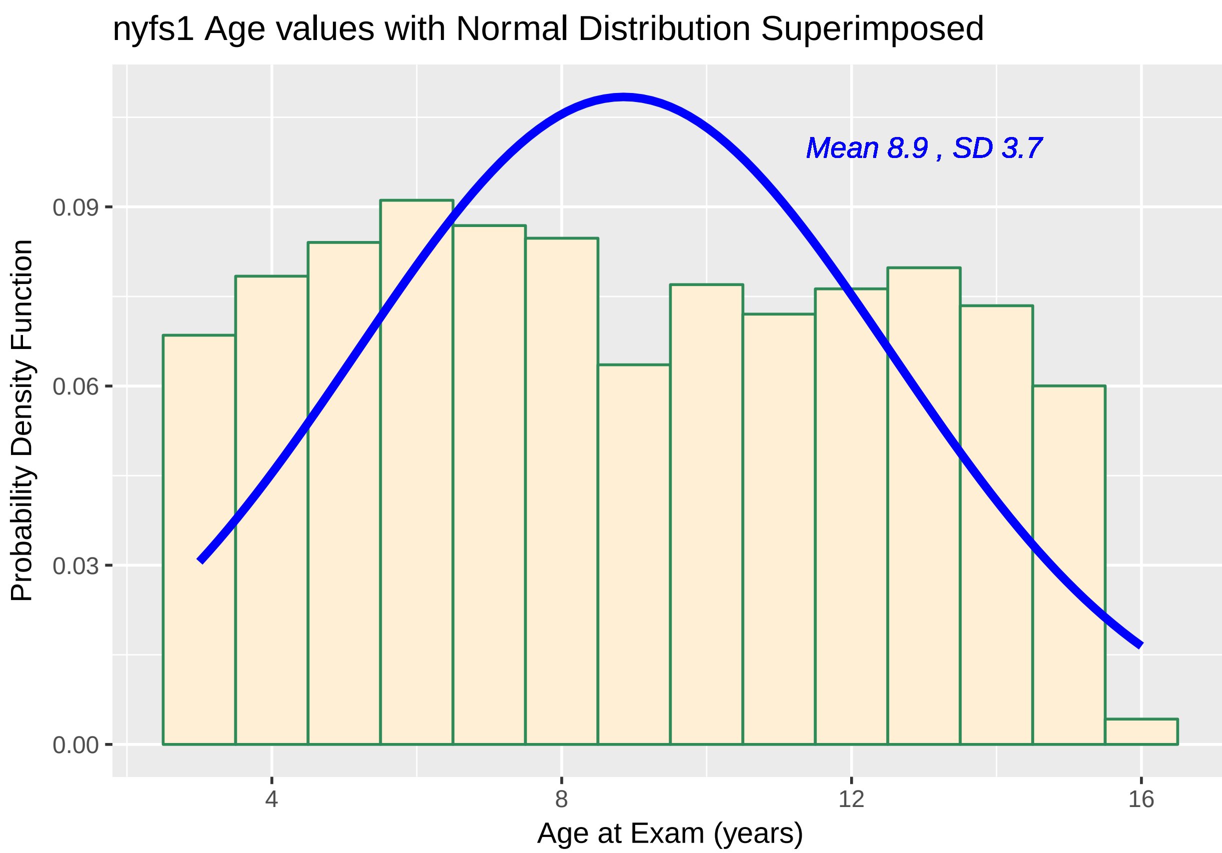
```
    geom_text(aes(label = paste("Mean", round(mean(nyfs1$bmi),1),
                                ", SD", round(sd(nyfs1$bmi),1))),
              x = 30, y = 50, color="blue", fontface = "italic") +
    labs(title = "Histogram of BMI, with Normal Model",
         x = "Body-Mass Index", y = "Counts of BMI values")
```



8.4 Does a Normal model work well for the Ages?

Now, suppose we instead look at the `age.exam` data. Do these data appear to follow a Normal distribution?

```
ggplot(nyfs1, aes(x=age.exam)) +
    geom_histogram(aes(y = ..density..), binwidth=1,
                   fill = "papayawhip", color = "seagreen") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(nyfs1$age.exam),
                              sd = sd(nyfs1$age.exam)),
                  lwd = 1.5, col = "blue") +
    geom_text(aes(label = paste("Mean", round(mean(nyfs1$age.exam),1),
                                ", SD", round(sd(nyfs1$age.exam),1))),
              x = 13, y = 0.1, color="blue", fontface = "italic") +
    labs(title = "nyfs1 Age values with Normal Distribution Superimposed",
         x = "Age at Exam (years)", y = "Probability Density Function")
```




```
mosaic::favstats(nyfs1$age.exam)
```

```
 min Q1 median Q3 max mean   sd    n missing
   3  6      9 12  16 8.86 3.68 1416       0
```

The mean is 8.86 and the standard deviation is 3.68 so if the `age.exam` data really were Normally distributed, we'd expect to see: - About 68% of the data in the range (5.17, 12.54). In fact, 781 of the 1416 Age values are in this range, or 55.2%. - About 95% of the data in the range (1.49, 16.22). In fact, 1416 of the 1416 Age values are in this range, or 100%. - About 99.7% of the data in the range (-2.19, 19.9). In fact, 1416 of the 1416 Age values are in this range, or 100%.

How does the Normal approximation work for age, according to the Empirical Rule?

There is a function in the `Love-boost.R` script called `Emp_Rule` that can be used to do these calculations, so long as the variable has no missing data.

```
Emp_Rule(nyfs1$bmi)
```

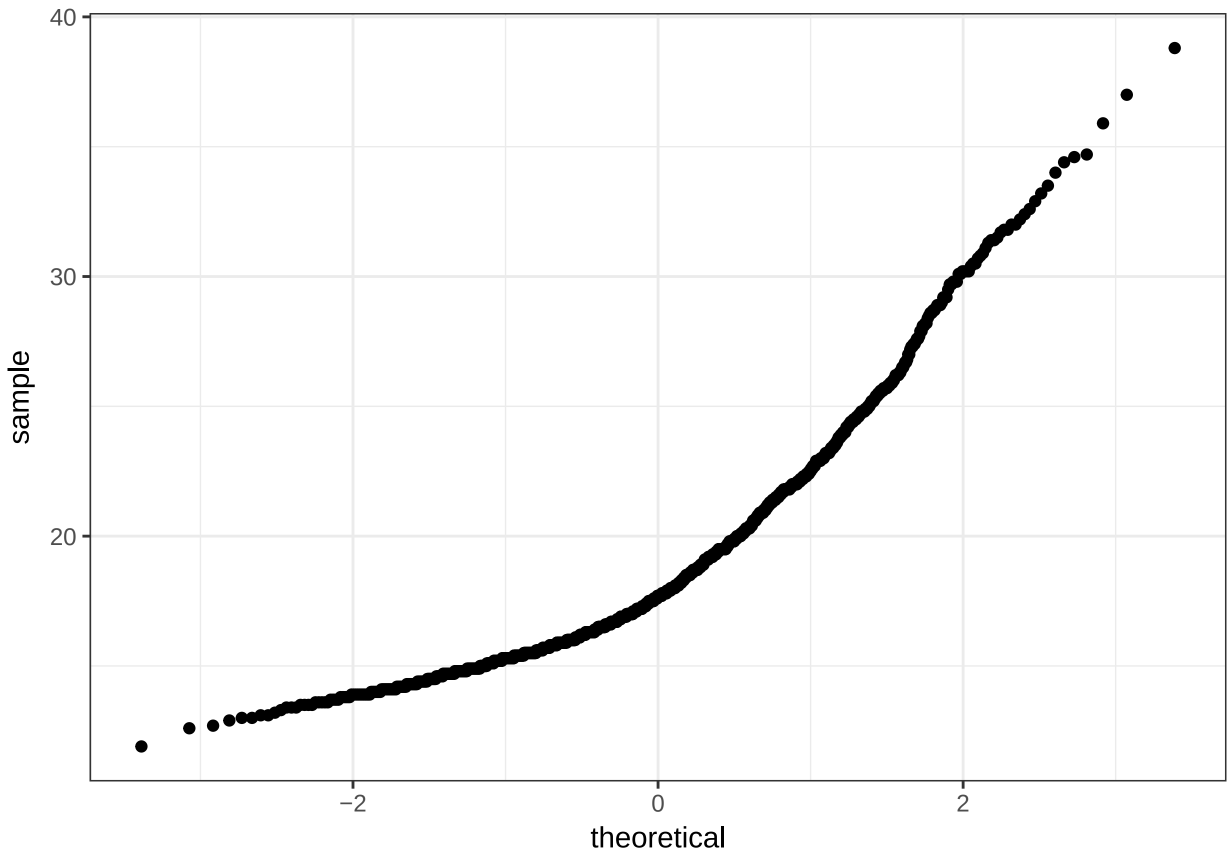
	count	proportion
Mean +/- 1 SD	1074	0.7585
Mean +/- 2 SD	1344	0.9492
Mean +/- 3 SD	1393	0.9838
Entire Data Set	1416	1

8.5 The Normal Q-Q Plot

A normal probability plot (or normal quantile-quantile plot) of the BMI results from the `nyfs1` data, developed using `ggplot2` is shown below. In this case, this is a picture of 1416 BMI results. The idea of a normal Q-Q plot is that it plots the observed sample values (on the vertical axis) and then, on the horizontal, the expected or theoretical quantiles that would be observed in a standard normal distribution (a Normal distribution with mean 0 and standard deviation 1) with the same number of observations.

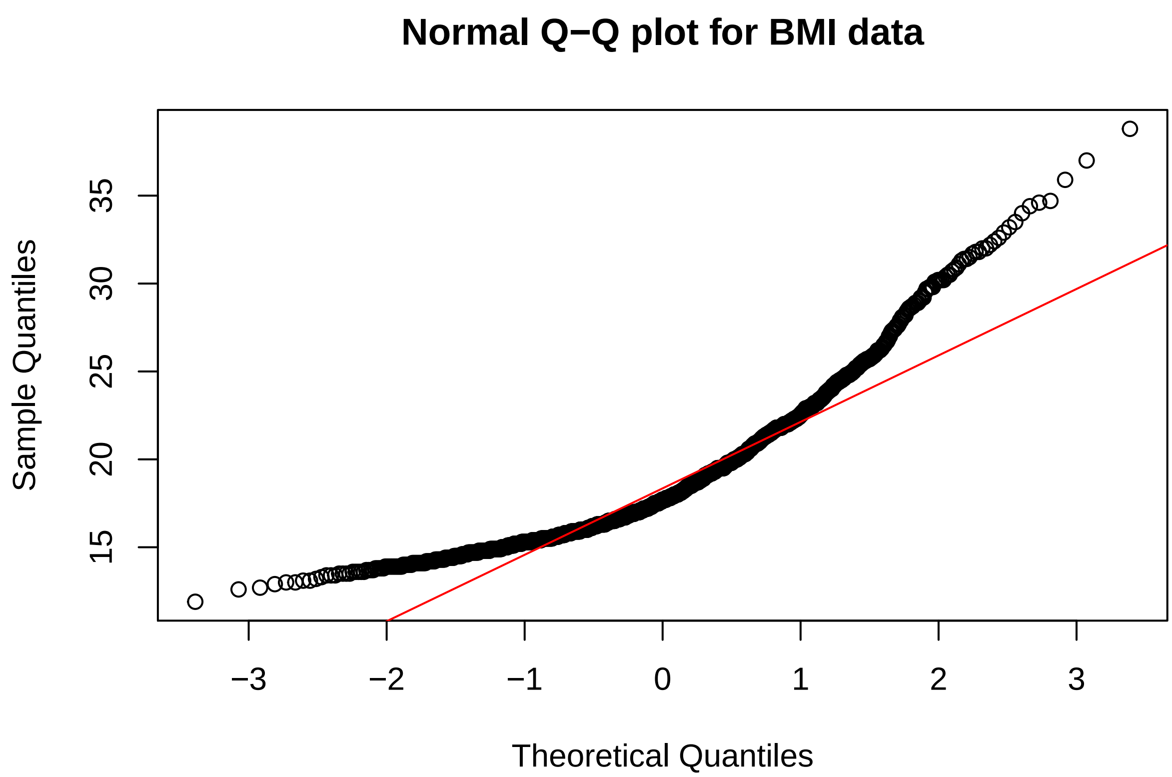
A Normal Q-Q plot will follow a straight line when the data are (approximately) Normally distributed. When the data have a different shape, the plot will reflect that.

```
ggplot(nyfs1, aes(sample = bmi)) +
    geom_point(stat="qq") +
    theme_bw() # eliminate the gray background
```



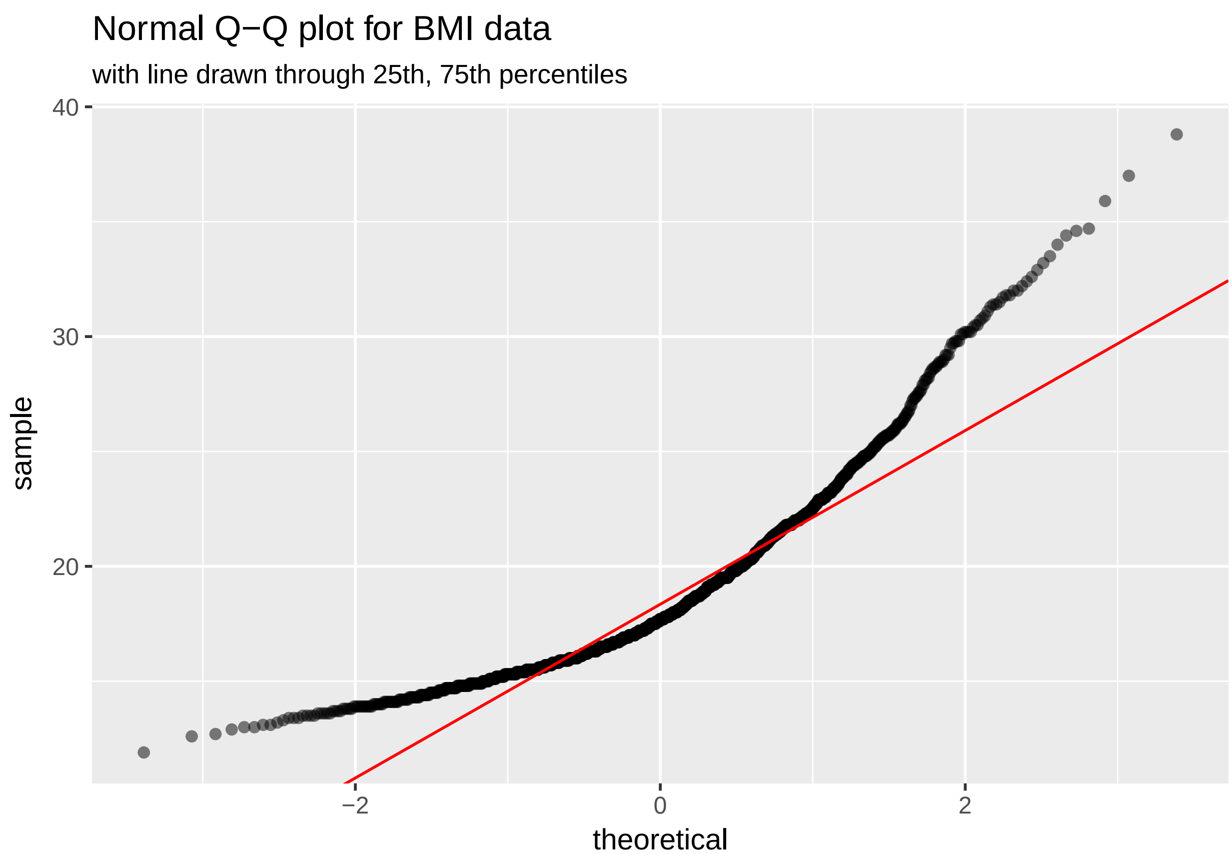
This is a case where the base graphics approach may be preferable.

```
qqnorm(nyfs1$bmi, main = "Normal Q-Q plot for BMI data")
qqline(nyfs1$bmi, col = "red")
```



It is possible to get the base graphics result with `ggplot2`, though. For example, we might use this modification of a response at https://stackoverflow.com/questions/4357031/qqnorm-and-qqline-in-ggplot2/

```
dat <- nyfs1 %>% filter(complete.cases(bmi))
y <- quantile(dat$bmi, c(0.25, 0.75))
x <- qnorm(c(0.25, 0.75))
slope <- diff(y)/diff(x)
int <- y[1L] - slope * x[1L]

ggplot(nyfs1, aes(sample = bmi)) +
geom_qq(alpha = 0.5) +
geom_abline(slope = slope, intercept = int, col = "red") +
labs(title = "Normal Q-Q plot for BMI data",
subtitle = "with line drawn through 25th, 75th percentiles")
```




```
rm(x, y, slope, int, dat)
```

8.6 Interpreting the Normal Q-Q Plot

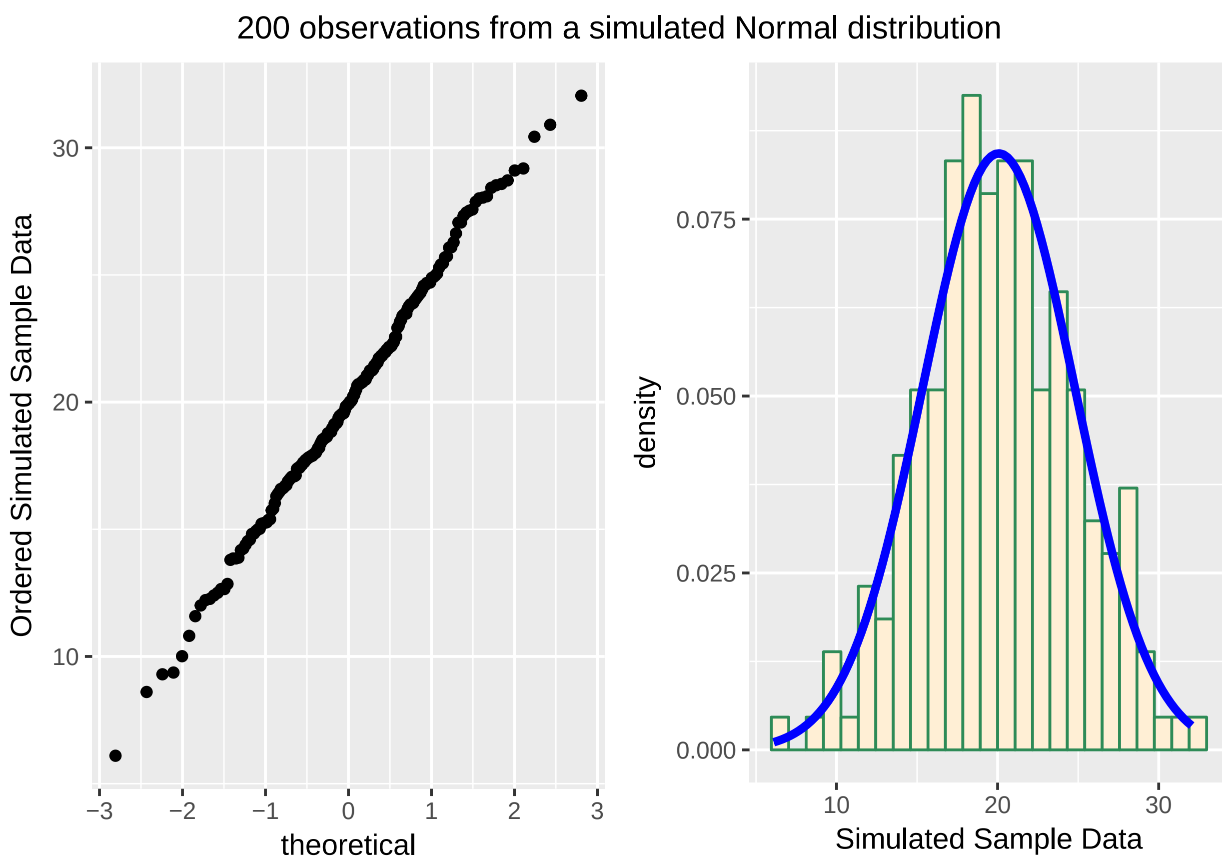
The purpose of a Normal Q-Q plot is to help point out distinctions from a Normal distribution. A Normal distribution is symmetric and has certain expectations regarding its tails. The Normal Q-Q plot can help us identify data as - well approximated by a Normal distribution, or not because of - skew (including distinguishing between right skew and left skew) - behavior in the tails (which could be heavy-tailed [more outliers than expected] or light-tailed)

8.6.1 Data from a Normal distribution shows up as a straight line in a Normal Q-Q plot

We'll demonstrate the looks that we can obtain from a Normal Q-Q plot in some simulations. First, here is an example of a Normal Q-Q plot, and its associated histogram, for a sample of 200 observations simulated from a Normal distribution.

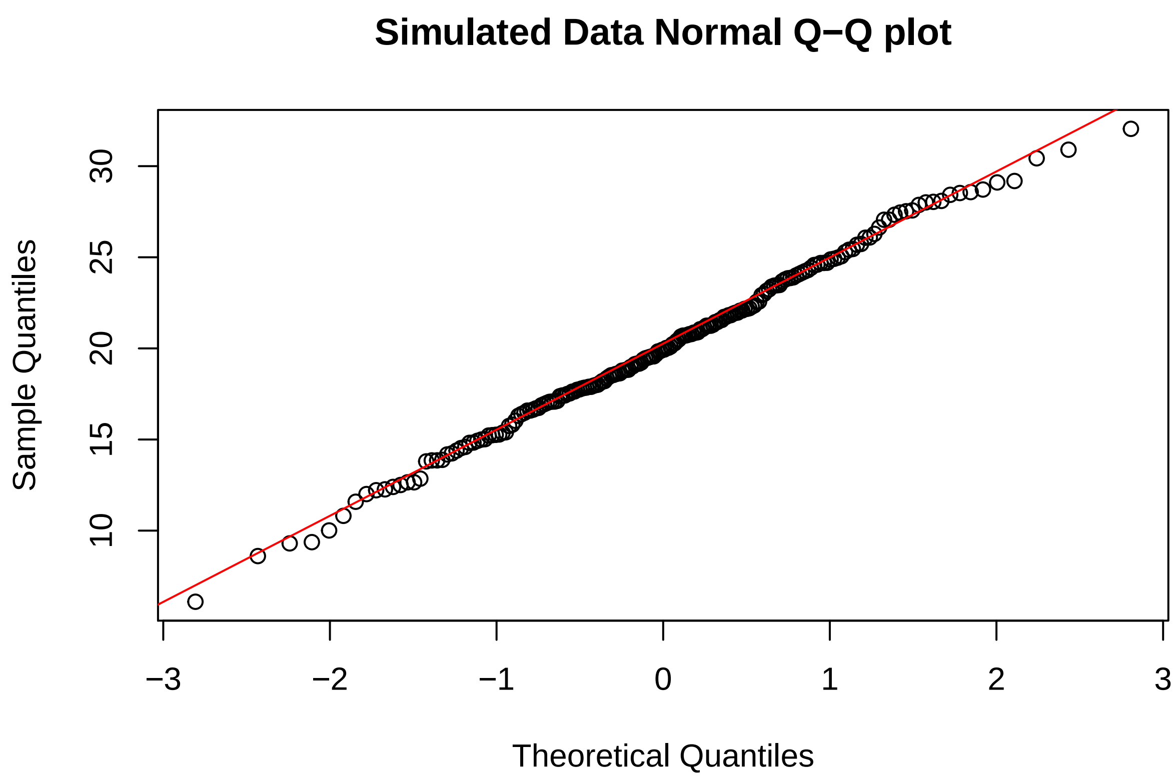
```
set.seed(123431) # so the results can be replicated

# simulate 200 observations from a Normal(20, 5) distribution and place them
# in the d variable within the temp.1 data frame
temp.1 <- data.frame(d = rnorm(200, mean = 20, sd = 5))

# left plot - basic Normal Q-Q plot of simulated data
```

```
p1 <- ggplot(temp.1, aes(sample = d)) +
    geom_point(stat="qq") +
    labs(y = "Ordered Simulated Sample Data")

# right plot - histogram with superimposed normal distribution
p2 <- ggplot(temp.1, aes(x = d)) +
    geom_histogram(aes(y = ..density..),
                   bins=25, fill = "papayawhip", color = "seagreen") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(temp.1$d),
                              sd = sd(temp.1$d)),
                  lwd = 1.5, col = "blue") +
    labs(x = "Simulated Sample Data")

gridExtra::grid.arrange(p1, p2, ncol=2,
             top ="200 observations from a simulated Normal distribution")
```



And here is another look at that same simulated data...

```
qqnorm(temp.1$d, main = "Simulated Data Normal Q-Q plot")
qqline(temp.1$d, col = "red")
```

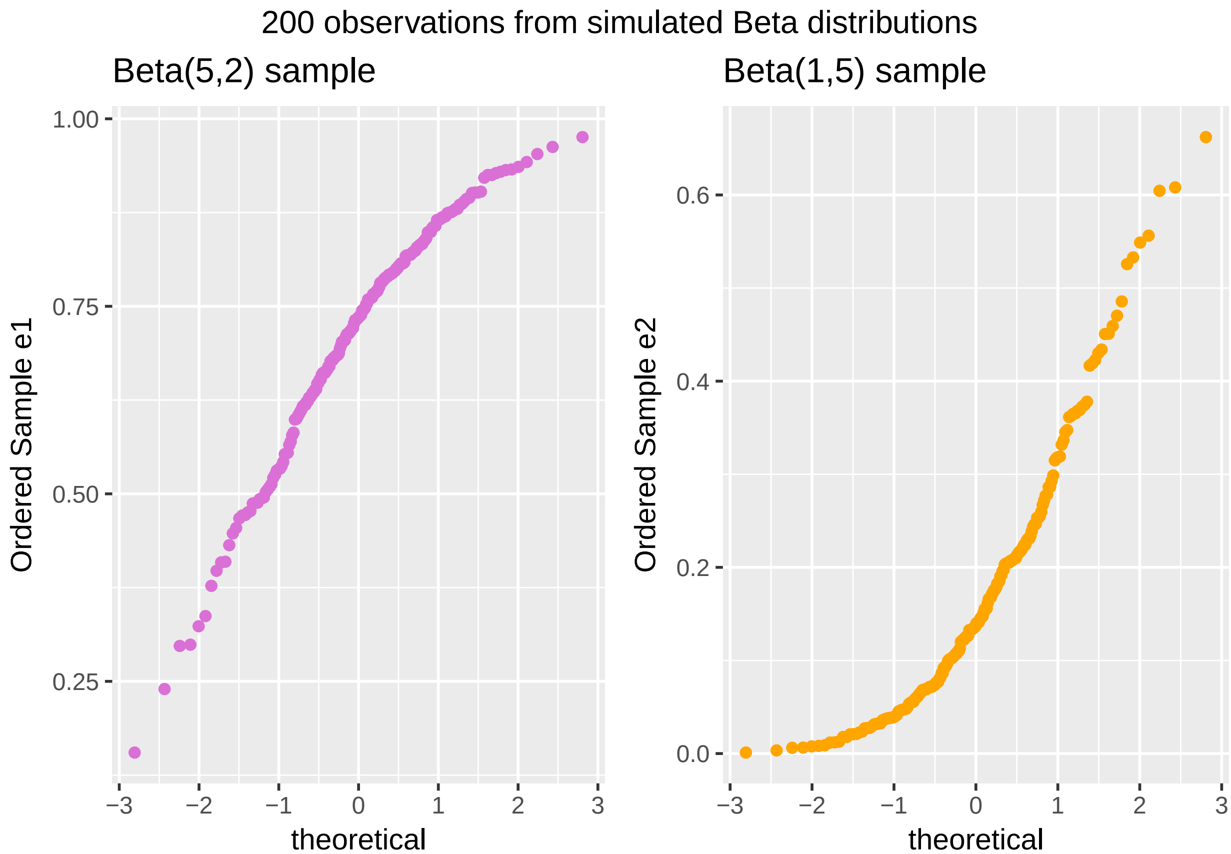


So, what are the characteristics of this simulation? The data appear to be well-modeled by the Normal distribution, because: - the points on the Normal Q-Q plot follow a straight line, in particular - there is no substantial curve (such as we'd see with data that were skewed) - there is no particularly surprising behavior (curves away from the line) at either tail, so there's no obvious problem with outliers

8.6.2 Skew is indicated by monotonic curves in the Normal Q-Q plot

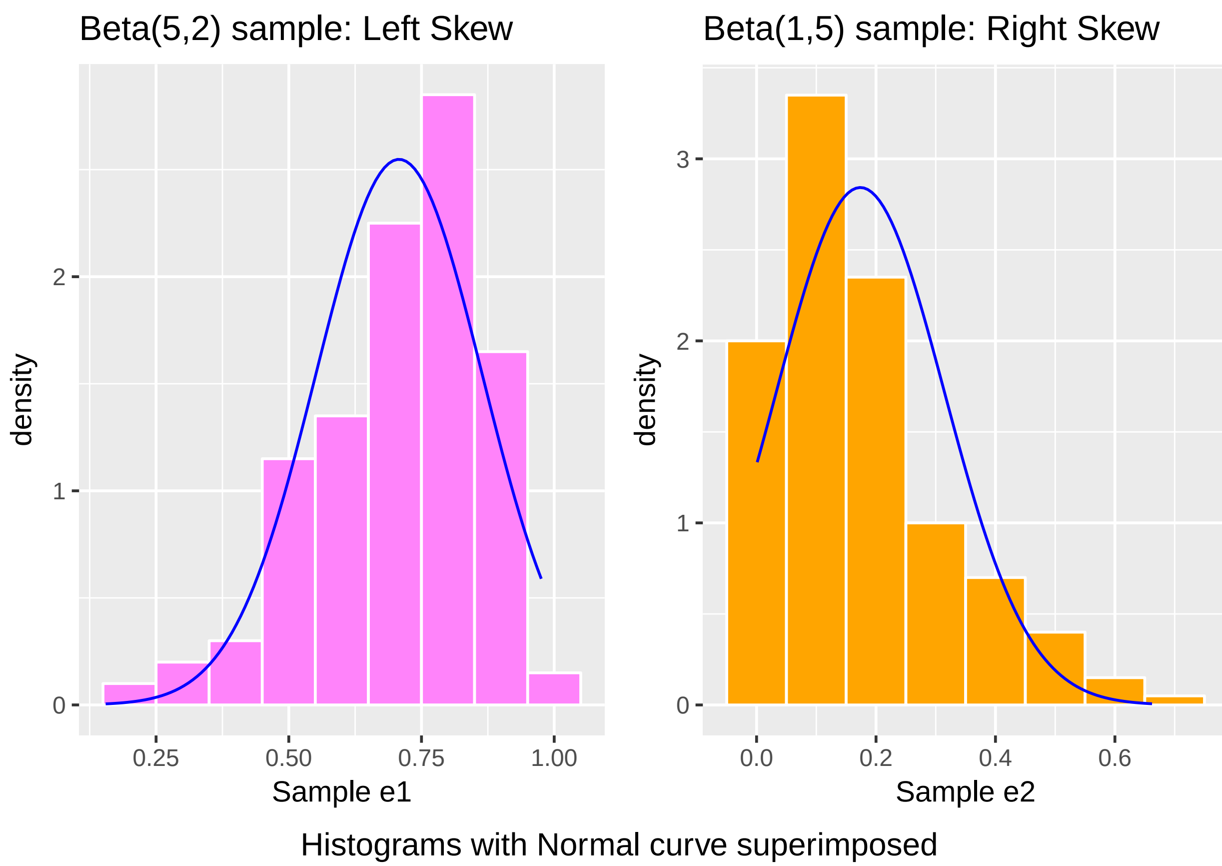
Data that come from a skewed distribution appear to curve away from a straight line in the Q-Q plot.

```
set.seed(123431) # so the results can be replicated

# simulate 200 observations from a beta(5, 2) distribution into the e1 variable
# simulate 200 observations from a beta(1, 5) distribution into the e2 variable
temp.2 <- data.frame(e1 = rbeta(200, 5, 2), e2 = rbeta(200, 1, 5))

p1 <- ggplot(temp.2, aes(sample = e1)) +
    geom_point(stat="qq", color = "orchid") +
    labs(y = "Ordered Sample e1", title = "Beta(5,2) sample")

p2 <- ggplot(temp.2, aes(sample = e2)) +
    geom_point(stat="qq", color = "orange") +
    labs(y = "Ordered Sample e2", title = "Beta(1,5) sample")

gridExtra::grid.arrange(p1, p2, ncol=2, top ="200 observations from simulated Beta distributions")
```



Note the bends away from a straight line in each sample. The non-Normality may be easier to see in a histogram.

```
p1 <- ggplot(temp.2, aes(x = e1)) +
    geom_histogram(aes(y = ..density..),
                   binwidth=0.1, fill = "orchid1", color = "white") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(temp.2$e1),
                              sd = sd(temp.2$e1)),
                  col = "blue") +
labs(x = "Sample e1", title = "Beta(5,2) sample: Left Skew")

p2 <- ggplot(temp.2, aes(x = e2)) +
    geom_histogram(aes(y = ..density..),
                   binwidth=0.1, fill = "orange1", color = "white") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(temp.2$e2),
                              sd = sd(temp.2$e2)),
                  col = "blue") +
labs(x = "Sample e2", title = "Beta(1,5) sample: Right Skew")

gridExtra::grid.arrange(p1, p2, ncol=2,
bottom ="Histograms with Normal curve superimposed")
```



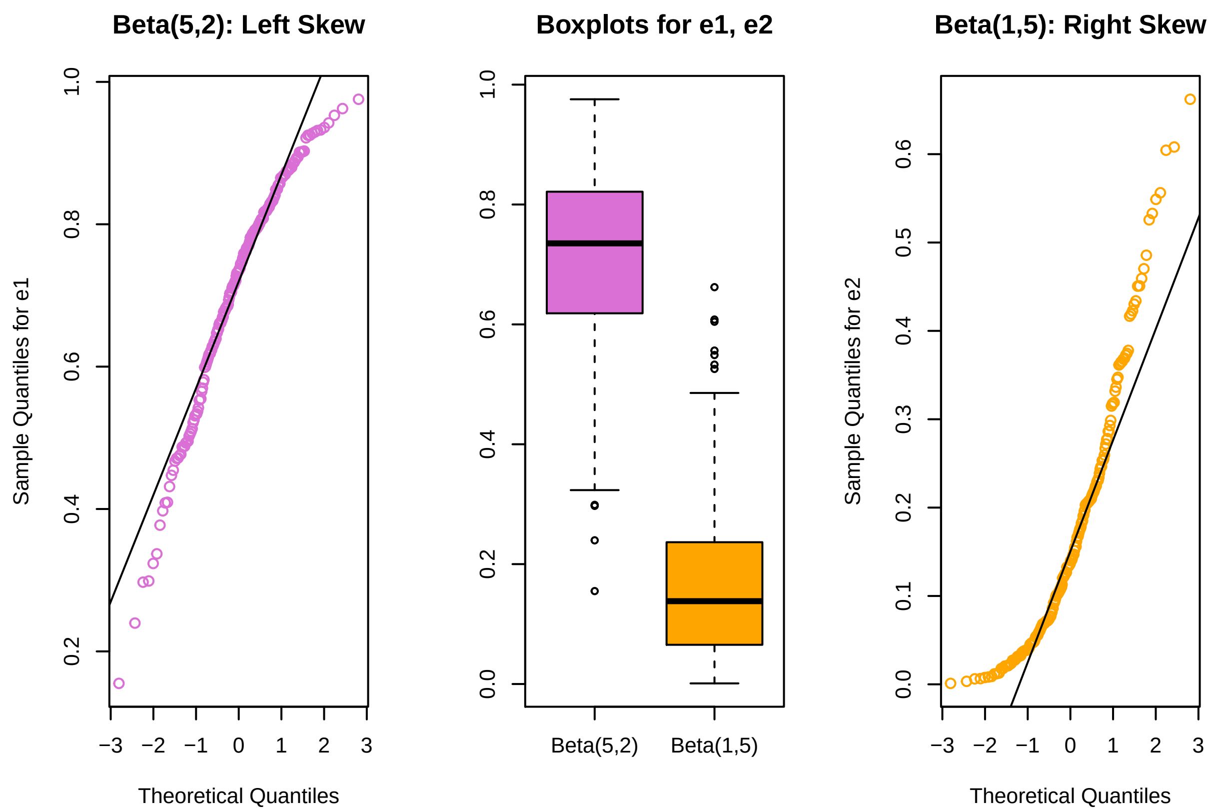
8.6.3 Direction of Skew

In each of these pairs of plots, we see the same basic result.

- The left plot (for data e1) shows left skew, with a longer tail on the left hand side and more clustered data at the right end of the distribution.
- The right plot (for data e2) shows right skew, with a longer tail on the right hand side, the mean larger than the median, and more clustered data at the left end of the distribution.

You may want to see the lines to help you see what's happening in the Q-Q plots. You can do this with our fancy approach, or with the qqnorm-qqline combination from base R.

```
par(mfrow=c(1,3))
qqnorm(temp.2$e1, col="orchid", main="Beta(5,2): Left Skew",
       ylab="Sample Quantiles for e1")
qqline(temp.2$e1)
boxplot(temp.2$e1, temp.2$e2, names=c("Beta(5,2)", "Beta(1,5)"),
        col=c("orchid", "orange"), main="Boxplots for e1, e2")
qqnorm(temp.2$e2, col="orange", main="Beta(1,5): Right Skew",
       ylab="Sample Quantiles for e2")
qqline(temp.2$e2)
```

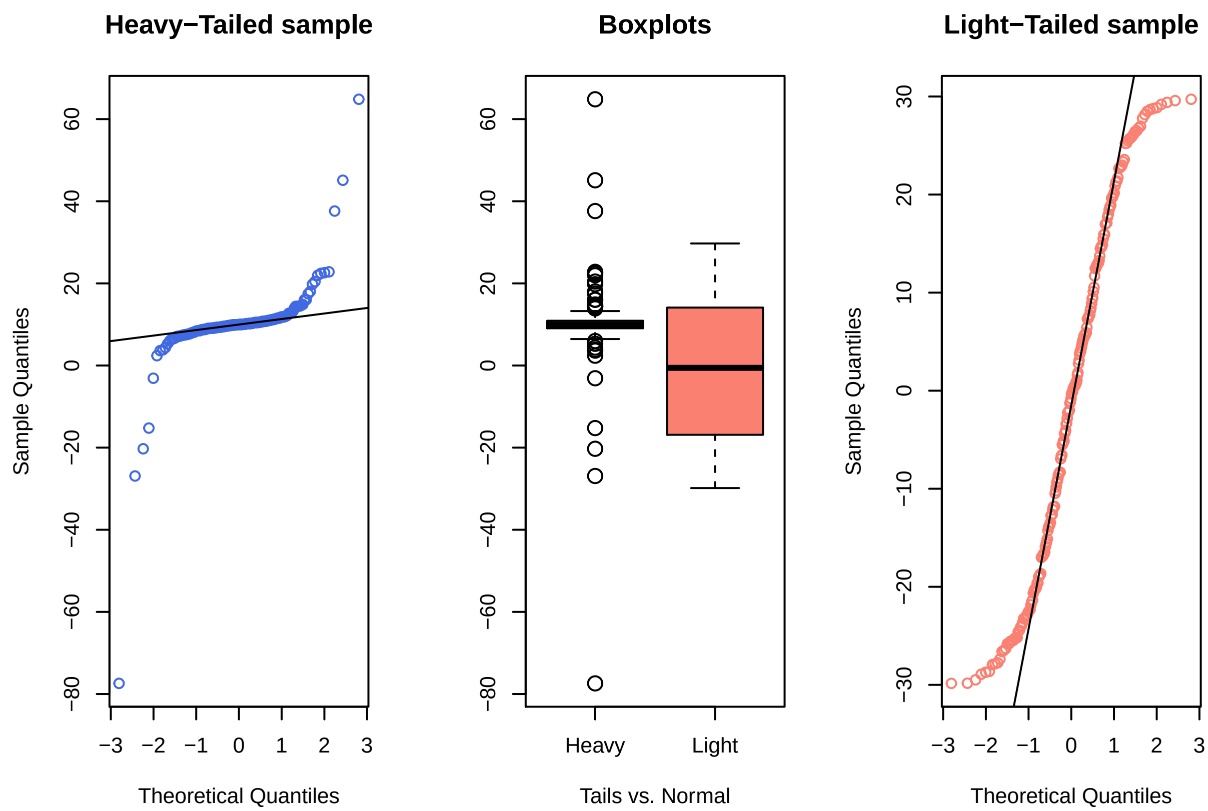



```
par(mfrow=c(1,1))
```

8.6.4 Outlier-proneness is indicated by "s-shaped" curves in a Normal Q-Q plot

- Heavy-tailed but symmetric distributions are indicated by reverse "S"-shapes, as shown on the left below.
- Light-tailed but symmetric distributions are indicated by "S" shapes in the plot, as shown on the right below.

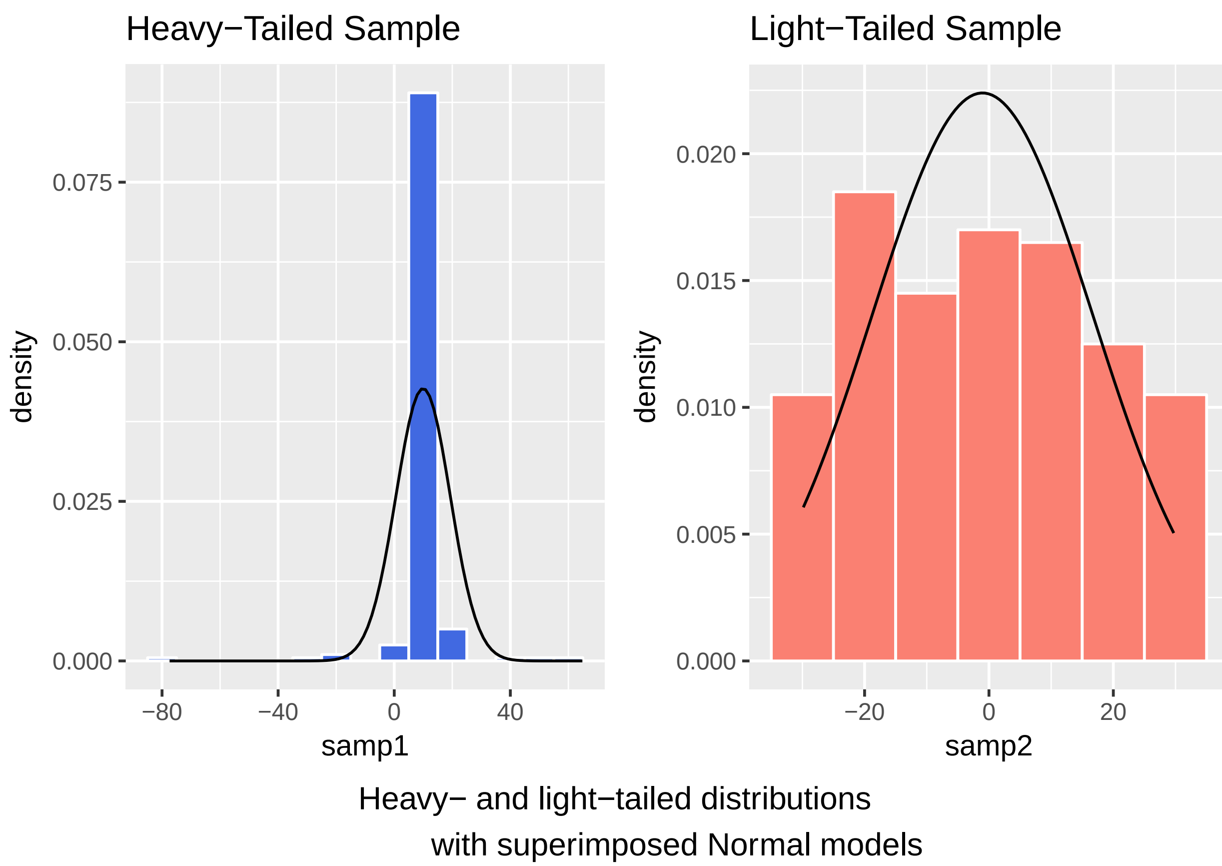
```
set.seed(4311)
# sample 200 observations from each of two probability distributions
samp1 <- rcauchy(200, location=10, scale = 1) # use a Cauchy distribution
samp2 <- runif(200, -30, 30) # a uniform distribution on (-30, 30)

par(mfrow=c(1,3))  ## set up plot window for one row, three columns
qqnorm(samp1, col="royalblue", main="Heavy-Tailed sample")
qqline(samp1)
boxplot(samp1, samp2, names=c("Heavy", "Light"), cex=1.5,
        col=c("royalblue", "salmon"), main="Boxplots",
        xlab="Tails vs. Normal")
qqnorm(samp2, col="salmon", main="Light-Tailed sample")
qqline(samp2)
```




```
par(mfrow=c(1,1))  ## return to usual plot window
```

And, we can verify these initial conclusions with histograms.

```
temp.3 <- data.frame(samp1, samp2)
p1 <- ggplot(temp.3, aes(x = samp1)) +
    geom_histogram(aes(y = ..density..),
                   binwidth=10, fill = "royalblue", color = "white") +
    stat_function(fun = dnorm, args = list(mean = mean(temp.3$samp1),
                                           sd = sd(temp.3$samp1))) +
    labs(x = "samp1", title = "Heavy-Tailed Sample")

p2 <- ggplot(temp.3, aes(x = samp2)) +
    geom_histogram(aes(y = ..density..),
                   binwidth=10, fill = "salmon", color = "white") +
    stat_function(fun = dnorm, args = list(mean = mean(temp.3$samp2),
                                           sd = sd(temp.3$samp2))) +
    labs(x = "samp2", title = "Light-Tailed Sample")

gridExtra::grid.arrange(p1, p2, ncol=2,
             bottom ="Heavy- and light-tailed distributions
             with superimposed Normal models")
```

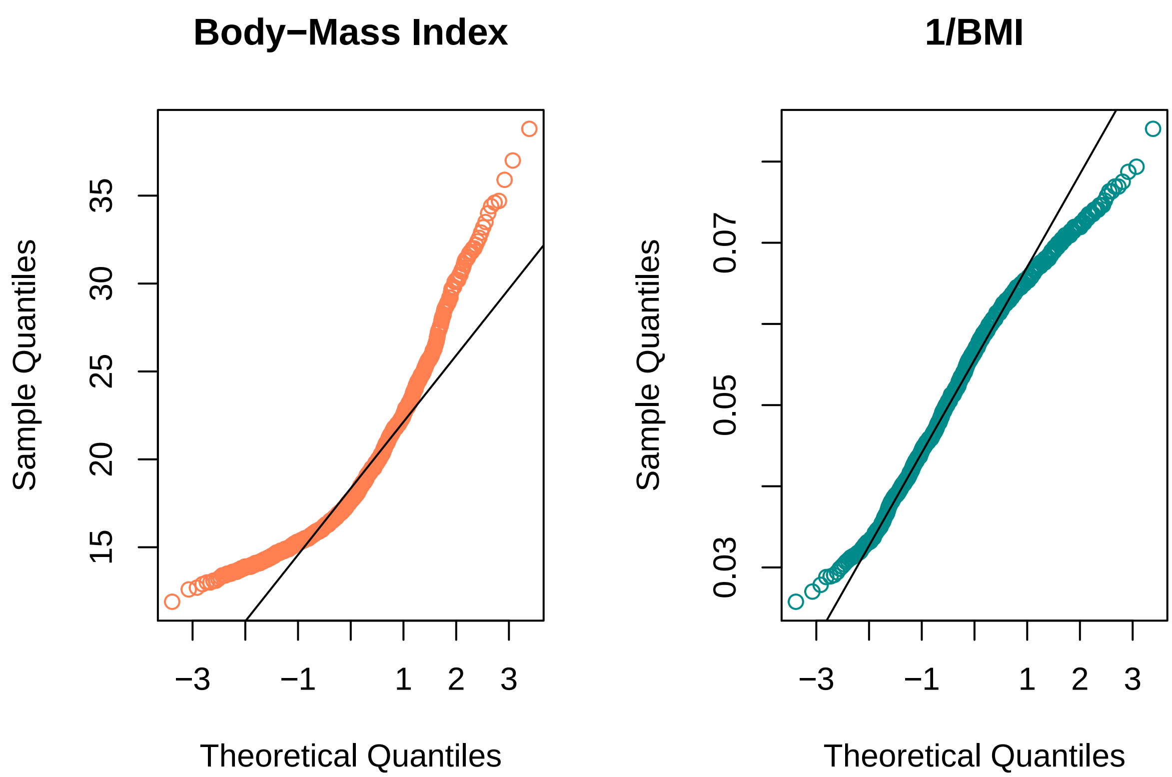



```
rm(samp1, samp2, temp.1, temp.2, temp.3, p1, p2) # cleaning up
```

8.7 Does a Normal Distribution Fit the `nyfs1` Data Well?

- Skewness is indicated by curves in the Normal Q-Q plot. Compare these two plots - the left is the original BMI data from the NYFS data frame, and the right plot shows the inverse of those values.

```
par(mfrow=c(1,2))  ## set up plot window for one row, two columns
qqnorm(nyfs1$bmi, main="Body-Mass Index", col="coral")
qqline(nyfs1$bmi)
qqnorm(1/(nyfs1$bmi), main="1/BMI", col="darkcyan")
qqline(1/nyfs1$bmi)
```

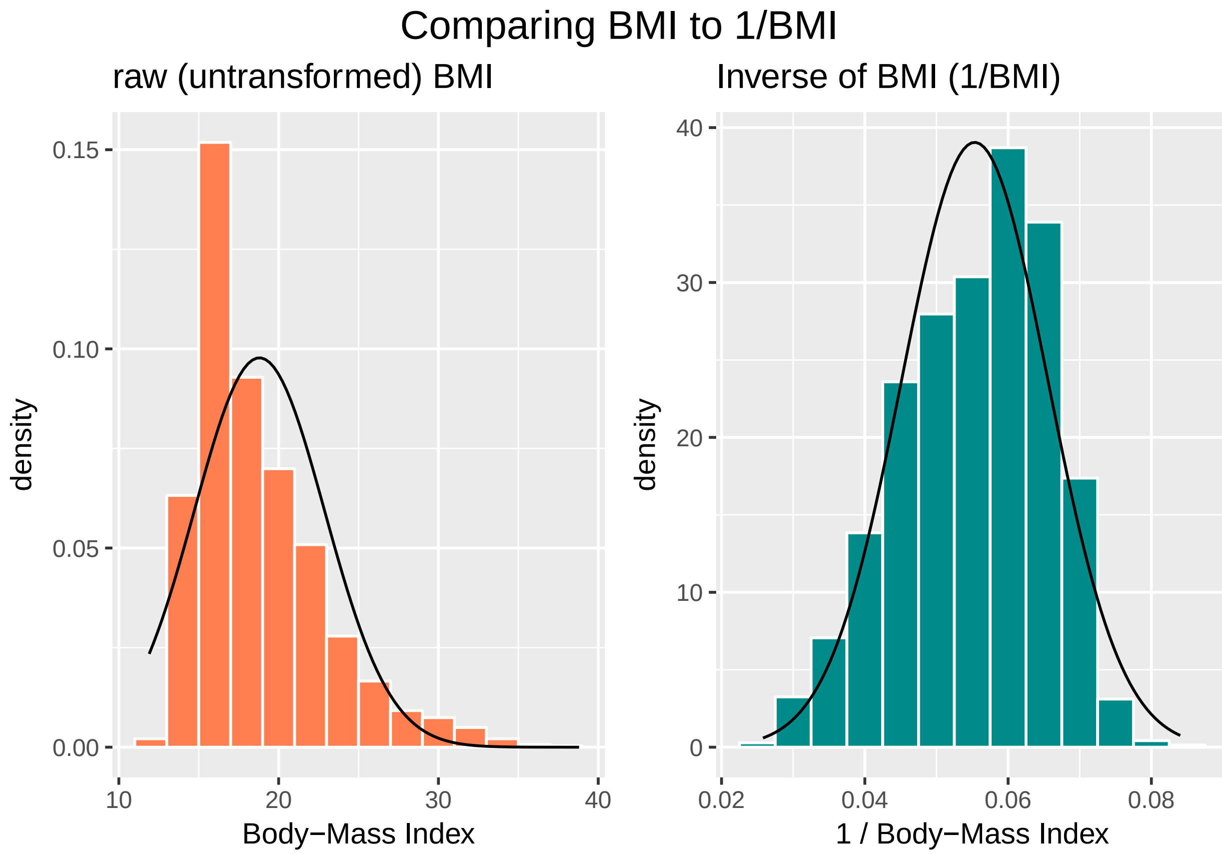



```
par(mfrow=c(1,1))  ## return to usual plot window
```

- The left plot shows fairly substantial **right** or *positive* skew
- The right plot shows there's much less skew after the inverse has been taken.
- Our conclusion is that a Normal model is a far better fit to 1/BMI than it is to BMI.

The effect of taking the inverse here may be clearer from the histograms below, with Normal density functions superimposed.

```
p1 <- ggplot(nyfs1, aes(x = bmi)) +
    geom_histogram(aes(y = ..density..),
                   binwidth=2, fill = "coral", color = "white") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(nyfs1$bmi), sd = sd(nyfs1$bmi))) +
    labs(x = "Body-Mass Index", title = "raw (untransformed) BMI")

p2 <- ggplot(nyfs1, aes(x = 1/bmi)) +
    geom_histogram(aes(y = ..density..),
                   binwidth=0.005, fill = "darkcyan", color = "white") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(1/nyfs1$bmi),
                              sd = sd(1/nyfs1$bmi))) +
labs(x = "1 / Body-Mass Index",
     title = "Inverse of BMI (1/BMI)")

gridExtra::grid.arrange(p1, p2, ncol=2,
top = textGrob("Comparing BMI to 1/BMI", gp=gpar(fontsize=15)))
```




```r
# this approach to top label lets us adjust the size of type used
# in the main title
# note that you'll need to have called library(grid) or
# require(grid) for this to work properly
rm(p1, p2) # cleanup
```

When we are confronted with a variable that is not Normally distributed but that we wish was Normally distributed, it is sometimes useful to consider whether working with a **transformation** of the data will yield a more helpful result. The next Section provides some initial guidance about choosing between a class of power transformations that can reduce the impact of non-Normality in unimodal data.

Chapter 9

Using Transformations to "Normalize" Distributions

- When we are confronted with a variable that is not Normally distributed but that we wish was Normally distributed, it is sometimes useful to consider whether working with a transformation of the data will yield a more helpful result.
- Many statistical methods, including t tests and analyses of variance, assume Normal distributions.
- We'll discuss using R to assess a range of what are called Box-Cox power transformations, via plots, mainly.

9.1 The Ladder of Power Transformations

The key notion in re-expression of a single variable to obtain a distribution better approximated by the Normal or re-expression of an outcome in a simple regression model is that of a **ladder of power transformations**, which applies to any unimodal data.

Power	Transformation
3	x^3
2	x^2
1	x (unchanged)
0.5	$x^{0.5} = \sqrt{x}$
0	ln x
-0.5	$x^{-0.5} = 1/\sqrt{x}$
-1	$x^{-1} = 1/x$
-2	$x^{-2} = 1/x^2$

9.2 Using the Ladder

As we move further away from the *identity* function (power = 1) we change the shape more and more in the same general direction.

- For instance, if we try a logarithm, and this seems like too much of a change, we might try a square root instead.
- Note that this ladder (which like many other things is due to John Tukey) uses the logarithm for the

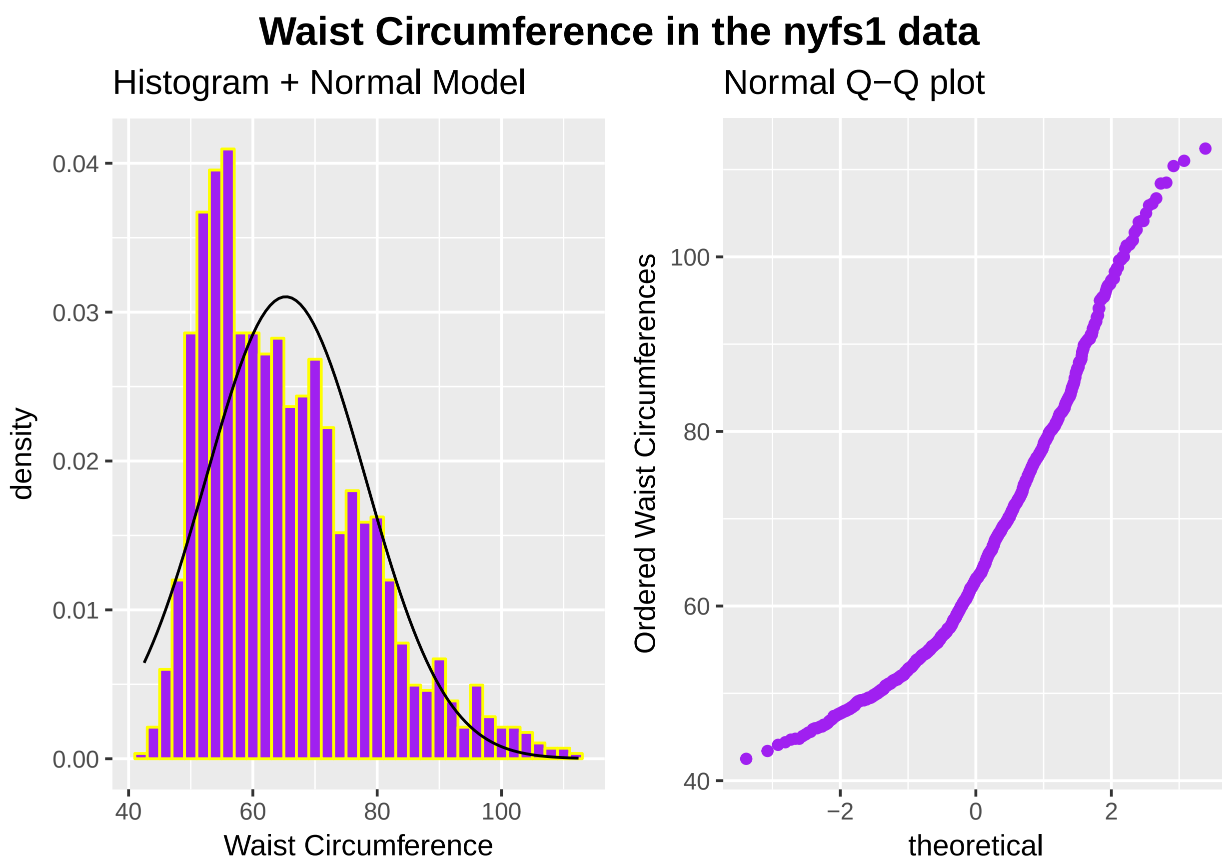
“power zero” transformation rather than the constant, which is what x^0 actually is.

- If the variable x can take on negative values, we might take a different approach. If x is a count of something that could be zero, we often simply add 1 to x before transformation.

9.3 Can we transform Waist Circumferences?

Here are a pair of plots describing the waist circumference data in the NYFS data.

```
p1 <- ggplot(nyfs1, aes(x = waist.circ)) +
    geom_histogram(aes(y = ..density..),
                   binwidth=2, fill = "purple", color = "yellow") +
    stat_function(fun = dnorm, args = list(mean = mean(nyfs1$waist.circ),
                                           sd = sd(nyfs1$waist.circ))) +
    labs(x = "Waist Circumference", title="Histogram + Normal Model")

p2 <- ggplot(nyfs1, aes(sample = waist.circ)) +
    geom_point(stat="qq", color = "purple") +
    labs(y = "Ordered Waist Circumferences", title="Normal Q-Q plot")

library(grid)
# this approach to top label lets us adjust
# the size and font (here bold) used in the main title
gridExtra::grid.arrange(p1, p2, ncol=2,
             top = textGrob("Waist Circumference in the nyfs1 data",
                            gp=gpar(fontsize=15,font=2)))
```



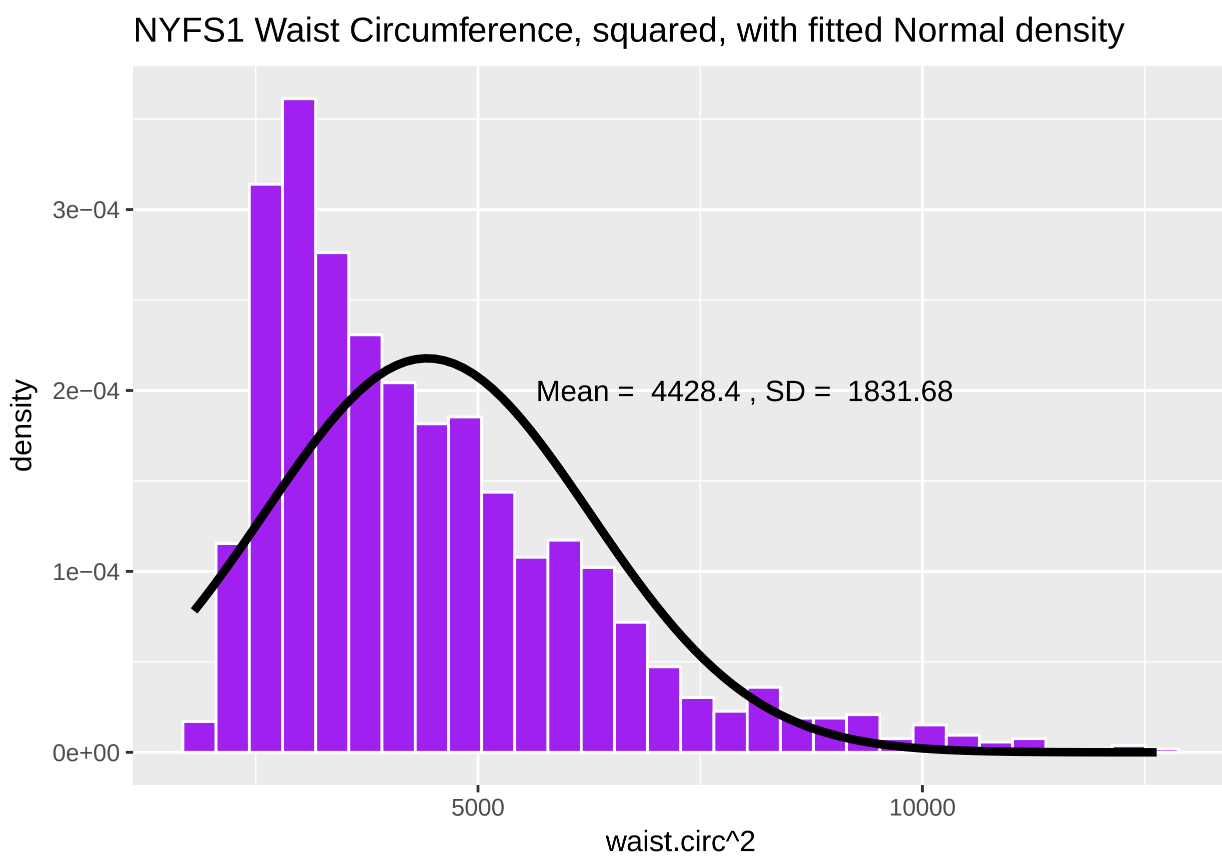
```r
## clean up
rm(p1, p2)
```

All of the values are positive, naturally, and there is some sign of skew. If we want to use the tools of the Normal distribution to describe these data, we might try taking a step "up" our ladder from power 1 to power 2.

9.3.1 The Square

Does squaring the Waist Circumference data help to "Normalize" the histogram?

```r
ggplot(nyfs1, aes(x = waist.circ^2)) +
    geom_histogram(aes(y = ..density..), bins = 30, fill = "purple", col="white") +
    stat_function(fun = dnorm, lwd = 1.5, col = "black",
                  args = list(mean = mean(nyfs1$waist.circ^2), sd = sd(nyfs1$waist.circ^2))) +
    annotate("text", x = 8000, y = 0.0002, col = "black",
             label = paste("Mean = ", round(mean(nyfs1$waist.circ^2),2),
                           ", SD = ", round(sd(nyfs1$waist.circ^2),2))) +
    labs(title = "NYFS1 Waist Circumference, squared, with fitted Normal density")
```

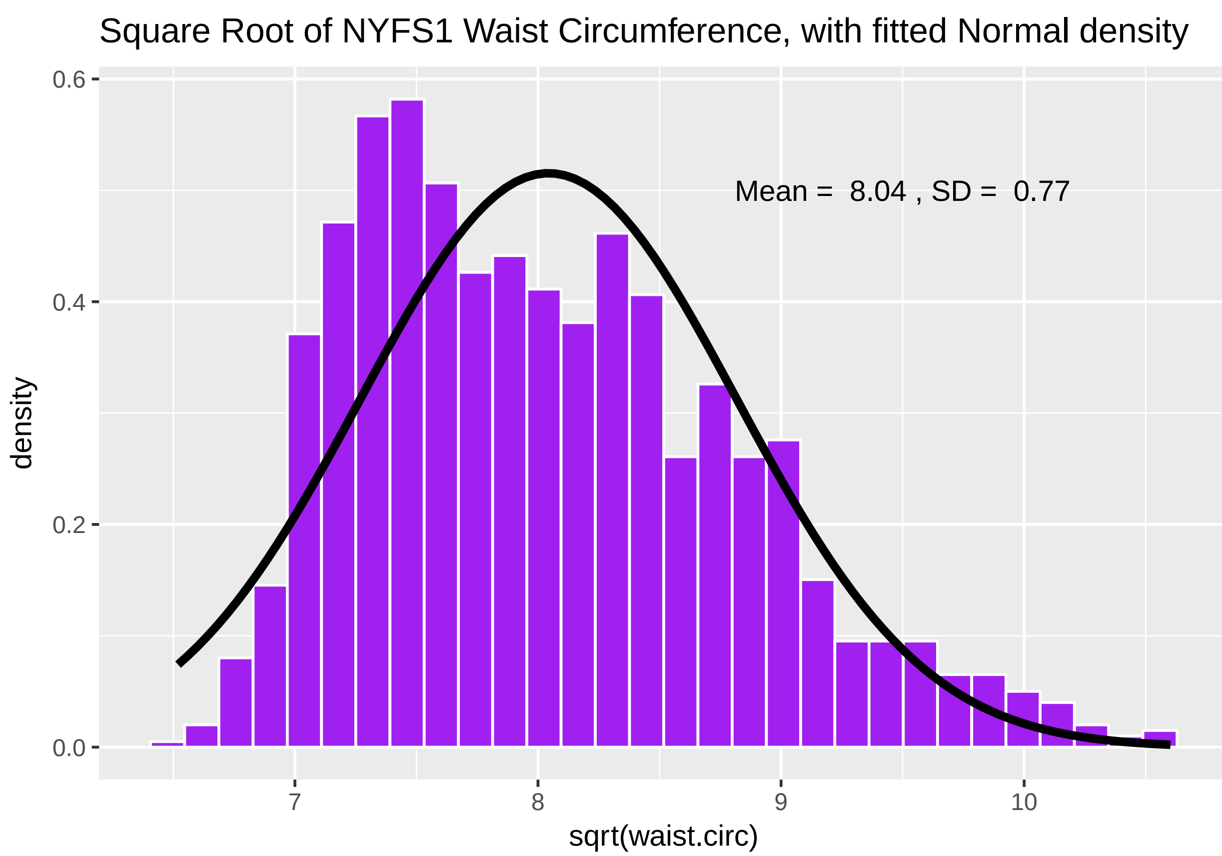


Looks like that was the wrong direction. Shall we try moving down the ladder instead?

9.3.2 The Square Root

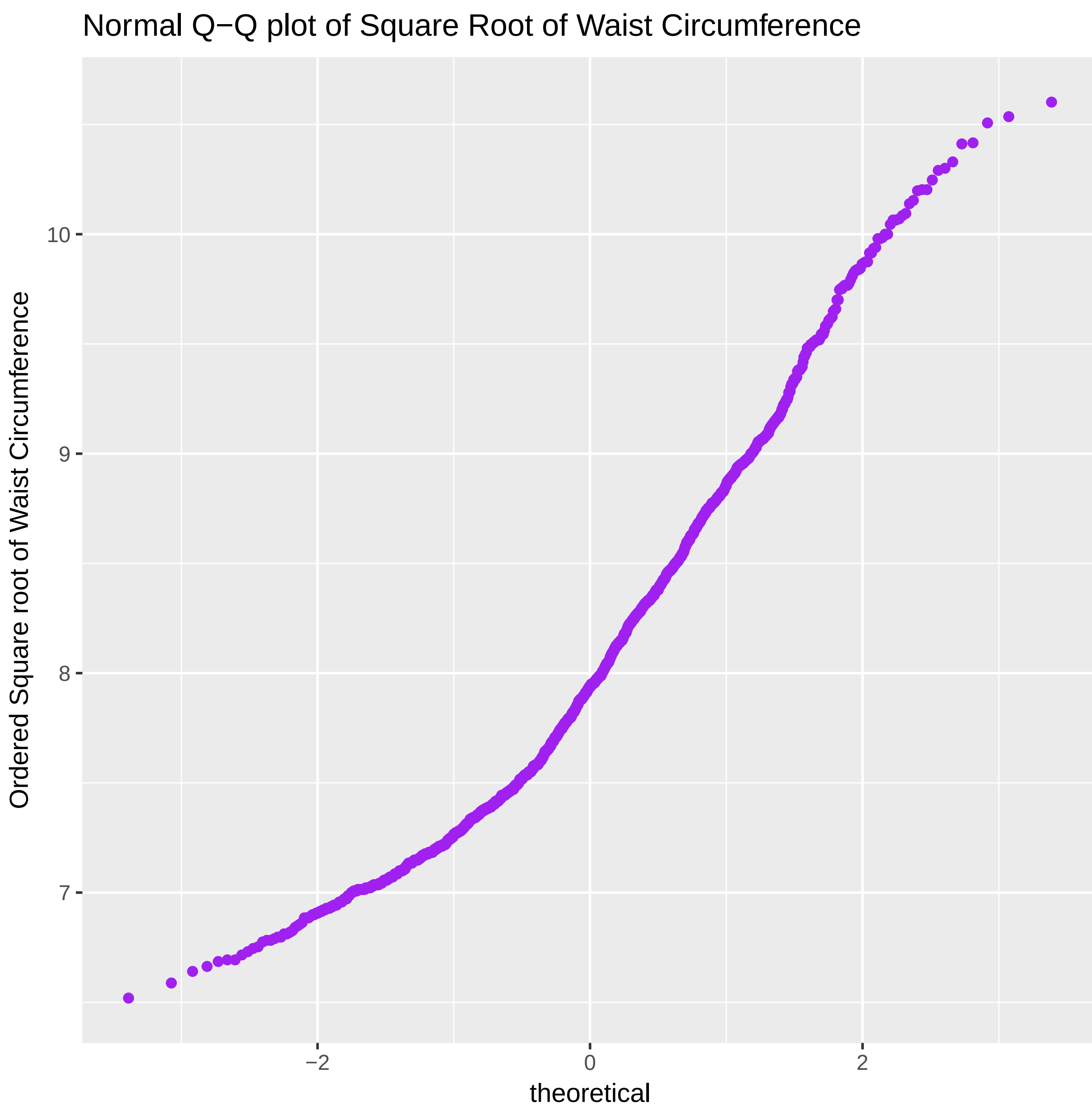
Would a square root applied to the waist circumference data help alleviate that right skew?

```
ggplot(nyfs1, aes(x = sqrt(waist.circ))) +
    geom_histogram(aes(y = ..density..), bins = 30,
                   fill = "purple", col="white") +
    stat_function(fun = dnorm, lwd = 1.5, col = "black",
                  args = list(mean = mean(sqrt(nyfs1$waist.circ)),
                              sd = sd(sqrt(nyfs1$waist.circ)))) +
    annotate("text", x = 9.5, y = 0.5, col = "black",
             label = paste("Mean = ", round(mean(sqrt(nyfs1$waist.circ)),2),
                           ", SD = ", round(sd(sqrt(nyfs1$waist.circ)),2))) +
    labs(title = "Square Root of NYFS1 Waist Circumference, with fitted Normal density")
```



That looks a lot closer to a Normal distribution. Consider the Normal Q-Q plot below.

```
ggplot(nyfs1, aes(sample = sqrt(waist.circ))) +
    geom_point(stat="qq", color = "purple") +
    labs(y = "Ordered Square root of Waist Circumference",
         title="Normal Q-Q plot of Square Root of Waist Circumference")
```

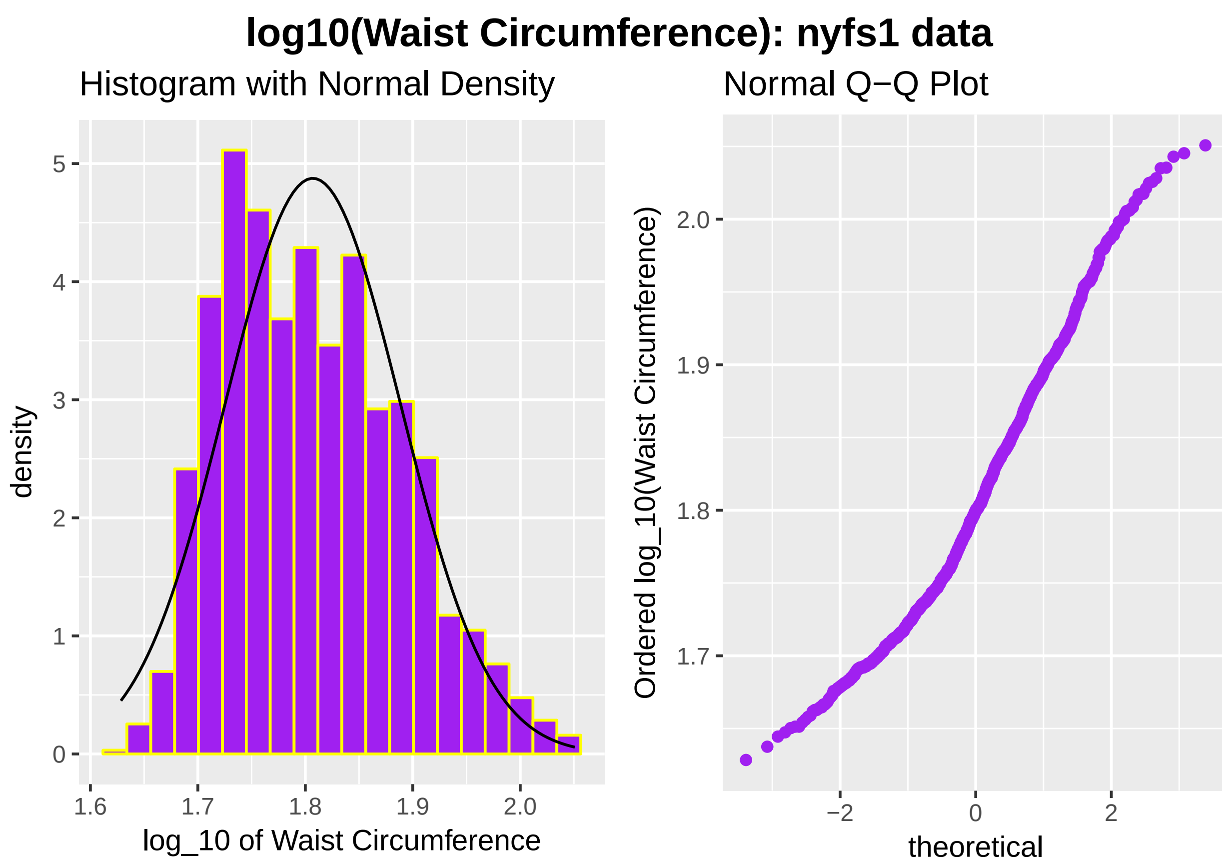


9.3.3 The Logarithm

We might also try a logarithm of the waist circumference data. We can use either the natural logarithm (log, in R) or the base-10 logarithm (log10, in R) - either will have the same impact on skew.

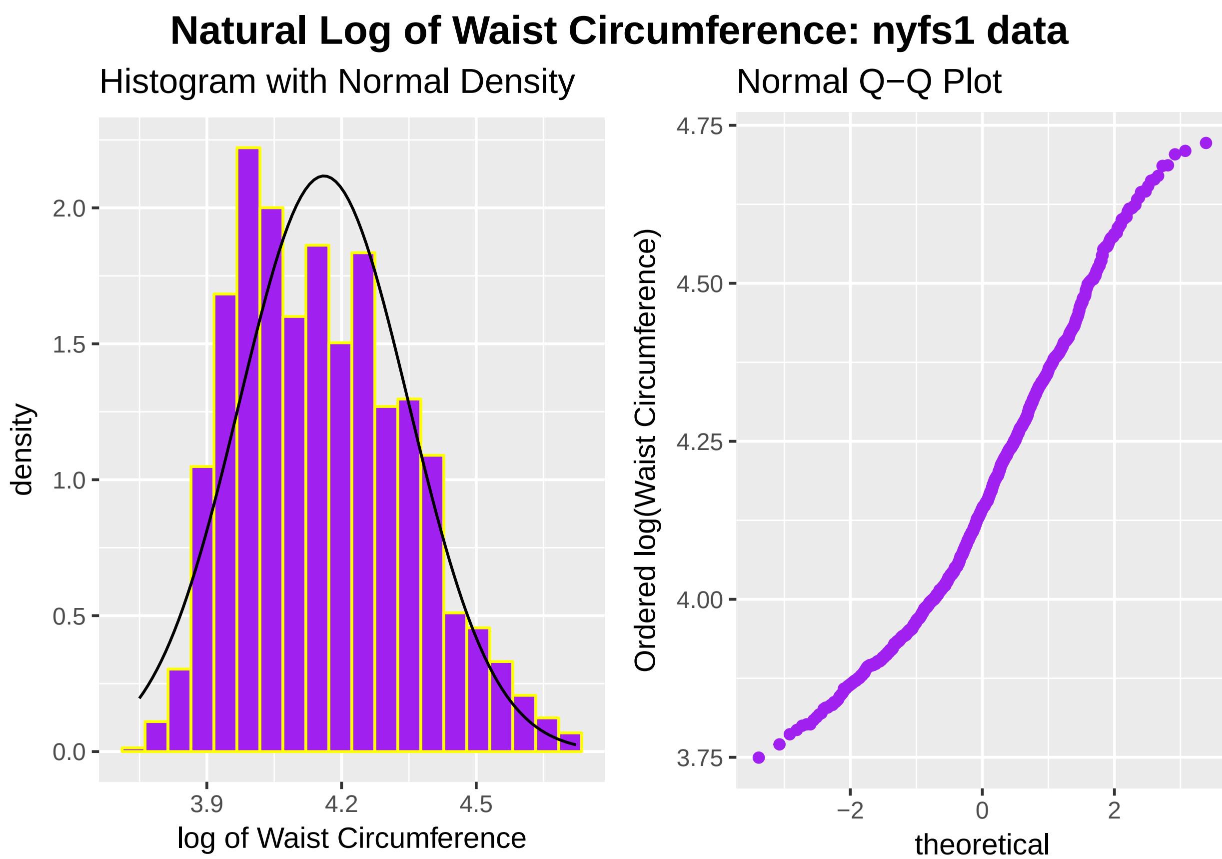
```
p1 <- ggplot(nyfs1, aes(x = log10(waist.circ))) +
    geom_histogram(aes(y = ..density..),
                   bins=20, fill = "purple", color = "yellow") +
    stat_function(fun = dnorm, args = list(mean = mean(log10(nyfs1$waist.circ)),
                                          sd = sd(log10(nyfs1$waist.circ)))) +
    labs(x = "log_10 of Waist Circumference", title="Histogram with Normal Density")

p2 <- ggplot(nyfs1, aes(sample = log10(waist.circ))) +
```

```
    geom_point(stat="qq", color = "purple") +
    labs(y = "Ordered log_10(Waist Circumference)", title="Normal Q-Q Plot")

gridExtra::grid.arrange(p1, p2, ncol=2,
             top = textGrob("log10(Waist Circumference): nyfs1 data",
                            gp=gpar(fontsize=15,font=2)))
```




```
## clean up
rm(p1, p2)
```

```
p1 <- ggplot(nyfs1, aes(x = log(waist.circ))) +
    geom_histogram(aes(y = ..density..),
                   bins=20, fill = "purple", color = "yellow") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(log(nyfs1$waist.circ)),
                              sd = sd(log(nyfs1$waist.circ)))) +
    labs(x = "log of Waist Circumference", title="Histogram with Normal Density")

p2 <- ggplot(nyfs1, aes(sample = log(waist.circ))) +
    geom_point(stat="qq", color = "purple") +
    labs(y = "Ordered log(Waist Circumference)", title="Normal Q-Q Plot")

gridExtra::grid.arrange(p1, p2, ncol=2,
             top = textGrob("Natural Log of Waist Circumference: nyfs1 data",
                            gp=gpar(fontsize=15,font=2)))
```



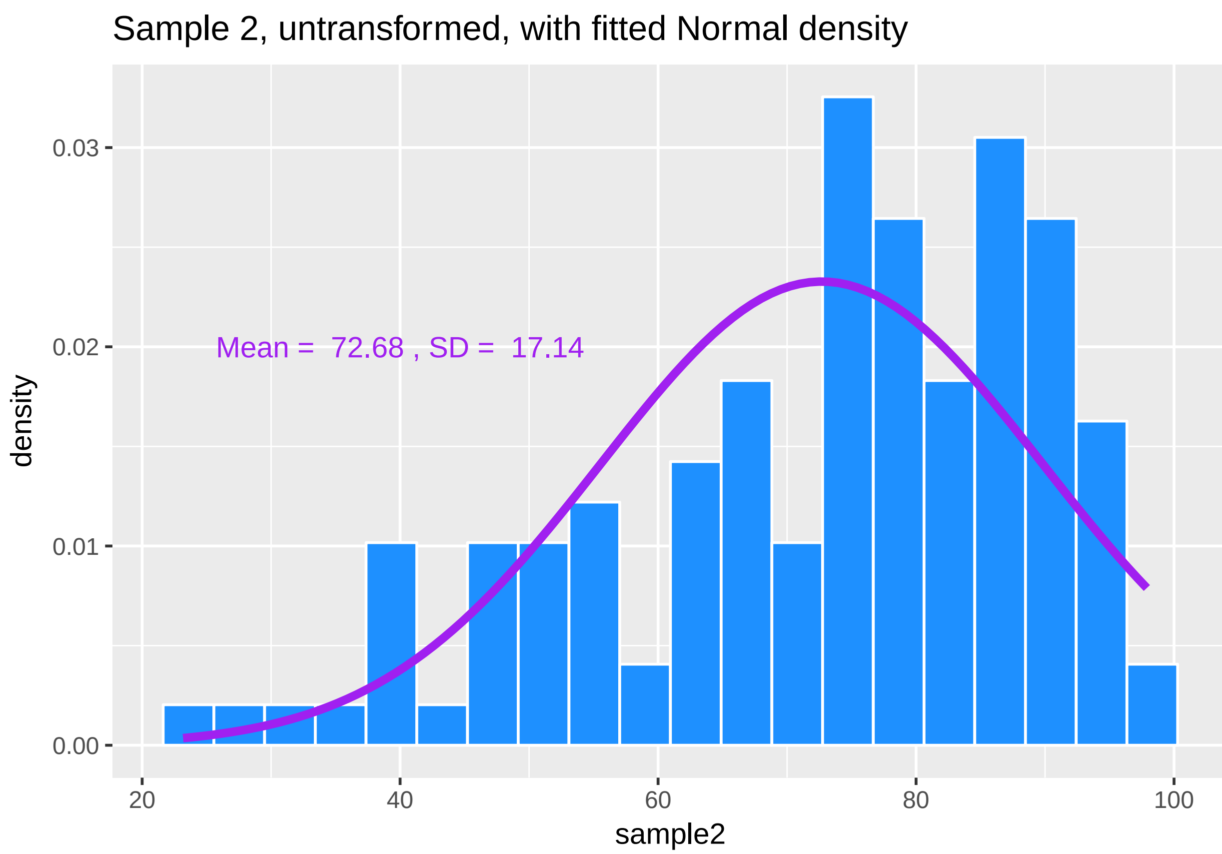

```
## clean up
rm(p1, p2)
```

9.4 A Simulated Data Set with Left Skew

```
set.seed(431); data2 <- data.frame(sample2 = 100*rbeta(n = 125, shape1 = 5, shape2 = 2))
```

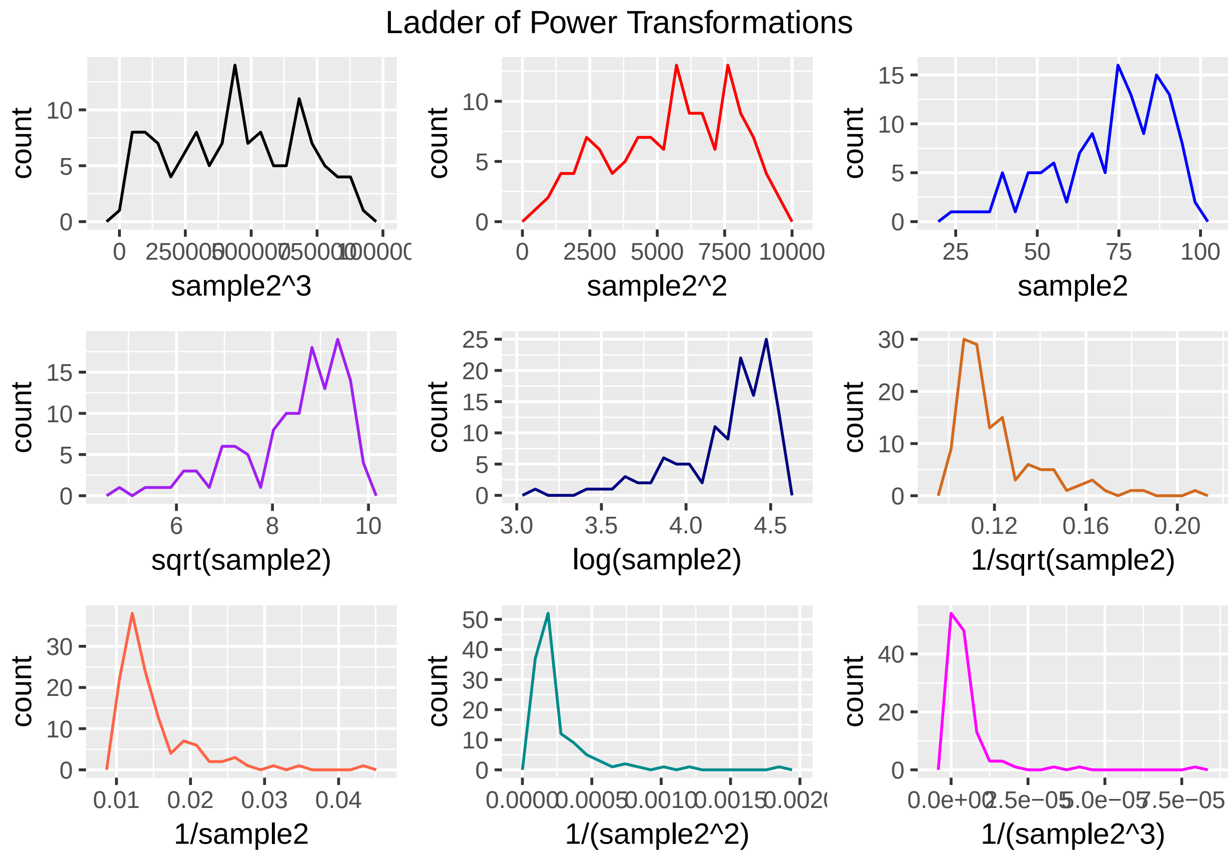
If we'd like to transform these data so as to better approximate a Normal distribution, where should we start? What transformation do you suggest?

```
ggplot(data2, aes(x = sample2)) +
    geom_histogram(aes(y = ..density..),
                   bins = 20, fill = "dodgerblue", col="white") +
    stat_function(fun = dnorm, lwd = 1.5, col = "purple",
                  args = list(mean = mean(data2$sample2),
                              sd = sd(data2$sample2))) +
    annotate("text", x = 40, y = 0.02, col = "purple",
             label = paste("Mean = ", round(mean(data2$sample2),2),
                           ", SD = ", round(sd(data2$sample2),2))) +
    labs(title = "Sample 2, untransformed, with fitted Normal density")
```



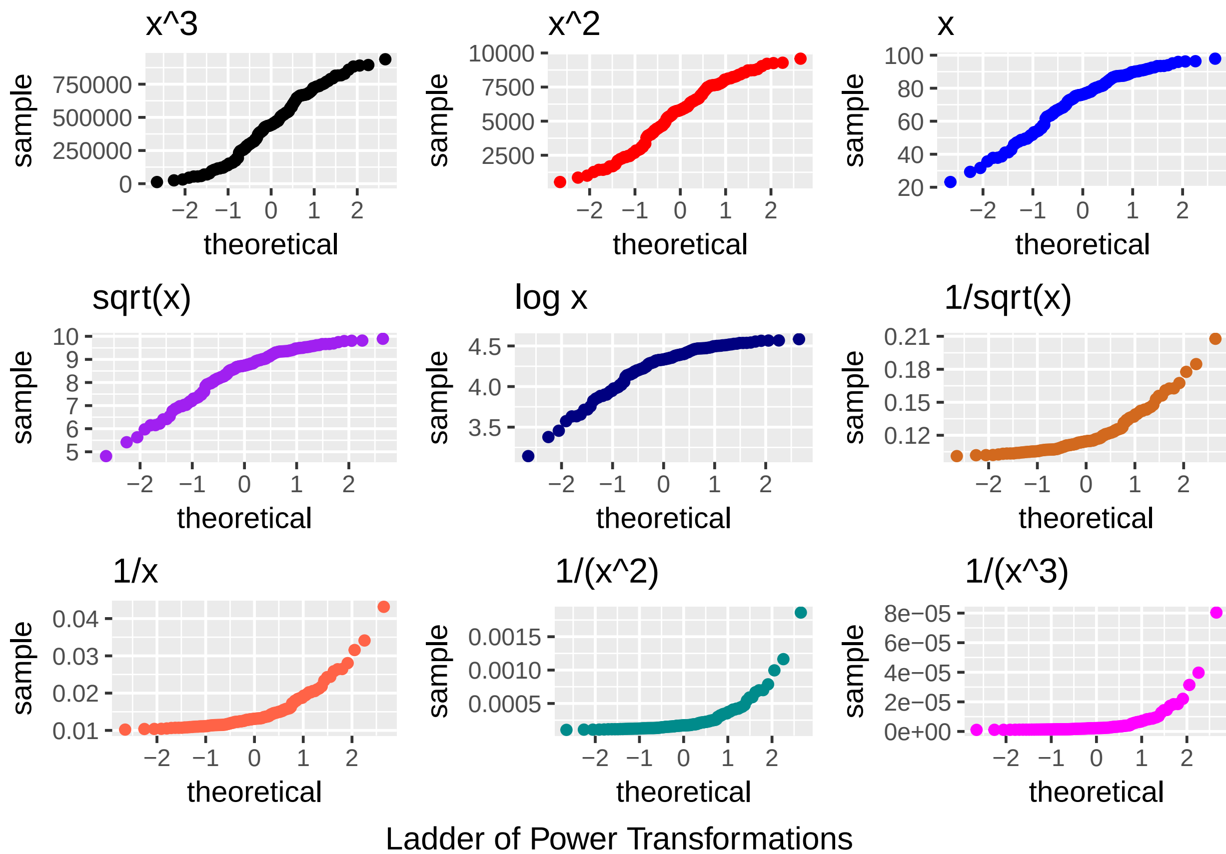
9.5 Transformation Example 2: Ladder of Potential Transformations in Frequency Polygons

```
p1 <- ggplot(data2, aes(x = sample2^3)) + geom_freqpoly(col = "black", bins = 20)
p2 <- ggplot(data2, aes(x = sample2^2)) + geom_freqpoly(col = "red", bins = 20)
p3 <- ggplot(data2, aes(x = sample2)) + geom_freqpoly(col = "blue", bins = 20)
p4 <- ggplot(data2, aes(x = sqrt(sample2))) + geom_freqpoly(col = "purple", bins = 20)
p5 <- ggplot(data2, aes(x = log(sample2))) + geom_freqpoly(col = "navy", bins = 20)
p6 <- ggplot(data2, aes(x = 1/sqrt(sample2))) + geom_freqpoly(col = "chocolate", bins = 20)
p7 <- ggplot(data2, aes(x = 1/sample2)) + geom_freqpoly(col = "tomato", bins = 20)
p8 <- ggplot(data2, aes(x = 1/(sample2^2))) + geom_freqpoly(col = "darkcyan", bins = 20)
p9 <- ggplot(data2, aes(x = 1/(sample2^3))) + geom_freqpoly(col = "magenta", bins = 20)

gridExtra::grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, p9, nrow=3,
top="Ladder of Power Transformations")
```



9.6 Transformation Example 2 Ladder with Normal Q-Q Plots

```
p1 <- ggplot(data2, aes(sample = sample2^3)) +
geom_point(stat="qq", col = "black") + labs(title = "x^3")
p2 <- ggplot(data2, aes(sample = sample2^2)) +
geom_point(stat="qq", col = "red") + labs(title = "x^2")
p3 <- ggplot(data2, aes(sample = sample2)) +
geom_point(stat="qq", col = "blue") + labs(title = "x")
p4 <- ggplot(data2, aes(sample = sqrt(sample2))) +
geom_point(stat="qq", col = "purple") + labs(title = "sqrt(x)")
p5 <- ggplot(data2, aes(sample = log(sample2))) +
geom_point(stat="qq", col = "navy") + labs(title = "log x")
p6 <- ggplot(data2, aes(sample = 1/sqrt(sample2))) +
geom_point(stat="qq", col = "chocolate") + labs(title = "1/sqrt(x)")
p7 <- ggplot(data2, aes(sample = 1/sample2)) +
geom_point(stat="qq", col = "tomato") + labs(title = "1/x")
p8 <- ggplot(data2, aes(sample = 1/(sample2^2))) +
geom_point(stat="qq", col = "darkcyan") + labs(title = "1/(x^2)")
p9 <- ggplot(data2, aes(sample = 1/(sample2^3))) +
geom_point(stat="qq", col = "magenta") + labs(title = "1/(x^3)")

gridExtra::grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, p9, nrow=3,
bottom="Ladder of Power Transformations")
```



It looks like taking the square of the data produces the most “Normalish” plot in this case.

Chapter 10

Summarizing data within subgroups

10.1 Using dplyr and summarise to build a tibble of summary information

```
nyfs1 %>%
    group_by(sex) %>%
    select(bmi, waist.circ, sex) %>%
    summarise_all(funs(median))
```

```
# A tibble: 2 x 3
     sex   bmi waist.circ
  <fctr> <dbl>      <dbl>
1 Female  17.6       63.6
2   Male  17.7       62.5
```

```
nyfs1 %>%
    group_by(bmi.cat) %>%
    summarise(mean = mean(waist.circ), sd = sd(waist.circ), median = median(waist.circ),
              skew_1 = round((mean(waist.circ) - median(waist.circ)) / sd(waist.circ),3))
```

```
# A tibble: 4 x 5
          bmi.cat  mean    sd median skew_1
           <fctr> <dbl> <dbl>  <dbl>  <dbl>
1   1 Underweight  54.9  7.63   53.9  0.136
2 2 Normal weight  61.0  9.10   59.2  0.193
3    3 Overweight  71.1 11.80   72.0 -0.075
4         4 Obese  79.9 15.01   79.9 -0.003
```

While patients in the heavier groups generally had higher waist circumferences, this is not inevitably the case.

The data transformation with dplyr cheat sheet found under the Help menu in R Studio is a great resource. And, of course, for more details, visit Grolemund and Wickham (2017).

10.2 Using the by function to summarize groups numerically

We can summarize our data numerically in multiple ways, but to use the `favstats` or `Hmisc::describe` tools to each individual BMI subgroup separately, we might consider applying the `by` function.

```
by(nyfs1$waist.circ, nyfs1$bmi.cat, mosaic::favstats)
```

```
nyfs1$bmi.cat: 1 Underweight
  min   Q1 median   Q3  max mean   sd  n missing
 42.5 49.2   53.9 62.4 68.5 54.9 7.63 42       0
--------------------------------------------------------
nyfs1$bmi.cat: 2 Normal weight
  min   Q1 median Q3  max mean  sd   n missing
 44.1 53.8   59.2 68 85.5   61 9.1 926       0
--------------------------------------------------------
nyfs1$bmi.cat: 3 Overweight
  min   Q1 median   Q3  max mean   sd   n missing
 49.3 60.8     72 80.6 98.3 71.1 11.8 237       0
--------------------------------------------------------
nyfs1$bmi.cat: 4 Obese
  min   Q1 median   Q3 max mean sd   n missing
 52.1 66.7   79.9 91.6 112 79.9 15 211       0
```

As shown below, we could do this in pieces with `dplyr`, but the `by` approach can be faster for this sort of thing.

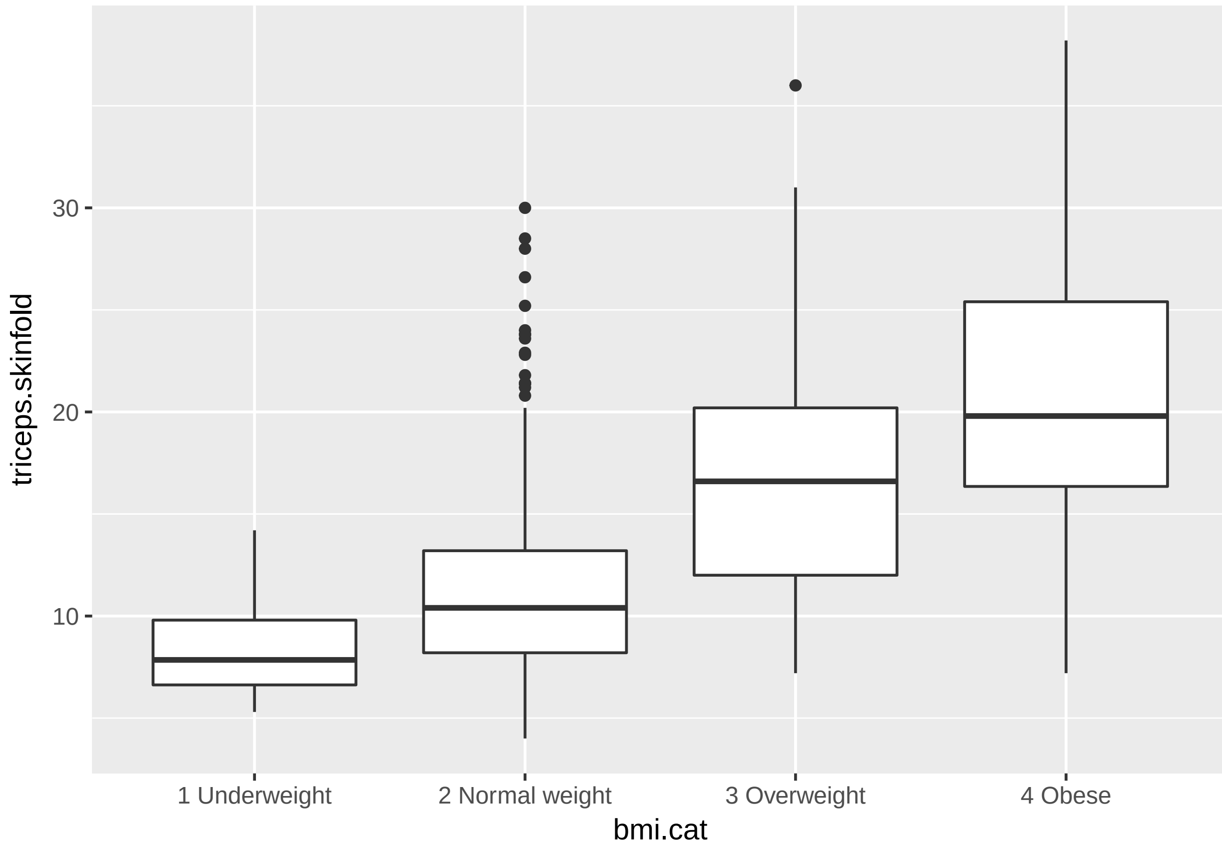
```
nyfs1 %>%
    group_by(bmi.cat) %>%
    summarise(min = min(waist.circ), Q1 = quantile(waist.circ, 0.25),
              median = median(waist.circ), Q3 = quantile(waist.circ, 0.75),
              max = max(waist.circ), mean = mean(waist.circ),
              sd = sd(waist.circ), n = length(waist.circ),
              missing = sum(is.na(waist.circ)))
```

```
# A tibble: 4 x 10
          bmi.cat   min    Q1 median    Q3   max  mean    sd     n missing
           <fctr> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <int>   <int>
1   1 Underweight  42.5  49.2   53.9  62.4  68.5  54.9  7.63    42       0
2 2 Normal weight  44.1  53.8   59.2  68.0  85.5  61.0  9.10   926       0
3    3 Overweight  49.3  60.8   72.0  80.6  98.3  71.1 11.80   237       0
4         4 Obese  52.1  66.7   79.9  91.6 112.4  79.9 15.01   211       0
```

10.3 Boxplots to Relate an Outcome to a Categorical Predictor

Boxplots are much more useful when comparing samples of data. For instance, consider this comparison boxplot describing the triceps skinfold results across the four levels of BMI category.

```
ggplot(nyfs1, aes(x=bmi.cat, y=triceps.skinfold)) +
    geom_boxplot()
```

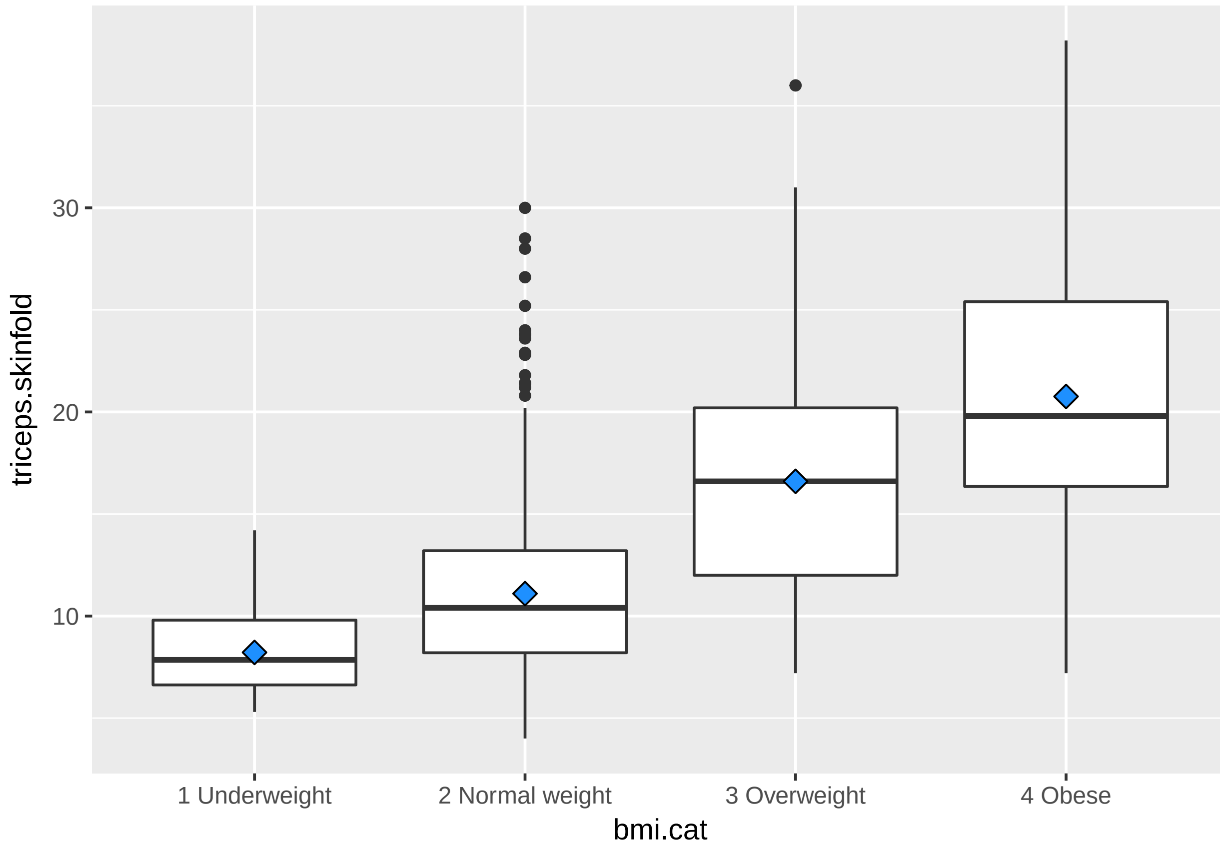


As always, the boxplot shows the five-number summary (minimum, 25th percentile, median, 75th percentile and maximum) in addition to highlighting candidate outliers.

10.3.1 Augmenting the Boxplot with the Sample Mean

Often, we want to augment such a plot, perhaps with the **sample mean** within each category, so as to highlight skew (in terms of whether the mean is meaningfully different from the median.)

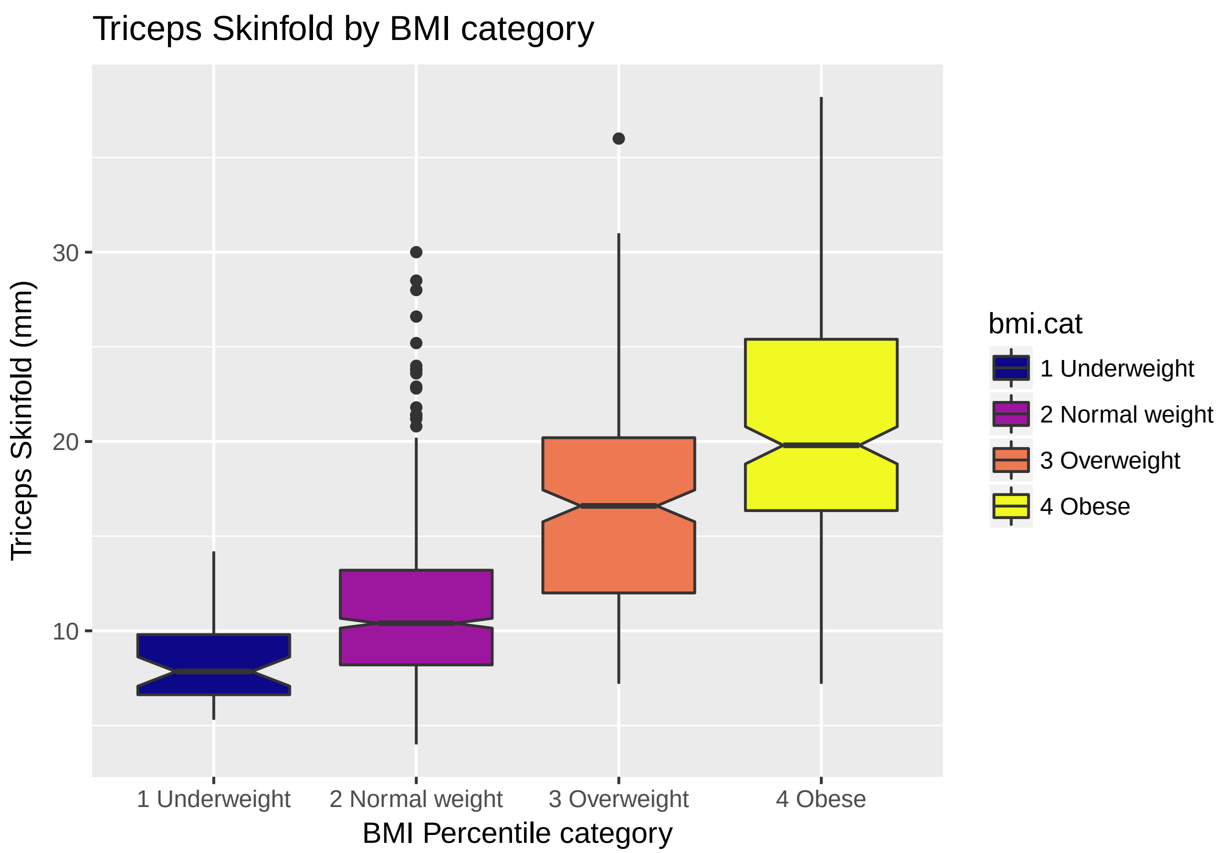
```
ggplot(nyfs1, aes(x=bmi.cat, y=triceps.skinfold)) +
    geom_boxplot() +
    stat_summary(fun.y="mean", geom="point", shape=23, size=3, fill="dodgerblue")
```



10.3.2 Adding Notches to a Boxplot

Notches are used in boxplots to help visually assess whether the medians of the distributions across the various groups actually differ to a statistically detectable extent Think of them as confidence regions around the medians. If the notches do not overlap, as in this situation, this provides some evidence that the medians in the populations represented by these samples may be different.

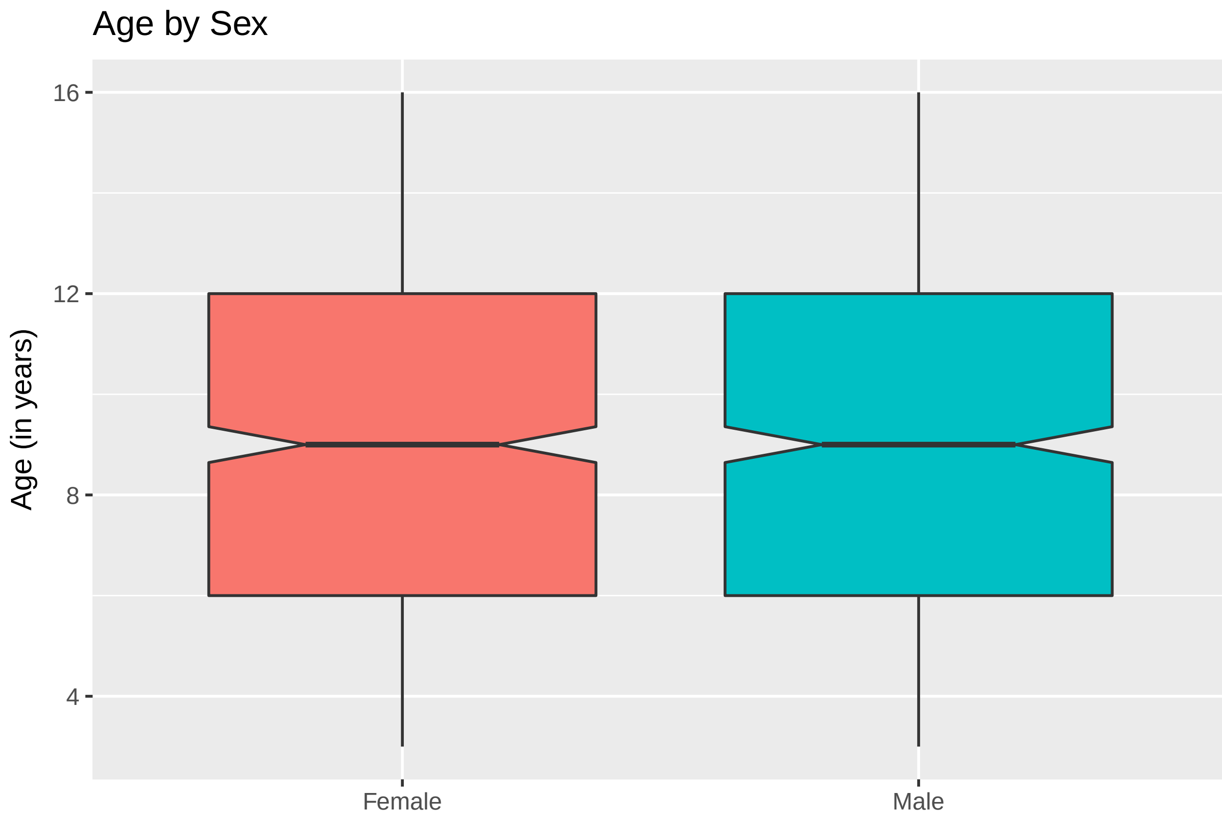
```
ggplot(nyfs1, aes(x=bmi.cat, y=triceps.skinfold, fill = bmi.cat)) +
    geom_boxplot(notch=TRUE) +
    scale_fill_viridis(discrete=TRUE, option="plasma") +
    labs(title = "Triceps Skinfold by BMI category",
         x = "BMI Percentile category", y = "Triceps Skinfold (mm)")
```



There is no overlap between the notches for each of the four categories, so we might reasonably conclude that the true median triceps skinfold values across the four categories are statistically significantly different.

For an example where the notches overlap, consider the comparison of ages across sex.

```
ggplot(nyfs1, aes(x=sex, y=age.exam, fill=sex)) +
    geom_boxplot(notch=TRUE) +
    guides(fill = "none") + ## drops the legend
    labs(title = "Age by Sex", x = "", y = "Age (in years)")
```

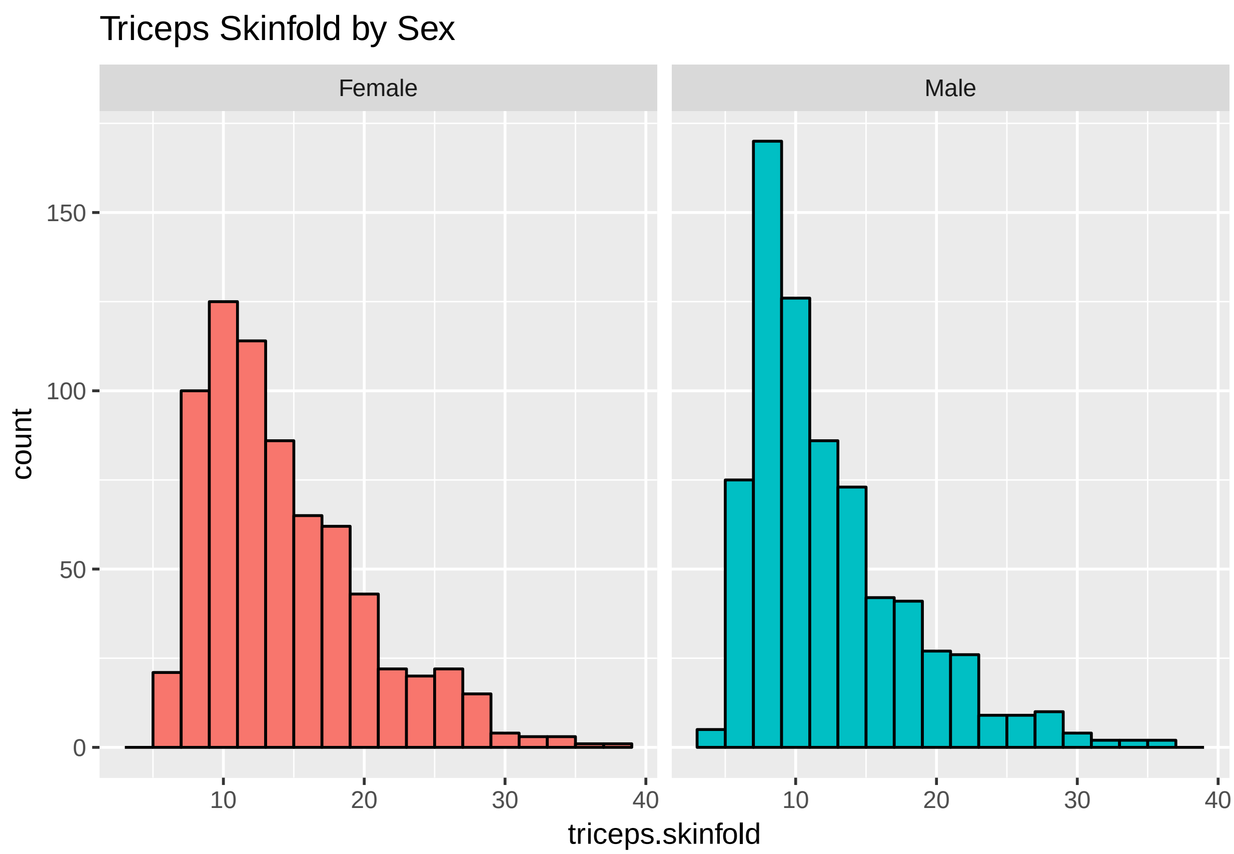


In this case, the overlap in the notches suggests that the median ages in the population of interest don't necessarily differ by sex.

10.4 Using Multiple Histograms to Make Comparisons

We can make an array of histograms to describe multiple groups of data, using `ggplot2` and the notion of **faceting** our plot.

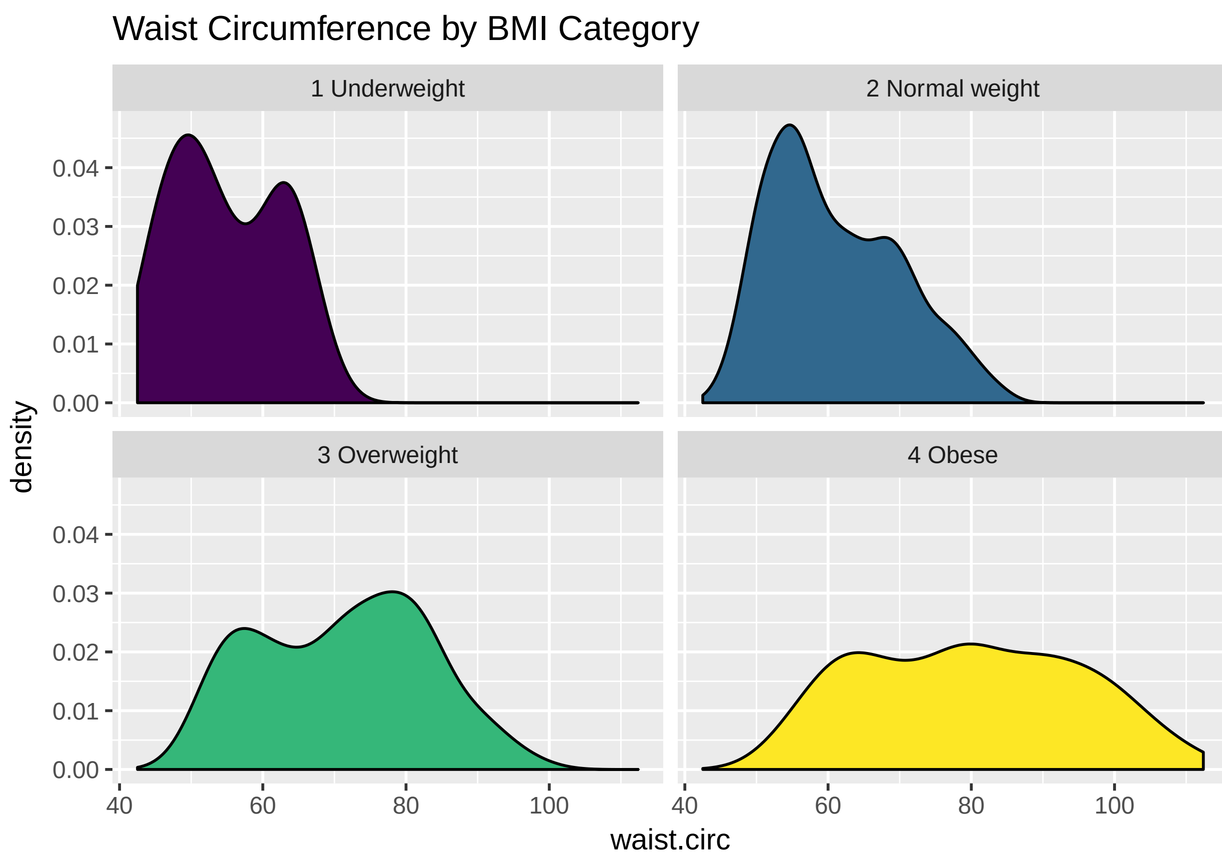
```
ggplot(nyfs1, aes(x=triceps.skinfold, fill = sex)) +
    geom_histogram(binwidth = 2, color = "black") +
    facet_wrap(~ sex) +
    guides(fill = "none") +
    labs(title = "Triceps Skinfold by Sex")
```



10.5 Using Multiple Density Plots to Make Comparisons

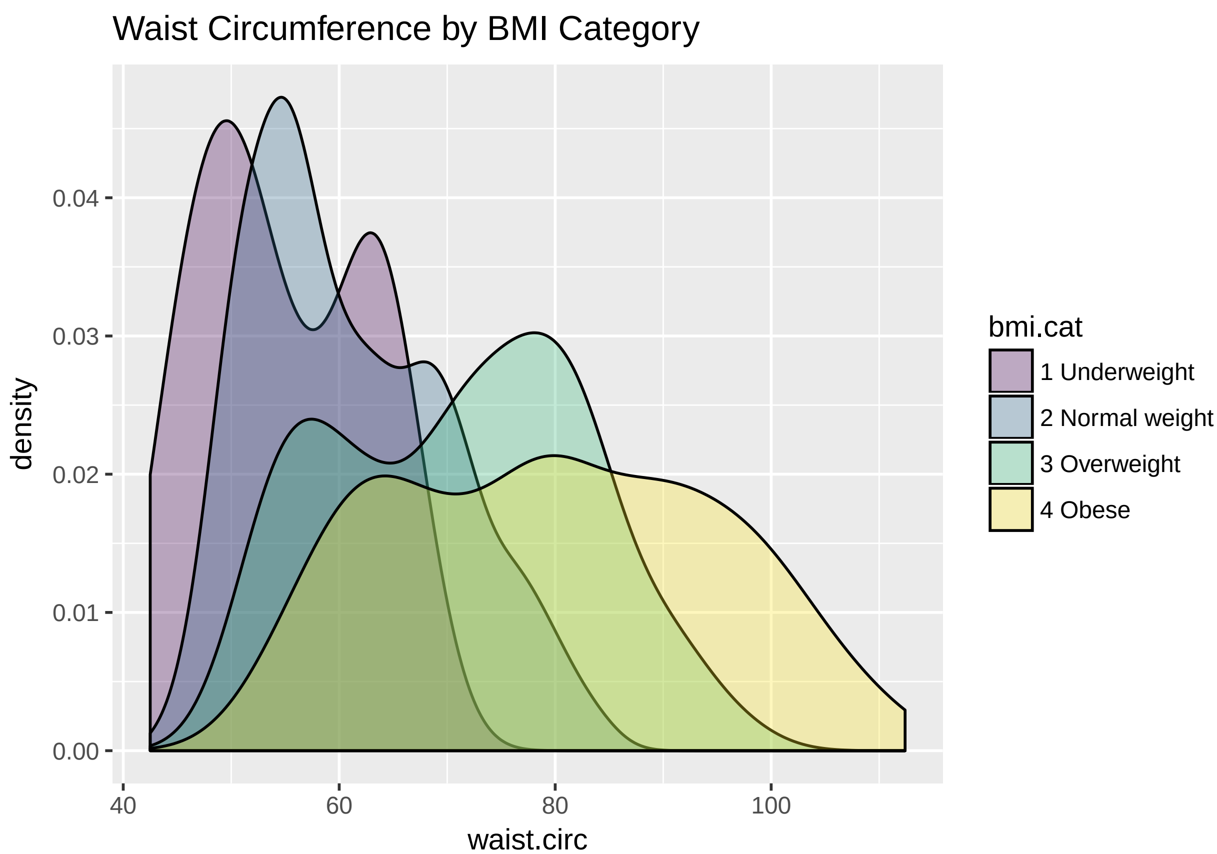
Or, we can make a series of density plots to describe multiple groups of data.

```
ggplot(nyfs1, aes(x=waist.circ, fill = bmi.cat)) +
    geom_density() +
    facet_wrap(~ bmi.cat) +
    scale_fill_viridis(discrete=T) +
    guides(fill = "none") +
    labs(title = "Waist Circumference by BMI Category")
```



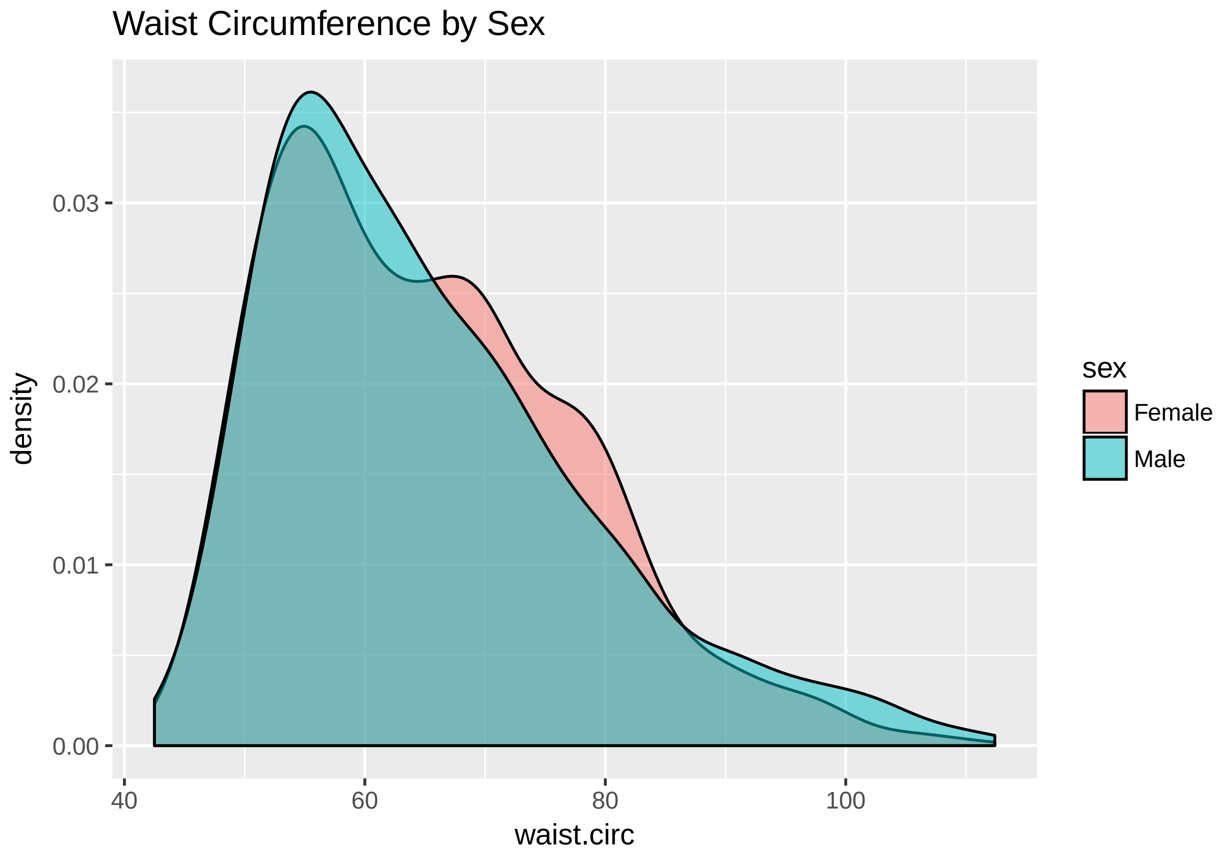
Or, we can plot all of the densities on top of each other with semi-transparent fills.

```
ggplot(nyfs1, aes(x=waist.circ, fill=bmi.cat)) +
    geom_density(alpha=0.3) +
    scale_fill_viridis(discrete=T) +
    labs(title = "Waist Circumference by BMI Category")
```



This really works better when we are comparing only two groups, like females to males.

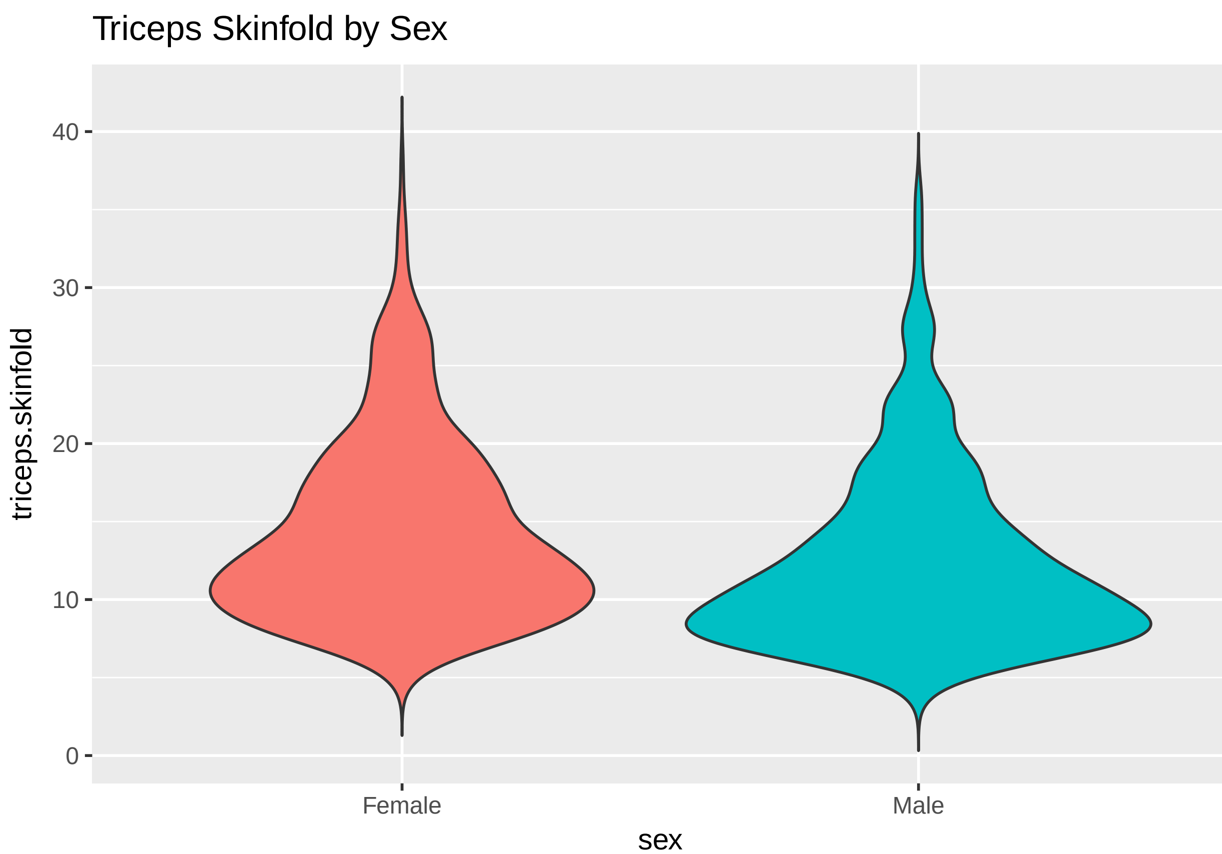
```
ggplot(nyfs1, aes(x=waist.circ, fill=sex)) +
    geom_density(alpha=0.5) +
    labs(title = "Waist Circumference by Sex")
```



10.6 Building a Violin Plot

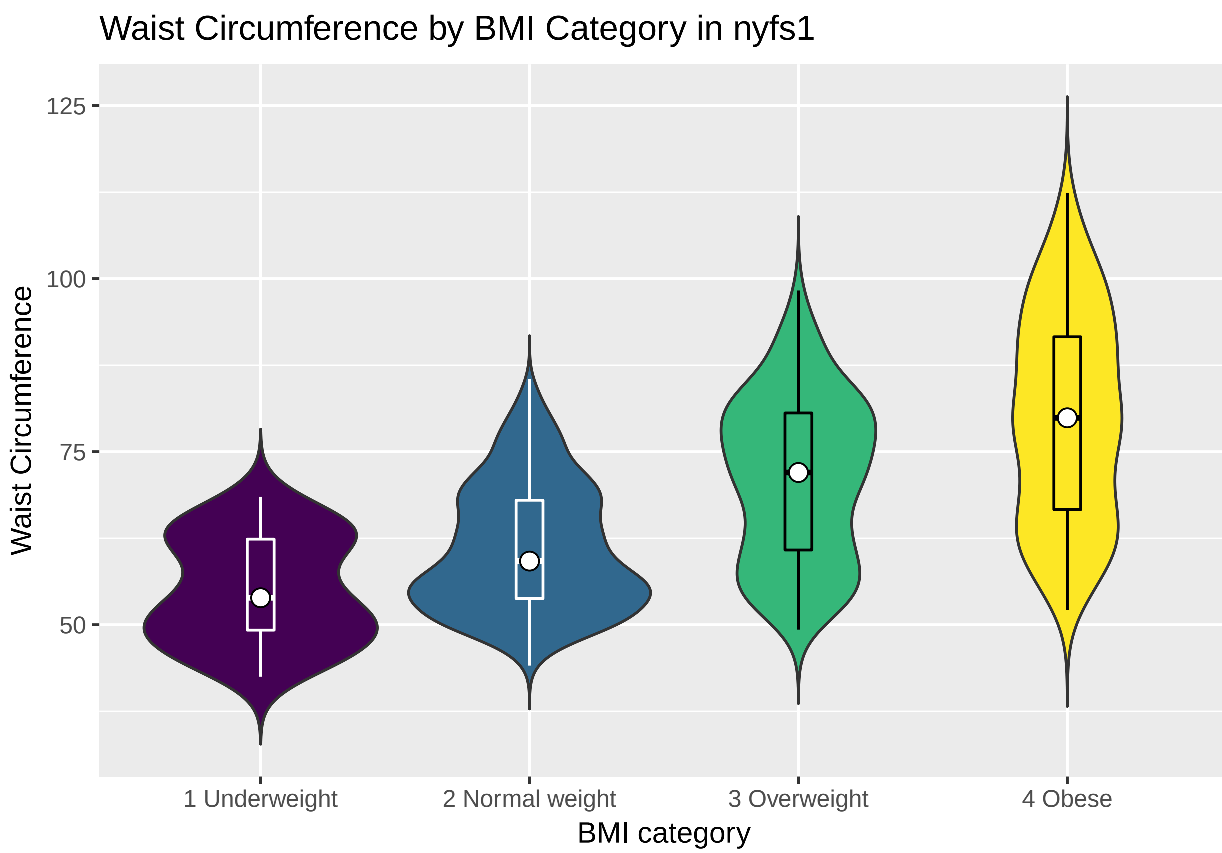
There are a number of other plots which compare distributions of data sets. An interesting one is called a **violin plot**. A violin plot is a kernel density estimate, mirrored to form a symmetrical shape.

```
ggplot(nyfs1, aes(x=sex, y=triceps.skinfold, fill = sex)) +
    geom_violin(trim=FALSE) +
    guides(fill = "none") +
    labs(title = "Triceps Skinfold by Sex")
```



Traditionally, these plots are shown with overlaid boxplots and a white dot at the median, like this.

```
ggplot(nyfs1, aes(x=bmi.cat, y=waist.circ, fill = bmi.cat)) +
    geom_violin(trim=FALSE) +
    geom_boxplot(width=.1, outlier.colour=NA,
                 color = c(rep("white",2), rep("black",2))) +
    stat_summary(fun.y=median, geom="point",
                 fill="white", shape=21, size=3) +
    scale_fill_viridis(discrete=T) +
    guides(fill = "none") +
    labs(title = "Waist Circumference by BMI Category in nyfs1",
         x = "BMI category", y = "Waist Circumference")
```



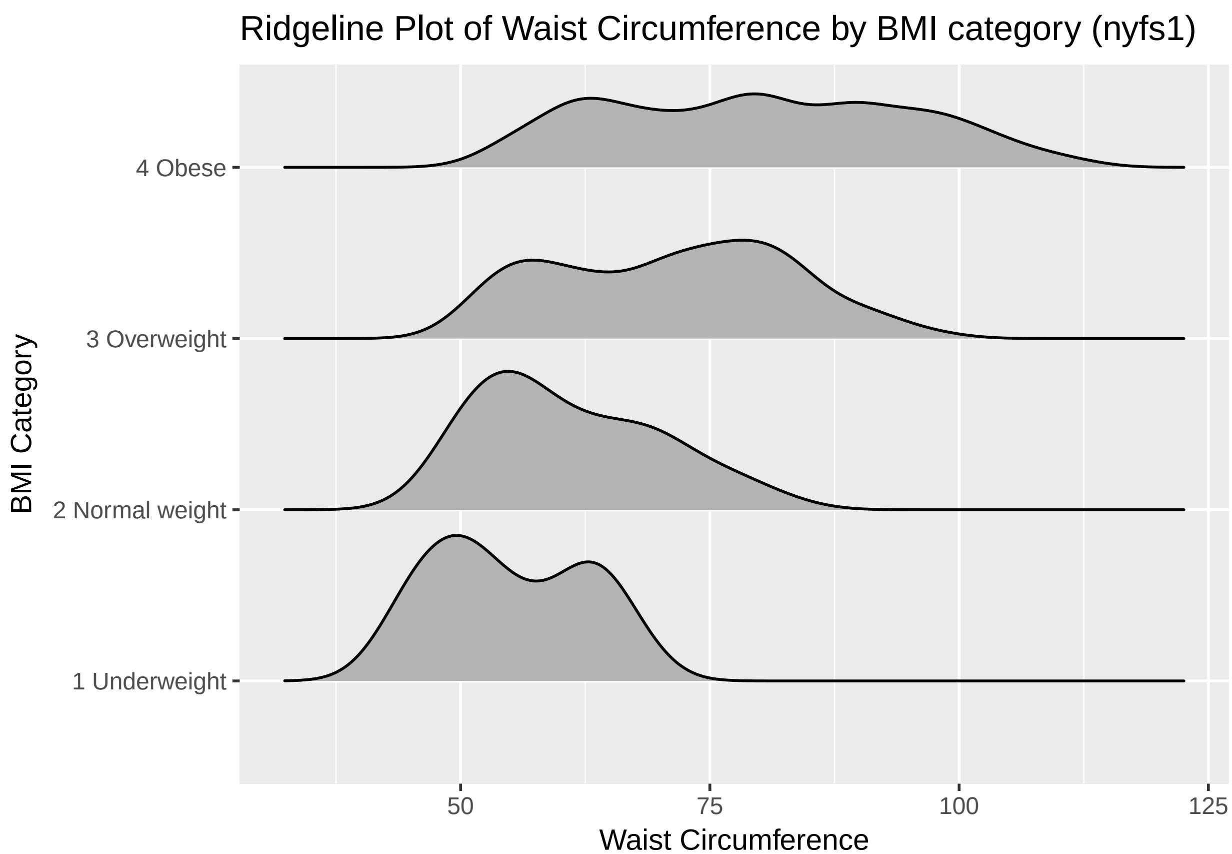
10.7 A Ridgeline Plot

Some people don't like violin plots - for example, see https://simplystatistics.org/2017/07/13/the-joy-of-no-more-violin-plots/. A very new and attractive alternative plot is available. This shows the distribution of several groups simultaneously, especially when you have lots of subgroup categories, and is called a **ridgeline plot**[1].

```
nyfs1 %>%
    ggplot(aes(x = waist.circ, y = bmi.cat, height = ..density..)) +
    ggridges::geom_density_ridges(scale = 0.85) +
    labs(title = "Ridgeline Plot of Waist Circumference by BMI category (nyfs1)",
         x = "Waist Circumference", y = "BMI Category")
```

```
Picking joint bandwidth of 3.38
```

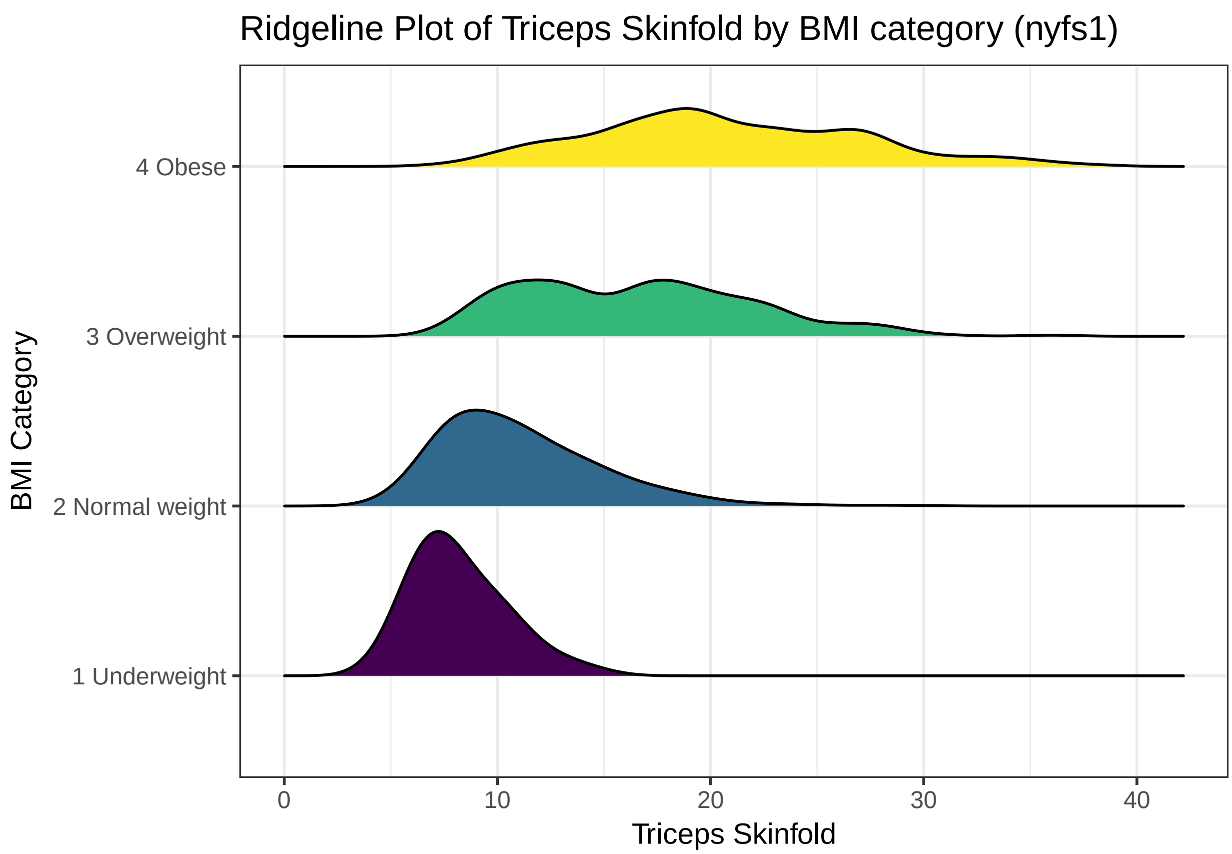
[1]These were originally called joy plots, and the tools were contained in the `ggjoy` package but that name and package has been deprecated in favor of `ggridges`.



And here's a ridgeline plot for the triceps skinfold. We'll start by sorting the subgroups by the median value of our outcome (triceps skinfold) in this case, though it turns out not to matter. We'll also add some color.

```
nyfs1 %>%
    mutate(bmi.cat = reorder(bmi.cat, triceps.skinfold, median)) %>%
    ggplot(aes(x = triceps.skinfold, y = bmi.cat, fill = bmi.cat, height = ..density..)) +
    ggridges::geom_density_ridges(scale = 0.85) +
    scale_fill_viridis(discrete = TRUE) +
    guides(fill = FALSE) +
    labs(title = "Ridgeline Plot of Triceps Skinfold by BMI category (nyfs1)",
         x = "Triceps Skinfold", y = "BMI Category") +
    theme_bw()
```

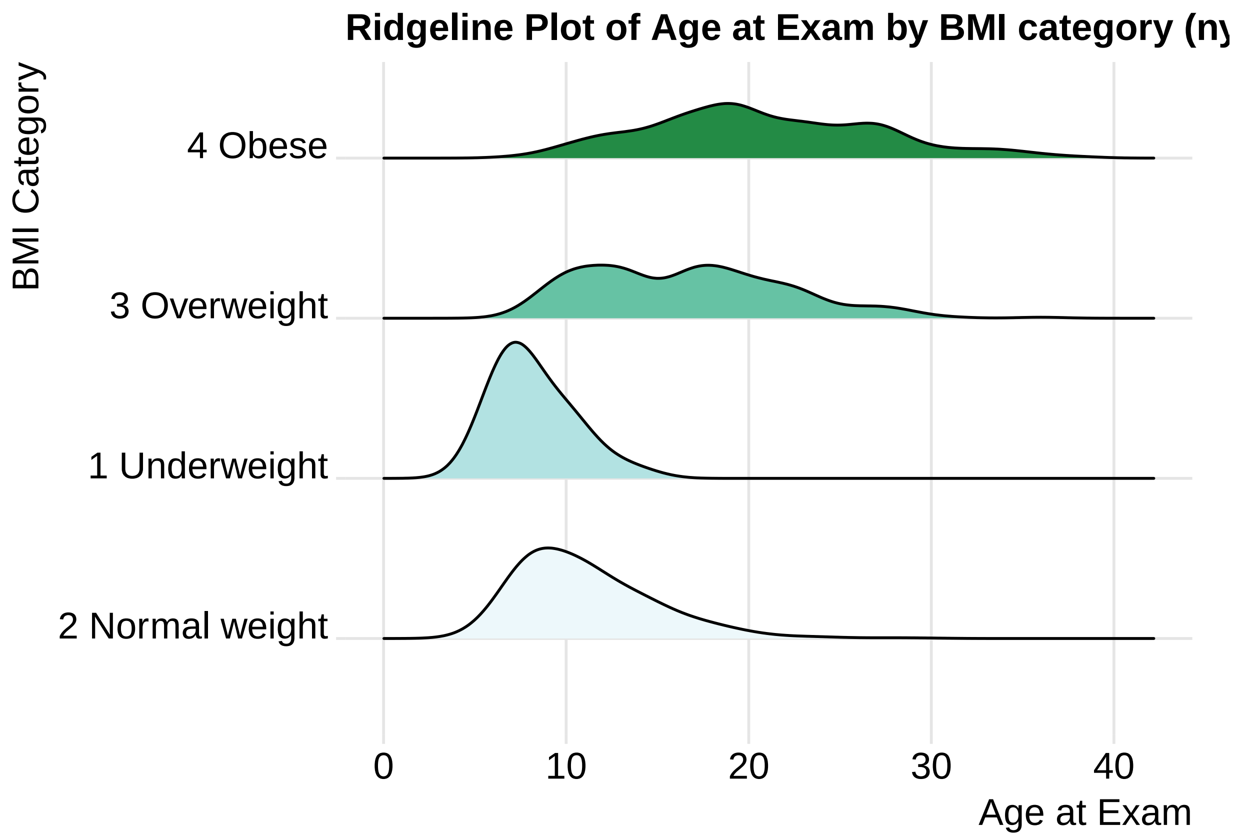
```
Picking joint bandwidth of 1.33
```



For one last example, we'll look at age by BMI category, so that sorting the BMI subgroups by the median matters, and we'll try an alternate color scheme, and a theme specially designed for the ridgeline plot.

```
nyfs1 %>%
    mutate(bmi.cat = reorder(bmi.cat, age.exam, median)) %>%
    ggplot(aes(x = triceps.skinfold, y = bmi.cat, fill = bmi.cat, height = ..density..)) +
    ggridges::geom_density_ridges(scale = 0.85) +
    scale_fill_brewer(palette = 2) +
    guides(fill = FALSE) +
    labs(title = "Ridgeline Plot of Age at Exam by BMI category (nyfs1)",
         x = "Age at Exam", y = "BMI Category") +
    ggridges::theme_ridges()
```

```
Picking joint bandwidth of 1.33
```


Ridgeline Plot of Age at Exam by BMI category (ny
BMI Category
4 Obese
3 Overweight
1 Underweight
2 Normal weight
0
10
20
30
40
Age at Exam

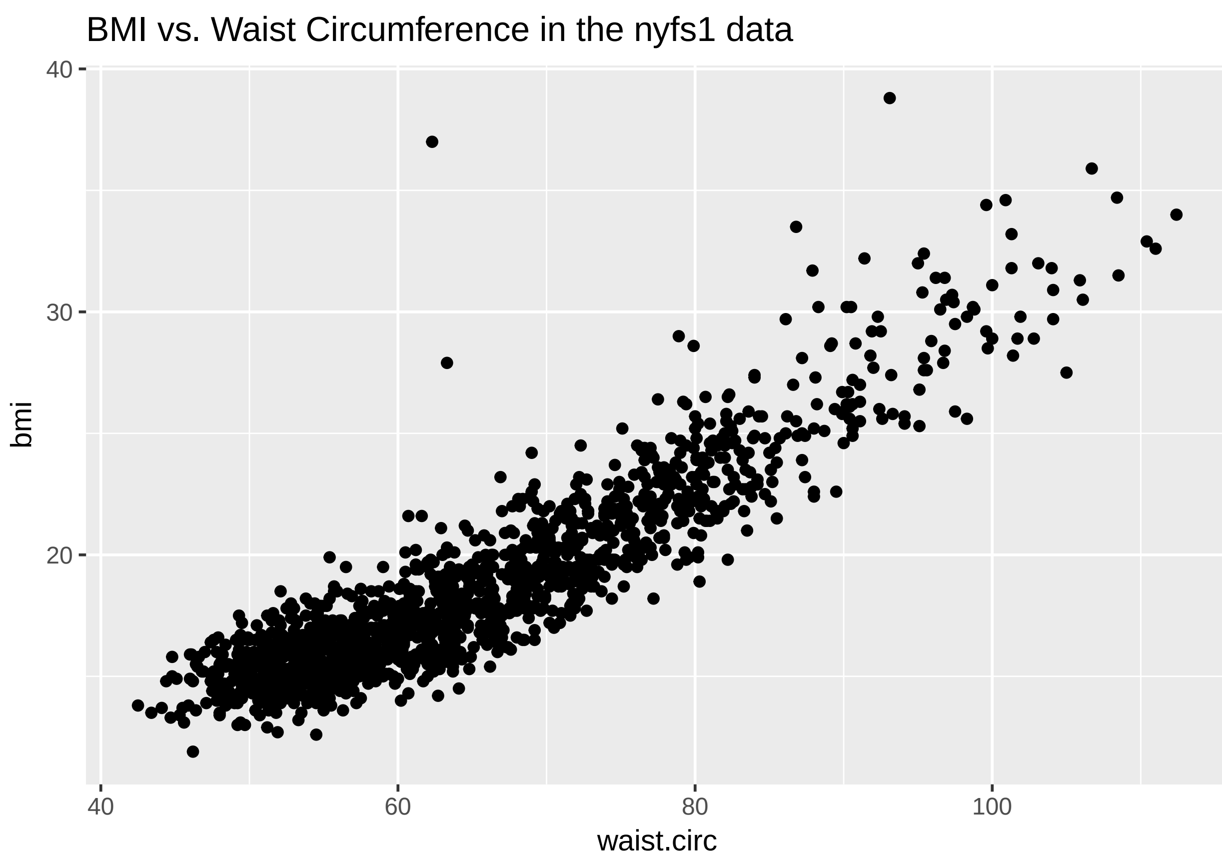
Chapter 11

Straight Line Models and Correlation

11.1 Assessing A Scatterplot

Let's consider the relationship of `bmi` and `waist.circ` in the `nyfs1` data. We'll begin our investigation, as we always should, by drawing a relevant picture. For the association of two quantitative variables, a **scatterplot** is usually the right start. Each subject in the `nyfs1` data is represented by one of the points below.

```
ggplot(data = nyfs1, aes(x = waist.circ, y = bmi)) +
    geom_point() +
    labs(title = "BMI vs. Waist Circumference in the nyfs1 data")
```



Here, I've arbitrarily decided to place `bmi` on the vertical axis, and `waist.circ` on the horizontal. Fitting a prediction model to this scatterplot will then require that we predict `bmi` on the basis of `waist.circ`.

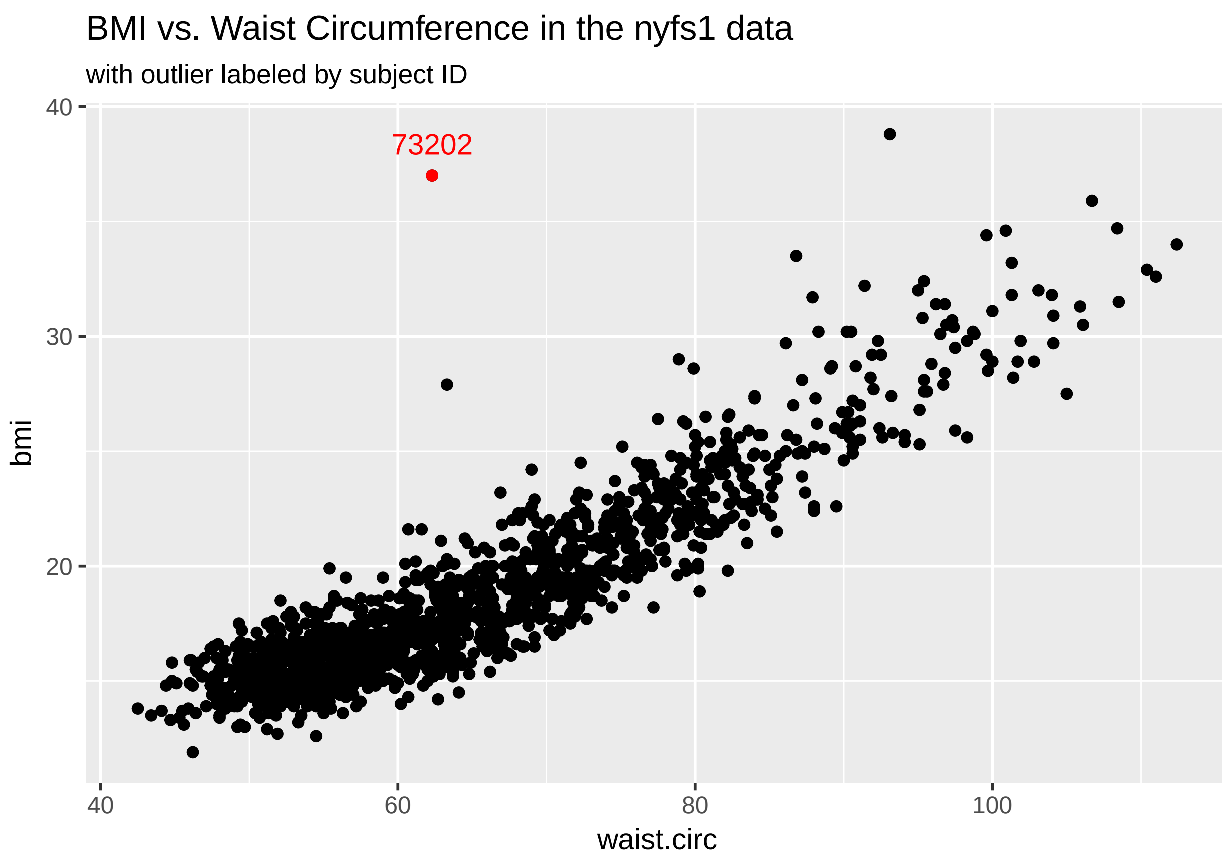
In this case, the pattern appears to be:

1. **direct**, or positive, in that the values of the x variable (`waist.circ`) increase, so do the values of the y variable (`bmi`). Essentially, it appears that subjects with larger waist circumferences also have larger BMIs, but we don't know cause and effect here.
2. fairly **linear** in that most of the points cluster around what appears to be a pattern which is well-fitted by a straight line.
3. **strong** in that the range of values for `bmi` associated with any particular value of `waist.circ` is fairly tight. If we know someone's waist circumference, we can pretty accurately predict their BMI, among the subjects in these data.
4. that we see at least one fairly substantial **outlier** value at the upper left of the plot, which I'll identify in the plot below with a red dot.

11.1.1 Highlighting an unusual point

To highlight the outlier, I'll note that it's the only point with BMI > 35 and waist.circ < 70. So I'll create a subset of the `nyfs1` data containing the point that meets that standard, and then add a red point and a label to the plot.

```r
# identify outlier and place it in data frame s1
s1 <- filter(nyfs1, bmi>35 & waist.circ < 70)

ggplot(data = nyfs1, aes(x = waist.circ, y = bmi)) +
    geom_point() +
    # next two lines add outlier color, and then a label
    geom_point(data = s1, col = "red") +
    geom_text(data = s1, label = s1$subject.id, vjust = -1, col = "red") +
    labs(title = "BMI vs. Waist Circumference in the nyfs1 data",
         subtitle = "with outlier labeled by subject ID")
```




```
s1
```

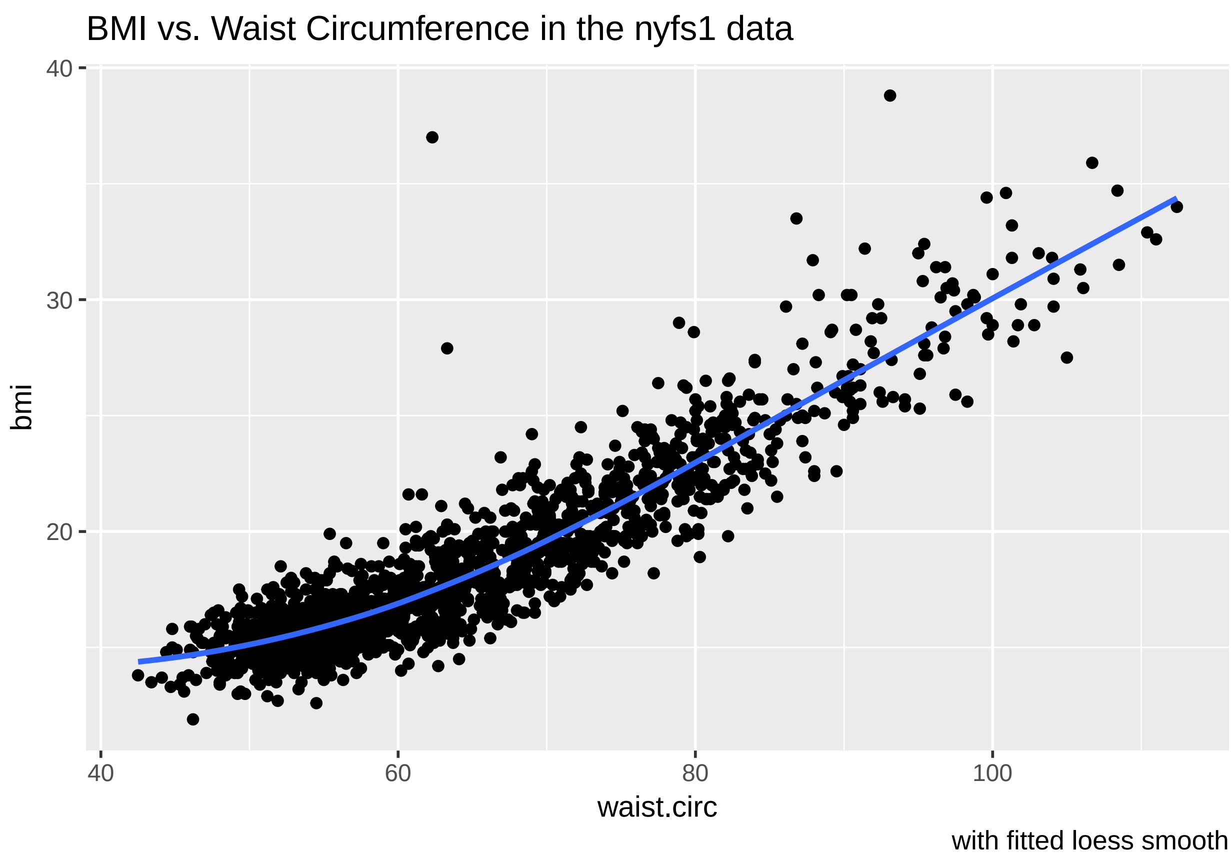
```
# A tibble: 1 x 7
  subject.id    sex age.exam   bmi bmi.cat waist.circ triceps.skinfold
       <int> <fctr>    <int> <dbl>  <fctr>      <dbl>            <dbl>
1      73202   Male       13    37 4 Obese       62.3              7.2
```

Does it seem to you like a straight line model will describe this relationship well?

11.1.2 Adding a Scatterplot Smooth using loess

We'll use the **loess** procedure to fit a smooth curve to the data, which attempts to capture the general pattern.

```
ggplot(data = nyfs1, aes(x = waist.circ, y = bmi)) +
    geom_point() +
    geom_smooth(method = "loess", se = FALSE) +
    labs(title = "BMI vs. Waist Circumference in the nyfs1 data",
         caption = "with fitted loess smooth")
```

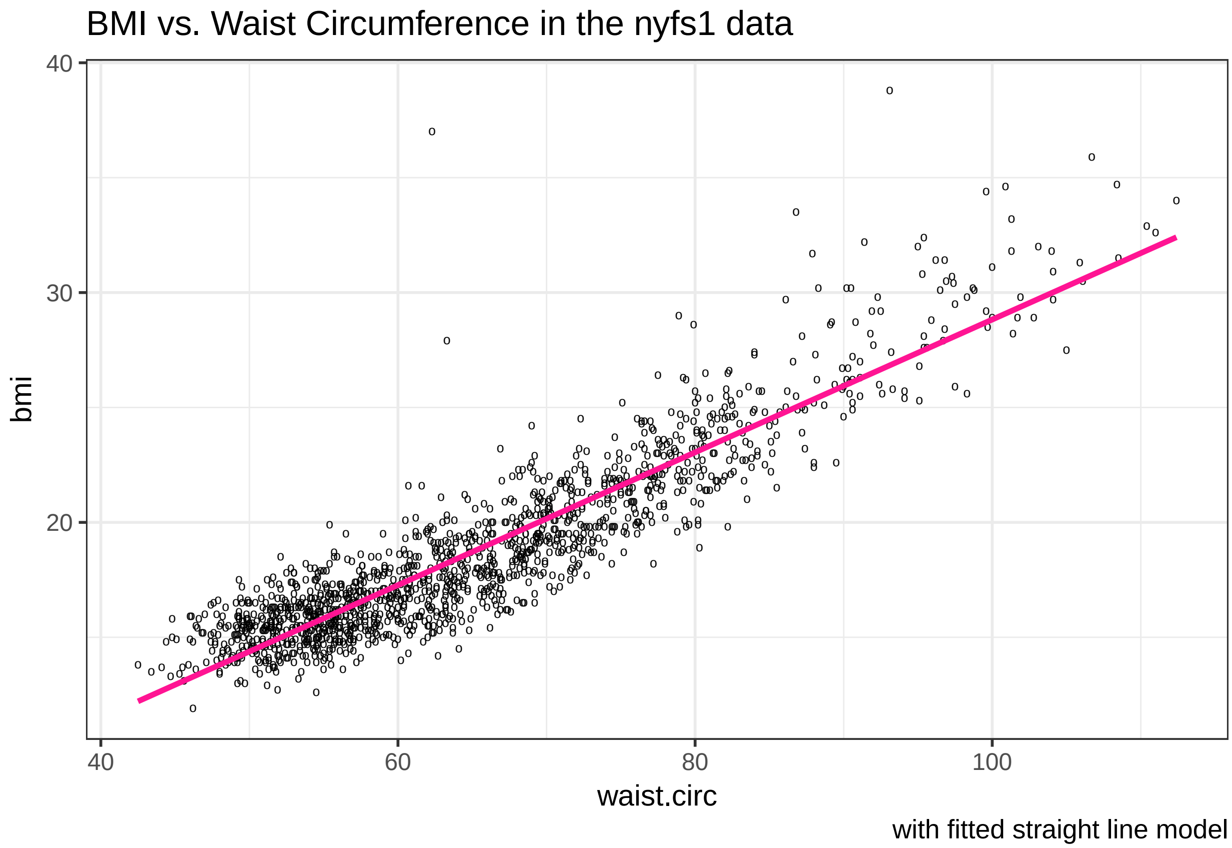


The smooth curve backs up our earlier thought that a straight line might fit the data well. More on the loess smooth in the next chapter.

11.1.3 Adding a Straight Line to the Scatterplot

Let's go ahead and add a straight line to the plot, and we'll change the shape of the points to emphasize the fitted line a bit more.

```
ggplot(data = nyfs1, aes(x = waist.circ, y = bmi)) +
    geom_point(shape = "o") +
    geom_smooth(method = "lm", se = FALSE, col = "deeppink") +
    labs(title = "BMI vs. Waist Circumference in the nyfs1 data",
         caption = "with fitted straight line model") +
    theme_bw()
```



How can we, mathematically, characterize that line? As with any straight line, our model equation requires us to specify two parameters: a slope and an intercept (sometimes called the y-intercept.)

11.1.4 What Line Does R Fit?

To identify the equation R used to fit this line (using the method of least squares), we use the `lm` command

```
lm(bmi ~ waist.circ, data = nyfs1)
```

```
Call:
lm(formula = bmi ~ waist.circ, data = nyfs1)

Coefficients:
(Intercept)   waist.circ
    -0.0665       0.2889
```

So the fitted line is specified as

$$\text{BMI} = -0.066 + 0.289 \text{ Waist Circumference}$$

A detailed summary of the fitted linear regression model is also available.

```
summary(lm(bmi ~ waist.circ, data = nyfs1))
```

```
Call:
```

```
lm(formula = bmi ~ waist.circ, data = nyfs1)

Residuals:
   Min     1Q Median     3Q    Max
-4.234 -1.094 -0.074  0.925 19.066

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.0665     0.2329   -0.29     0.78
waist.circ    0.2889     0.0035   82.55   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.69 on 1414 degrees of freedom
Multiple R-squared:  0.828, Adjusted R-squared:  0.828
F-statistic: 6.81e+03 on 1 and 1414 DF,  p-value: <2e-16
```

We'll spend a lot of time working with these regression summaries, especially in Part C of the course.

For now, it will suffice to understand the following:

- The outcome variable in this model is **bmi**, and the predictor variable is **waist.circ**.
- The straight line model for these data fitted by least squares is `bmi` = -0.066 + 0.289 `waist.circ`
- The slope of `waist.circ` is positive, which indicates that as `waist.circ` increases, we expect that `bmi` will also increase. Specifically, we expect that for every additional cm of waist circumference, the BMI will be 0.289 kg/m^2 larger.
- The multiple R-squared (squared correlation coefficient) is 0.828, which implies that 82.8% of the variation in `bmi` is explained using this linear model with `waist.circ` It also implies that the Pearson correlation between force and height is the square root of 0.828, or 0.91. More on the Pearson correlation soon.

So, if we plan to use a simple (least squares) linear regression model to describe BMI as a function of waist circumference, does it look like a least squares model is likely to be an effective choice here?

11.2 Correlation Coefficients

Two different correlation measures are worth our immediate attention.

- The one most often used is called the *Pearson* correlation coefficient, and is symbolized with the letter r or sometimes the Greek letter rho (ρ).
- Another tool is the Spearman rank correlation coefficient, also occasionally symbolized by ρ).

For the `nyfs1` data, the Pearson correlation of `bmi` and `waist.circ` can be found using the `cor()` function.

```r
cor(nyfs1$bmi, nyfs1$waist.circ)
```

```
[1] 0.91
```

```r
nyfs1 %>%
    select(bmi, waist.circ) %>%
    cor()
```

```
            bmi waist.circ
bmi        1.00       0.91
waist.circ 0.91       1.00
```

Note that the correlation of any variable with itself is 1, and that the correlation of `bmi` with `waist.circ` is the same regardless of whether you enter `bmi` first or `waist.circ` first.

11.3 The Pearson Correlation Coefficient

Suppose we have n observations on two variables, called X and Y. The Pearson correlation coefficient assesses how well the relationship between X and Y can be described using a linear function.

- The Pearson correlation is **dimension-free**.
- It falls between -1 and +1, with the extremes corresponding to situations where all the points in a scatterplot fall exactly on a straight line with negative and positive slopes, respectively.
- A Pearson correlation of zero corresponds to the situation where there is no linear association.
- Unlike the estimated slope in a regression line, the sample correlation coefficient is symmetric in X and Y, so it does not depend on labeling one of them (Y) the response variable, and one of them (X) the predictor.

Suppose we have n observations on two variables, called X and Y, where $\bar{X}$ is the sample mean of X and s_x is the standard deviation of X. The **Pearson** correlation coefficient r_{XY} is:

$$r_{XY} = \frac{1}{n-1}\sum_{i=1}^{n}(\frac{x_i - \bar{x}}{s_x})(\frac{y_i - \bar{y}}{s_y})$$

11.4 A simulated example

The `correx1` data file contains six different sets of (x,y) points, identified by the `set` variable.

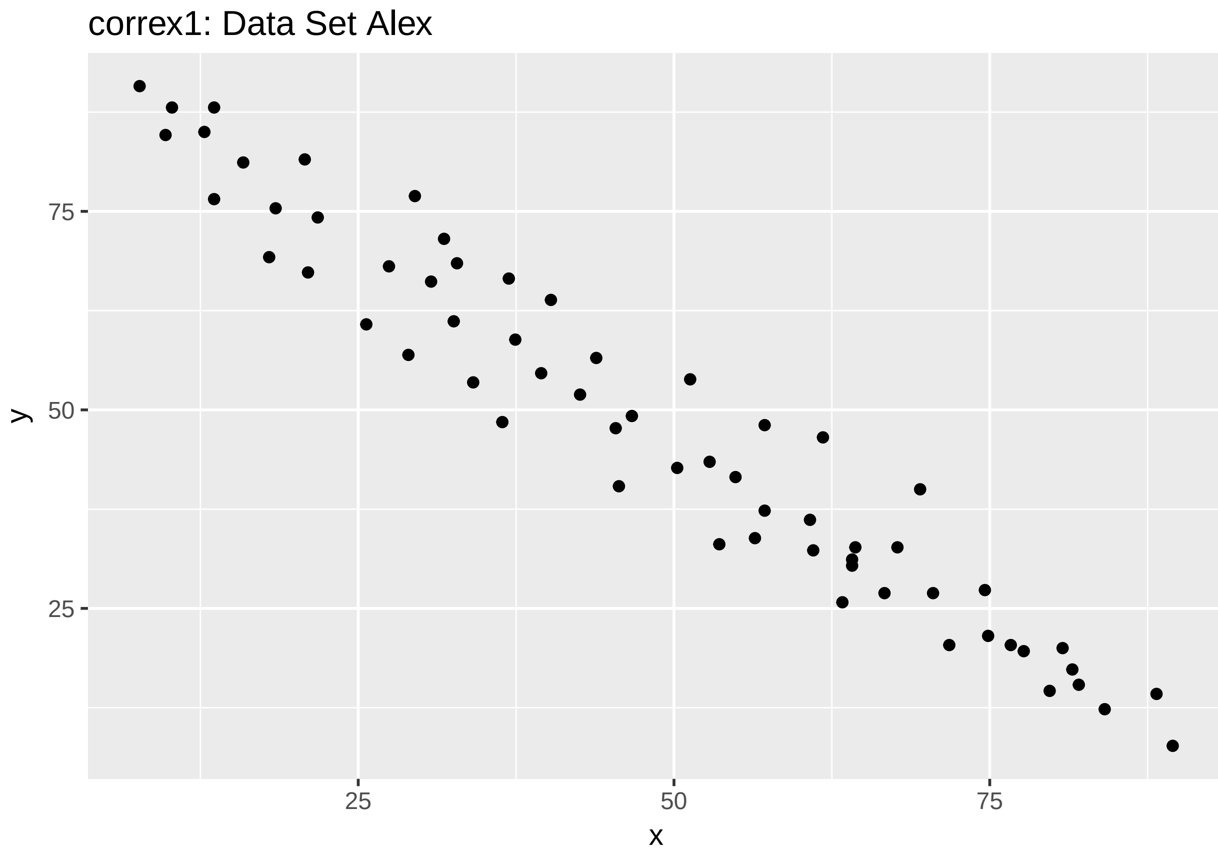
```
correx1 <- read.csv("data/correx1.csv") %>% tbl_df

summary(correx1)
```

```
       set           x               y       
 Alex     :62   Min.   : 5.9   Min.   : 7.3  
 Bonnie   :37   1st Qu.:29.5   1st Qu.:30.4  
 Colin    :36   Median :46.2   Median :46.9  
 Danielle:70   Mean   :46.5   Mean   :49.1  
 Earl     :15   3rd Qu.:63.3   3rd Qu.:68.1  
 Fiona    :57   Max.   :98.2   Max.   :95.4  
```

11.4.1 Data Set Alex

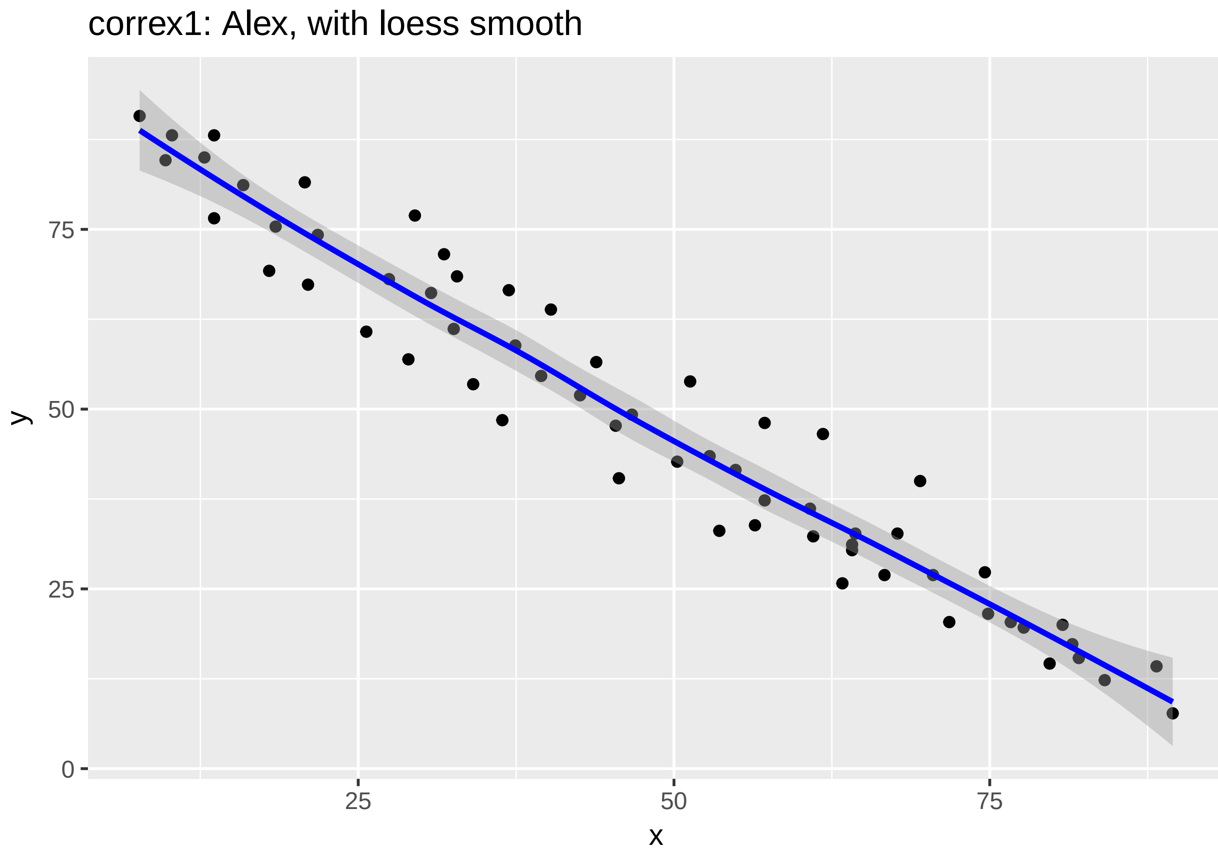
Let’s start by working with the **Alex** data set.

```
ggplot(filter(correx1, set == "Alex"), aes(x = x, y = y)) +
    geom_point() +
    labs(title = "correx1: Data Set Alex")
```

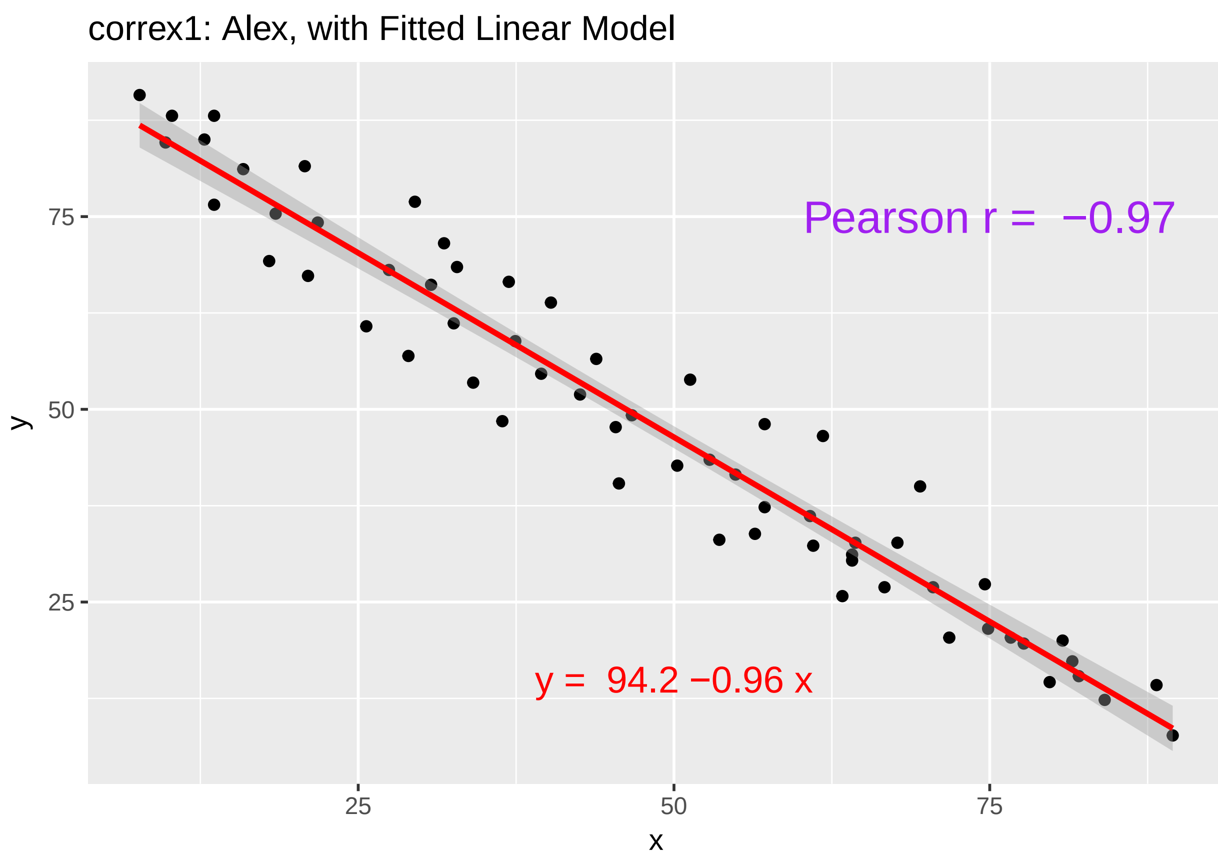



```
ggplot(filter(correx1, set == "Alex"), aes(x = x, y = y)) +
    geom_point() +
    geom_smooth(col = "blue") +
    labs(title = "correx1: Alex, with loess smooth")
```

```
`geom_smooth()` using method = 'loess'
```

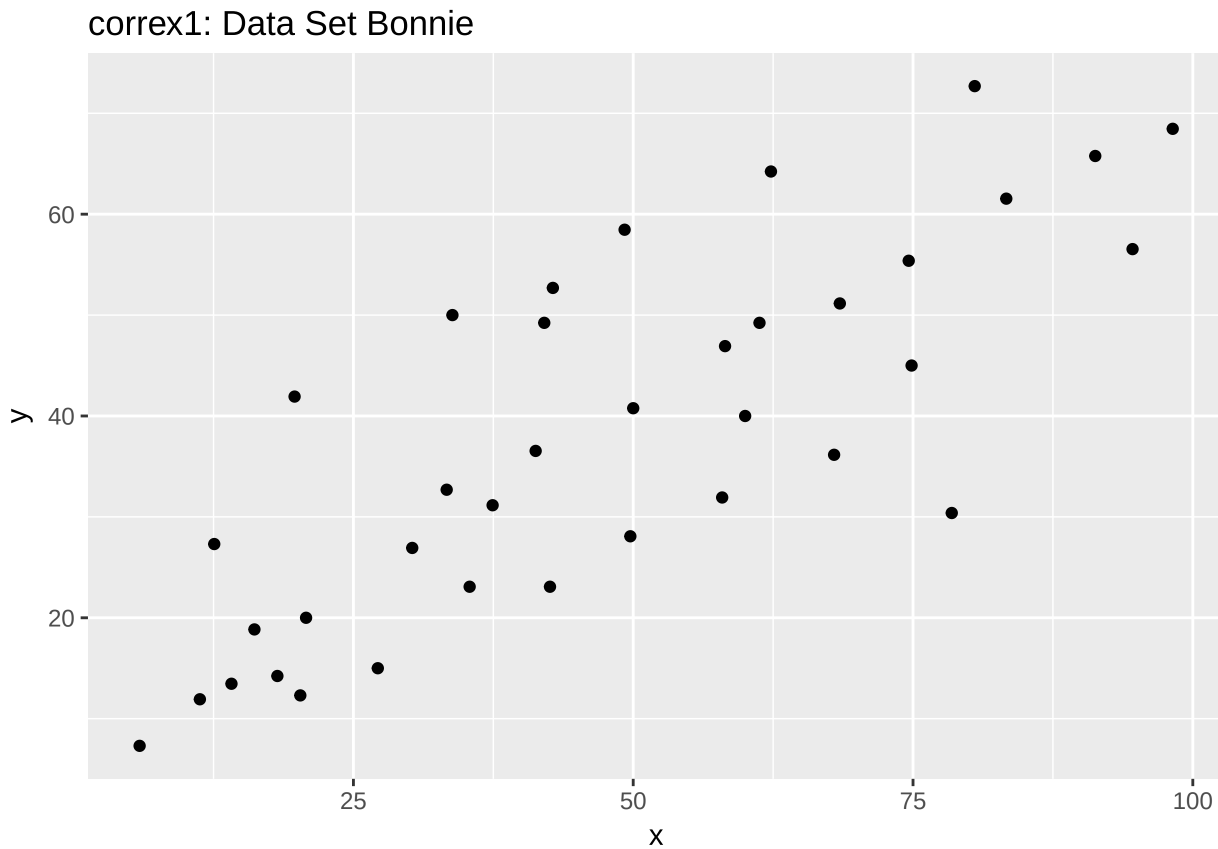



```
setA <- filter(correx1, set == "Alex")

ggplot(setA, aes(x = x, y = y)) +
    geom_point() +
    geom_smooth(method = "lm", col = "red") +
    labs(title = "correx1: Alex, with Fitted Linear Model") +
    annotate("text", x = 75, y = 75, col = "purple", size = 6,
             label = paste("Pearson r = ", signif(cor(setA$x, setA$y),3))) +
    annotate("text", x = 50, y = 15,  col = "red", size = 5,
             label = paste("y = ", signif(coef(lm(setA$y ~ setA$x))[1],3),
                           signif(coef(lm(setA$y ~ setA$x))[2],2), "x"))
```

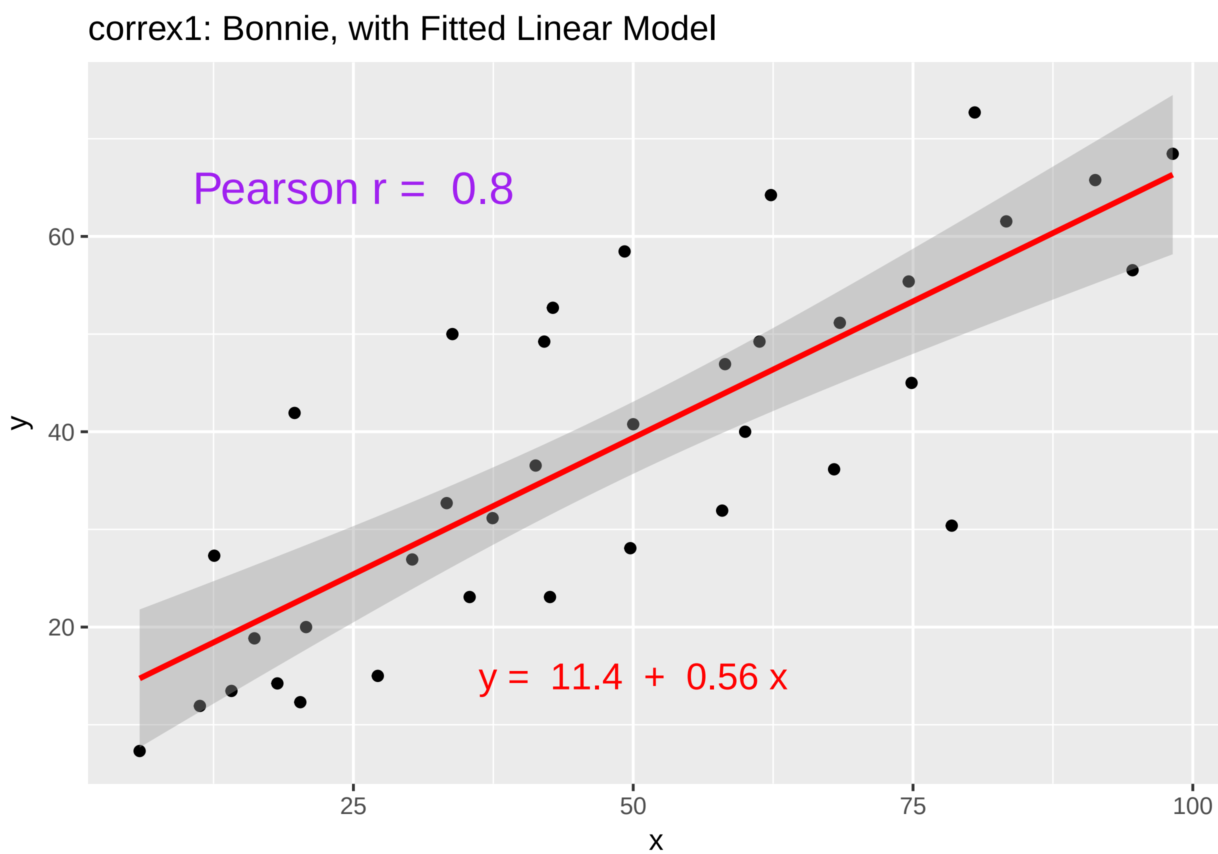


11.4.2 Data Set Bonnie

```
setB <- dplyr::filter(correx1, set == "Bonnie")

ggplot(setB, aes(x = x, y = y)) +
    geom_point() +
    labs(title = "correx1: Data Set Bonnie")
```




```
ggplot(setB, aes(x = x, y = y)) +
    geom_point() +
    geom_smooth(method = "lm", col = "red") +
    labs(title = "correx1: Bonnie, with Fitted Linear Model") +
    annotate("text", x = 25, y = 65, col = "purple", size = 6,
             label = paste("Pearson r = ", signif(cor(setB$x, setB$y),2))) +
    annotate("text", x = 50, y = 15,  col = "red", size = 5,
             label = paste("y = ", signif(coef(lm(setB$y ~ setB$x))[1],3),
                           " + ",
                           signif(coef(lm(setB$y ~ setB$x))[2],2), "x"))
```



11.4.3 Correlations for All Six Data Sets in the Correx1 Example

Let's look at the Pearson correlations associated with each of the six data sets contained in the `correx1` example.

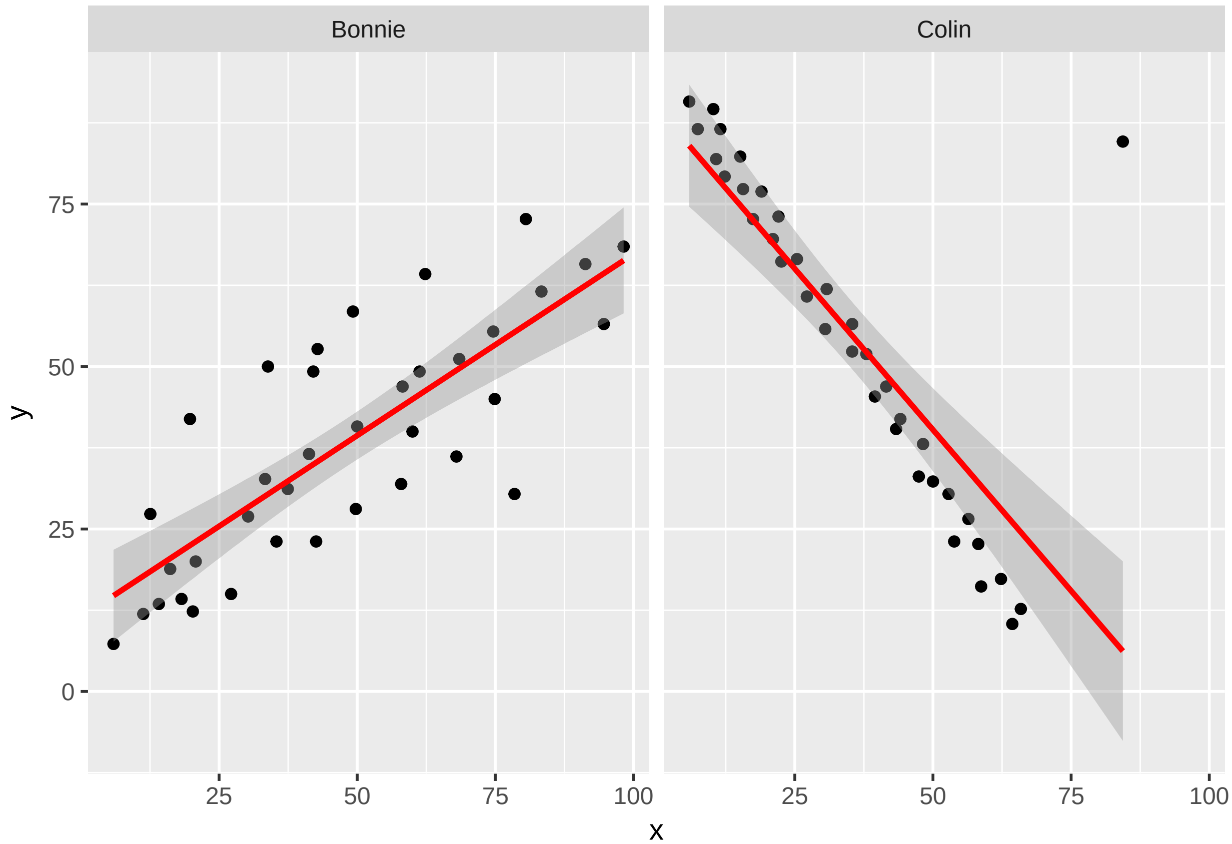
```
tab1 <- correx1 %>%
    group_by(set) %>%
    summarise("Pearson r" = round(cor(x, y, use="complete"),2))

knitr::kable(tab1)
```

set	Pearson r
Alex	-0.97
Bonnie	0.80
Colin	-0.80
Danielle	0.00
Earl	-0.01
Fiona	0.00

11.4.4 Data Set Colin

It looks like the picture for Colin should be very similar (in terms of scatter) to the picture for Bonnie, except that Colin will have a negative slope, rather than the positive one Bonnie has. Is that how this plays out?

```
setBC <- filter(correx1, set == "Bonnie" | set == "Colin")

ggplot(setBC, aes(x = x, y = y)) +
    geom_point() +
    geom_smooth(method = "lm", col = "red") +
    facet_wrap(~ set)
```

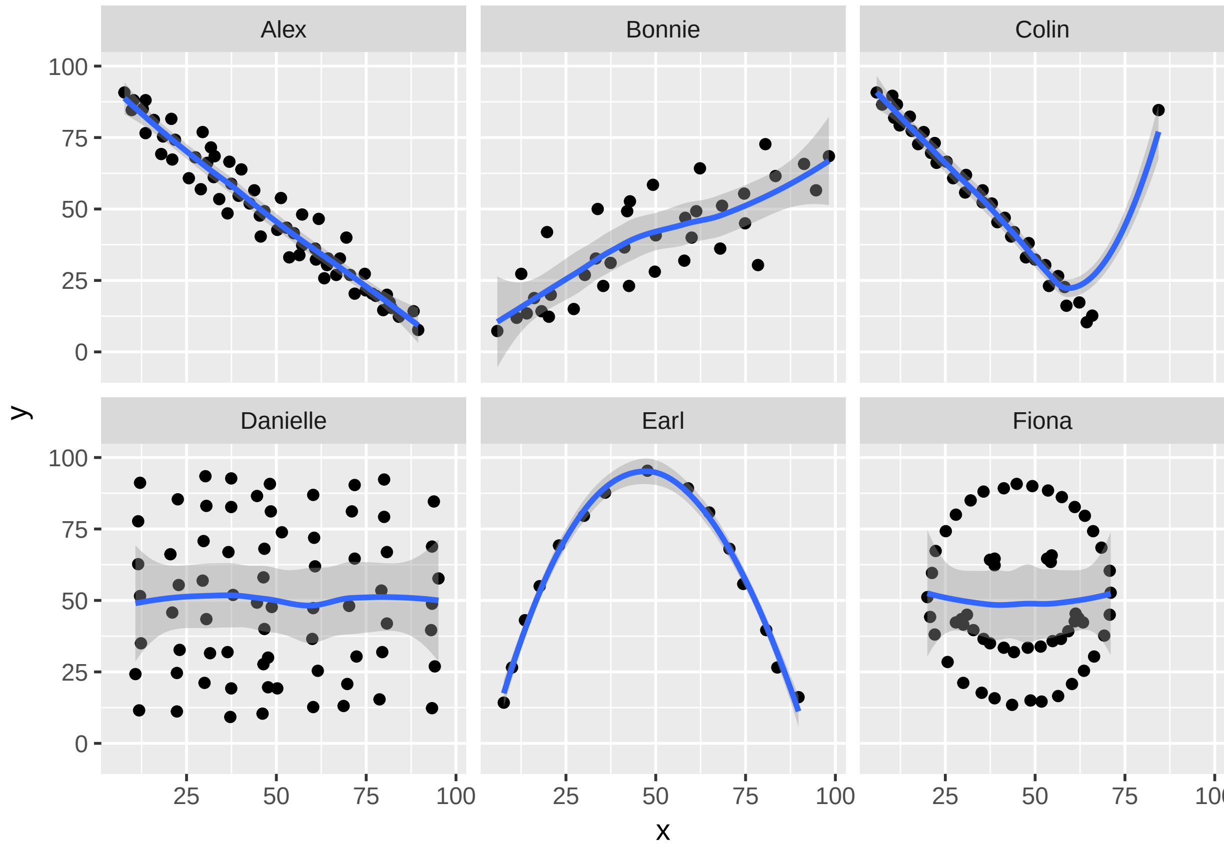


Uh, oh. It looks like the point in Colin at the top right is twisting what would otherwise be a very straight regression model with an extremely strong negative correlation. There's no better way to look for outliers than to examine the scatterplot.

11.4.5 Draw the Picture!

We've seen that Danielle, Earl and Fiona all show Pearson correlations of essentially zero. However, the three data sets look very different in a scatterplot.

```
ggplot(correx1, aes(x = x, y = y)) +
    geom_point() +
    geom_smooth(method = "loess") +
    facet_wrap(~ set)
```



When we learn that the correlation is zero, we tend to assume we have a picture like the Danielle data set. If Danielle were our real data, we might well think that x would be of little use in predicting y.

- But what if our data looked like Earl? In the Earl data set, x is incredibly helpful in predicting y, but we can't use a straight line model - instead, we need a non-linear modeling approach.
- You'll recall that the Fiona data set also had a Pearson correlation of zero. But here, the picture is rather more interesting.

So, remember, draw the d%$# picture whenever you make use of a summary statistic, like a correlation coefficient, or linear model.

```
rm(setA, setB, setBC, tab1)
```

11.5 Estimating Correlation from Scatterplots

The correx2 data set is designed to help you calibrate yourself a bit in terms of estimating a correlation from a scatterplot. There are 11 data sets buried within the correx2 example, and they are labeled by their Pearson correlation coefficients, ranging from r = 0.01 to r = 0.999

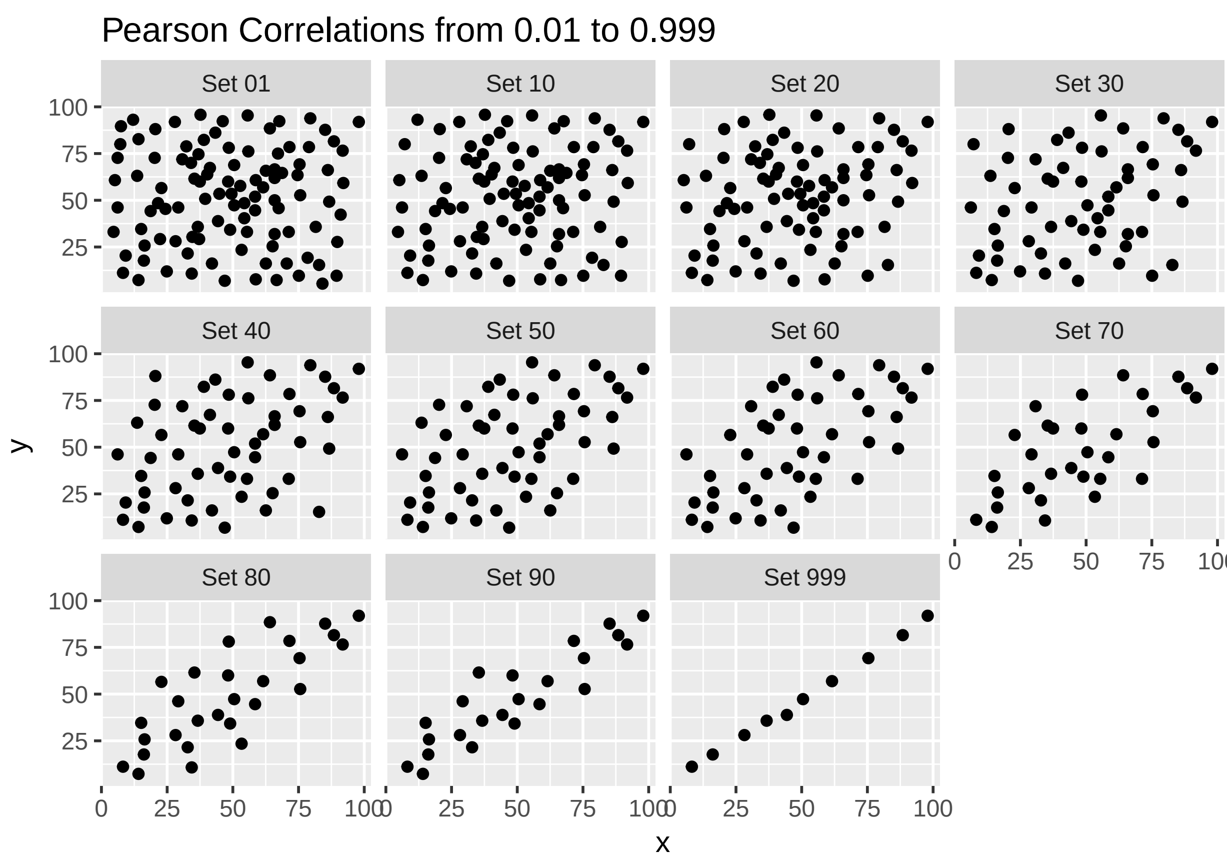
```
correx2 <- read.csv("data/correx2.csv") %>% tbl_df

correx2 %>%
    group_by(set) %>%
    summarise(cor = round(cor(x, y, use="complete"),3))
```

```
# A tibble: 11 x 2
       set   cor
```

```
    <fctr> <dbl>
1   Set 01 0.010
2   Set 10 0.102
3   Set 20 0.202
4   Set 30 0.301
5   Set 40 0.403
6   Set 50 0.499
7   Set 60 0.603
8   Set 70 0.702
9   Set 80 0.799
10  Set 90 0.902
11 Set 999 0.999
```

Here is a plot of the 11 data sets, showing the increase in correlation from 0.01 (in Set 01) to 0.999 (in Set 999).

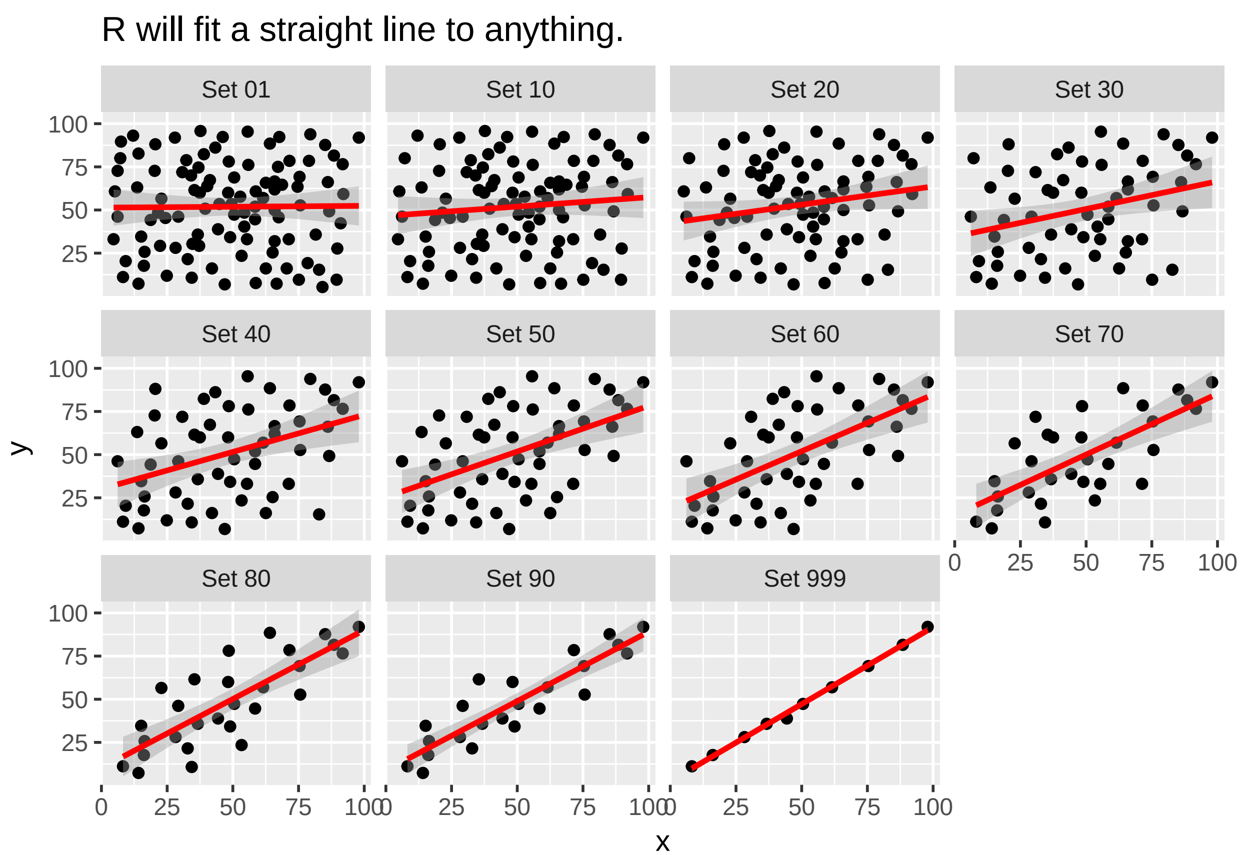
```
ggplot(correx2, aes(x = x, y = y)) +
    geom_point() +
    facet_wrap(~ set) +
    labs(title = "Pearson Correlations from 0.01 to 0.999")
```



Note that R will allow you to fit a straight line model to any of these relationships, no matter how appropriate it might be to do so.

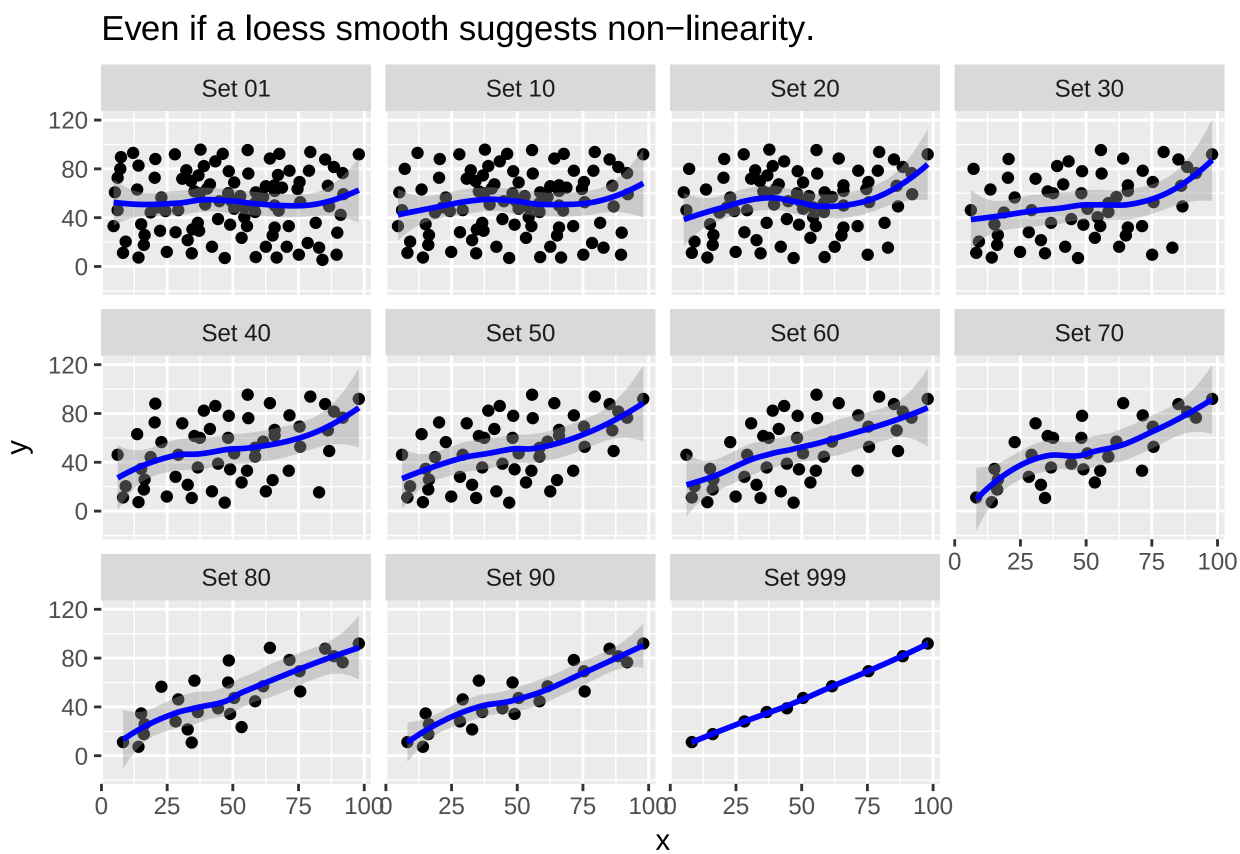
```
ggplot(correx2, aes(x = x, y = y)) +
    geom_point() +
    geom_smooth(method = "lm", col = "red") +
    facet_wrap(~ set) +
```

```
    labs(title = "R will fit a straight line to anything.")
```

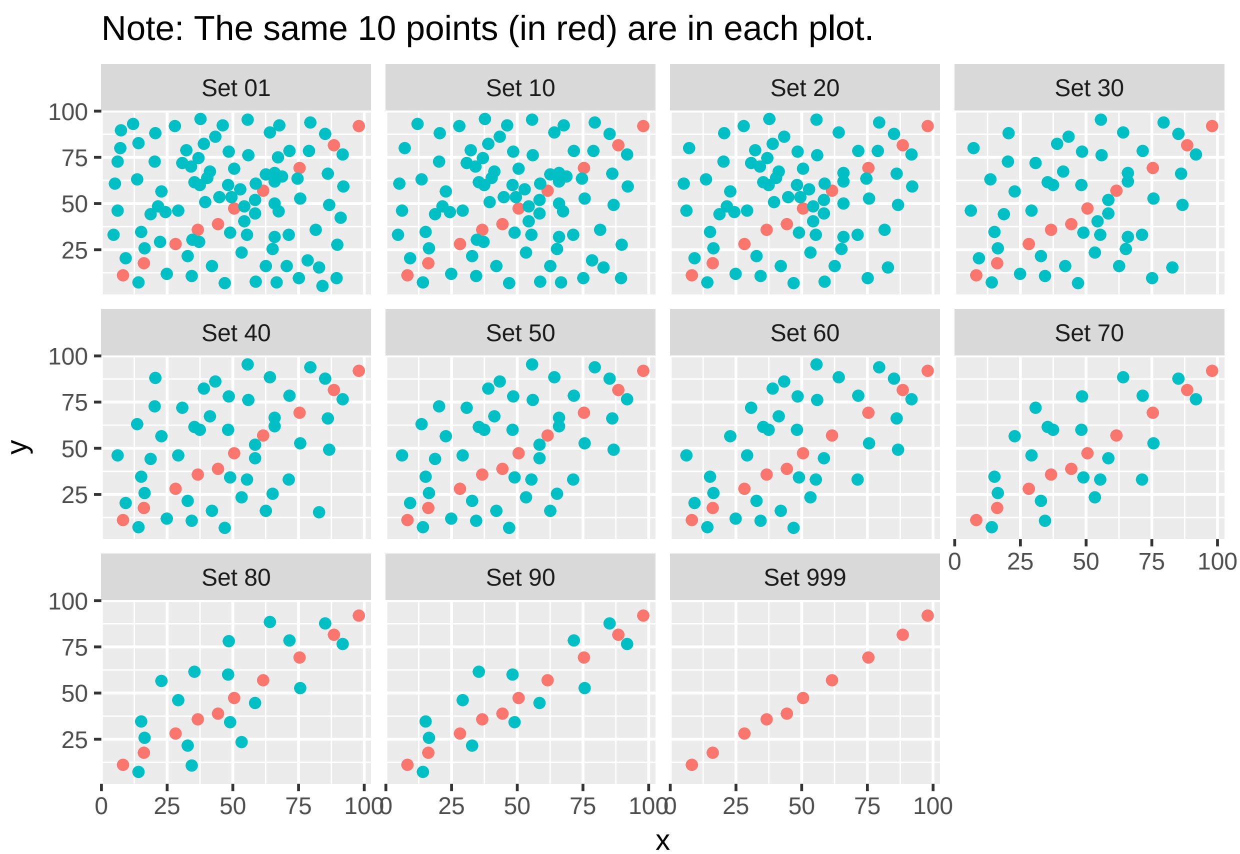



```
ggplot(correx2, aes(x = x, y = y)) +
    geom_point() +
    geom_smooth(col = "blue") +
    facet_wrap(~ set) +
    labs(title = "Even if a loess smooth suggests non-linearity.")
```

```
`geom_smooth()` using method = 'loess'
```




```r
ggplot(correx2, aes(x = x, y = y, color = factor(group))) +
    geom_point() +
    guides(color = "none") +
    facet_wrap(~ set) +
    labs(title = "Note: The same 10 points (in red) are in each plot.")
```



Note that the same 10 points are used in each of the data sets. It's always possible that a lurking subgroup of the data within a scatterplot follows a very strong linear relationship. This is why it's so important (and difficult) not to go searching for such a thing without a strong foundation of logic, theory and prior empirical evidence.

11.6 The Spearman Rank Correlation

The Spearman rank correlation coefficient is a rank-based measure of statistical dependence that assesses how well the relationship between X and Y can be described using a **monotone function** even if that relationship is not linear.

- A monotone function preserves order, that is, Y must either be strictly increasing as X increases, or strictly decreasing as X increases.
- A Spearman correlation of 1.0 indicates simply that as X increases, Y always increases.
- Like the Pearson correlation, the Spearman correlation is dimension-free, and falls between -1 and +1.
- A positive Spearman correlation corresponds to an increasing (but not necessarily linear) association between X and Y, while a negative Spearman correlation corresponds to a decreasing (but again not necessarily linear) association.

11.6.1 Spearman Formula

To calculate the Spearman rank correlation, we take the ranks of the X and Y data, and then apply the usual Pearson correlation. To find the ranks, sort X and Y into ascending order, and then number them from 1 (smallest) to n (largest). In the event of a tie, assign the average rank to the tied subjects.

11.6.2 Comparing Pearson and Spearman Correlations

Let's look at the `nyfs1` data again.

```
cor(nyfs1$bmi, nyfs1$waist.circ)
```

```
[1] 0.91
```

```
cor(nyfs1$bmi, nyfs1$waist.circ, method = "spearman")
```

```
[1] 0.889
```

```
nyfs1 %>%
    select(bmi, waist.circ) %>%
    cor(., method = "spearman")
```

```
             bmi waist.circ
bmi        1.000      0.889
waist.circ 0.889      1.000
```

The Spearman and Pearson correlations are not especially different in this case.

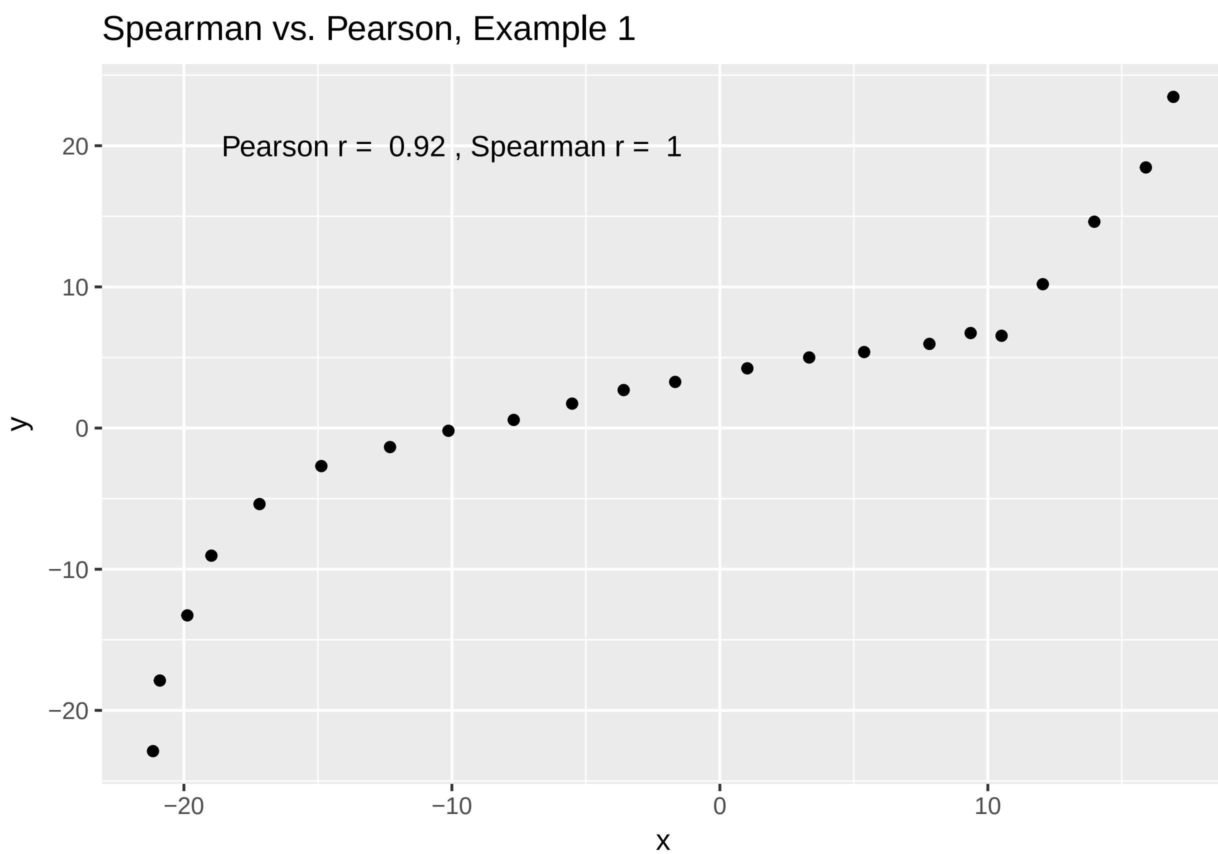
11.6.3 Spearman vs. Pearson Example 1

The next few plots describe relationships where we anticipate the Pearson and Spearman correlations might differ in their conclusions.

```
spear1 <- read.csv("data/spear1.csv")
spear2 <- read.csv("data/spear2.csv")
spear3 <- read.csv("data/spear3.csv")
spear4 <- read.csv("data/spear4.csv")
# used read.csv above because these are just toy examples with
# two columns per data set and no row numbering
```

Example 1 shows a function where the Pearson correlation is 0.925 (a strong but not perfect linear relation), but the Spearman correlation is `signif(cor(spear1$x, spear1$y, method = "spearman"),2)` because the relationship is monotone, even though it is not perfectly linear.

```
ggplot(spear1, aes(x = x, y = y)) +
    geom_point() +
    labs(title = "Spearman vs. Pearson, Example 1") +
    annotate("text", x = -10, y = 20,
             label = paste("Pearson r = ",
                 signif(cor(spear1$x, spear1$y),2),
                 ", Spearman r = ",
                 signif(cor(spear1$x, spear1$y, method = "spearman"),2)))
```

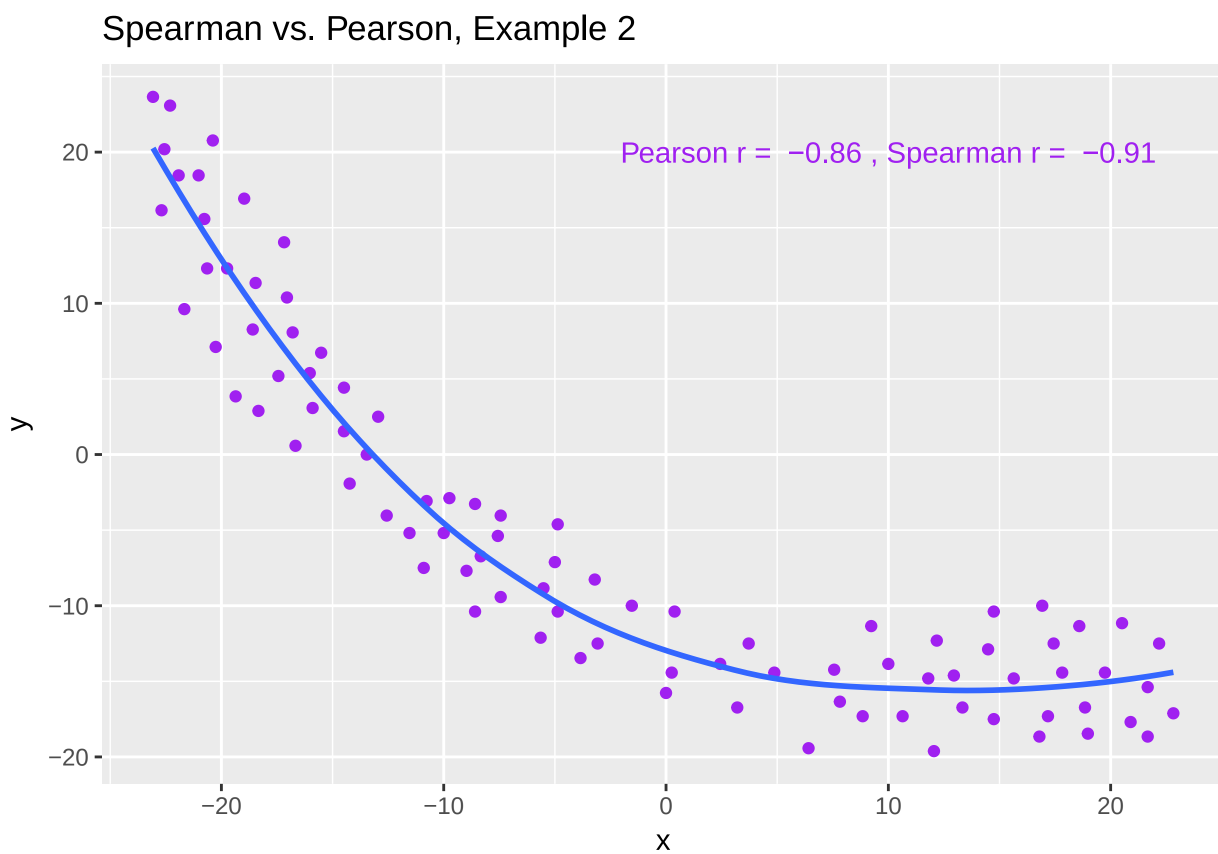


So, a positive Spearman correlation corresponds to an increasing (but not necessarily linear) association between x and y.

11.6.4 Spearman vs. Pearson Example 2

Example 2 shows that a negative Spearman correlation corresponds to a decreasing (but, again, not necessarily linear) association between x and y.

```
ggplot(spear2, aes(x = x, y = y)) +
    geom_point(col = "purple") +
    geom_smooth(method = "loess", se = FALSE) +
    labs(title = "Spearman vs. Pearson, Example 2") +
    annotate("text", x = 10, y = 20, col = "purple",
             label = paste("Pearson r = ",
                    signif(cor(spear2$x, spear2$y),2),
                    ", Spearman r = ",
                    signif(cor(spear2$x, spear2$y, method = "spearman"),2)))
```

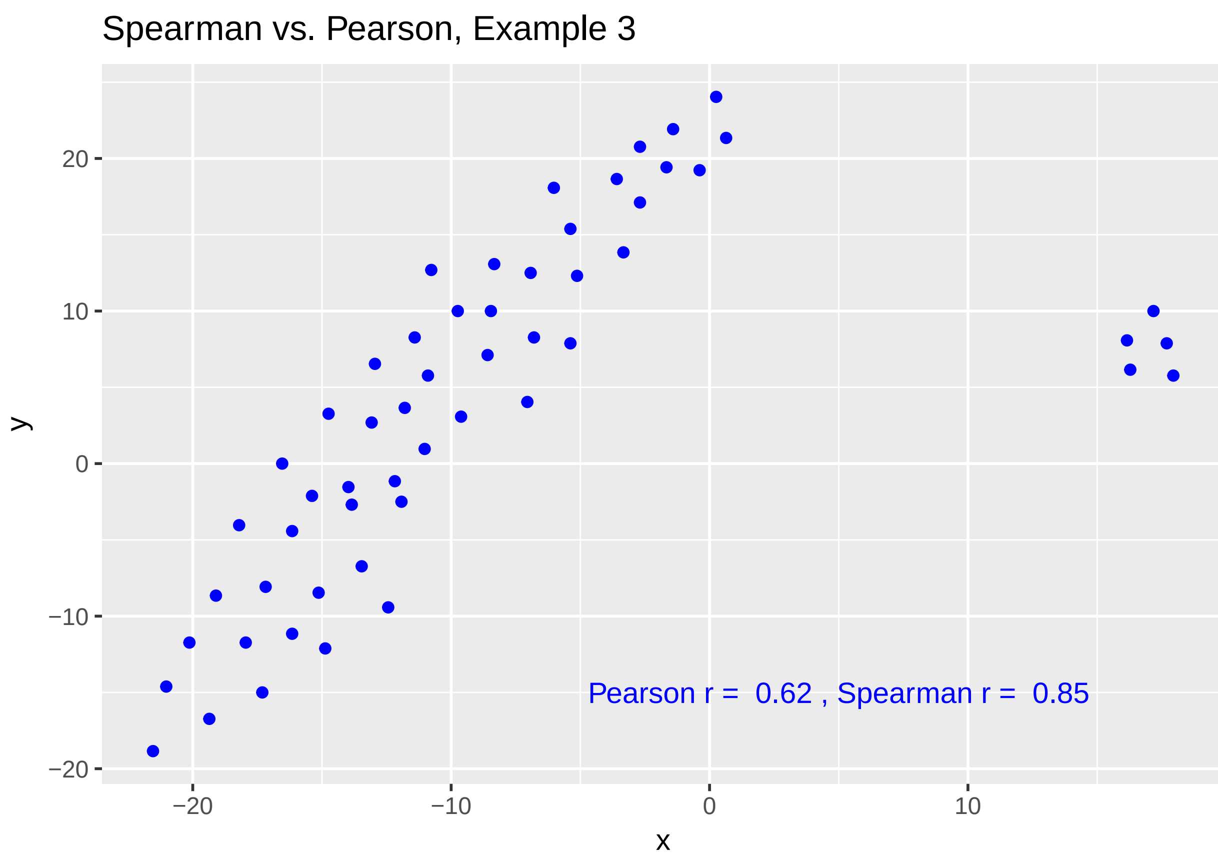


11.6.5 Spearman vs. Pearson Example 3

The Spearman correlation is less sensitive than the Pearson correlation is to strong outliers that are unusual on either the X or Y axis, or both. That is because the Spearman rank coefficient limits the outlier to the value of its rank.

In Example 3, for instance, the Spearman correlation reacts much less to the outliers around X = 12 than does the Pearson correlation.

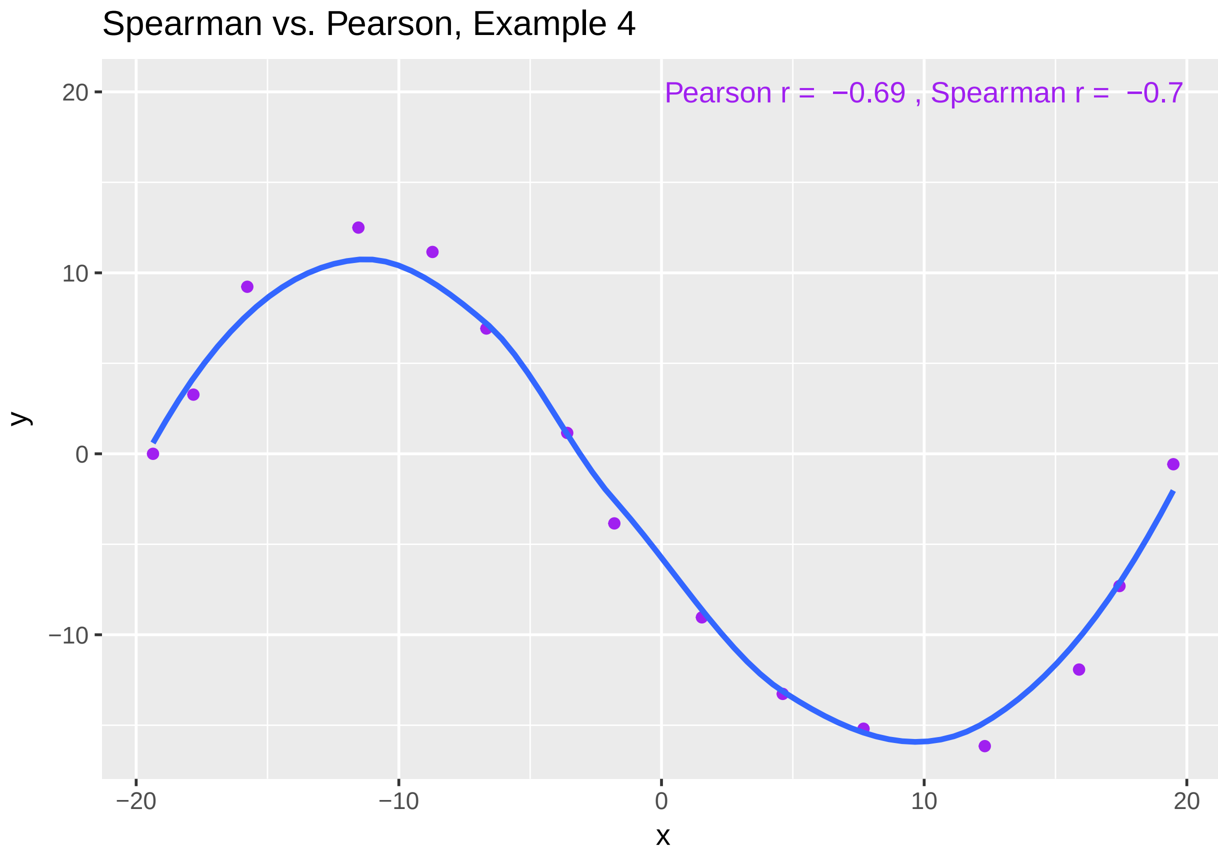
```
ggplot(spear3, aes(x = x, y = y)) +
    geom_point(col = "blue") +
    labs(title = "Spearman vs. Pearson, Example 3") +
    annotate("text", x = 5, y = -15, col = "blue",
             label = paste("Pearson r = ",
                signif(cor(spear3$x, spear3$y),2),
                ", Spearman r = ",
                signif(cor(spear3$x, spear3$y, method = "spearman"),2)))
```



11.6.6 Spearman vs. Pearson Example 4

The use of a Spearman correlation is no substitute for looking at the data. For non-monotone data like what we see in Example 4, neither the Spearman nor the Pearson correlation alone provides much guidance, and just because they are (essentially) telling you the same thing, that doesn't mean what they're telling you is all that helpful.

```
ggplot(spear4, aes(x = x, y = y)) +
    geom_point(col = "purple") +
    geom_smooth(method = "loess", se = FALSE) +
    labs(title = "Spearman vs. Pearson, Example 4") +
    annotate("text", x = 10, y = 20, col = "purple",
             label = paste("Pearson r = ",
                 signif(cor(spear4$x, spear4$y),2),
                 ", Spearman r = ",
                 signif(cor(spear4$x, spear4$y, method = "spearman"),2)))
```


Spearman vs. Pearson, Example 4
Pearson r = −0.69 , Spearman r = −0.7
20
10
0
−10
y
−20
−10
0
10
20
x

Chapter 12

Studying Crab Claws (crabs)

For our next example, we'll consider a study from zoology, specifically carcinology - the study of crustaceans. My source for these data is Chapter 7 in Ramsey and Schafer (2002) which drew the data from a figure in Yamada and Boulding (1998).

The available data are the mean closing forces (in Newtons) and the propodus heights (mm) of the claws on 38 crabs that came from three different species. The *propodus* is the segment of the crab's clawed leg with an immovable finger and palm.

This was part of a study of the effects that predatory intertidal crab species have on populations of snails. The three crab species under study are:

- 14 Hemigraspus nudus, also called the purple shore crab (14 crabs)
- 12 Lophopanopeus bellus, also called the black-clawed pebble crab, and
- 12 Cancer productus, one of several species of red rock crabs (12)

```
crabs <- read.csv("data/crabs.csv") %>% tbl_df

crabs
```

```
# A tibble: 38 x 4
    crab              species force height
   <int>               <fctr> <dbl>  <dbl>
 1     1    Hemigraspus nudus   4.0    8.0
 2     2 Lophopanopeus bellus  15.1    7.9
 3     3      Cancer productus   5.0    6.7
 4     4 Lophopanopeus bellus   2.9    6.6
 5     5    Hemigraspus nudus   3.2    5.0
 6     6    Hemigraspus nudus   9.5    7.9
 7     7      Cancer productus  22.5    9.4
 8     8    Hemigraspus nudus   7.4    8.3
 9     9      Cancer productus  14.6   11.2
10    10 Lophopanopeus bellus   8.7    8.6
# ... with 28 more rows
```

Here's a quick summary of the data. Take care to note the useless results for the first two variables. At least the function flags with a * those variables it thinks are non-numeric.

```
psych::describe(crabs)
```

```
         vars  n  mean    sd median trimmed   mad min  max range skew
crab        1 38 19.50 11.11  19.50   19.50 14.08   1 38.0  37.0 0.00
```

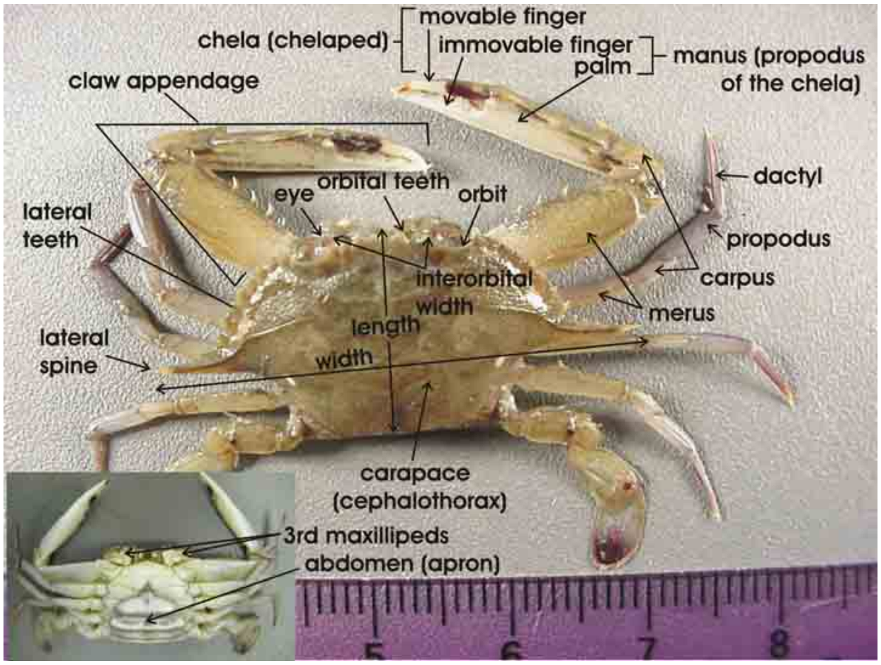


Figure 12.1: Source: http://txmarspecies.tamug.edu/crustglossary.cfm

```
species*    2 38  2.00  0.81   2.00    2.00  1.48   1  3.0   2.0 0.00
force       3 38 12.13  8.98   8.70   11.53  9.04   2 29.4  27.4 0.47
height      4 38  8.81  2.23   8.25    8.78  2.52   5 13.1   8.1 0.19
         kurtosis   se
crab        -1.30 1.80
species*    -1.50 0.13
force       -1.25 1.46
height      -1.14 0.36
```

Actually, we're more interested in these results after grouping by species.

```
crabs %>%
    group_by(species) %>%
    summarise(n = n(), median(force), median(height))
```

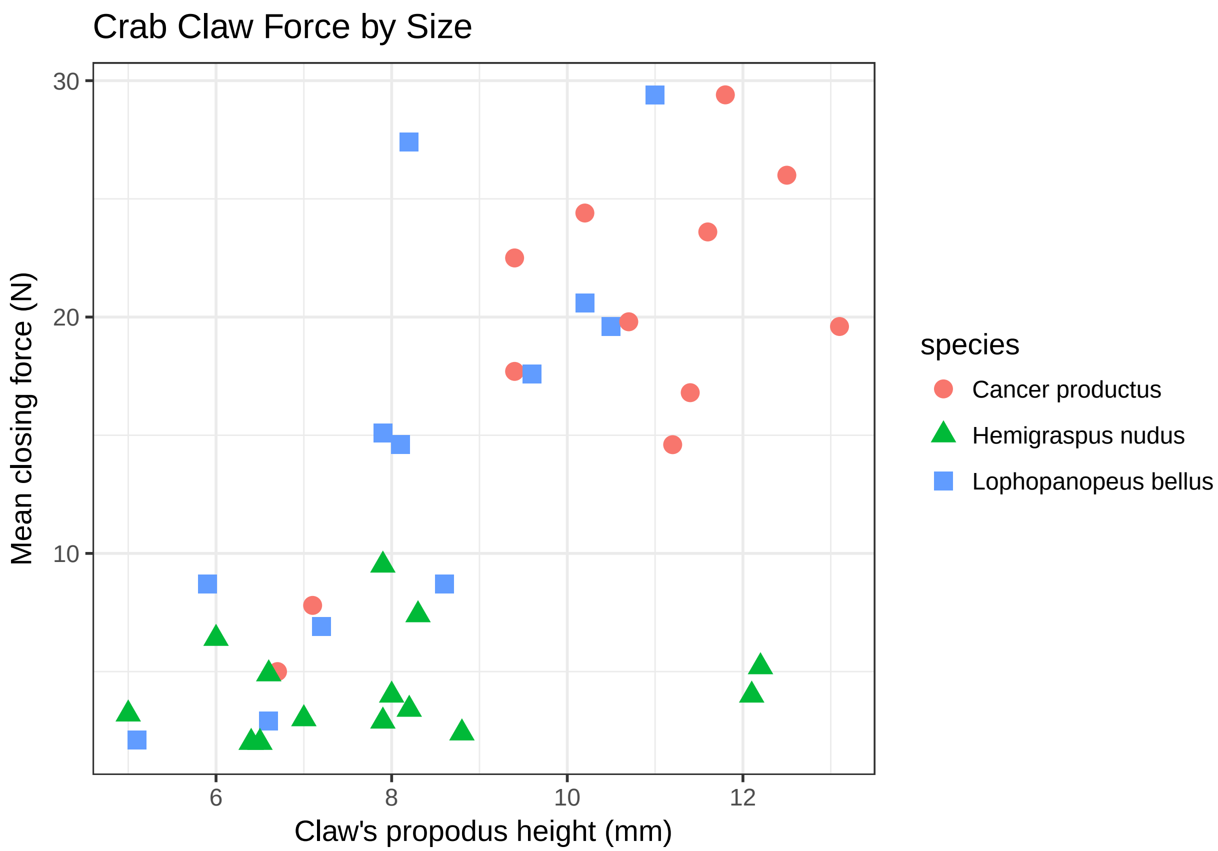
```
# A tibble: 3 x 4
               species     n `median(force)` `median(height)`
                <fctr> <int>           <dbl>            <dbl>
1     Cancer productus    12            19.7            10.95
2    Hemigraspus nudus    14             3.7             7.90
3 Lophopanopeus bellus    12            14.8             8.15
```

12.1 Association of Size and Force

Suppose we want to describe force on the basis of height, across all 38 crabs. We'll add titles and identify the three species of crab, using shape and color.

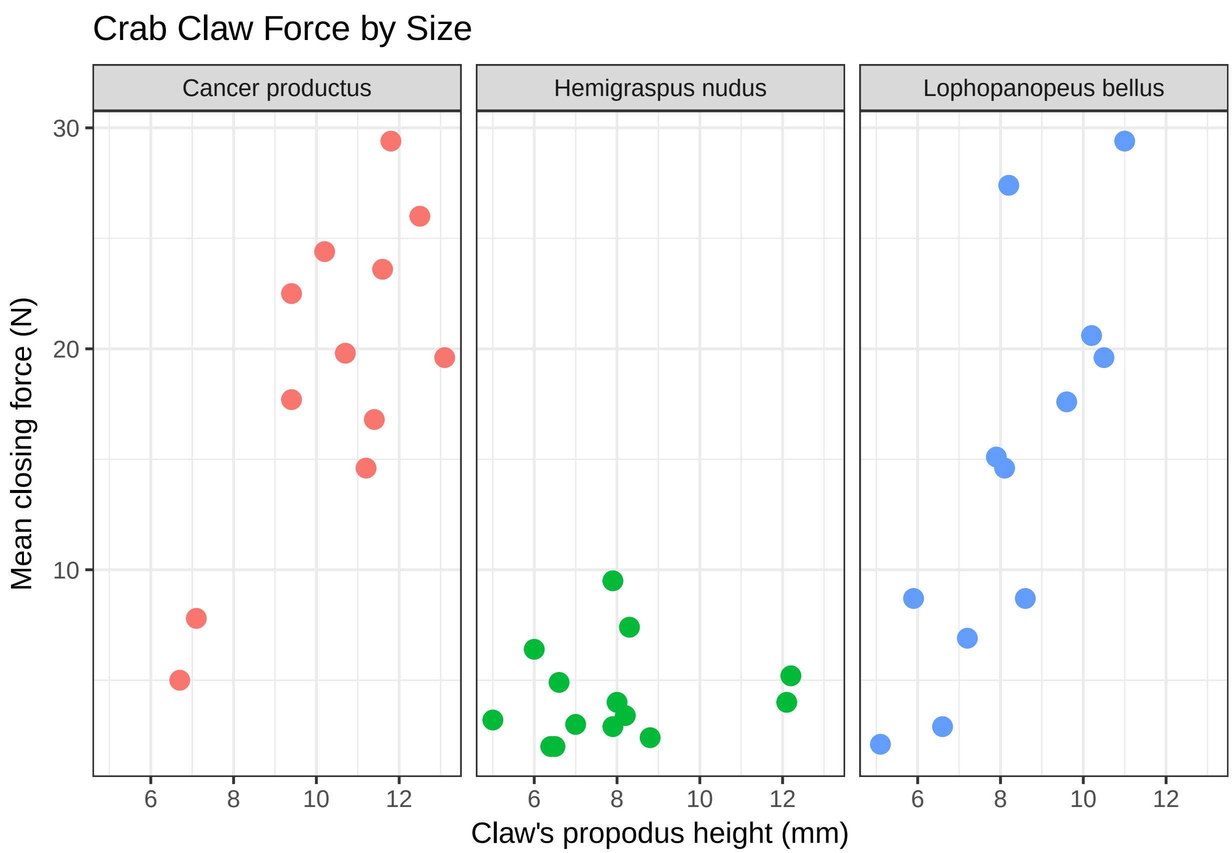
```
ggplot(crabs, aes(x = height, y = force, color = species, shape = species)) +
    geom_point(size = 3) +
    labs(title = "Crab Claw Force by Size",
```

```
        x = "Claw's propodus height (mm)", y = "Mean closing force (N)") +
    theme_bw()
```



A faceted plot for each species really highlights the difference in force between the *Hemigraspus nudus* and the other two species of crab.

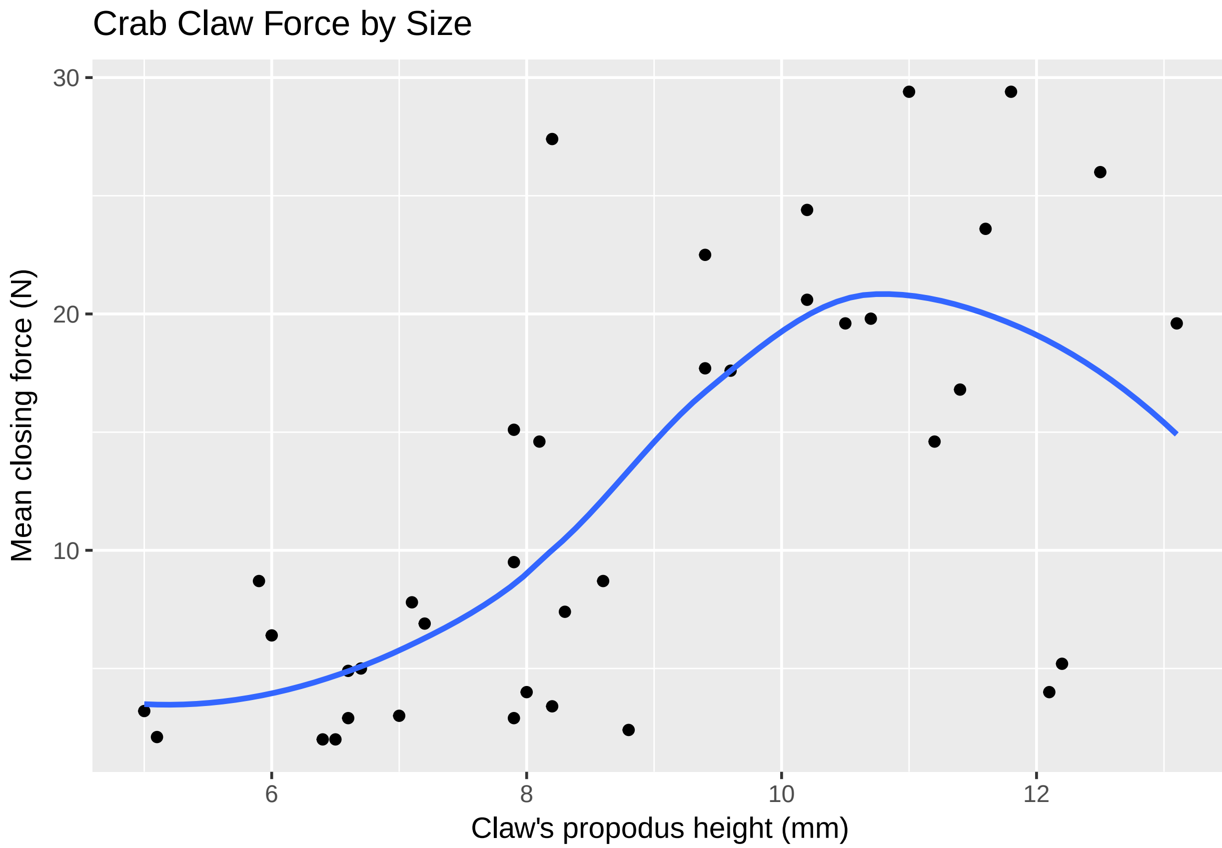
```
ggplot(crabs, aes(x = height, y = force, color = species)) +
    geom_point(size = 3) +
    facet_wrap(~ species) +
    guides(color = FALSE) +
    labs(title = "Crab Claw Force by Size",
        x = "Claw's propodus height (mm)", y = "Mean closing force (N)") +
    theme_bw()
```



12.2 The `loess` smooth

We can obtain a smoothed curve (using several different approaches) to summarize the pattern presented by the data in any scatterplot. For instance, we might build such a plot for the complete set of 38 crabs, adding in a non-linear smooth function (called a loess smooth.)

```
ggplot(crabs, aes(x = height, y = force)) +
    geom_point() +
    geom_smooth(method = "loess", se = FALSE) +
    labs(title = "Crab Claw Force by Size",
         x = "Claw's propodus height (mm)", y = "Mean closing force (N)")
```

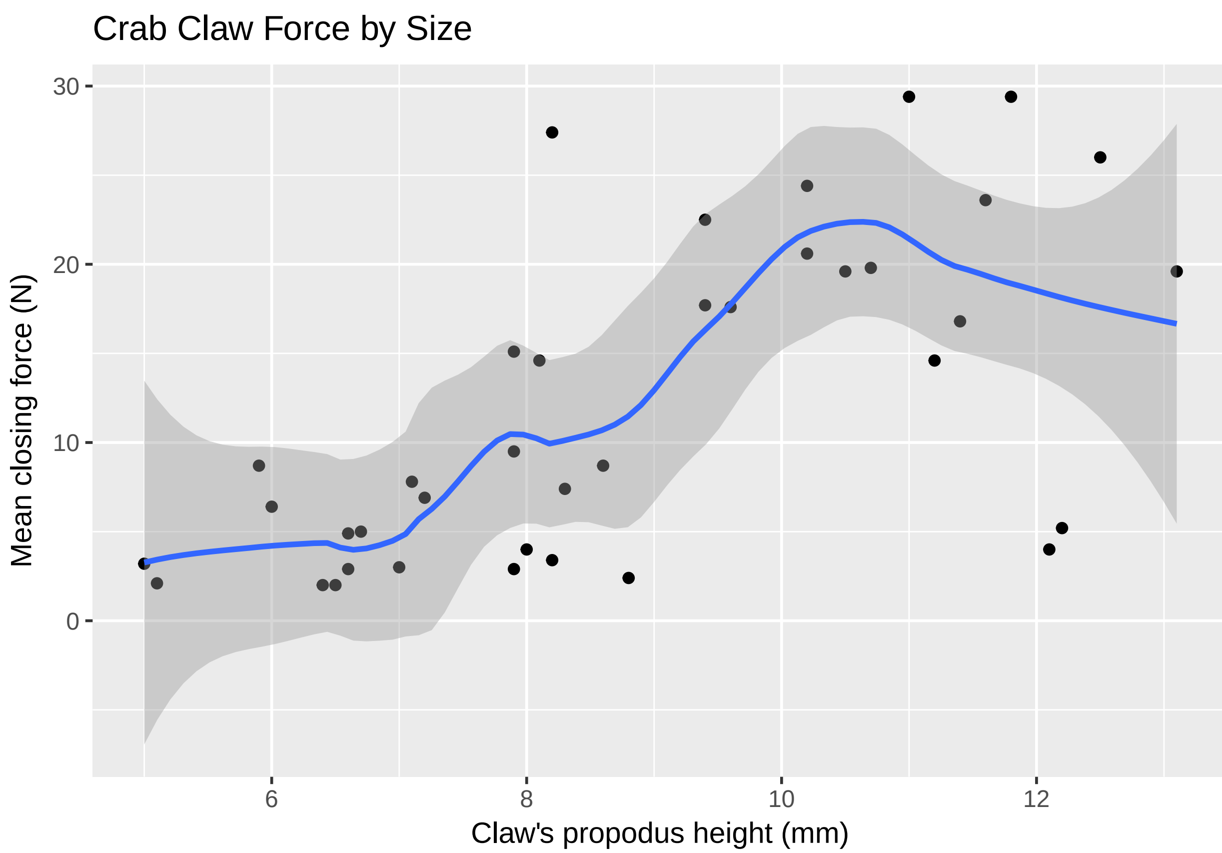


A **loess smooth** is a method of fitting a local polynomial regression model that R uses as its generic smooth for scatterplots with fewer than 1000 observations. Think of the loess as a way of fitting a curve to data by tracking (at point x) the points within a neighborhood of point x, with more emphasis given to points near x. It can be adjusted by tweaking two specific parameters, in particular:

- a `span` parameter (defaults to 0.75) which is also called α in the literature, that controls the degree of smoothing (essentially, how larger the neighborhood should be), and
- a `degree` parameter (defaults to 2) which specifies the degree of polynomial to be used. Normally, this is either 1 or 2 - more complex functions are rarely needed for simple scatterplot smoothing.

In addition to the curve, smoothing procedures can also provide confidence intervals around their main fitted line. Consider the following plot, which adjusts the span and also adds in the confidence intervals.

```
ggplot(crabs, aes(x = height, y = force)) +
    geom_point() +
    geom_smooth(method = "loess", span = 0.5, se = TRUE) +
    labs(title = "Crab Claw Force by Size",
         x = "Claw's propodus height (mm)", y = "Mean closing force (N)")
```

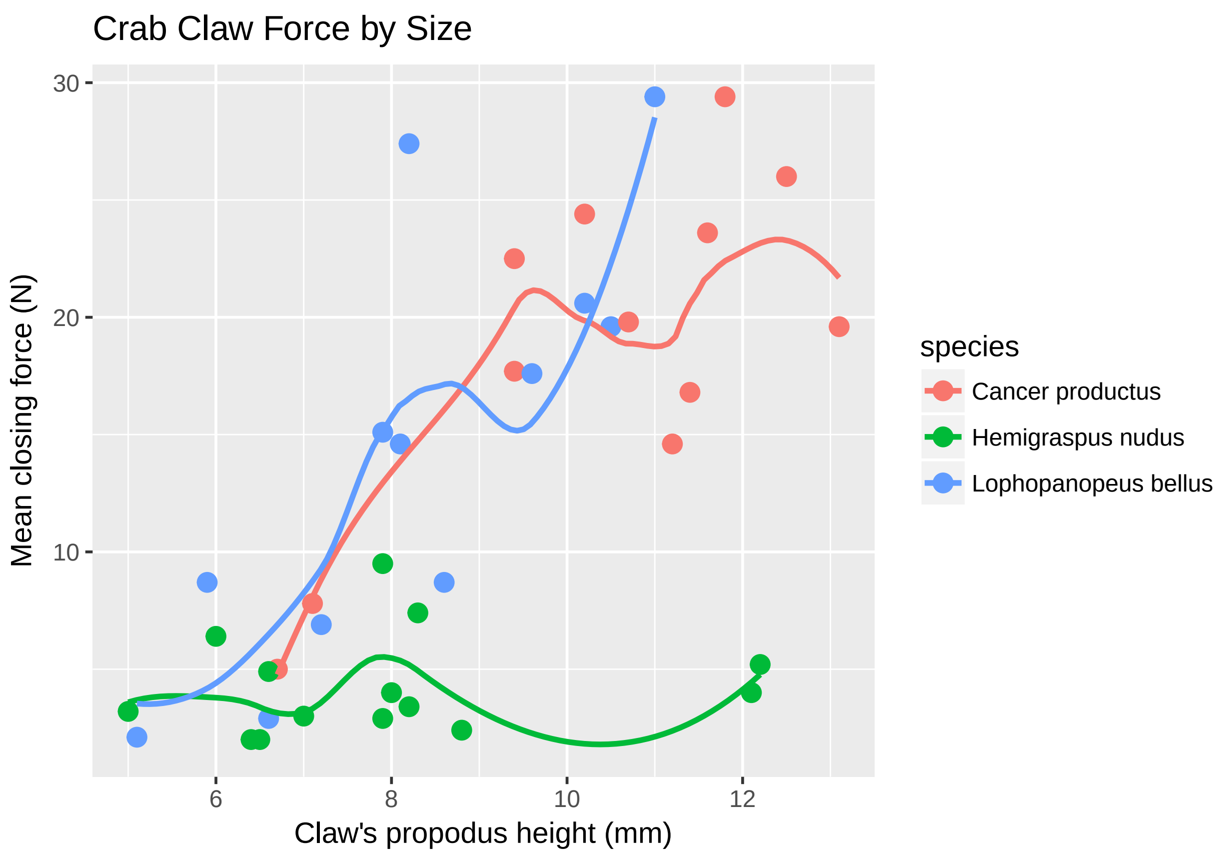


By reducing the size of the span, our resulting picture shows a much less smooth function that we generated previously.

12.2.1 Smoothing within Species

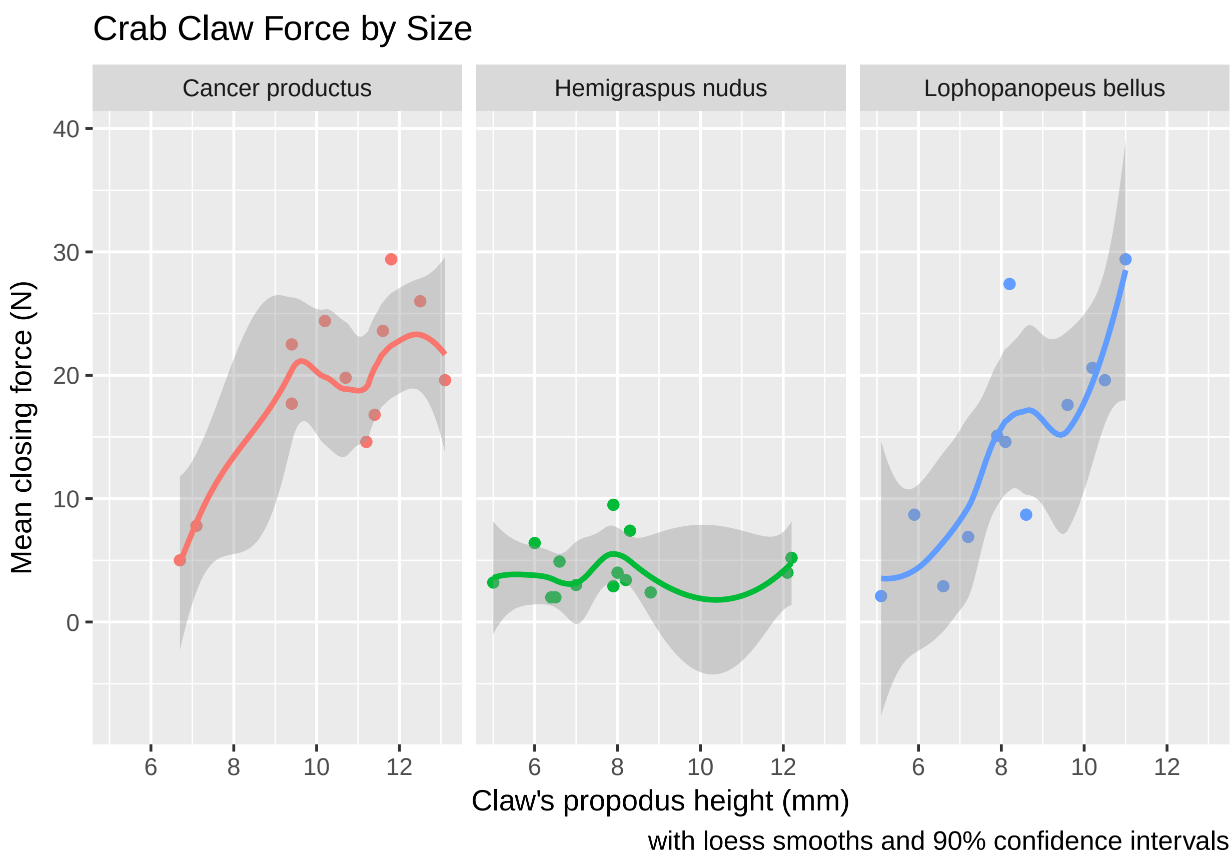
We can, of course, produce the plot above with separate smooths for each of the three species of crab.

```
ggplot(crabs, aes(x = height, y = force, group = species, color = species)) +
    geom_point(size = 3) +
    geom_smooth(method = "loess", se = FALSE) +
    labs(title = "Crab Claw Force by Size",
        x = "Claw's propodus height (mm)", y = "Mean closing force (N)")
```



If we want to add in the confidence intervals (here I'll show them at 90% rather than the default of 95%) then this plot should be faceted. Note that by default, what is displayed when `se = TRUE` are 95% prediction intervals - the `level` function in `stat_smooth` [which can be used in place of `geom_smooth`] is used here to change the coverage percentage from 95% to 90%.

```
ggplot(crabs, aes(x = height, y = force, group = species, color = species)) +
    geom_point() +
    stat_smooth(method = "loess", level = 0.90, se = TRUE) +
    guides(color = FALSE) +
    labs(title = "Crab Claw Force by Size",
         caption = "with loess smooths and 90% confidence intervals",
         x = "Claw's propodus height (mm)", y = "Mean closing force (N)") +
    facet_wrap(~ species)
```



More on these and other confidence intervals later, especially in part B.

12.3 Fitting a Linear Regression Model

Suppose we plan to use a simple (least squares) linear regression model to describe force as a function of height. Is a least squares model likely to be an effective choice here?

The plot below shows the regression line predicting closing force as a function of propodus height. Here we annotate the plot to show the actual fitted regression line, which required fitting it with the `lm` statement prior to developing the graph.

```
mod <- lm(force ~ height, data = crabs)

ggplot(crabs, aes(x = height, y = force)) +
    geom_point() +
    geom_smooth(method = "lm", color = "red") +
    labs(title = "Crab Claw Force by Size with Linear Regression Model",
         x = "Claw's propodus height (mm)", y = "Mean closing force (N)") +
    annotate("text", x = 11, y = 0, color = "red", fontface = "italic",
             label = paste( "Force - ", signif(coef(mod)[1],3), " + ",
                            signif(coef(mod)[2],3), " Height" ))
```


```
rm(mod)
```

The **lm** function, again, specifies the linear model we fit to predict force using height. Here's the summary.

```
summary(lm(force ~ height, data = crabs))
```

```

Call:
lm(formula = force ~ height, data = crabs)

Residuals:
    Min      1Q  Median      3Q     Max 
-16.794  -3.811  -0.239   4.144  16.881 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -11.087      4.622   -2.40    0.022 *  
height         2.635      0.509    5.18  8.7e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.89 on 36 degrees of freedom
Multiple R-squared:  0.427,	Adjusted R-squared:  0.411 
F-statistic: 26.8 on 1 and 36 DF,  p-value: 8.73e-06
```

Again, the key things to realize are:

- The outcome variable in this model is **force**, and the predictor variable is **height**.

- The straight line model for these data fitted by least squares is force = -11.1 + 2.63 height.
- The slope of height is positive, which indicates that as height increases, we expect that force will also increase. Specifically, we expect that for every additional mm of height, the force will increase by 2.63 Newtons.
- The multiple R-squared (squared correlation coefficient) is 0.427, which implies that 42.7% of the variation in force is explained using this linear model with height. It also implies that the Pearson correlation between force and height is the square root of 0.427, or 0.653.

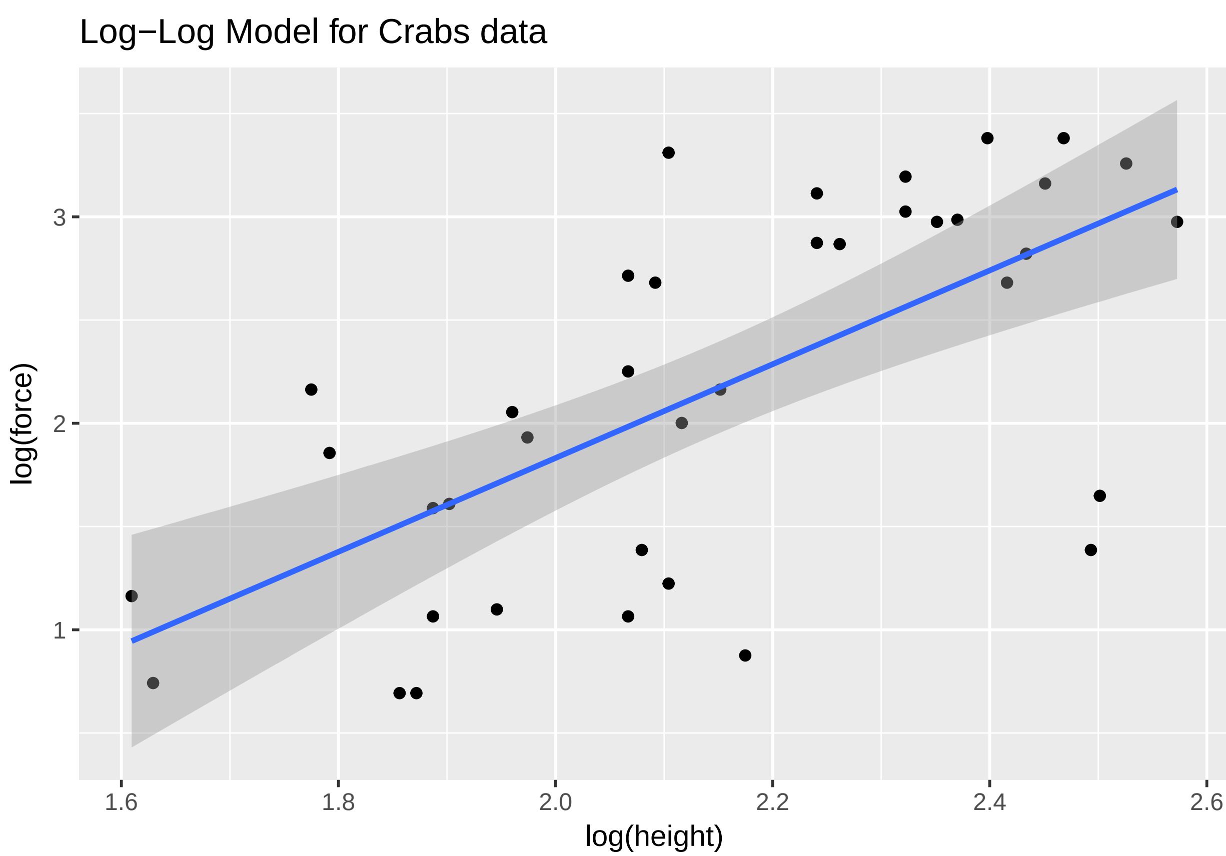
12.4 Is a Linear Model Appropriate?

The zoology (at least as described in Ramsey and Schafer (2002)) suggests that the actual nature of the relationship would be represented by a log-log relationship, where the log of force is predicted by the log of height.

This log-log model is an appropriate model when we think that percentage increases in X (height, here) lead to constant percentage increases in Y (here, force).

To see the log-log model in action, we plot the log of force against the log of height. We could use either base 10 (log10 in R) or natural (log in R) logarithms.

```
ggplot(crabs, aes(x = log(height), y = log(force))) +
    geom_point() +
    geom_smooth(method = "lm") +
    labs(title = "Log-Log Model for Crabs data")
```



The correlations between the raw force and height and between their logarithms turn out to be quite similar, and because the log transformation is monotone in these data, there's actually no change at all in the Spearman correlations.

Correlation of	Pearson r	Spearman r
force and height	0.653	0.657
log(force) and log(height)	0.662	0.657

12.4.1 The log-log model

```
crab_loglog <- lm(log(force) ~ log(height), data = crabs)

summary(crab_loglog)
```

```

Call:
lm(formula = log(force) ~ log(height), data = crabs)

Residuals:
   Min     1Q Median     3Q    Max 
-1.566 -0.445  0.188  0.480  1.242 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   -2.710      0.925   -2.93   0.0059 ** 
log(height)    2.271      0.428    5.30    6e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.675 on 36 degrees of freedom
Multiple R-squared:  0.438, Adjusted R-squared:  0.423 
F-statistic: 28.1 on 1 and 36 DF,  p-value: 5.96e-06
```

Our regression equation is log(force) = -2.71 + 2.27 log(height).

So, for example, if we found a crab with propodus height = 10 mm, our prediction for that crab's claw force (in Newtons) based on this log-log model would be…

- log(force) = -2.71 + 2.27 log(10)
- log(force) = -2.71 + 2.27 x 2.303
- log(force) = 2.519
- and so predicted force = exp(2.519) = 12.417 Newtons, which, naturally, we would round to 12.4 Newtons to match the data set's level of precision.

12.4.2 How does this compare to our original linear model?

```
crab_linear <- lm(force ~ height, data = crabs)

summary(crab_linear)
```

```

Call:
lm(formula = force ~ height, data = crabs)
```

```
Residuals:
    Min      1Q  Median      3Q     Max 
-16.794  -3.811  -0.239   4.144  16.881 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  -11.087      4.622   -2.40    0.022 *  
height         2.635      0.509    5.18  8.7e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.89 on 36 degrees of freedom
Multiple R-squared:  0.427,	Adjusted R-squared:  0.411 
F-statistic: 26.8 on 1 and 36 DF,  p-value: 8.73e-06
```

The linear regression equation is force = -11.1 + 2.63 height.

So, for example, if we found a crab with propodus height = 10 mm, our prediction for that crab's claw force (in Newtons) based on this linear model would be…

- force = -11.087 + 2.635 x 10
- force = -11.087 + 26.348
- so predicted force = 15.261, which we would round to 15.3 Newtons.

So, it looks like the two models give meaningfully different predictions.

12.5 Making Predictions with a Model

A simpler way to get predictions for a new value like height = 10 mm from our models is available.

```
predict(crab_linear, data.frame(height = 10), interval = "prediction")
```

```
   fit  lwr  upr
1 15.3 1.05 29.5
```

We'd interpret this result as saying that the linear model's predicted force associated with a single new crab claw with propodus height 10 mm is 15.3 Newtons, and that a 95% prediction interval for the true value of such a force for such a claw is between 1.0 and 29.5 Newtons. More on prediction intervals later.

12.5.1 Predictions After a Transformation

We can also get predictions from the log-log model.

```
predict(crab_loglog, data.frame(height = 10), interval = "prediction")
```

```
   fit  lwr  upr
1 2.52 1.13 3.91
```

Of course, this prediction is of the `log(force)` for such a crab claw. To get the prediction in terms of simple force, we'd need to back out of the logarithm, by exponentiating our point estimate and the prediction interval endpoints.

```
exp(predict(crab_loglog, data.frame(height = 10), interval = "prediction"))
```

```
   fit  lwr upr
1 12.4 3.08  50
```

We'd interpret this result as saying that the log-log model's predicted force associated with a single new crab claw with propodus height 10 mm is 12.4 Newtons, and that a 95% prediction interval for the true value of such a force for such a claw is between 3.1 and 50.0 Newtons.

12.5.2 Comparing Model Predictions

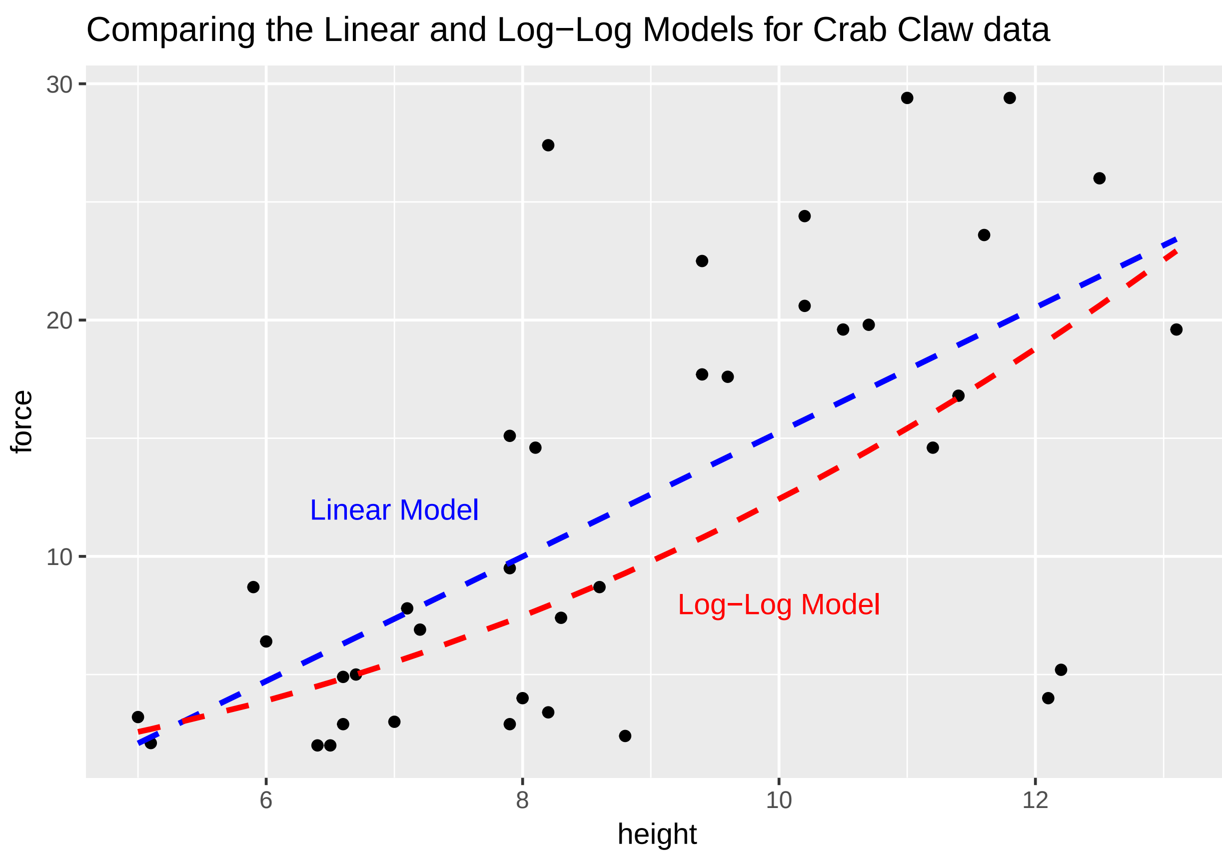
Suppose we wish to build a plot of force vs height with a straight line for the linear model's predictions, and a new curve for the log-log model's predictions, so that we can compare and contrast the implications of the two models on a common scale. The `predict` function, when not given a new data frame, will use the existing predictor values that are in our `crabs` data. Such predictions are often called fitted values.

To put the two sets of predictions on the same scale despite the differing outcomes in the two models, we'll exponentiate the results of the log-log model, and build a little data frame containing the heights and the predicted forces from that model.

```
loglogdat <- data.frame(height = crabs$height, force = exp(predict(crab_loglog)))
```

Now, we're ready to use the `geom_smooth` approach to plot the linear fit, and `geom_line` (which also fits curves) to display the log-log fit.

```
ggplot(crabs, aes(x = height, y = force)) +
    geom_point() +
    geom_smooth(method = "lm", se = FALSE, col="blue", linetype = 2) +
    geom_line(data = loglogdat, col = "red", linetype = 2, size = 1) +
    annotate("text", 7, 12, label = "Linear Model", col = "blue") +
    annotate("text", 10, 8, label = "Log-Log Model", col = "red") +
    labs(title = "Comparing the Linear and Log-Log Models for Crab Claw data")
```



Based on these 38 crabs, we see some modest differences between the predictions of the two models, with the log-log model predicting generally lower closing force for a given propodus height than would be predicted by a linear model.

```
rm(loglogdat, crab_linear, crab_loglog)
```

Chapter 13

The Western Collaborative Group Study

13.1 The Western Collaborative Group Study (wcgs) data set

Vittinghoff et al. (2012) explore data from the Western Collaborative Group Study (WCGS) in great detail[1]. We'll touch lightly on some key issues in this Chapter.

```
wcgs <- read.csv("data/wcgs.csv") %>% tbl_df

wcgs
```

```
# A tibble: 3,154 x 22
      id   age   agec height weight lnwght wghtcat   bmi   sbp lnsbp   dbp
   <int> <int> <fctr>  <int>  <int>  <dbl>  <fctr> <dbl> <int> <dbl> <int>
 1  2343    50  46-50     67    200   5.30 170-200  31.3   132  4.88    90
 2  3656    51  51-55     73    192   5.26 170-200  25.3   120  4.79    74
 3  3526    59  56-60     70    200   5.30 170-200  28.7   158  5.06    94
 4 22057    51  51-55     69    150   5.01 140-170  22.1   126  4.84    80
 5 12927    44  41-45     71    160   5.08 140-170  22.3   126  4.84    80
 6 16029    47  46-50     64    158   5.06 140-170  27.1   116  4.75    76
 7  3894    40  35-40     70    162   5.09 140-170  23.2   122  4.80    78
 8 11389    41  41-45     70    160   5.08 140-170  23.0   130  4.87    84
 9 12681    50  46-50     71    195   5.27 170-200  27.2   112  4.72    70
10 10005    43  41-45     68    187   5.23 170-200  28.4   120  4.79    80
# ... with 3,144 more rows, and 11 more variables: chol <int>,
#   behpat <fctr>, dibpat <fctr>, smoke <fctr>, ncigs <int>, arcus <int>,
#   chd69 <fctr>, typchd69 <int>, time169 <int>, t1 <dbl>, uni <dbl>
```

Here, we have 3154 rows (subjects) and 22 columns (variables).

13.1.1 Structure of wcgs

We can specify the (sometimes terrible) variable names, through the `names` function, or we can add other elements of the structure, so that we can identify elements of particular interest.

[1]For more on the WCGS, you might look at http://www.epi.umn.edu/cvdepi/study-synopsis/western-collaborative-group-study/

```r
str(wcgs)
```

```
Classes 'tbl_df', 'tbl' and 'data.frame':   3154 obs. of  22 variables:
 $ id      : int  2343 3656 3526 22057 12927 16029 3894 11389 12681 10005 ...
 $ age     : int  50 51 59 51 44 47 40 41 50 43 ...
 $ agec    : Factor w/ 5 levels "35-40","41-45",..: 3 4 5 4 2 3 1 2 3 2 ...
 $ height  : int  67 73 70 69 71 64 70 70 71 68 ...
 $ weight  : int  200 192 200 150 160 158 162 160 195 187 ...
 $ lnwght  : num  5.3 5.26 5.3 5.01 5.08 ...
 $ wghtcat : Factor w/ 4 levels "< 140","> 200",..: 4 4 4 3 3 3 3 3 4 4 ...
 $ bmi     : num  31.3 25.3 28.7 22.1 22.3 ...
 $ sbp     : int  132 120 158 126 126 116 122 130 112 120 ...
 $ lnsbp   : num  4.88 4.79 5.06 4.84 4.84 ...
 $ dbp     : int  90 74 94 80 80 76 78 84 70 80 ...
 $ chol    : int  249 194 258 173 214 206 190 212 130 233 ...
 $ behpat  : Factor w/ 4 levels "A1","A2","B3",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ dibpat  : Factor w/ 2 levels "Type A","Type B": 1 1 1 1 1 1 1 1 1 1 ...
 $ smoke   : Factor w/ 2 levels "No","Yes": 2 2 1 1 1 2 1 2 1 2 ...
 $ ncigs   : int  25 25 0 0 0 80 0 25 0 25 ...
 $ arcus   : int  1 0 1 1 0 0 0 0 1 0 ...
 $ chd69   : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ typchd69: int  0 0 0 0 0 0 0 0 0 0 ...
 $ time169 : int  1367 2991 2960 3069 3081 2114 2929 3010 3104 2861 ...
 $ t1      : num  -1.63 -4.06 0.64 1.12 2.43 ...
 $ uni     : num  0.486 0.186 0.728 0.624 0.379 ...
```

13.1.2 Codebook for `wcgs`

This table was lovingly hand-crafted, and involved a lot of typing. We'll look for better ways in 432.

Name	Stored As	Type	Details (units, levels, etc.)
`id`	integer	(nominal)	ID #, nominal and uninteresting
`age`	integer	quantitative	age, in years - no decimal places
`agec`	factor (5)	(ordinal)	age: 35-40, 41-45, 46-50, 51-55, 56-60
`height`	integer	quantitative	height, in inches
`weight`	integer	quantitative	weight, in pounds
`lnwght`	number	quantitative	natural logarithm of weight
`wghtcat`	factor (4)	(ordinal)	wt: < 140, 140-170, 170-200, > 200
`bmi`	number	quantitative	body-mass index: 703 * weight in lb / (height in in)^2
`sbp`	integer	quantitative	systolic blood pressure, in mm Hg
`lnsbp`	number	quantitative	natural logarithm of sbp
`dbp`	integer	quantitative	diastolic blood pressure, mm Hg
`chol`	integer	quantitative	total cholesterol, mg/dL
`behpat`	factor (4)	(nominal)	behavioral pattern: A1, A2, B3 or B4
`dibpat`	factor (2)	(binary)	behavioral pattern: A or B
`smoke`	factor (2)	(binary)	cigarette smoker: Yes or No
`ncigs`	integer	quantitative	number of cigarettes smoked per day
`arcus`	integer	(nominal)	arcus senilis present (1) or absent (0)
`chd69`	factor (2)	(binary)	CHD event: Yes or No
`typchd69`	integer	(4 levels)	event: 0 = no CHD, 1 = MI or SD, 2 = silent MI, 3 = angina

Name	Stored As	Type	Details (units, levels, etc.)
`time169`	integer	quantitative	follow-up time in days
`t1`	number	quantitative	heavy-tailed (random draws)
`uni`	number	quantitative	light-tailed (random draws)

13.1.3 Quick Summary

```
summary(wcgs)
```

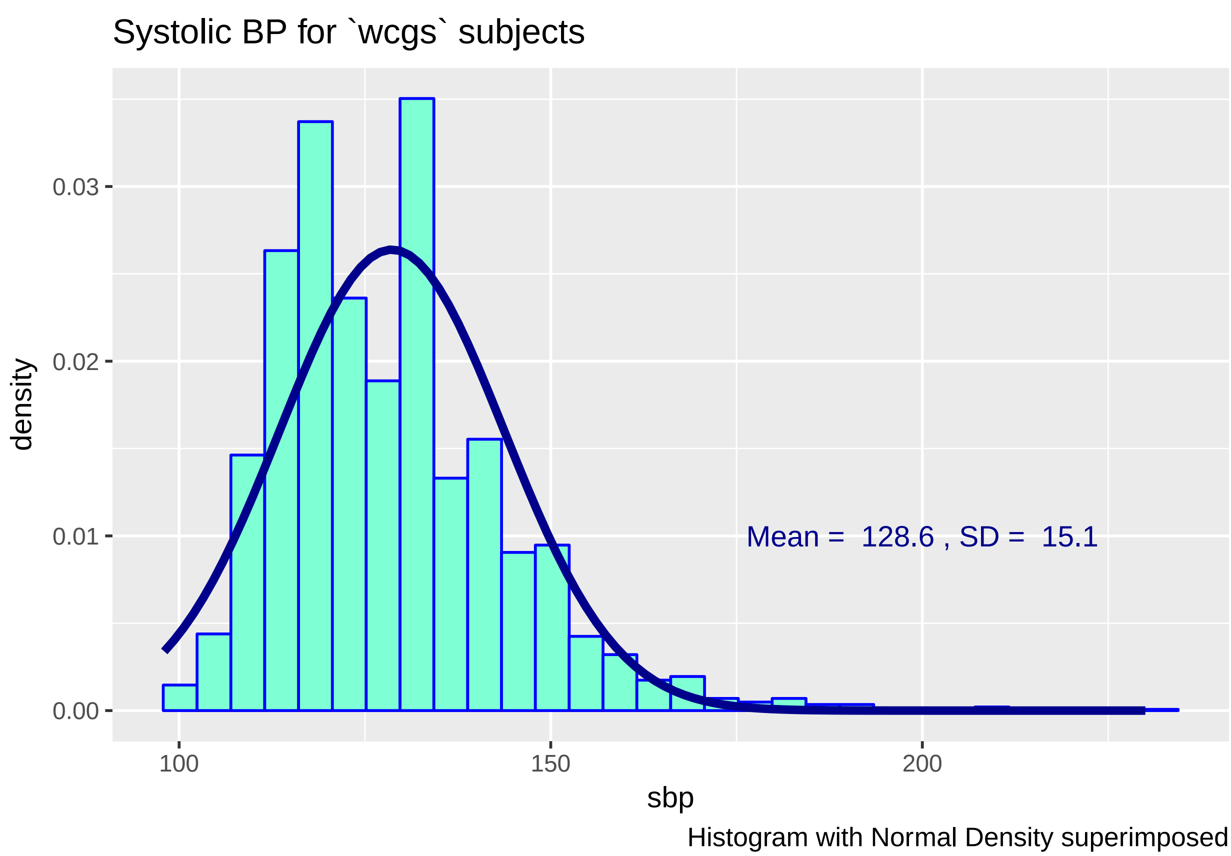
```
       id             age          agec          height         weight   
 Min.   : 2001   Min.   :39.0   35-40: 543   Min.   :60.0   Min.   : 78  
 1st Qu.: 3741   1st Qu.:42.0   41-45:1091   1st Qu.:68.0   1st Qu.:155  
 Median :11406   Median :45.0   46-50: 750   Median :70.0   Median :170  
 Mean   :10478   Mean   :46.3   51-55: 528   Mean   :69.8   Mean   :170  
 3rd Qu.:13115   3rd Qu.:50.0   56-60: 242   3rd Qu.:72.0   3rd Qu.:182  
 Max.   :22101   Max.   :59.0                Max.   :78.0   Max.   :320  

     lnwght        wghtcat          bmi            sbp           lnsbp     
 Min.   :4.36   < 140  : 232   Min.   :11.2   Min.   : 98   Min.   :4.58  
 1st Qu.:5.04   > 200  : 213   1st Qu.:23.0   1st Qu.:120   1st Qu.:4.79  
 Median :5.14   140-170:1538   Median :24.4   Median :126   Median :4.84  
 Mean   :5.13   170-200:1171   Mean   :24.5   Mean   :129   Mean   :4.85  
 3rd Qu.:5.20                  3rd Qu.:25.8   3rd Qu.:136   3rd Qu.:4.91  
 Max.   :5.77                  Max.   :38.9   Max.   :230   Max.   :5.44  

      dbp           chol       behpat      dibpat     smoke     
 Min.   : 58   Min.   :103   A1: 264   Type A:1589   No :1652  
 1st Qu.: 76   1st Qu.:197   A2:1325   Type B:1565   Yes:1502  
 Median : 80   Median :223   B3:1216                            
 Mean   : 82   Mean   :226   B4: 349                            
 3rd Qu.: 86   3rd Qu.:253                                      
 Max.   :150   Max.   :645                                      
               NA's   :12                                       
     ncigs         arcus        chd69        typchd69        time169    
 Min.   : 0.0   Min.   :0.000   No :2897   Min.   :0.000   Min.   :  18  
 1st Qu.: 0.0   1st Qu.:0.000   Yes: 257   1st Qu.:0.000   1st Qu.:2842  
 Median : 0.0   Median :0.000              Median :0.000   Median :2942  
 Mean   :11.6   Mean   :0.299              Mean   :0.136   Mean   :2684  
 3rd Qu.:20.0   3rd Qu.:1.000              3rd Qu.:0.000   3rd Qu.:3037  
 Max.   :99.0   Max.   :1.000              Max.   :3.000   Max.   :3430  
                NA's   :2                                                
       t1             uni       
 Min.   :-47.4   Min.   :0.001  
 1st Qu.: -1.0   1st Qu.:0.257  
 Median :  0.0   Median :0.516  
 Mean   :  0.0   Mean   :0.505  
 3rd Qu.:  1.0   3rd Qu.:0.756  
 Max.   : 47.0   Max.   :0.999  
 NA's   :39                     
```

For a more detailed description, we might consider `Hmisc::describe`, `psych::describe`, `mosaic::favstats`, etc.

13.2 Are the SBPs Normally Distributed?

Consider the question of whether the distribution of the systolic blood pressure results is well-approximated by the Normal.

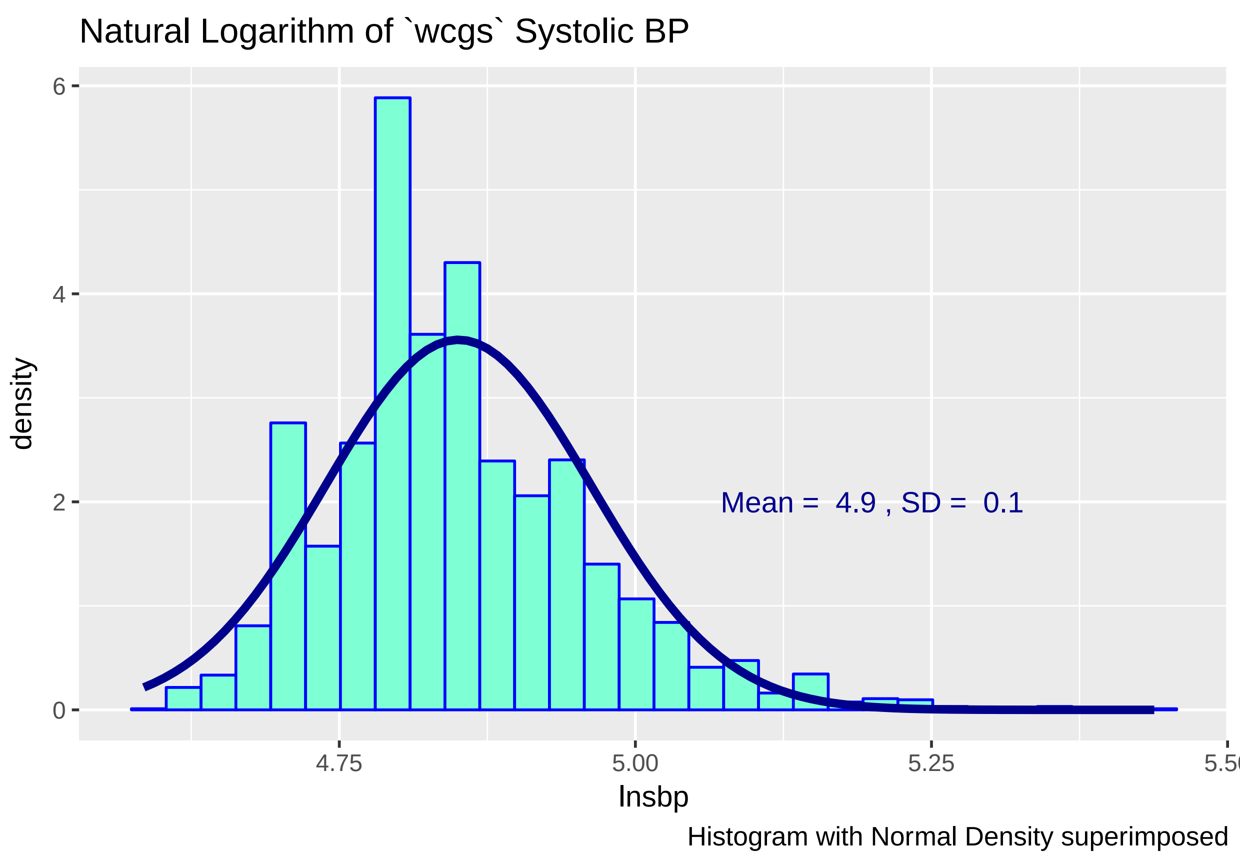
```
ggplot(wcgs, aes(x = sbp)) +
    geom_histogram(aes(y = ..density..),
                   bins = 30, fill = "aquamarine", col="blue") +
    stat_function(fun = dnorm, lwd = 1.5, col = "darkblue",
                  args = list(mean = mean(wcgs$sbp), sd = sd(wcgs$sbp))) +
    annotate("text", x = 200, y = 0.01, col = "darkblue",
             label = paste("Mean = ", round(mean(wcgs$sbp),1),
                           ", SD = ", round(sd(wcgs$sbp),1))) +
    labs(title = "Systolic BP for `wcgs` subjects",
         caption = "Histogram with Normal Density superimposed")
```



Since the data contain both `sbp` and `lnsbp` (its natural logarithm), let's compare them. Note that in preparing the graph, we'll need to change the location for the text annotation.

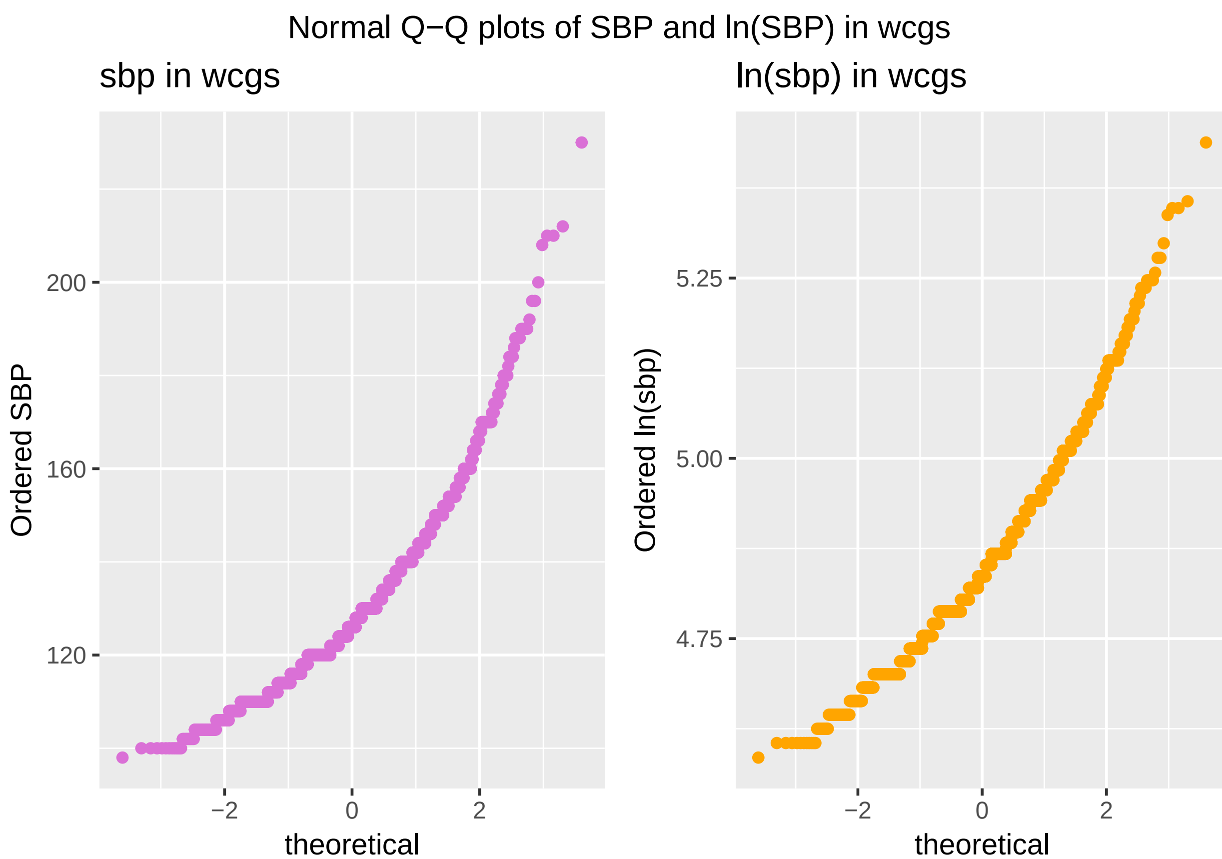
```
ggplot(wcgs, aes(x = lnsbp)) +
    geom_histogram(aes(y = ..density..),
                   bins = 30, fill = "aquamarine", col="blue") +
    stat_function(fun = dnorm, lwd = 1.5, col = "darkblue",
                  args = list(mean = mean(wcgs$lnsbp),
                              sd = sd(wcgs$lnsbp))) +
    annotate("text", x = 5.2, y = 2, col = "darkblue",
             label = paste("Mean = ", round(mean(wcgs$lnsbp),1),
```

```
                    ", SD = ", round(sd(wcgs$lnsbp),1))) +
    labs(title = "Natural Logarithm of `wcgs` Systolic BP",
         caption = "Histogram with Normal Density superimposed")
```



We can also look at Normal Q-Q plots, for instance...

```
p1 <- ggplot(wcgs, aes(sample = sbp)) +
    geom_point(stat="qq", color = "orchid") +
    labs(y = "Ordered SBP", title = "sbp in wcgs")

p2 <- ggplot(wcgs, aes(sample = lnsbp)) +
    geom_point(stat="qq", color = "orange") +
    labs(y = "Ordered ln(sbp)", title = "ln(sbp) in wcgs")

gridExtra::grid.arrange(p1, p2, ncol=2, top ="Normal Q-Q plots of SBP and ln(SBP) in wcgs")
```

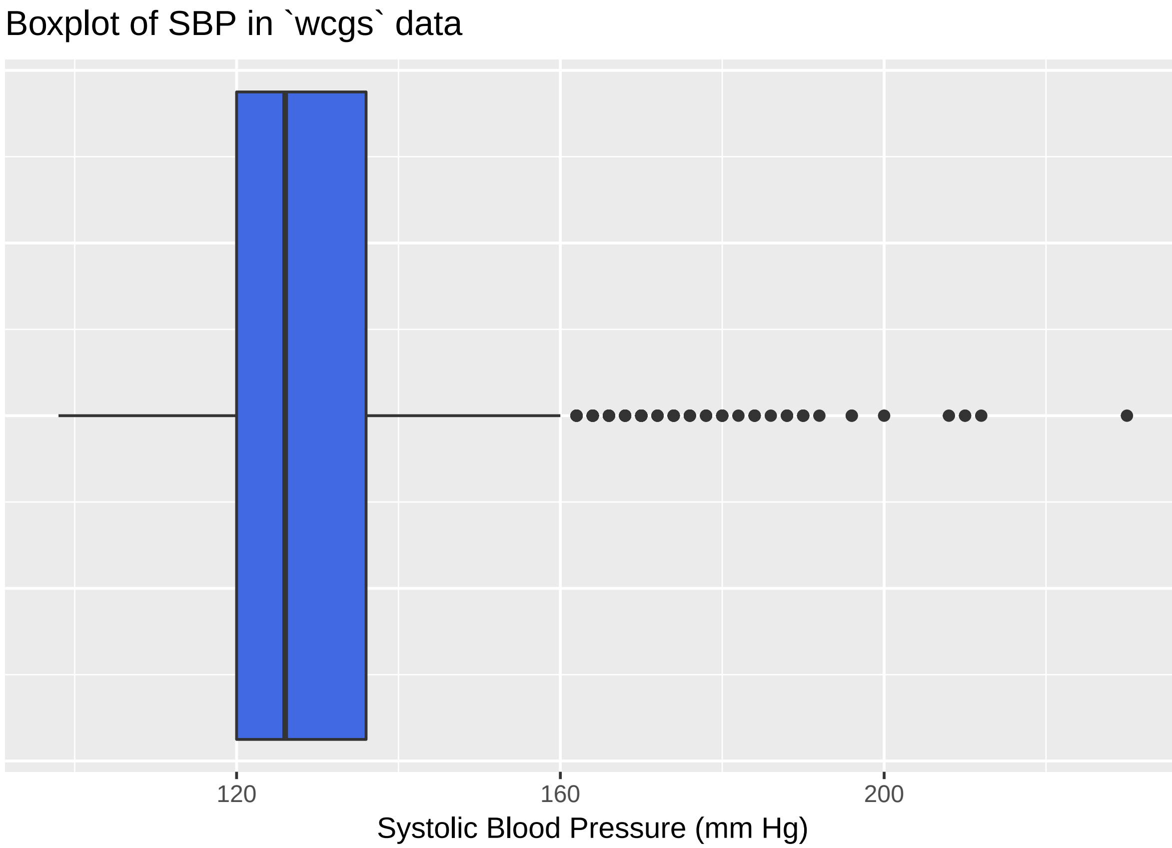


There's at best a small improvement from `sbp` to `lnsbp` in terms of approximation by a Normal distribution.

13.3 Describing Outlying Values with Z Scores

It looks like there's an outlier (or a series of them) in the SBP data.

```
ggplot(wcgs, aes(x = 1, y = sbp)) +
    geom_boxplot(fill = "royalblue") +
    labs(title = "Boxplot of SBP in `wcgs` data",
         y = "Systolic Blood Pressure (mm Hg)",
         x = "") +
    theme(axis.text.y = element_blank(),
          axis.ticks.y = element_blank()) +
    coord_flip()
```




```
Hmisc::describe(wcgs$sbp)
```

```
wcgs$sbp
       n  missing distinct     Info     Mean      Gmd      .05      .10
    3154        0       62    0.996    128.6    16.25      110      112
     .25      .50      .75      .90      .95
     120      126      136      148      156

lowest :  98 100 102 104 106, highest: 200 208 210 212 230
```

The maximum value here is 230, and is clearly the most extreme value in the data set. One way to gauge this is to describe that observation's **Z score**, the number of standard deviations away from the mean that the observation falls. Here, the maximum value, 230 is 6.71 standard deviations above the mean, and thus has a Z score of 6.7.

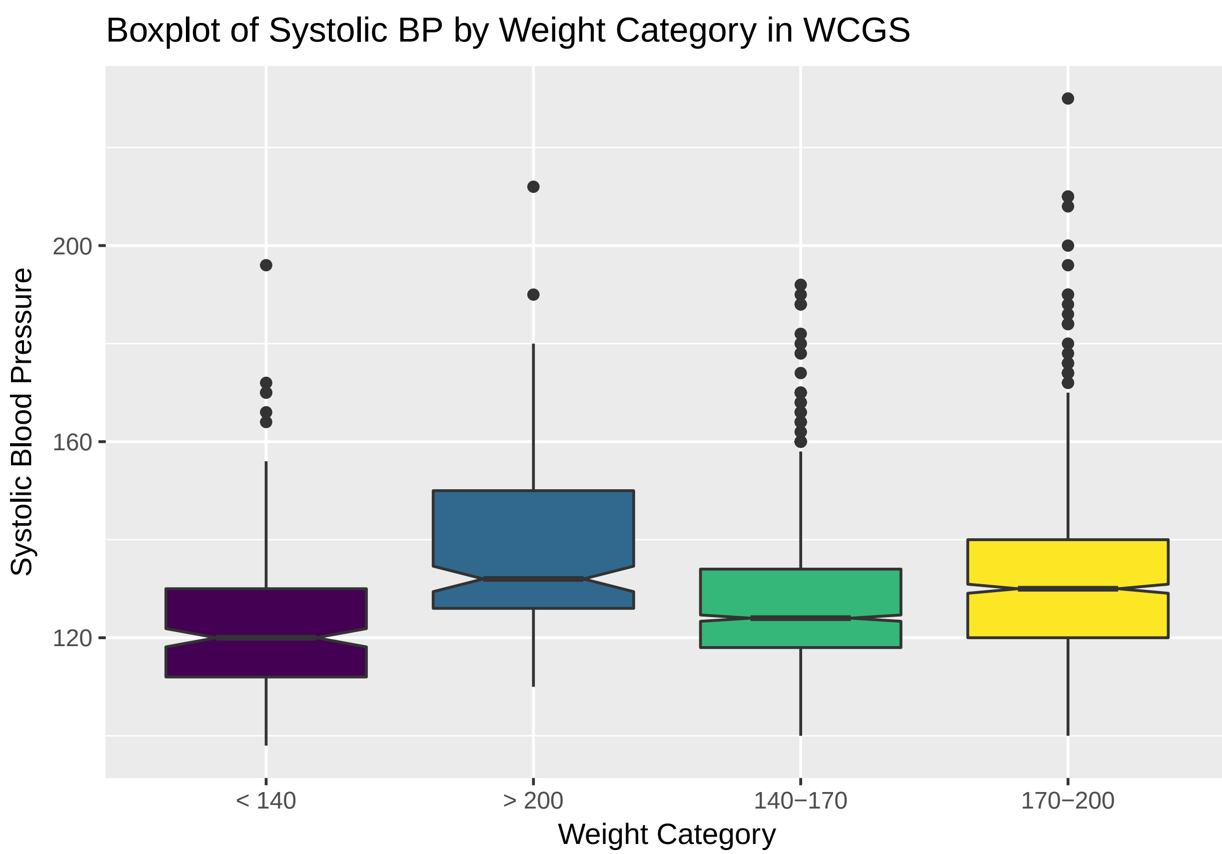
A negative Z score would indicate a point below the mean, while a positive Z score indicates, as we've seen, a point above the mean. The minimum systolic blood pressure, 98 is 2.03 standard deviations *below* the mean, so it has a Z score of -2.

Recall that the Empirical Rule suggests that if a variable follows a Normal distribution, it would have approximately 95% of its observations falling inside a Z score of (-2, 2), and 99.74% falling inside a Z score range of (-3, 3). Do the systolic blood pressures appear Normally distributed?

13.4 Does Weight Category Relate to SBP?

The data are collected into four groups based on the subject's weight (in pounds).

```
ggplot(wcgs, aes(x = wghtcat, y = sbp, fill = wghtcat)) +
    geom_boxplot(notch = TRUE) +
    scale_fill_viridis(discrete=TRUE) +
    guides(fill = FALSE) +
    labs(title = "Boxplot of Systolic BP by Weight Category in WCGS",
         x = "Weight Category", y = "Systolic Blood Pressure")
```



13.5 Re-Leveling a Factor

Well, that's not so good. We really want those weight categories (the *levels*) to be ordered more sensibly.

```
table(wcgs$wghtcat)
```

```

  < 140   > 200 140-170 170-200 
    232     213    1538    1171 
```

Like all *factor* variables in R, the categories are specified as levels.

```
levels(wcgs$wghtcat)
```

```
[1] "< 140"   "> 200"   "140-170" "170-200"
```

We want to change the order of the levels in a new version of this factor variable so they make sense. There are multiple ways to do this, but I prefer the `fct_relevel` function from the `forcats` package. Which order is more appropriate?

```r
table(fct_relevel(wcgs$wghtcat, "< 140", "140-170", "170-200", "> 200"), wcgs$wghtcat)
```

```
          < 140 > 200 140-170 170-200
  < 140     232     0       0       0
  140-170     0     0    1538       0
  170-200     0     0       0    1171
  > 200       0   213       0       0
```

I'll add a new variable to the `wcgs` data called `weight_f` that relevels the `wghtcat` data.

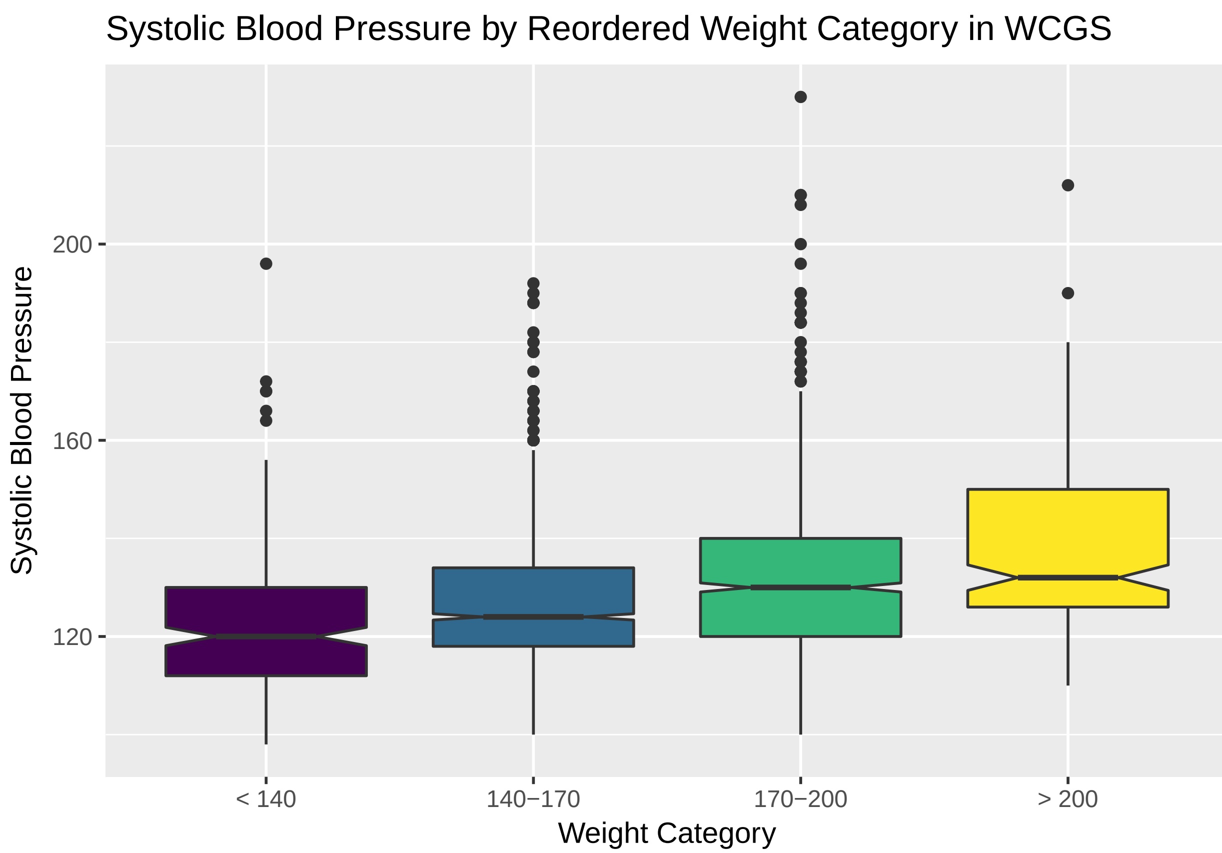
```r
wcgs <- wcgs %>%
    mutate(weight_f = fct_relevel(wghtcat, "< 140", "140-170", "170-200", "> 200"))

table(wcgs$weight_f)
```

```
  < 140 140-170 170-200   > 200 
    232    1538    1171     213 
```

For more on the `forcats` package, check out Grolemund and Wickham (2017), especially the Section on Factors.

13.5.1 SBP by Weight Category

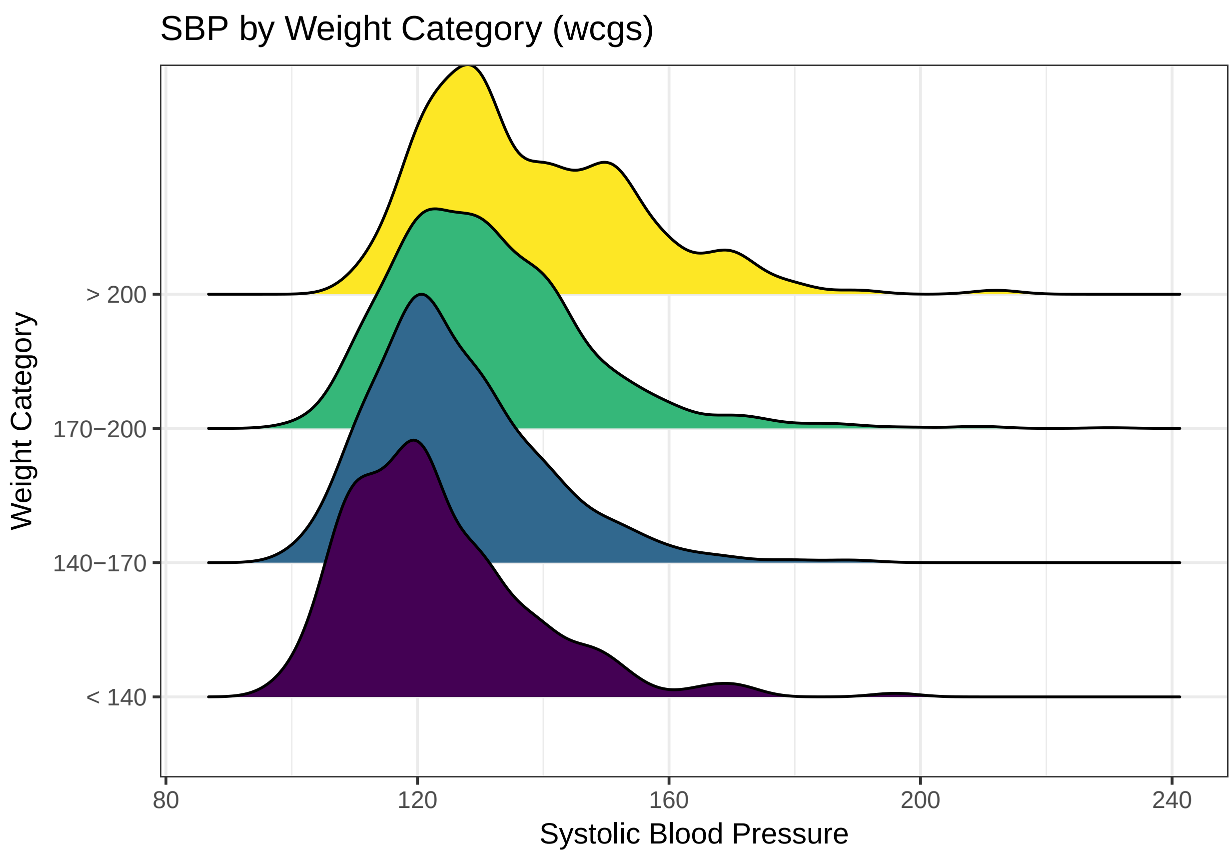
```r
ggplot(wcgs, aes(x = weight_f, y = sbp, fill = weight_f)) +
    geom_boxplot(notch = TRUE) +
    scale_fill_viridis(discrete=TRUE) +
    guides(fill = FALSE) +
    labs(title = "Systolic Blood Pressure by Reordered Weight Category in WCGS",
         x = "Weight Category", y = "Systolic Blood Pressure")
```



We might see some details well with a **ridgeline plot**, too.

```
wcgs %>%
    ggplot(aes(x = sbp, y = weight_f, fill = weight_f, height = ..density..)) +
    ggridges::geom_density_ridges(scale = 2) +
    scale_fill_viridis(discrete = TRUE) +
    guides(fill = FALSE) +
    labs(title = "SBP by Weight Category (wcgs)",
         x = "Systolic Blood Pressure",
         y = "Weight Category") +
    theme_bw()
```

```
Picking joint bandwidth of 3.74
```



As the plots suggest, patients in the heavier groups generally had higher systolic blood pressures.

```
by(wcgs$sbp, wcgs$weight_f, mosaic::favstats)
```

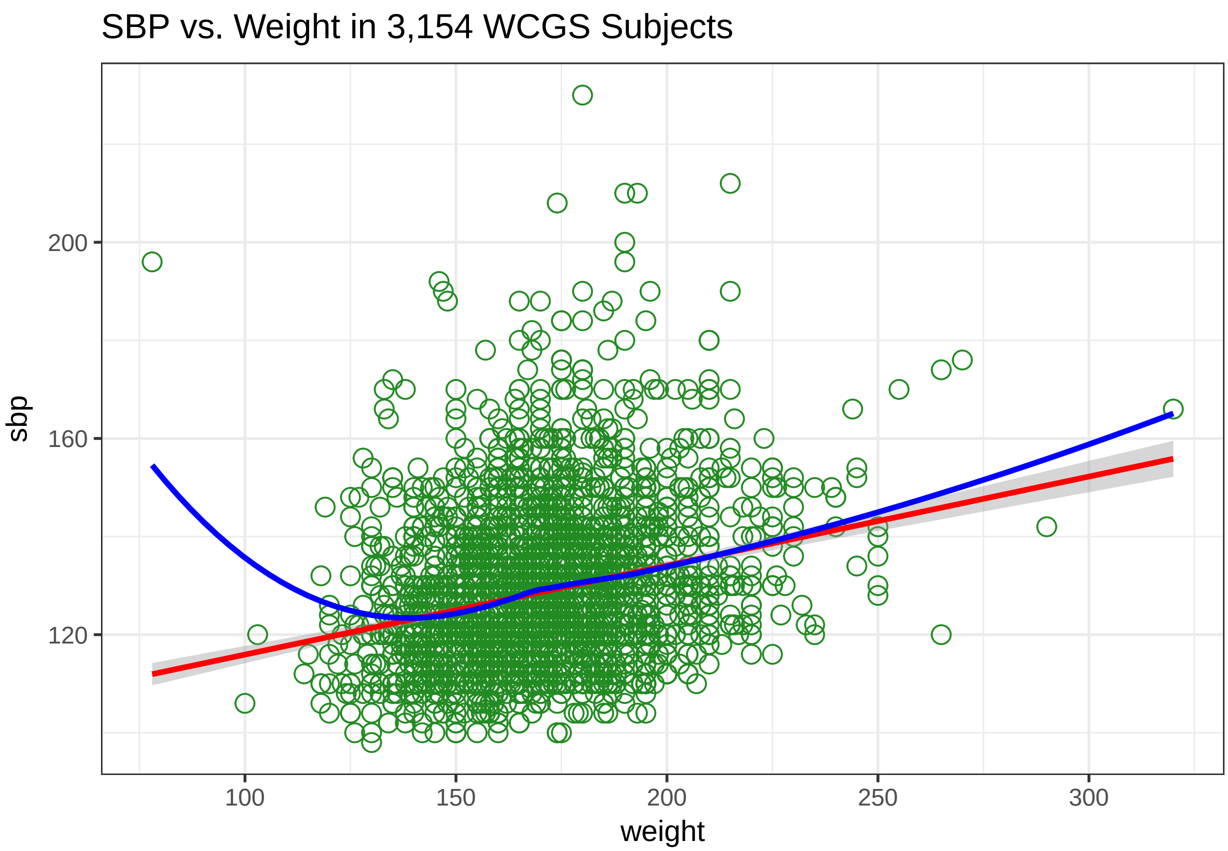
```
wcgs$weight_f: < 140
 min  Q1 median  Q3 max mean   sd   n missing
  98 112    120 130 196  123 14.7 232       0
--------------------------------------------------------
wcgs$weight_f: 140-170
 min  Q1 median  Q3 max mean   sd    n missing
 100 118    124 134 192  126 13.7 1538       0
--------------------------------------------------------
wcgs$weight_f: 170-200
 min  Q1 median  Q3 max mean   sd    n missing
 100 120    130 140 230  131 15.6 1171       0
--------------------------------------------------------
wcgs$weight_f: > 200
 min  Q1 median  Q3 max mean   sd   n missing
 110 126    132 150 212  138 16.8 213       0
```

13.6 Are Weight and SBP Linked?

Let's build a scatter plot of SBP (Outcome) by Weight (Predictor), rather than breaking down into categories.

```
ggplot(wcgs, aes(x = weight, y = sbp)) +
    geom_point(size=3, shape=1, color="forestgreen") + ## default size = 2
```

```
  stat_smooth(method=lm, color="red") + ## add se=FALSE to hide conf. interval
  stat_smooth(method=loess, se=FALSE, color="blue") +
  ggtitle("SBP vs. Weight in 3,154 WCGS Subjects") +
  theme_bw()
```



- The mass of the data is hidden from us - showing 3154 points in one plot can produce little more than a blur where there are lots of points on top of each other.
- Here the least squares regression line (in red), and loess scatterplot smoother, (in blue) can help.

The relationship between systolic blood pressure and weight appears to be very close to linear, but of course there is considerable scatter around that generally linear relationship. It turns out that the Pearson correlation of these two variables is 0.253.

13.7 SBP and Weight by Arcus Senilis groups?

An issue of interest to us will be to assess whether the SBP-Weight relationship we see above is similar among subjects who have arcus senilis and those who do not.

> Arcus senilis is an old age syndrome where there is a white, grey, or blue opaque ring in the corneal margin (peripheral corneal opacity), or white ring in front of the periphery of the iris. It is present at birth but then fades; however, it is quite commonly present in the elderly. It can also appear earlier in life as a result of hypercholesterolemia.
>
> Wikipedia article on Arcus Senilis, retrieved 2017-08-15

Let's start with a quick look at the `arcus` data.

```
wcgs %>%
    select(arcus) %>%
    summary()
```

```
     arcus
 Min.   :0.000
 1st Qu.:0.000
 Median :0.000
 Mean   :0.299
 3rd Qu.:1.000
 Max.   :1.000
 NA's   :2
```

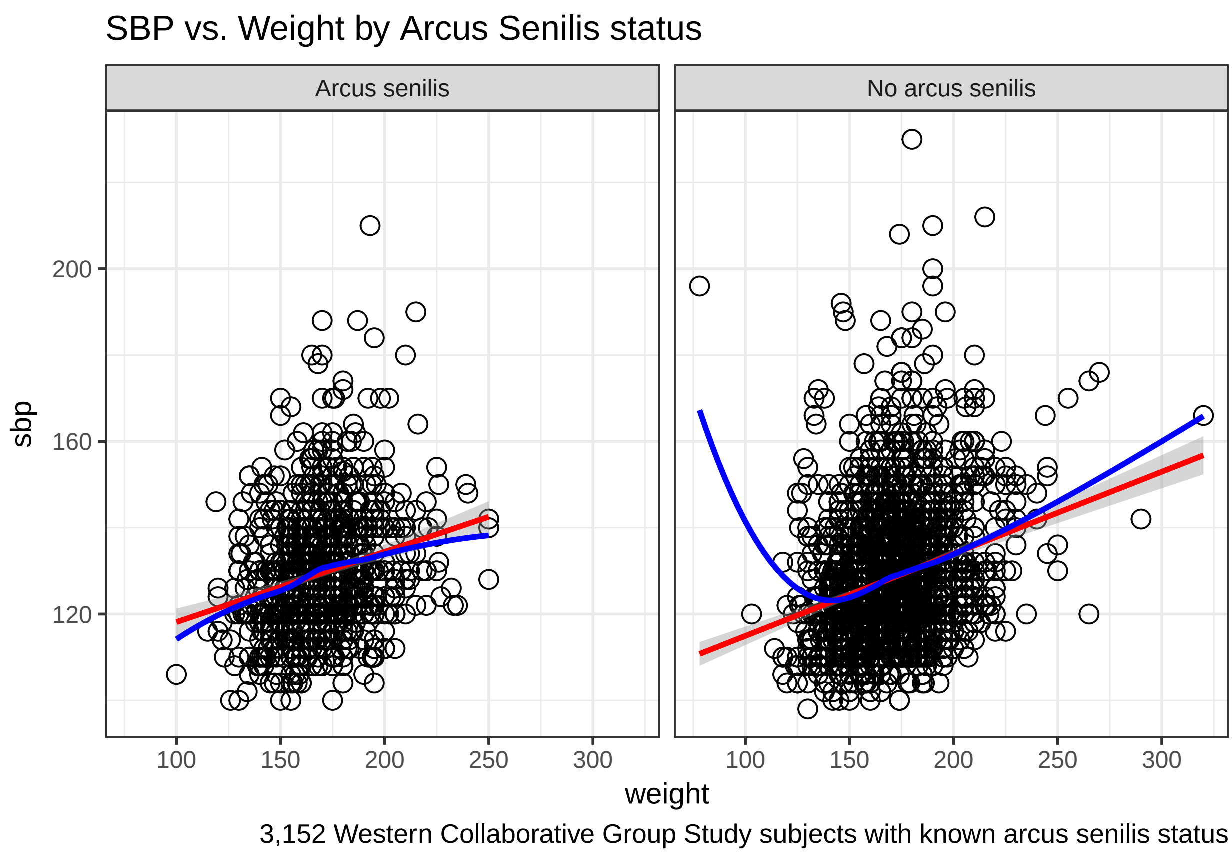
We have 2 missing values, so we probably want to do something about that before plotting the data, and we may also want to create a factor variable with more meaningful labels than 1 (which means yes, arcus senilis is present) and 0 (which means no, it isn't.) We'll use the

```
wcgs <- wcgs %>%
    mutate(arcus_f = fct_recode(factor(arcus),
                                "Arcus senilis" = "1",
                                "No arcus senilis" = "0"),
           arcus_f = fct_relevel(arcus_f, "Arcus senilis"))

table(wcgs$arcus_f, wcgs$arcus, useNA = "ifany")
```

```
                      0    1 <NA>
  Arcus senilis       0  941    0
  No arcus senilis 2211    0    0
  <NA>                0    0    2
```

Let's build a version of the `wcgs` data that eliminates all missing data in the variables of immediate interest, and then plot the SBP-weight relationship in groups of patients with and without arcus senilis.

```
wcgs %>%
    filter(complete.cases(arcus_f, sbp, weight)) %>%
    ggplot(aes(x = weight, y = sbp, group = arcus_f)) +
    geom_point(size=3, shape = 1) +
    stat_smooth(method=lm, color="red") +
    stat_smooth(method=loess, se=FALSE, color="blue") +
    labs(title = "SBP vs. Weight by Arcus Senilis status",
         caption = "3,152 Western Collaborative Group Study subjects with known arcus senilis status") +
    facet_wrap(~ arcus_f) +
    theme_bw()
```



13.8 Linear Model for SBP-Weight Relationship: subjects without Arcus Senilis

```
model.noarcus <-
    lm(sbp ~ weight, data = filter(wcgs, arcus == 0))

summary(model.noarcus)
```

```

Call:
lm(formula = sbp ~ weight, data = filter(wcgs, arcus == 0))

Residuals:
   Min     1Q Median     3Q    Max 
-29.01 -10.25  -2.45   7.55  99.85 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  95.9219     2.5552    37.5   <2e-16 ***
weight        0.1902     0.0149    12.8   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 14.8 on 2209 degrees of freedom
Multiple R-squared:  0.0687,    Adjusted R-squared:  0.0683 
F-statistic:  163 on 1 and 2209 DF,  p-value: <2e-16
```

The linear model for the 2211 patients without Arcus Senilis has $R^2 = 6.87\%$.

- The regression equation is 95.92 - 0.19 weight, for those patients without Arcus Senilis.

13.9 Linear Model for SBP-Weight Relationship: subjects with Arcus Senilis

```r
model.witharcus <- 
    lm(sbp ~ weight, data = filter(wcgs, arcus == 1))

summary(model.witharcus)
```

```

Call:
lm(formula = sbp ~ weight, data = filter(wcgs, arcus == 1))

Residuals:
   Min     1Q Median     3Q    Max 
-30.34  -9.64  -1.96   7.97  76.74 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  101.879      3.756   27.13  < 2e-16 ***
weight         0.163      0.022    7.39  3.3e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 14.2 on 939 degrees of freedom
Multiple R-squared:  0.0549,    Adjusted R-squared:  0.0539 
F-statistic: 54.6 on 1 and 939 DF,  p-value: 3.29e-13
```

The linear model for the 941 patients with Arcus Senilis has $R^2 = 5.49\%$.

- The regression equation is 101.88 - 0.163 weight, for those patients with Arcus Senilis.

13.10 Including Arcus Status in the model

```r
model3 <- lm(sbp ~ weight * arcus, data = filter(wcgs, !is.na(arcus)))

summary(model3)
```

```

Call:
lm(formula = sbp ~ weight * arcus, data = filter(wcgs, !is.na(arcus)))

Residuals:
   Min     1Q Median     3Q    Max 
-30.34 -10.15  -2.35   7.67  99.85 
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    95.9219     2.5244   38.00   <2e-16 ***
weight          0.1902     0.0147   12.92   <2e-16 ***
arcus           5.9566     4.6197    1.29     0.20
weight:arcus   -0.0276     0.0270   -1.02     0.31
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 14.6 on 3148 degrees of freedom
Multiple R-squared:  0.066, Adjusted R-squared:  0.0651
F-statistic: 74.1 on 3 and 3148 DF,  p-value: <2e-16
```

The actual regression equation in this setting includes both weight, and an indicator variable (1 = yes, 0 = no) for arcus senilis status, and the product of weight and that 1/0 indicator.

- Note the use of the product term `weight*arcus` in the setup of the model to allow both the slope of weight and the intercept term in the model to change depending on arcus senilis status.
  - For a patient who has arcus, the regression equation is SBP = 95.92 - 0.19 weight + 5.96 (1) - 0.028 weight (1) = 101.88 + 0.162 weight.
  - For a patient without arcus senilis, the regression equation is SBP = 95.92 - 0.19 weight + 5.96 (0) - 0.028 weight (0) = 95.92 - 0.19 weight.

The linear model including the interaction of weight and arcus to predict sbp for the 3152 patients with known Arcus Senilis status has R^2 = 6.6%.

13.11 Predictions from these Linear Models

What is our predicted SBP for a subject weighing 175 pounds?

How does that change if our subject weighs 200 pounds?

Recall that

- *Without* Arcus Senilis, linear model for SBP = 95.9 + 0.19 x weight
- *With* Arcus Senilis, linear model for SBP = 101.9 + 0.16 x weight

So the predictions for a 175 pound subject are: - 95.9 + 0.19 x 175 = 129 mm Hg without Arcus Senilis, and - 101.9 + 0.16 x 175 = 130 mm Hg with Arcus Senilis.

And thus, the predictions for a 200 pound subject are: - 95.9 + 0.19 x 200 = 134 mm Hg without Arcus Senilis, and - 101.9 + 0.16 x 200 = 134.4 mm Hg with Arcus Senilis.

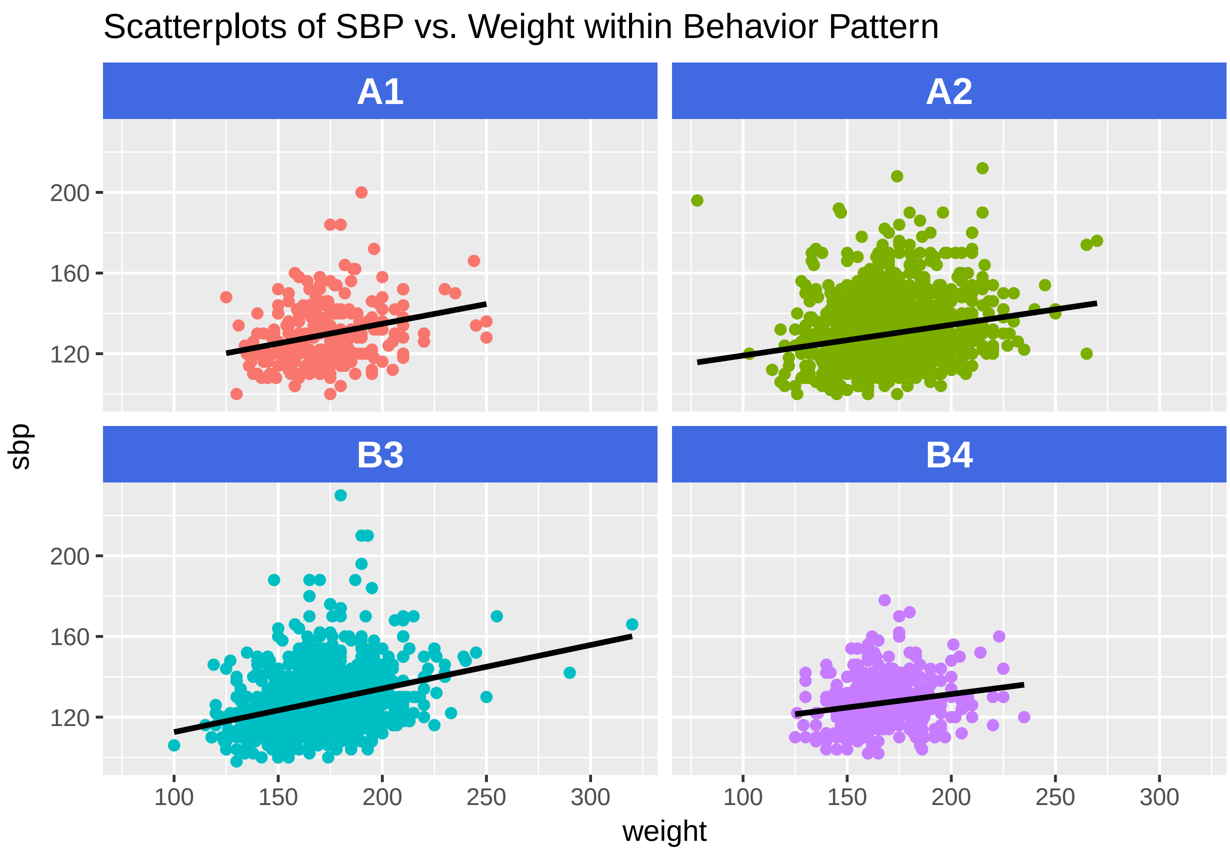
```r
rm(model.noarcus, model.witharcus)
```

13.12 Scatterplots with Facets Across a Categorical Variable

We can use facets in `ggplot2` to show scatterplots across the levels of a categorical variable, like `behpat`.

```r
ggplot(wcgs, aes(x = weight, y = sbp, col = behpat)) +
    geom_point() +
    facet_wrap(~ behpat) +
    geom_smooth(method = "lm", se = FALSE, col = "black") +
    guides(color = FALSE) +
```

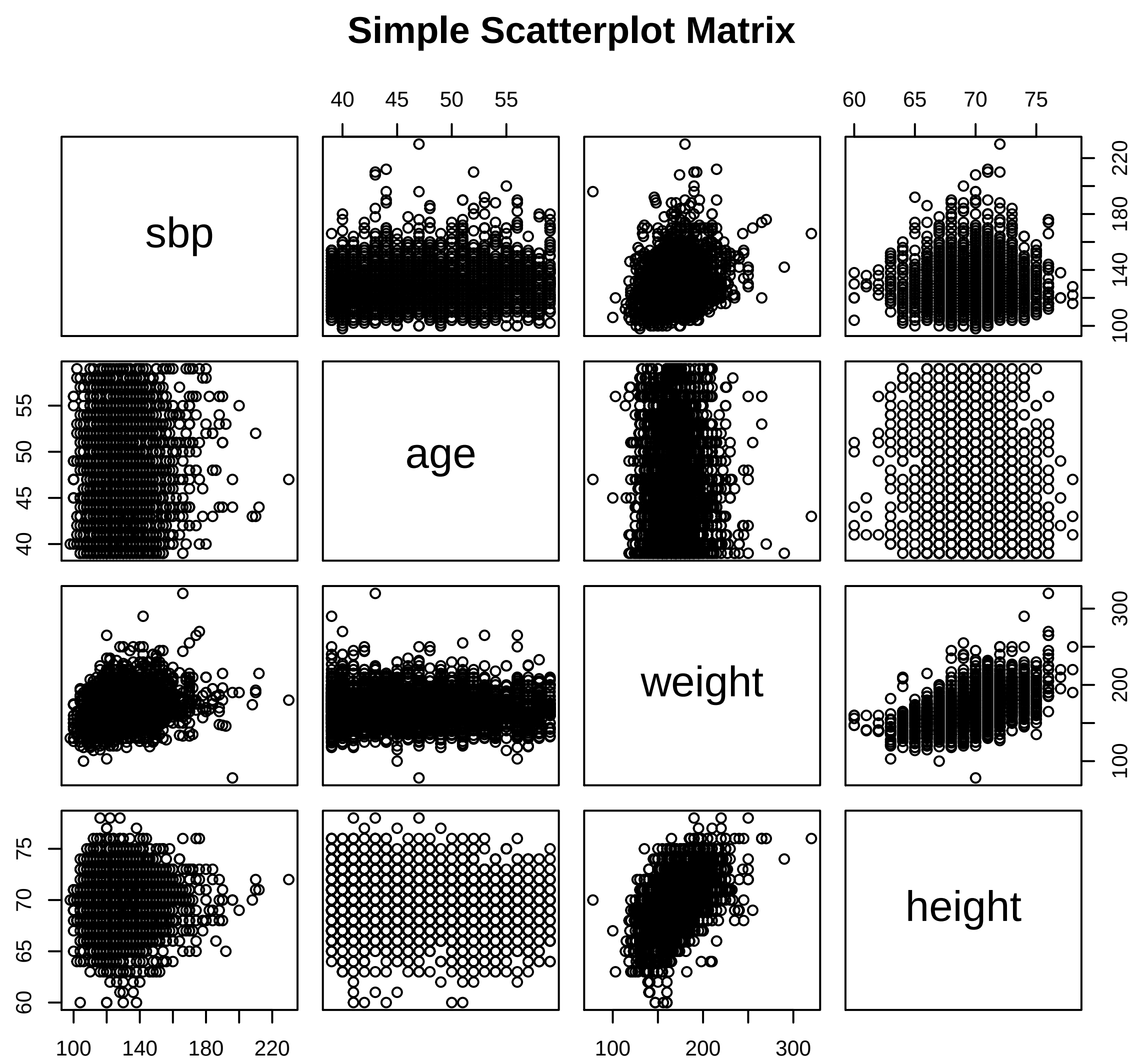
```
    theme(strip.text = element_text(face="bold", size=rel(1.25), color="white"),
          strip.background = element_rect(fill="royalblue")) +
    labs(title = "Scatterplots of SBP vs. Weight within Behavior Pattern")
```



13.13 Scatterplot and Correlation Matrices

A **scatterplot matrix** can be very helpful in understanding relationships between multiple variables simultaneously. There are several ways to build such a thing, including the `pairs` function...

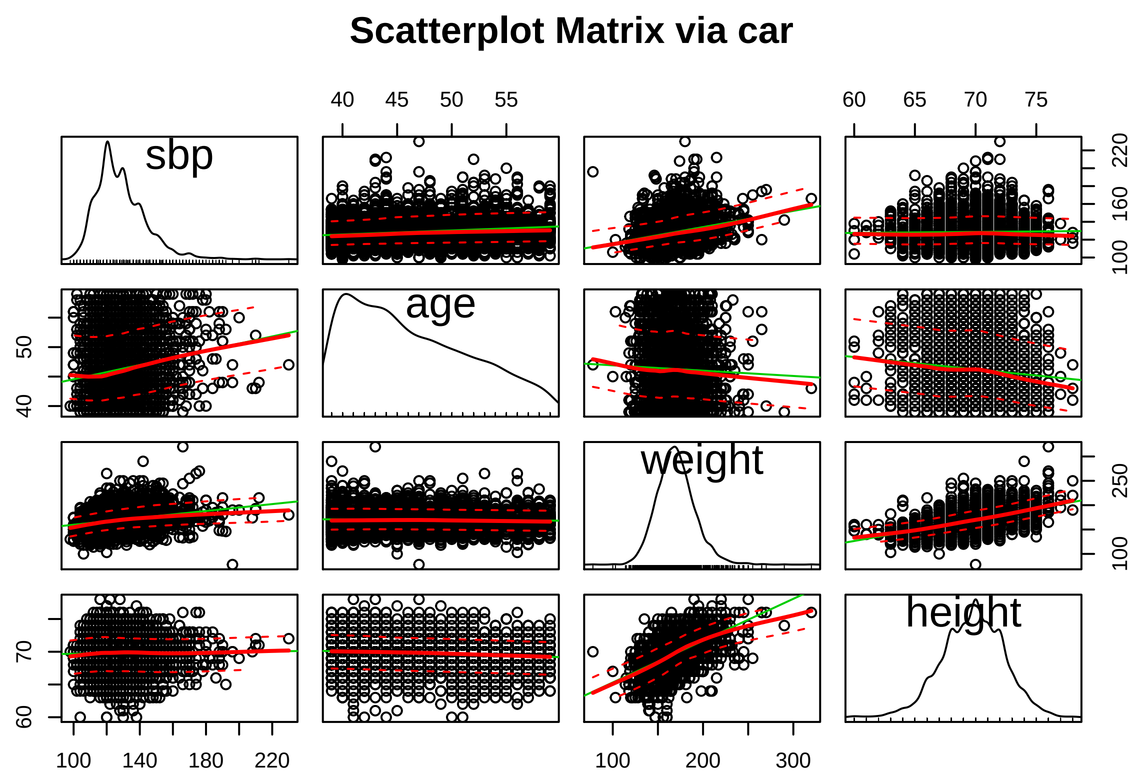
```
pairs (~ sbp + age + weight + height, data=wcgs, main="Simple Scatterplot Matrix")
```



13.13.1 Using the car package

Or, we can use the `scatterplotMatrix` function from the `car` package, which adds some detail and fitting to the plots, and places density estimates (with rug plots) on the diagonals.

```
car::scatterplotMatrix(~ sbp + age + weight + height,
                       data=wcgs,
                       main="Scatterplot Matrix via car")
```



13.13.2 Displaying a Correlation Matrix

```
wcgs %>%
    dplyr::select(sbp, age, weight, height) %>%
    cor() %>% # obtain correlation coefficients for this subgroup
    signif(., 3) # round them off to three significant figures before printing
```

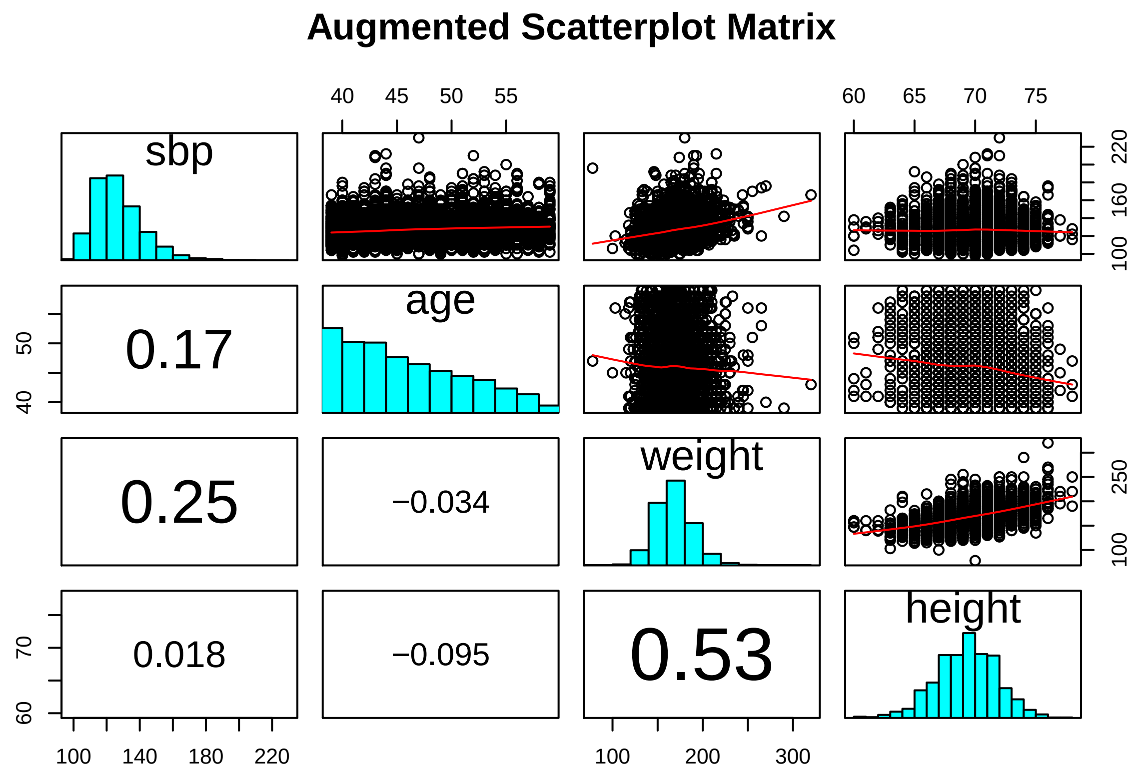
```
          sbp     age  weight  height
sbp    1.0000  0.1660  0.2530  0.0184
age    0.1660  1.0000 -0.0344 -0.0954
weight 0.2530 -0.0344  1.0000  0.5330
height 0.0184 -0.0954  0.5330  1.0000
```

13.13.3 Augmented Scatterplot Matrix

Dr. Love's favorite way to augment a scatterplot matrix adds LOWESS smoothed lines in the upper panel, and correlations in the lower panel, with histograms down the diagonal. To do this, I revised two functions in the Love-boost script (these modifications come from Chang's R Graphics Cookbook), called `panel.hist` and `panel.cor`.

```
# requires Love-boost.R is sourced

pairs (~ sbp + age + weight + height, data=wcgs,
       main="Augmented Scatterplot Matrix",
```

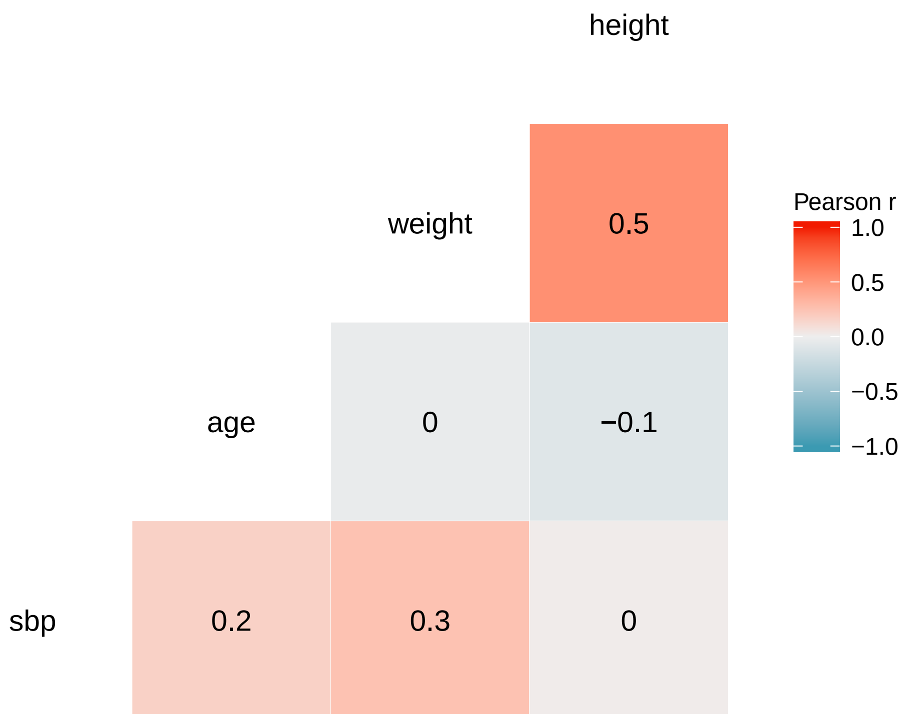
```
      upper.panel = panel.smooth,
      diag.panel = panel.hist,
      lower.panel = panel.cor)
```



13.13.4 Using the `GGally` package

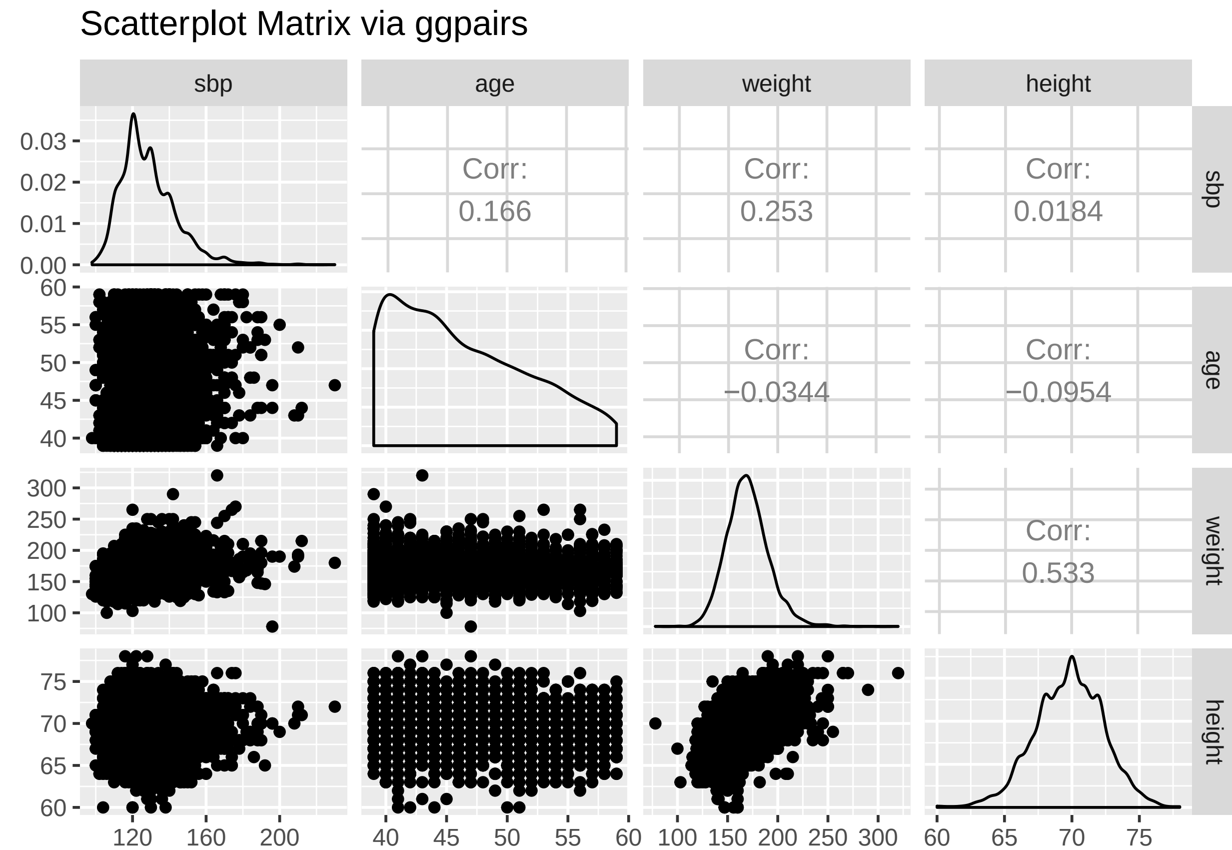
The `ggplot2` system doesn't have a built-in scatterplot system. There are some nice add-ins in the world, though. One option I sort of like is in the `GGally` package, which can produce both correlation matrices and scatterplot matrices.

```
GGally::ggcorr(select(wcgs, sbp, age, weight, height),
               name = "Pearson r", label = TRUE)
```



The `ggpairs` function provides a density plot on each diagonal, Pearson correlations on the upper right and scatterplots on the lower left of the matrix.

```
GGally::ggpairs(select(wcgs, sbp, age, weight, height),
                title = "Scatterplot Matrix via ggpairs")
```


Scatterplot Matrix via ggpairs
sbp
age
weight
height
Corr:
0.166
Corr:
0.253
Corr:
0.0184
Corr:
−0.0344
Corr:
−0.0954
Corr:
0.533
0.00
0.01
0.02
0.03
40
45
50
55
60
100
150
200
250
300
60
65
70
75
120
160
200
40
45
50
55
60
100
150
200
250
300
60
65
70
75

Chapter 14

Part A: A Few of the Key Points

14.1 Key Graphical Descriptive Summaries for Quantitative Data

- **Histograms** and their variants, including smooth density curves, and normal density functions based on the sample mean and sample standard deviation
- **Boxplots** and the like, including ridgeline plots and violin plots, that show more of the distribution in a compact format that is especially useful for comparisons
- **Normal QQ Plots** which are plots of the ordered data (technically, the order statistics) against certain quantiles of the Normal distribution - show curves to indicate skew, and "S" shaped arcs to indicate seriously heavy- or light-tailed distributions compared to the Normal.

14.2 Key Numerical Descriptive Summaries for Quantitative Data

- Measures of *Location* (including Central Tendency), such as the **mean**, **median**, **quantiles** and even the **mode**.
- Measures of *Spread*, including the **range**, **IQR** (which measures the variability of points near the center of the distribution), **standard deviation** (which is less appropriate as a summary measure if the data show substantial skew or heavy-tailedness), **variance**, **standard error**, **median absolute deviation** (which is less affected by outlying values in the tails of the distribution than a standard deviation).
- I'll mention the **coefficient of variation** (ratio of the standard deviation to the mean, expressed as a percentage, note that this is only appropriate for variables that take only positive values.)
- One Key Measure of *Shape* is nonparametric skew (skew_1), which can be used to help confirm plot-based decisions about data shape.

14.3 The Empirical Rule - Interpreting a Standard Deviation

If the data are approximately Normally distributed, then the mean and median will be very similar, and there will be minimal skew and no large outlier problem.

Should this be the case, the mean and standard deviation describe the distribution well, and the **Empirical Rule** will hold reasonably well.

If the data are (approximately) Normally distributed, then

- About 68% of the data will fall within one standard deviation of the mean
- Approximately 95% of the data will fall within two standard deviations of the mean

- Approximately 99.7% of the data will fall within three standard deviations of the mean.

14.4 Identifying "Outliers" Using Fences and/or Z Scores

- Distributions can be symmetric, but still not Normally distributed, if they are either outlier-prone (heavy-tailed) or light-tailed.
- Outliers can have an important impact on other descriptive measures.
- John Tukey described **fences** which separated non-outlier from outlier values in a distribution. Generally, the fences are set 1.5 IQR away from the 25th and 75th percentiles in a boxplot.
- Or, we can use **Z scores** to highlight the relationship between values and what we might expect if the data were normally distributed.
- The Z score for an individual value is that value minus the data's mean, all divided by the data's standard deviation.
- If the data are normally distributed, we'd expect all but 5% of its observations to have Z scores between -2 and +2, for example.

14.5 Summarizing Bivariate Associations: Scatterplots and Regression Lines

- The most important tools are various **scatterplots**, often accompanied by **regression lines** estimated by the method of least squares, and by (loess) **smooths** which permit local polynomial functions to display curved relationships.
- In a multivariate setting, we will occasionally consider plots in the form of a **scatterplot matrix** to enable simultaneous comparisons of multiple two-way associations.
- We fit linear models to our data using the `lm` function, and we evaluate the models in terms of their ability to predict an outcome given a predictor, and through R^2, which is interpreted as the proportion of variation in the outcome accounted for by the model.

14.6 Summarizing Bivariate Associations With Correlations

- **Correlation coefficients,** of which by far the most commonly used is the **Pearson correlation**, which is a unitless (scale-free) measure of bivariate linear association for the variables X and Y, symbolized by r, and ranging from -1 to +1. The Pearson correlation is a function of the slope of the least squares regression line, divided by the product of the standard deviations of X and Y.
- Also relevant to us is the **Spearman rank correlation coefficient**, which is obtained by using the usual formula for a Pearson correlation, but on the ranks (1 = minimum, n = maximum, with average ranks are applied to the ties) of the X and Y values. This approach (running a correlation of the orderings of the data) substantially reduces the effect of outliers. The result still ranges from -1 to +1, with 0 indicating no monotone association.

Part B. Making Comparisons

Chapter 15

Introduction to Part B

15.1 Point Estimation and Confidence Intervals

The basic theory of estimation can be used to indicate the probable accuracy and potential for bias in estimating based on limited samples. A point estimate provides a single best guess as to the value of a population or process parameter.

A confidence interval is a particularly useful way to convey to people just how much error one must allow for in a given estimate. In particular, a confidence interval allows us to quantify just how close we expect, for instance, the sample mean to be to the population or process mean. The computer will do the calculations; we need to interpret the results.

The key tradeoffs are cost vs. precision, and precision vs. confidence in the correctness of the statement. Often, if we are dissatisfied with the width of the confidence interval and want to make it smaller, we have little choice but to reconsider the sample – larger samples produce shorter intervals.

15.2 One-Sample Confidence Intervals and Hypothesis Testing

Very often, sample data indicate that something has happened – a change in the proportion, a shift in the mean, etc. Before we get excited, it's worth checking whether the apparent result might possibly be the result of random sampling error. The next few classes will be devoted to ideas of testing–seeing whether an apparent result might possibly be attributable to sheer randomness. Confidence intervals provide a way to assess this chance.

Statistics provides a number of tools for reaching an informed choice (informed by sample information, of course.) Which tool, or statistical method, to use depends on various aspects of the problem at hand. In addition, a p value, (often part of a computer output) gives an index of how much evidence we have that an apparent result is more than random.

15.3 Comparing Two Groups

In making a choice between two alternatives, questions such as the following become paramount.

- Is there a status quo?
- Is there a standard approach?
- What are the costs of incorrect decisions?
- Are such costs balanced?

The process of comparing the means/medians/proportions/rates of the populations represented by two independently obtained samples can be challenging, and such an approach is not always the best choice. Often, specially designed experiments can be more informative at lower cost (i.e. smaller sample size). As one might expect, using these more sophisticated procedures introduces trade-offs, but the costs are typically small relative to the gain in information.

When faced with such a comparison of two alternatives, a test based on **paired** data is often much better than a test based on two distinct, independent samples. Why? If we have done our experiment properly, the pairing lets us eliminate background variation that otherwise hides meaningful differences.

15.3.1 Model-Based Comparisons and ANOVA/Regression

Comparisons based on independent samples of quantitative variables are also frequently accomplished through other equivalent methods, including the analysis of variance approach and dummy variable regression, both of which produce the identical p values and confidence intervals to the pooled variance t test for the same comparison.

We will also discuss some of the main ideas in developing, designing and analyzing statistical experiments, specifically in terms of making comparisons. The ideas we will present in this section allow for the comparison of more than two populations in terms of their population means. The statistical techniques employed analyze the sample variance in order to test and estimate the population means and for this reason the method is called the analysis of variance (ANOVA), and we will discuss this approach alone, and within the context of a linear regression model using dummy or indicator variables.

15.4 Special Tools for Categorical Data

We will also turn briefly to some methods for dealing with qualitative, categorical variables. In particular, we begin with a test of how well the frequencies of various categories fit a theoretical set of probabilities. We also consider a test for the relation between two qualitative variables. We'll examine some of the key measures used in describing such relationships, like odds ratios and relative risks.

15.5 Our First Three Studies

We'll focus, for a while, on three studies, and the next three Sections of these Notes summarize each of them, graphically and numerically.

- The Serum Zinc study, which uses a single sample of quantitative data.
- The Lead in the Blood of Children study, which uses a *paired samples* design to compare two samples of quantitative data.
- A randomized controlled trial comparing ibuprofen vs. placebo in patients with sepsis, which uses an *independent samples* design to compare two samples of quantitative data.

15.6 Data Sets used in Part B

```
serzinc <- read_csv("data/serzinc.csv")
bloodlead <- read_csv("data/bloodlead.csv")
sepsis <- read_csv("data/sepsis.csv")
```

```
schiztwin <- read.csv("data/schiztwin.csv") %>% tbl_df
hers <- read.csv("data/hers.csv") %>% tbl_df
cotinine <- read.csv("data/cotinine.csv") %>% tbl_df
battery <- read.csv("data/battery.csv") %>% tbl_df
survey1 <- read.csv("data/surveyday1.csv") %>% tbl_df
active2x3 <- read.csv("data/active2x3.csv") # deliberately NOT a tibble
breakfast <- read.csv("data/breakfast.csv") %>% tbl_df
hairloss <- read.csv("data/hairloss.csv") %>% tbl_df
darwin <- read.csv("data/darwin.csv") %>% tbl_df
```

We'll also continue to make use of the Love-boost.R script of functions loaded in Section 2.

Chapter 16

The Serum Zinc Study

16.1 Serum Zinc Levels in 462 Teenage Males (`serzinc`)

The `serzinc` data include serum zinc levels in micrograms per deciliter that have been gathered for a sample of 462 males aged 15-17, My source for these data is Appendix B1 of Pagano and Gauvreau (2000). Serum zinc deficiency has been associated with anemia, loss of strength and endurance, and it is thought that 25% of the world's population is at risk of zinc deficiency. Such a deficiency can indicate poor nutrition, and can affect growth and vision, for instance. "Typical" values[1] are said to be 0.66-1.10 mcg/ml, which is 66 - 110 micrograms per deciliter.

```
serzinc
```

```
# A tibble: 462 x 2
      ID  zinc
   <chr> <int>
 1 M-001   142
 2 M-002    88
 3 M-003    83
 4 M-004   100
 5 M-005   123
 6 M-006    63
 7 M-007   102
 8 M-008    80
 9 M-009   117
10 M-010    86
# ... with 452 more rows
```

16.2 Our Goal: A Confidence Interval for the Population Mean

After we assess the data a bit, and are satisfied that we understand it, our first inferential goal will be to produce a **confidence interval for the true (population) mean** of males age 15-17 based on this sample, assuming that these 462 males are a random sample from the population of interest, that each serum zinc level is drawn independently from an identical distribution describing that population.

To do this, we will have several different procedures available, including:

[1] Reference values for those over the age of 10 years at http://www.mayomedicallaboratories.com/test-catalog/Clinical+and+Interpretive/8620 , visited 2017-08-17.

1. A confidence interval for the population mean based on a t distribution, when we assume that the data are drawn from an approximately Normal distribution, using the sample standard deviation. (Interval corresponding to a t test, and it will be a good choice when the data really are approximately Normally distributed.)
2. A resampling approach to generate a bootstrap confidence interval for the population mean, which does not require that we assume either that the population standard deviation is known, nor that the data are drawn from an approximately Normal distribution, but which has some other weaknesses.
3. A rank-based procedure called the Wilcoxon signed rank test can also be used to yield a confidence interval statement about the population pseudo-median, a measure of the population distribution's center (but not the population's mean).

16.3 Exploratory Data Analysis for Serum Zinc

16.3.1 Comparison to "Normal" Zinc Levels

Recall that the "Normal" zinc level would be between 66 and 110. What percentage of the sampled 462 teenagers meet that standard?

```
serzinc %>%
    count(zinc > 65 & zinc < 111) %>%
    mutate(proportion = n / sum(n), percentage = 100 * n / sum(n))
```

```
# A tibble: 2 x 4
  `zinc > 65 & zinc < 111`     n proportion percentage
                     <lgl> <int>      <dbl>      <dbl>
1                    FALSE    67      0.145       14.5
2                     TRUE   395      0.855       85.5
```

16.3.2 Graphical Summaries

The code presented below builds:

- a histogram (with Normal model superimposed),
- a boxplot (with median notch) and
- a Normal Q-Q plot (with guiding straight line through the quartiles)

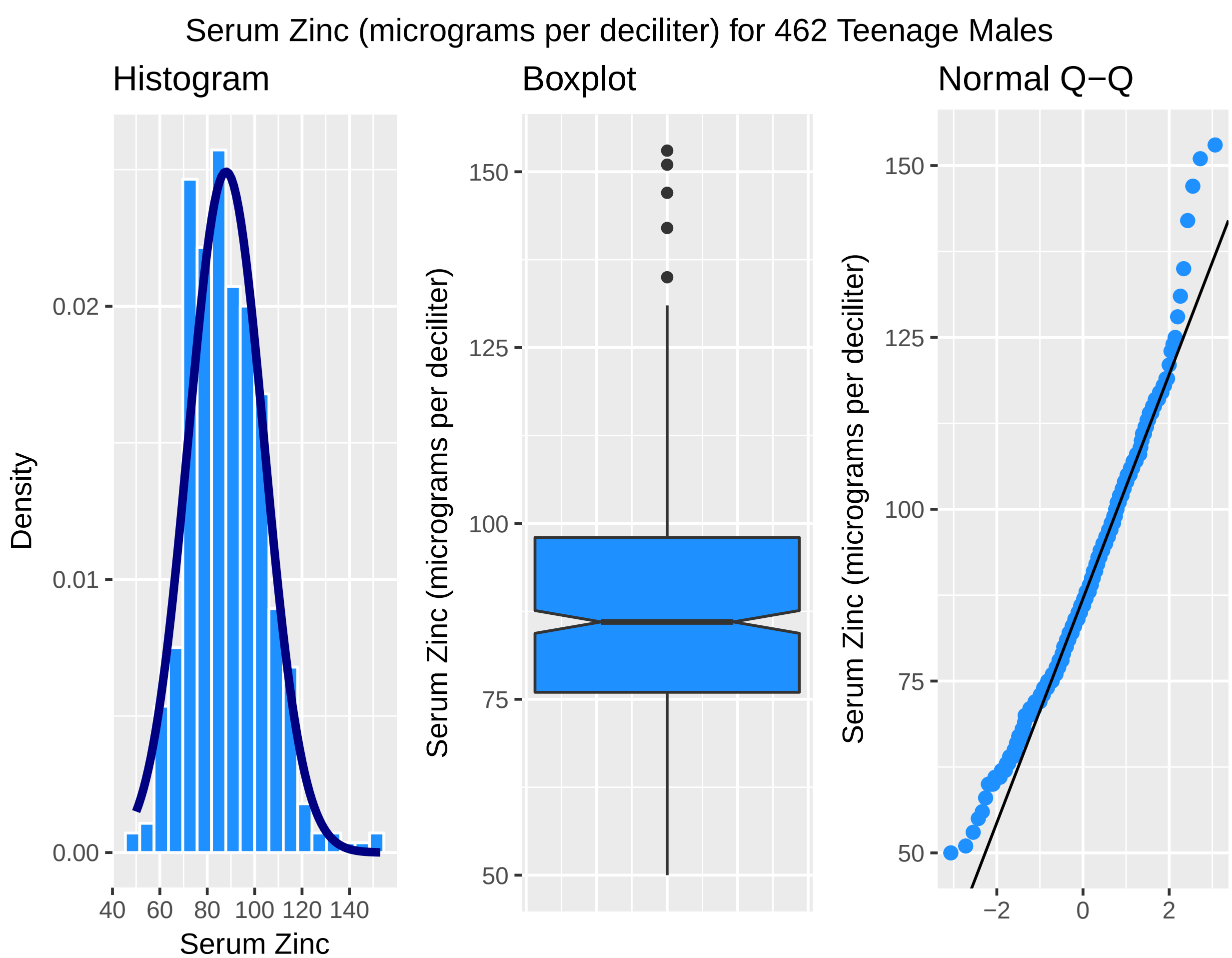
for the `zinc` results from the `serzinc` tibble. It does this while making use of several functions contained in the script `Love-R-functions.R`.

These functions include:

- `fd_bins` to estimate the Freedman-Diaconis bins setting for the histogram
- `qq_int` and `qq_slope` to facilitate the drawing of a line on the Normal Q-Q plot

```
p1 <- ggplot(serzinc, aes(x = zinc)) +
  geom_histogram(aes(y = ..density..), bins = fd_bins(serzinc$zinc),
                 fill = "dodgerblue", col = "white") +
  stat_function(fun = dnorm,
                args = list(mean = mean(serzinc$zinc),
                            sd = sd(serzinc$zinc)),
                lwd = 1.5, col = "navy") +
  labs(title = "Histogram",
       x = "Serum Zinc", y = "Density")
```

```
p2 <- ggplot(serzinc, aes(x = 1, y = zinc)) +
  geom_boxplot(fill = "dodgerblue", notch = TRUE) +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank()) +
  labs(title = "Boxplot",
       y = "Serum Zinc (micrograms per deciliter)", x = "")

p3 <- ggplot(serzinc, aes(sample = zinc)) +
  geom_qq(col = "dodgerblue", size = 2) +
  geom_abline(intercept = qq_int(serzinc$zinc),
              slope = qq_slope(serzinc$zinc)) +
  labs(title = "Normal Q-Q",
       y = "Serum Zinc (micrograms per deciliter)", x = "")

gridExtra::grid.arrange(p1, p2, p3, nrow=1,
   top = "Serum Zinc (micrograms per deciliter) for 462 Teenage Males")
```



These results include some of the more useful plots and numerical summaries when assessing shape, center and spread. The `zinc` data in the `serzinc` data frame appear to be slightly right skewed, with five outlier values on the high end of the scale, in particular.

You could potentially add `coord_flip() +` to the histogram, and this would have the advantage of getting all three plots oriented in the same direction, but then we (or at least I) lose the ability to tell the direction of skew at a glance from the direction of the histogram.

16.3.3 Numerical Summaries

This section describes some numerical summaries of interest to augment the plots in summarizing the center, spread and shape of the distribution of serum zinc among these 462 teenage males.

The tables below are built using two functions from the `Love-R-functions.R` script.

- `skew1` provides the skew$_1$ value for the `zinc` data and
- `Emp_Rule` provides the results of applying the `68-95-99.7` Empirical Rule to the `zinc` data.

```
pander(mosaic::favstats(serzinc$zinc))
```

min	Q1	median	Q3	max	mean	sd	n	missing
50	76	86	98	153	87.94	16	462	0

```
signif(skew1(serzinc$zinc),3)
```

```
[1] 0.121
```

The skew$_1$ value backs up our graphical assessment, that the data are slightly right skewed.

We can also assess how well the 68-95-99.7 Empirical Rule for a Normal distribution holds up for these data. Not too badly, as it turns out.

```
Emp_Rule(serzinc$zinc)
```

	count	proportion
Mean +/- 1 SD	323	0.6991
Mean +/- 2 SD	447	0.9675
Mean +/- 3 SD	458	0.9913
Entire Data Set	462	1

```
pander(psych::describe(serzinc$zinc))
```

Table 16.3: Table continues below

	vars	n	mean	sd	median	trimmed	mad	min	max
X1	1	462	87.94	16	86	87.17	16.31	50	153

	range	skew	kurtosis	se
X1	103	0.6191	0.8732	0.7446

Rounded to two decimal places, the standard deviation of the serum zinc data turns out to be 16, and so the standard error of the mean, shown as `se` in the `psych::describe` output, is 16 divided by the square root of the sample size, $n = 462$. This standard error is about to become quite important to us in building statistical inferences about the mean of the entire population of teenage males based on this sample.

Chapter 17

A Paired Sample Study: Lead in the Blood of Children

> One of the best ways to eliminate a source of variation and the errors of interpretation associated with it is through the use of matched pairs. Each subject in one group is matched as closely as possible by a subject in the other group. If a 45-year-old African-American male with hypertension is given a [treatment designed to lower their blood pressure], then we give a second, similarly built 45-year old African-American male with hypertension a placebo.

- Good (2005), section 5.2.4

17.1 The Lead in the Blood of Children Study

Morton et al. (1982) studied the absorption of lead into the blood of children. This was a matched-sample study, where the exposed group of interest contained 33 children of parents who worked in a battery manufacturing factory (where lead was used) in the state of Oklahoma. Specifically, each child with a lead-exposed parent was matched to another child of the same age, exposure to traffic, and living in the same neighborhood whose parents did not work in lead-related industries. So the complete study had 66 children, arranged in 33 matched pairs. The outcome of interest, gathered from a sample of whole blood from each of the children, was lead content, measured in mg/dl.

One motivation for doing this study is captured in the Abstract from Morton et al. (1982).

> It has been repeatedly reported that children of employees in a lead-related industry are at increased risk of lead absorption because of the high levels of lead found in the household dust of these workers.

The data are available in several places, including Table 5 of Pruzek and Helmreich (2009), in the BloodLead data set within the `PairedData` package in R, but we also make them available in the `bloodlead.csv` file. A table of the first three pairs of observations (blood lead levels for one child exposed to lead and the matched control) is shown below.

```r
head(bloodlead, 3)
```

```
# A tibble: 3 x 3
   pair exposed control
  <chr>   <int>   <int>
1   P01      38      16
2   P02      23      18
```

```
3   P03        41        18
```

- In each pair, one child was exposed (to having a parent working in the factory) and the other was not.
- Otherwise, though, each child was very similar to its matched partner.
- The data under `exposed` and `control` are the blood lead content, in mg/dl.

Our primary goal will be to estimate the difference in lead content between the exposed and control children, and then use that sample estimate to make inferences about the difference in lead content between the population of all children like those in the exposed group and the population of all children like those in the control group.

17.1.1 Our Key Questions for a Paired Samples Comparison

1. What is the **population** under study?

- All pairs of children living in Oklahoma near the factory in question, in which one had a parent working in a factory that exposed them to lead, and the other did not.

2. What is the **sample**? Is it representative of the population?

- The sample consists of 33 pairs of one exposed and one control child.
- This is a case-control study, where the children were carefully enrolled to meet the design criteria. Absent any other information, we're likely to assume that there is no serious bias associated with these pairs, and that assuming they represent the population effectively (and perhaps the broader population of kids whose parents work in lead-based industries more generally) may well be at least as reasonable as assuming they don't.

3. Who are the subjects / **individuals** within the sample?

- Each of our 33 pairs of children includes one exposed child and one unexposed (control) child.

4. What **data** are available on each individual?

- The blood lead content, as measured in mg/dl of whole blood.

17.1.2 Lead Study Caveats

Note that the children were not randomly selected from general populations of kids whose parents did and did not work in lead-based industries.

- To make inferences to those populations, we must make **strong assumptions** to believe, for instance, that the sample of exposed children is as representative as a random sample of children with similar exposures across the world would be.
- The researchers did have a detailed theory about how the exposed children might be at increased risk of lead absorption, and in fact as part of the study gathered additional information about whether a possible explanation might be related to the quality of hygiene of the parents (all of them were fathers, actually) who worked in the factory.
- This is an observational study, so that the estimation of a causal effect between parental work in a lead-based industry and children's blood lead content can be made, without substantial (and perhaps heroic) assumptions.

17.2 Exploratory Data Analysis for Paired Samples

We'll begin by adjusting the data in two ways.

- We'd like that first variable (`pair`) to be a `factor` rather than a `character` type in R, because we want to be able to summarize it more effectively. So we'll make that change.
- Also, we'd like to calculate the difference in lead content between the exposed and the control children in each pair, and we'll save that within-pair difference in a variable called `leaddiff`. We'll take `leaddiff` = `exposed - control` so that positive values indicate increased lead in the exposed child.

```
bloodlead <- bloodlead %>%
    mutate(pair = factor(pair),
           leaddiff = exposed - control)

bloodlead
```

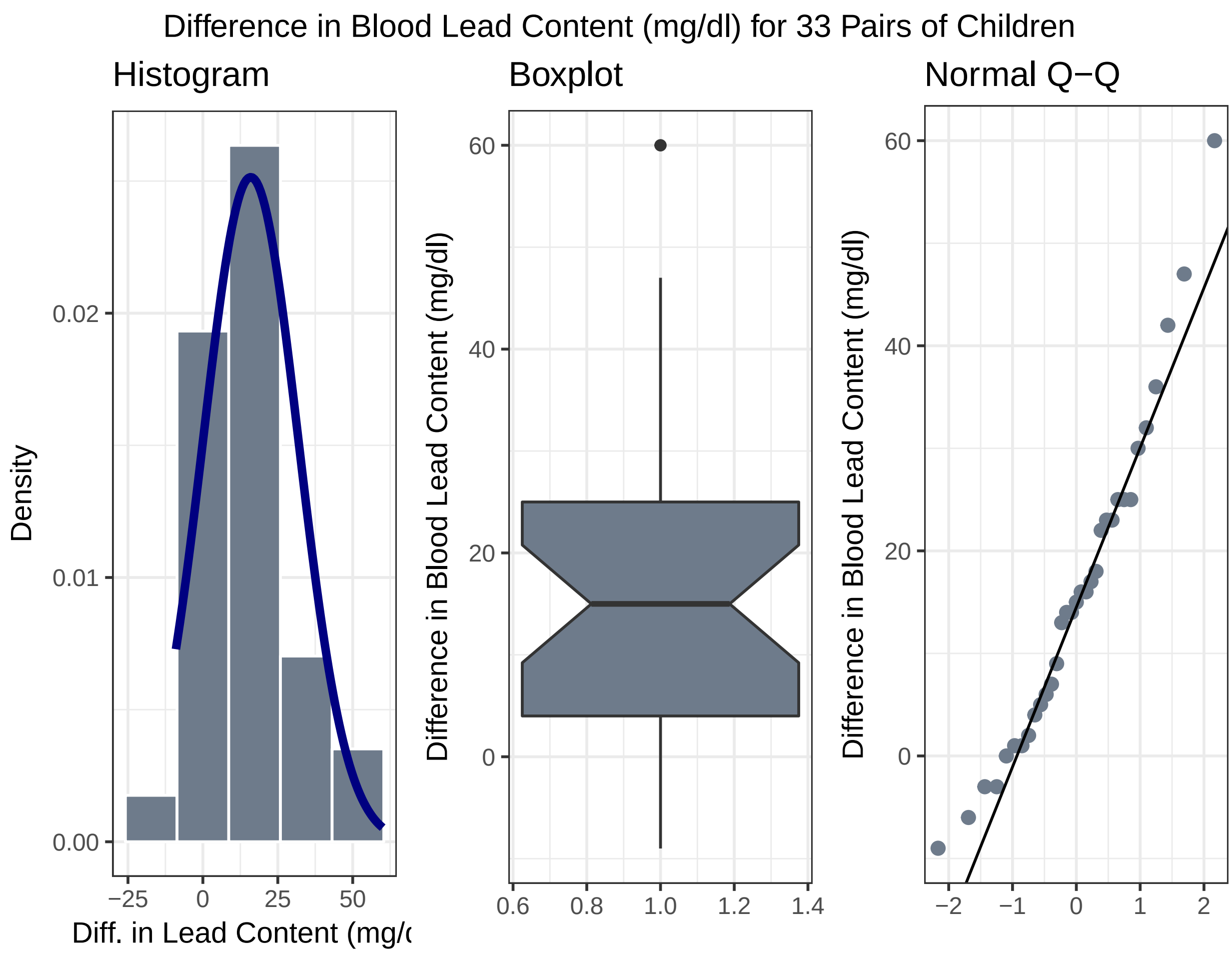
```
# A tibble: 33 x 4
     pair exposed control leaddiff
   <fctr>   <int>   <int>    <int>
 1    P01      38      16       22
 2    P02      23      18        5
 3    P03      41      18       23
 4    P04      18      24       -6
 5    P05      37      19       18
 6    P06      36      11       25
 7    P07      23      10       13
 8    P08      62      15       47
 9    P09      31      16       15
10    P10      34      18       16
# ... with 23 more rows
```

17.2.1 The Paired Differences

To begin, we focus on `leaddiff` for our exploratory work, which is the `exposed - control` difference in lead content within each of the 33 pairs. So, we'll have 33 observations, as compared to the 462 in the serum zinc data, but most of the same tools are still helpful.

```
p1 <- ggplot(bloodlead, aes(x = leaddiff)) +
    geom_histogram(aes(y = ..density..), bins = fd_bins(bloodlead$leaddiff),
                   fill = "lightsteelblue4", col = "white") +
    stat_function(fun = dnorm,
                  args = list(mean = mean(bloodlead$leaddiff),
                              sd = sd(bloodlead$leaddiff)),
                  lwd = 1.5, col = "navy") +
    labs(title = "Histogram",
         x = "Diff. in Lead Content (mg/dl)", y = "Density") +
    theme_bw()

p2 <- ggplot(bloodlead, aes(x = 1, y = leaddiff)) +
    geom_boxplot(fill = "lightsteelblue4", notch = TRUE) +
    theme(axis.text.x = element_blank(), axis.ticks.x = element_blank()) +
    labs(title = "Boxplot",
         y = "Difference in Blood Lead Content (mg/dl)", x = "") +
    theme_bw()

p3 <- ggplot(bloodlead, aes(sample = leaddiff)) +
    geom_qq(col = "lightsteelblue4", size = 2) +
    geom_abline(intercept = qq_int(bloodlead$leaddiff),
```

```
                slope = qq_slope(bloodlead$leaddiff)) +
    labs(title = "Normal Q-Q",
         y = "Difference in Blood Lead Content (mg/dl)", x = "") +
    theme_bw()

gridExtra::grid.arrange(p1, p2, p3, nrow=1,
    top = "Difference in Blood Lead Content (mg/dl) for 33 Pairs of Children")
```



Note that in all of this work, I plotted the paired differences. One obvious way to tell if you have paired samples is that you can pair every single subjects from one exposure group to the subjects in the other exposure group. Everyone has to be paired, so the sample sizes will always be the same in the two groups.

17.2.2 Numerical Summaries

```
pander(mosaic::favstats(bloodlead$leaddiff))
```

min	Q1	median	Q3	max	mean	sd	n	missing
-9	4	15	25	60	15.97	15.86	33	0

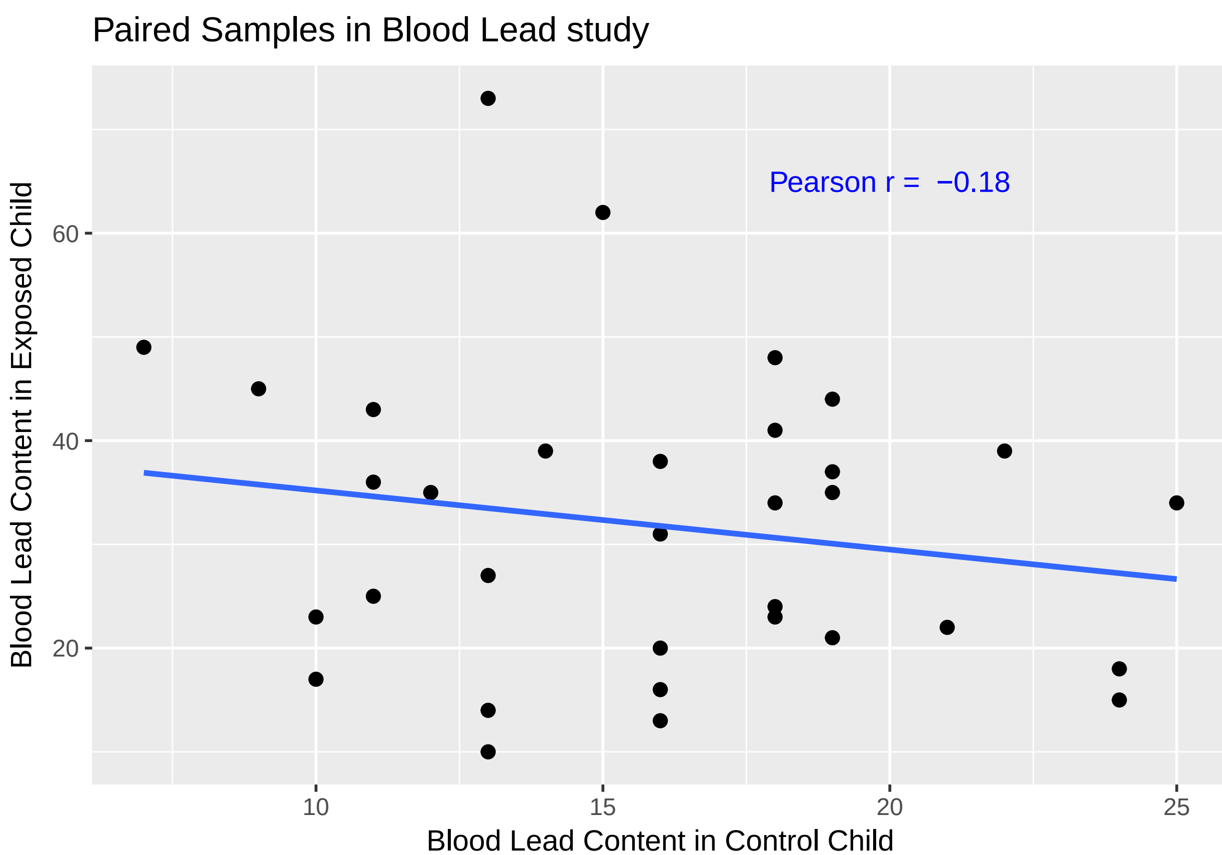
```
signif(skew1(bloodlead$leaddiff),3)
```

```
[1] 0.0611
```

17.2.3 Impact of Matching - Scatterplot and Correlation

Here, the data are paired by the study through matching on neighborhood, age and exposure to traffic. Each individual child's outcome value is part of a pair with the outcome value for his/her matching partner. We can see this pairing in several ways, perhaps by drawing a scatterplot of the pairs.

```
ggplot(bloodlead, aes(x = control, y = exposed)) +
    geom_point(size = 2) +
    geom_smooth(method = "lm", se = FALSE) +
    annotate("text", 20, 65, col = "blue",
             label = paste("Pearson r = ",
                           round(cor(bloodlead$control, bloodlead$exposed),2))) +
    labs(title = "Paired Samples in Blood Lead study",
         x = "Blood Lead Content in Control Child",
         y = "Blood Lead Content in Exposed Child")
```



If there is a strong linear relationship (usually with a positive slope, thus positive correlation) between the paired outcomes, then the pairing will be more helpful in terms of improving statistical power of the estimates we build than if there is a weak relationship.

- The stronger the Pearson correlation coefficient, the more helpful pairing will be.

- Here, a straight line model using the control child's blood lead content accounts for about 3% of the variation in blood lead content in the exposed child.
- As it turns out, pairing will have only a modest impact here on the inferences we draw in the study.

17.3 Looking at the Individual Samples: Tidying the Data with `gather`

For the purpose of estimating the difference between the exposed and control children, the summaries of the paired differences are what we'll need.

In some settings, however, we might also look at a boxplot, or violin plot, or ridgeline plot that showed the distributions of exposed and control children separately. But we will run into trouble because one variable (blood lead content) is spread across multiple columns (control and exposed.) The solution is to `gather` up that variable so as to build a new, tidy tibble.

Because the data aren't *tidied* here, so that we have one row for each subject and one column for each variable, we have to do some work to get them in that form for our usual plotting strategy to work well. For more on this approach (gathering and its opposite, spreading the data), visit the Tidy data chapter in Grolemund and Wickham (2017).

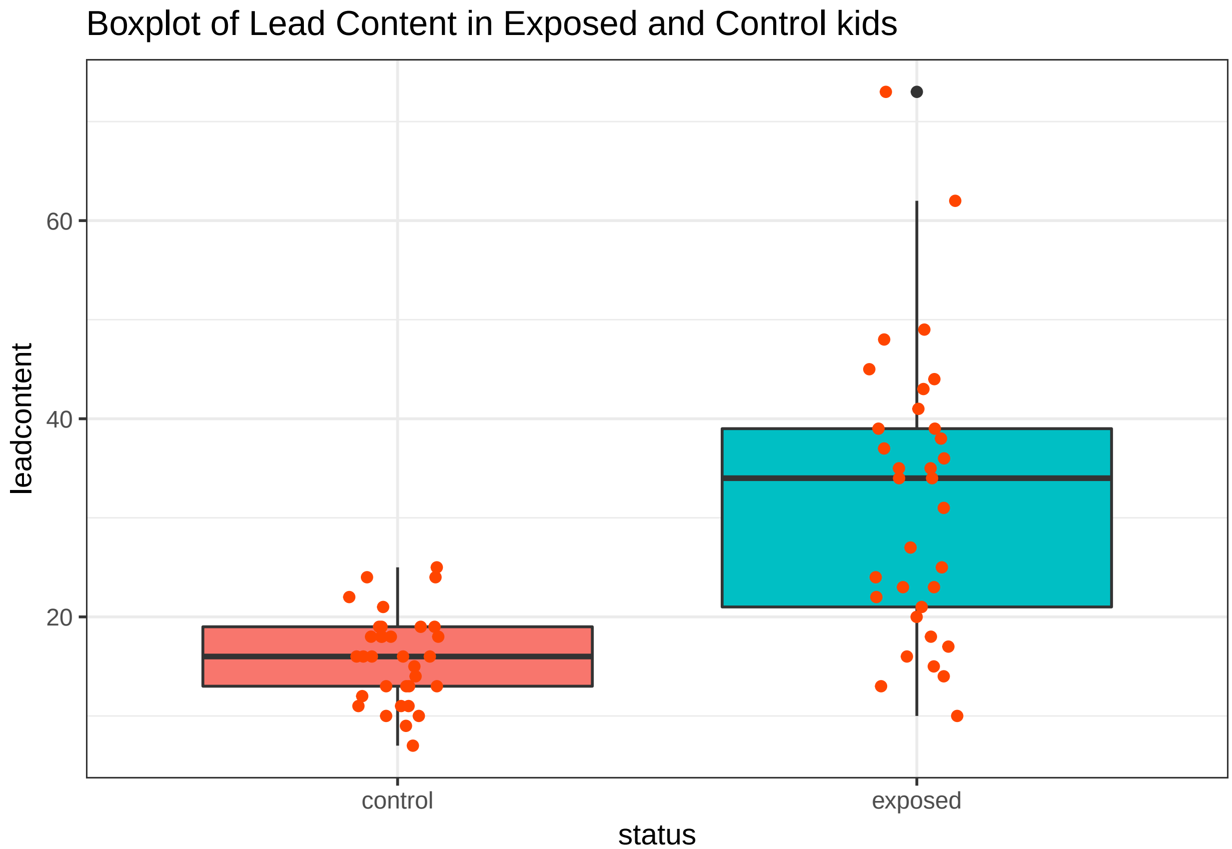
```
blead_tidied <- bloodlead %>%
    gather(control, exposed, key = "status", value = "leadcontent") %>%
    mutate(status = factor(status)) %>%
    select(-leaddiff)

blead_tidied
```

```
# A tibble: 66 x 3
     pair  status leadcontent
   <fctr>  <fctr>       <int>
 1    P01 control          16
 2    P02 control          18
 3    P03 control          18
 4    P04 control          24
 5    P05 control          19
 6    P06 control          11
 7    P07 control          10
 8    P08 control          15
 9    P09 control          16
10    P10 control          18
# ... with 56 more rows
```

And now, we can plot as usual to compare the two samples.

First, we'll look at a boxplot, showing all of the data.

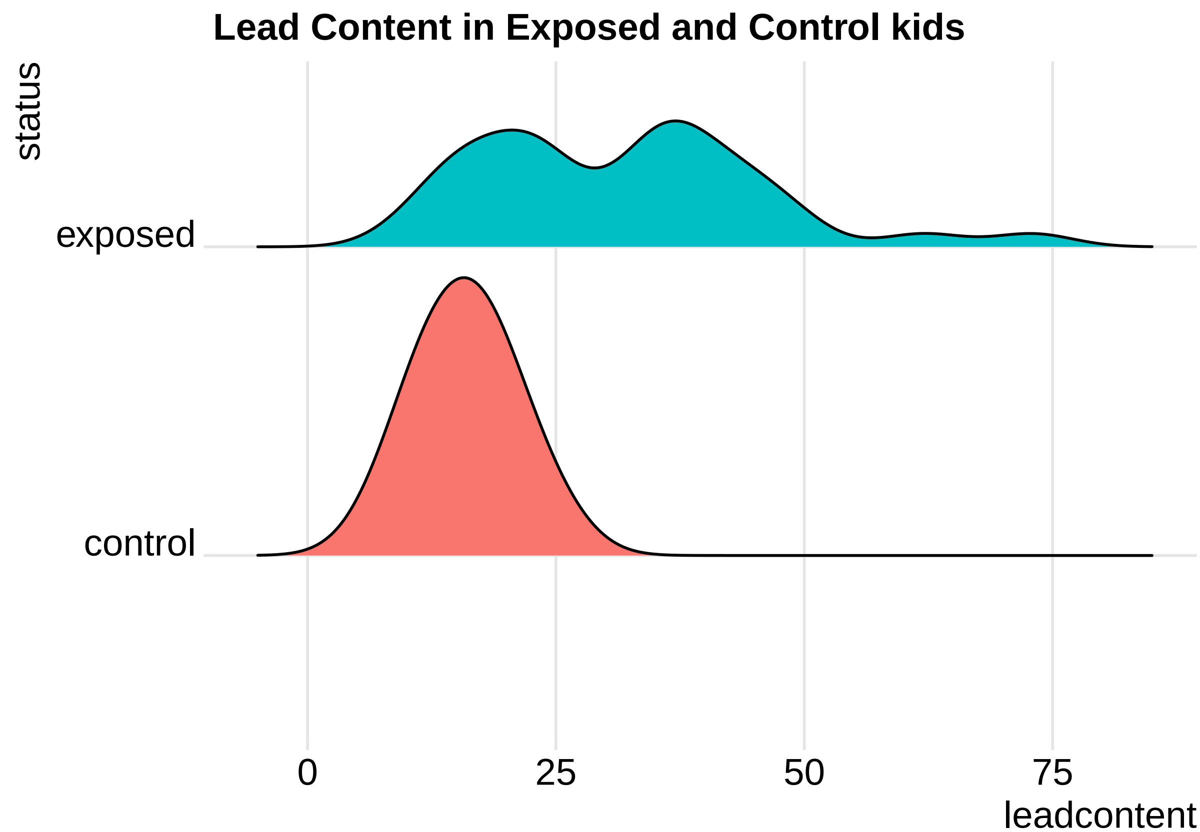
```
ggplot(blead_tidied, aes(x = status, y = leadcontent, fill = status)) +
    geom_boxplot() +
    geom_jitter(width = 0.1, height = 0, color = "orangered") +
    guides(fill = FALSE) +
    labs(title = "Boxplot of Lead Content in Exposed and Control kids") +
    theme_bw()
```



We'll also look at a ridgeline plot, because Dr. Love likes them, even though they're really more useful when we're comparing more than two samples.

```
ggplot(blead_tidied, aes(x = leadcontent, y = status, fill = status)) +
    ggridges::geom_density_ridges(scale = 0.9) +
    guides(fill = FALSE) +
    labs(title = "Lead Content in Exposed and Control kids") +
    ggridges::theme_ridges()
```

```
Picking joint bandwidth of 4.01
```



Both the center and the spread of the distribution are substantially larger in the exposed group than in the matched controls. Of course, numerical summaries show these patterns, too.

```
blead_tidied %>% group_by(status) %>%
    summarise(n = n(),
              median = median(leadcontent),
              Q1 = quantile(leadcontent, 0.25),
              Q3 = quantile(leadcontent, 0.75),
              mean = mean(leadcontent),
              sd = sd(leadcontent))
```

```
# A tibble: 2 x 7
   status     n median    Q1    Q3  mean    sd
   <fctr> <int>  <int> <dbl> <dbl> <dbl> <dbl>
1 control    33     16    13    19  15.9  4.54
2 exposed    33     34    21    39  31.8 14.41
```

Chapter 18

A Study Comparing Two Independent Samples: Ibuprofen in Sepsis Trial

18.1 The Ibuprofen in Sepsis Randomized Clinical Trial

We will be working with a sample from the Ibuprofen in Sepsis study, as reported in Bernard et al. (1997). My source for these data is Dupont (2002).

> Ibuprofen has been shown to have effects on sepsis in humans, but because of their small samples (fewer than 30 patients), previous studies have been inadequate to assess effects on mortality. We sought to determine whether ibuprofen can alter rates of organ failure and mortality in patients with the sepsis syndrome, how the drug affects the increased metabolic demand in sepsis (e.g., fever, tachypnea, tachycardia, hypoxemia, and lactic acidosis), and what potential adverse effects the drug has in the sepsis syndrome.

- Bernard et al. (1997), Abstract.

In this study, patients meeting specific criteria (including elevated temperature) for a diagnosis of sepsis were recruited if they fulfilled an additional set of study criteria (see Bernard et al. (1997)) in the intensive care unit at one of seven participating centers. The full trial involved 455 patients, of which our sample includes 300. 150 of our patients were randomly assigned to the Ibuprofen group and 150 to the Placebo group. In either case, the patient received intravenous treatment (ibuprofen or placebo.) This was also a *double-blind* study, where neither the patients nor their care providers know, during the execution of the trial, what intervention group was assigned to each patient.

For the moment, we will focus on two variables:

- **treat**, which specifies the treatment group (Ibuprofen or Placebo), which was assigned via randomization to each patient, and
- **temp_drop**, the outcome of interest, measured as the change from baseline to 2 hours later in degrees Celsius. Positive values indicate improvement, that is, a *drop* in temperature over the 2 hours following the baseline measurement.

The data in the `sepsis.csv` file also contains the subject's

- *id*, which is just a code
- *race* (three levels: White, AfricanA or Other)
- *apache* = baseline APACHE II score, a severity of disease score ranging from 0 to 71 with higher scores indicating more severe disease and a higher mortality risk
- *temp_0* = baseline temperature, degrees Celsius.

but we'll ignore those for now.

```r
sepsis
```

```
# A tibble: 300 x 6
      id     treat     race apache temp_0 temp_drop
   <chr>     <chr>    <chr>  <int>  <dbl>     <dbl>
 1  S002 Ibuprofen AfricanA     14   38.7       1.4
 2  S004 Ibuprofen    White      3   38.3       0.4
 3  S005   Placebo    White      5   38.6       0.0
 4  S006 Ibuprofen    White     13   38.2      -0.2
 5  S009 Ibuprofen    White     25   38.2       0.6
 6  S011 Ibuprofen    White     21   38.1      -0.4
 7  S012   Placebo    White     14   38.6      -0.1
 8  S014   Placebo    White     23   37.9       0.3
 9  S016   Placebo    White     16   38.1       0.1
10  S020 Ibuprofen    Other     20   39.2       1.5
# ... with 290 more rows
```

```r
sepsis <- sepsis %>%
    mutate(treat = factor(treat),
           race = factor(race))

summary(select(sepsis, treat, temp_drop))
```

```
       treat       temp_drop
 Ibuprofen:150   Min.   :-2.700
 Placebo  :150   1st Qu.:-0.100
                 Median : 0.300
                 Mean   : 0.308
                 3rd Qu.: 0.700
                 Max.   : 3.100
```

Again, the complete study included 455 patients, but our sample includes 300. We have exactly 150 in the Ibuprofen group and 150 in the Placebo group, as it turns out. I picked the sample so as to exclude patients with missing values for our outcome of interest, and then selected a random sample of 150 Ibuprofen and 150 Placebo patients from the rest of the group, and converted the temperatures and changes from Fahrenheit to Celsius.

18.1.1 Matched Pairs vs. Two Independent Samples

These data were obtained from two independent samples, rather than as matched pairs.

- Remember that if the sample sizes were different, we'd know we have independent samples, because matched pairs requires that each subject in the "treated" group be matched to a single, unique member of the "control" group, and thus that we have exactly as many "treated" as "control" subjects.
- But having as many subjects in one treatment group as the other (which is called a *balanced design*) is only necessary, and not sufficient, for us to conclude that matched pairs are used.
- We only have matched pairs if each individual observation in the "treatment" group is matched to one and only one observation in the "control" group by the way in which the data were gathered.
    - Paired data can arise in several ways. The most common is a "pre-post" study where subjects are measured both before and after an exposure happens. In observational studies, we often match up subjects who did and did not receive an exposure so as to account for differences on things like age, sex, race and other covariates. This, of course, is what happens in the Lead in the Blood of Children study from Section 17.

- If the data are from paired samples, we should (and in fact) must form paired differences, with no subject left unpaired.
- If we cannot line up the data comparing two samples of quantitative data so that the links between the individual "treated" and "control" observations to form matched pairs are evident, then the data are not paired.

As Bock, Velleman, and De Veaux (2004) suggest,

> … if you know the data are paired, you can take advantage of that fact - in fact, you *must* take advantage of it. … You must decide whether the data are paired from understanding how they were collected and what they mean. … There is no test to determine whether the data are paired.

18.1.2 Our Key Questions for an Independent Samples Comparison

1. What is the **population** under study?

- All patients in the intensive care unit with sepsis who meet the inclusion and exclusion criteria of the study, at the entire population of health centers like the ones included in the trial.

2. What is the **sample**? Is it representative of the population?

- The sample consists of 300 patients. It is a convenient sample from the population under study.
- This is a randomized clinical trial. 150 of the patients were assigned to Ibuprofen, and the rest to Placebo. It is this treatment assignment that is randomized, not the selection of the sample as a whole.
- In expectation, randomization of individuals to treatments, as in this study, should be expected to eliminate treatment selection bias.

3. Who are the subjects / **individuals** within the sample?

- 150 patients who received Ibuprofen and a completely different set of 150 patients who received Placebo.
- There is no match or link between the patients. They are best thought of as independent samples.

4. What **data** are available on each individual?

- The key variables are the treatment indicator (Ibuprofen or Placebo) and the outcome (drop in temperature in the 2 hours following administration of the randomly assigned treatment.)

18.1.3 RCT Caveats

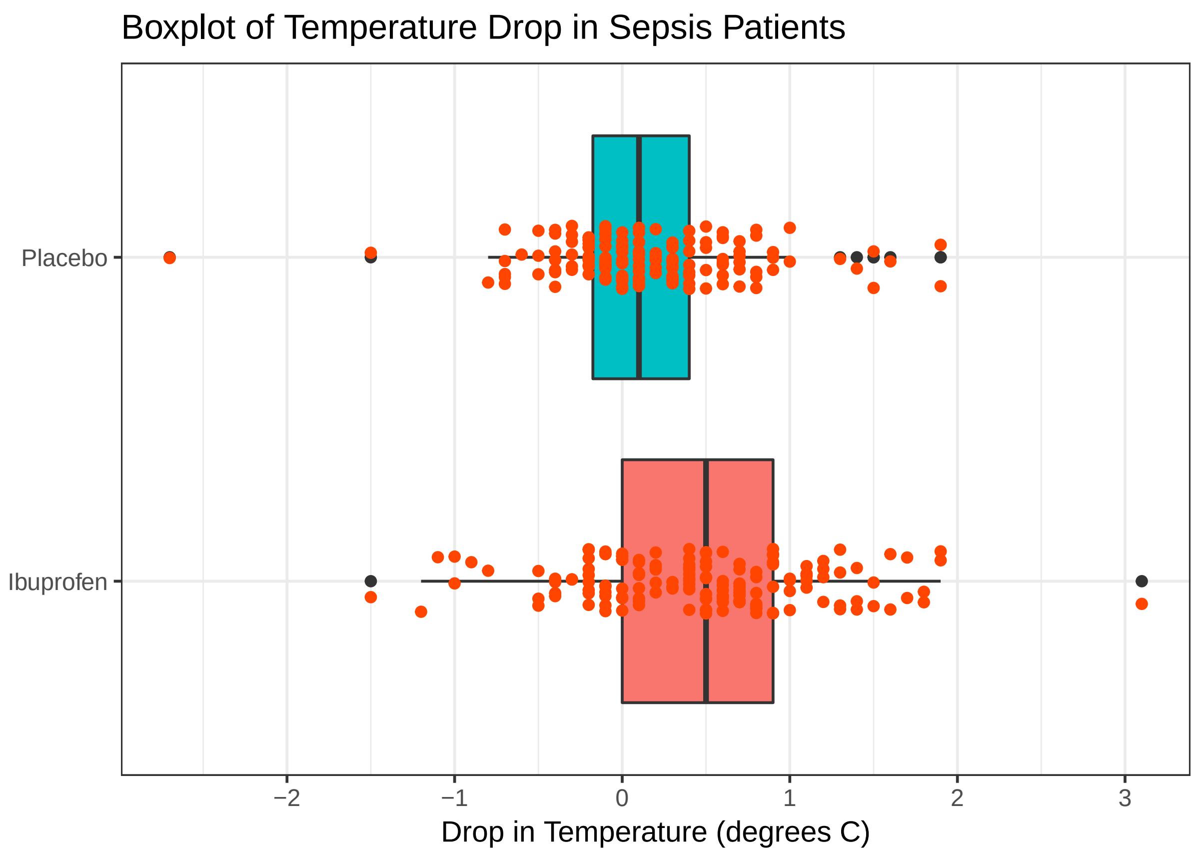
The placebo-controlled, double-blind randomized clinical trial, especially if pre-registered, is often considered the best feasible study for assessing the effectiveness of a treatment. While that's not always true, it is a very solid design. The primary caveat is that the patients who are included in such trials are rarely excellent representations of the population of potentially affected patients as a whole.

18.2 Exploratory Data Analysis

First, we'll look at a boxplot, showing all of the individual data as added-on dots.

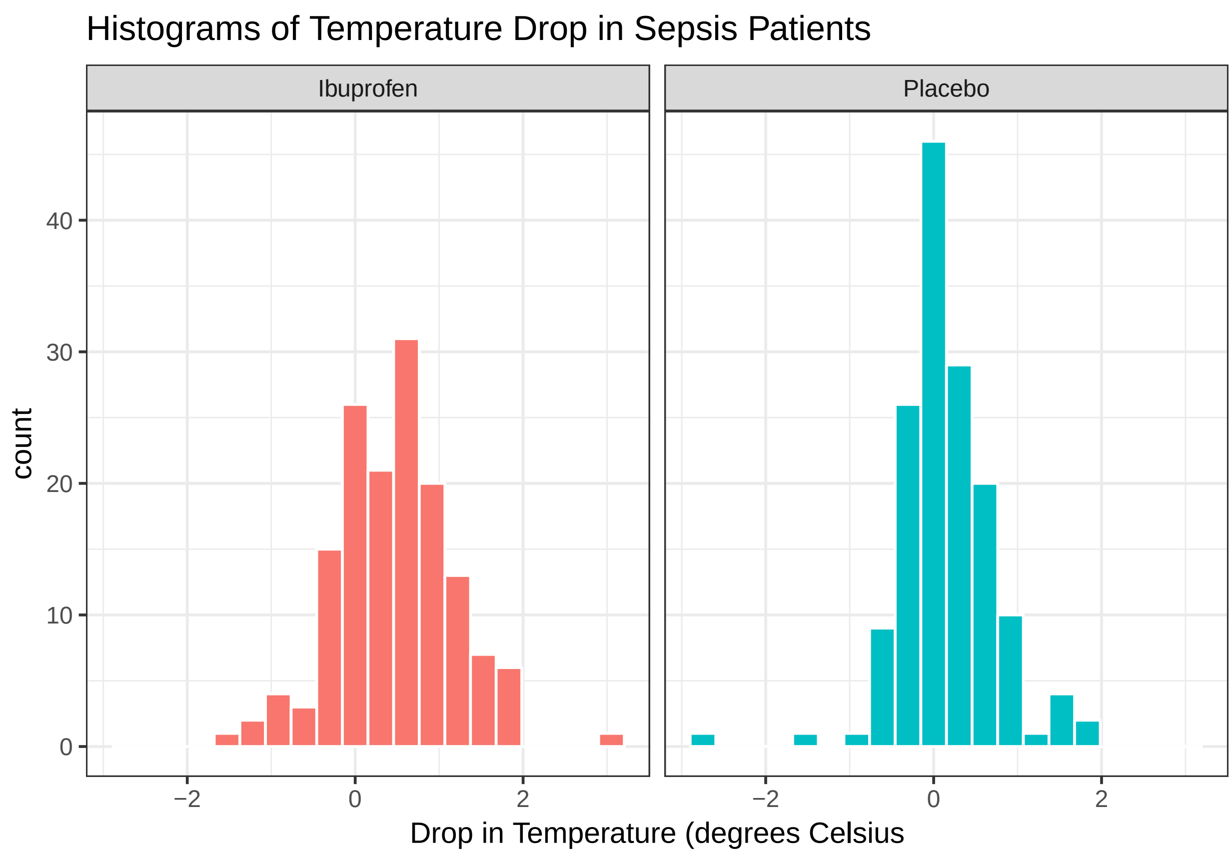
```
ggplot(sepsis, aes(x = treat, y = temp_drop, fill = treat)) +
    geom_boxplot() +
    geom_jitter(width = 0.1, height = 0, color = "orangered") +
    guides(fill = FALSE) +
    labs(title = "Boxplot of Temperature Drop in Sepsis Patients",
         x = "", y = "Drop in Temperature (degrees C)") +
```

```
    coord_flip() +
    theme_bw()
```



Next, we'll consider faceted histograms of the data.

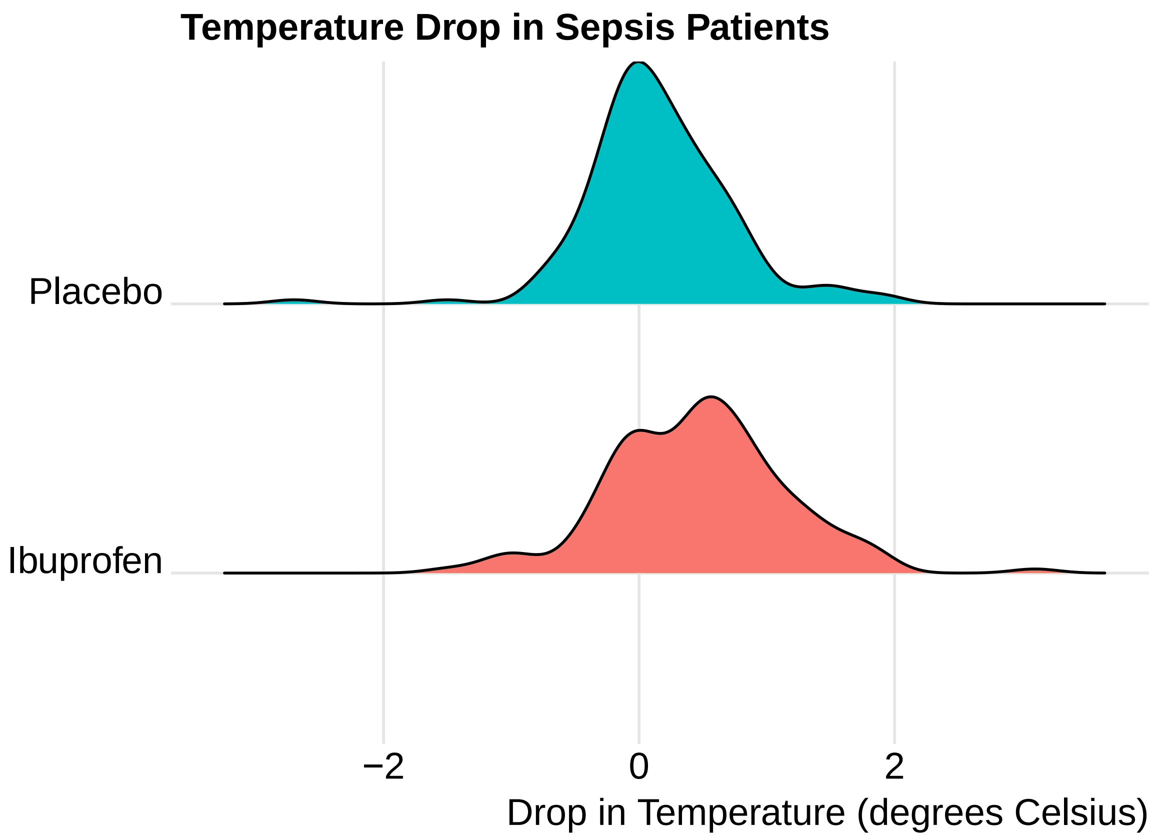
```
ggplot(sepsis, aes(x = temp_drop, fill = treat)) +
    geom_histogram(color = "white", bins = 20) +
    guides(fill = FALSE) +
    labs(title = "Histograms of Temperature Drop in Sepsis Patients",
         x = "Drop in Temperature (degrees Celsius")) +
    theme_bw() +
    facet_wrap(~ treat)
```



We'll also look at a ridgeline plot.

```
ggplot(sepsis, aes(x = temp_drop, y = treat, fill = treat)) +
    ggridges::geom_density_ridges(scale = 0.9) +
    guides(fill = FALSE) +
    labs(title = "Temperature Drop in Sepsis Patients",
         x = "Drop in Temperature (degrees Celsius)", y = "") +
    ggridges::theme_ridges()
```

```
Picking joint bandwidth of 0.182
```



The center of the ibuprofen distribution is shifted a bit towards the more positive (greater improvement) direction, it seems, than is the distribution for the placebo patients. Here are some key numerical summaries, within the treatment groups, which buoy this conclusion.

```
sepsis %>% group_by(treat) %>%
    summarise(n = n(),
              median = median(temp_drop),
              Q1 = quantile(temp_drop, 0.25),
              Q3 = quantile(temp_drop, 0.75),
              mean = mean(temp_drop),
              sd = sd(temp_drop))
```

```
# A tibble: 2 x 7
      treat     n median     Q1    Q3  mean    sd
     <fctr> <int>  <dbl>  <dbl> <dbl> <dbl> <dbl>
1 Ibuprofen   150    0.5  0.000   0.9 0.464 0.688
2   Placebo   150    0.1 -0.175   0.4 0.153 0.571
```

Chapter 19

Confidence Intervals for a Single Sample of Quantitative Data

Suppose that we are interested in learning something about a population or process, from which we can obtain a sample that consists of a subset of potential results from that population or process. The main goal for many of the parametric models that are a large part of statistics is to estimate population parameters, like a population mean, or regression coefficient, on the basis of a sample. When we do this, we want to describe not only our best guess at the parameter – referred to as a *point estimate*, but also say something useful about the uncertainty in our estimate, to let us more completely assess what the data have to tell us. A key tool for doing this is a **confidence interval**, described here in some detail.

Essentially every textbook on introductory statistics describes the development of a confidence interval, at least for a mean. Good supplemental resources include Diez, Barr, and Çetinyaka-Rundel (n.d.), Bock, Velleman, and De Veaux (2004) and Pagano and Gauvreau (2000), for instance.

We'll develop confidence intervals to compare parameters about two populations (either through matched pairs or independent samples) with confidence intervals soon. Here, we'll consider the problem of estimating a confidence interval to describe the mean (or median) of the population represented by a single sample of quantitative data. Our main example uses data from the Serum Zinc study, as described in Section 16.

19.1 Defining a Confidence Interval

A confidence interval for a population or process mean uses data from a sample (and perhaps some additional information) to identify a range of potential values for the population mean, which, if certain assumptions hold, can be assumed to provide a reasonable estimate for the true population mean. A confidence interval consists of:

1. An interval estimate describing the population parameter of interest (here the population mean), and
2. A probability statement, expressed in terms of a confidence level.

19.2 Estimating the Population Mean from the Serum Zinc data

As an example, suppose that we are willing to assume that the mean serum zinc level across the entire population of teenage males, μ, follows a Normal distribution (and so, summarizing it with a mean is a rational thing to do.) Suppose that we are also willing to assume that the 462 teenage males contained in the `serzinc` tibble are a random sample from that complete population. While we know the mean of the sample of 462 boys, we don't know μ, the mean across all teenage males. So we need to estimate it.

Later, we will find that, with these assumptions in place, we can find a 90% confidence interval for the mean serum zinc level across the entire population of teenage males. This 90% confidence interval for μ turns out to be (86.71, 89.16) micrograms per deciliter. How would you interpret this result?

- Some people think this means that there is a 90% chance that the true mean of the population, μ, falls between 86.71 and 89.16 micrograms per deciliter. That's not correct.
- The population mean is a constant **parameter** of the population of interest. That constant is not a random variable, and does not change. So the actual probability of the population mean falling inside that range is either 0 or 1.
- Our confidence is in our process.
  - It's in the sampling method (random sampling) used to generate the data, and in the assumption that the population follows a Normal distribution.
  - It's captured in our accounting for one particular type of error (called *sampling error*) in developing our interval estimate, while assuming all other potential sources of error are negligible.

So, what's closer to the truth is:

- If we used this same method to sample data from the true population of teenage males, and built 100 such 90% confidence intervals, then about 90 of them would contain the true population mean.

19.3 Confidence vs. Significance Level

We've estimated a 90% confidence interval for the population mean serum zinc level among teenage boys using the `serzinc` data.

- We call 100(1-α)%, here, 90%, or 0.90, the *confidence* level, and
- α = 10%, or 0.10 is called the *significance* level.

If we had instead built a series of 100 different 95% confidence intervals, then about 95 of them would contain the true value of μ.

Let's look more closely at the issue of estimating a population **mean** based on a sample of observations. We will need three critical pieces - the sample, the confidence level, and the margin of error, which is based on the standard error of a sample mean, when we are estimating a population mean.

19.4 The Standard Error of a Sample Mean

The standard error, generally, is the name we give to the standard deviation associated with any particular parameter estimate.

- If we are using a sample mean based on a sample of size n to estimate a population mean, the **standard error of that sample mean** is $\sigma/\sqrt{n}$, where σ is the standard deviation of the measurements in the population.

- We often estimate this particular standard error with its sample analogue, $s/\sqrt{n}$, where s is the sample standard deviation.

- Other statistics have different standard errors.
  - $\sqrt{p(1-p)/n}$ is the standard error of the sample proportion p estimated using a sample of size n.
  - $\sqrt{\frac{1-r^2}{n-2}}$ is the standard error of the sample Pearson correlation r estimated using n pairs of observations.

In developing a confidence interval for a population mean, we may be willing to assume that the data in our sample are drawn from a Normally distributed population. If so, the most common and useful means of building a confidence interval makes use of the t distribution (sometimes called Student's t) and the notion of a *standard error*.

19.5 The t distribution and Confidence Intervals for μ

In practical settings, we will use the t distribution to estimate a confidence interval from a population mean whenever we:

- are willing to assume that the sample is drawn at random from a population or process with a Normal distribution,
- are using our sample to estimate both the mean and standard deviation, and
- have a small sample size.

19.5.1 The Formula

We can build a 100(1-α)% confidence interval using the t distribution, using the sample mean $\bar{x}$, the sample size n, and the sample standard deviation s.

The two-sided 100(1-α)% confidence interval (based on a t test) is:

$$\bar{x} \pm t_{\alpha/2,n-1}(s/\sqrt{n})$$

where $t_{\alpha/2,n-1}$ is the value that cuts off the top $\alpha/2$ percent of the t distribution, with $n-1$ degrees of freedom.

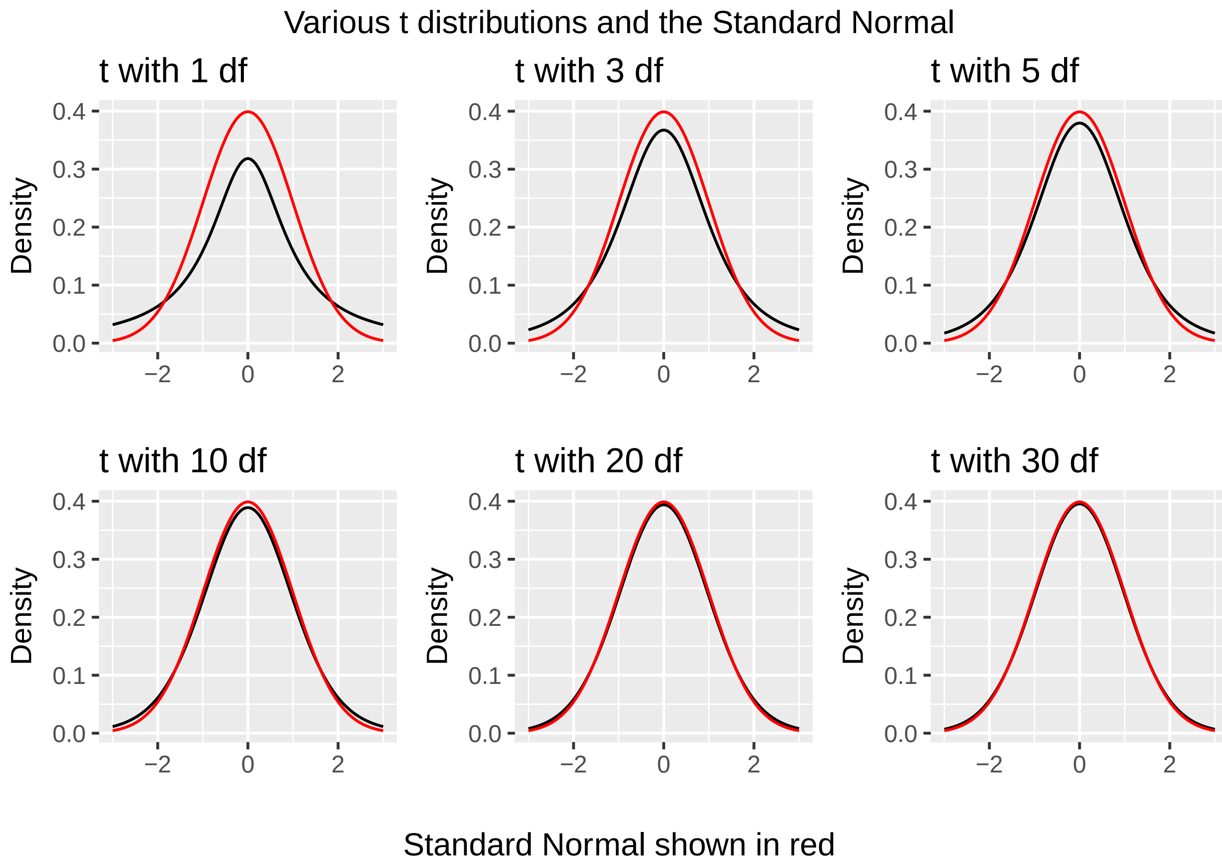
We obtain the relevant cutoff value in R by substituting in values for `alphaover2` and `n-1` into the following line of R code:

```
qt(alphaover2, df = n-1, lower.tail=FALSE)
```

19.5.2 Student's t distribution

Student's t distribution looks a lot like a Normal distribution, when the sample size is large. Unlike the normal distribution, which is specified by two parameters, the mean and the standard deviation, the t distribution is specified by one parameter, the degrees of freedom.

- t distributions with large numbers of degrees of freedom are more or less indistinguishable from the standard Normal distribution.
- t distributions with smaller degrees of freedom (say, with df < 30, in particular) are still symmetric, but are more outlier-prone than a Normal distribution



19.5.3 Building the CI "by hand" for the Serum Zinc data

In the serum zinc data, we observe the following results in our sample.

```
mosaic::favstats(serzinc$zinc)
```

```
 min Q1 median Q3 max mean sd   n missing
  50 76     86 98 153 87.9 16 462       0
```

Suppose we wish to build a 90% confidence interval for the true mean serum zinc level across the entire population of teenage males. The confidence level will be 90%, or 0.90, and so the α value, which is 1 - confidence = 0.10.

So what we know going in is that:

- We want $\alpha = 0.10$, because we're creating a 90% confidence interval.
- The sample size $n = 462$ serum zinc measurements.
- The sample mean of those measurements, $\bar{x} = 87.937$ micrograms per deciliter.
- The sample standard deviation of those measurements, $s = 16.005$ micrograms per deciliter.
- As a result, our standard error of the sample mean is estimated well with $s/\sqrt{n} = 16.005/\sqrt{462} = 0.745$.

So now, we are ready to calculate our 90% confidence interval.

The two-sided 100(1-α)% confidence interval (based on a t test) is: $\bar{x} \pm t_{\alpha/2,n-1}(s/\sqrt{n})$, or

- The 90% CI for μ is thus $87.937 \pm t_{0.10/2,462-1}$ (0.745)
  - To calculate the t cutoff value for $\alpha = 0.10$ and $n = 462$, we use

`qt(0.10/2, df = 462-1, lower.tail=FALSE)` = 1.648

- So the 90% CI for μ is 87.937 $\pm$ 1.648 x 0.745, or
- 87.937 $\pm$ 1.228, or
- (86.71, 89.16)

So, our 90% confidence interval for the true population mean serum zinc level, based on our sample of 462 patients, is (86.71, 89.16) micrograms per deciliter.

19.5.4 Getting R to build a CI for the Serum Zinc data

Happily, R does all of this work, and with less inappropriate rounding.

```
t.test(serzinc$zinc, conf.level = 0.90, alternative = "two.sided")
```

```

	One Sample t-test

data:  serzinc$zinc
t = 100, df = 500, p-value <2e-16
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
 86.7 89.2
sample estimates:
mean of x
     87.9
```

And again, our 90% confidence interval for the true population mean serum zinc level, based on our sample of 462 patients, is (86.7, 89.2) micrograms per deciliter[1].

19.5.5 Interpreting the Result

An appropriate interpretation of the 90% two-sided confidence interval above follows:

- (86.71, 89.16) micrograms per deciliter is a 90% two-sided confidence interval for the population mean serum zinc level among teenage males.
- Our point estimate for the true population mean serum zinc level is 87.9. The values in the interval (86.71, 89.16) represent a reasonable range of estimates for the true population mean serum zinc level, and we are 90% confident that this method of creating a confidence interval will produce a result containing the true population mean serum zinc level.
- Were we to draw 100 samples of size 462 from the population described by this sample, and use each such sample to produce a confidence interval in this manner, approximately 90 of those confidence intervals would cover the true population mean serum zinc level.

19.5.6 What if we want a 95% or 99% confidence interval instead?

The `t.test` function in R has an argument to specify the desired confidence level.

```
t.test(serzinc$zinc, conf.level = 0.95, alternative = "two.sided")
```

```

	One Sample t-test

data:  serzinc$zinc
```

[1]Since the measured zinc levels appear as integers, we should certainly not include any more than one additional significant figure in our confidence interval.

```
t = 100, df = 500, p-value <2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 86.5 89.4
sample estimates:
mean of x
     87.9
```

```r
t.test(serzinc$zinc, conf.level = 0.99, alternative = "two.sided")
```

```

    One Sample t-test

data:  serzinc$zinc
t = 100, df = 500, p-value <2e-16
alternative hypothesis: true mean is not equal to 0
99 percent confidence interval:
 86.0 89.9
sample estimates:
mean of x
     87.9
```

Below, we see two-sided confidence intervals for various levels of α.

Confidence Level	α	Two-Sided Interval Estimate for Zinc Level Population Mean, μ	Point Estimate for Zinc Level Population Mean, μ
80% or 0.80	0.20	(87, 88.9)	87.9
90% or 0.90	0.10	(86.7, 89.2)	87.9
95% or 0.95	0.05	(86.5, 89.4)	87.9
99% or 0.99	0.01	(86, 89.9)	87.9

What happens to the width of the confidence interval in this table as the confidence level changes?

19.5.7 One-sided vs. Two-sided Confidence Intervals

In some situations, we are concerned with either an upper limit for the population mean μ or a lower limit for μ, but not both.

If we, as before, have a sample of size n, with sample mean $\bar{x}$ and sample standard deviation s, then:

- The upper bound for a one-sided 100(1-α)% confidence interval for the population mean is $\mu \leq \bar{x} + t_{\alpha,n-1}(\frac{s}{\sqrt{n}})$, with lower "bound" $-\infty$.
- The corresponding lower bound for a one-sided 100(1 - α) CI for μ would be $\mu \geq \bar{x} - t_{\alpha,n-1}(\frac{s}{\sqrt{n}})$, with upper "bound" ∞.

19.5.8 Calculating a one-sided confidence interval for the population mean

```r
t.test(serzinc$zinc, conf.level = 0.90, alternative = "greater")
```

```

    One Sample t-test
```

```
data:  serzinc$zinc
t = 100, df = 500, p-value <2e-16
alternative hypothesis: true mean is greater than 0
90 percent confidence interval:
  87 Inf
sample estimates:
mean of x
     87.9
```

```r
t.test(serzinc$zinc, conf.level = 0.90, alternative = "less")
```

```

    One Sample t-test

data:  serzinc$zinc
t = 100, df = 500, p-value = 1
alternative hypothesis: true mean is less than 0
90 percent confidence interval:
 -Inf 88.9
sample estimates:
mean of x
     87.9
```

19.5.9 Relationship between One-Sided and Two-Sided CIs

Note the relationship between the *two-sided* 80% confidence interval, and the *one-sided* 90% confidence intervals.

Confidence	α	Type of Interval	Interval Estimate for Zinc Level Population Mean, μ
80% (.80)	0.20	Two-Sided	(86.98, 88.89)
90% (.90)	0.10	One-Sided (Less Than)	$\mu < 88.89$.
90% (.90)	0.10	One-Sided (Greater Than)	$\mu > 86.98$.

Why does this happen? The 80% two-sided interval is placed so as to cut off the top 10% of the distribution with its upper bound, and the bottom 10% of the distribution with its lower bound. The 90% "less than" one-sided interval is placed so as to have its lower bound cut off the top 10% of the distribution.

The same issue appears when we consider two-sided 90% and one-sided 95% confidence intervals.

Confidence	α	Type of Interval	Interval Estimate for Zinc Level Population Mean, μ
90% (.90)	0.10	Two-Sided	(86.71, 89.16)
95% (.95)	0.05	One-Sided (Less Than)	$\mu < 89.16$.
95% (.95)	0.05	One-Sided (Greater Than)	$\mu > 86.71$.

Again, the 90% two-sided interval cuts off the top 5% and bottom 5% of the distribution with its bounds. The 95% "less than" one-sided interval also has its lower bound cut off the top 5% of the distribution.

19.5.10 Using the `broom` package with the t test

The `broom` package takes the messy output of built-in functions in R, such as `lm`, `t.test` or `wilcox.test`, and turns them into tidy data frames. A detailed description of the package and three of its key functions is found at https://github.com/tidyverse/broom.

For example, we can use the `tidy` function within `broom` to create a single-row tibble of the key results from a t test.

```
tt <- t.test(serzinc$zinc, conf.level = 0.95, alternative = "two.sided")
broom::tidy(tt)
```

```
  estimate statistic p.value parameter conf.low conf.high
1     87.9       118       0       461     86.5      89.4
             method alternative
1 One Sample t-test   two.sided
```

We can thus pull the endpoints of a 95% confidence interval directly from this output. `broom` also has a `glance` function, which returns the same information as `tidy` in the case of a t-test.

```
tt2 <- t.test(serzinc$zinc, conf.level = 0.90, alternative = "less")
broom::glance(tt2)
```

```
  estimate statistic p.value parameter conf.low conf.high
1     87.9       118       1       461     -Inf      88.9
             method alternative
1 One Sample t-test        less
```

19.6 Bootstrap Confidence Intervals for μ

19.6.1 What is a Bootstrap and Why Should I Care?

The bootstrap (and in particular, what's known as bootstrap resampling) is a really good idea that you should know a little bit about. Good (2005) and Good and Hardin (2006) are excellent resources, for instance.

If we want to know how accurately a sample mean estimates the population mean, we would ideally like to take a very, very large sample, because if we did so, we could conclude with something that would eventually approach mathematical certainty that the sample mean would be very close to the population mean.

But we can rarely draw enormous samples. So what can we do?

19.6.2 Resampling is A Big Idea

One way to find out how precise our estimates are is to run them on multiple samples of the same size. This *resampling* approach was codified originally by Brad Efron in, for example, Efron (1979).

Oversimplifying a lot, the idea is that if we sample (with replacement) from our current sample, we can draw a new sample of the same size as our original.

- And if we repeat this many times, we can generate as many samples of, say, 462 zinc levels, as we like.
- Then we take these thousands of samples and calculate (for instance) the sample mean for each, and plot a histogram of those means.
- If we then cut off the top and bottom 5% of these sample means, we obtain a reasonable 90% confidence interval for the population mean.

19.6.3 When is a Bootstrap Confidence Interval for μ Reasonable?

The interval will be reasonable as long as we're willing to believe that:

- the original sample was a random sample (or at least a completely representative sample) from a population,
- and that the samples are independent of each other
- and that the samples are identically distributed (even though that distribution may not be Normal.)

A downside is that you and I will get (somewhat) different answers if we resample from the same data without setting the same random seed.

19.6.4 Bootstrap: Steps to estimate a confidence interval for μ

To avoid the Normality assumption, and take advantage of modern computing power, we use R to obtain a bootstrap confidence interval for the population mean based on a sample.

What the computer does:

1. Resample the data with replacement, until it obtains a new sample that is equal in size to the original data set.
2. Calculates the statistic of interest (here, a sample mean.)
3. Repeat the steps above many times (the default is 1,000 using our approach) to obtain a set of 1,000 sample means.
4. Sort those 1,000 sample means in order, and estimate the 95% confidence interval for the population mean based on the middle 95% of the 1,000 bootstrap samples.
5. Send us a result, containing the sample mean, and a 95% confidence interval for the population mean

19.6.5 Using R to estimate a 90% CI for μ with the bootstrap

The command that we use to obtain a CI for μ using the basic nonparametric bootstrap and without assuming a Normally distributed population, is `smean.cl.boot`, a part of the `Hmisc` package in R.

```
set.seed(431); Hmisc::smean.cl.boot(serzinc$zinc, B = 1000, conf.int = 0.90)
```

```
 Mean Lower Upper
 87.9  86.8  89.2
```

- Remember that the t-based 90% CI for μ was (86.71, 89.16), according to the following output…

```
t.test(serzinc$zinc, conf.level = 0.90, alternative = "two.sided")
```

```

	One Sample t-test

data:  serzinc$zinc
t = 100, df = 500, p-value <2e-16
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
 86.7 89.2
sample estimates:
mean of x
     87.9
```

19.6.6 Comparing Bootstrap and T-Based Confidence Intervals

- The `smean.cl.boot` function (unlike most R functions) deletes missing data automatically, as does the `smean.cl.normal` function, which produces the t-based confidence interval.

```
set.seed(431); Hmisc::smean.cl.boot(serzinc$zinc, B = 1000, conf.int = 0.90)
```

```
Mean Lower Upper
87.9  86.8  89.2
```

```
Hmisc::smean.cl.normal(serzinc$zinc, conf.int = 0.90)
```

```
Mean Lower Upper
87.9  86.7  89.2
```

Bootstrap resampling confidence intervals do not follow the general confidence interval strategy using a point estimate $\pm$ a margin for error.

- A bootstrap interval is often asymmetric, and while it will generally have the point estimate (the sample mean) near its center, for highly skewed data, this will not necessarily be the case.
- We will usually use either 1,000 (the default) or 10,000 bootstrap replications for building confidence intervals – practically, it makes little difference.

19.6.7 90% CI for μ via bootstrap, changing minor details

Suppose we change the random seed that we set, or change the number (`B`) of desired bootstrap replications.

```
set.seed(431); Hmisc::smean.cl.boot(serzinc$zinc, B = 1000, conf.int = 0.90)
```

```
Mean Lower Upper
87.9  86.8  89.2
```

```
set.seed(431212); Hmisc::smean.cl.boot(serzinc$zinc, B = 1000, conf.int = 0.90)
```

```
Mean Lower Upper
87.9  86.7  89.2
```

```
set.seed(431212); Hmisc::smean.cl.boot(serzinc$zinc, B = 2000, conf.int = 0.90)
```

```
Mean Lower Upper
87.9  86.7  89.2
```

19.6.8 Bootstrap: Changing the Confidence Level

```
set.seed(431654); Hmisc::smean.cl.boot(serzinc$zinc, conf.int = 0.95, B = 1000)
```

```
Mean Lower Upper
87.9  86.4  89.3
```

```
set.seed(431321); Hmisc::smean.cl.boot(serzinc$zinc, conf.int = 0.99, B = 1000)
```

```
Mean Lower Upper
87.9  86.1  89.8
```

19.6.9 Bootstrap: Obtaining a One-sided Confidence Interval

If you want to estimate a one tailed confidence interval for the population mean using the bootstrap, then the procedure is as follows:

1. Determine α, the significance level you want to use in your one-sided confidence interval. Remember that α is 1 minus the confidence level. Let's assume we want a 90% one-sided interval, so $\alpha = 0.10$.
2. Double α to determine the significance level we will use in the next step to fit a two-sided confidence interval.
3. Fit a two-sided confidence interval with confidence level $100(1 - 2 * \alpha)$. Let the bounds of this interval be (a, b).
4. The one-sided (greater than) confidence interval will have a as its lower bound.
5. The one-sided (less than) confidence interval will have b as its upper bound.

Suppose that we want to find a 95% one-sided upper bound for the population mean serum zinc level among teenage males, μ, using the bootstrap.

Since we want a 95% confidence interval, we have $\alpha = 0.05$. We double that to get $\alpha = 0.10$, which implies we need to instead fit a two-sided 90% confidence interval.

```
set.seed(43101); Hmisc::smean.cl.boot(serzinc$zinc, conf.int = 0.90, B = 1000)
```

```
 Mean Lower Upper
 87.9  86.7  89.3
```

Since the upper bound of this two-sided 90% CI is 89.27, that will also be the upper bound for a 95% one-sided CI.

19.6.10 Bootstrap CI for the Population Median

If we are willing to do a small amount of programming work in R, we can obtain bootstrap confidence intervals for other population parameters besides the mean. One statistic of common interest is the median. How do we find a confidence interval for the population median using a bootstrap approach? The easiest way I know of makes use of the `boot` package, as follows.

In step 1, we specify a new function to capture the medians from our sample.

```
f.median <- function(y, id)
{    median ( y[id])  }
```

In step 2, we summon the `boot` package and call the `boot.ci` function:

```
set.seed(431787)
boot.ci(boot (serzinc$zinc, f.median, 1000), conf=0.90, type="basic")
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = boot(serzinc$zinc, f.median, 1000), conf = 0.9,
    type = "basic")

Intervals :
Level      Basic
90%   (84, 87 )
Calculations and Intervals on Original Scale
```

This yields a 90% confidence interval[2] for the population median serum zinc level.

Recall that the sample median for the serum zinc levels in our sample of 462 teenage males was 86 micrograms per deciliter.

19.6.11 Bootstrap CI for the IQR

If for some reason, we want to find a 95% confidence interval for the population value of the inter-quartile range via the bootstrap, we can do it.

```
IQR(serzinc$zinc)
```

```
[1] 22
```

```
f.IQR <- function(y, id)
{    IQR (y[id]) }

set.seed(431207); boot.ci(boot (serzinc$zinc, f.IQR, 1000),
                          conf=0.95, type="basic")
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 1000 bootstrap replicates

CALL :
boot.ci(boot.out = boot(serzinc$zinc, f.IQR, 1000), conf = 0.95,
    type = "basic")

Intervals :
Level      Basic
95%   (19, 24 )
Calculations and Intervals on Original Scale
```

19.6.12 Bootstrap Resampling: Advantages and Caveats

The bootstrap may seem like the solution to all estimation problems. In theory, we could use the same approach to find a confidence interval for any other parameter – it's not perfect, but it is very useful. Bootstrap procedures exist for virtually any statistical comparison - the t-test analog is just one of many possibilities, and bootstrap methods are rapidly gaining on more traditional approaches in the literature thanks mostly to faster computers.

The great advantage of the bootstrap is its relative simplicity, but don't forget that many of the original assumptions of the t-based confidence interval still hold.

- Using a bootstrap does eliminate the need to worry about the Normality assumption in small sample size settings, but it still requires independent and identically distributed samples from the population of interest.

The bootstrap produces clean and robust inferences (such as confidence intervals) in many tricky situations. It is still possible that the results can be both:

- **inaccurate** (i.e. they can, include the true value of the unknown population mean less often than the stated confidence probability) and
- **imprecise** (i.e., they can include more extraneous values of the unknown population mean than is desirable).

[2]Actually, the boot.ci function can provide up to five different types of confidence interval (see the help file) if we change to type="all", and some of those other versions have attractive properties. However, we'll stick with the basic approach in 431.

19.7 Large-Sample Normal Approximation CIs for μ

If we were in the position of knowing the standard deviation of the population of interest precisely[3], we could use that information to build a 100(1-α)% confidence interval using the Normal distribution, based on the sample mean $\bar{x}$, the sample size n, and the (known) population standard deviation σ.

When we have a large sample size (often as little as 60 observations), we can use this approach to get a very close approximation to the result we would get using the t distribution, and there are many settings where obtaining the Z test result is more appropriate in estimating more complicated parameters than the population mean.

19.7.1 The Large Sample Formula for the CI around μ

The two-sided 100(1-α)% confidence interval for a population mean μ (based on the Normal distribution) is:

- The Lower Bound is $\bar{x} - Z_{\alpha/2}(\sigma/\sqrt{n})$ and the Upper Bound is $\bar{x} + Z_{\alpha/2}(\sigma/\sqrt{n})$

where $Z_{\alpha/2}$ is the value that cuts off the top $\alpha/2$ percent of the standard Normal distribution (the Normal distribution with mean 0 and standard deviation 1).

19.7.2 Obtaining the $Z_{\alpha/2}$ value using `qnorm`

We can obtain this cutoff value from R by substituting in the desired proportion for `alphaover2` into the `qnorm` function as follows:

`qnorm(alphaover2, lower.tail=FALSE)`

For example, if we are building a 95% confidence interval, we have 100(1-α) = 95, so that α is 0.05, or 5%. This means that the cutoff value we need to find is $Z_{0.05/2} = Z_{.025}$, and this turns out to be 1.96.

```r
qnorm(0.025, lower.tail=FALSE)
```

```
[1] 1.96
```

19.7.3 Commonly Used Cutoffs based on the Normal Distribution

- If we're building a two-sided 95% confidence interval, we'll use $Z_{.025} = 1.96$
- For a two-sided 90% confidence interval, we use $Z_{.05} = 1.645$
- For a two-sided 99% confidence interval, we use $Z_{.005} = 2.576$
- For a two-sided 50% confidence interval, we use $Z_{.25} = 0.67$
- For a two-sided 68% confidence interval, we use $Z_{.16} = 0.99$

19.7.4 Lots of CIs use the Normal distribution

- The usual 95% confidence interval for large samples is an estimate $\pm$ 2 standard errors[4].
- Also, from the Normal distribution, an estimate $\pm$ 1 standard error is a 68% confidence interval, and an estimate $\pm$ 2/3 of a standard error is a 50% confidence interval.
- A 50% interval is particularly easy to interpret because the true value should be inside the interval about as often as it is not.

[3] Practical applications usually demand a subtler approach, but this normal distribution-based approach can help us fix some key ideas

[4] The use of 2 standard errors for a confidence interval for a population mean is certainly reasonable whenever n is 60 or more. This is because the t distribution with 59 degrees of freedom has a 0.025 cutoff of 2.0, anyway.

- A 95% interval is thus about three times as wide as a 50% interval.
- In general, the larger the confidence required, the wider the interval will need to be.

19.7.5 Large-Sample Confidence Interval for Zinc Levels

Since we have a fairly large sample ($n = 462$) in the `serzinc` data, we could consider using a large-sample approach (assuming the sample standard deviation is equal to the population standard deviation, and then using the Normal distribution) to estimate a confidence interval for the mean zinc levels in the population of all 15-17 year old males like those in our sample.

In the zinc levels within the `serzinc` data, we have

- a sample of $n = 462$ observations
- with sample mean $\bar{x} = 87.94$ and standard deviation $s = 16$
- and suppose we want to, at first, find a 95% confidence interval, so $\alpha = 0.05$

The 95% confidence interval is calculated as $\bar{x} \pm Z_{\alpha/2}(\sigma/\sqrt{n})$, and here we will assume that $s = \sigma$ which may be reasonable with a fairly large sample size:

87.94 ± (1.96)(16 / $\sqrt{462}$) = 87.94 ± 1.46, or (86.48, 89.4)

Our 95% confidence interval for the population mean is (86.48, 89.4) μg/dl. Were we to generate 100 such intervals, approximately 95 of those intervals would be expected to include the true mean of the entire population of 15-17 year old males like those in our sample.

19.7.6 Comparing Z and t-based Intervals for Serum Zinc

For the serum zinc data, we had $n = 462$ observations in our sample.

Do the z-based and t-based confidence intervals differ much?

α	Confidence Level	Confidence Interval	Method
0.05	95%	(86.48, 89.40)	Z (known σ; large n)
0.05	95%	(86.47, 89.40)	t (σ unknown)
0.10	90%	(86.72, 89.16)	Z (known σ; large n)
0.10	90%	(86.71, 89.16)	t (σ unknown)

19.7.7 One-Sided Confidence Intervals in Large Samples

The upper bound for a one-sided 100(1-α)% confidence interval for the population mean is:

$\mu \leq \bar{x} + Z_{\alpha}(\frac{\sigma}{\sqrt{n}})$, with lower "bound" $-\infty$.

The corresponding lower bound for a one-sided 100(1 - α) CI for μ would be:

$\mu \geq \bar{x} - Z_{\alpha}(\frac{\sigma}{\sqrt{n}})$, with upper "bound" ∞.

19.8 Wilcoxon Signed Rank Procedure for CIs

19.8.1 Confidence Intervals for the Median of a Population

It turns out to be difficult, without the bootstrap, to estimate an appropriate confidence interval for the median of a population, which might be an appealing thing to do, particularly if the sample data are clearly not Normally distributed, so that a median seems like a better summary of the center of the data. Bootstrap procedures are available to perform the task.

The Wilcoxon signed rank approach can be used as an alternative to t-based procedures to build interval estimates for the population *pseudo-median* when the population cannot be assumed to follow a Normal distribution.

As it turns out, if you're willing to assume the population is **symmetric** (but not necessarily Normally distributed) then the pseudo-median is actually equal to the population median.

19.8.2 What is a Pseudo-Median?

The pseudo-median of a particular distribution G is the median of the distribution of (u + v)/2, where both u and v have the same distribution (G).

- If the distribution G is symmetric, then the pseudomedian is equal to the median.
- If the distribution is skewed, then the pseudomedian is not the same as the median.
- For any sample, the pseudomedian is defined as the median of all of the midpoints of pairs of observations in the sample.

19.8.3 Getting the Wilcoxon Signed Rank-based CI in R

```
wilcox.test(serzinc$zinc, conf.int=TRUE, conf.level=0.95)
```

```

    Wilcoxon signed rank test with continuity correction

data:  serzinc$zinc
V = 1e+05, p-value <2e-16
alternative hypothesis: true location is not equal to 0
95 percent confidence interval:
 86.0 88.5
sample estimates:
(pseudo)median
          87.5
```

19.8.4 Interpreting the Wilcoxon CI for the Population Median

If we're willing to believe the `zinc` levels come from a population with a symmetric distribution, the 95% Confidence Interval for the population median would be (86, 88.5)

For a non-symmetric population, this only applies to the *pseudo-median*.

Note that the pseudo-median (87.5) is actually closer here to the sample mean (86) than it is to the sample median (87.9).

19.8.5 Using the `broom` package with the Wilcoxon test

We can also use the `tidy` function within `broom` to create a single-row tibble of the key results from a Wilcoxon test.

```
wt <- wilcox.test(serzinc$zinc, conf.int=TRUE, conf.level=0.95)
broom::tidy(wt)
```

```
  estimate statistic p.value conf.low conf.high
1     87.5    106953   2e-77       86      88.5
                                                method alternative
1 Wilcoxon signed rank test with continuity correction   two.sided
```

19.9 General Advice

We have described four different approaches to estimating a confidence interval for the center of a distribution of quantitative data.

1. The most commonly used approach uses the t distribution to estimate a confidence interval for a population/process mean. This requires some extra assumptions, most particularly that the underlying distribution of the population values is at least approximately Normally distributed.
2. A more modern and very general approach uses the idea of the bootstrap to estimate a confidence for a population/process parameter, which could be a mean, median or other summary statistic. The bootstrap, and the underlying notion of *resampling* is an important idea that lets us avoid some of the assumptions (in particular Normality) that are required by other methods. Bootstrap confidence intervals involve random sampling, so that the actual values obtained will differ a bit across replications.
3. A third approach makes more substantial assumptions - it uses the Normal distribution rather than a t, and assumes (among other things) very large samples. For estimating a single mean, we'll rarely use this, but for estimating more complex parameters, particularly in Part C when discussing modeling, we will occasionally use this approach.
4. Finally, the Wilcoxon signed-rank method is one of a number of inferential tools which transform the data to their *ranks* before estimating a confidence interval. This avoids some assumptions, but yields inferences about a less-familiar parameter - the pseudo-median.

Most of the time, the **bootstrap** provides an adequate solution when estimating a confidence interval to describe the population value of a parameter (mean or median, most commonly) from a distribution, when our data consists of a single sample of quantitative information.

Chapter 20

Confidence Intervals from Two Paired Samples of Quantitative Data

Here, we'll consider the problem of estimating a confidence interval to describe the difference in population means (or medians) based on a comparison of two samples of quantitative data, gathered using a matched pairs design. Specifically, we'll use as our example the Lead in the Blood of Children study, described in Section 17.

Recall that in that study, we measured blood lead content, in mg/dl, for 33 matched pairs of children, one of which was exposed (had a parent working in a battery factory) and the other of which was control (no parent in the battery factory, but matched to the exposed child by age, exposure to traffic and neighborhood). We then created a variable called `leaddiff` which contained the (exposed - control) differences within each pair.

```
bloodlead
```

```
# A tibble: 33 x 4
     pair exposed control leaddiff
   <fctr>   <int>   <int>    <int>
 1    P01      38      16       22
 2    P02      23      18        5
 3    P03      41      18       23
 4    P04      18      24       -6
 5    P05      37      19       18
 6    P06      36      11       25
 7    P07      23      10       13
 8    P08      62      15       47
 9    P09      31      16       15
10    P10      34      18       16
# ... with 23 more rows
```

20.1 t-based CI for Population Mean of Paired Differences, μ_d.

In R, there are at least three different methods for obtaining the t-based confidence interval for the population difference in means between paired samples. They are all mathematically identical. The key idea is to calculate the paired differences (exposed - control, for example) in each pair, and then treat the result as if it were a single sample and apply the methods discussed in Section 19.

20.1.1 Method 1

We can use the single-sample approach, applied to the variable containing the paired differences. Let's build a **90%** two-sided confidence interval for the population mean of the difference in blood lead content across all possible pairs of an exposed (parent works in a lead-based industry) and a control (parent does not) child, μ_d.

```
t.test(bloodlead$leaddiff, conf.level = 0.90, alt = "two.sided")
```

```

    One Sample t-test

data:  bloodlead$leaddiff
t = 6, df = 30, p-value = 2e-06
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
 11.3 20.6
sample estimates:
mean of x 
       16 
```

The 90% confidence interval is (11.29, 20.65) according to this t-based procedure. An appropriate interpretation of the 90% two-sided confidence interval would be:

- (11.29, 20.65) milligrams per deciliter is a 90% two-sided confidence interval for the population mean difference in blood lead content between exposed and control children.
- Our point estimate for the true population difference in mean blood lead content is 15.97 mg.dl. The values in the interval (11.29, 20.65) mg/dl represent a reasonable range of estimates for the true population difference in mean blood lead content, and we are 90% confident that this method of creating a confidence interval will produce a result containing the true population mean difference.
- Were we to draw 100 samples of 33 matched pairs from the population described by this sample, and use each such sample to produce a confidence interval in this manner, approximately 90 of those confidence intervals would cover the true population mean difference in blood lead content levels.

20.1.2 Method 2

Or, we can apply the single-sample approach to a calculated difference in blood lead content between the exposed and control groups. Here, we'll get a **95%** two-sided confidence interval for μ_d, instead of the 90% interval we obtained above.

```
t.test(bloodlead$exposed - bloodlead$control,
       conf.level = 0.95, alt = "two.sided")
```

```

    One Sample t-test

data:  bloodlead$exposed - bloodlead$control
t = 6, df = 30, p-value = 2e-06
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 10.3 21.6
sample estimates:
mean of x 
       16 
```

20.1.3 Method 3

Or, we can provide R with two separate samples (unaffected and affected) and specify that the samples are paired. Here, we'll get a **99% one-sided** confidence interval (lower bound) for μ_d, the population mean difference in blood lead content.

```
t.test(bloodlead$exposed, bloodlead$control, conf.level = 0.99,
       paired = TRUE, alt = "greater")
```

```

	Paired t-test

data:  bloodlead$exposed and bloodlead$control
t = 6, df = 30, p-value = 1e-06
alternative hypothesis: true difference in means is greater than 0
99 percent confidence interval:
 9.21  Inf
sample estimates:
mean of the differences
                     16
```

Again, the three different methods using `t.test` for paired samples will all produce identical results if we feed them the same confidence level and type of interval (two-sided, greater than or less than).

20.1.4 Assumptions

If we are building a confidence interval for the mean μ of a population or process based on a sample of observations drawn from that population, then we must pay close attention to the assumptions of those procedures. The confidence interval procedure for the population mean μ using the t distribution assumes that:

1. We want to estimate the population mean μ.
2. We have drawn a sample of observations at random from the population of interest.
3. The sampled observations are drawn from the population independently and have identical distributions.
4. The population follows a Normal distribution. At the very least, the sample itself is approximately Normal.

20.1.5 Using `broom` for a t test using paired samples

The `broom` package places the results of a t test, among other things, into a tidy data frame.

```
broom::tidy(t.test(bloodlead$exposed - bloodlead$control,
       conf.level = 0.95, alt = "two.sided"))
```

```
  estimate statistic  p.value parameter conf.low conf.high
1       16      5.78 2.04e-06        32     10.3      21.6
             method alternative
1 One Sample t-test   two.sided
```

20.2 Bootstrap CI for mean difference using paired samples

The same bootstrap approach is used for paired differences as for a single sample. We again use the `smean.cl.boot()` function in the `Hmisc` package to obtain bootstrap confidence intervals for the population mean, μ_d, of the paired differences in blood lead content.

```
set.seed(431555)
Hmisc::smean.cl.boot(bloodlead$leaddiff, conf.int = 0.95, B = 1000)
```

```
Mean Lower Upper
16.0  10.8  21.3
```

Note that in this case, the confidence interval for the difference in means is a bit less wide than the 95% confidence interval generated by the t test, which was (10.34, 21.59). It's common for the bootstrap to produce a narrower range (i.e. an apparently more precise estimate) for the population mean, but it's not automatic that the endpoints from the bootstrap will be inside those provided by the t test, either.

For example, this bootstrap CI doesn't contain the t-test based interval, since its upper bound exceeds that of the t-based interval:

```
set.seed(4310003)
Hmisc::smean.cl.boot(bloodlead$leaddiff, conf.int = 0.95, B = 1000)
```

```
Mean Lower Upper
16.0  11.0  21.8
```

And neither does this one, which actually covers a wider range than the t-based interval.

```
set.seed(4310018)
Hmisc::smean.cl.boot(bloodlead$leaddiff, conf.int = 0.95, B = 1000)
```

```
Mean Lower Upper
16.0  10.3  21.8
```

This demonstration aside, the appropriate thing to do when applying the bootstrap to specify a confidence interval is select a seed and the number (`B` = 1,000 or 10,000, usually) of desired bootstrap replications, then run the bootstrap just once and move on, rather than repeating the process multiple times looking for a particular result.

20.2.1 Assumptions

The bootstrap confidence interval procedure for the population mean (or median) assumes that:

1. We want to estimate the population mean μ (or the population median).
2. We have drawn a sample of observations at random from the population of interest.
3. The sampled observations are drawn from the population independently and have identical distributions.
4. We are willing to put up with the fact that different people (not using the same random seed) will get somewhat different confidence interval estimates using the same data.

As we've seen, a major part of the bootstrap's appeal is the ability to relax some assumptions.

20.3 Wilcoxon Signed Rank-based CI for paired samples

We could also use the Wilcoxon signed rank procedure to generate a CI for the pseudo-median of the paired differences.

```
wilcox.test(bloodlead$leaddiff,
            conf.int = TRUE,
            conf.level = 0.90,
            exact = FALSE)
```

```
    Wilcoxon signed rank test with continuity correction

data:  bloodlead$leaddiff
V = 500, p-value = 1e-05
alternative hypothesis: true location is not equal to 0
90 percent confidence interval:
 11.0 20.5
sample estimates:
(pseudo)median
          15.5
```

As in the one sample case, we can revise this code slightly to specify a different confidence level, or gather a one-sided rather than a two-sided confidence interval.

20.3.1 Assumptions

The Wilcoxon signed rank confidence interval procedure assumes that:

1. We want to estimate the population **median**.
2. We have drawn a sample of observations at random from the population of interest.
3. The sampled observations are drawn from the population independently and have identical distributions.
4. The population follows a symmetric distribution. At the very least, the sample itself shows no substantial skew, so that the sample pseudo-median is a reasonable estimate for the population median.

20.3.2 Using `broom` and the `tidy` function with a Wilcoxon procedure

```
broom::tidy(wilcox.test(bloodlead$leaddiff,
                        conf.int = TRUE,
                        conf.level = 0.90,
                        exact = FALSE))
```

```
  estimate statistic  p.value conf.low conf.high
1     15.5       499 1.15e-05       11      20.5
                                               method alternative
1 Wilcoxon signed rank test with continuity correction   two.sided
```

20.4 Choosing a Confidence Interval Approach

Suppose we want to find a confidence interval for the mean of a population, μ, or, the population mean difference μ_d between two populations based on matched pairs.

1. If we are willing to assume that the population distribution is **Normal**
   - and that the population SD σ is known, we can use a Z-based CI.
   - and the population SD σ **isn't** known, we use a t-based CI.
2. If we are **unwilling** to assume that the population is Normal,
   - use a **bootstrap** procedure to get a CI for the population mean, or even the median
   - but are willing to assume the population is symmetric, consider a **Wilcoxon signed rank** procedure to get a CI for the median, rather than the mean.

The two methods you'll use most often are the bootstrap (especially if the data don't appear to be at least pretty well fit by a Normal model) and the t-based confidence intervals (if the data do appear to fit a Normal model well.)

Chapter 21

Confidence Intervals from Two Independent Samples of Quantitative Data

Here, we'll consider the problem of estimating a confidence interval to describe the difference in population means (or medians) based on a comparison of two samples of quantitative data, gathered using an independent samples design. Specifically, we'll use as our example the randomized controlled trial of Ibuprofen in Sepsis patients, as described in Section @ref(Sepsis_RCT).

In that trial, 300 patients meeting specific criteria (including elevated temperature) for a diagnosis of sepsis were randomly assigned to either the Ibuprofen group (150 patients) and 150 to the Placebo group. Group information (our exposure) is contained in the `treat` variable. The key outcome of interest to us was `temp_drop`, the change in body temperature (in °C) from baseline to 2 hours later, so that positive numbers indicate drops in temperature (a good outcome.)

```
sepsis
```

```
# A tibble: 300 x 6
      id     treat     race apache temp_0 temp_drop
   <chr>    <fctr>   <fctr>  <int>  <dbl>     <dbl>
 1  S002 Ibuprofen AfricanA     14   38.7       1.4
 2  S004 Ibuprofen    White      3   38.3       0.4
 3  S005   Placebo    White      5   38.6       0.0
 4  S006 Ibuprofen    White     13   38.2      -0.2
 5  S009 Ibuprofen    White     25   38.2       0.6
 6  S011 Ibuprofen    White     21   38.1      -0.4
 7  S012   Placebo    White     14   38.6      -0.1
 8  S014   Placebo    White     23   37.9       0.3
 9  S016   Placebo    White     16   38.1       0.1
10  S020 Ibuprofen    Other     20   39.2       1.5
# ... with 290 more rows
```

21.1 t-based CI for population mean difference $mu_1 - \mu_2$ from Independent Samples

21.1.1 The Welch t procedure

The default confidence interval based on the t test for independent samples in R uses something called the Welch test, in which the two populations being compared are not assumed to have the same variance. Each population is assumed to follow a Normal distribution.

```
t.test(sepsis$temp_drop ~ sepsis$treat, conf.level = 0.90, alt = "two.sided")
```

```

    Welch Two Sample t-test

data:  sepsis$temp_drop by sepsis$treat
t = 4, df = 300, p-value = 3e-05
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
 0.191 0.432
sample estimates:
mean in group Ibuprofen   mean in group Placebo
                  0.464                   0.153
```

21.1.2 The Pooled t procedure

The most commonly used t-procedure for building a confidence interval assumes not only that each of the two populations being compared follows a Normal distribution, but also that they have the same population variance. This is the pooled t-test, and it is what people usually mean when they describe a two-sample t test.

```
t.test(sepsis$temp_drop ~ sepsis$treat, conf.level = 0.90, alt = "two.sided", var.equal = TRUE)
```

```

    Two Sample t-test

data:  sepsis$temp_drop by sepsis$treat
t = 4, df = 300, p-value = 3e-05
alternative hypothesis: true difference in means is not equal to 0
90 percent confidence interval:
 0.191 0.432
sample estimates:
mean in group Ibuprofen   mean in group Placebo
                  0.464                   0.153
```

21.1.3 Using linear regression to obtain a pooled t confidence interval

A linear regression model, using the same outcome and predictor (group) as the pooled t procedure, produces the same confidence interval, again, under the assumption that the two populations we are comparing follow a Normal distribution with the same (population) variance.

```
model1 <- lm(temp_drop ~ treat, data = sepsis)
model1
```

```
Call:
lm(formula = temp_drop ~ treat, data = sepsis)

Coefficients:
 (Intercept)  treatPlacebo
       0.464        -0.311
```

```r
confint(model1, level = 0.90)
```

```
                5 %   95 %
(Intercept)   0.379  0.549
treatPlacebo -0.432 -0.191
```

We see that our point estimate from the linear regression model is that the difference in `temp_drop` is -0.311, where Ibuprofen subjects have higher `temp_drop` values than do Placebo subjects, and that the 90% confidence interval for this difference ranges from -0.432 to -0.191.

We can obtain a t-based confidence interval for each of the parameter estimates in a linear model directly using `confint`. Linear models usually summarize only the estimate and standard error. Remember that a reasonable approximation in large samples to a 95% confidence interval for a regression estimate (slope or intercept) can be obtained from estimate $\pm$ 2 * standard error.

```r
summary(model1)
```

```
Call:
lm(formula = temp_drop ~ treat, data = sepsis)

Residuals:
    Min      1Q  Median      3Q     Max
-2.8527 -0.3640 -0.0527  0.3473  2.6360

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.4640     0.0516    8.99  < 2e-16 ***
treatPlacebo  -0.3113     0.0730   -4.27  2.7e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.632 on 298 degrees of freedom
Multiple R-squared:  0.0575,	Adjusted R-squared:  0.0544
F-statistic: 18.2 on 1 and 298 DF,  p-value: 2.68e-05
```

So, in the case of the `treatPlacebo` estimate, we can obtain an approximate 95% confidence interval with -0.311 $\pm$ 2 x 0.073 or (-0.457, -0.165). Compare this to the 95% confidence interval available from the model directly, shown below, and you'll see only a small difference.

```r
confint(model1, level = 0.95)
```

```
              2.5 % 97.5 %
(Intercept)   0.362  0.566
treatPlacebo -0.455 -0.168
```

21.2 Bootstrap CI for $mu_1 - \mu_2$ from Independent Samples

The `bootdif` function contained in the `Love-boost.R` script, that we will use in this setting is a slightly edited version of the function at http://biostat.mc.vanderbilt.edu/wiki/Main/BootstrapMeansSoftware. Note that this approach uses a comma to separate the outcome variable (here, `temp_drop`) from the variable identifying the exposure groups (here, `treat`).

```
set.seed(431212)
bootdif(sepsis$temp_drop, sepsis$treat, conf.level = 0.90)
```

```
Mean Difference            0.05            0.95
         -0.311          -0.431          -0.197
```

21.3 Wilcoxon Rank Sum-based CI from Independent Samples

As in the one-sample case, a rank-based alternative attributed to Wilcoxon (and sometimes to Mann and Whitney) provides a two-sample comparison of the pseudomedians in the two `treat` groups in terms of `temp_drop`. This is called a **rank sum** test, rather than the **signed rank** test for a single sample. Here's the resulting 90% confidence interval.

```
wilcox.test(sepsis$temp_drop ~ sepsis$treat,
            conf.int = TRUE, conf.level = 0.90,
            alt = "two.sided")
```

```

    Wilcoxon rank sum test with continuity correction

data:  sepsis$temp_drop by sepsis$treat
W = 10000, p-value = 7e-06
alternative hypothesis: true location shift is not equal to 0
90 percent confidence interval:
 0.2 0.4
sample estimates:
difference in location
                   0.3
```

21.4 Using the `tidy` function from `broom` for t and Wilcoxon procedures

The `tidy` function is again available to us in dealing with a t-test or Wilcoxon rank sum test.

```
broom::tidy(t.test(sepsis$temp_drop ~ sepsis$treat,
                   conf.level = 0.90,
                   alt = "two.sided"))
```

```
  estimate estimate1 estimate2 statistic  p.value parameter conf.low
1    0.311     0.464     0.153      4.27 2.71e-05       288    0.191
  conf.high                  method alternative
1     0.432 Welch Two Sample t-test   two.sided
```

```
broom::tidy(wilcox.test(sepsis$temp_drop ~ sepsis$treat,
                        conf.int = TRUE,
```

```
                        conf.level = 0.90,
                        alt = "two.sided"))
```

```
  estimate statistic  p.value conf.low conf.high
1      0.3     14614 7.28e-06      0.2       0.4
                                             method alternative
1 Wilcoxon rank sum test with continuity correction   two.sided
```

We can also use `broom` functions to place the elements of the linear model `model1` into a tidy data frame. This provides the estimate of the Placebo-Ibuprofen difference, and its standard error, which we could use to formulate a confidence interval.

```
broom::tidy(model1)
```

```
          term estimate std.error statistic  p.value
1  (Intercept)    0.464    0.0516      8.99 2.91e-17
2 treatPlacebo   -0.311    0.0730     -4.27 2.68e-05
```

```
rm(model1)
```

Chapter 22

Hypothesis Testing of a Population Mean

Hypothesis testing or significance testing uses sample data to attempt to reject the hypothesis that nothing interesting is happening – that is, to reject the notion that chance alone can explain the sample results[1]. We can, in many settings, use confidence intervals to summarize the results, as well, and confidence intervals and hypothesis tests are closely connected. Significance tests have a valuable role to play, but this role is more limited than many scientists realize, and it is unfortunate that tests are widely misused.

In particular, it's worth stressing that:

- **A significant effect is not necessarily the same thing as an interesting effect.** For example, results calculated from large samples are nearly always "significant" even when the effects are quite small in magnitude. Before doing a test, always ask if the effect is large enough to be of any practical interest. If not, why do the test?
- **A non-significant effect is not necessarily the same thing as no difference.** A large effect of real practical interest may still produce a non-significant result simply because the sample is too small.
- **There are assumptions behind all statistical inferences.** Checking assumptions is crucial to validating the inference made by any test or confidence interval.

22.1 Five Steps Required in Completing a Hypothesis Test

1. Specify the null hypothesis, H_0 (which usually indicates that there is no difference or no association between the results in various groups of subjects)
2. Specify the research or alternative hypothesis, H_A, sometimes called H_1 (which usually indicates that there is some difference or some association between the results in those same groups of subjects).
3. Specify the test procedure or test statistic to be used to make inferences to the population based on sample data. Here is where we usually specify α, the probability of incorrectly rejecting H_0 that we are willing to accept. In the absence of other information, we often use $\alpha = 0.05$
4. Obtain the data, and summarize it to obtain the relevant test statistic, which gets summarized as a p value.
5. Use the p value to either
   - **reject** H_0 in favor of the alternative H_A (concluding that there is a statistically significant difference/association at the α significance level) or

[1] Some of this is adapted from @GoodHardin, and @Utts1999

- **retain** H_0 (and conclude that there is no statistically significant difference/association at the α significance level)

22.2 Hypothesis Testing for the Serum Zinc Example

We previously studied serum zinc levels in micrograms per deciliter gathered for a sample of 462 males aged 15-17. "Typical" values are said to be 70-110 μg/dl. Suppose we want to conduct a hypothesis test to see whether our observed zinc values are statistically significantly different from a value we hypothesize might be a reasonable guess for the population as a whole, let's specify **90** μg/dl.

22.2.1 Our Research Question

Is there reasonable evidence, based on this sample of 462 males aged 15-17, for us to conclude that the population of males aged 15-17 from which this sample was drawn will have a mean serum zinc level that is statistically significantly different from 90 μg/dl, the midpoint of the range of "typical" values in the general population?

22.3 Step 1. Specify the null hypothesis

Our population parameter μ = the mean serum zinc level (in μg/dl) across the entire population of males aged 15-17.

- We're testing whether μ is significantly different from a pre-specified value, 90 μg/dl.
- To do this, we apply our pre-specified value in our null hypothesis, so $H_0 : \mu = 90$.

22.4 Step 2. Specify the research hypothesis

The research hypothesis is the opposite of the null hypothesis. Here, that's just $H_A : \mu \neq 90$.

22.5 Step 3. Specify the test procedure

Again, we'll opt for the usual $\alpha = 0.05$. The main procedures for this one-sample setting include three of the four options we used with paired samples, specifically a one-sample t-test, a one-sample Wilcoxon signed rank test, or a bootstrap confidence interval.

- Remember our H_0 specifies $\mu = 90$, rather than $\mu = 0$, as is often the case.

22.6 Step 4. Obtain the *p* value and/or confidence interval

Of course, we've already collected the data. If we're willing to assume the 462 serum zinc levels we have are a random (or sufficiently representative) sample of the population of interest, and that the data were gathered in such a way that each sample is independent of every other sample, and identically distributed, then our methods might work.

22.6.1 Assuming a Normal distribution in the population yields a t test.

```
t.test(serzinc$zinc)
```

```

	One Sample t-test

data:  serzinc$zinc
t = 100, df = 500, p-value <2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 86.5 89.4
sample estimates:
mean of x
     87.9
```

Whoops! This is **WRONG**. Remember that we need to specify that our alternative hypothesis is that the true mean is not equal to 90, not to zero. To change this, we specify our null hypothesis `mu` value in the `t.test` function, as follows…

```
t.test(serzinc$zinc, mu=90)
```

```

	One Sample t-test

data:  serzinc$zinc
t = -3, df = 500, p-value = 0.006
alternative hypothesis: true mean is not equal to 90
95 percent confidence interval:
 86.5 89.4
sample estimates:
mean of x
     87.9
```

You'll note that the only changes here are in the t statistic, p value and alternative hypothesis. The degrees of freedom, confidence interval and sample mean are unchanged.

So the correct p value from the t test would be 0.006, which is less than our pre-specified α of 0.05, and so we'd reject H_0 and conclude that the population mean serum zinc level is statistically significantly different from 90.

- Notice that we would come to the same conclusion using the confidence interval. Specifically, using a 5% significance level (i.e. a 95% confidence level) a reasonable range for the true value of the population mean is entirely below 90 – it's (86.5, 89.4). So if 90 is not in the reasonable range, we'd reject $H_0 : \mu = 90$.

22.6.2 Using `broom` to tidy the results of our t test

```
broom::tidy(t.test(serzinc$zinc, mu=90))
```

```
  estimate statistic p.value parameter conf.low conf.high
1     87.9     -2.77 0.00583       461     86.5      89.4
             method alternative
1 One Sample t-test   two.sided
```

We can use the `tidy` function within the `broom` package to summarize the results of a t test, just as we did with a t-based confidence interval.

22.6.3 Wilcoxon signed rank test (doesn't require Normal assumption).

```
wilcox.test(serzinc$zinc, mu=90, conf.int=TRUE, exact = FALSE)
```

```

    Wilcoxon signed rank test with continuity correction

data:  serzinc$zinc
V = 40000, p-value = 3e-04
alternative hypothesis: true location is not equal to 90
95 percent confidence interval:
 85.5 88.5
sample estimates:
(pseudo)median
            87
```

Using the Wilcoxon signed rank test, we obtain a two-sided p value of 0.0003, which is far less than our pre-specified α of 0.05, so we would, again, reject $H_0 : \mu = 90$.

- Again, the confidence interval suggests that the reasonable range for the population pseudomedian does not contain 90, so we'd reject $H_0 : \mu = 90$ by that standard, too.
- We can again use the `tidy` function from the `broom` package to summarize the results of the Wilcoxon signed rank test.

```
broom::tidy(wilcox.test(serzinc$zinc, mu=90, conf.int = TRUE, exact = FALSE))
```

```
  estimate statistic  p.value conf.low conf.high
1       87     41613 0.000335     85.5      88.5
                                              method alternative
1 Wilcoxon signed rank test with continuity correction   two.sided
```

22.6.4 Bootstrap Confidence Interval

```
set.seed(43123)
Hmisc::smean.cl.boot(serzinc$zinc)
```

```
 Mean Lower Upper
 87.9  86.6  89.4
```

The 95% confidence interval using the bootstrap procedure, again, does not include 90, so we would reject $H_0 : \mu = 90$, in favor of the alternative hypothesis $H_A : \mu \neq 90$.

22.7 Step 5. Reject or Retain H_0 and Draw Conclusions

Using any of these procedures, we would conclude that the null hypothesis (that the true mean serum zinc level for this population is 90 μg/dl) is not tenable, and that it should be rejected at the 5% significance level. The smaller the p value, the stronger is the evidence that the null hypothesis is incorrect, and in this case, we have some fairly tiny p values.

Of course, the confidence intervals suggest that the population mean is reasonably close to 90, and so the difference we can detect (using a fairly large sample of 462 subjects) may not be a clinically meaningful one.

22.8 A One-Sided Test of a Single Sample: What R Reports

Let's walk through a one-sided t test based on a single sample, including a one-sided 90% confidence interval. For instance, suppose we want to test whether the population (of males aged 15-17) has a mean serum zinc level that is statistically significantly **less than** 90 μg/dl, based on the sample of 462 males aged 15-17 that we discussed earlier.

```
t.test(serzinc$zinc, mu = 90, conf = 0.90, alt="less")
```

```

	One Sample t-test

data:  serzinc$zinc
t = -3, df = 500, p-value = 0.003
alternative hypothesis: true mean is less than 90
90 percent confidence interval:
 -Inf 88.9
sample estimates:
mean of x 
     87.9 
```

Here's a brief summary of what R is calculating

1. A specification of the group being studied – here the `zinc` results
2. A specification as to which alternative hypothesis is being tested
    - Note that we are trying to see here if the population mean is less than 90, not 0.
    - here we have a one-sided, specifically a "less than" alternative hypothesis, and it means that we have the following null and alternative hypotheses, $H_0 : \mu \geq 90$ and H_A: $\mu < 90$, where μ = population mean serum zinc level
3. The point estimate (sample mean) of the population mean serum zinc level
    - The sample mean is given as 87.94, so it's at least possible that the true population mean could be less than 90.
4. A 90% confidence interval for the population mean serum zinc level
    - This is a one-sided confidence interval, done with 90% confidence.
    - Since it's one-sided, and we have a "less than" alternative hypothesis, we will only be specifying an upper bound for the population mean.
        - If we had a "greater than" alternative, we would specify a lower bound, instead.
    - The upper bound from a 100(1-α)% one-sided confidence interval for a population mean using the t distribution is $\bar{x} \pm t_{\alpha,n-1}(s/\sqrt{n})$
    - As before, we sample n observations from the population, and $\bar{x}$ = the sample mean, s = the sample standard deviation, and α is the significance level (so that 100[1-α] is the confidence level, and $t_{\alpha,n-1}$ is the upper tail cutoff value for a probability of α for the t distribution with $n-1$ degrees of freedom.
5. R then calculates ...
    - the sample mean of the n = 462 serum zinc levels ($\bar{x} = 87.9372$), and
    - the sample standard deviation of the paired differences (which turns out to be s = 16.0047).
    - In order to find a 90% confidence interval, we would need $\alpha = 0.10$, so we use the appropriate tool in R to find the t cutoff for $\alpha = 0.10$ with appropriate degrees of freedom ($n - 1$ = 462-1 or 461).

```
qt(0.10, 461, lower.tail=FALSE)
```

```
[1] 1.28
```

So $t_{\alpha,n-1} = t_{0.10,461} = 1.283$, and we can now complete the calculation.

$\bar{x} \pm t_{\alpha,n-1}$ (s / $\sqrt{n}$) = 87.9372 + 1.283(16.0047 / $\sqrt{462}$) = 87.9372 + 0.9553 = 88.893

6. A t statistic, degrees of freedom and p value, based on the data that test the null and alternative hypotheses under study
    - Here, this is t = -2.7703, df = 461, p-value = 0.002913.
    - The t statistic again is the sample mean minus the null hypothesized value of the population mean, all divided by the standard error of the sample mean (i.e. the sample standard deviation divided by the square root of the sample size.)
    - Or, in mathematical terms, t=($\bar{x}$ - μ_0)/(s / $\sqrt{n}$) = (87.9372 - 90) / (16.0047 / $\sqrt{462}$) = (-2.0628) / 0.7446 = -2.77.
    - We can interpret the t statistic as the "number of standard errors the sample mean is away from the null hypothesized value of the population mean".
    - The degrees of freedom for a single sample comparison like this is just the number of observations minus 1. Here, we have 462 serum zinc results; 461 degrees of freedom.
    - Given the test statistic, t = -2.77, and the degrees of freedom n-1 = 461, R can now calculate a p value, specifically the probability (given that H_0 is true) of observing a result as much in favor of the alternative hypothesis HA as these data suggest.
    - We want a one-sided p value here, since we have a one-sided alternative hypothesis (i.e. a "less than" alternative).
    - Find the probability of getting a result this small or smaller (since we have a "less than" alternative, if it was a "greater than" alternative, we'd find the probability of a result this large or larger) as follows…

```
pt(-2.77, df=461, lower.tail=TRUE)
```

```
[1] 0.00292
```

Chapter 23

Type I and Type II Error: Power and Confidence

Once we know how unlikely the results would have been if the null hypothesis were true, we must make one of two choices:

1. The p value is not small enough to convincingly rule out chance. Therefore, we cannot reject the null hypothesis as an explanation for the results.
2. The p value was small enough to convincingly rule out chance. We reject the null hypothesis and accept the alternative hypothesis.

Making choice 2 is equivalent to declaring that the result is statistically significant. We can rephrase the two choices as:

1. There is no statistically significant difference or relationship in the data.
2. There is a statistically significant difference or relationship in the data.

How small must the p value be in order to rule out the null hypothesis? The standard choice is 5%. This standardization has advantages and disadvantages[1], and it is not compulsory. It is simply a convention that has become accepted over the years, and there are many situations for which a 5% cutoff may be unwise. While it does give a specific, objectively chosen level to keep in mind, it suggests a rather mindless cutpoint having nothing to do with the importance of the decision nor the costs or losses associated with outcomes.

23.1 The Courtroom Analogy

Consider the analogy of the jury in a courtroom.

1. The evidence is not strong enough to convincingly rule out that the defendant is innocent. Therefore, we cannot reject the null hypothesis, or innocence of the defendant.
2. The evidence was strong enough that we are willing to rule out the possibility that an innocent person (as stated in the null hypothesis) produced the observed data. We reject the null hypothesis, that the defendant is innocent, and assert the alternative hypothesis.

Consistent with our thinking in hypothesis testing, in many cases we would not accept the hypothesis that the defendant is innocent. We would simply conclude that the evidence was not strong enough to rule out the possibility of innocence.

[1] Ingelfinger JA, Mosteller F, Thibodeau LA and Ware JH (1987) Biostatistics in Clinical Medicine, 2nd Edition, New York: MacMillan. pp. 156-157.

The p value is the probability of getting a result as extreme or more extreme than the one observed if the proposed null hypothesis is true. Notice that it is not valid to actually accept that the null hypothesis is true. To do so would be to say that we are essentially convinced that chance alone produced the observed results – a common mistake.

23.2 Significance vs. Importance

Remember that a statistically significant relationship or difference does not necessarily mean an important one. A result that is significant in the statistical meaning of the word may not be significant clinically. Statistical significance is a technical term. Findings can be both statistically significant and practically significant or either or neither.

When we have very large samples, we may find small differences statistically significant even though they have no clinical importance. At the other extreme, with small samples, even large differences will often not be significant at the levels usually required to recognize the difference as real. We must distinguish between statistical and practical/clinical significance.

23.3 Errors in Hypothesis Testing

In testing hypotheses, there are two potential decisions and each one brings with it the possibility that a mistake has been made.

Let's use the courtroom analogy. Here are the potential choices and associated potential errors. Although the seriousness of errors depends on the seriousness of the crime and punishment, the potential error for choice 2 is usually more serious.

1. We cannot rule out that the defendant is innocent, so (s)he is set free without penalty.
    - Potential Error: A criminal has been erroneously freed.
2. We believe that there is enough evidence to conclude that the defendant is guilty.
    - Potential Error: An innocent person is convicted / penalized and a guilty person remains free.

As another example, consider being tested for disease. Most tests for diseases are not 100% accurate. The lab technician or physician must make a choice:

1. In the opinion of the medical practitioner, you are healthy. The test result was weak enough to be called "negative" for the disease.
    - Potential Error: You are actually diseased but have been told you are not. This is called a **false negative**.
2. In the opinion of the medical practitioner, you are diseased. The test results were strong enough to be called "positive" for the disease.
    - Potential Error: You are actually healthy but have been told you are diseased. This is called a **false positive**.

23.4 The Two Types of Hypothesis Testing Errors

–	H_A is true	H_0 is true
Test Rejects H_0	Correct Decision	Type I Error (False Positive)
Test Retains H_0	Type II Error (False Negative)	Correct Decision

- A Type I error can only be made if the null hypothesis is actually true.

- A Type II error can only be made if the alternative hypothesis is actually true.

23.5 The Significance Level, α, is the Probability of a Type I Error

If the null hypothesis is true, the p value is the probability of making an error by choosing the alternative hypothesis instead. Alpha (α) is defined as the probability of concluding significance [rejection of H_0] when there isn't (and H_0 is true, making a Type I error), also called the significance level, so that 100(1-α) is the confidence level – the probability of correctly concluding that there is no difference (retaining H_0) when H_0 is true.

23.6 The Probability of avoiding a Type I Error is called Power, symbolized 1-β

A Type II error is made if the alternative hypothesis is true, but you fail to choose it. The probability depends on exactly which part of the alternative hypothesis is true, so that computing the probability of making a Type II error is not feasible. The power of a test is the probability of making the correct decision when the alternative hypothesis is true. Beta (β) is defined as the probability of concluding that there was no difference, when in fact there was one (a Type II error). Power is then just 1 - β, the probability of concluding that there was a difference, when, in fact, there was one.

Traditionally, people like the power of a test to be at least 80%, meaning that β is at most 0.20. Often, I'll be arguing for 90% as a minimum power requirement, or we'll be presenting a range of power calculations for a variety of sample size choices.

23.7 Incorporating the Costs of Various Types of Errors

Which error is more serious in medical testing, where we think of our H_0: patient is healthy vs. H_A: disease is present?

It depends on the disease and on the consequences of a negative or positive test result. A false negative in a screening test for cancer could lead to a fatal delay in treatment, whereas a false positive would probably lead to a retest. A more troublesome example occurs in testing for an infectious disease. Inevitably, there is a trade-off between the two types of errors. It all depends on the consequences.

It would be nice if we could specify the probability that we were making an error with each potential decision. We could then weigh the consequence of the error against its probability. Unfortunately, in most cases, we can only specify the conditional probability of making a Type I error, given that the null hypothesis is true.

In deciding whether to reject a null hypothesis, we will need consider the consequences of the two potential types of errors. If a Type I error is very serious, then you should reject the null hypothesis only if the p value is very small. Conversely, if a Type II error is more serious, you should be willing to reject the null hypothesis with a larger p value, perhaps 0.10 or 0.20, instead of 0.05.

23.8 Relation of α and β to Error Types

- α is the probability of rejecting H_0 when H_0 is true.
  - So $1 - \alpha$, the confidence level, is the probability of retaining H_0 when that's the right thing to do.
- β is the probability of retaining H_0 when H_A is true.
  - So $1 - \beta$, the power, is the probability of rejecting H_0 when that's the right thing to do.

–	H_A is True	H_0 is True
Test Rejects H_0	Correct Decision (1 - β)	Type I Error (α)
Test Retains H_0	Type II Error (β)	Correct Decision (1 - α)

23.9 Power and Sample Size Calculations

- For most statistical tests, it is theoretically possible to estimate the power of the test in the design stage, (before any data are collected) for various sample sizes, so we can hone in on a sample size choice which will enable us to collect data only on as many subjects are truly necessary.
- A power calculation is likely the most common element of an scientific grant proposal on which a statistician is consulted. This is a fine idea in theory, but in practice…
- The tests that have power calculations worked out in intensive detail using R are mostly those with more substantial assumptions. Examples include t tests that assume population normality, common population variance and balanced designs in the independent samples setting, or paired t tests that assume population normality in the paired samples setting.
- These power calculations are also usually based on tests rather than confidence intervals, which would be much more useful in most settings. Simulation is your friend here.
- Even more unfortunately, this process of doing power and related calculations is **far more of an art than a science**.
- As a result, the value of many power calculations is negligible, since the assumptions being made are so arbitrary and poorly connected to real data.
- On several occasions, I have stood in front of a large audience of medical statisticians actively engaged in clinical trials and other studies that require power calculations for funding. When I ask for a show of hands of people who have had power calculations prior to such a study whose assumptions matched the eventual data perfectly, I get lots of laughs. It doesn't happen.
- Even the underlying framework that assumes a power of 80% with a significance level of 5% is sufficient for most studies is pretty silly.

All that said, I feel obliged to show you some examples of power calculations done using R, and provide some insight on how to make some of the key assumptions in a way that won't alert reviewers too much to the silliness of the enterprise.

23.10 Sample Size and Power Considerations for a Single-Sample t test

For a t test, R can estimate any one of the following elements, given the other four, using the `power.t.test` command, for either a one-tailed or two-tailed single-sample t test…

- n = the sample size
- δ = delta = the true difference in population means between the null hypothesis value and a particular alternative
- s = sd = the true standard deviation of the population
- α = sig.level = the significance level for the test (maximum acceptable risk of Type I error)
- 1 - β = power = the power of the t test to detect the effect of size δ

23.10.1 A Toy Example

Suppose that in a recent health survey, the average beef consumption in the U.S. per person was 90 pounds per year. Suppose you are planning a new study to see if beef consumption levels have changed. You plan to take a random sample of 25 people to build your new estimate, and test whether the current pounds of beef consumed per year is 90. Suppose you want to do a two-sided (two-tailed) test at 95% confidence (so $\alpha = 0.05$), and that you expect that the true difference will need to be at least $\delta = 5$ pounds (i.e. 85 or less or 95 or more) in order for the result to be of any real, practical interest. Suppose also that you are willing to assume that the true standard deviation of the measurements in the population is 10 pounds.

That is, of course, a lot to suppose.

Now, we want to know what power the proposed experiment will have to detect a change of 5 pounds (or more) away from the original 90 pounds, with these specifications, and how tweaking these specifications will affect the power of the study.

So, we have - n = 25 data points to be collected - $\delta = 5$ pounds is the minimum clinically meaningful effect size - s = 10 is the assumed population standard deviation, in pounds per year - α is 0.05, and we'll do a two-sided test

23.10.2 Using the `power.t.test` function

```r
power.t.test(n = 25, delta = 5, sd = 10, sig.level = 0.05,
             type="one.sample", alternative="two.sided")
```

```

     One-sample t test power calculation

              n = 25
          delta = 5
             sd = 10
      sig.level = 0.05
          power = 0.67
    alternative = two.sided
```

So, under this study design, we would expect to detect an effect of size $\delta = 5$ pounds with just under 67% power, i.e. with a probability of incorrect retention of H_0 of just about 1/3. Most of the time, we'd like to improve this power, and to do so, we'd need to adjust our assumptions.

23.10.3 Changing Assumptions in a Power Calculation

We made assumptions about the sample size n, the minimum clinically meaningful effect size (change in the population mean) δ, the population standard deviation s, and the significance level α, not to mention decisions about the test, like that we'd do a one-sample t test, rather than another sort of test for a single sample, and that we'd do a two-tailed, or two-sided test. Often, these assumptions are tweaked a bit to make the power look more like what a reviewer/funder is hoping to see.

23.10.4 Increasing the Sample Size, absent other changes, will Increase the Power

Suppose, we committed to using more resources and gathering data from 40 subjects instead of the 25 we assumed initially – what effect would this have on our power?

```
power.t.test(n = 40, delta = 5, sd = 10, sig.level = 0.05,
             type="one.sample", alternative="two.sided")
```

```

     One-sample t test power calculation

              n = 40
          delta = 5
             sd = 10
      sig.level = 0.05
          power = 0.869
    alternative = two.sided
```

With more samples, we should have a more powerful test, able to detect the difference with greater probability. In fact, a sample of 40 paired differences yields 87% power. As it turns out, we would need at least 44 observations with this scenario to get to 90% power, as shown in the calculation below, which puts the power in, but leaves out the sample size.

```
power.t.test(power=0.9, delta = 5, sd = 10, sig.level = 0.05,
             type="one.sample", alternative="two.sided")
```

```

     One-sample t test power calculation

              n = 44
          delta = 5
             sd = 10
      sig.level = 0.05
          power = 0.9
    alternative = two.sided
```

We see that we would need at least 44 observations to achieve 90% power. Note: we always round the sample size up in doing a power calculation – if this calculation had actually suggested n = 43.1 paired differences were needed, we would still have rounded up to 44.

23.10.5 Increasing the Effect Size, absent other changes, will increase the Power

A larger effect should be easier to detect. If we go back to our original calculation, which had 67% power to detect an effect of size $\delta = 5$, and now change the desired effect size to $\delta = 6$ pounds (i.e. a value of 84 or less or 96 or more), we should obtain a more powerful design.

```
power.t.test(n = 25, delta = 6, sd = 10, sig.level = 0.05,
             type="one.sample", alternative="two.sided")
```

```

     One-sample t test power calculation

              n = 25
          delta = 6
             sd = 10
      sig.level = 0.05
          power = 0.821
    alternative = two.sided
```

We see that this change in effect size from 5 to 6, leaving everything else the same, increases our power from 67% to 82%. To reach 90% power, we'd need to increase the effect size we were trying to detect to at least

6.76 pounds.

```r
power.t.test(n = 25, power = 0.9, sd = 10, sig.level = 0.05,
             type="one.sample", alternative="two.sided")
```

```

     One-sample t test power calculation 

              n = 25
          delta = 6.76
             sd = 10
      sig.level = 0.05
          power = 0.9
    alternative = two.sided
```

- Again, note that I am rounding up here.
- Using $\delta = 6.75$ would not quite make it to 90.00% power.
- Using $\delta = 6.76$ guarantees that the power will be 90% or more, and not just round up to 90%..

23.10.6 Decreasing the Standard Deviation, absent other changes, will increase the Power

The choice of standard deviation is usually motivated by a pilot study, or else pulled out of thin air - it's relatively easy to convince yourself that the true standard deviation might be a little smaller than you'd guessed initially. Let's see what happens to the power if we reduce the sample standard deviation from 10 pounds to 9. This should make the effect of 5 pounds easier to detect, because it will have smaller variation associated with it.

```r
power.t.test(n = 25, delta = 5, sd = 9, sig.level = 0.05,
             type="one.sample", alternative="two.sided")
```

```

     One-sample t test power calculation 

              n = 25
          delta = 5
             sd = 9
      sig.level = 0.05
          power = 0.76
    alternative = two.sided
```

This change in standard deviation from 10 to 9, leaving everything else the same, increases our power from 67% to nearly 76%. To reach 90% power, we'd need to decrease the standard deviation of the population paired differences to no more than 7.39 days.

```r
power.t.test(n = 25, delta = 5, sd = NULL, power = 0.9, sig.level = 0.05,
             type="one.sample", alternative="two.sided")
```

```

     One-sample t test power calculation 

              n = 25
          delta = 5
             sd = 7.4
      sig.level = 0.05
          power = 0.9
```

```
    alternative = two.sided
```

Note I am rounding down here.

- Using s = 7.4 days would not quite make it to 90.00% power.

Note also that in order to get R to treat the sd as unknown, I must specify it as NULL in the formula…

23.10.7 Tolerating a Larger α (Significance Level), without other changes, increases Power

We can trade off some of our Type II error (lack of power) for Type I error. If we are willing to trade off some Type I error (as described by the α), we can improve the power. For instance, suppose we decided to run the original test with 90% confidence.

```
power.t.test(n = 25, delta = 5, sd = 10, sig.level = 0.1,
             type="one.sample", alternative="two.sided")
```

```

     One-sample t test power calculation

              n = 25
          delta = 5
             sd = 10
      sig.level = 0.1
          power = 0.783
    alternative = two.sided
```

The calculation suggests that our power would thus increase from 67% to just over 78%.

Chapter 24

Comparing Two Means Using Paired Samples

In this section, we apply several methods of testing the null hypothesis that two populations have the same distribution of a quantitative variable. In particular, we'll focus on the comparison of means using paired sample t tests, signed rank tests, and bootstrap approaches. Our example comes from the Lead in the Blood of Children study, described in Section 17 and the methods outlined in Section 20.

Recall that in that study, we measured blood lead content, in mg/dl, for 33 matched pairs of children, one of which was exposed (had a parent working in a battery factory) and the other of which was control (no parent in the battery factory, but matched to the exposed child by age, exposure to traffic and neighborhood). We then created a variable called `leaddiff` which contained the (exposed - control) differences within each pair.

24.1 Specifying A Two-Sample Study Design

These questions will help specify the details of the study design involved in any comparison of means.

1. What is the outcome under study?
2. What are the (in this case, two) treatment/exposure groups?
3. Were the data collected using matched / paired samples or independent samples?
4. Are the data a random sample from the population(s) of interest? Or is there at least a reasonable argument for generalizing from the sample to the population(s)?
5. What is the significance level (or, the confidence level) we require here?
6. Are we doing one-sided or two-sided testing/confidence interval generation?
7. If we have paired samples, did pairing help reduce nuisance variation?
8. If we have paired samples, what does the distribution of sample paired differences tell us about which inferential procedure to use?

24.1.1 For the `bloodlead` study

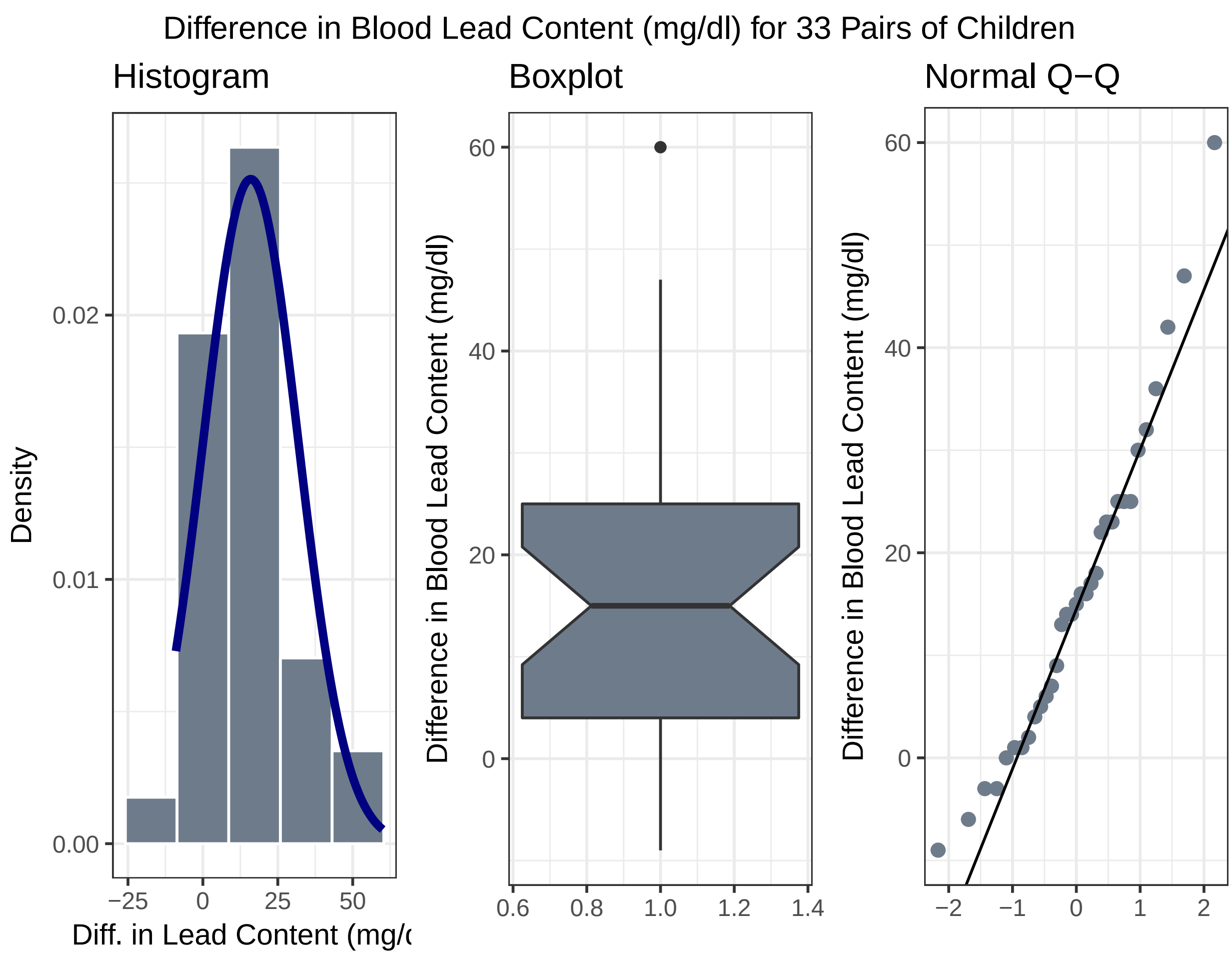
1. The outcome is blood lead content in mg/dl.
2. The groups are **exposed** (had a parent working in a battery factory) and **control** (no parent in the battery factory, but matched to the exposed child by age, exposure to traffic and neighborhood) children.
3. The data were collected using matched samples. The pairs of subjects are matched by age, exposure to traffic and neighborhood.
4. The data aren't a random sample of the population of interest, but we will assume for now that there's no serious issue with representing that population.

5. We'll use a 10% significance level (or 90% confidence level) in this setting.
6. We'll use a two-sided testing and confidence interval approach.

To answer question 7 (did pairing help reduce nuisance variation), we return to Section 17 where we saw that:

- The stronger the Pearson correlation coefficient, the more helpful pairing will be.
- Here, a straight line model using the control child's blood lead content accounts for about 3% of the variation in blood lead content in the exposed child.
- So, as it turns out, pairing will have only a modest impact here on the inferences we draw in the study.

To address question 8, we'll need to look at the data - specifically the paired differences. Repeating a panel of plots from Section 17, we will see that the paired differences appear to follow a Normal distribution, at least approximately (there's a single outlier, but with 33 pairs, that's not much of a concern, and the data are basically symmetric), so that a t-based procedure may be appropriate, or, at least, we'd expect to see similar results with t and bootstrap procedures.



24.2 Hypothesis Testing for the Blood Lead Example

24.2.1 Our Research Question

Is there reasonable evidence, based on these paired sample of 33 exposed and 33 control children, for us to conclude that the population of children similar to those in the exposed group will have a distribution of blood lead content that is statistically significantly different from the population of children similar to those in the control group. In other words, if we generated the population of all exposed-control differences across the entire population of such pairs, would that distribution of paired differences be centered at zero (indicating no difference in the means)?

Again, the key idea is to calculate the paired differences (exposed - control, for example) in each pair, and then treat the result as if it were a single sample and apply the methods discussed in Section 22.

24.2.2 Specify the null hypothesis

Our null hypothesis here is that the population (true) mean blood lead content in the exposed group is the same as the population mean blood lead content in the control group plus a constant value (which we'll symbolize with Δ_0 which is most often taken to be zero. Since we have paired samples, we can instead describe this hypothesis in terms of the difference between exposed and control within each pair. So, our null hypothesis can be written either as:

$$H_0 : \mu_{Exposed} = \mu_{Control} + \Delta_0$$

where Δ_0 is a constant, usually taken to be zero, or

$$H_0 : \mu_{Exposed-Control} = \Delta_0,$$

where, again, Δ_0 is usually zero.

We will generally take this latter approach, where the population mean of the paired differences (here, exposed - control, but we could have just as easily selected control - exposed: the order is arbitrary so long as we are consistent) is compared to a constant value, usually 0.

For the `bloodlead` example, our population parameter $\mu_{Exposed-Control}$ = the mean difference in blood lead content between the exposed and control groups (in mg/dl) across the entire population.

- We're testing whether μ is significantly different from a pre-specified value, 0 mg/dl.

24.2.3 Specify the research hypothesis

The research hypothesis is that the population mean of the exposed - control differences is not equal to our constant value Δ_0.

$$H_A : \mu_{Exposed-Control} \neq \Delta_0,$$

For the `bloodlead` example, we have $H_A : \mu_{Exposed-Control} \neq 0$.

24.2.4 Specify the test procedure and α

As we've seen in Section 20, there are several ways to build a confidence interval to address these hypotheses, and each of those approaches provides information about a related hypothesis test. This includes several methods for obtaining a paired t test, plus a Wilcoxon signed rank test, and a bootstrap comparison of means (or medians, etc.) using paired samples. We'll specify an α value of .10 here, indicating a 10% significance level (and 90% confidence level.)

24.2.5 Calculate the test statistic and p value

For the paired t test and Wilcoxon signed rank test, Section 20 demonstrated the relevant R code for the `bloodlead` example to obtain p values. For the bootstrap procedure, we again build a confidence interval. We repeat that work below.

Of course, we've already collected the data. If we're willing to assume the 462 serum zinc levels we have are a random (or sufficiently representative) sample of the population of interest, and that the data were gathered in such a way that each sample is independent of every other sample, and identically distributed, then our methods might work.

24.2.6 Draw a conclusion

As we've seen, we use the p value to either

- **reject** H_0 in favor of the alternative H_A (concluding that there is a statistically significant difference/association at the α significance level) if the p value is less than our desired α or
- **retain** H_0 (and conclude that there is no statistically significant difference/association at the α significance level) if the p value is greater than or equal to α.

24.3 Assuming a Normal distribution in the population of paired differences yields a paired t test.

```
t.test(bloodlead$exposed - bloodlead$control, conf = 0.90, alt = "two.sided")
```

```

	One Sample t-test

data:  bloodlead$exposed - bloodlead$control
t = 6, df = 30, p-value = 2e-06
alternative hypothesis: true mean is not equal to 0
90 percent confidence interval:
 11.3 20.6
sample estimates:
mean of x 
       16 
```

The t test statistic here is 6, based on 30 degrees of freedom, and this yields a p value of `2.036e-06` or 2.036×10^{-6}, or $p = .000002036$. This p value is certainly less than our pre-specified α of 0.10, and so we'd reject H_0 and conclude that the population mean of the exposed-control paired differences is statistically significantly different from 0.

- Notice that we would come to the same conclusion using the confidence interval. Specifically, using a 10% significance level (i.e. a 90% confidence level) a reasonable range for the true value of the population mean is entirely above 0 – it's (11.3, 20.6). So if 0 is not in the 90% confidence interval, we'd reject $H_0 : \mu_{Exposed-Control} = 0$ at the 10% significance level.

24.3.1 Assumptions of the paired t test

We must be willing to believe that

1. the paired differences data are a random (or failing that, representative) sample from the population of interest, and
2. that the samples were drawn independently, from an identical population distribution

regardless of what testing procedure we use. For the paired t test, we must also assume that:

3. the paired differences come from a Normally distributed population.

24.3.2 Using `broom` to tidy the paired t test

```
broom::tidy(t.test(bloodlead$exposed - bloodlead$control,
                   conf = 0.90, alt = "two.sided"))
```

```
  estimate statistic  p.value parameter conf.low conf.high
1       16      5.78 2.04e-06        32     11.3      20.6
             method alternative
1 One Sample t-test   two.sided
```

We can use the `tidy` function within the `broom` package to summarize the results of a t test, just as we did with a t-based confidence interval.

24.3.3 Calculation Details: The Paired t test

The paired t test is calculated using:

- $\bar{d}$, the sample mean of the paired differences,
- the null hypothesized value Δ_0 for the differences (which is usually 0),
- s_d, the sample standard deviation, and
- n, the sample size (number of pairs).

We have

$$t = \frac{\bar{d} - \Delta_0}{s_d/\sqrt{n}}$$

which is then compared to a t distribution with $n - 1$ degrees of freedom to obtain a p value.

Wikipedia's page on Student's t test is a good resource for these calculations.

24.4 The Bootstrap Approach: Build a Confidence Interval

The same bootstrap approach is used for paired differences as for a single sample. We again use the `smean.cl.boot()` function in the `Hmisc` package to obtain bootstrap confidence intervals for the population mean, μ_d, of the paired differences in blood lead content.

```
set.seed(431888)
Hmisc::smean.cl.boot(bloodlead$exposed - bloodlead$control, conf.int = 0.90, B = 1000)
```

```
 Mean Lower Upper
 16.0  11.6  20.6
```

Since 0 is not contained in this 90% confidence interval, we reject the null hypothesis (that the mean of the paired differences in the population is zero) at the 10% significance level, so we know that $p < 0.10$.

24.4.1 Assumptions of the paired samples bootstrap procedure

We still must be willing to believe that

1. the paired differences data are a random (or failing that, representative) sample from the population of interest, and
2. that the samples were drawn independently, from an identical population distribution

regardless of what testing procedure we use. But, for the bootstrap, we do not also need to assume Normality of the population distribution of paired differences.

24.5 The Wilcoxon signed rank test (doesn't require Normal assumption).

We could also use the Wilcoxon signed rank procedure to generate a CI for the pseudo-median of the paired differences.

```
wilcox.test(bloodlead$leaddiff,
            conf.int = TRUE,
            conf.level = 0.90,
            exact = FALSE)
```

```

    Wilcoxon signed rank test with continuity correction

data:  bloodlead$leaddiff
V = 500, p-value = 1e-05
alternative hypothesis: true location is not equal to 0
90 percent confidence interval:
 11.0 20.5
sample estimates:
(pseudo)median
          15.5
```

Using the Wilcoxon signed rank test, we obtain a two-sided p value of 1.155 $\times 10^{-5}$, which is far less than our pre-specified α of 0.10, so we would, again, reject $H_0 : \mu_{Exposed-Control} = 0$ at the 10% significance level.

- We can again use the `tidy` function from the `broom` package to summarize the results of the Wilcoxon signed rank test.

```
broom::tidy(wilcox.test(bloodlead$leaddiff, conf.int = TRUE,
                        conf.level = 0.90, exact = FALSE))
```

```
  estimate statistic  p.value conf.low conf.high
1     15.5       499 1.15e-05       11      20.5
                                                method alternative
1 Wilcoxon signed rank test with continuity correction   two.sided
```

24.5.1 Assumptions of the Wilcoxon Signed Rank procedure

We still must be willing to believe that

1. the paired differences data are a random (or failing that, representative) sample from the population of interest, and
2. that the samples were drawn independently, from an identical population distribution

regardless of what testing procedure we use. But, for the Wilcoxon signed rank test, we also assume

3. that the population distribution of the paired differences is symmetric, but potentially outlier-prone.

24.5.2 Calculation Details: The Wilcoxon Signed Rank test

- Calculate the paired difference for each pair, and drop those with difference = 0.
- Let N be the number of pairs, so there are 2N data points.
- Rank the pairs in order of smallest (rank = 1) to largest (rank = N) absolute difference.
- Calculate W, the sum of the signed ranks by

$$W = \sum_{i=1}^{N} [sgn(x_{2,i} - x_{1,i})] \prod R_i)]$$

- The sign function sgn(x) = -1 if x < 0, 0 if x = 0, and +1 if x > 0.
- Statistical software will convert W into a p value, given N.

Wikipedia's page on the Wilcoxon signed-rank test is a good resource for example calculations.

24.6 Step 5. Reject or Retain H_0 and Draw Conclusions

Using any of these procedures, we would conclude that the null hypothesis (that the true mean of the paired differences is 0 mg/dl) is not tenable, and that it should be rejected at the 10% significance level. The smaller the p value, the stronger is the evidence that the null hypothesis is incorrect, and in this case, we have some fairly tiny p values.

Procedure	p value	90% CI for $\mu_{Exposed-Control}$	Conclusion
Paired t	2×10^{-6}	11.3, 20.6	Reject H_0.
Wilcoxon signed rank	1×10^{-5}	11, 20.5	Reject H_0.
Bootstrap CI	$p < 0.10$	11.6, 20.6	Reject H_0.

Note that **one-sided** or **one-tailed** hypothesis testing procedures work the same way for paired samples as they did for single samples in Section **??**Test-One-Mean).

24.7 The Sign test

The **sign test** is something we've skipped in our discussion so far. It is a test for consistent differences between pairs of observations, just as the paired t test, Wilcoxon signed rank test and bootstrap for paired samples can provide. It has the advantage that it is relatively easy to calculate by hand, and that it doesn't require the paired differences to follow a Normal distribution. In fact, it will even work if the data are substantially skewed.

- Calculate the paired difference for each pair, and drop those with difference = 0.
- Let N be the number of pairs that remain, so there are 2N data points.
- Let W, the test statistic, be the number of pairs (out of N) in which the difference is positive.
- Assuming that H_0 is true, then W follows a binomial distribution with probability 0.5 on N trials.

For example, consider our data on blood lead content:

```
bloodlead$leaddiff
```

```
 [1] 22  5 23 -6 18 25 13 47 15 16  6  1  2  7  0  4 -9 -3 36 25  1 16 42
[24] 30 25 23 32 17  9 -3 60 14 14
```

Difference	# of Pairs
Greater than zero	28
Equal to zero	1
Less than zero	4

So we have $N = 32$ pairs, with $W = 28$ that are positive. We can calculate the p value using the `binom.test` approach in R:

```
binom.test(x = 28, n = 32, p = 0.5, alternative = "two.sided")
```

```

    Exact binomial test

data:  28 and 32
number of successes = 30, number of trials = 30, p-value = 2e-05
alternative hypothesis: true probability of success is not equal to 0.5
95 percent confidence interval:
 0.710 0.965
sample estimates:
probability of success
                 0.875
```

24.8 Building a Decision Support Tool: Comparing Means

1. Are these paired or independent samples?

2. If paired samples, then are the paired differences approximately Normally distributed?

   a. If yes, then a paired t test or confidence interval is likely the best choice.
   b. If no, is the main concern outliers (with generally symmetric data), or skew?
      1. If the paired differences appear to be generally symmetric but with substantial outliers, a Wilcoxon signed rank test is an appropriate choice, as is a bootstrap confidence interval for the population mean of the paired differences.
      2. If the paired differences appear to be seriously skewed, then we'll usually build a bootstrap confidence interval, although a sign test is another reasonable possibility.

Baumer, Benjamin S., Daniel T. Kaplan, and Nicholas J. Horton. 2017. *Modern Data Science with R*. Boca Raton, FL: CRC Press. https://mdsr-book.github.io/.

Bernard, Gordon R., Arthur P. Wheeler, James A. Russell, Roland Schein, Warren R. Summer, Kenneth P. Steinberg, William J. Fulkerson, et al. 1997. "The Effects of Ibuprofen on the Physiology and Survival of Patients with Sepsis." *New England Journal of Medicine* 336: 912–18. http://www.nejm.org/doi/full/10.

1056/NEJM199703273361303#t=article.

Bock, David E., Paul F. Velleman, and Richard D. De Veaux. 2004. *Stats: Modelling the World*. Boston MA: Pearson Addison-Wesley.

Çetinkaya-Rundel, Mine. 2017. "Teaching Data Science to New useRs." bit.ly/user2017.

Diez, David M., Christopher D. Barr, and Mine Çetinyaka-Rundel. n.d. *OpenIntro Statistics*. Third. https://www.openintro.org/stat/textbook.php?stat_book=os.

Dupont, William D. 2002. *Statistical Modeling for Biomedical Researchers*. New York: Cambridge University Press.

Efron, Bradley. 1979. "Bootstrap Methods: Another Look at the Jackknife." *Annals of Statistics* 7(1): 1–26. https://projecteuclid.org/euclid.aos/1176344552.

Fox, John, and Sanford Weisberg. 2011. *An R Companion to Applied Regression*. Second. Thousand Oaks CA: Sage. http://socserv.socsci.mcmaster.ca/jfox/Books/Companion.

Gelman, Andrew, and Jennifer Hill. 2007. *Data Analysis Using Regression and Multilevel-Hierarchical Models*. New York: Cambridge University Press. http://www.stat.columbia.edu/~gelman/arm/.

Gelman, Andrew, and Deborah Nolan. 2017. *Teaching Statistics: A Bag of Tricks*. Second. Oxford, UK: Oxford University Press.

Good, Phillip I. 2005. *Introduction to Statistics Through Resampling Methods and R/S-Plus*. Hoboken, NJ: Wiley.

Good, Phillip I., and James W. Hardin. 2006. *Common Errors in Statistics (and How to Avoid Them)*. Second. Hoboken, NJ: Wiley.

Grolemund, Garrett, and Hadley Wickham. 2017. *R for Data Science*. O'Reilly. http://r4ds.had.co.nz/.

Harrell, Frank E., and James C. Slaughter. 2017. *Biostatistics for Biomedical Research*. Vanderbilt University School of Medicine. biostat.mc.vanderbilt.edu/ClinStat.

Ismay, Chester, and Albert Y. Kim. 2017. *ModernDive: An Introduction to Statistical and Data Sciences via R*. http://moderndive.com/.

Morton, D., A. Saah, S. Silberg, W. Owens, M. Roberts, and M. Saah. 1982. "Lead Absorption in Children of Employees in a Lead Related Industry." *American Journal of Epidemiology* 115: 549–55.

Norman, Geoffrey R., and David L. Streiner. 2014. *Biostatistics: The Bare Essentials*. Fourth. People's Medical Publishing House.

Pagano, Marcello, and Kimberlee Gauvreau. 2000. *Principles of Biostatistics*. Second. Duxbury Press.

Pruzek, Robert M., and James E. Helmreich. 2009. "Enhancing Dependent Sample Analyses with Graphics." *Journal of Statistics Education* 17(1). http://ww2.amstat.org/publications/jse/v17n1/helmreich.html.

Ramsey, Fred L., and Daniel W. Schafer. 2002. *The Statistical Sleuth: A Course in Methods of Data Analysis*. Second. Pacific Grove, CA: Duxbury.

Vittinghoff, Eric, David V. Glidden, Stephen C. Shiboski, and Charles E. McCulloch. 2012. *Regression Methods in Biostatistics: Linear, Logistic, Survival, and Repeated Measures Models*. Second. Springer-Verlag, Inc. http://www.biostat.ucsf.edu/vgsm/.

Wainer, Howard. 1997. *Visual Revelations: Graphical Tales of Fate and Deception from Napoleon Bonaparte to Ross Perot*. New York: Springer-Verlag.

———. 2005. *Graphic Discovery: A Trout in the Milk and Other Visual Adventures*. Princeton, NJ: Princeton University Press.

———. 2013. *Medical Illuminations: Using Evidence, Visualization and Statistical Thinking to Improve Healthcare*. New York: Oxford University Press.

Yamada, SB, and EG Boulding. 1998. "Claw Morphology, Prey Size Selection and Foraging Efficiency in Generalist and Specialist Shell-Breaking Crabs." *Journal of Experimental Marine Biology and Ecology* 220: 191–211. http://www.science.oregonstate.edu/~yamadas/SylviaCV/BehrensYamada_Boulding1998.pdf.