Data Science for Biological, Medical and Health Research: Notes for PQHS/CRSP/MPHP 431

Thomas E. Love

2020-08-25

Contents

Working with These Notes

1. This document is broken down into multiple chapters. Use the table of contents on the left side of the screen to navigate, and use the hamburger icon (horizontal bars) at the top of the document to open or close the table of contents.
2. At the top of the document, you'll see additional icons which you can click to
   - search the document,
   - change the size, font or color scheme of the page, and
   - download a PDF or EPUB (Kindle-readable) version of the entire document.
3. The document will be updated (unpredictably) throughout the semester.

The 431 Course online

The **main web page** for the 431 course in Fall 2020 is https://thomaselove.github.io/431/. Go there for all information related to the course.

What You'll Find Here

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

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.

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 RStudio (the "program" we use to interface with the R language) in class.

All data and R code related to these notes are also available to you.

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 (2019).

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 (2019).

Initial Setup of 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(knitr)
library(magrittr)
library(janitor)
library(NHANES)
library(palmerpenguins)
library(patchwork)
library(rms)
library(tidyverse) # note: tidyverse includes the dplyr and ggplot2 packages
```

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

Additional R Packages installed for this book

Some packages need to be installed on the user's system, but do not need to be loaded by R in order to run the code presented in this set of notes. These additional packages include the following.

```
car
Epi
gt
psych
mosaic
naniar
tidymodels
visdat
```

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.

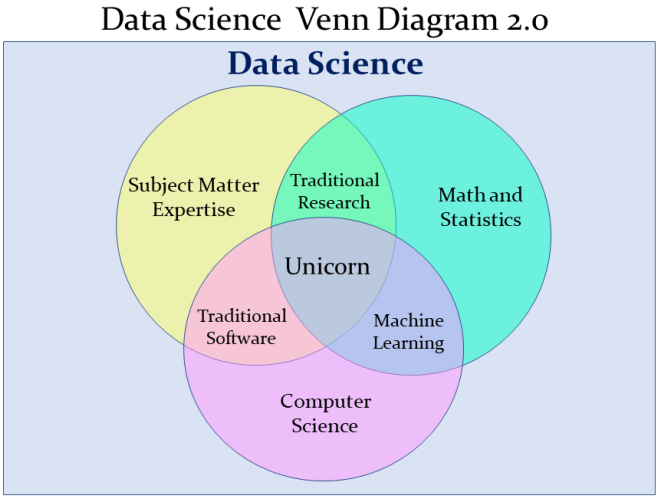


Figure 1.1: Data Science Venn Diagram from Steven Geringer

- 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.

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

1.1 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, 2019).

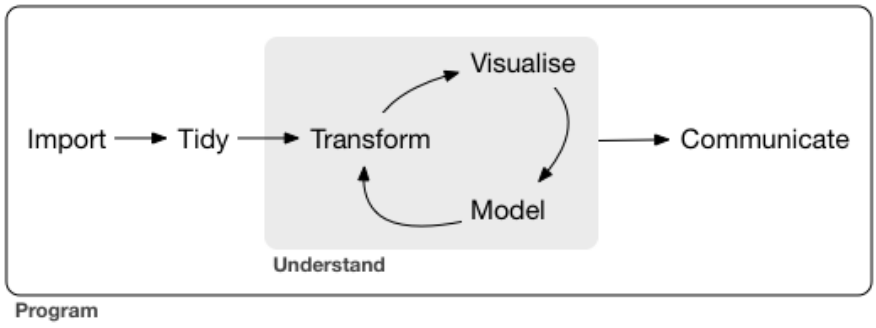


Figure 1.2: Source: R for Data Science: Introduction

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 (2019).

1.2 Data Science and the 431 Course

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 (2019)
- 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 (2019). 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.

1.3 What The Course Is and Isn't

The 431 course is about **getting things done**. 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 collect and use data effectively to address questions of interest.

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

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

It also nearly completely avoids formalism and is extremely applied - this is absolutely **not** a course in theoretical or mathematical statistics, and these Notes reflect that approach.

There's very little of the mathematical underpinnings here:

$$f(x) = \frac{e^{-(x-\mu)^2/(2\sigma^2)}}{\sigma\sqrt{2\pi}}$$

Instead, these notes (and the course) focus on how we get R to do the things we want to do, and how we interpret the results of our work. Our next Chapter provides a first example.

Part A. Exploring Data

Chapter 2

Looking at the Palmer Penguins

The data in the `palmerpenguins` package in R include size measurements, clutch observations, and blood isotope ratios for adult foraging Adélie, Chinstrap, and Gentoo penguins observed on islands in the Palmer Archipelago near Palmer Station, Antarctica. The data were collected and made available by Dr. Kristen Gorman and the Palmer Station Long Term Ecological Research (LTER) Program.

For more on the `palmerpenguins` package, visit https://allisonhorst.github.io/palmerpenguins/.

2.1 Package Loading, then Dealing with Missing Data

To start, let's load up the necessary R packages to manage the data and summarize it in a small table, and a plot. We've actually done this previously, but we'll repeat the steps here, because it's worth seeing what R is doing.

In this case, we'll load up five packages.

```
library(palmerpenguins)  # source for the data set
library(janitor)         # some utilities for cleanup and simple tables
library(magrittr)        # provides us with the pipe %>% for code management
library(dplyr)           # part of the tidyverse: data management tools
library(ggplot2)         # part of the tidyverse: tools for plotting data
```

It's worth remembering that everything after the `#` on each line above is just a comment for the reader, and is ignored by R. We'll see later that the loading

of a single package (called `tidyverse`) gives us both the `dplyr` and `ggplot2` packages, as well as several other useful things.

Next, let's take the `penguins` data from the `palmerpenguins` package, and identify those observations which have complete data (so, no missing values) in four variables of interest. We'll store that result in a new data frame (think of this as a data set) called `new_penguins` and then take a look at that result using the following code.

```
new_penguins <- penguins %>%
    filter(complete.cases(flipper_length_mm, body_mass_g, species, sex))

new_penguins
```

```
# A tibble: 333 x 8
  species island bill_length_mm bill_depth_mm flipper_length_~ body_mass_g
  <fct>   <fct>          <dbl>         <dbl>           <int>       <int>
 1 Adelie  Torge~          39.1          18.7             181        3750
 2 Adelie  Torge~          39.5          17.4             186        3800
 3 Adelie  Torge~          40.3          18               195        3250
 4 Adelie  Torge~          36.7          19.3             193        3450
 5 Adelie  Torge~          39.3          20.6             190        3650
 6 Adelie  Torge~          38.9          17.8             181        3625
 7 Adelie  Torge~          39.2          19.6             195        4675
 8 Adelie  Torge~          41.1          17.6             182        3200
 9 Adelie  Torge~          38.6          21.2             191        3800
10 Adelie  Torge~          34.6          21.1             198        4400
# ... with 323 more rows, and 2 more variables: sex <fct>, year <int>
```

2.2 Counting Things and Making Tables

So, how many penguins are in our `new_penguins` data? When we printed out the result, we got an answer, but (as with many things in R) there are many ways to get the same result.

```
nrow(new_penguins)
```

```
[1] 333
```

How do our `new_penguins` data break down by sex and species?

```
new_penguins %>%
    tabyl(sex, species) # tabyl comes from the janitor package
```

```
    sex Adelie Chinstrap Gentoo
 female     73        34     58
   male     73        34     61
```

Note the strange spelling of `tabyl` here. The output is reasonably clear, but could we make that table a little prettier, and while we're at it, can we add the row and column totals to it?

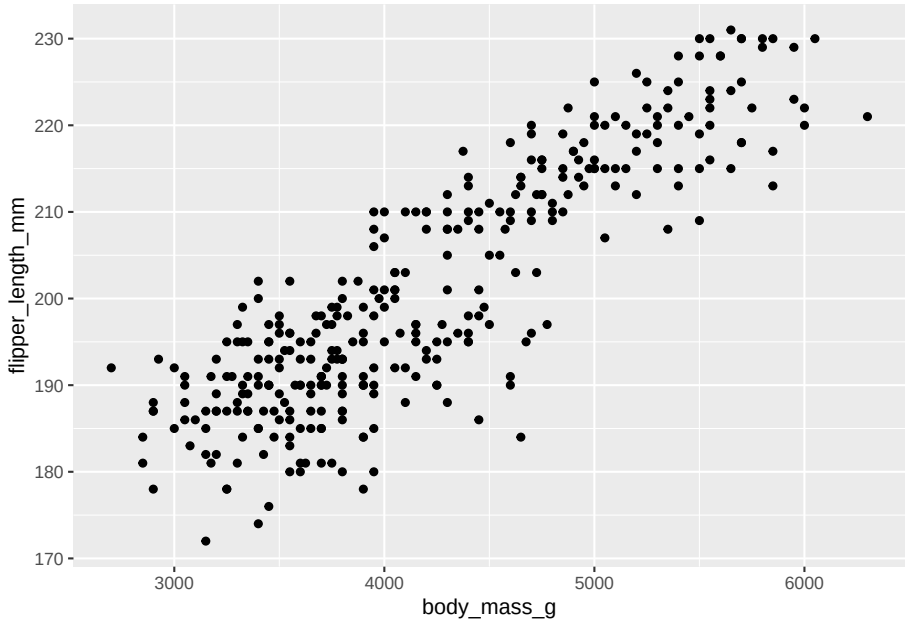
```
new_penguins %>%
    tabyl(sex, species) %>%
    adorn_totals(where = c("row", "col")) %>% # add row, column totals
    kable  # one convenient way to make the table prettier
```

sex	Adelie	Chinstrap	Gentoo	Total
female	73	34	58	165
male	73	34	61	168
Total	146	68	119	333

2.3 Visualizing the Data in a Graph (or a few…)

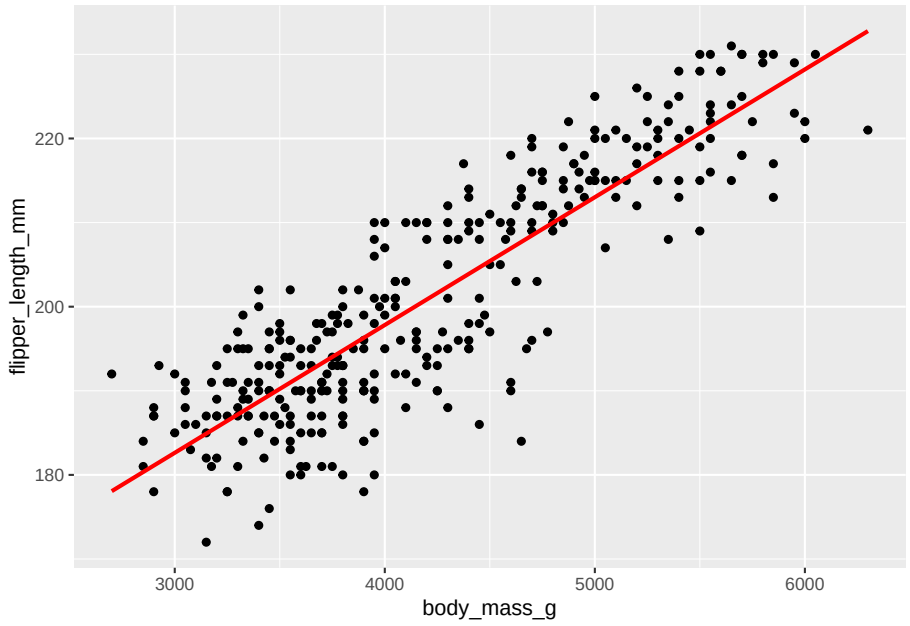
Now, let's look at the other two variables of interest. Let's create a graph showing the association of body mass with flipper length across the complete set of 333 penguins.

```
ggplot(new_penguins, aes(x = body_mass_g, y = flipper_length_mm)) +
    geom_point()
```



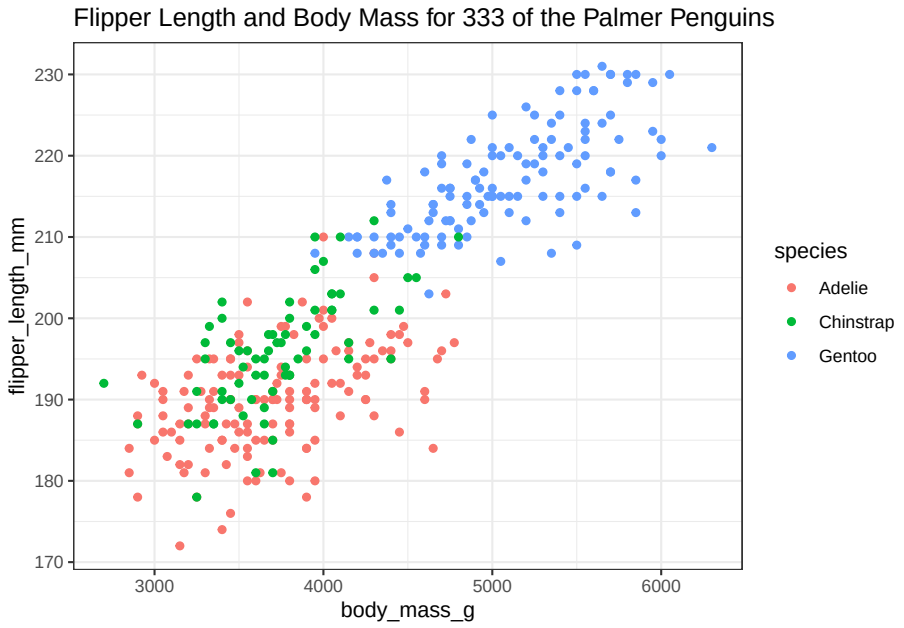
Some of you may want to include a straight-line model (fit by a classical linear regression) to this plot. One way to do that in R involves the addition of a single line of code, like this:

```
ggplot(new_penguins, aes(x = body_mass_g, y = flipper_length_mm)) +
    geom_point() +
    geom_smooth(method = "lm", col = "red", se = FALSE)
```



Whenever we build a graph for ourselves, these default choices may be sufficient. But I'd like to see a prettier version if I was going to show it to someone else. So, I might use a different color for each species, and I might neaten up the theme (to get rid of the default grey background) and add a title, like this.

```
ggplot(new_penguins, aes(x = body_mass_g, y = flipper_length_mm, col = species)) +
    geom_point() +
    theme_bw() +
    labs(title = "Flipper Length and Body Mass for 333 of the Palmer Penguins")
```



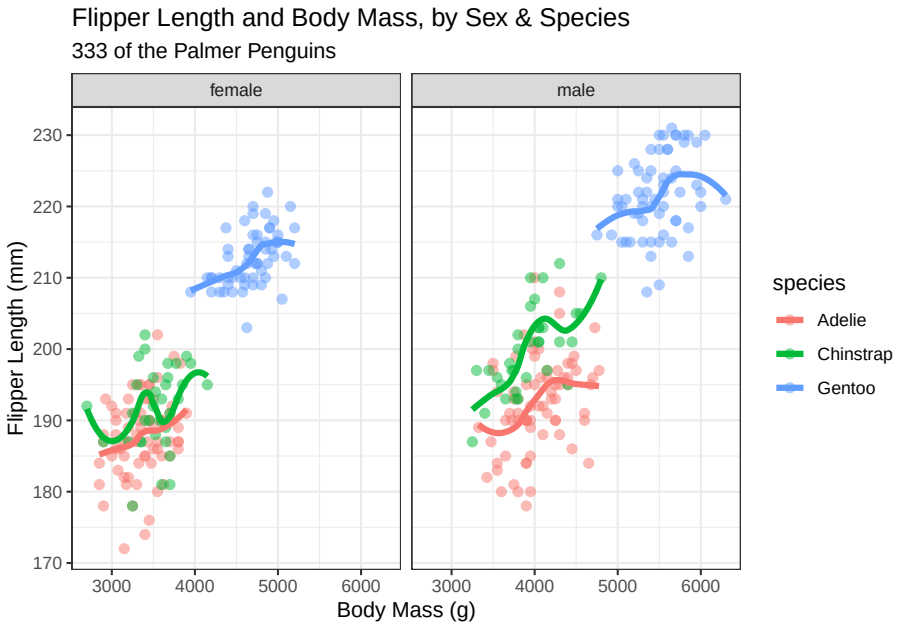
2.4 Six Ways To "Improve" This Graph

Now, let's build a new graph. Here, I want to:

1. plot the relationship between body mass and flipper length in light of both Sex and Species
2. increase the size of the points and add a little transparency so we can see if points overlap,
3. add some smooth curves to summarize the relationships between the two quantities (body mass and flipper length) within each combination of species and sex,
4. split the graph into two "facets" (one for each sex),
5. improve the axis labels,
6. improve the titles by adding a subtitle, and also adding in some code to count the penguins (rather than hard-coding in the total number.)

```
ggplot(new_penguins, aes(x = body_mass_g, y = flipper_length_mm,
                         col = species)) +
    geom_point(size = 2, alpha = 0.5) +
    geom_smooth(method = "loess", se = FALSE, size = 1.5) +
    facet_grid(~ sex) +
    theme_bw() +
    labs(title = "Flipper Length and Body Mass, by Sex & Species",
         subtitle = paste0(nrow(new_penguins), " of the Palmer Penguins"),
```

```
        x = "Body Mass (g)",
        y = "Flipper Length (mm)")
```



2.5 A Little Reflection

What can we learn from these plots and their construction? In particular,

- What do these plots suggest about the center of the distribution of each quantity (body mass and flipper length) overall, and within each combination of Sex and Species?
- What does the final plot suggest about the spread of the distribution of each of those quantities in each combination of Sex and Species?
- What do the plots suggest about the association of body mass and flipper length across the complete set of penguins?
- How does the shape and nature of this body mass - flipper length relationship change based on Sex and Species?
- Do you think it would be helpful to plot a straight-line relationship (rather than a smooth curve) within each combination of Sex and Species in the final plot? Why or why not? (Also, what would we have to do to the code to accomplish this?)
- How was the R code for the plot revised to accomplish each of the six "wants" specified above?

Chapter 3

NHANES: Initial Exploring

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]. Some of the motivation for this example came from a Figure in Baumer et al. (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_dat1 <- sample_n(NHANES, size = 1000) %>%
    select(ID, Gender, Age, Height)

nh_dat1
```

```
# A tibble: 1,000 x 4
      ID Gender   Age Height
   <int> <fct>  <int>  <dbl>
 1 69638 female     5   106.
 2 70782 male      64   176.
```

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

```
 3 52408 female    54   162.
 4 59031 female    15   155.
 5 64530 male      53   185.
 6 71040 male      63   169.
 7 55186 female    30   168.
 8 60211 male       5   103.
 9 55730 male      66   161.
10 68229 female    36   170.
# ... with 990 more rows
```

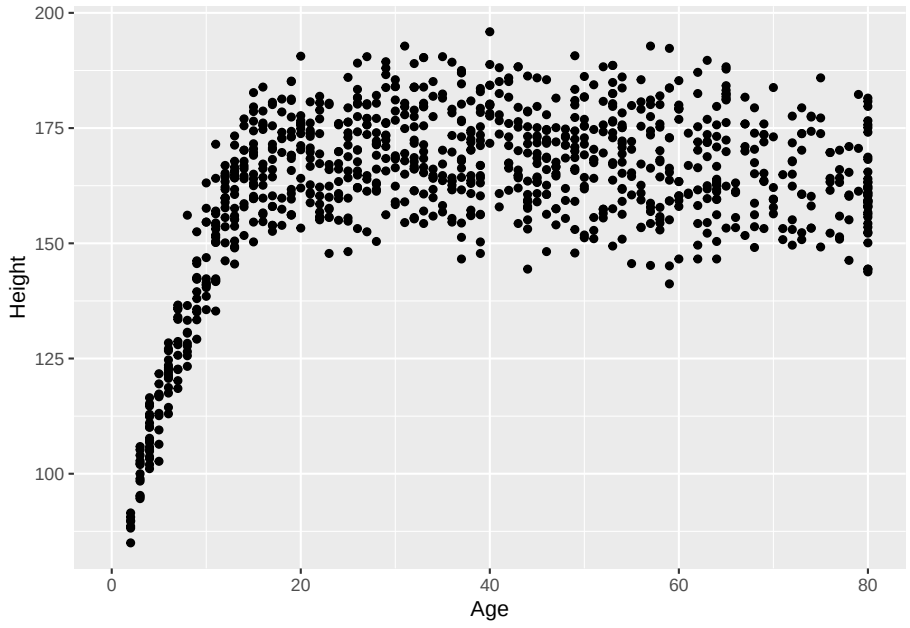
We have 1000 rows (observations) and 4 columns (variables) that describe the subjects listed in the rows.

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_dat1, aes(x = Age, y = Height)) +
    geom_point()
```

```
Warning: Removed 37 rows containing missing values (geom_point).
```



We note several interesting results here.

1. As a warning, R tells us that it has "Removed 37 rows containing missing values (geom_point)." Only 963 subjects plotted here, because the remaining 37 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 (2019).

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_dat1 %>%
    filter(complete.cases(Age, Height))

summary(nh_dat2)
```

```
       ID           Gender         Age            Height     
 Min.   :51624   female:484   Min.   : 2.00   Min.   : 85.0  
 1st Qu.:57034   male  :479   1st Qu.:19.00   1st Qu.:156.2  
 Median :62056                Median :37.00   Median :165.0  
 Mean   :61967                Mean   :38.29   Mean   :162.3  
 3rd Qu.:67269                3rd Qu.:56.00   3rd Qu.:174.5  
 Max.   :71875                Max.   :80.00   Max.   :195.9  
```

Note that the units and explanations for these variables are contained in the NHANES help file, available via typing `?NHANES` in the Console of R Studio, or by typing `NHANES` into the Search bar in R Studio's Help window.

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

3.3.1 The Distinction between `Gender` and `Sex`

The `Gender` variable here is a mistake. These data refer to the biological status of these subjects, which is their `Sex`, and not the social construct of `Gender` which can be quite different. In our effort to avoid further confusion, we'll rename the variable `Gender` to instead more accurately describe what is actually measured here.

To do this, we can use this approach…

```
nh_dat2 <- nh_dat1 %>%
    rename(Sex = Gender) %>%
    filter(complete.cases(Age, Height))

summary(nh_dat2)
```

```
       ID            Sex           Age             Height
 Min.   :51624   female:484   Min.   : 2.00   Min.   : 85.0
 1st Qu.:57034   male  :479   1st Qu.:19.00   1st Qu.:156.2
 Median :62056                Median :37.00   Median :165.0
 Mean   :61967                Mean   :38.29   Mean   :162.3
 3rd Qu.:67269                3rd Qu.:56.00   3rd Qu.:174.5
 Max.   :71875                Max.   :80.00   Max.   :195.9
```

That's better. How many observations do we have now? We could use `dim` to find out the number of rows and columns in this new data set.

```
dim(nh_dat2)
```

```
[1] 963   4
```

Or, we could simply list the data set and read off the result.

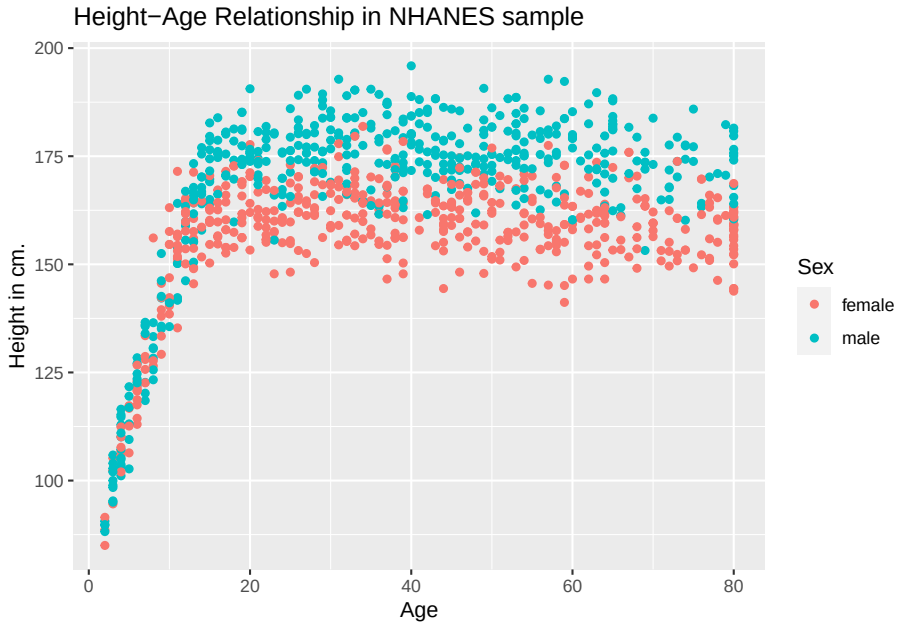
```
nh_dat2
```

```
# A tibble: 963 x 4
      ID Sex      Age Height
   <int> <fct>  <int>  <dbl>
 1 69638 female     5   106.
 2 70782 male      64   176.
 3 52408 female    54   162.
 4 59031 female    15   155.
 5 64530 male      53   185.
 6 71040 male      63   169.
 7 55186 female    30   168.
 8 60211 male       5   103.
 9 55730 male      66   161.
10 68229 female    36   170.
# ... with 953 more rows
```

3.4 Age-Height and Sex?

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

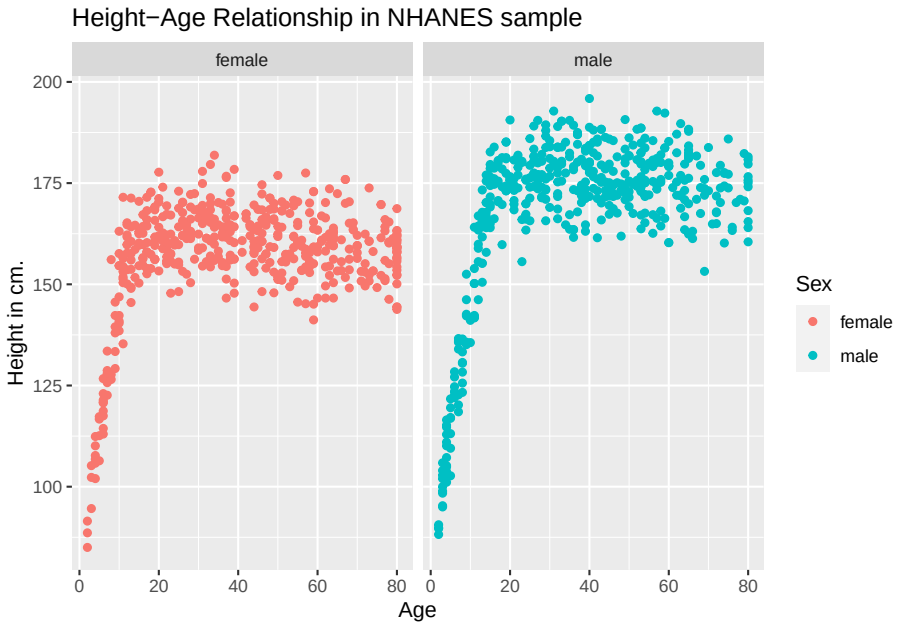
```
ggplot(data = nh_dat2, aes(x = Age, y = Height, color = Sex)) +
    geom_point() +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.")
```



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 = Sex)) +
    geom_point() +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    facet_wrap(~ Sex)
```

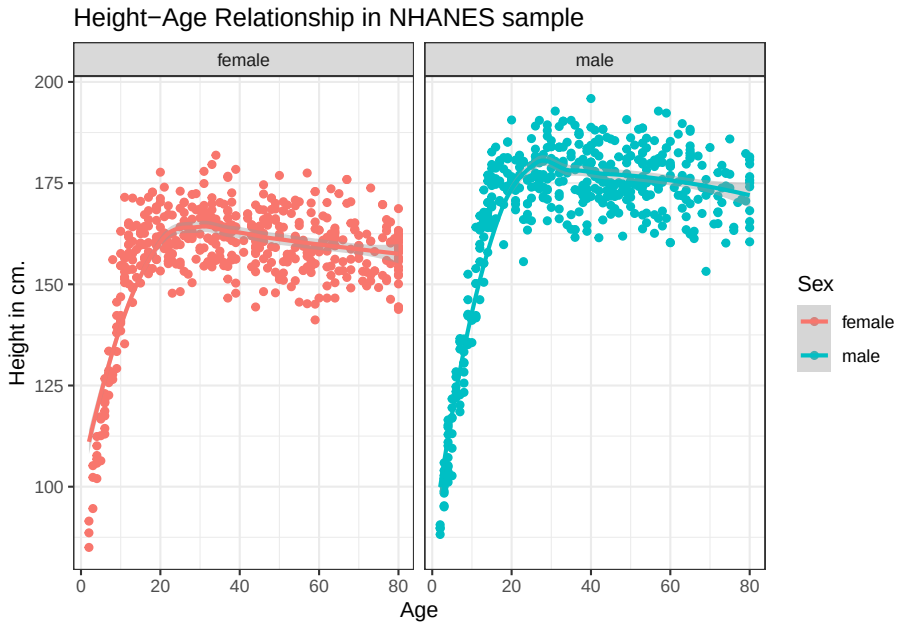


3.4.2 Can we add a smooth curve to show the relationship in each plot?

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 = Sex)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    theme_bw() +
    facet_wrap(~ Sex)
```

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

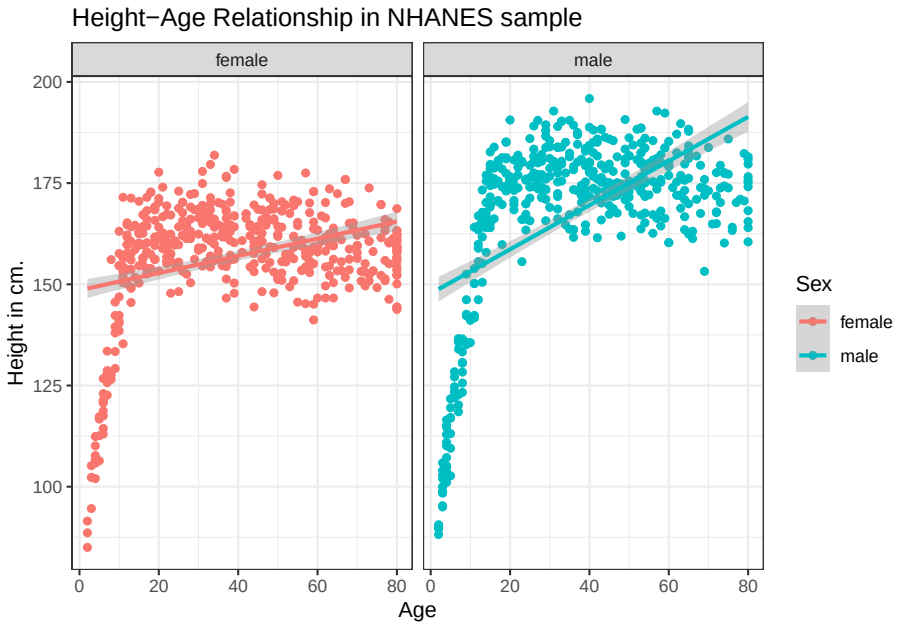


3.4.3 What if we want to assume straight line relationships?

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 = Sex)) +
    geom_point() +
    geom_smooth(method = "lm") +
    labs(title = "Height-Age Relationship in NHANES sample",
         y = "Height in cm.") +
    theme_bw() +
    facet_wrap(~ Sex)
```

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



3.5 Creating A New Subset: Ages 21-79

Suppose we wanted to look only at those observations (subjects) whose Age is at least 21 and at most 79. Suppose also that we want to look at some of the additional variables available in NHANES. To start, we'll do the following:

1. Set the same seed for random sampling that we used earlier, so that we start with the original sample of 1000 people we built earlier. Draw that same sample of 1,000 people.
2. Filter the sample to only those people whose age is more than 20 and less than 80 years.
3. Select the variables we will use in the rest of this chapter:
    - `Age` as we've seen before, in years.
    - `Height` as we've seen before, in centimeters.
    - `Gender` which we'll rename as `Sex` again.
    - `Pulse` = 60 second pulse rate (in beats per minute).
    - `BPSysAve` = Systolic Blood Pressure, in mm Hg (and we'll rename this `SBP`).
    - `SleepTrouble` = Yes means the subject has told a health professional that they had trouble sleeping.
    - `PhysActive` = Yes means the subject does moderate or vigorous-intensity sports, fitness or recreational activity.
    - `MaritalStatus` = one of Married, Widowed, Divorced, Separated, NeverMarried or LivePartner (living with partner.)

- `HealthGen` = self-reported rating of general health, one of Excellent, Vgood (Very Good), Good, Fair or Poor.

4. Rename `Gender` as `Sex`, to more accurately describe what is being measured.
5. Omit subjects with any missingness on *any* of the variables we've selected.

Can you see how the code below accomplishes these tasks?

```
set.seed(431001) # again, this will ensure the same sample

nh_dat3 <- sample_n(NHANES, size = 1000) %>%
    filter(Age > 20 & Age < 80) %>%
    select(ID, Gender, Age, Height,
           Pulse, BPSysAve, SleepTrouble, PhysActive,
           MaritalStatus, HealthGen) %>%
    rename(Sex = Gender, SBP = BPSysAve) %>%
    na.omit

nh_dat3
```

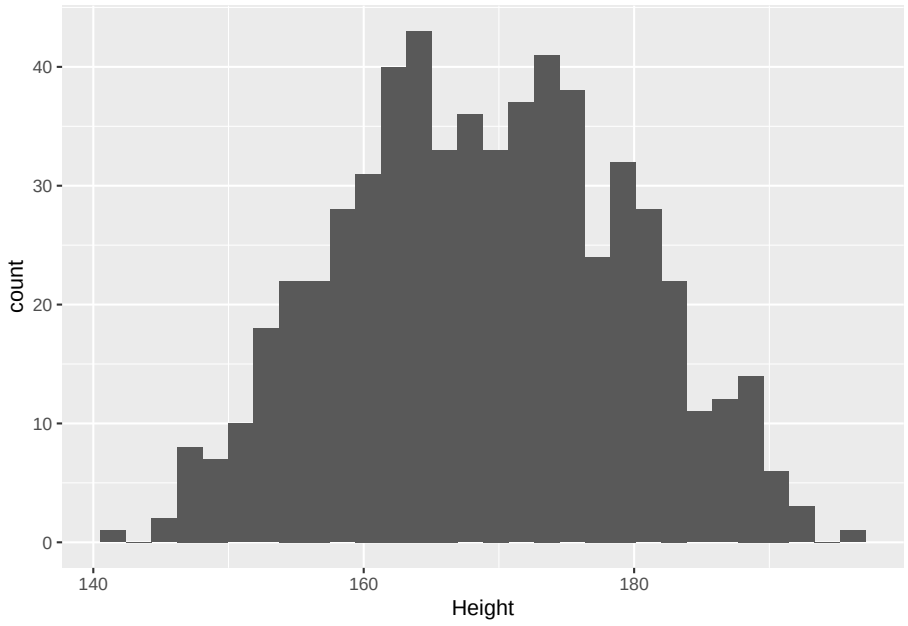
```
# A tibble: 603 x 10
    ID Sex    Age Height Pulse   SBP SleepTrouble PhysActive MaritalStatus
   <int> <fct> <int>  <dbl> <int> <int> <fct>       <fct>      <fct>
 1 70782 male     64   176.    78   127 No          No         Married
 2 52408 fema~    54   162.    80   135 No          No         LivePartner
 3 64530 male     53   185.   100   131 No          No         Married
 4 71040 male     63   169.    70   124 Yes         Yes        Married
 5 55186 fema~    30   168.    76   107 No          No         Married
 6 55730 male     66   161.    78   133 No          No         Married
 7 68229 fema~    36   170.    90   105 No          Yes        Married
 8 63762 male     23   180.    66   118 No          No         Married
 9 66290 fema~    63   162.    88   116 No          No         Married
10 66984 male     75   174.    84   141 No          No         Married
# ... with 593 more rows, and 1 more variable: HealthGen <fct>
```

3.6 Distribution of Heights

What is the distribution of height in this new sample?

```
ggplot(data = nh_dat3, aes(x = Height)) +
    geom_histogram()
```

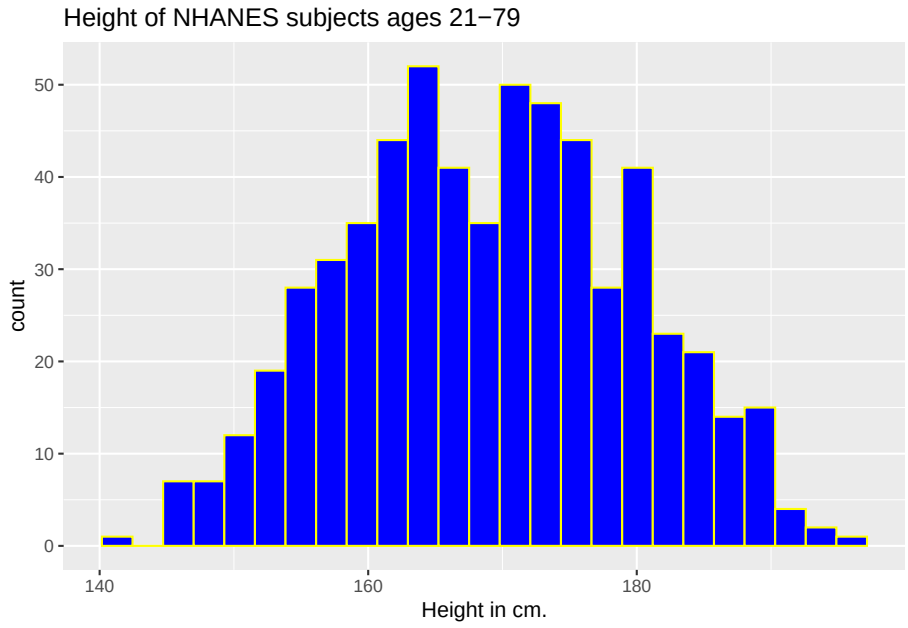
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



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_dat3, aes(x = Height)) +
    geom_histogram(bins = 25, col = "yellow", fill = "blue") +
    labs(title = "Height of NHANES subjects ages 21-79",
         x = "Height in cm.")
```

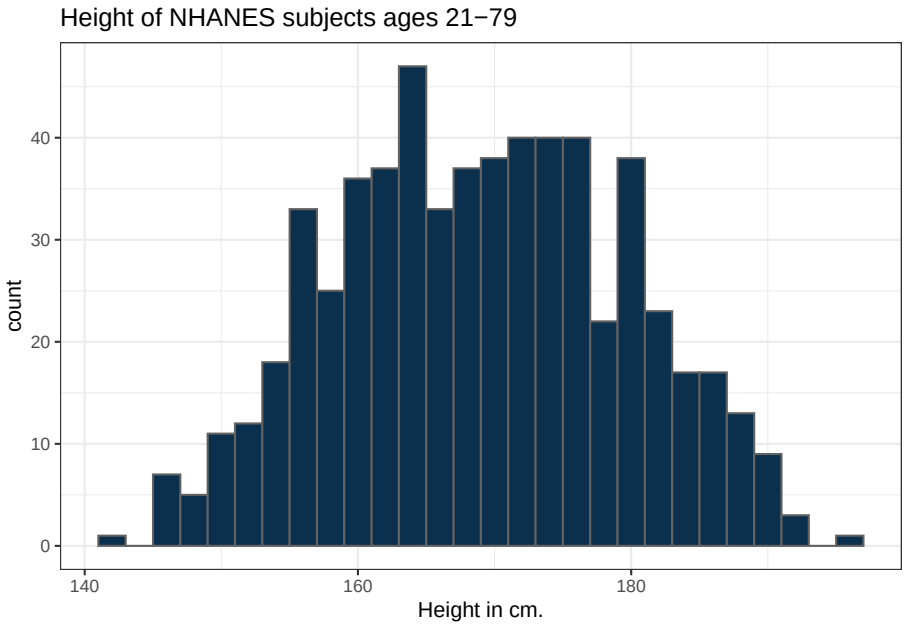


3.6.1 Changing a Histogram's Fill and Color

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_dat3, 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()
```

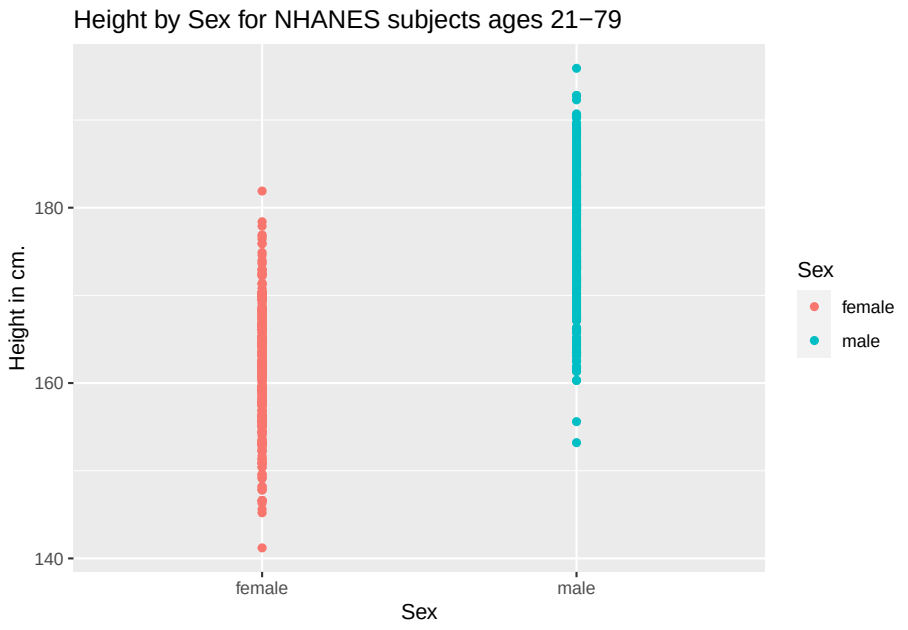


Note the other changes to the graph above.

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.

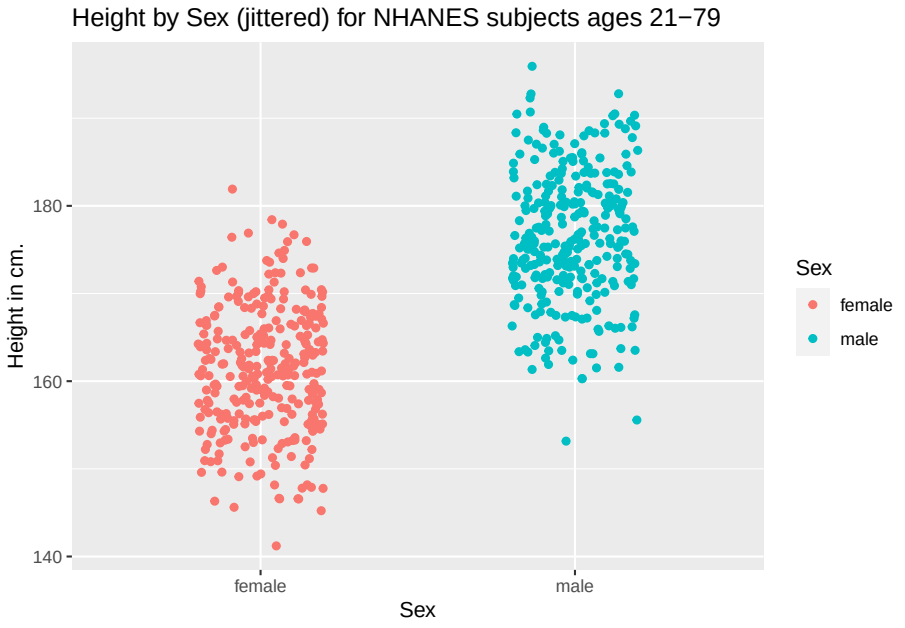
3.7 Height and Sex

```
ggplot(data = nh_dat3, aes(x = Sex, y = Height, color = Sex)) +
    geom_point() +
    labs(title = "Height by Sex 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.

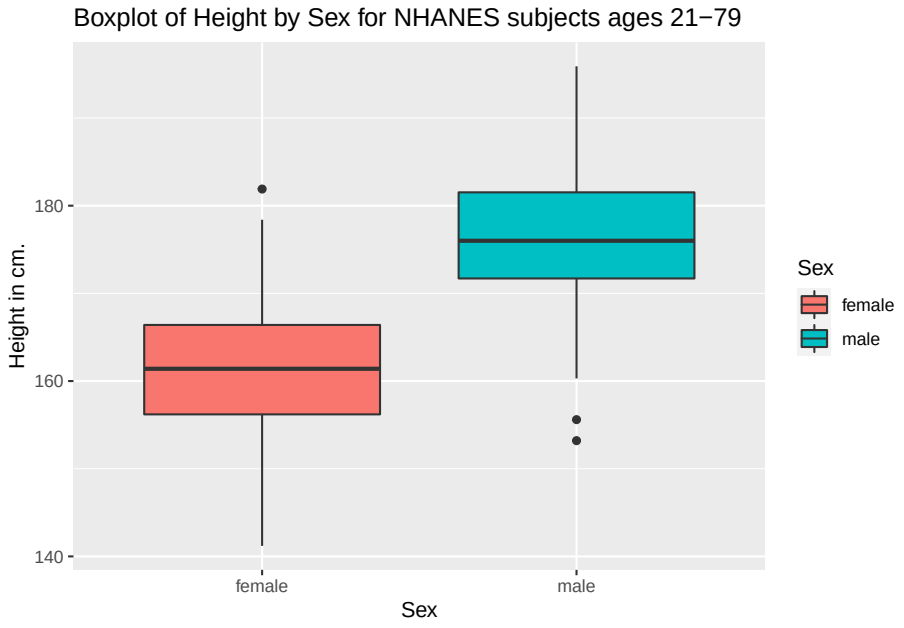
```
ggplot(data = nh_dat3, aes(x = Sex, y = Height, color = Sex)) +
    geom_jitter(width = 0.2) +
    labs(title = "Height by Sex (jittered) for NHANES subjects ages 21-79",
         y = "Height in cm.")
```



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 Sex

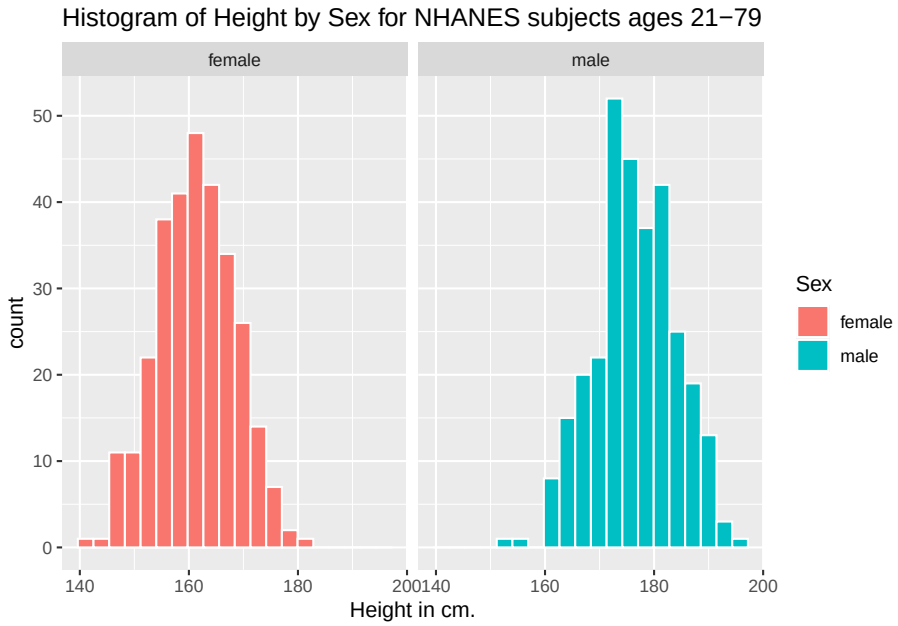
```
ggplot(data = nh_dat3, aes(x = Sex, y = Height, fill = Sex)) +
    geom_boxplot() +
    labs(title = "Boxplot of Height by Sex for NHANES subjects ages 21-79",
         y = "Height in cm.")
```



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

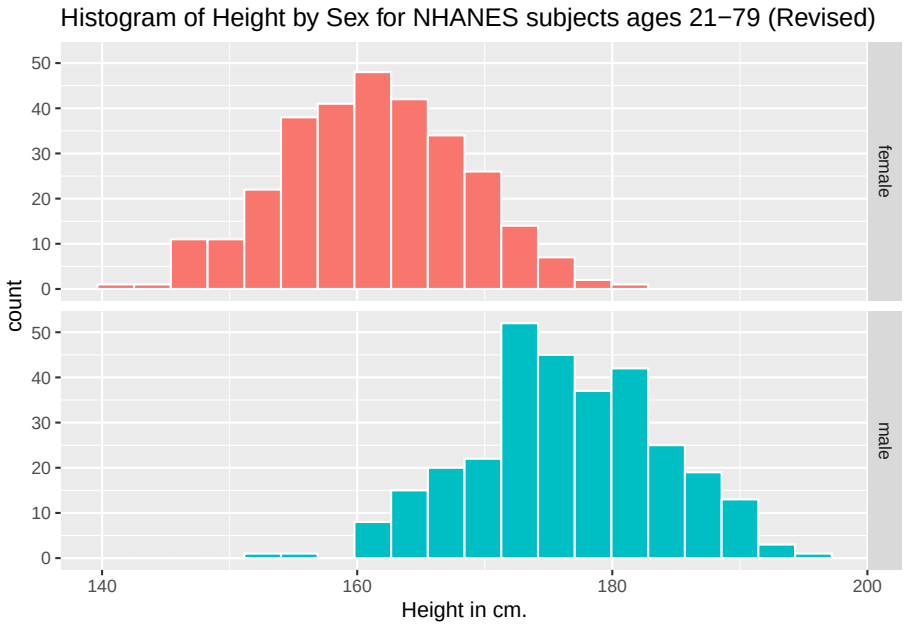
3.7.2 Histograms of Height by Sex

```
ggplot(data = nh_dat3, aes(x = Height, fill = Sex)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of Height by Sex for NHANES subjects ages 21-79",
         x = "Height in cm.") +
    facet_wrap(~ Sex)
```



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

```
ggplot(data = nh_dat3, aes(x = Height, fill = Sex)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of Height by Sex for NHANES subjects ages 21-79 (Revised)"
         x = "Height in cm.") +
    guides(fill = FALSE) +
    facet_grid(Sex ~ .)
```

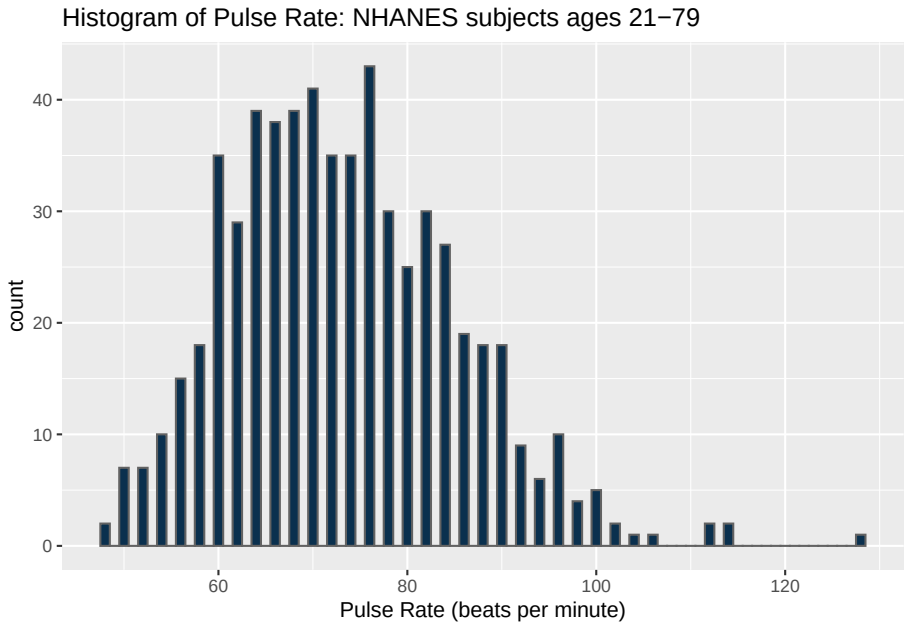


3.8 Looking at Pulse Rate

Let's look at a different outcome, the *pulse rate* for our subjects.

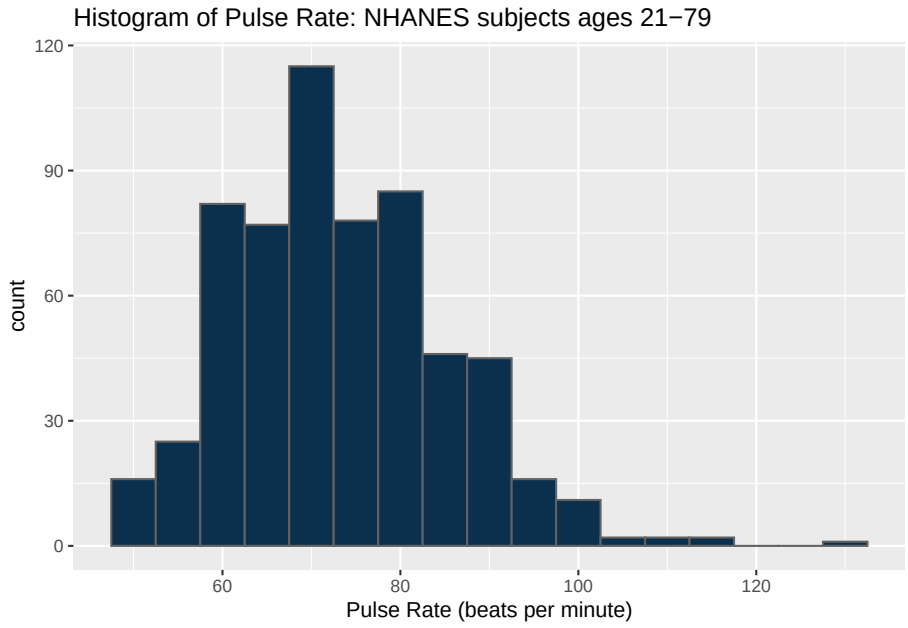
Here's a histogram, again with CWRU colors, for the pulse rates in our sample.

```
ggplot(data = nh_dat3, aes(x = Pulse)) +
    geom_histogram(binwidth = 1, fill = cwru.blue, col = cwru.gray) +
    labs(title = "Histogram of Pulse Rate: NHANES subjects ages 21-79",
         x = "Pulse Rate (beats per minute)")
```



Suppose we instead bin up groups of 5 beats per minute together as we plot the Pulse rates.

```
ggplot(data = nh_dat3, aes(x = Pulse)) +
    geom_histogram(binwidth = 5, fill = cwru.blue, col = cwru.gray) +
    labs(title = "Histogram of Pulse Rate: NHANES subjects ages 21-79",
         x = "Pulse Rate (beats per minute)")
```

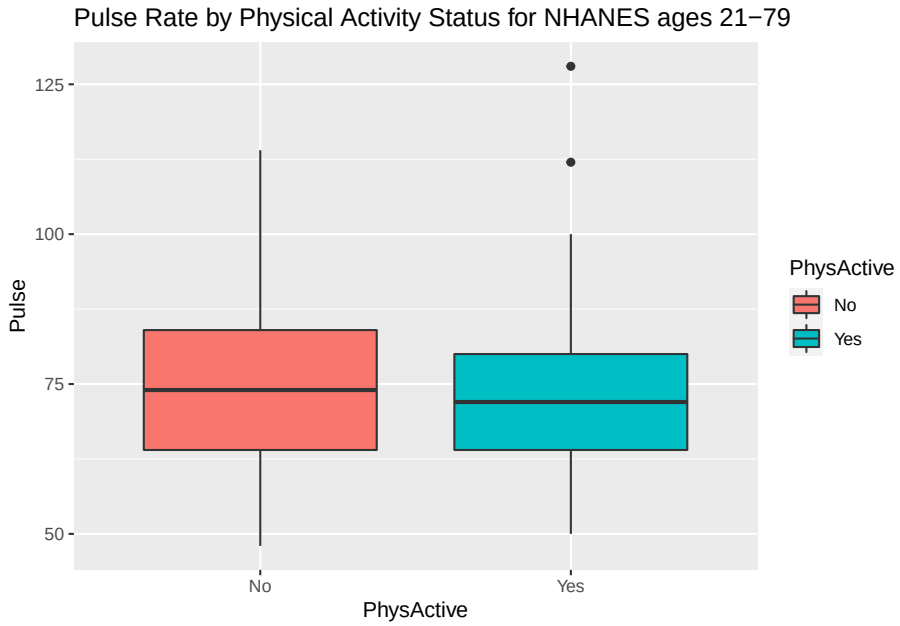


Which is the more useful representation will depend a lot on what questions you're trying to answer.

3.8.1 Pulse Rate and Physical Activity

We can also split up our data into groups based on whether the subjects are physically active. Let's try a boxplot.

```
ggplot(data = nh_dat3, aes(y = Pulse, x = PhysActive, fill = PhysActive)) +
    geom_boxplot() +
    labs(title = "Pulse Rate by Physical Activity Status for NHANES ages 21-79")
```



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

```
nh_dat3 %>%
    group_by(PhysActive) %>%
    summarise(count = n(), mean(Pulse), median(Pulse)) %>%
    knitr::kable(digits = 2)
```

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

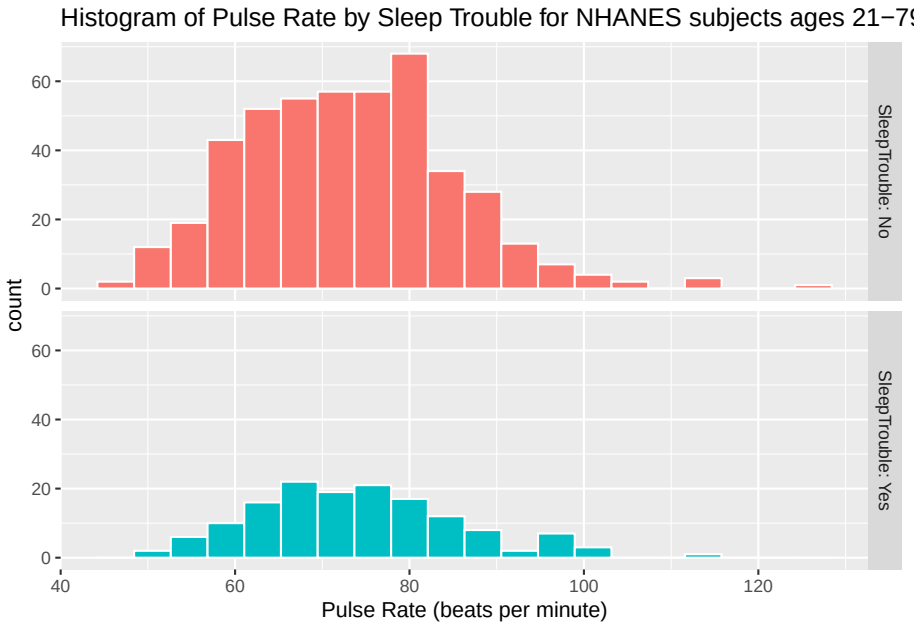
PhysActive	count	mean(Pulse)	median(Pulse)
No	293	74.21	74
Yes	310	72.37	72

The `knitr::kable(digits = 2)` piece of this command tells R Markdown to generate a table with some attractive formatting, and rounding any decimals to two figures.

3.8.2 Pulse by Sleeping Trouble

```
ggplot(data = nh_dat3, aes(x = Pulse, fill = SleepTrouble)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "Histogram of Pulse Rate by Sleep Trouble for NHANES subjects ages 21-
         x = "Pulse Rate (beats per minute)") +
```

```
    guides(fill = FALSE) +
    facet_grid(SleepTrouble ~ ., labeller = "label_both")
```



How many people fall into each of these `SleepTrouble` categories, and what is their "average" Pulse rate?

```
nh_dat3 %>%
    group_by(SleepTrouble) %>%
    summarise(count = n(), mean(Pulse), median(Pulse)) %>%
    knitr::kable(digits = 2)
```

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

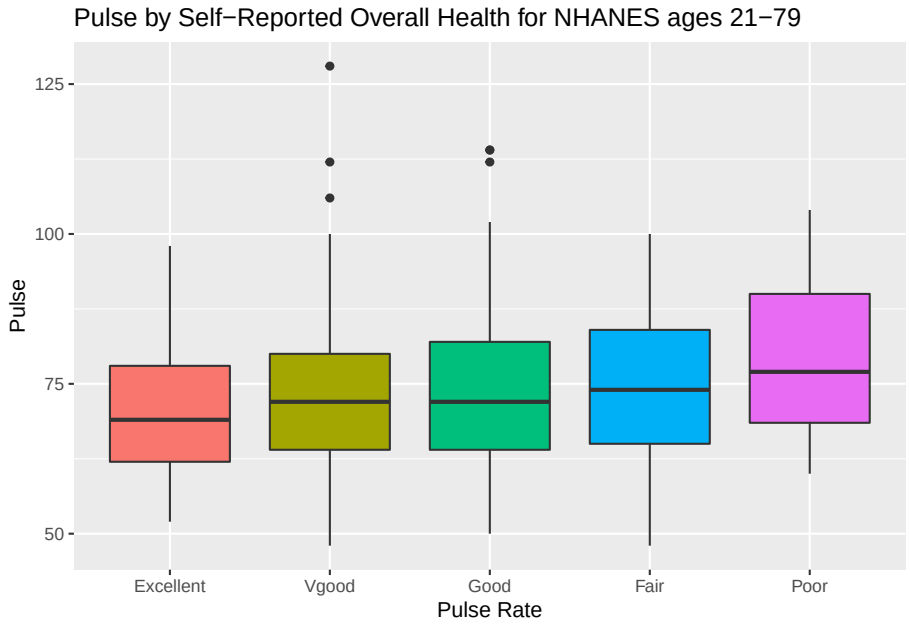
SleepTrouble	count	mean(Pulse)	median(Pulse)
No	457	73.05	72
Yes	146	73.96	72

3.8.3 Pulse and HealthGen

We can compare the distribution of Pulse rate across groups by the subject's self-reported overall health (`HealthGen`), as well.

```
ggplot(data = nh_dat3, aes(x = HealthGen, y = Pulse, fill = HealthGen)) +
    geom_boxplot() +
    labs(title = "Pulse by Self-Reported Overall Health for NHANES ages 21-79",
```

```
         x = "Pulse Rate") +
    guides(fill = FALSE)
```



How many people fall into each of these `HealthGen` categories, and what is their "average" Pulse rate?

```
nh_dat3 %>%
    group_by(HealthGen) %>%
    summarise(count = n(), mean(Pulse), median(Pulse)) %>%
    knitr::kable(digits = 2)
```

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

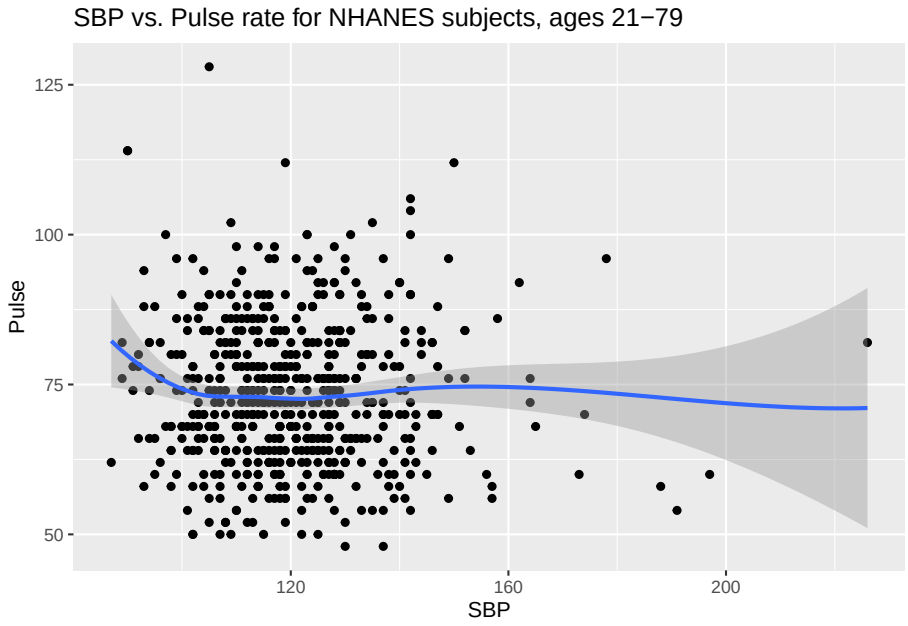
HealthGen	count	mean(Pulse)	median(Pulse)
Excellent	64	69.97	69
Vgood	196	72.81	72
Good	238	73.66	72
Fair	83	74.22	74
Poor	22	79.09	77

3.8.4 Pulse Rate and Systolic Blood Pressure

```
ggplot(data = nh_dat3, aes(x = SBP, y = Pulse)) +
    geom_point() +
```

```
    geom_smooth(method = "loess") +
    labs(title = "SBP vs. Pulse rate for NHANES subjects, ages 21-79")
```

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



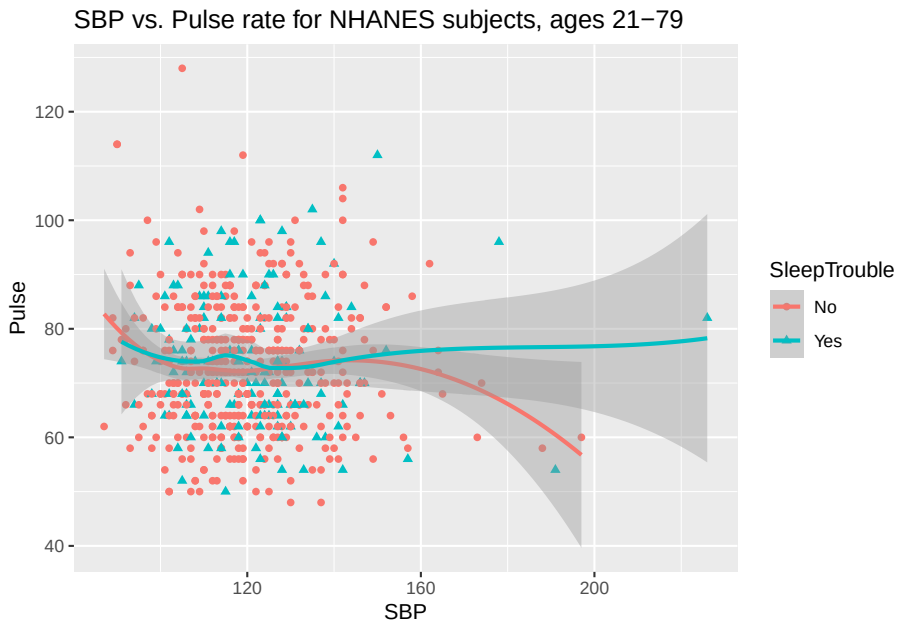
3.8.5 Sleep Troubls vs. No Sleep Trouble?

Could we see whether subjects who have described `SleepTrouble` show different SBP-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 `SleepTrouble`.

```
ggplot(data = nh_dat3,
       aes(x = SBP, y = Pulse,
           color = SleepTrouble, shape = SleepTrouble)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "SBP vs. Pulse rate for NHANES subjects, ages 21-79")
```

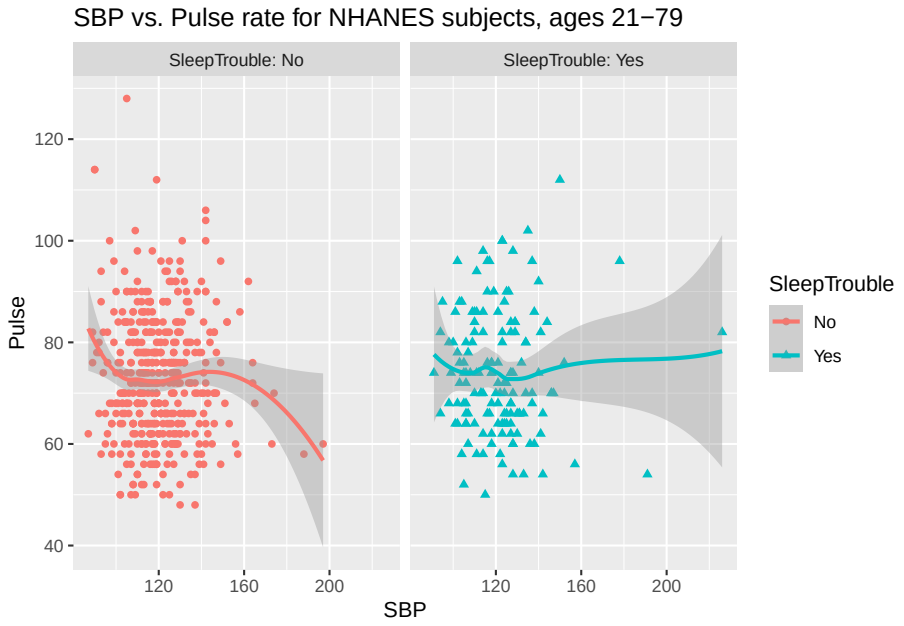
```
`geom_smooth()` using formula 'y ~ x'
```



This plot might be easier to interpret if we faceted by `SleepTrouble`, as well.

```
ggplot(data = nh_dat3,
       aes(x = SBP, y = Pulse,
           color = SleepTrouble, shape = SleepTrouble)) +
    geom_point() +
    geom_smooth(method = "loess") +
    labs(title = "SBP vs. Pulse rate for NHANES subjects, ages 21-79") +
    facet_wrap(~ SleepTrouble, labeller = "label_both")
```

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



3.9 General Health Status

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

```
nh_dat3 %>%
    select(HealthGen) %>%
    table()
```

```
.
Excellent     Vgood      Good      Fair      Poor
       64       196       238        83        22
```

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

Another, somewhat simpler way to get a table of this sort of information uses the `tabyl` function from the `janitor` package in R.

```
# tabyl is part of the janitor package
# already loaded: library(janitor)

nh_dat3 %>%
    tabyl(HealthGen)
```

```
 HealthGen   n    percent
 Excellent  64 0.10613599
     Vgood 196 0.32504146
      Good 238 0.39469320
      Fair  83 0.13764511
      Poor  22 0.03648425
```

I don't actually like the title of `percent` here, as it's really a proportion, but that can be adjusted, and we can add a total.

```
nh_dat3 %>%
    tabyl(HealthGen) %>%
    adorn_totals() %>%
    adorn_pct_formatting()
```

```
 HealthGen   n percent
 Excellent  64   10.6%
     Vgood 196   32.5%
      Good 238   39.5%
      Fair  83   13.8%
      Poor  22    3.6%
     Total 603  100.0%
```

When working with an unordered categorical variable, like `MaritalStatus`, the same approach can work.

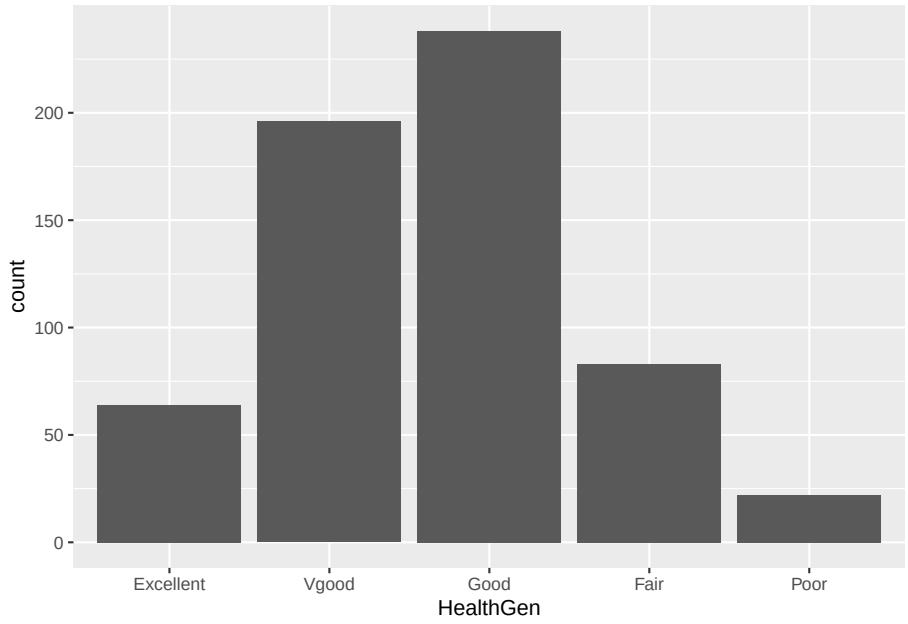
```
nh_dat3 %>%
    tabyl(MaritalStatus) %>%
    adorn_totals() %>%
    adorn_pct_formatting()
```

```
 MaritalStatus   n percent
      Divorced  61   10.1%
   LivePartner  43    7.1%
       Married 349   57.9%
  NeverMarried 104   17.2%
     Separated   8    1.3%
       Widowed  38    6.3%
         Total 603  100.0%
```

3.9.1 Bar Chart for Categorical Data

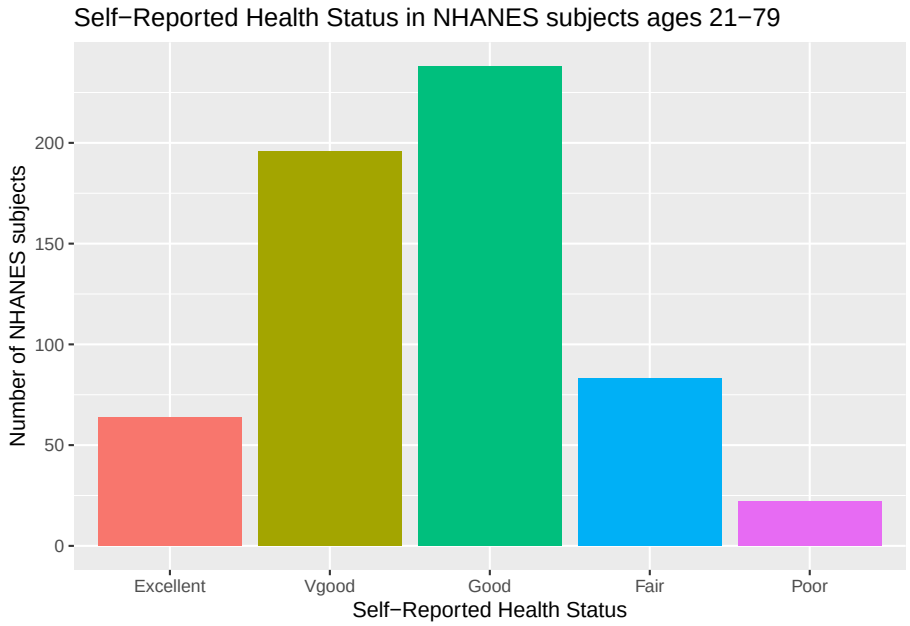
Usually, a **bar chart** is the best choice for a graphing a variable made up of categories.

```
ggplot(data = nh_dat3, aes(x = HealthGen)) +
    geom_bar()
```



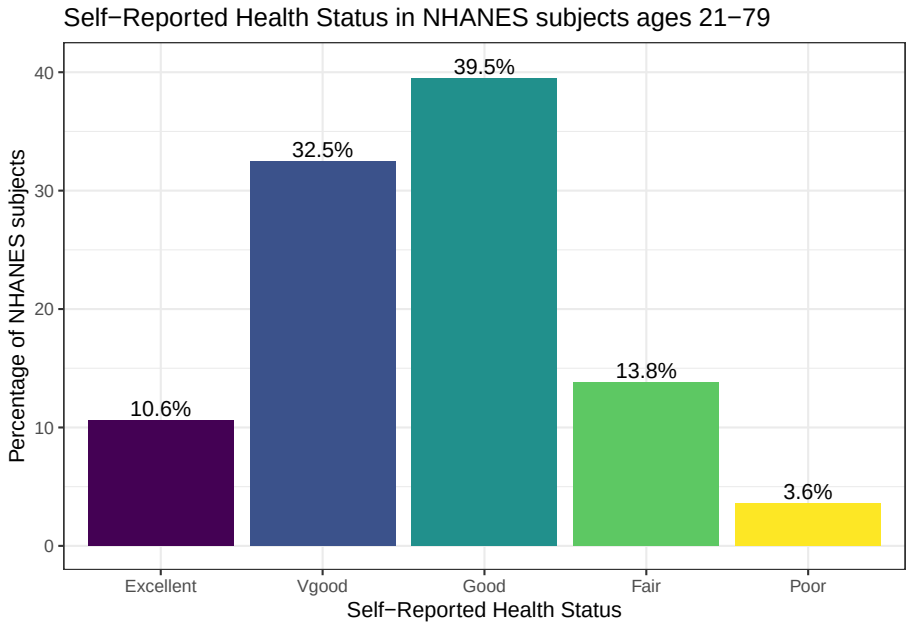
There are lots of things we can do to make this plot fancier.

```
ggplot(data = nh_dat3, 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_dat3 %>%
    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_d() +
    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 both row and column marginal totals, and compare subjects by Sex, as follows...

```
nh_dat3 %>%
    tabyl(Sex, HealthGen) %>%
    adorn_totals(c("row", "col"))
```

```
    Sex Excellent Vgood Good Fair Poor Total
 female        27    96  121   41   14   299
   male        37   100  117   42    8   304
  Total        64   196  238   83   22   603
```

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

```
nh_dat3 %>%
    tabyl(Sex, HealthGen) %>%
    adorn_totals(c("row", "col")) %>%
    knitr::kable()
```

Sex	Excellent	Vgood	Good	Fair	Poor	Total
female	27	96	121	41	14	299
male	37	100	117	42	8	304
Total	64	196	238	83	22	603

Or, we can get a complete cross-tabulation, including (in this case) the percentages of people within each Sex that fall in each HealthGen category (percentages within each row) like this.

```
nh_dat3 %>%
    tabyl(Sex, HealthGen) %>%
    adorn_totals("row") %>%
    adorn_percentages("row") %>%
    adorn_pct_formatting() %>%
    adorn_ns() %>%
    knitr::kable()
```

Sex	Excellent	Vgood	Good	Fair	Poor
female	9.0% (27)	32.1% (96)	40.5% (121)	13.7% (41)	4.7% (14)
male	12.2% (37)	32.9% (100)	38.5% (117)	13.8% (42)	2.6% (8)
Total	10.6% (64)	32.5% (196)	39.5% (238)	13.8% (83)	3.6% (22)

And, if we wanted the column percentages, to determine which sex had the higher rate of each HealthGen status level, we can get that by changing the adorn_percentages to describe results at the column level:

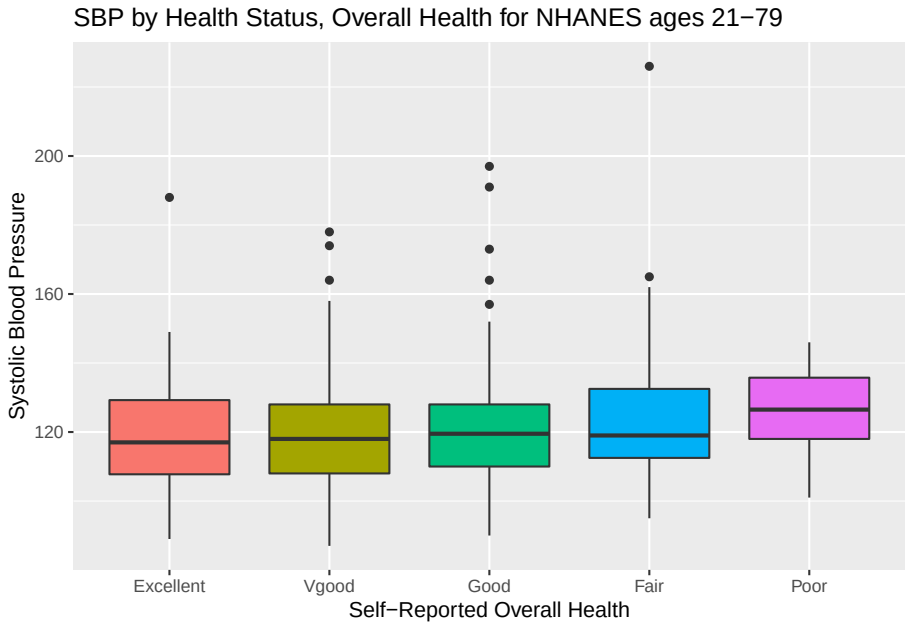
```
nh_dat3 %>%
    tabyl(Sex, HealthGen) %>%
    adorn_totals("col") %>%
    adorn_percentages("col") %>%
    adorn_pct_formatting() %>%
    adorn_ns() %>%
    knitr::kable()
```

Sex	Excellent	Vgood	Good	Fair	Poor	Total
female	42.2% (27)	49.0% (96)	50.8% (121)	49.4% (41)	63.6% (14)	49.6% (299)
male	57.8% (37)	51.0% (100)	49.2% (117)	50.6% (42)	36.4% (8)	50.4% (304)

3.9.3 SBP by General Health Status

Let's consider now the relationship between self-reported overall health and systolic blood pressure.

```
ggplot(data = nh_dat3, aes(x = HealthGen, y = SBP, fill = HealthGen)) +
    geom_boxplot() +
    labs(title = "SBP by Health Status, Overall Health for NHANES ages 21-79",
         y = "Systolic Blood Pressure", x = "Self-Reported Overall Health") +
    guides(fill = FALSE)
```



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

```
nh_dat3 %>%
    group_by(HealthGen) %>%
    summarise(count = n(), mean(SBP), median(SBP)) %>%
    knitr::kable()
```

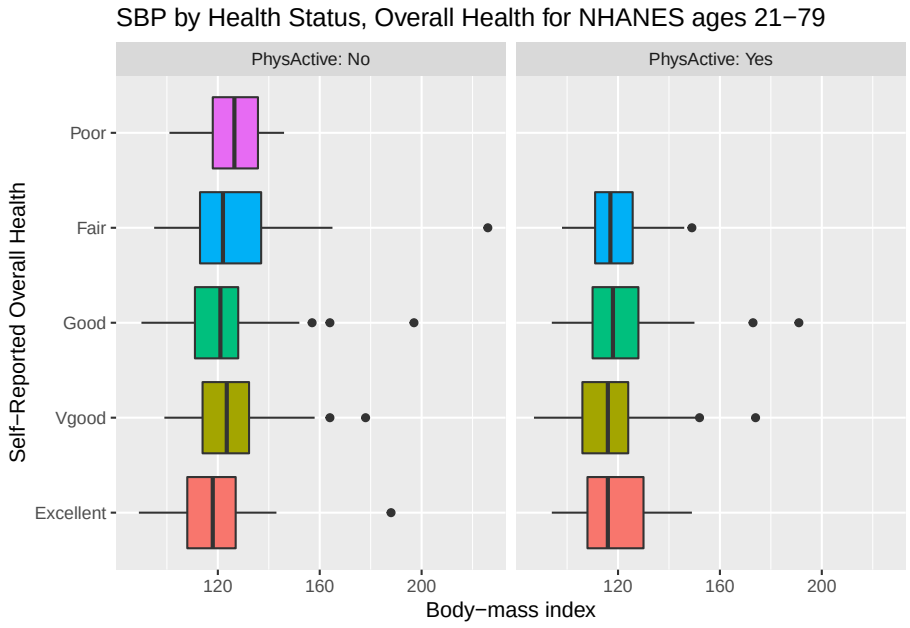
```
`summarise()` ungrouping output (override with `.groups` argument)
```

HealthGen	count	mean(SBP)	median(SBP)
Excellent	64	119.1562	117.0
Vgood	196	119.0714	118.0
Good	238	120.4244	119.5
Fair	83	123.9398	119.0
Poor	22	125.8636	126.5

3.9.4 SBP by Physical Activity and General Health Status

We'll build a panel of boxplots to try to understand the relationships between Systolic Blood Pressure, General Health Status and Physical Activity. Note the use of `coord_flip` to rotate the graph 90 degrees, and the use of `labeller` within `facet_wrap` to include both the name of the (Physical Activity) variable and its value.

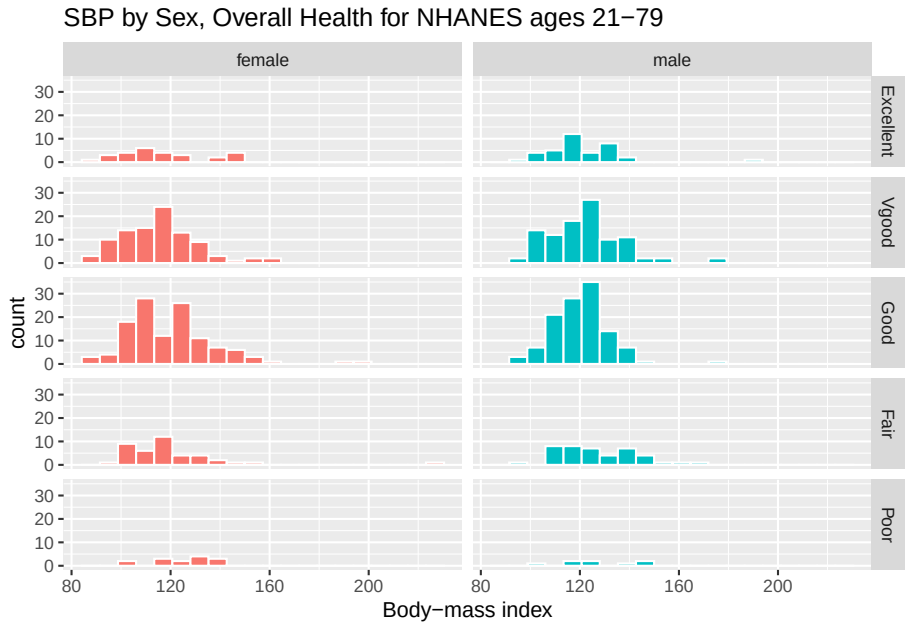
```
ggplot(data = nh_dat3, aes(x = HealthGen, y = SBP, fill = HealthGen)) +
    geom_boxplot() +
    labs(title = "SBP by Health Status, Overall Health for NHANES ages 21-79",
         y = "Body-mass index", x = "Self-Reported Overall Health") +
    guides(fill = FALSE) +
    facet_wrap(~ PhysActive, labeller = "label_both") +
    coord_flip()
```



3.9.5 SBP by Sleep Trouble and General Health Status

Here's a plot of faceted histograms, which might be used to address similar questions related to the relationship between Overall Health, Systolic Blood Pressure and Sex.

```
ggplot(data = nh_dat3, aes(x = SBP, fill = Sex)) +
    geom_histogram(color = "white", bins = 20) +
    labs(title = "SBP by Sex, Overall Health for NHANES ages 21-79",
         x = "Body-mass index") +
    guides(fill = FALSE) +
    facet_grid(HealthGen ~ Sex)
```



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 et al. (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 Chapter 3, 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,
           MaritalStatus, HealthGen)

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 14
    ID Sex    Age Race  Education  BMI   SBP   DBP Pulse PhysActive Smoke100
  <int> <fct> <int> <fct> <fct>    <dbl> <int> <int> <int> <fct>      <fct>
 1 71531 male    35 White Some Col~  22.4   143    90    84 Yes        No
 2 68613 fema~   61 White Some Col~  27.7   119    86   112 No         No
 3 67064 male    31 White College ~  26.6   110    76    86 Yes        Yes
 4 63924 fema~   29 Black High Sch~  41.9    98    56    74 No         Yes
 5 62840 male    60 White 8th Grade  35.8   127     0   110 No         Yes
 6 68058 male    50 White Some Col~  30.6    NA    NA    NA No         Yes
 7 68936 fema~   36 Black High Sch~  30.5   119    69    60 No         No
 8 71189 male    51 White College ~  25.6   112    70    54 Yes        Yes
 9 69936 fema~   54 Asian College ~  21.8   126    80    78 Yes        No
10 70687 male    59 White College ~  25.5   149    89    62 Yes        No
# ... with 490 more rows, and 3 more variables: SleepTrouble <fct>,
#   MaritalStatus <fct>, HealthGen <fct>
```

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  14
```

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

```
str(nh_adults)
```

```
tibble [500 x 14] (S3: tbl_df/tbl/data.frame)
 $ ID        : int [1:500] 71531 68613 67064 63924 62840 68058 68936 71189 69936 70687 ...
 $ Sex       : Factor w/ 2 levels "female","male": 2 1 2 1 2 2 1 2 1 2 ...
 $ Age           : int [1:500] 35 61 31 29 60 50 36 51 54 59 ...
 $ Race      : Factor w/ 6 levels "Asian","Black",..: 5 5 5 2 5 5 2 5 1 5 ...
 $ Education   : Factor w/ 5 levels "8th Grade","9 - 11th Grade",..: 4 4 5 3 1 4 3 5 5 5 ...
 $ BMI       : num [1:500] 22.4 27.7 26.6 41.9 35.8 30.6 30.5 25.6 21.8 25.5 ...
 $ SBP         : int [1:500] 143 119 110 98 127 NA 119 112 126 149 ...
 $ DBP           : int [1:500] 90 86 76 56 0 NA 69 70 80 89 ...
 $ Pulse         : int [1:500] 84 112 86 74 110 NA 60 54 78 62 ...
 $ PhysActive  : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 1 1 2 2 2 ...
 $ Smoke100    : Factor w/ 2 levels "No","Yes": 1 1 2 2 2 2 1 2 1 1 ...
 $ SleepTrouble : Factor w/ 2 levels "No","Yes": 2 1 1 2 2 2 1 1 1 1 ...
 $ MaritalStatus: Factor w/ 6 levels "Divorced","LivePartner",..: 4 6 3 5 3 3 4 3 3 6 ...
 $ HealthGen   : Factor w/ 5 levels "Excellent","Vgood",..: 3 2 3 4 5 3 3 NA 3 1 ...
```

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 14
     ID Sex    Age Race  Education  BMI   SBP   DBP Pulse PhysActive Smoke100
  <int> <fct> <int> <fct> <fct>    <dbl> <int> <int> <int> <fct>      <fct>
1 66770 fema~    22 White Some Col~  44.6   100    90    92 Yes        No
2 68754 male     57 White Some Col~  23.2   124    85    82 No         Yes
3 70911 male     59 White College ~  24.5   118    57    76 No         Yes
4 71393 male     27 White High Sch~  25.7   116    61    88 Yes        No
5 70458 fema~    35 Black 9 - 11th~  21.9   115    64    84 No         No
# ... with 3 more variables: SleepTrouble <fct>, MaritalStatus <fct>,
#   HealthGen <fct>
```

4.2.2 What are the variables?

We can use the `glimpse` function to get a short preview of the data.

```
glimpse(nh_adults)
```

```
Rows: 500
Columns: 14
$ ID          <int> 71531, 68613, 67064, 63924, 62840, 68058, 68936, 7118...
$ Sex         <fct> male, female, male, female, male, male, female, male,...
$ Age         <int> 35, 61, 31, 29, 60, 50, 36, 51, 54, 59, 59, 27, 44, 4...
$ Race        <fct> White, White, White, Black, White, White, Black, Whit...
$ Education   <fct> Some College, Some College, College Grad, High School...
$ BMI         <dbl> 22.4, 27.7, 26.6, 41.9, 35.8, 30.6, 30.5, 25.6, 21.8,...
$ SBP         <int> 143, 119, 110, 98, 127, NA, 119, 112, 126, 149, 122, ...
$ DBP         <int> 90, 86, 76, 56, 0, NA, 69, 70, 80, 89, 75, 78, 69, 78...
$ Pulse       <int> 84, 112, 86, 74, 110, NA, 60, 54, 78, 62, 82, 68, 76,...
$ PhysActive  <fct> Yes, No, Yes, No, No, No, No, Yes, Yes, Yes, No, Yes,...
$ Smoke100    <fct> No, No, Yes, Yes, Yes, Yes, No, Yes, No, No, No, No, ...
$ SleepTrouble <fct> Yes, No, No, Yes, Yes, Yes, No, No, No, No, No, No, N...
$ MaritalStatus <fct> NeverMarried, Widowed, Married, Separated, Married, M...
$ HealthGen   <fct> Good, Vgood, Good, Fair, Poor, Good, Good, NA, Good, ...
```

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
MaritalStatus	Marital Status	Married, Divorced
HealthGen	Self-report general health rating (5 lev.)	Vgood, Good

[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.

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.
- **MaritalStatus**: Married, Widowed, Divorced, Separated, NeverMarried or LivePartner (living with partner).
- **HealthGen**: Excellent, Vgood, Good, Fair or Poor.

Some details can be obtained using the `summary` function.

```
summary(nh_adults)
```

```
       ID             Sex            Age              Race       
 Min.   :62199   female:221   Min.   :21.00   Asian   : 42  
 1st Qu.:64522   male  :279   1st Qu.:31.00   Black   : 63  
 Median :67192                Median :42.00   Hispanic: 26  
 Mean   :67122                Mean   :41.91   Mexican : 38  
 3rd Qu.:69654                3rd Qu.:53.00   White   :313  
 Max.   :71911                Max.   :64.00   Other   : 18  

          Education        BMI              SBP              DBP       
 8th Grade     : 24   Min.   :17.30   Min.   : 84.0   Min.   :  0.00  
 9 - 11th Grade: 60   1st Qu.:23.80   1st Qu.:110.0   1st Qu.: 66.00  
 High School   : 81   Median :27.50   Median :118.0   Median : 72.00  
 Some College  :153   Mean   :28.48   Mean   :119.2   Mean   : 72.13  
 College Grad  :182   3rd Qu.:31.60   3rd Qu.:127.0   3rd Qu.: 78.00  
                      Max.   :63.30   Max.   :209.0   Max.   :103.00  
                       NA's   :5       NA's   :15      NA's   :15      
    Pulse       PhysActive Smoke100  SleepTrouble     MaritalStatus 
 Min.   : 40.00   No :215    No :297   No :380      Divorced    : 51  
 1st Qu.: 64.00   Yes:285    Yes:203   Yes:120      LivePartner : 51  
 Median : 72.00                                    Married     :259  
 Mean   : 73.41                                    NeverMarried:112  
 3rd Qu.: 82.00                                    Separated   : 16  
 Max.   :112.00                                    Widowed     : 11  
 NA's   :15                                                          
    HealthGen   
 Excellent: 50  
 Vgood    :154  
 Good     :184  
 Fair     : 49  
 Poor     : 14  
 NA's     : 49  
```

Note the appearance of `NA's` (indicating missing values) in some columns, and that some variables are summarized by a list of their (categorical) values and some (quantitative/numeric) variables are summarized with a minimum, quartiles and mean.

4.3 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 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.

4.3.1 A look at BMI (Body-Mass Index)

The definition of BMI (*body-mass index*) 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

4.4 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 a nominal variable with multiple unordered categories. So is `MaritalStatus`.
- 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        
 Min.   :21.00   Min.   :17.30   Min.   : 84.0   Min.   :  0.00  
 1st Qu.:31.00   1st Qu.:23.80   1st Qu.:110.0   1st Qu.: 66.00  
 Median :42.00   Median :27.50   Median :118.0   Median : 72.00  
 Mean   :41.91   Mean   :28.48   Mean   :119.2   Mean   : 72.13  
 3rd Qu.:53.00   3rd Qu.:31.60   3rd Qu.:127.0   3rd Qu.: 78.00  
 Max.   :64.00   Max.   :63.30   Max.   :209.0   Max.   :103.00  
                 NA's   :5       NA's   :15      NA's   :15      
     Pulse       
 Min.   : 40.00  
```

```
 1st Qu.: 64.00
 Median : 72.00
 Mean   : 73.41
 3rd Qu.: 82.00
 Max.   :112.00
 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 41.9,
- `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 5 of the BMI values, for example.

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  41.9     42 20953
```

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

And again, the Mean is just the Sum (20953), divided by the number of non-missing values of Age (500), or 41.906.

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    22
10    22
# ... 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 22 22 22 22 22 22 22 22 22 22 22 22 23 23 23 23 23
 [26] 23 23 23 23 23 23 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 25 25 25 25
 [51] 25 25 25 25 25 26 26 26 26 26 26 26 26 26 26 26 26 26 26 27 27 27 27 27 27 27
 [76] 27 27 27 27 27 27 27 27 28 28 28 28 28 28 28 28 28 28 28 28 28 28 29 29 29 29
[101] 29 29 29 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 31 31 31 31 31 31 31
[126] 31 31 31 31 31 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 33 33 33 33 33 33
[151] 33 33 33 33 33 33 33 34 34 34 34 34 34 34 35 35 35 35 35 35 35 35 35 36 36
[176] 36 36 36 36 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 38 38
[201] 38 38 38 38 38 38 38 38 38 39 39 39 39 39 39 39 39 39 39 39 39 40 40
[226] 40 40 40 40 40 40 40 40 40 41 41 41 41 41 41 41 41 41 41 41 41 42 42 42 42
[251] 42 42 42 42 42 42 42 42 43 43 43 43 43 43 43 43 43 43 43 43 43 43 43 44 44
[276] 44 44 44 44 44 44 44 44 44 45 45 45 45 45 45 45 45 45 45 45 45 45 45 45 46
[301] 46 46 46 46 46 46 47 47 47 47 47 47 47 47 47 47 48 48 48 48 48 48 49 49 49
[326] 49 49 49 49 49 49 49 49 49 49 49 49 49 50 50 50 50 50 50 50 50 50 50 50
[351] 50 50 50 50 51 51 51 51 51 51 51 51 51 52 52 52 52 52 52 52 52 53 53 53 53
```

```
[376] 53 53 53 53 53 53 53 53 53 53 54 54 54 54 54 54 54 54 54 54 54 54 55 55 55
[401] 55 55 55 55 55 55 55 55 56 56 56 56 56 56 56 56 56 56 56 56 56 56 56 57 57
[426] 57 57 57 57 57 57 57 57 57 58 58 58 58 58 58 58 58 58 59 59 59 59 59 59 59
[451] 59 59 59 59 59 59 59 59 60 60 60 60 60 60 60 60 60 60 60 61 61 61 61 61 61
[476] 61 61 61 61 61 62 62 62 62 62 63 63 63 63 63 63 63 63 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 250^{th} and 251^{st} 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          73.4              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>
1                        73.4                            72
```

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

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))
```

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

```
# A tibble: 44 x 2
     Age count
   <int> <int>
 1    37    18
 2    49    17
 3    24    15
 4    27    15
 5    30    15
 6    43    15
 7    45    15
 8    50    15
 9    56    15
10    59    15
# ... 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 (2019).

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

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

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()
```

```
[1] 21 64
```

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

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

```
[1] 17.3 63.3
```

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       
 Min.   :21.00   Min.   :17.30   Min.   : 84.0   Min.   :  0.00  
 1st Qu.:31.00   1st Qu.:23.80   1st Qu.:110.0   1st Qu.: 66.00  
 Median :42.00   Median :27.50   Median :118.0   Median : 72.00  
 Mean   :41.91   Mean   :28.48   Mean   :119.2   Mean   : 72.13  
 3rd Qu.:53.00   3rd Qu.:31.60   3rd Qu.:127.0   3rd Qu.: 78.00  
```

```
 Max.   :64.00   Max.   :63.30   Max.   :209.0   Max.   :103.00
                 NA's   :5      NA's   :15      NA's   :15
     Pulse
 Min.   : 40.00
 1st Qu.: 64.00
 Median : 72.00
 Mean   : 73.41
 3rd Qu.: 82.00
 Max.   :112.00
 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  152.  39.7  233.  123.  144.
```

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  12.3  6.30  15.3  11.1  12.0
```

5.3.4 Defining the Variance and Standard Deviation

Bock et al. (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.

[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.

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.

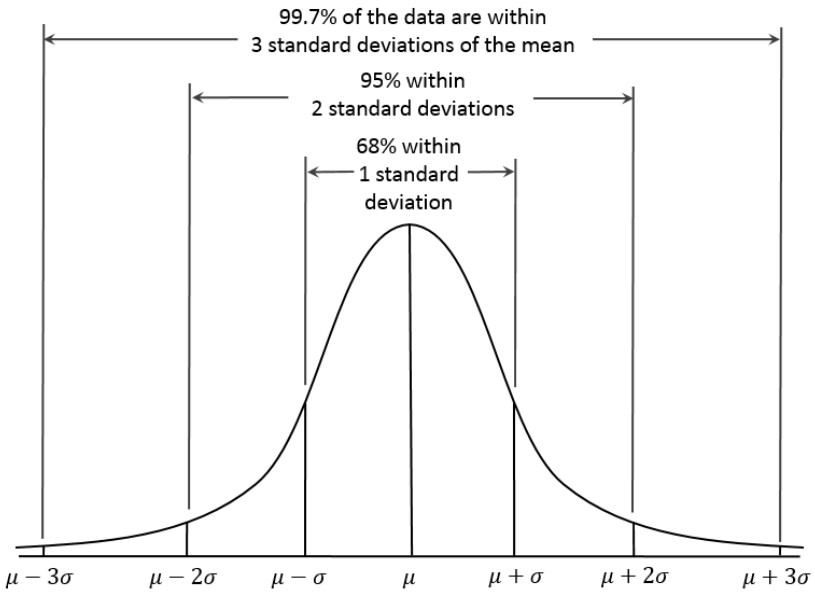


Figure 5.1: The Normal Distribution and the Empirical Rule

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

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.

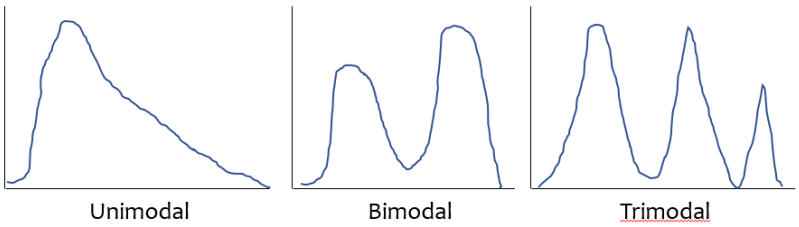


Figure 5.2: Unimodal and Multimodal Sketches

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:

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

- 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*.

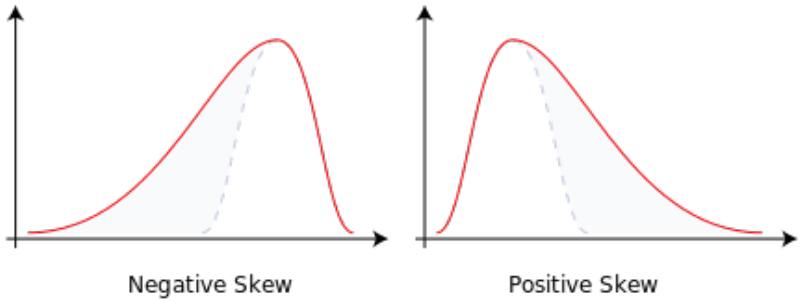


Figure 5.3: Negative (Left) Skew and Positive (Right) Skew

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.

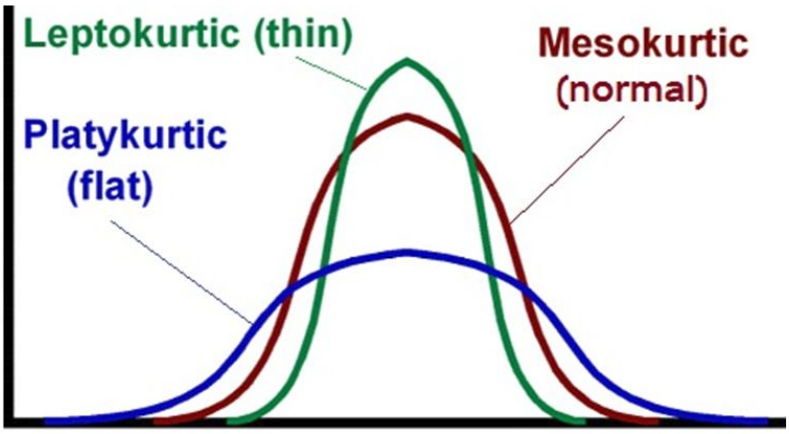


Figure 5.4: The Impact of Kurtosis

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
  84 110    118 127 209 119.2495 15.25735 485      15
```

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
  <int> <dbl>  <int> <dbl> <int> <dbl> <dbl> <int>   <int>
1    84   110    118   127   209  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  95 114    122 131.50 167 123.7273 18.86085  22      2
2 9 - 11th Grade  92 108    114 125.25 170 117.3833 13.66189  60      0
3   High School  91 112    119 129.00 209 122.6104 19.68111  77      4
4  Some College  85 110    119 128.00 165 119.1812 13.52778 149      4
5  College Grad  84 109    118 126.00 171 117.9209 14.26831 177      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)))
```

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

```
# A tibble: 5 x 10
  Education      min    Q1 median    Q3   max  mean    sd     n missing
  <fct>        <int> <dbl>  <dbl> <dbl> <int> <dbl> <dbl> <int>   <int>
1 8th Grade       95   114    122  132.   167  124.  18.9    22       0
2 9 - 11th Grade  92   108    114  125.   170  117.  13.7    60       0
3 High School     91   112    119  129    209  123.  19.7    77       0
4 Some College    85   110    119  128    165  119.  13.5   149       0
5 College Grad    84   109    118  126    171  118.  14.3   177       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  41.91 12.35   42.0   41.86 16.31 21.0  64.0    43  0.03
BMI      2 495  28.48  6.30   27.5   27.80  5.63 17.3  63.3    46  1.35
SBP      3 485 119.25 15.26  118.0  118.25 13.34 84.0 209.0   125  1.27
```

```
DBP     4 485  72.13 11.10   72.0   72.33  8.90  0.0 103.0   103 -0.58
Pulse   5 485  73.41 12.01   72.0   73.01 11.86 40.0 112.0    72  0.30
      kurtosis   se
Age      -1.20 0.55
BMI       3.32 0.28
SBP       4.63 0.69
DBP       3.58 0.50
Pulse     0.15 0.55
```

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    41.91    14.27       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
    495        5      198        1    28.48    6.704    20.70    21.90
     .25      .50      .75      .90      .95
   23.80    27.50    31.60    35.68    41.00

lowest : 17.3 17.8 18.2 18.3 18.4, highest: 47.7 54.1 54.4 56.8 63.3
---------------------------------------------------------------------------
SBP
      n  missing distinct     Info     Mean      Gmd      .05      .10
    485       15       73    0.999    119.2    16.18       98      102
     .25      .50      .75      .90      .95
     110      118      127      137      143

lowest :  84  85  91  92  93, highest: 170 171 182 202 209
---------------------------------------------------------------------------
DBP
      n  missing distinct     Info     Mean      Gmd      .05      .10
    485       15       61    0.999    72.13    12.02     54.0     58.0
     .25      .50      .75      .90      .95
    66.0     72.0     78.0     85.6     89.0

lowest :   0  41  42  44  45, highest:  98  99 100 102 103
---------------------------------------------------------------------------
Pulse
      n  missing distinct     Info     Mean      Gmd      .05      .10
    485       15       35    0.997    73.41    13.47     54.4     60.0
     .25      .50      .75      .90      .95
    64.0     72.0     82.0     88.0     94.0

lowest :  40  44  48  50  52, highest: 104 106 108 110 112
---------------------------------------------------------------------------
```

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.

5.5.4 Other options

The package `summarytools` has a function called `dfSummary` which I like and Dominic Comtois has also published Recommendations for Using summarytools with R Markdown. Note that this isn't really for Word documents.

The `naniar` package is helpful for wrangling and visualizing missing values, and checking imputations.

`DataExplorer` can be used for more automated exploratory data analyses (and some people also like `skimr`) and `visdat`, as well.

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 4.2 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, MaritalStatus) %>%
  summary()
```

```
    Sex           Race              Education   PhysActive Smoke100
 female:221   Asian   : 42   8th Grade     : 24   No :215    No :297
 male  :279   Black   : 63   9 - 11th Grade: 60   Yes:285    Yes:203
              Hispanic: 26   High School   : 81
              Mexican : 38   Some College  :153
              White   :313   College Grad  :182
              Other   : 18
 SleepTrouble     HealthGen        MaritalStatus
 No :380      Excellent: 50   Divorced    : 51
 Yes:120      Vgood    :154   LivePartner : 51
              Good     :184   Married     :259
              Fair     : 49   NeverMarried:112
              Poor     : 14   Separated   : 16
              NA's     : 49   Widowed     : 11
```

6.2 Tables to describe One Categorical Variable

Suppose we build a table (using the `tabyl` function from the `janitor` package) to describe the `HealthGen` distribution.

```
nh_adults %>%
    tabyl(HealthGen) %>%
    adorn_pct_formatting()
```

```
 HealthGen   n percent valid_percent
 Excellent  50   10.0%         11.1%
     Vgood 154   30.8%         34.1%
      Good 184   36.8%         40.8%
      Fair  49    9.8%         10.9%
      Poor  14    2.8%          3.1%
      <NA>  49    9.8%             -
```

Note how the missing (`<NA>`) values are not included in the `valid_percent` calculation, but are in the `percent` calculation. Note also the use of percentage formatting.

What if we want to add a total count, sometimes called the *marginal* total?

```
nh_adults %>%
    tabyl(HealthGen) %>%
    adorn_totals() %>%
    adorn_pct_formatting()
```

```
 HealthGen   n percent valid_percent
 Excellent  50   10.0%         11.1%
     Vgood 154   30.8%         34.1%
      Good 184   36.8%         40.8%
```

```
      Fair  49    9.8%         10.9%
      Poor  14    2.8%          3.1%
      <NA>  49    9.8%             -
     Total 500  100.0%        100.0%
```

What about marital status, which has no missing data in our sample?

```
nh_adults %>%
    tabyl(MaritalStatus) %>%
    adorn_totals() %>%
    adorn_pct_formatting()
```

```
 MaritalStatus   n percent
      Divorced  51   10.2%
   LivePartner  51   10.2%
       Married 259   51.8%
  NeverMarried 112   22.4%
     Separated  16    3.2%
       Widowed  11    2.2%
         Total 500  100.0%
```

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))
```

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

```
# A tibble: 6 x 2
  HealthGen count
  <fct>     <int>
1 Good        184
2 Vgood       154
3 Excellent    50
4 Fair         49
5 <NA>         49
6 Poor         14
```

6.4 `describe` in the `Hmisc` package

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

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

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

Value      female   male
Frequency     221    279
Proportion  0.442  0.558
--------------------------------------------------------------------------------
Race
       n  missing distinct
     500        0        6

lowest : Asian    Black    Hispanic Mexican  White
highest: Black    Hispanic Mexican  White    Other

Value         Asian    Black Hispanic  Mexican    White    Other
Frequency        42       63       26       38      313       18
Proportion    0.084    0.126    0.052    0.076    0.626    0.036
--------------------------------------------------------------------------------
Education
       n  missing distinct
     500        0        5

lowest : 8th Grade    9 - 11th Grade High School   Some College  College Grad
highest: 8th Grade    9 - 11th Grade High School   Some College  College Grad

Value         8th Grade 9 - 11th Grade    High School   Some College
Frequency            24            60            81            153
Proportion        0.048         0.120         0.162         0.306

Value        College Grad
Frequency             182
Proportion          0.364
--------------------------------------------------------------------------------
```

```
PhysActive
       n  missing distinct
     500        0        2

Value        No  Yes
Frequency   215  285
Proportion 0.43 0.57
---------------------------------------------------------------------------
Smoke100
       n  missing distinct
     500        0        2

Value         No   Yes
Frequency    297   203
Proportion 0.594 0.406
---------------------------------------------------------------------------
SleepTrouble
       n  missing distinct
     500        0        2

Value        No  Yes
Frequency   380  120
Proportion 0.76 0.24
---------------------------------------------------------------------------
HealthGen
       n  missing distinct
     451       49        5

lowest : Excellent Vgood     Good      Fair      Poor
highest: Excellent Vgood     Good      Fair      Poor

Value      Excellent     Vgood      Good      Fair      Poor
Frequency         50       154       184        49        14
Proportion     0.111     0.341     0.408     0.109     0.031
---------------------------------------------------------------------------
MaritalStatus
       n  missing distinct
     500        0        6

lowest : Divorced    LivePartner  Married     NeverMarried Separated
highest: LivePartner  Married     NeverMarried Separated   Widowed

Value       Divorced LivePartner    Married NeverMarried   Separated
Frequency         51         51        259         112          16
Proportion     0.102      0.102      0.518       0.224       0.032
```

```
Value           Widowed
Frequency            11
Proportion        0.022
--------------------------------------------------------------------------------
```

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 %>%
    tabyl(Education, SleepTrouble) %>%
    adorn_totals(where = c("row", "col"))
```

```
      Education  No Yes Total
      8th Grade  18   6    24
 9 - 11th Grade  45  15    60
    High School  62  19    81
   Some College 118  35   153
   College Grad 137  45   182
          Total 380 120   500
```

Note the use of `adorn_totals` to get the marginal counts, and how we specify that we want both the row and column totals. We can add a title for the columns with…

```
nh_adults %>%
    tabyl(Education, SleepTrouble) %>%
    adorn_totals(where = c("row", "col")) %>%
    adorn_title(placement = "combined")
```

```
 Education/SleepTrouble  No Yes Total
              8th Grade  18   6    24
          9 - 11th Grade  45  15    60
             High School  62  19    81
           Some College 118  35   153
           College Grad 137  45   182
                  Total 380 120   500
```

Often, we'll want to show percentages in a cross-tabulation like this. To get row percentages so that we can directly see the probability of `SleepTrouble = Yes` for each level of `Education`, we can use:

```
nh_adults %>%
    tabyl(Education, SleepTrouble) %>%
    adorn_totals(where = "row") %>%
```

```
    adorn_percentages(denominator = "row") %>%
    adorn_pct_formatting() %>%
    adorn_title(placement = "combined")
```

```
 Education/SleepTrouble    No   Yes
              8th Grade 75.0% 25.0%
         9 - 11th Grade 75.0% 25.0%
            High School 76.5% 23.5%
           Some College 77.1% 22.9%
           College Grad 75.3% 24.7%
                  Total 76.0% 24.0%
```

If we want to compare the distribution of `Education` between the two levels of `SleepTrouble` with column percentages, we can use the following...

```
nh_adults %>%
    tabyl(Education, SleepTrouble) %>%
    adorn_totals(where = "col") %>%
    adorn_percentages(denominator = "col") %>%
    adorn_pct_formatting() %>%
    adorn_title(placement = "combined")
```

```
 Education/SleepTrouble    No   Yes Total
              8th Grade  4.7%  5.0%  4.8%
         9 - 11th Grade 11.8% 12.5% 12.0%
            High School 16.3% 15.8% 16.2%
           Some College 31.1% 29.2% 30.6%
           College Grad 36.1% 37.5% 36.4%
```

If we want overall percentages in the cells of the table, so that the total across all combinations of `Education` and `SleepTrouble` is 100%, we can use:

```
nh_adults %>%
    tabyl(Education, SleepTrouble) %>%
    adorn_totals(where = c("row", "col")) %>%
    adorn_percentages(denominator = "all") %>%
    adorn_pct_formatting() %>%
    adorn_title(placement = "combined")
```

```
 Education/SleepTrouble    No   Yes  Total
              8th Grade  3.6%  1.2%   4.8%
         9 - 11th Grade  9.0%  3.0%  12.0%
            High School 12.4%  3.8%  16.2%
           Some College 23.6%  7.0%  30.6%
           College Grad 27.4%  9.0%  36.4%
                  Total 76.0% 24.0% 100.0%
```

Another common approach is to include both counts and percentages in a cross-tabulation. Let's look at the breakdown of `HealthGen` by `MaritalStatus`.

```
nh_adults %>%
    tabyl(MaritalStatus, HealthGen) %>%
    adorn_totals(where = c("row")) %>%
    adorn_percentages(denominator = "row") %>%
    adorn_pct_formatting() %>%
    adorn_ns(position = "front") %>%
    adorn_title(placement = "combined") %>%
    knitr::kable()
```

MaritalStatus/HealthGen	Excellent	Vgood	Good	Fair	Poor	NA_
Divorced	7 (13.7%)	14 (27.5%)	20 (39.2%)	5 (9.8%)	2 (3.9%)	3 (5.9
LivePartner	1 (2.0%)	18 (35.3%)	16 (31.4%)	11 (21.6%)	1 (2.0%)	4 (7.8
Married	23 (8.9%)	84 (32.4%)	102 (39.4%)	15 (5.8%)	4 (1.5%)	31 (1
NeverMarried	14 (12.5%)	31 (27.7%)	43 (38.4%)	13 (11.6%)	3 (2.7%)	8 (7.1
Separated	4 (25.0%)	4 (25.0%)	1 (6.2%)	4 (25.0%)	1 (6.2%)	2 (12
Widowed	1 (9.1%)	3 (27.3%)	2 (18.2%)	1 (9.1%)	3 (27.3%)	1 (9.1
Total	50 (10.0%)	154 (30.8%)	184 (36.8%)	49 (9.8%)	14 (2.8%)	49 (9

What if we wanted to ignore the missing `HealthGen` values? Most often, I filter down to the complete observations.

```
nh_adults %>%
    filter(complete.cases(MaritalStatus, HealthGen)) %>%
    tabyl(MaritalStatus, HealthGen) %>%
    adorn_totals(where = c("row")) %>%
    adorn_percentages(denominator = "row") %>%
    adorn_pct_formatting() %>%
    adorn_ns(position = "front") %>%
    adorn_title(placement = "combined")
```

```
 MaritalStatus/HealthGen  Excellent      Vgood       Good      Fair
             Divorced  7 (14.6%)  14 (29.2%)  20 (41.7%)  5 (10.4%)
          LivePartner  1  (2.1%)  18 (38.3%)  16 (34.0%) 11 (23.4%)
              Married 23 (10.1%)  84 (36.8%) 102 (44.7%) 15  (6.6%)
         NeverMarried 14 (13.5%)  31 (29.8%)  43 (41.3%) 13 (12.5%)
            Separated  4 (28.6%)   4 (28.6%)   1  (7.1%)  4 (28.6%)
              Widowed  1 (10.0%)   3 (30.0%)   2 (20.0%)  1 (10.0%)
                Total 50 (11.1%) 154 (34.1%) 184 (40.8%) 49 (10.9%)
       Poor
  2  (4.2%)
  1  (2.1%)
  4  (1.8%)
  3  (2.9%)
  1  (7.1%)
  3 (30.0%)
 14  (3.1%)
```

For more on working with `tabyls`, see the vignette in the `janitor` package. There you'll find a complete list of all of the `adorn` functions, for example.

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            3    11   17    3    0
  Black            8    11   19   11    6
  Hispanic         3     3   11    4    1
  Mexican          2     8   17    6    3
  White           33   113  114   22    4
  Other            1     8    6    3    0
```

6.5.1 Cross-Classifying Three Categorical Variables

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

```
nh_adults %>%
    tabyl(Smoke100, PhysActive, SleepTrouble) %>%
    adorn_title(placement = "top")
```

```
$No
          PhysActive
 Smoke100         No Yes
       No         99 142
      Yes         62  77

$Yes
          PhysActive
 Smoke100         No Yes
       No         21  35
      Yes         33  31
```

The result here is a tabyl of `Smoke100` (rows) by `PhysActive` (columns), split into a list by `SleepTrouble`. Another approach to get the same table is:

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

```
, , SleepTrouble = No

        PhysActive
Smoke100  No Yes
     No   99 142
     Yes  62  77
```

```
, , SleepTrouble = Yes

        PhysActive
Smoke100  No Yes
     No   21  35
     Yes  33  31
```

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                    21  33
Yes        No                    142  77
           Yes                    35  31
```

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   99 142
     Yes  62  77

, , SleepTrouble = Yes

        PhysActive
Smoke100  No Yes
     No   21  35
     Yes  33  31
```

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,	1,167
Arizona	10.7 | (8.3	(8.3-13.6) | 1,52	1,520
Arkansas	17.8 | (15.7-	(15.7-20.1) |	1,470
Connecticut	12.3 |	(10.2-14.7) | 2,2	2,270
Delaware	14.2 | (12	(12.9-15.6) |	2,475
Florida	11.6 | (10.5-1	(10.5-12.8) |	5,491
…			
Wisconsin	11.6 | (	(9.7-13.9) | 2,7	2,771
Wyoming	10.7	(9.4-12.2) | 2,910	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.

6.7 Gaining Control over Tables in R: the `gt` package

With the `gt` package, anyone can make wonderful-looking tables using the R programming language. The `gt` package is described in substantial detail at https://gt.rstudio.com/ and we'll get started with it soon.

Bibliography

Baumer, B. S., Kaplan, D. T., and Horton, N. J. (2017). *Modern Data Science with R*. CRC Press, Boca Raton, FL.

Bock, D. E., Velleman, P. F., and De Veaux, R. D. (2004). *Stats: Modelling the World*. Pearson Addison-Wesley, Boston MA.

Gelman, A. and Nolan, D. (2017). *Teaching Statistics: A Bag of Tricks*. Oxford University Press, Oxford, UK, second edition.

Grolemund, G. and Wickham, H. (2019). *R for Data Science*. O'Reilly.

Ismay, C. and Kim, A. Y. (2019). *ModernDive: Statistical Inference via Data Science*.

Norman, G. R. and Streiner, D. L. (2014). *Biostatistics: The Bare Essentials*. People's Medical Publishing House, fourth edition.

Vittinghoff, E., Glidden, D. V., Shiboski, S. C., and McCulloch, C. E. (2012). *Regression Methods in Biostatistics: Linear, Logistic, Survival, and Repeated Measures Models*. Springer-Verlag, Inc., second edition.

Wainer, H. (1997). *Visual Revelations: Graphical Tales of Fate and Deception from Napoleon Bonaparte to Ross Perot*. Springer-Verlag, New York.

Wainer, H. (2005). *Graphic Discovery: A Trout in the Milk and Other Visual Adventures*. Princeton University Press, Princeton, NJ.

Wainer, H. (2013). *Medical Illuminations: Using Evidence, Visualization and Statistical Thinking to Improve Healthcare*. Oxford University Press, New York.