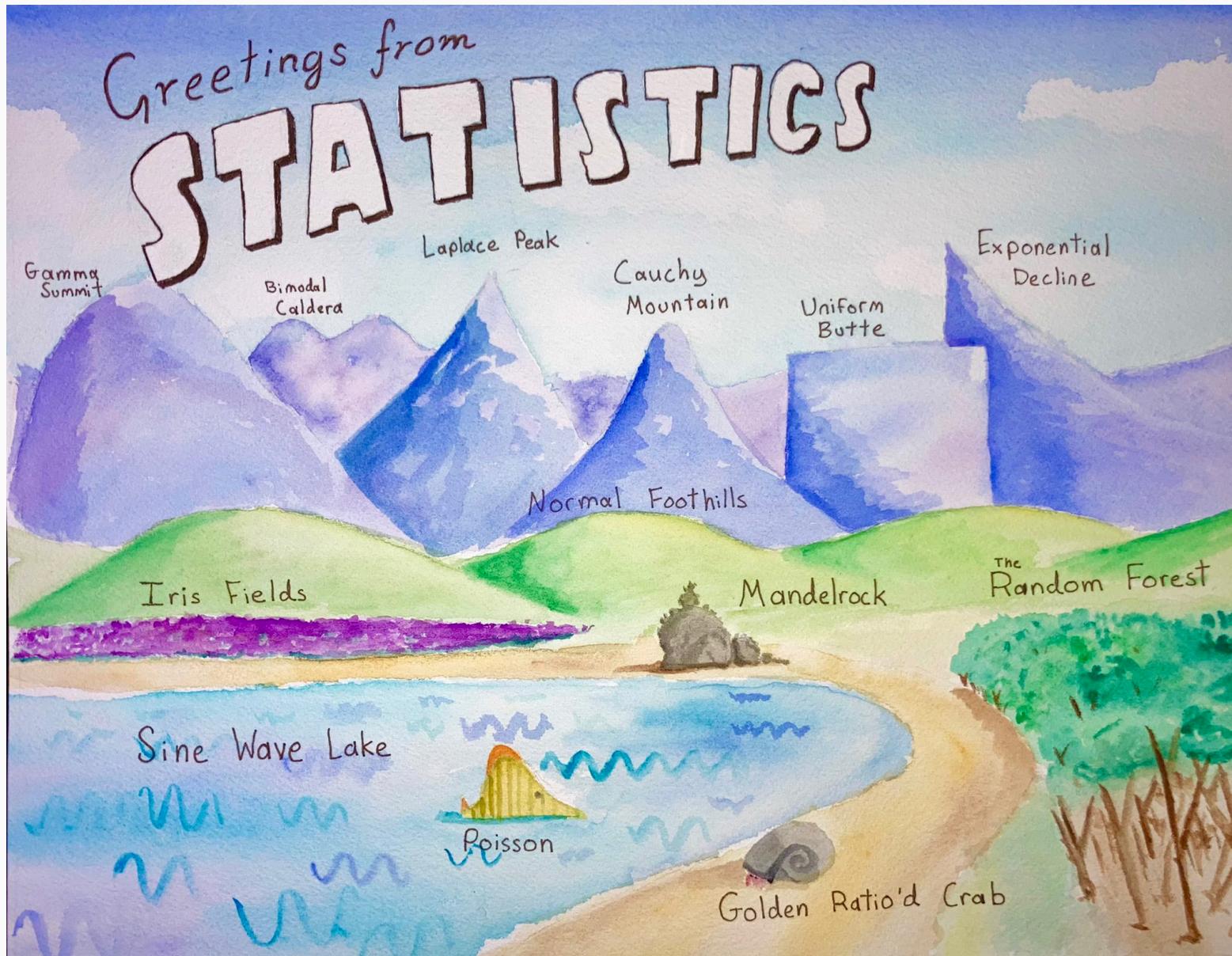


# Introduction to DAV 5300

## Computational Mathematics and Statistics

Jason Bryer, Ph.D.

August 27, 2024



@skyetetra

# Agenda

- Introductions
- Syllabus
- Class meetups
- Course Schedule
- Assignments (how you will be graded)
- Software setup
- Brief introduction to R

While waiting, please complete this formative assessment:



# A little about me...

- Earned my Ph.D. in Educational Psychology and Methodology from the University at Albany.  
Dissertation: [A National Study Comparing Charter and Traditional Public Schools Using Propensity Score Analysis](#)
- Assistant Professor at CUNY in Data Science and Information Systems
- Principal Investigator for a Department of Education Grant (part of their FIPSE First in the World program) to develop a Diagnostic Assessment and Achievement of College Skills ([www.DAACS.net](http://www.DAACS.net))
- Authored over a dozen R packages including:
  - [likert](#)
  - [sqlutils](#)
  - [timeline](#)
- Specialize in propensity score methods. Three new methods/R packages developed include:
  - [multilevelPSA](#)
  - [TriMatch](#)
  - [PSAboot](#)

# Also a Father...



# Runner...

---



And photographer.



# Syllabus

Syllabus and course materials are here:

<https://fall2024.dav5300.net>

We will use Canvas primary for submitting assignments only. Please submit PDFs.

PDFs are preferred for the homework as there is some LaTeX formatting in the R markdown files. The `tinytex` R package helps with install LaTeX, but you can also install LaTeX using [MiKTeX](#) (for Windows) and [BasicTeX](#) (for Mac).



# Class Meetings

Class will meet every Tuesday.

In order to get the most out of this class attendance is required.

**One Minute Papers** - Complete the one minute paper after each Meetup (whether you watch live or watch the recordings). It should take approximately one to two minutes to complete.

# Schedule

Start	Topic
Tuesday, August 27, 2024	Intro to the Course
Tuesday, September 03, 2024	Intro to Data
Tuesday, September 10, 2024	Summarizing Data
Tuesday, September 17, 2024	Probability
Tuesday, September 24, 2024	Distributions
Tuesday, October 01, 2024	Foundation for Inference
Tuesday, October 08, 2024	Inference for Categorical Data
Tuesday, October 15, 2024	Inference for Numerical Data
Tuesday, October 22, 2024	Linear Regression
Tuesday, October 29, 2024	Maximum Likelihood Estimation
Tuesday, November 05, 2024	Multiple Regression
Tuesday, November 12, 2024	Conferences (online)
Tuesday, November 19, 2024	Predictive Modeling
Tuesday, November 26, 2024	NO MEETUP - Thanksgiving
Tuesday, December 03, 2024	Bayesian Analysis
Tuesday, December 10, 2024	Presentations
Tuesday, December 17, 2024	Final Exam

Assignments are due on Monday before the next class.

# Textbooks

*OpenIntro Statistics* by David Diaz, Mine Çetinkaya-Rundel, and Christopher D Barr.

*Learning Statistics with R* by Danielle Navaro - We will only use the Bayesian chapter from this book.

## Optional

*R for Data Science* by Hadley Wickham and Garrett Grolemund - Recommended reference for those new to R.

# Assignments

**Labs** (30%) - Labs are designed to provide you an opportunity to apply statistical concepts using statistical software.

**Textbook questions** (15%) - The assigned questions from the textbook provide an opportunity to assess conceptional understandings.

**Participation** (10%) - You are expected to attend every class and to complete a **one minute paper** at the conclusion of class.

**Data Project** (25%) - In a group of 2 to 3 students will present the results of analysis using a data set of your choice. More details will be provided a few weeks into the class.

**Final exam** (20%) - A multiple choice exam will be given on the last day of class.

**All assignments are due on Monday.** Assignments submitted late will be penalized. Assignments will not be accepted more than one week after their due date.

# Academic Integrity

With the exception of the data project, I expect you to complete all assignments (e.g. homework, labs) on your own. It is fine to ask questions of your peers and professor, but working together and/or sharing answers is not allowed.

## Yeshiva's Policy

The submission by a student of any examination, course assignment, or degree requirement is assumed to guarantee that the thoughts and expressions therein not expressly credited to another are literally the student's own. Evidence to the contrary will result in appropriate penalties. For more information, visit <https://www.yu.edu/academic-integrity>.

# Communication

- Email: [jason.bryer@yu.edu](mailto:jason.bryer@yu.edu).
- Canvas
- Office hours before and after class and by appointment.

# Software Setup

# Why R?

There are many languages data scientists use. R is specifically designed for statistics. We will leverage many R packages that are specifically designed to conduct, teach, and communicate statistical analysis.

To be a well rounded data scientists, I believe you need to have experience in both R and Python. For this course:

- Use R for the labs (they are designed to help you learn the core commands).
- You may use Python or R for the homework and data project.



# Software



This is an applied statistics course so we will make extensive use of the **R statistical programming language**.

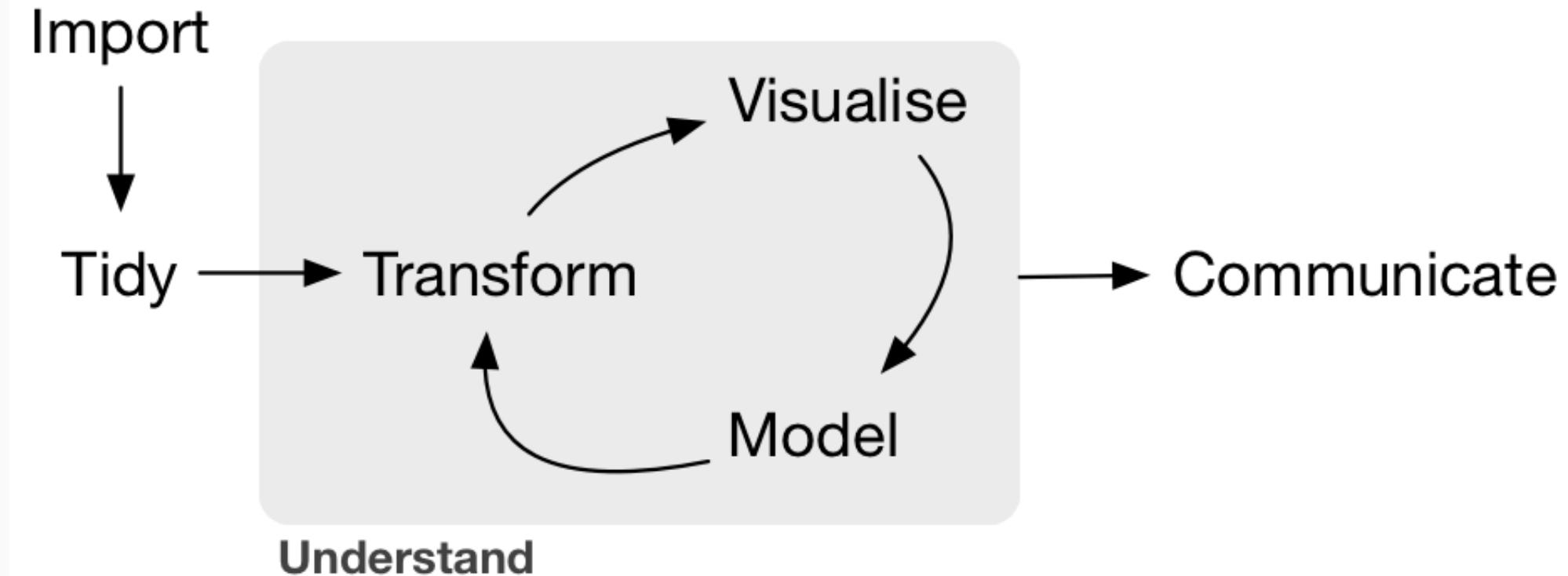
- Install **R** and **RStudio** on your own computer. I encourage everyone to do this at some point by the end of the semester.

You will also need to have **LaTeX** installed as well in order to create PDFs. The **tinytex** R package helps with this process:

```
install.packages('tinytex')
tinytex::install_tinytex()
```

# Introduction to R

# Workflow



Source: Wickham & Golemud, 2017

# Tidy Data

“**TIDY DATA** is a standard way of mapping the meaning of a dataset to its structure.”

—HADLEY WICKHAM

## In tidy data:

- each variable forms a column
- each observation forms a row
- each cell is a single measurement

each column a variable

each row an observation

id	name	color
1	floof	gray
2	max	black
3	cat	orange
4	donut	gray
5	merlin	black
6	panda	calico

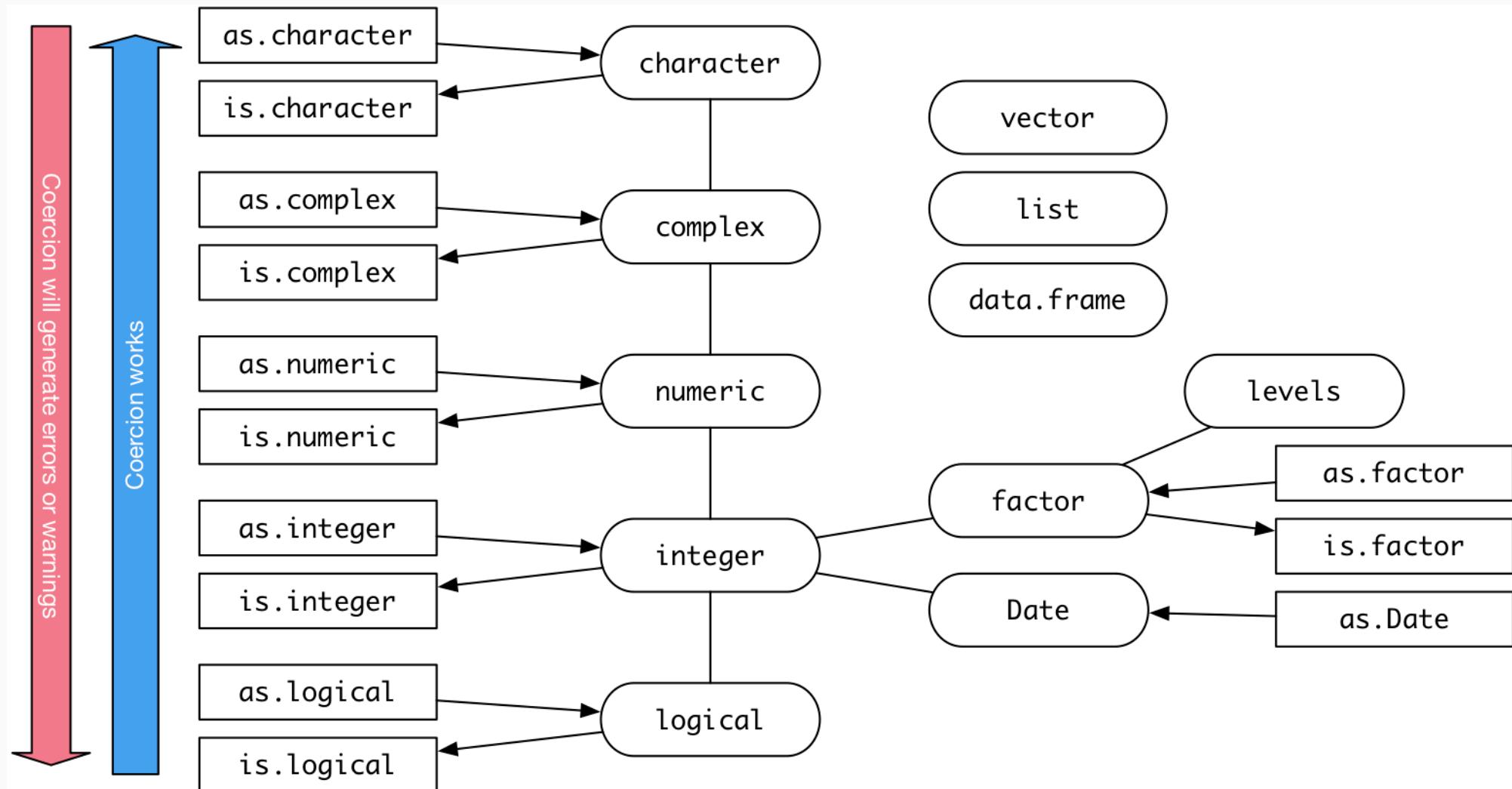
Wickham, H. (2014). Tidy Data. Journal of Statistical Software 59 (10). DOI: 10.18637/jss.v059.i10

# Types of Data

- Numerical (quantitative)
  - Continuous
  - Discrete
- Categorical (qualitative)
  - Regular categorical
  - Ordinal



# Data Types in R



# Data Types / Descriptives / Visualizations

Data Type	Descriptive Stats	Visualization
Continuous	mean, median, mode, standard deviation, IQR	histogram, density, box plot
Discrete	contingency table, proportional table, median	bar plot
Categorical	contingency table, proportional table	bar plot
Ordinal	contingency table, proportional table, median	bar plot
Two quantitative	correlation	scatter plot
Two qualitative	contingency table, chi-squared	mosaic plot, bar plot
Quantitative & Qualitative	grouped summaries, ANOVA, t-test	box plot

# Variance

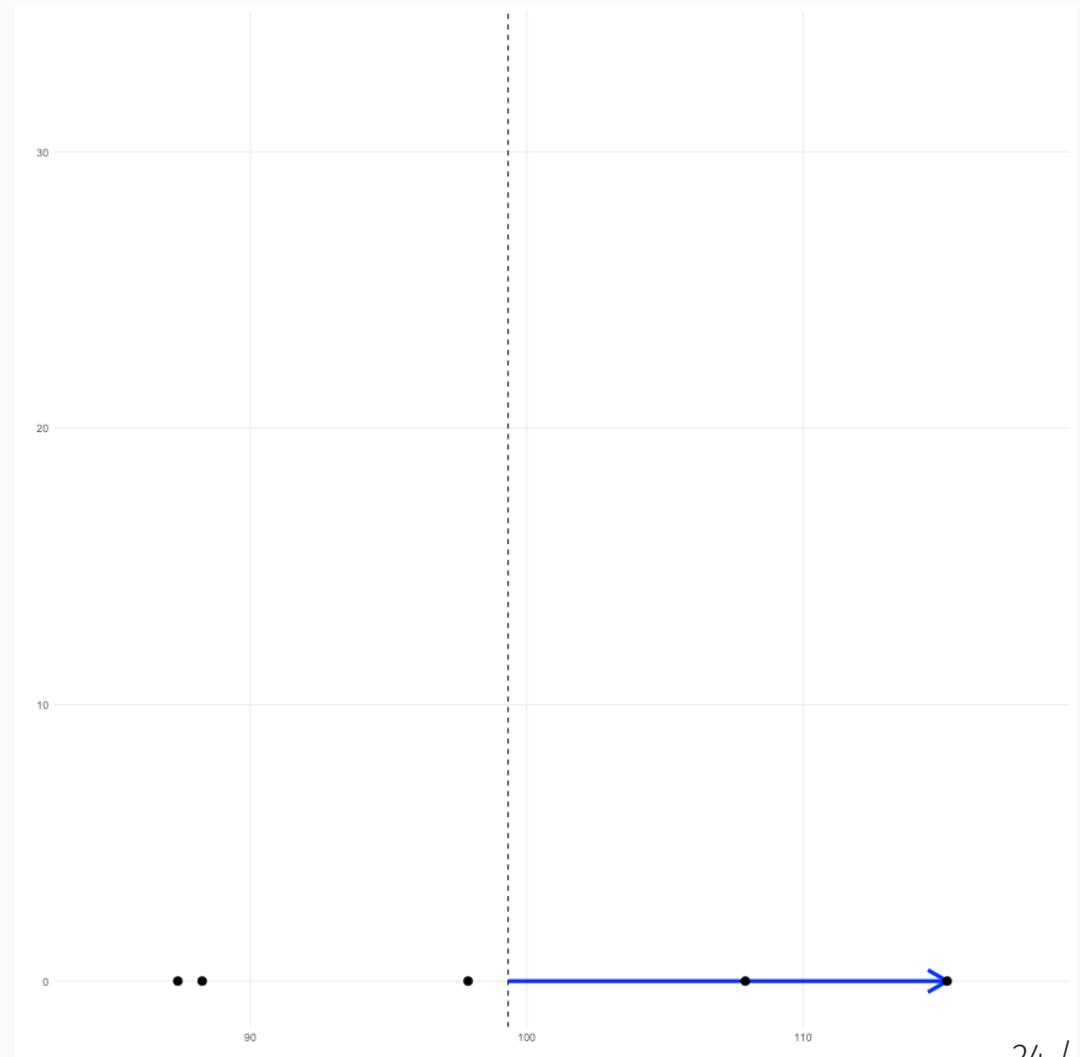
Population Variance:

$$S^2 = \frac{\Sigma(x_i - \bar{x})^2}{N}$$

Consider a dataset with five values (black points in the figure). For the largest value, the deviance is represented by the blue line ( $x_i - \bar{x}$ ).

See also:

<https://shiny.rit.albany.edu/stat/visualizess/>  
<https://github.com/jbryer/VisualStats/>

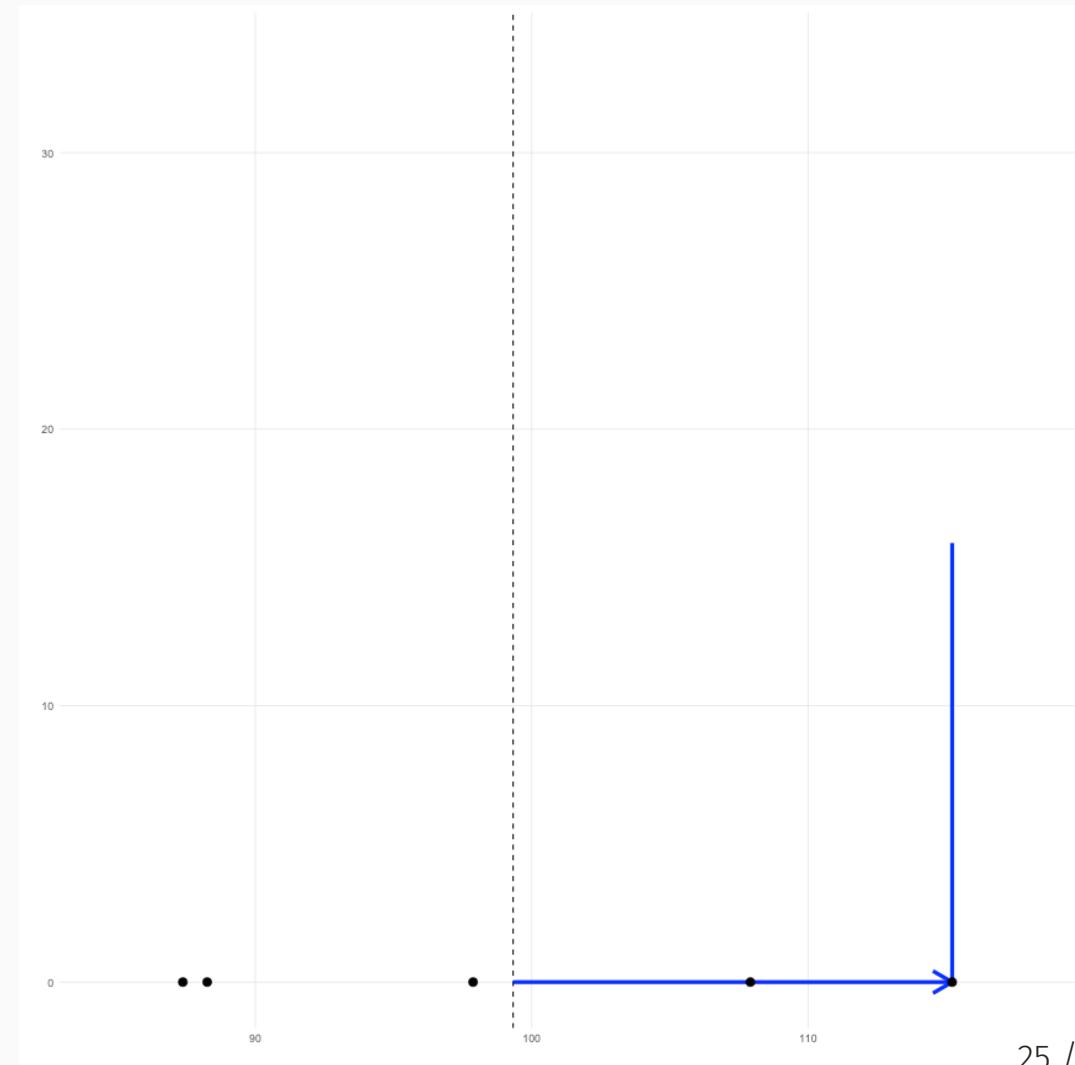


# Variance (cont.)

Population Variance:

$$S^2 = \frac{\Sigma(x_i - \bar{x})^2}{N}$$

In the numerator, we square each of these deviances. We can conceptualize this as a square. Here, we add the deviance in the  $y$  direction.

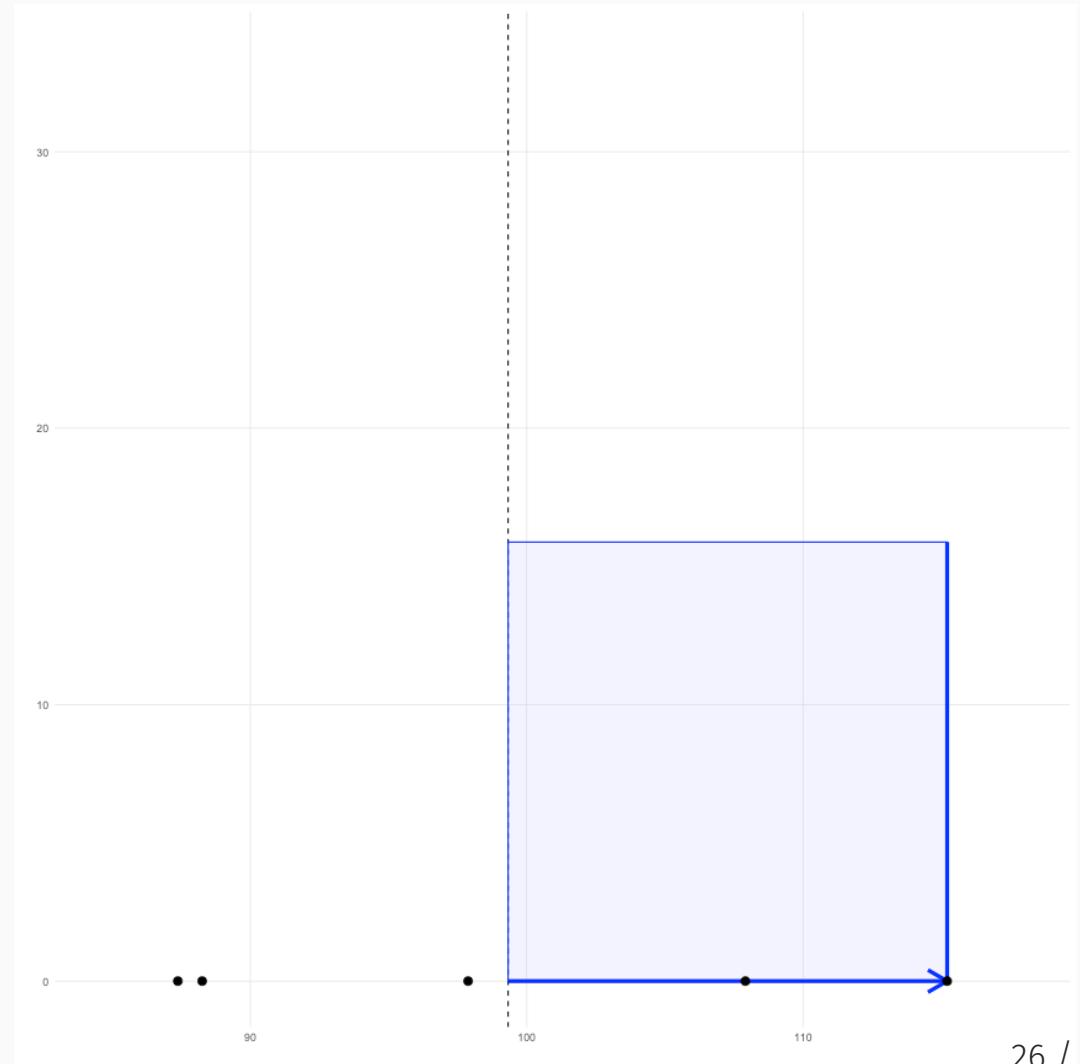


# Variance (cont.)

Population Variance:

$$S^2 = \frac{\Sigma(x_i - \bar{x})^2}{N}$$

We end up with a square.

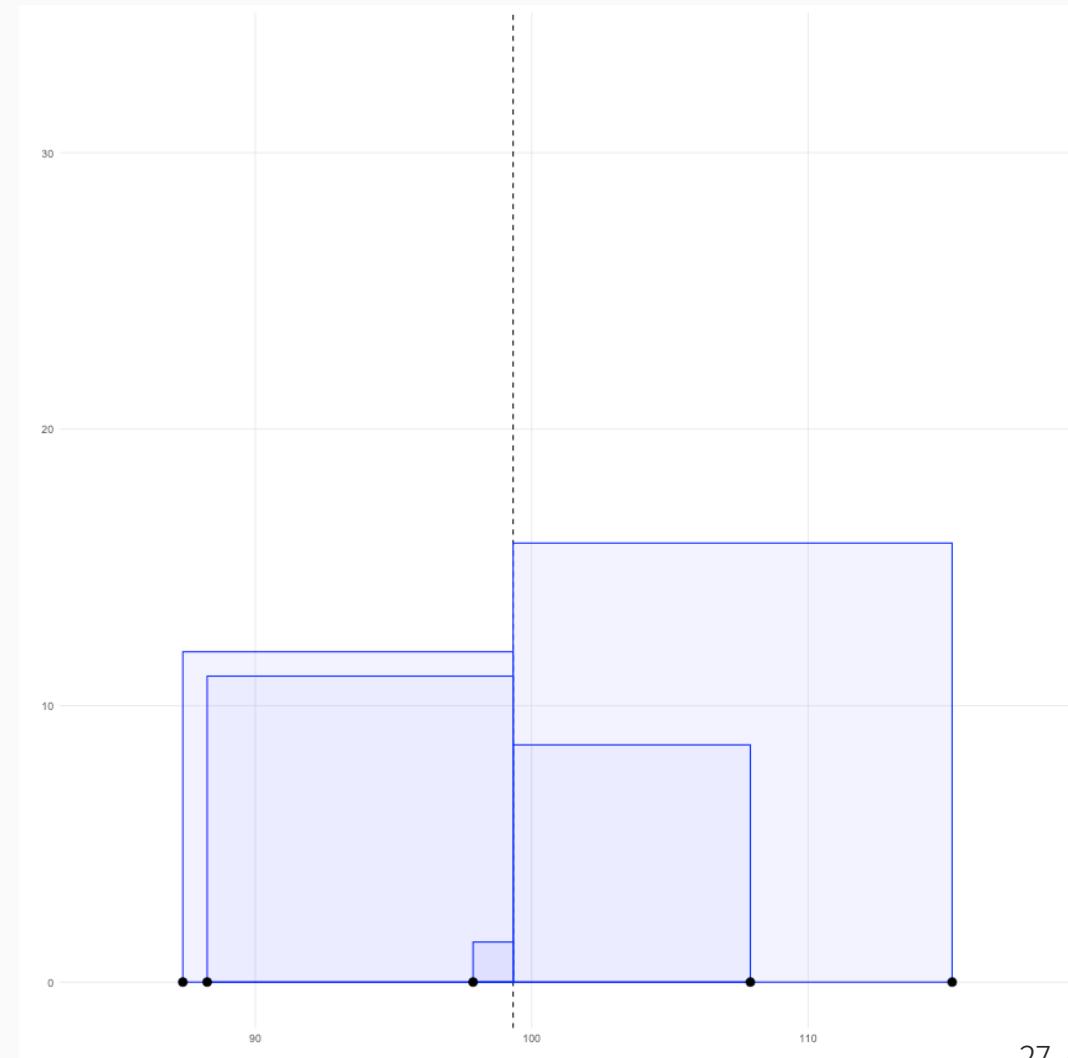


# Variance (cont.)

Population Variance:

$$S^2 = \frac{\Sigma(x_i - \bar{x})^2}{N}$$

We can plot the squared deviance for all the data points. That is, each component in the numerator is the area of each of these squares.

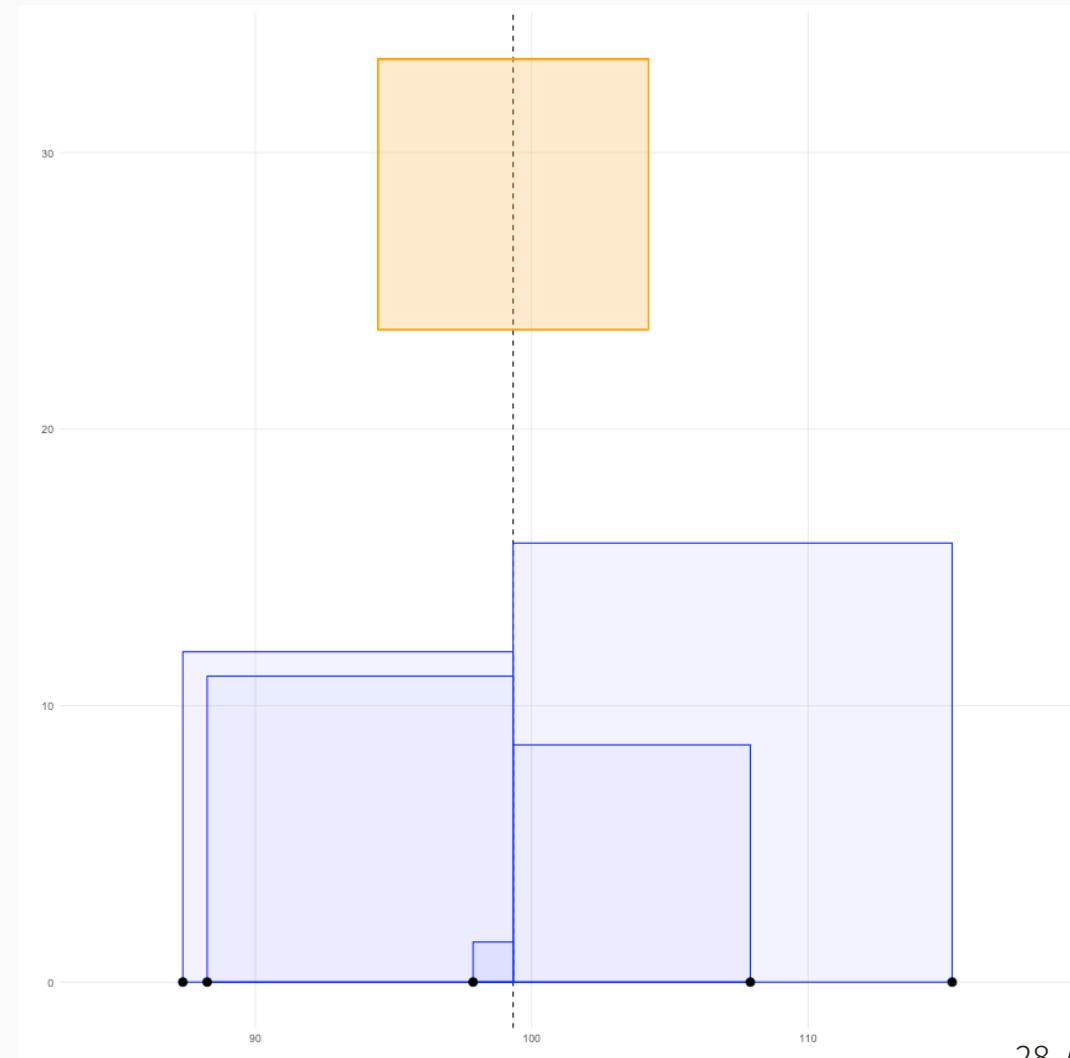


# Variance (cont.)

Population Variance:

$$S^2 = \frac{\Sigma(x_i - \bar{x})^2}{N}$$

The variance is therefore the average of the area of all these squares, here represented by the orange square.



# Population versus Sample Variance

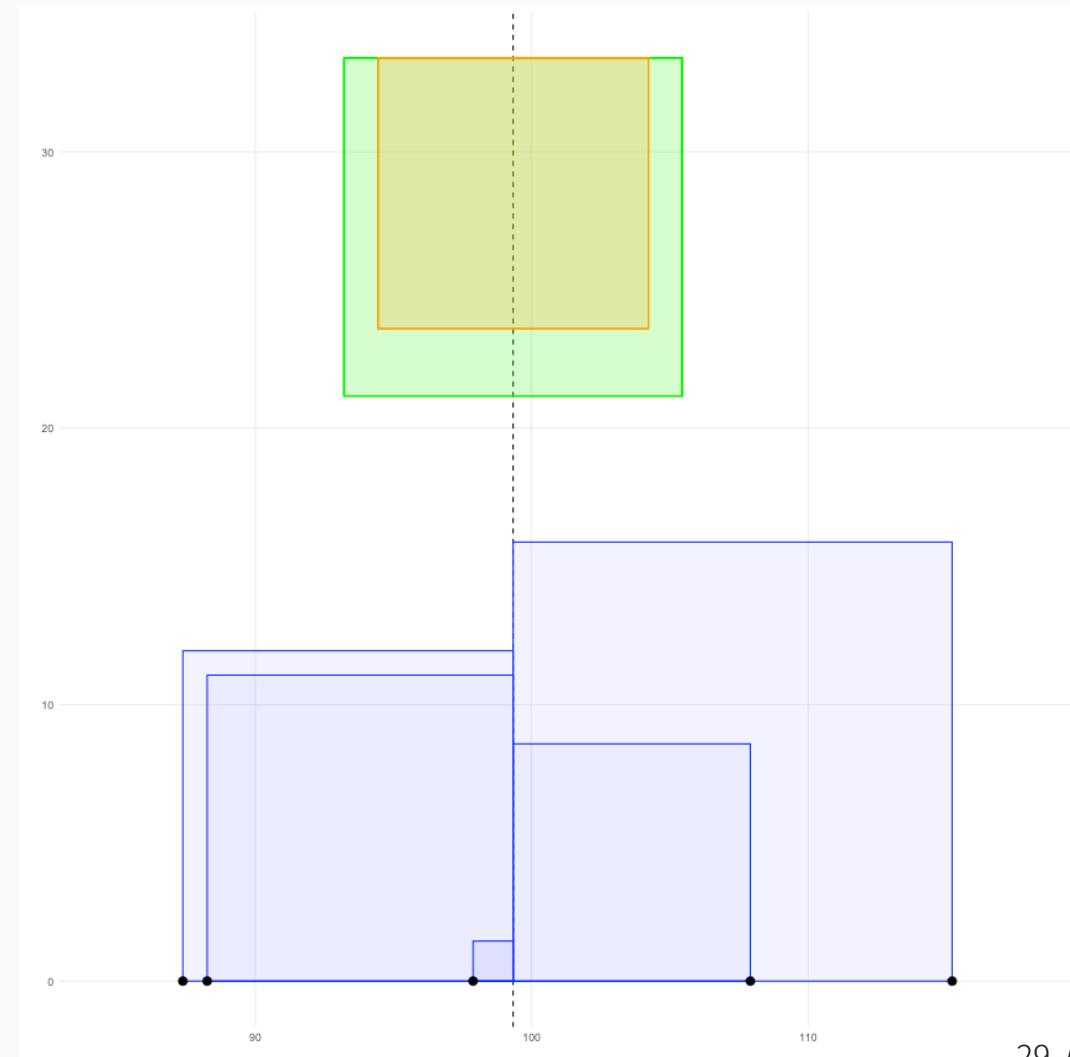
Typically we want the sample variance. The difference is we divide by  $n - 1$  to calculate the sample variance. This results in a slightly larger area (variance) then if we divide by  $n$ .

Population Variance (yellow):

$$S^2 = \frac{\sum(x_i - \bar{x})^2}{N}$$

Sample Variance (green):

$$s^2 = \frac{\sum(x_i - \bar{x})^2}{n - 1}$$



# Robust Statistics

Consider the following data randomly selected from the normal distribution:

```
set.seed(41)
x <- rnorm(30, mean = 100, sd = 15)
mean(x); sd(x)
```

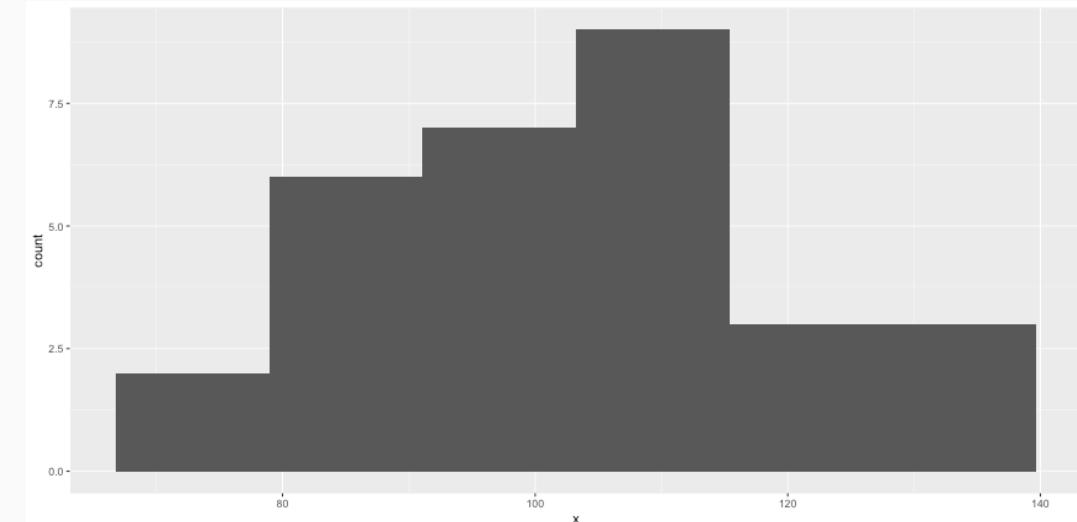
```
## [1] 103.1934
```

```
## [1] 16.8945
```

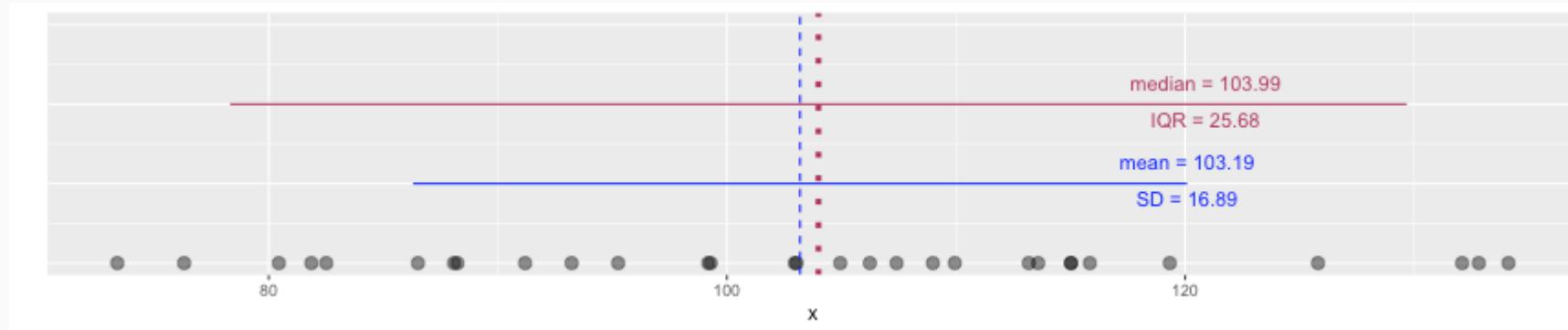
```
median(x); IQR(x)
```

```
## [1] 103.9947
```

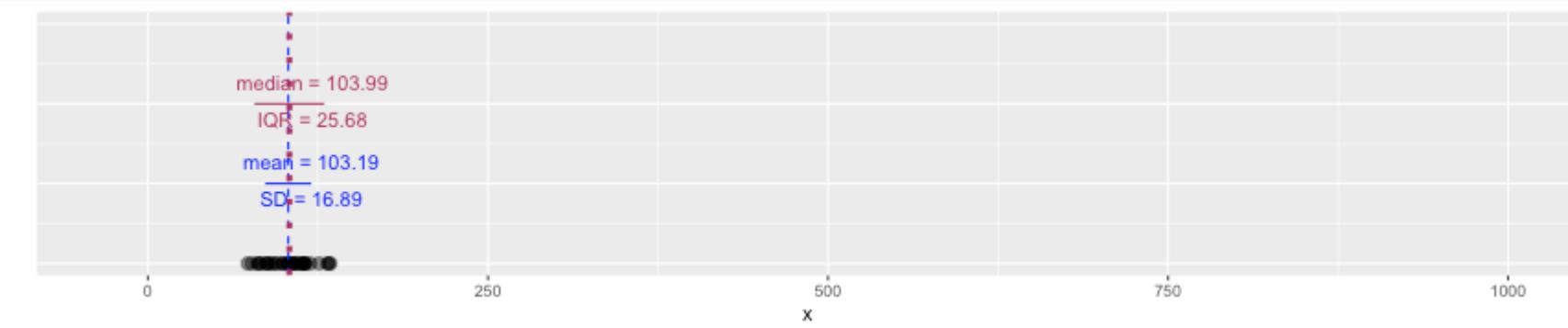
```
## [1] 25.68004
```



# Robust Statistics

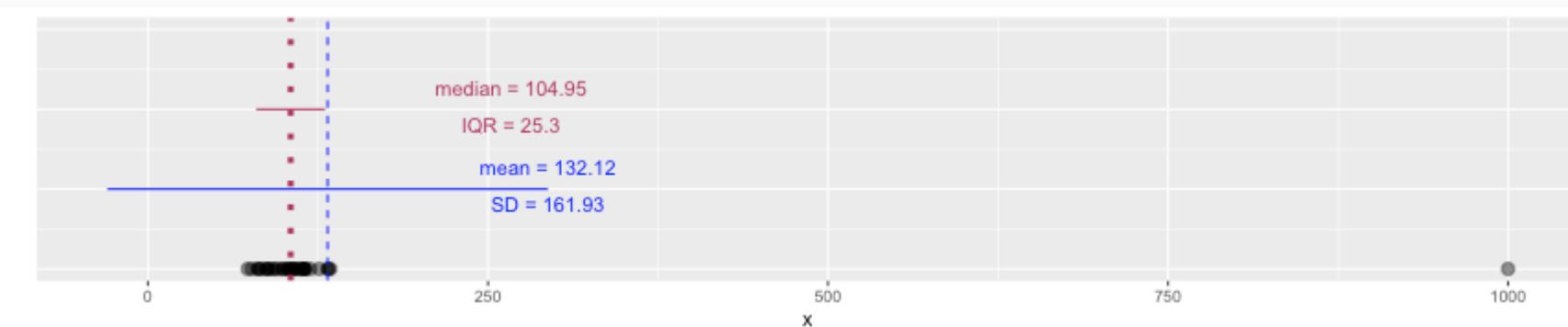


# Robust Statistics



Let's add an extreme value:

```
x <- c(x, 1000)
```



# Robust Statistics

Median and IQR are more robust to skewness and outliers than mean and SD. Therefore,

- for skewed distributions it is often more helpful to use median and IQR to describe the center and spread
- for symmetric distributions it is often more helpful to use the mean and SD to describe the center and spread



# About legosets

To install the `brickset` package:

```
remotes::install_github('jbryer/brickset')
```

To load the `legosets` dataset.

```
data('legosets', package = 'brickset')
```

The `legosets` data has 19409 observations of 36 variables.

```
names(legosets)
```

```
## [1] "setID"                 "number"                "numberVariant"  
## [4] "name"                  "year"                  "theme"  
## [7] "themeGroup"             "subtheme"              "category"  
## [10] "released"               "pieces"                "minifigs"  
## [13] "bricksetURL"            "rating"                "reviewCount"  
## [16] "packagingType"          "availability"          "agerange_min"  
## [19] "thumbnailURL"           "imageURL"              "US_retailPrice"  
## [22] "US_dateFirstAvailable" "US_dateLastAvailable" "UK_retailPrice"  
## [25] "UK_dateFirstAvailable" "UK_dateLastAvailable" "CA_retailPrice"  
## [28] "CA_dateFirstAvailable" "CA_dateLastAvailable" "DE_retailPrice"  
## [31] "DE_dateFirstAvailable" "DE_dateLastAvailable" "height"  
## [34] "width"                  "depth"                 "weight"
```

# Structure (str)

```
str(legosets)
```

```
## 'data.frame': 19409 obs. of 36 variables:
## $ setID      : int 7693 7695 7697 7698 25534 ...
## $ number     : chr "1" "2" "3" "4" ...
## $ numberVariant : int 8 8 6 4 6 1 1 1 3 4 ...
## $ name       : chr "Small house set" "Medium house set" "Medium house set" "Large house set" ...
## $ year        : int 1970 1970 1970 1970 1970 1970 1970 1970 1970 ...
## $ theme      : chr "Minitalia" "Minitalia" "Minitalia" "Minitalia" ...
## $ themeGroup : chr "Vintage" "Vintage" "Vintage" "Vintage" ...
## $ subtheme   : chr NA NA NA NA ...
## $ category   : chr "Normal" "Normal" "Normal" "Normal" ...
## $ released    : logi TRUE TRUE TRUE TRUE TRUE ...
## $ pieces      : int 67 109 158 233 NA 1 1 60 65 NA ...
## $ minifigs   : int NA NA NA NA NA NA NA NA NA ...
## $ bricksetURL: chr "https://brickset.com/sets/1-8" "https://brickset.com/sets/2-8" "https://brickset.com/sets/3-6" "https://brickset.com/sets/4-4" ...
## $ rating      : num 0 0 0 0 0 0 0 0 0 ...
## $ reviewCount: int 0 0 1 0 0 0 0 0 0 ...
## $ packagingType: chr "{Not specified}" "{Not specified}" "{Not specified}" "{Not specified}" ...
## $ availability: chr "{Not specified}" "{Not specified}" "{Not specified}" "{Not specified}" ...
## $ agerange_min: int NA NA NA NA NA NA NA NA NA ...
## $ thumbnailURL: chr "https://images.brickset.com/sets/small/1-8.jpg" "https://images.brickset.com/sets/small/2-8.jpg" "https://images.brickset.com/sets/small/3-6.jpg" "https://images.brickset.com/sets/small/4-4.jpg" ...
## $ imageURL    : chr "https://images.brickset.com/sets/images/1-8.jpg" "https://images.brickset.com/sets/images/2-8.jpg" "https://images.brickset.com/sets/images/3-6.jpg" "https://images.brickset.com/sets/images/4-4.jpg" ...
## $ US_retailPrice: num NA NA NA NA NA NA NA NA NA ...
## $ US_dateFirstAvailable: Date, format: NA NA ...
## $ US_dateLastAvailable: Date, format: NA NA ...
## $ UK_retailPrice   : num NA NA NA NA NA NA NA NA NA ...
## $ UK_dateFirstAvailable: Date, format: NA NA ...
## $ UK_dateLastAvailable: Date, format: NA NA ...
## $ CA_retailPrice   : num NA NA NA NA NA NA NA NA NA ...
## $ CA_dateFirstAvailable: Date, format: NA NA ...
## $ CA_dateLastAvailable: Date, format: NA NA ...
## $ DE_retailPrice   : num NA NA NA NA NA NA NA NA NA ...
## $ DE_dateFirstAvailable: Date, format: NA NA ...
## $ DE_dateLastAvailable: Date, format: NA NA ...
## $ height         : num NA NA NA NA NA ...
## $ width          : num NA NA NA NA NA ...
```

# RStudio Environment tab can help

Environment History Connections Git Tutorial

Import Dataset | 

R Global Environment 

Data

legosets	16355 obs. of 34 variables
\$ setID	: int 7693 7695 7697 7698 25534 ...
\$ name	: chr "Small house set" "Medium house set" "Medium house set" "L...
\$ year	: int 1970 1970 1970 1970 1970 1970 1970 1970 1970 1970 1970 1970 ...
\$ theme	: chr "Minitalia" "Minitalia" "Minitalia" "Minitalia" ...
\$ themeGroup	: chr "Vintage" "Vintage" "Vintage" "Vintage" ...
\$ subtheme	: chr NA NA NA NA ...
\$ category	: chr "Normal" "Normal" "Normal" "Normal" ...
\$ released	: logi TRUE TRUE TRUE TRUE TRUE ...
\$ pieces	: int 67 109 158 233 NA 1 1 60 65 NA ...
\$ minifigs	: int NA NA NA NA NA NA NA NA NA ...
\$ bricksetURL	: chr "https://brickset.com/sets/1-8" "https://brickset.com/sets..."
\$ rating	: num 0 0 0 0 0 0 0 0 0 ...
\$ reviewCount	: int 0 0 1 0 0 0 0 1 0 0 ...
\$ packagingType	: chr "{Not specified}" "{Not specified}" "{No..."
\$ availability	: chr "{Not specified}" "{Not specified}" "{No..."
\$ agerange_min	: int NA NA NA NA NA NA NA NA NA ...
\$ US_retailPrice	: num NA NA NA NA NA 1.99 NA NA 4.99 NA ...
\$ US_dateFirstAvailable	: Date, format: NA NA NA NA ...
\$ US_dateLastAvailable	: Date, format: NA NA NA NA ...
\$ UK_retailPrice	: num NA NA NA NA NA NA NA NA NA ...
\$ UK_dateFirstAvailable	: Date, format: NA NA NA NA ...
\$ UK_dateLastAvailable	: Date, format: NA NA NA NA ...
\$ CA_retailPrice	: num NA NA NA NA NA NA NA NA NA ...
\$ CA_dateFirstAvailable	: Date, format: NA NA NA NA ...
\$ CA_dateLastAvailable	: Date, format: NA NA NA NA ...
\$ DE_retailPrice	: num NA NA NA NA NA NA NA NA NA ...
\$ DE_dateFirstAvailable	: Date, format: NA NA NA NA ...
\$ DE_dateLastAvailable	: Date, format: NA NA NA NA ...
\$ height	: num NA NA NA NA NA ...
\$ width	: num NA NA NA NA NA ...
\$ depth	: num NA NA NA NA NA NA NA 5.08 NA ...
\$ weight	: num NA NA NA NA NA NA NA NA NA ...
\$ thumbnailURL	: chr "https://images.brickset.com/sets/small/1-8.jpg" "https://..."
\$ imageURL	: chr "https://images.brickset.com/sets/images/1-8.jpg" "https:/..."

# Table View

Show  entries Search:

setID	name		year	theme	themeGroup	category	US_retailPrice	pieces	minifigs	rating
1	9712	Darth Maul	2012	Gear	Miscellaneous	Gear	4.99			3.7
2	28411	TIE Fighter Attack	2019	Star Wars	Licensed	Normal	19.99	77	2	3.5
3	24252	LEGO Minifigures - Series 14 - Monsters {Random bag}	2015	Collectable Minifigures	Miscellaneous	Random				0
4	1448	Head Stand	1998	Town	Modern day	Normal		96	3	0
5	32020	Mania Magazine March - April 1995	1995	Books	Miscellaneous	Book				0
6	28352	Genius LEGO Inventions with Bricks You Already Have	2018	Books	Miscellaneous	Book				0
7	34334	Friendship Flowers	2023	Friends	Modern day	Normal		84		3.8
8	31865	BRICK KICKS Summer 1991	1991	Books	Miscellaneous	Book				0
9	31061	LEGO Space Projects: 52 Galactic Models	2021	Books	Miscellaneous	Book				0
10	30688	James Potter	2020	Collectable Minifigures	Miscellaneous	Normal		7	1	3.8

Showing 1 to 10 of 100 entries

Previous [1](#) [2](#) [3](#) [4](#) [5](#) ... [10](#) Next



# Data Wrangling Cheat Sheet

## Data Transformation with dplyr :: CHEAT SHEET

dplyr functions work with pipes and expect **tidy data**. In tidy data:



Each **variable** is in its own **column**



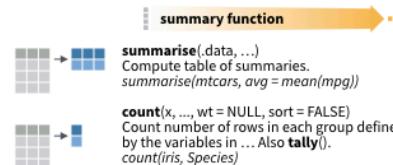
Each **observation**, or **case**, is in its own **row**



`x %>% f(y)` becomes `f(x, y)`

### Summarise Cases

These apply **summary functions** to columns to create a new table of summary statistics. Summary functions take vectors as input and return one value (see back).

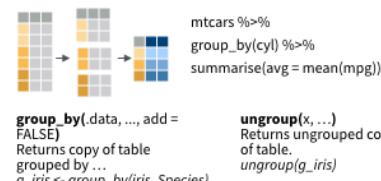


#### VARIATIONS

`summarise_all()` - Apply funs to every column.  
`summarise_at()` - Apply funs to specific columns.  
`summarise_if()` - Apply funs to all cols of one type.

### Group Cases

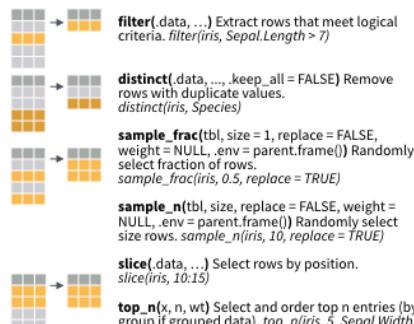
Use `group_by()` to create a "grouped" copy of a table. dplyr functions will manipulate each "group" separately and then combine the results.



### Manipulate Cases

#### EXTRACT CASES

`Row` functions return a subset of rows as a new table.

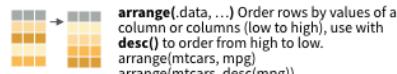


#### Logical and boolean operators to use with filter()

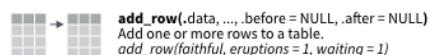
<	=	is.na()	%in%		xor()
>	==	!is.na()	!	&	

See `?base::logic` and `?Comparison` for help.

#### ARRANGE CASES



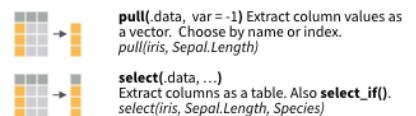
#### ADD CASES



### Manipulate Variables

#### EXTRACT VARIABLES

`Column` functions return a set of columns as a new vector or table.

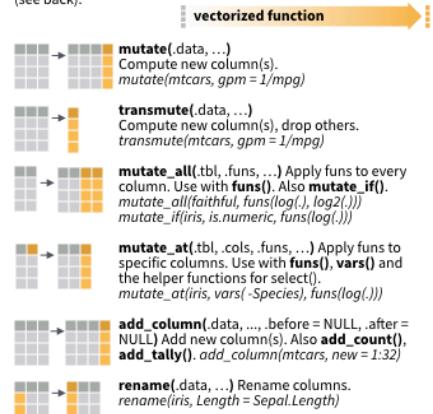


Use these helpers with `select()`, e.g. `select(iris, starts_with("Sepal"))`

<code>contains(match)</code>	<code>num_range(prefix, range)</code>	:	e.g. <code>mpg:cyl</code>
<code>ends_with(match)</code>	<code>one_of(...)</code>	:	e.g. <code>Species</code>
<code>matches(match)</code>	<code>starts_with(match)</code>		

#### MAKE NEW VARIABLES

These apply **vectorized functions** to columns. Vectorized funs take vectors as input and return vectors of the same length as output (see back).



RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more with `browseVignettes(package = c("dplyr", "tibble"))` • dplyr 0.7.0 • tibble 1.2.0 • Updated: 2017-03

# Tidyverse vs Base R



## R Syntax Comparison :: CHEAT SHEET

### Dollar sign syntax

```
goal(data$x, data$y)
```

#### SUMMARY STATISTICS:

one continuous variable:  
`mean(mtcars$mpg)`

one categorical variable:  
`table(mtcars$cyl)`

two categorical variables:  
`table(mtcars$cyl, mtcars$am)`

one continuous, one categorical:  
`mean(mtcars$mpg[mtcars$cyl==4])`  
`mean(mtcars$mpg[mtcars$cyl==6])`  
`mean(mtcars$mpg[mtcars$cyl==8])`

#### PLOTTING:

one continuous variable:  
`hist(mtcars$disp)`

`boxplot(mtcars$disp)`

one categorical variable:  
`barplot(table(mtcars$cyl))`

two continuous variables:  
`plot(mtcars$disp, mtcars$mpg)`

two categorical variables:  
`mosaicplot(table(mtcars$am, mtcars$cyl))`

one continuous, one categorical:  
`histogram(mtcars$disp[mtcars$cyl==4])`  
`histogram(mtcars$disp[mtcars$cyl==6])`  
`histogram(mtcars$disp[mtcars$cyl==8])`

`boxplot(mtcars$disp[mtcars$cyl==4])`  
`boxplot(mtcars$disp[mtcars$cyl==6])`  
`boxplot(mtcars$disp[mtcars$cyl==8])`

#### WRANGLING:

subsetting:  
`mtcars[mtcars$mpg>30, ]`

making a new variable:  
`mtcars$efficient[mtcars$mpg>30] <- TRUE`  
`mtcars$efficient[mtcars$mpg<30] <- FALSE`

### Formula syntax

```
goal(y~x|z, data=data, group=w)
```

#### SUMMARY STATISTICS:

one continuous variable:  
`mosaic::mean(~mpg, data=mtcars)`

one categorical variable:  
`mosaic::tally(~cyl, data=mtcars)`

two categorical variables:  
`mosaic::tally(cyl~am, data=mtcars)`

one continuous, one categorical:  
`mosaic::mean(mpg~cyl, data=mtcars)`

tilde

#### PLOTTING:

one continuous variable:  
`lattice::histogram(~disp, data=mtcars)`

`lattice::bwplot(~disp, data=mtcars)`

one categorical variable:  
`mosaic::bargraph(~cyl, data=mtcars)`

two continuous variables:  
`lattice::xyplot(mpg~disp, data=mtcars)`

two categorical variables:  
`mosaic::bargraph(~am, data=mtcars, group=cyl)`

one continuous, one categorical:  
`lattice::histogram(~disp|cyl, data=mtcars)`

`lattice::bwplot(cyl~disp, data=mtcars)`

The variety of R syntaxes give  
you many ways to “say” the  
same thing

read across the cheatsheet to see how different  
syntaxes approach the same problem

### Tidyverse syntax

```
data %>% goal(x)
```

#### SUMMARY STATISTICS:

one continuous variable:  
`mtcars %>% dplyr::summarize(mean(mpg))`

the pipe

one categorical variable:  
`mtcars %>% dplyr::group_by(cyl) %>%  
dplyr::summarize(n())`

two categorical variables:  
`mtcars %>% dplyr::group_by(cyl, am) %>%  
dplyr::summarize(n())`

one continuous, one categorical:  
`mtcars %>% dplyr::group_by(cyl) %>%  
dplyr::summarize(mean(mpg))`

PLOTTING:  
one continuous variable:  
`ggplot2::qplot(x=mpg, data=mtcars, geom = "histogram")`

`ggplot2::qplot(y=disp, x=1, data=mtcars, geom="boxplot")`

one categorical variable:  
`ggplot2::qplot(x=cyl, data=mtcars, geom="bar")`

two continuous variables:  
`ggplot2::qplot(x=disp, y=mpg, data=mtcars, geom="point")`

two categorical variables:  
`ggplot2::qplot(x=factor(cyl), data=mtcars, geom="bar") +  
facet_grid(.~am)`

one continuous, one categorical:  
`ggplot2::qplot(x=disp, data=mtcars, geom = "histogram") +  
facet_grid(.~cyl)`

`ggplot2::qplot(y=disp, x=factor(cyl), data=mtcars,  
geom="boxplot")`

WRANGLING:  
subsetting:  
`mtcars %>% dplyr::filter(mpg>30)`

making a new variable:  
`mtcars <- mtcars %>%  
dplyr::mutate(efficient = if_else(mpg>30, TRUE, FALSE))`

# Pipes %>% and |>



The pipe operator (`%>%`) introduced with the `magrittr` R package allows for the chaining of R operations. Base R has now added their own pipe operator (`|>`). They take the output from the left-hand side and passes it as the first parameter to the function on the right-hand side.



You can do this in two steps:

```
tab_out <- table(legosets$category)
prop.table(tab_out)
```

Or as nested function calls.

```
prop.table(table(legosets$category))
```

Using the pipe (`|>`) operator we can chain these calls in a what is arguably a more readable format:

```
table(legosets$category) |> prop.table()
```

```
##          Book Collection Extended      Gear      Normal       Other
## 0.034468546 0.031377196 0.028749549 0.154515946 0.684682364 0.062599825
```

## dplyr::filter()

KEEP ROWS THAT  
satisfy  
*your CONDITIONS*

keep rows from... this data... ONLY IF... type MATCHES "otter" AND site MATCHES "bay"  
filter(df, type == "otter" & site == "bay")



A cartoon illustration featuring three characters: an orange circle with a smiling face, a purple circle with a question mark, and a green circle with a neutral face. They are positioned around a small map of a coastal area with a blue bay labeled "BAY". The purple character is pointing at the map, and the green character is pointing at the orange character.

	type	food	site
1	otter	urchin	bay
2	Shark	seal	channel
3	otter	abalone	bay
4	otter	crab	wharf

@allisonhorst

# Logical Operators

- `!a` - TRUE if a is FALSE
- `a == b` - TRUE if a and be are equal
- `a != b` - TRUE if a and b are not equal
- `a > b` - TRUE if a is larger than b, but not equal
- `a >= b` - TRUE if a is larger or equal to b
- `a < b` - TRUE if a is smaller than be, but not equal
- `a <= b` - TRUE if a is smaller or equal to b
- `a %in% b` - TRUE if a is in b where b is a vector

```
which( letters %in% c('a','e','i','o','u') )
```

```
## [1] 1 5 9 15 21
```

- `a | b` - TRUE if a or b are TRUE
- `a & b` - TRUE if a and b are TRUE
- `isTRUE(a)` - TRUE if a is TRUE

# Filter



## dplyr

```
mylego <- legosets %>% filter(themeGroup == 'Educational' & year > 2015)
```

## Base R

```
mylego <- legosets[legosets$themeGroups == 'Educaitonal' & legosets$year > 2015,]
```

---

```
nrow(mylego)
```

```
## [1] 99
```

# Select



## dplyr

```
mylego <- mylego %>% select(setID, pieces, theme, availability, US_retailPrice, minifigs)
```

## Base R

```
mylego <- mylego[,c('setID', 'pieces', 'theme', 'availability', 'US_retailPrice', 'minifigs')]
```

---

```
head(mylego, n = 4)
```

	setID	pieces	theme	availability	US_retailPrice	minifigs
## 1	26803	103	Education	{Not specified}	NA	6
## 2	26689	142	Education	{Not specified}	NA	4
## 3	26804	98	Education	{Not specified}	NA	6
## 4	26277	188	Education	Educational	94.95	NA

# Relocate



dplyr::**relocate()**  
move COLUMNS around!

Default: move to FRONT  
or move to  
.before or .after  
A SPECIFIED COLUMN!



# Relocate



## dplyr

```
mylego %>% relocate(where(is.numeric), .after = where(is.character)) %>% head(n = 3)
```

```
##      theme availability setID pieces US_retailPrice minifigs
## 1 Education {Not specified} 26803     103          NA       6
## 2 Education {Not specified} 26689     142          NA       4
## 3 Education {Not specified} 26804      98          NA       6
```

## Base R

```
mylego2 <- mylego[,c('theme', 'availability', 'setID', 'pieces', 'US_retailPrice', 'minifigs')]
head(mylego2, n = 3)
```

```
##      theme availability setID pieces US_retailPrice minifigs
## 1 Education {Not specified} 26803     103          NA       6
## 2 Education {Not specified} 26689     142          NA       4
## 3 Education {Not specified} 26804      98          NA       6
```

# Rename



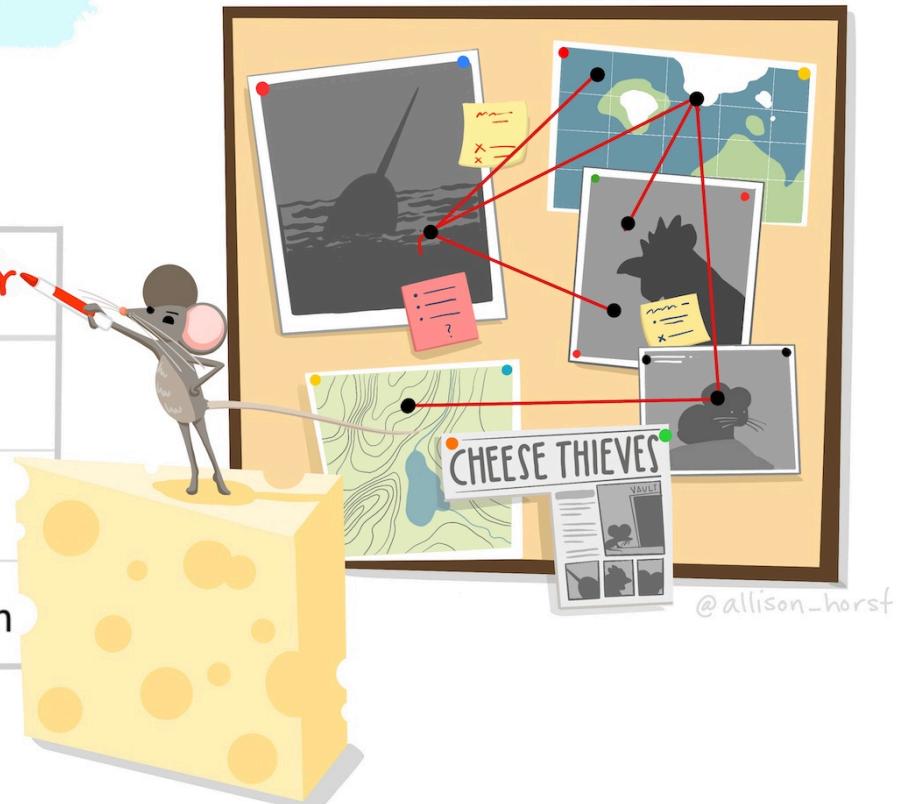
## dplyr::rename()

RENAME COLUMNS\*

df %>% rename(lair=site)

<del>species</del> nemesis	status	<del>site</del> lair
narwhal	unknown	ocean
chicken	active	coop
pika	active	mountain

\*See `rename_with()` to rename using a function.



# Rename



## dplyr

```
mylego %>% dplyr::rename(USD = US_retailPrice) %>% head(n = 3)
```

```
## #> #> #> #>
```

	setID	pieces	theme	availability	USD	minifigs
## 1	26803	103	Education	{Not specified}	NA	6
## 2	26689	142	Education	{Not specified}	NA	4
## 3	26804	98	Education	{Not specified}	NA	6

## Base R

```
names(mylego2)[5] <- 'USD'  
head(mylego2, n = 3)
```

```
## #> #> #> #>
```

	theme	availability	setID	pieces	USD	minifigs
## 1	Education	{Not specified}	26803	103	NA	6
## 2	Education	{Not specified}	26689	142	NA	4
## 3	Education	{Not specified}	26804	98	NA	6

# Mutate



# Mutate



## dplyr

```
mylego %>% filter(!is.na(pieces) & !is.na(US_retailPrice)) %>%  
  mutate(Price_per_piece = US_retailPrice / pieces) %>% head(n = 3)
```

```
## #> #> setID pieces theme availability US_retailPrice minifigs Price_per_piece  
## #> 1 26277 188 Education Educational 94.95 NA 0.5050532  
## #> 2 25949 280 Education Educational 224.95 NA 0.8033929  
## #> 3 25954 1 Education Educational 14.95 NA 14.9500000
```

## Base R

```
mylego2 <- mylego[!is.na(mylego$US_retailPrice) & !is.na(mylego$Price_per_piece),]  
mylego2$Price_per_piece <- mylego2$Price_per_piece / mylego2$US_retailPrice  
head(mylego2, n = 3)
```

```
## [1] setID pieces theme availability  
## [5] US_retailPrice minifigs Price_per_piece  
## <0 rows> (or 0-length row.names)
```



# Group By and Summarize

```
legosets %>% group_by(themeGroup) %>% summarize(mean_price = mean(US_retailPrice, na.rm = TRUE),  
                                sd_price = sd(US_retailPrice, na.rm = TRUE),  
                                median_price = median(US_retailPrice, na.rm = TRUE),  
                                n = n(),  
                                missing = sum(is.na(US_retailPrice)))
```

```
## # A tibble: 17 × 6  
##   themeGroup     mean_price    sd_price median_price      n missing  
##   <chr>          <dbl>       <dbl>      <dbl> <int>    <int>  
## 1 Action/Adventure  40.2        38.9      30.0  1474     779  
## 2 Art and crafts   34.9        47.7      17.5   97      9  
## 3 Basic             21.6        19.2      15.0   873     733  
## 4 Constraction     16.4        12.4      13.0   502     284  
## 5 Educational       182.        188.      130.   503     465  
## 6 Girls              35.8        24.0      23.0   240     227  
## 7 Historical         34.2        32.4      20.0   473     400  
## 8 Junior             22.0        10.1      20.0   228     165  
## 9 Licensed            53.3        71.7      30.0  2775    1066  
## 10 Miscellaneous      20.7        29.2      13.0  6253    3961  
## 11 Model making       74.3        92.1      40.0   771     384  
## 12 Modern day          38.2        35.6      30.0  2469    1535  
## 13 Pre-school           30.8        22.7      25.0  1562    1103  
## 14 Racing              26.8        26.5      15.0   270     176
```

# Describe and Describe By

```
library(psych)
```

```
describe(legosets$US_retailPrice)
```

```
##      vars     n   mean    sd median trimmed   mad   min   max range skew kurtosis
## X1      1 7483 38.96 56.5  19.99    27.7 17.79 1.49 849.99 848.5 5.32    44.74
##          se
## X1 0.65
```

```
describeBy(legosets$US_retailPrice, group = legosets$availability, mat = TRUE, skew = FALSE)
```

	item	group1	vars	n	mean	sd	median	min	max	range	se
## X11	1	{Not specified}	1	1831	26.84733	39.96747	19.99	1.49	789.99	788.5	0.9340335
## X12	2	Educational	1	12	212.86667	105.88283	222.45	14.95	399.95	385.0	30.5657410
## X13	3	LEGO exclusive	1	1039	57.21203	106.63125	12.99	1.99	849.99	848.0	3.3080857
## X14	4	LEGOLAND exclusive	1	2	4.99000	0.00000	4.99	4.99	4.99	0.0	0.0000000
## X15	5	Not sold	1	1	12.99000	NA	12.99	12.99	12.99	0.0	NA
## X16	6	Promotional	1	5	4.79000	0.83666	4.99	3.99	5.99	2.0	0.3741657
## X17	7	Promotional (Airline)	1	0	NaN	NA	NA	Inf	-Inf	-Inf	NA
## X18	8	Retail	1	4290	37.55889	38.44918	24.99	1.99	699.99	698.0	0.5870275
## X19	9	Retail - limited	1	302	63.54381	70.91908	39.99	2.49	449.99	447.5	4.0809343
## X110	10	Unknown	1	1	3.99000	NA	3.99	3.99	3.99	0.0	NA

# Additional Resources

For data wrangling:

- `dplyr` website: <https://dplyr.tidyverse.org>
- R for Data Science book: <https://r4ds.had.co.nz/wrangle-intro.html>
- Wrangling penguins tutorial: <https://allisonhorst.shinyapps.io/dplyr-learnr/#section-welcome>
- Data transformation cheat sheet: <https://github.com/rstudio/cheatsheets/raw/master/data-transformation.pdf>

# One Minute Paper

Complete the one minute paper:

<https://forms.gle/U4UXAosdjHorxY919>

1. What was the most important thing you learned during this class?
  
2. What important question remains unanswered for you?



# Good luck with the semester!

 jason.bryer@yu.edu

 @jbryer

 @jbryer@vis.social

 [github.com/jbryer/DAV5300-2024-Spring](https://github.com/jbryer/DAV5300-2024-Spring)