Intermediate R: `tidyverse`

PSYC 2020-A01 / PSYC 6022-A01 | 2025-09-19 | Lab 5

Jessica Helmer

Outline

- Assignment 4 Review
- `tidyverse` Workflow
- Full Analysis

Learning objectives:

R: `tidyverse`

Housekeeping

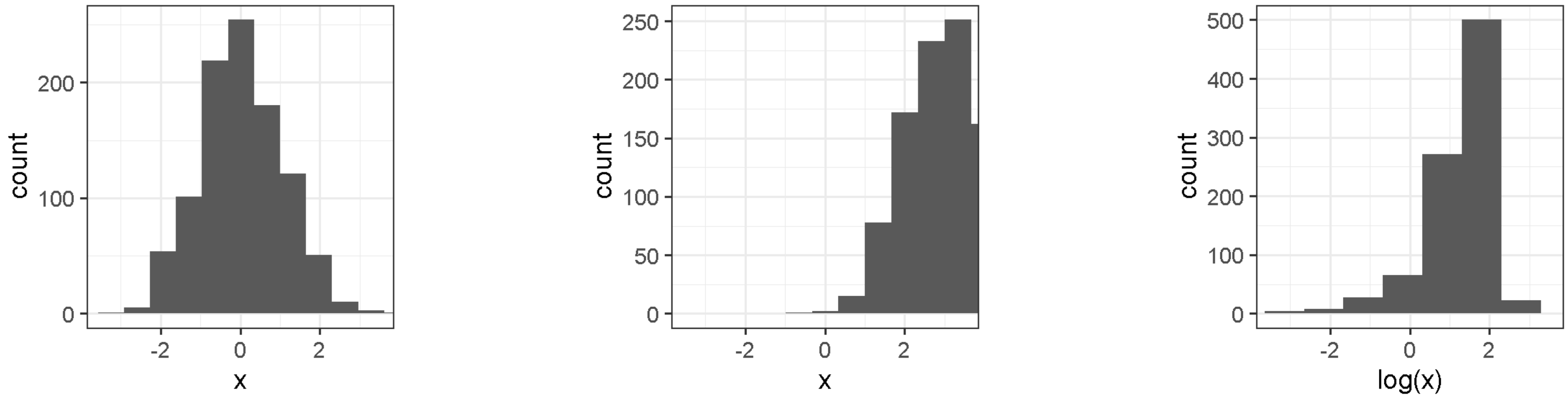
How was the exam?

How was `posit::conf(2025)`?

Assignment 4 Review

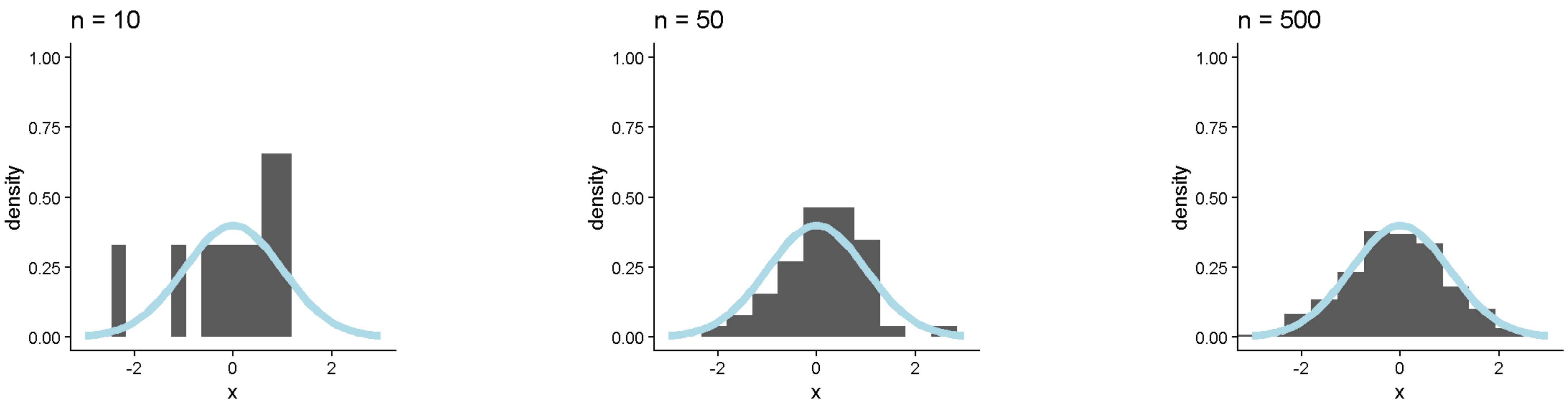
Skew vs Shift

Plot Code



Assignment 4 Review

More Draws: More Representative



Assignment 4 Review

RMarkdown and Working Directories

We won't have to worry about this today, but some things we can try:

Make sure your .Rmd assignment file is in the same folder as your .Rproj file

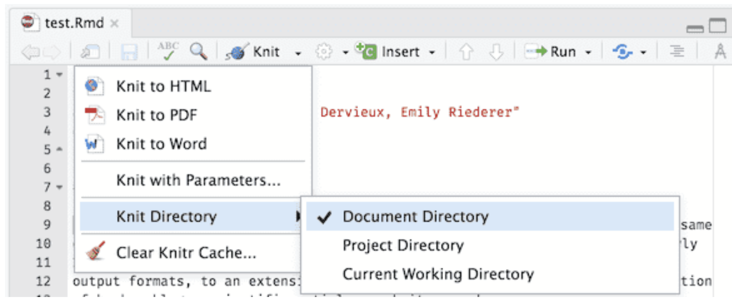
Default directory for code chunks is folder that contains .rmd document

If .Rmd file is in the same folder as your .Rproj file and `getwd()` returns a different directory than your project folder, we can try

```
knitr::opts_knit$set(root.dir = '/tmp')
```

or

```
here::i_am("any_project_subfolders/assign
```

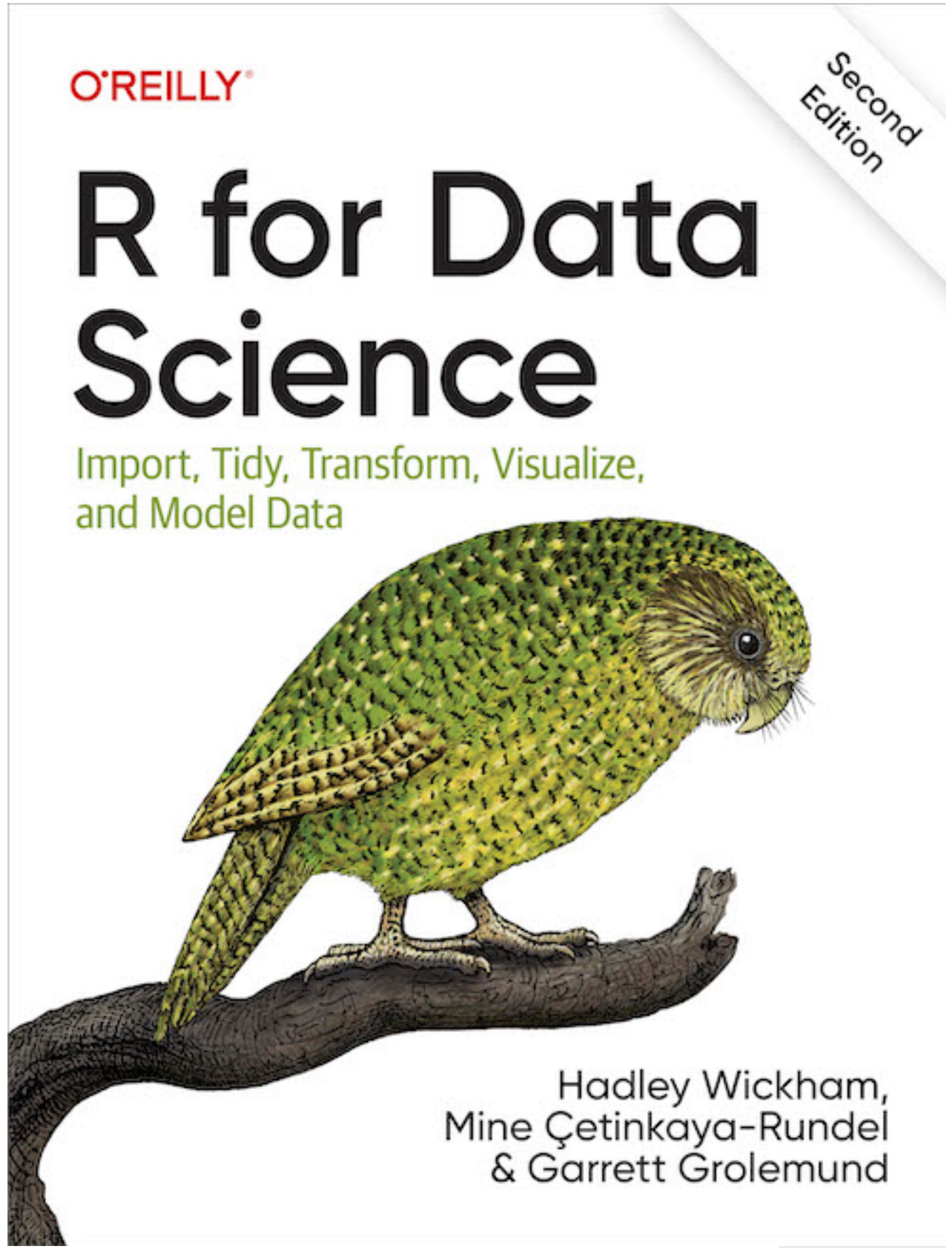


The `tidyverse`

What is the `tidyverse`???

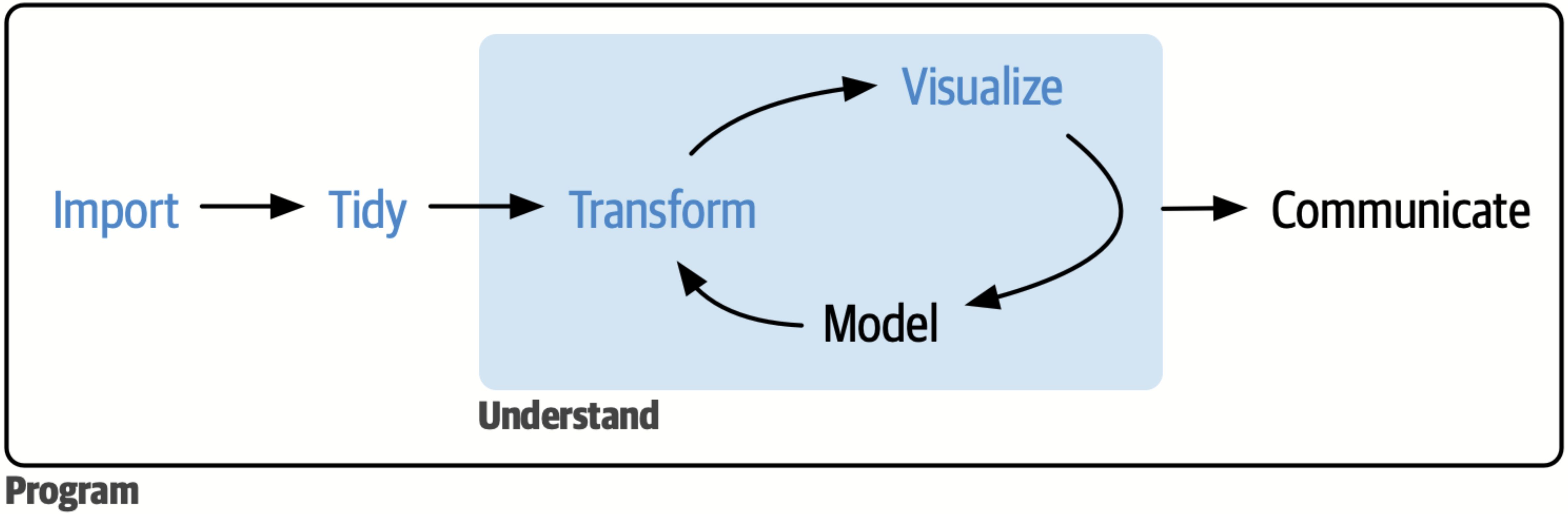
`tidyverse` webpage

R packages made for data science



Whole Game

Pieces of the "whole game" of data science



Whole Game

Today's plan:

- Introduce each of these principles
- Learn about `tidyverse` syntax
- Cover some examples of `tidyverse` functions that fall into each category
- Conduct our first "full analysis" by following the tidy workflow

Whole Game

Import → Tidy → Transform → Visualize → Model → Communicate

Import

Bringing our data into R

We've done this with the `read()` function family and / or the `rio` package

…Neither of those are `tidyverse` packages

But that's fine! This step is the simplest

Import: Today's dataset

Dataset on pulse rates before and after exercise

Can import from the internet

```r
pulse_data <- rio::import("http://www.statsci.org/data/oz/ms212.txt")
```

Or download the .rds file from Canvas

```r
pulse_data <- rio::import(here::here("labs", "Lab 5 - Intermediate R", "pulserates.rds"))
```

.rds is a file extension for R objects

Import: Today's dataset

Variable	Description
Height	Height (cm)
Weight	Weight (kg)
Age	Age (years)
Gender	Sex (1 = male, 2 = female)
Smokes	Regular smoker? (1 = yes, 2 = no)
Alcohol	Regular drinker? (1 = yes, 2 = no)
Exercise	Frequency of exercise (1 = high, 2 = moderate, 3 = low)
Ran	Whether the student ran or sat between the measurements (1 = ran, 2 = sat)
Pulse1	First pulse measurement (rate per minute)
Pulse2	Second pulse measurement (rate per minute)
Year	Year of class (93–98)

Import: Today's dataset

Let's check out our data

R Code Start Over Run Code

```
# let's look at our data!

# summary(), str(), head()
# psych::describe()
```

Whole Game

Import → **Tidy** → Transform → Visualize → Model → Communicate

Tidy

Refers to *tidy data*

> “Happy families are all alike; every unhappy family is unhappy in its own way.”
> — Leo Tolstoy

> “Tidy datasets are all alike, but every messy dataset is messy in its own way.”
> — Hadley Wickham

Tidy data follow a consistent set of organizational principles

More work up front but **much** easier to work with after

Tidy

	PersonId	Height	Weight	Year
1	1	173	57	93
2	2	179	58	93
3	3	167	62	93
4	4	195	84	93
5	5	173	64	93
6	6	184	74	93

	PersonId	Year	Variable	Value
1	1	93	Height	173
2	1	93	Weight	57
3	2	93	Height	179
4	2	93	Weight	58
5	3	93	Height	167
6	3	93	Weight	62

	PersonId	HeightToWeight	Year
1	1	173/57	93
2	2	179/58	93
3	3	167/62	93
4	4	195/84	93
5	5	173/64	93
6	6	184/74	93

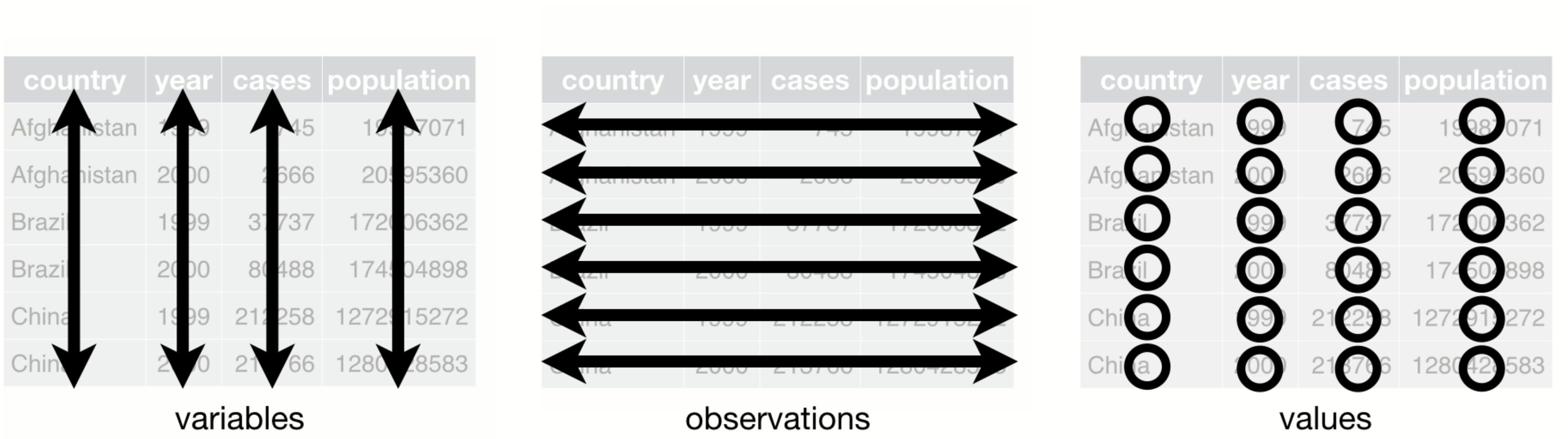
All of these dataframes contain the same information, but one of them is much easier to work with…

Why?

Tidy: Principles

Three rules to a tidy dataset

1. Each variable is a column; each column is a variable.
2. Each observation is a row; each row is an observation.
3. Each value is a cell; each cell is a single value.



Tidy: Why tidy?

1. Consistency! Any sort of consistency in your data management will be beneficial.

What's often the primary argument of functions we've used?

2. R loves vectors! Having variables in columns plays well with many functions

Tidy: Is our data tidy?

```
1  head(pulse_data, n = 10)
```

```
   Height Weight Age Gender Smokes Alcohol Exercise Ran Pulse1 Pulse2 Year
1     173     57  18      2      2       1        2   2     86     88   93
2     179     58  19      2      2       1        2   1     82    150   93
3     167     62  18      2      2       1        1   1     96    176   93
4     195     84  18      1      2       1        1   2     71     73   93
5     173     64  18      2      2       1        3   2     90     88   93
6     184     74  22      1      2       1        3   1     78    141   93
7     162     57  20      2      2       1        2   2     68     72   93
8     169     55  18      2      2       1        2   2     71     77   93
9     164     56  19      2      2       1        1   2     68     68   93
10    168     60  23      1      2       1        2   1     88    150   93
```

1. Each variable is a column; each column is a variable.
2. Each observation is a row; each row is an observation.
3. Each value is a cell; each cell is a single value.

Tidy: For later…

We will learn ways to make untidy data tidy in later classes

For now, we are good, and can proceed with our analysis!

Whole Game

Import → Tidy → **Transform** → Visualize → Model → Communicate

Transform

Usually, our data doesn't come to us with the variables exactly as we want them

Sometimes we need…

- New variables, based on our existing data or otherwise
- Summarized version of our dataset
- Reordered variables
- Etc., etc., etc. Any others?

Transform: `tidyverse` syntax

The `tidyverse` website mentioned that each package shares a common language for their functions

Based off of `dplyr` (one of the packages) verbs

1. The first argument is always a data frame.
2. The subsequent arguments typically describe which columns to operate on using the variable names (without quotes).
3. The output is always a new data frame.

Transform: Load the `tidyverse`

Let's get ready to actually use some `tidyverse` code!

```r
install.packages("tidyverse")
```

```r
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   4.0.0     ✔ tibble    3.2.1
✔ lubridate 1.9.3     ✔ tidyr     1.3.1
✔ purrr     1.0.2
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
errors
```

> **Note**
>
> Note the `Conflicts` section! Remember when we talked about overwriting?

Transform: More `tidyverse` syntax

One more thing…

`tidyverse` verbs work best in a **pipe**: |>

How we've been using functions:

```
output <- function(input)
```

Functions in a pipe:

```
output <- input |> function()
```

Why is this helpful?

Many functions together:

```
anotherfunction(otherfunction(function(input)))
```

Functions in a pipe:

```
input |>
  function() |>
  otherfunction() |>
  another function()
```

Transform: Goals for our data

In what ways should we transform our data?

First, we need to figure out our questions!

Transform: Goals for our data

Variable	Description
Height	Height (cm)
Weight	Weight (kg)
Age	Age (years)
Gender	Sex (1 = male, 2 = female)
Smokes	Regular smoker? (1 = yes, 2 = no)
Alcohol	Regular drinker? (1 = yes, 2 = no)
Exercise	Frequency of exercise (1 = high, 2 = moderate, 3 = low)
Ran	Whether the student ran or sat between the measurements (1 = ran, 2 = sat)
Pulse1	First pulse measurement (rate per minute)
Pulse2	Second pulse measurement (rate per minute)
Year	Year of class (93–98)

Transform: Goals for our data

My questions:

1. Is the difference in pulse between the first and second measurement points different for people who ran or not?
2. Does that difference vary by whether or not someone is a regular smoker?
3. What is the relationship between first and second pulse measurements? Does this differ by smoking status?
4. Class question!! (time pending)

Transform: Gameplan

Transformation goals:

1. Make a variable with the difference between the first and second pulse measurement points
2. Code some of our numeric variables into their categories
3. Subset our data to only include the relevant columns
4. Summarize information based on our questions

Transform: Factor Variables

New data type (to us): **factor** variable

How R works with categorical data

Sets fixed possibilities of values that the variable can take on

If you have a variable with days of the week…

```
days <- c("Tues", "Wed", "Thurs")
```

Nothing preventing typos and no sensible sorting

```
moredays <- c("Teus", "Wed", "Thrus")
sort(days)
```

```
[1] "Thurs" "Tues"  "Wed"
```

Factors help with both. Start by creating a vector of *levels*

```
weekday_levels <- c("Mon", "Tues", "Wed", "Thurs'
```

And now make a factor

```
days_fct <- factor(days, levels = weekday_levels)
```

Transform: Factor Variables

Will convert to NA if not valid level

```r
factor(moredays, levels = weekday_levels)
```

```
[1] <NA> Wed  <NA>
Levels: Mon Tues Wed Thurs Fri
```

Better sorting

```r
sort(days_fct)
```

```
[1] Tues  Wed   Thurs
Levels: Mon Tues Wed Thurs Fri
```

Transform: Some relevant functions

`mutate()`	adds, changes, renames columns
`factor()`	converts numeric to factor data type
`select()`	selects specified columns
`filter()`	selects specified rows
`summarize()`	summarizes data

Switching to RStudio…

Transform: Sample code

Goal 1, 2, and 3:

1. Make a variable with the difference between the first and second pulse measurement points
2. Code some of our numeric variables into their categories
3. Subset our data to only include the relevant columns

```r
pulse_subset <- pulse_data |>
  mutate(PulseDiff = Pulse2 - Pulse1,
         Smokes = factor(Smokes, levels = c(1, 2), labels = c("Yes", "No")),
         Ran = factor(Ran, levels = c(1, 2), labels = c("Ran", "Sat"))) |>
  select(Smokes, Ran, Pulse1, Pulse2, PulseDiff)
```

Transform: Sample code

Goal 4:

4. Summarize information based on our questions

```
pulse_subset |>
  summarize(.by = Ran,
            PulseDiffMean = mean(PulseDiff, na.rm=T))
```

```
  Ran PulseDiffMean
1 Sat       -1.0000
2 Ran       51.3913
```

```
pulse_subset |>
  summarize(.by = c(Ran, Smokes),
            PulseDiffMean = mean(PulseDiff, na.rm=T))
```

```
  Ran Smokes PulseDiffMean
1 Sat     No    -0.8363636
2 Ran     No    51.6511628
3 Sat    Yes    -2.1250000
4 Ran    Yes    47.6666667
```

Whole Game

Import → Tidy → Transform → **Visualize** → Model → Communicate

Visualize

So many numbers! Can we please visualize our data?

> "The simple graph has brought more information to the data analyst's mind than any other device." — John Tukey

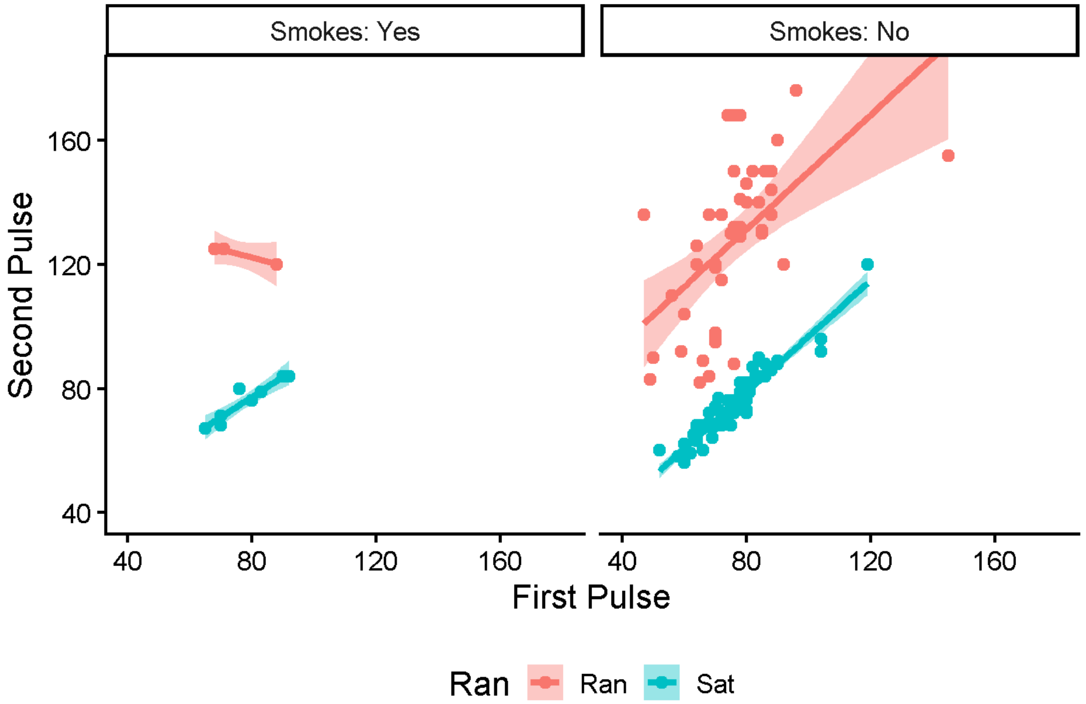
Yes! With the `ggplot2` package

A bit of a different plotting system than we've done before

But much more expansive and flexible (and much nicer looking, if you ask me!)

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?



Let's begin at the end:

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code

When we just call the `ggplot()` function with our data as the first argument, we get this blank area

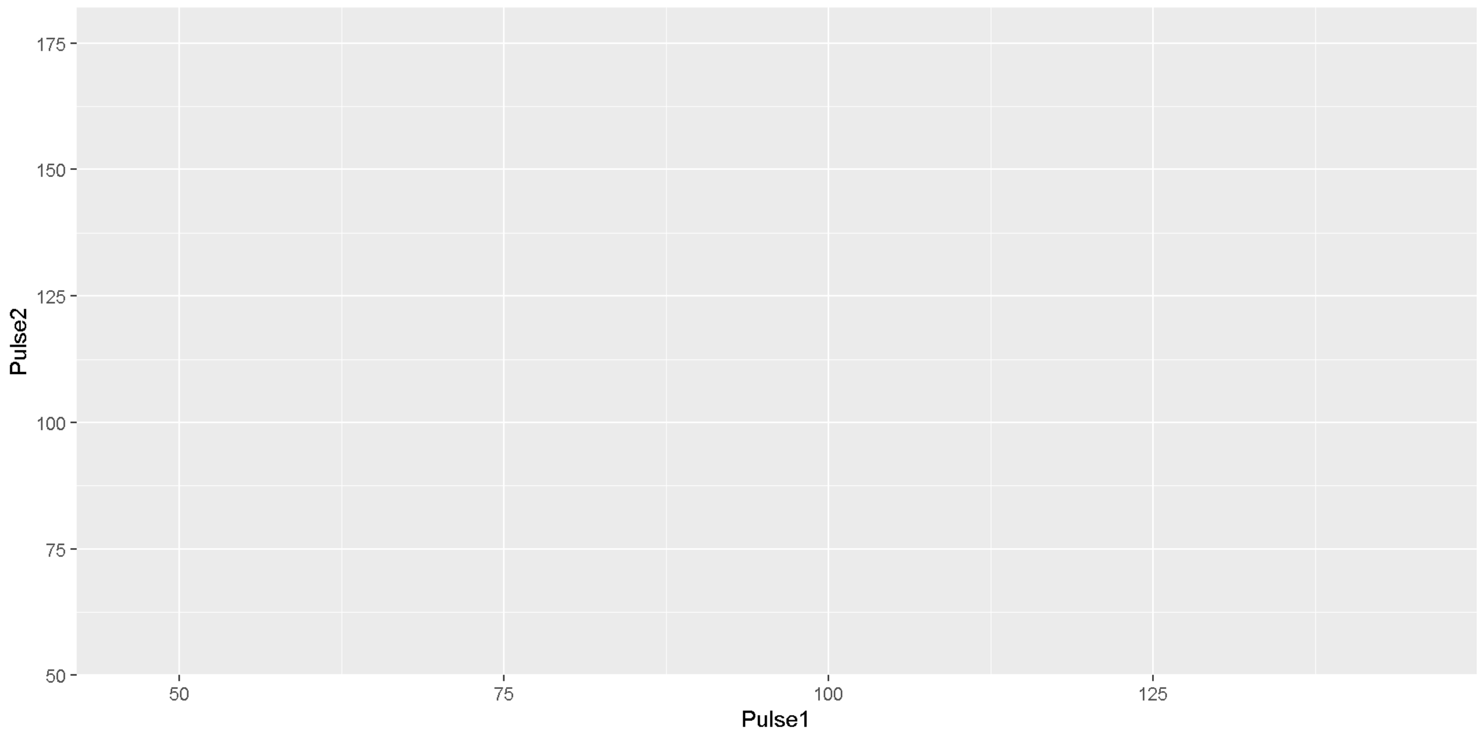
`ggplot()` works by adding layers and layers onto a plot object

> This is not a very exciting plot, but you can think of it like an empty canvas you'll paint the remaining layers of your plot onto. — R4DS

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code



Next, we want to tell `ggplot()` what information we want in our plot and where we want it

We do this with the `aes()` ("aesthetics") function in the `mapping =` argument

How variables are *mapped* onto specific parts of the plot

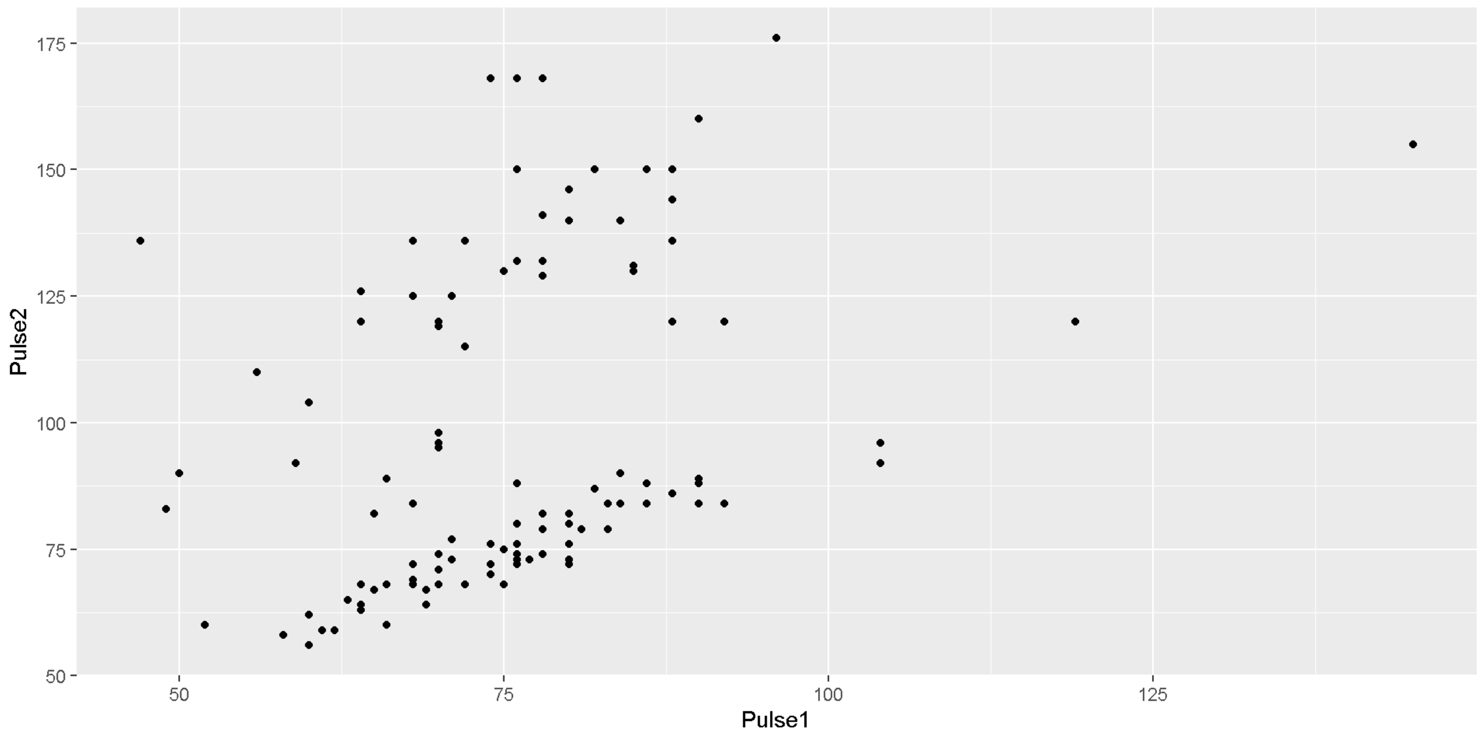
To investigate the relationship between the first and second pulse measurements, we can choose to put `Pulse1` on the x-axis and `Pulse2` on the y-axis

What do we get?

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code



Still no data, though

Need to define some **geom**, a geometric display of our data

Will usually start with `geom_`

`geom_line()`	line graph
`geom_bar()`	bar graph
`geom_boxplot()`	boxplot
`geom_point()`	scatterplot

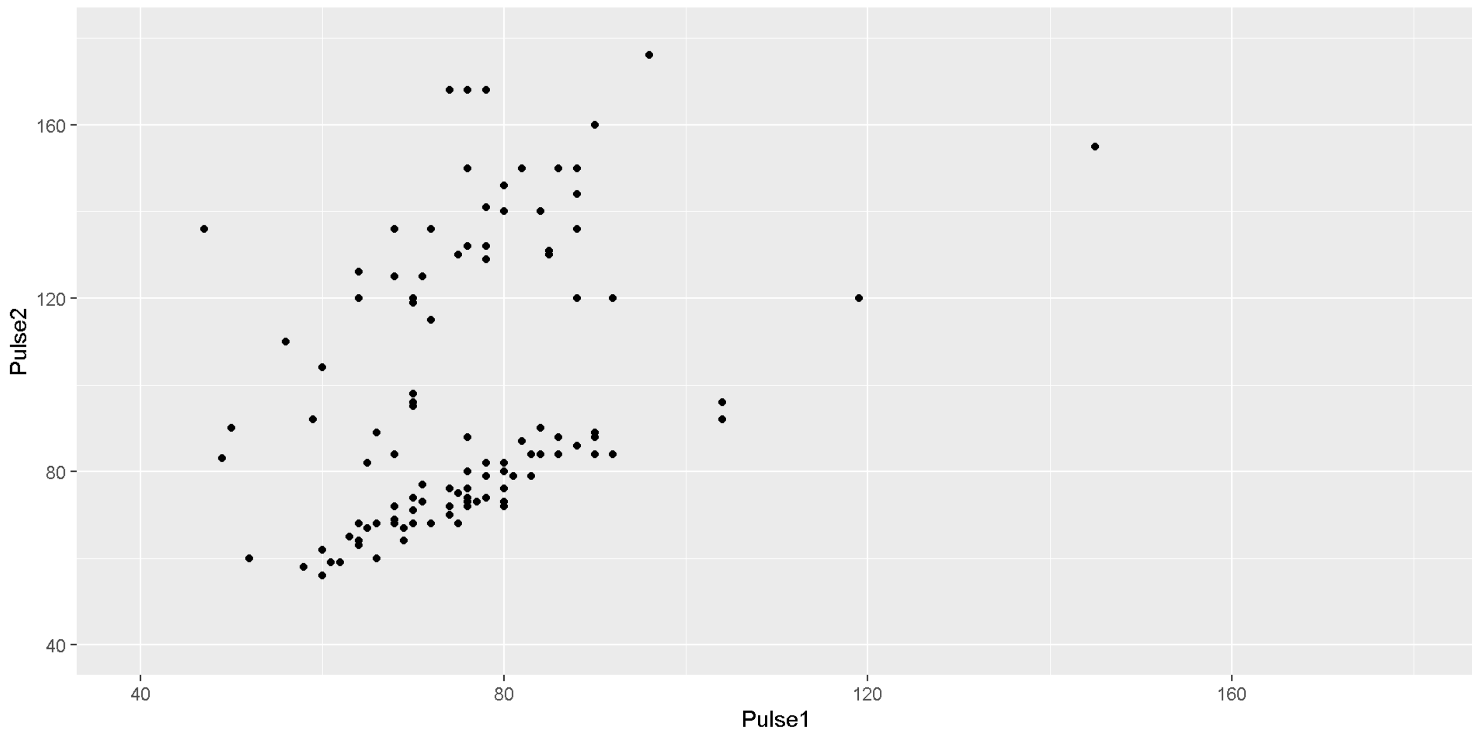
Can start addressing questions here!

> **Note**
>
> Take note of the missing values warning!

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code



These variables are conceptually similar and should probably have the same scale

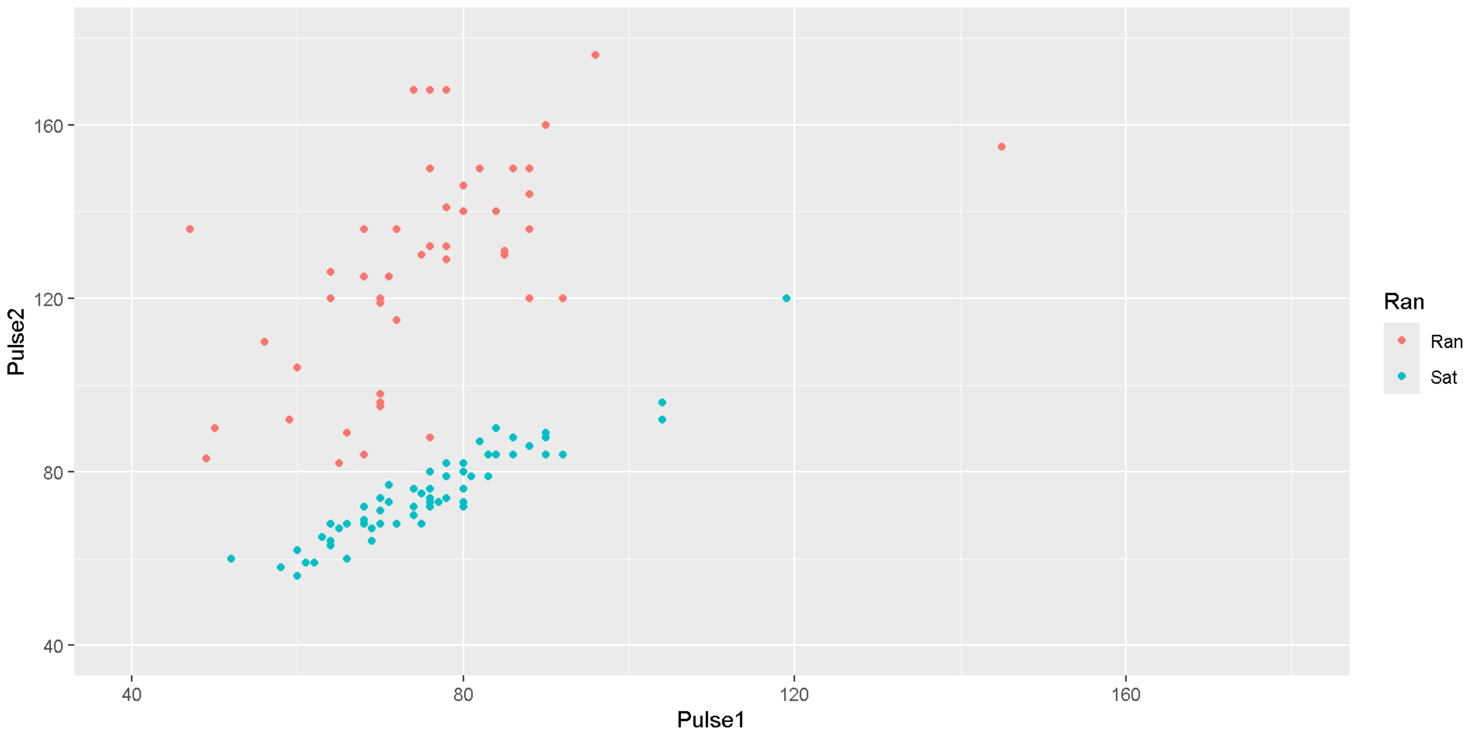
Automatically generated axes are a good place to start

But good practice to consider them each time

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code



Adding aesthetics and layers

Who in this group are the people who ran?

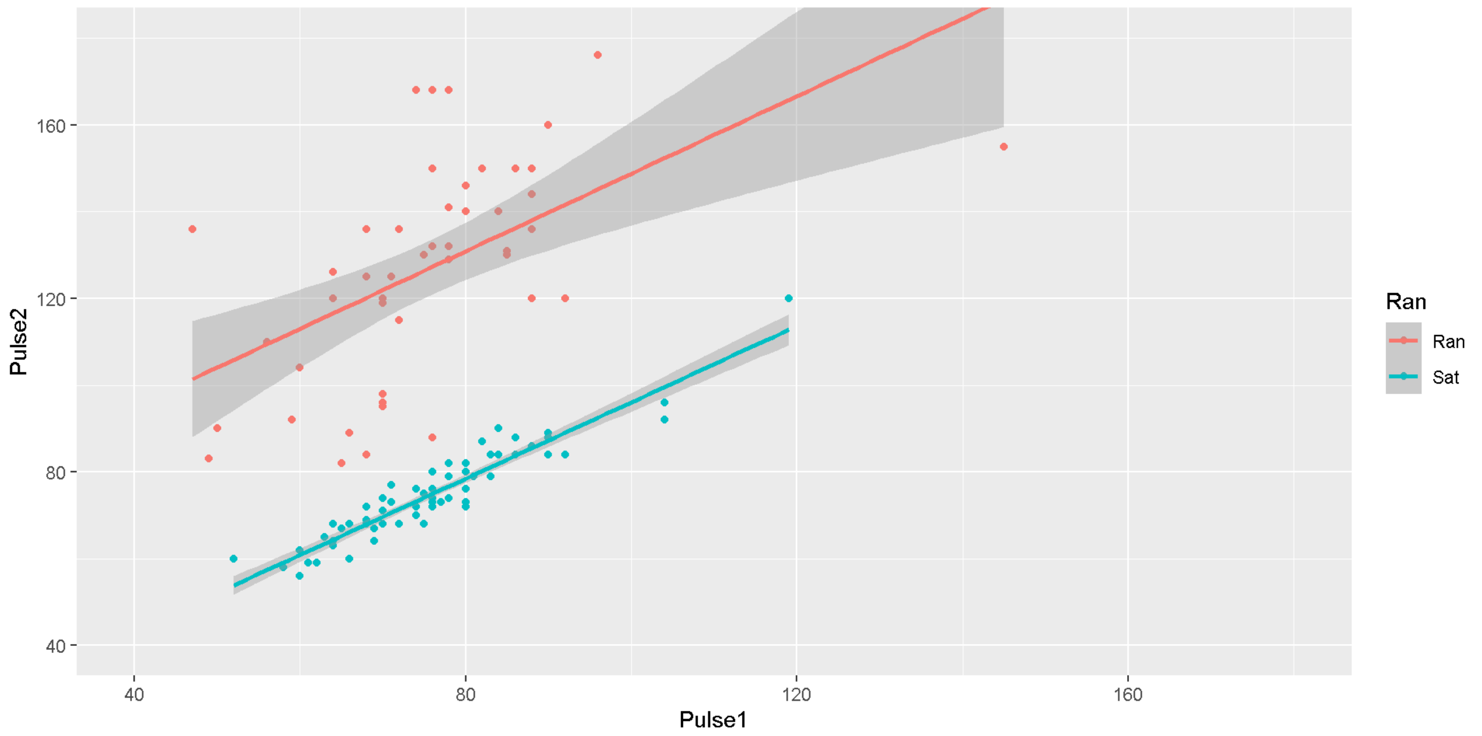
Start: do we need to modify the aesthetic or the geom?

What can we tell from this?

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code



Adding aesthetics and layers

Let's say we want to add a line to better represent the overall trend

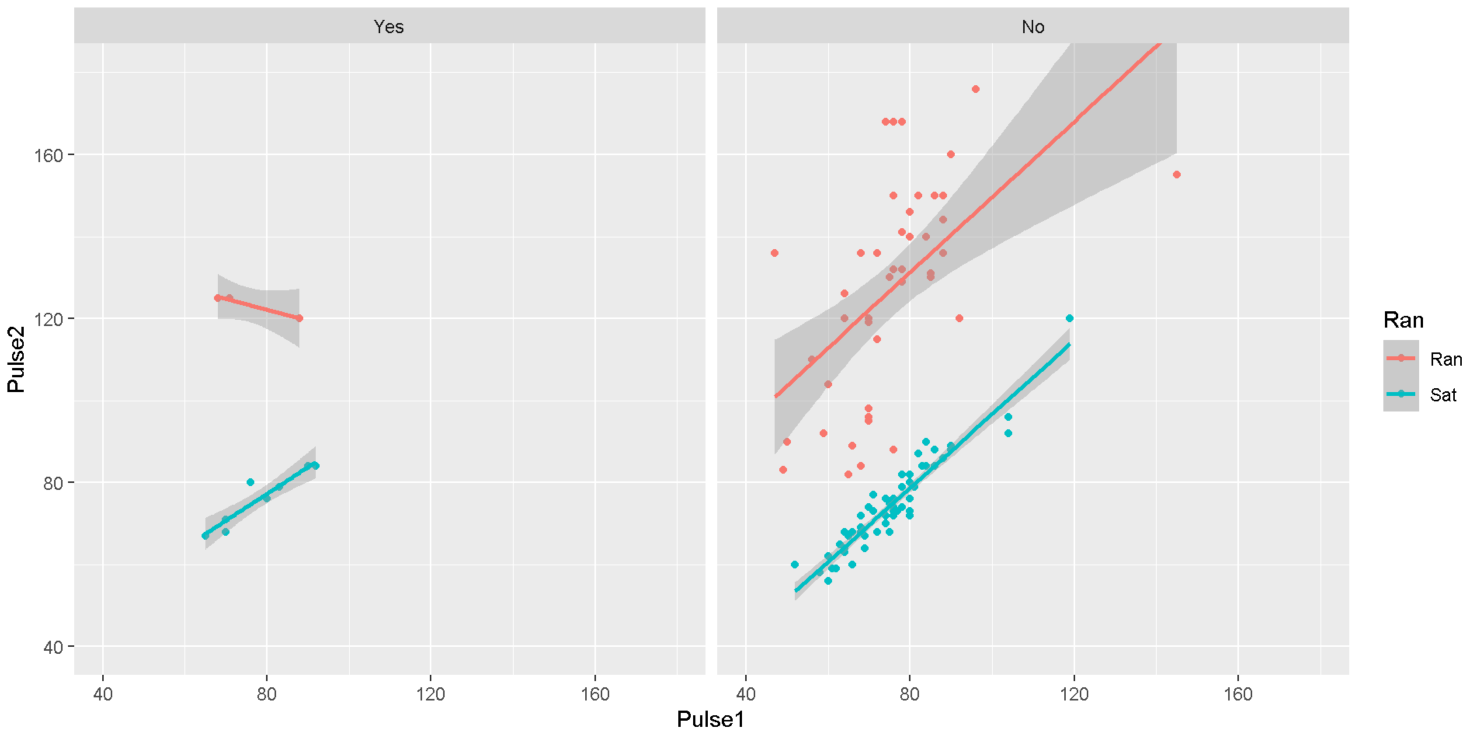
Start: do we need to modify the aesthetic or the geom?

What can we tell from this?

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code



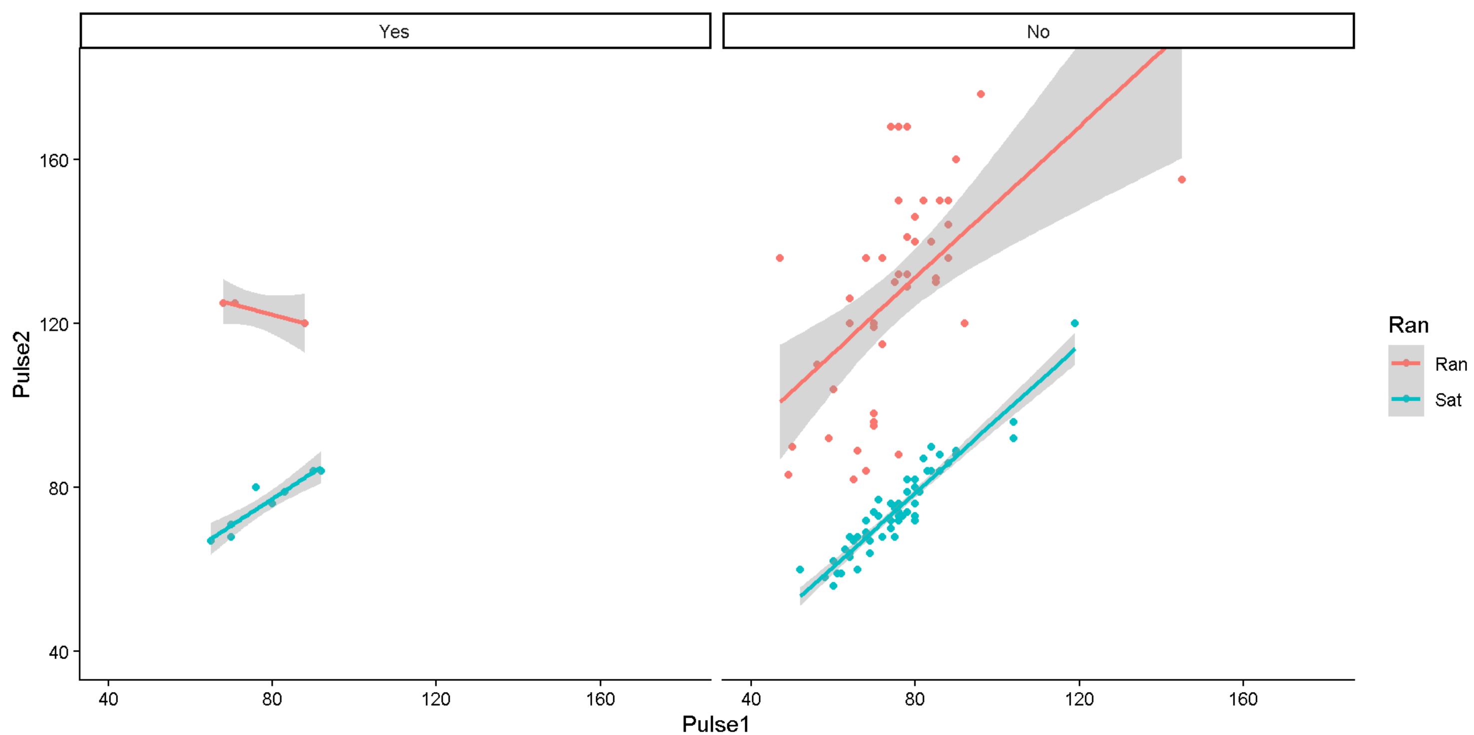
Let's say we want to look at this for smokers vs non-smokers

What can we tell from this?

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code



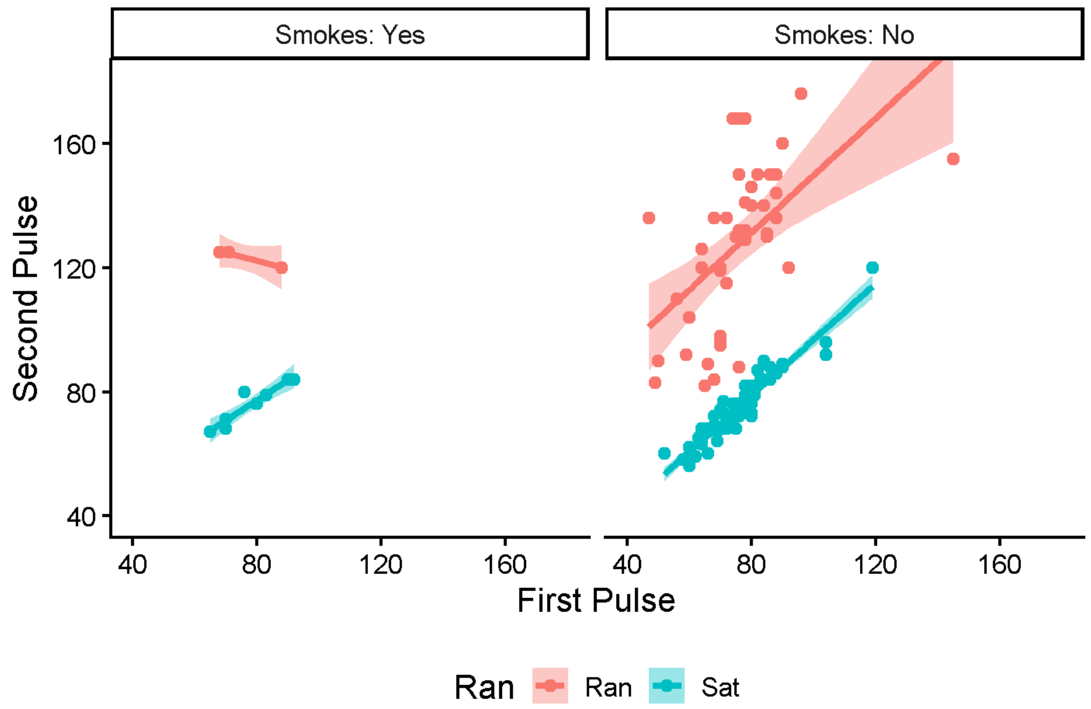
Let's say we want this plot to look nicer!!

Built in themes

Visualize: `ggplot2`

Let's briefly explore a different question: what is the relationship between the pulse measurements at the two different time points?

Plot | Code



Let's say we want this plot to look nicer!!

Additional theming with `theme()`, `labs()`, and various arguments

Whole Game

Import → Tidy → Transform → Visualize → **Model** → Communicate

Model

At this point, we could take the statistical models we know and apply them to the data

We haven't learned many yet, but we have learned some!

`mean()`, `sd()` could combine to model the distribution of your data, etc.

We will do more modeling later!

Whole Game

Import → Tidy → Transform → Visualize → Model → **Communicate**

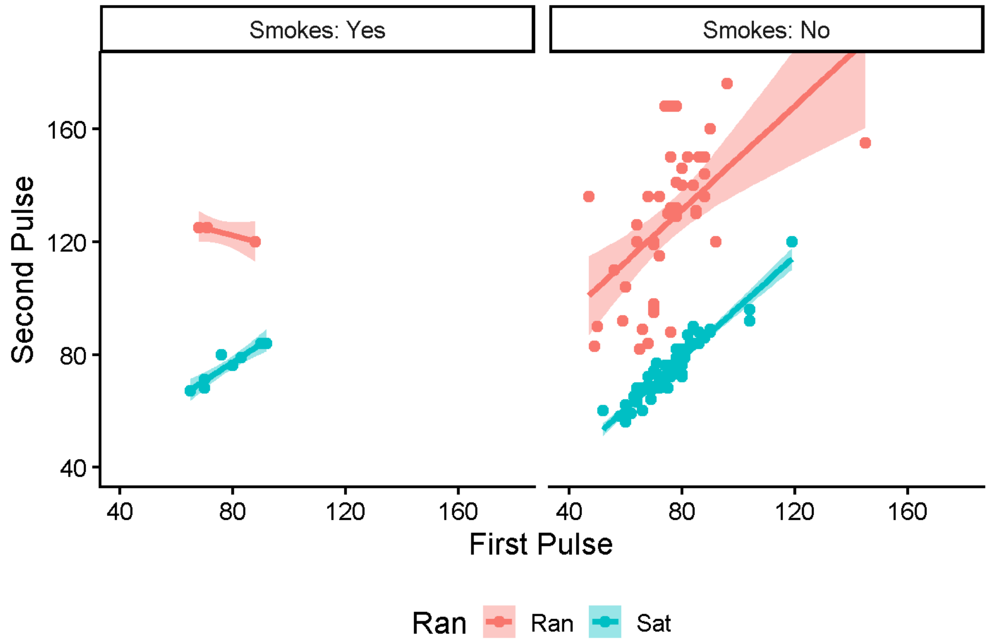
Communicate

We…have already learned ways to do this with RMarkdown!

Statistics and visualizations are great, but to tell a good story it is often useful to accompany these with text explaining the key features or importance.

And we’ve been practicing this all along by describing the plots!

Communicate: Pulse Measurements



In this visualization, we can see a strong positive relationship between participants' pulse measurements at the first time point and their pulse measurements at the second time point. We can see that the individuals who ran in between the measurements had similar values on the x-axis (the first measurement) but much higher values at the second measurement. While it seems like there may be some differences in pulse between regular smokers and non-smokers, there is much fewer

Assignment 5