

# Visualization for Data Science in R

Angela Zoss

Data Matters Spring 2023

<https://www.angelazoss.com/RVis-2Day/>

Try right now:

Open RStudio

Try running “library(tidyverse)”

Tell me about any errors

# Slides and files

<https://github.com/amzoss/RVis-2Day>

# Schedule, Day 1

Session	Topics	Duration
Session 1	Visualization and data science Intro, setup, basic ggplot2 syntax	9:30 a.m. – 10:35 a.m.
Morning break		10:35 a.m. – 10:50 a.m.
Session 2	Trying more charts	10:50 a.m. – 11:55 a.m.
Lunch		11:55 a.m. – 1:10 p.m.
Session 3	Customizing plots, saving charts out	1:10 p.m. – 2:15 p.m.
Afternoon break		2:15 p.m. – 2:30 p.m.
Session 4	Plot inheritance, advanced examples	2:30 p.m. – 3:35 p.m.
Q&A		3:35 p.m. – 3:40 p.m.

# Schedule, Day 2

Session	Topics	Duration
Session 1	ggplot2 review, advanced techniques	9:30 a.m. – 10:35 a.m.
Morning break		10:35 a.m. – 10:50 a.m.
Session 2	Working with text variables	10:50 a.m. – 11:55 a.m.
Lunch		11:55 a.m. – 1:10 p.m.
Session 3	Simple interactive plots	1:10 p.m. – 2:15 p.m.
Afternoon break		2:15 p.m. – 2:30 p.m.
Session 4	Building visualizations into layouts	2:30 p.m. – 3:35 p.m.
Q&A		3:35 p.m. – 3:40 p.m.

# Set up environment

- R
- RStudio
- packages

## Packages:

- **tidyverse**
- **readxl**
- **markdown**
- **knitr**
- **shiny**
- **plotly**
- maps
- mapproj
- sf
- here

# Visualization for Data Science

# Why visualize in R?

- Quickly explore data
- Save time switching to another tool
- Use charts to inspire new analyses and vice versa
- Reproducibility

# Why care about reproducibility?

- Open science makes review easier
- Increasingly a requirement
- Saves you a lot of time trying to figure out what you did last time!

*“Your closest collaborator is **you** six months ago, but you don’t reply to emails.”*

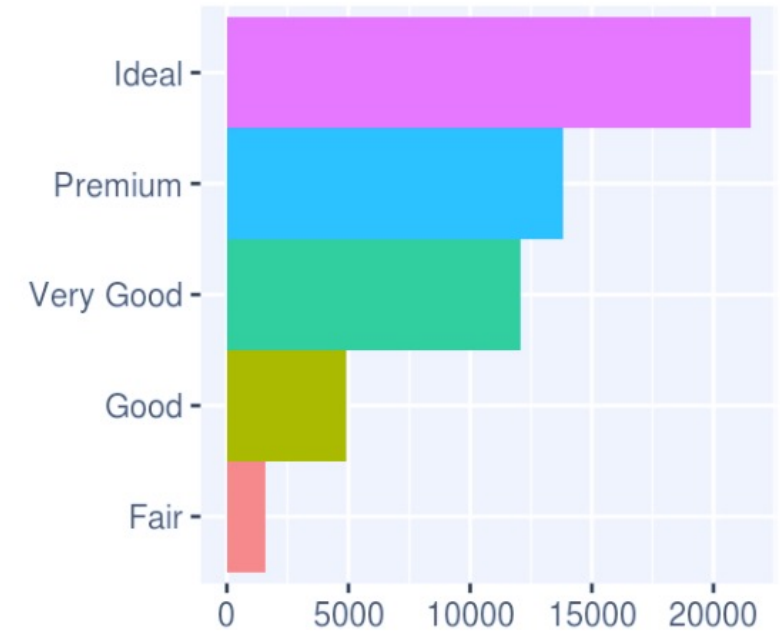
*- Mark Holder*



ggplot2

# What is ggplot2?

an R package designed to create plots based on a theory of the grammar of graphics.

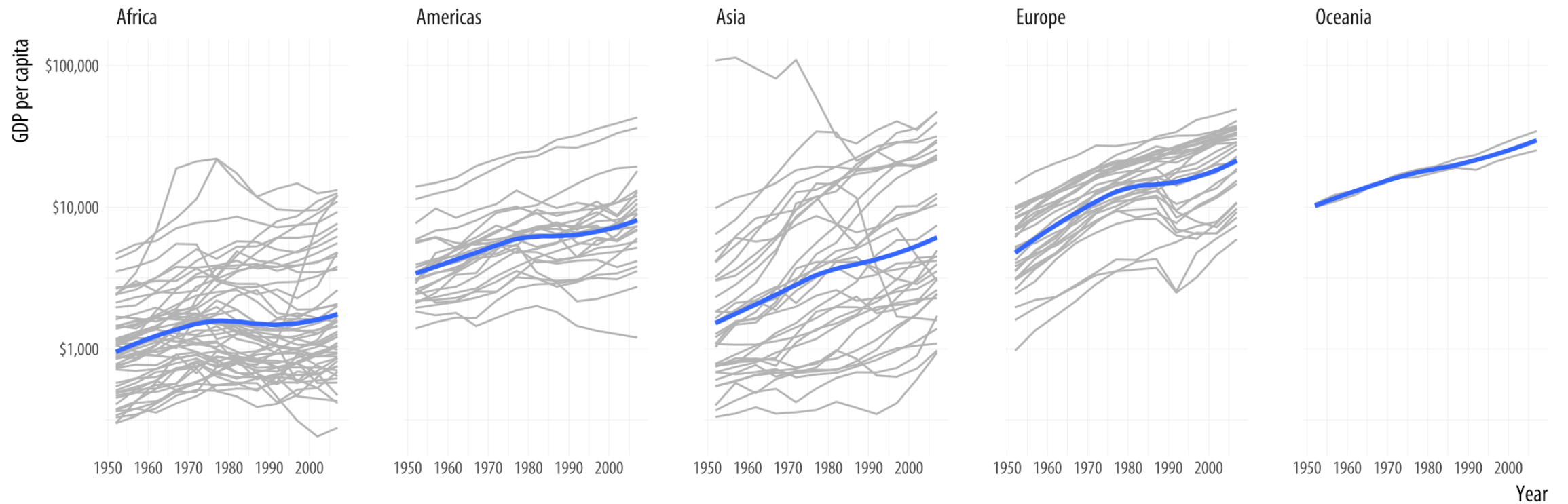


# Grammar of graphics

1. DATA: a set of data operations that create variables from datasets
2. TRANS: variable transformations (e.g., rank)
3. SCALE: scale transformations (e.g., log)
4. COORD: a coordinate system (e.g., polar)
5. ELEMENT: graphs (e.g., points) and their aesthetic attributes (e.g., color)
6. GUIDE: one or more guides (axes, legends, etc.).

ggplot2 examples

## GDP per capita on Five Continents

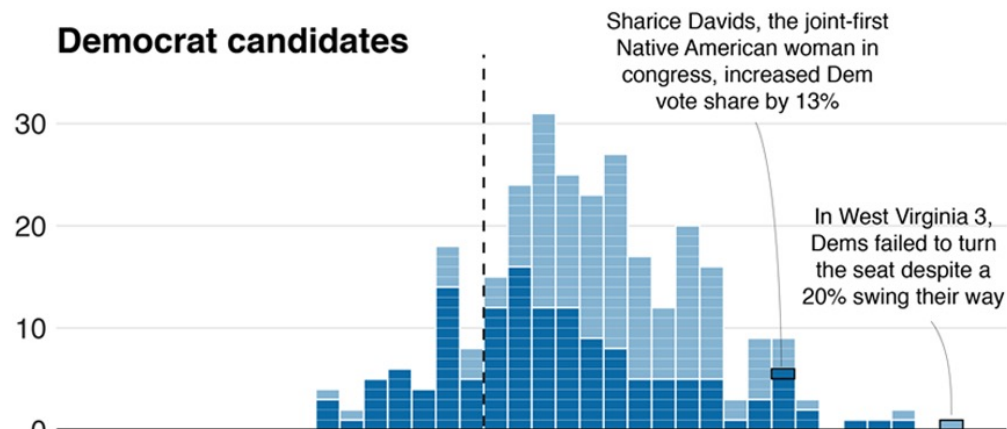


<http://socviz.co/groupfacettx.html>

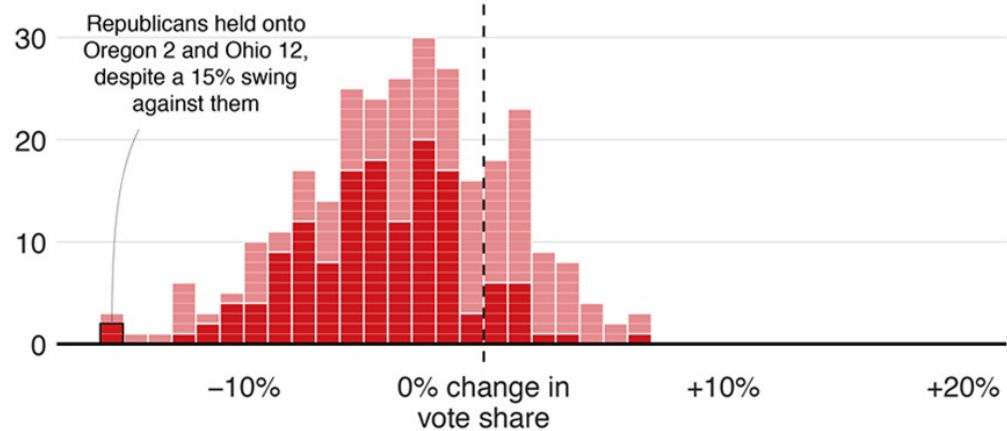
## Blue wave

■ Won seat ■ Didn't win

### Democrat candidates



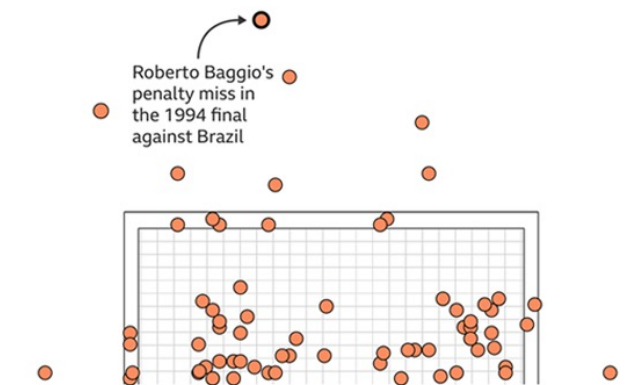
### Republican candidates



Source: AP, 19:01 ET

## Where penalties are saved

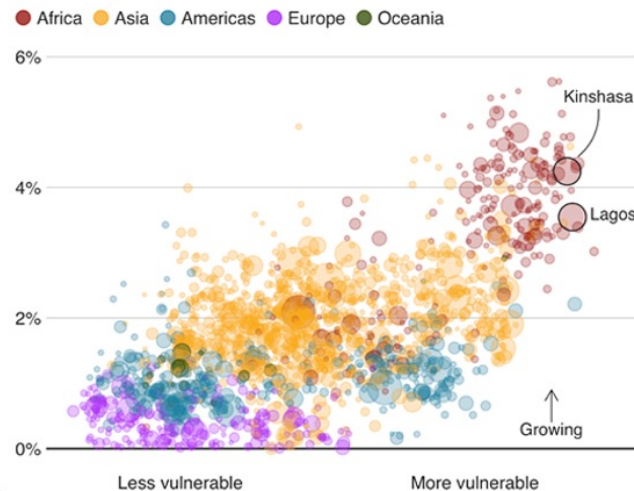
World Cup shootout misses and saves, 1982-2014



Source: Opta

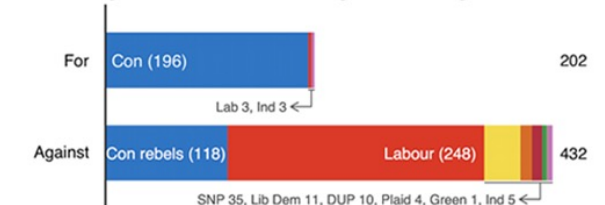
## Fast-growing cities face worse climate risks

Population growth 2018-2035 over climate change vulnerability



Source: Verisk Maplecroft. Circle size represents current population.

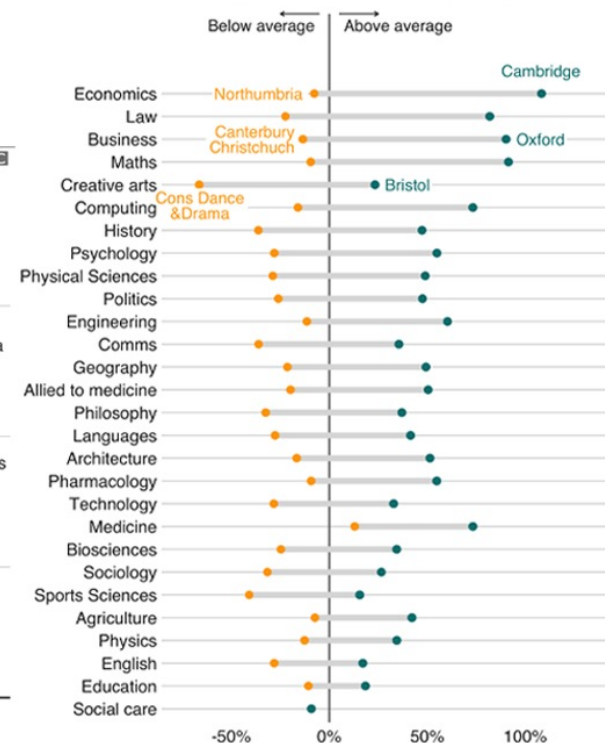
## MPs rejected Theresa May's deal by 230 votes



Source: Commons Votes Services. Excludes 'tellers', the Speaker and deputies

## Earnings vary across units even within subjects

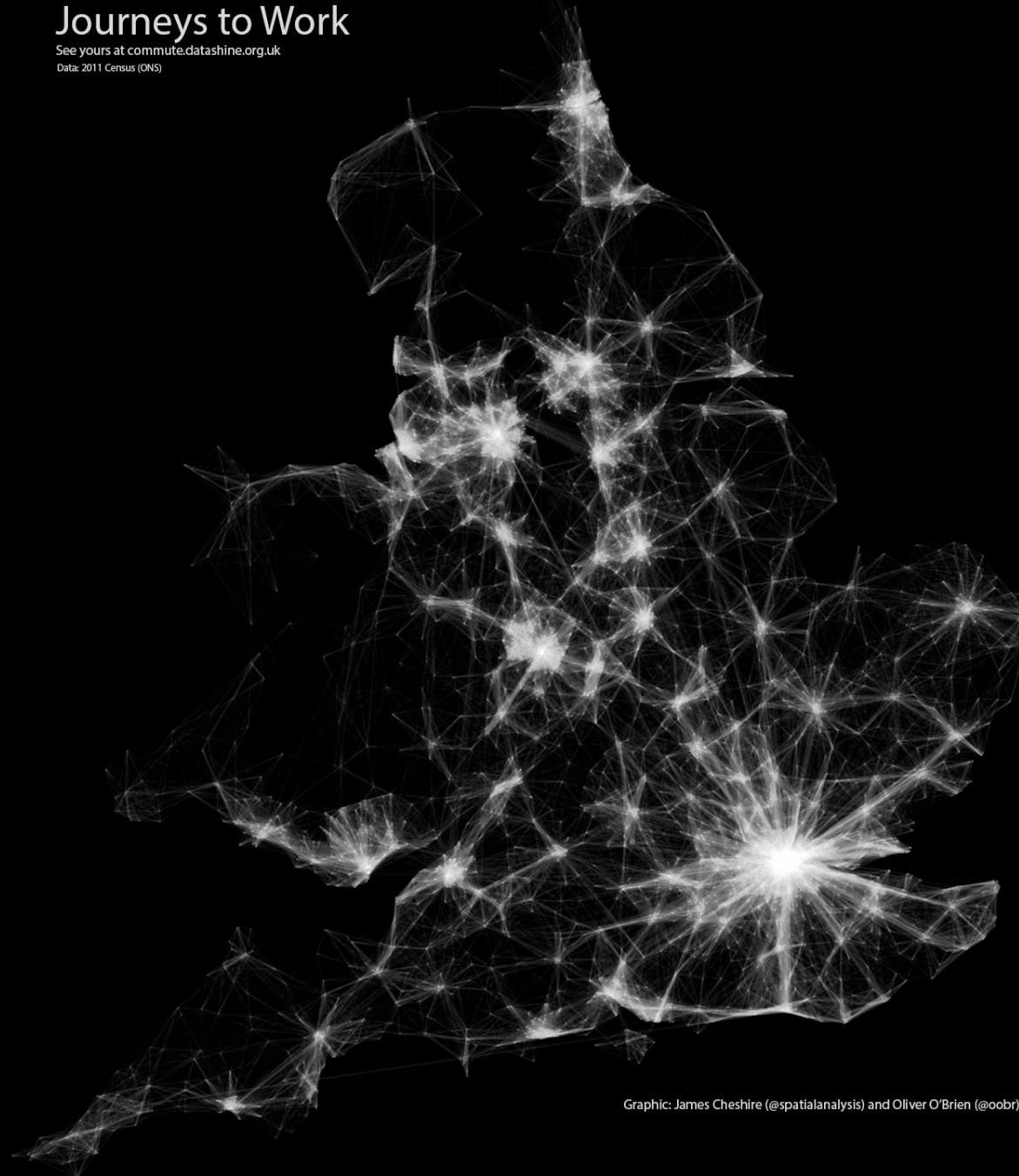
Impact on men's earnings relative to the average degree



Source: Institute for Fiscal Studies

# Journeys to Work

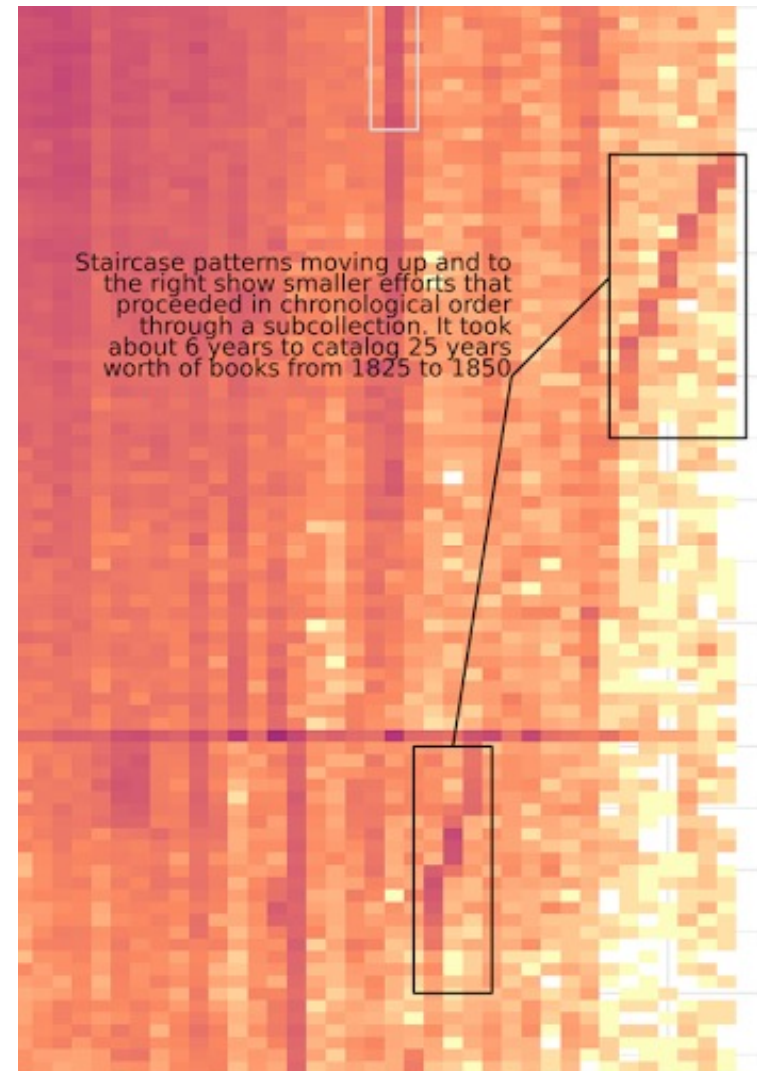
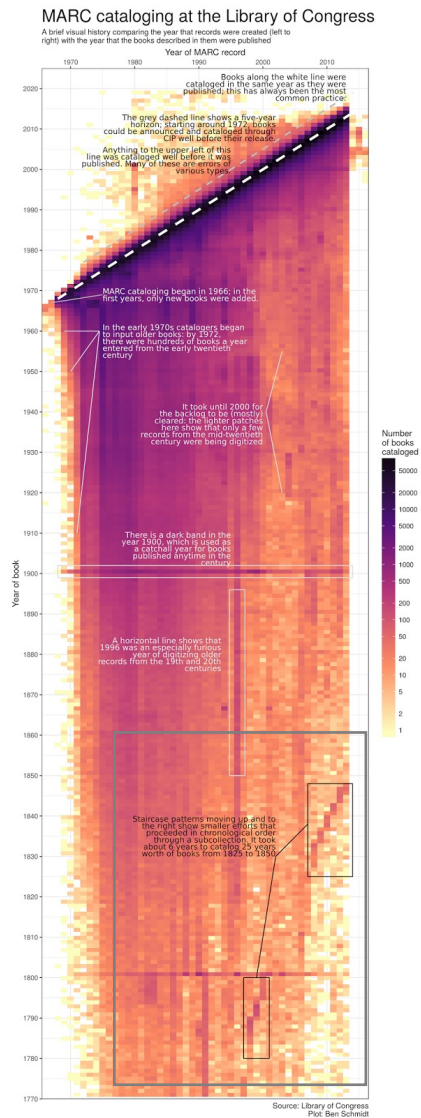
See yours at [commute.datashine.org.uk](http://commute.datashine.org.uk)  
Data: 2011 Census (ONS)



Graphic: James Cheshire (@spatialanalysis) and Oliver O'Brien (@oobr)

<http://spatial.ly/2015/03/mapping-flows/>





<http://sappingattention.blogspot.com/2017/05/a-brief-visual-history-of-marc.html>



# Why ggplot2 instead of base R?

- nice defaults
- easy faceting
- (arguably) more natural syntax
- can switch chart types more easily

“Why I use ggplot2”, David Robinson

<http://varianceexplained.org/r/why-i-use-ggplot2/>

# R vs. Excel, Tableau, etc.

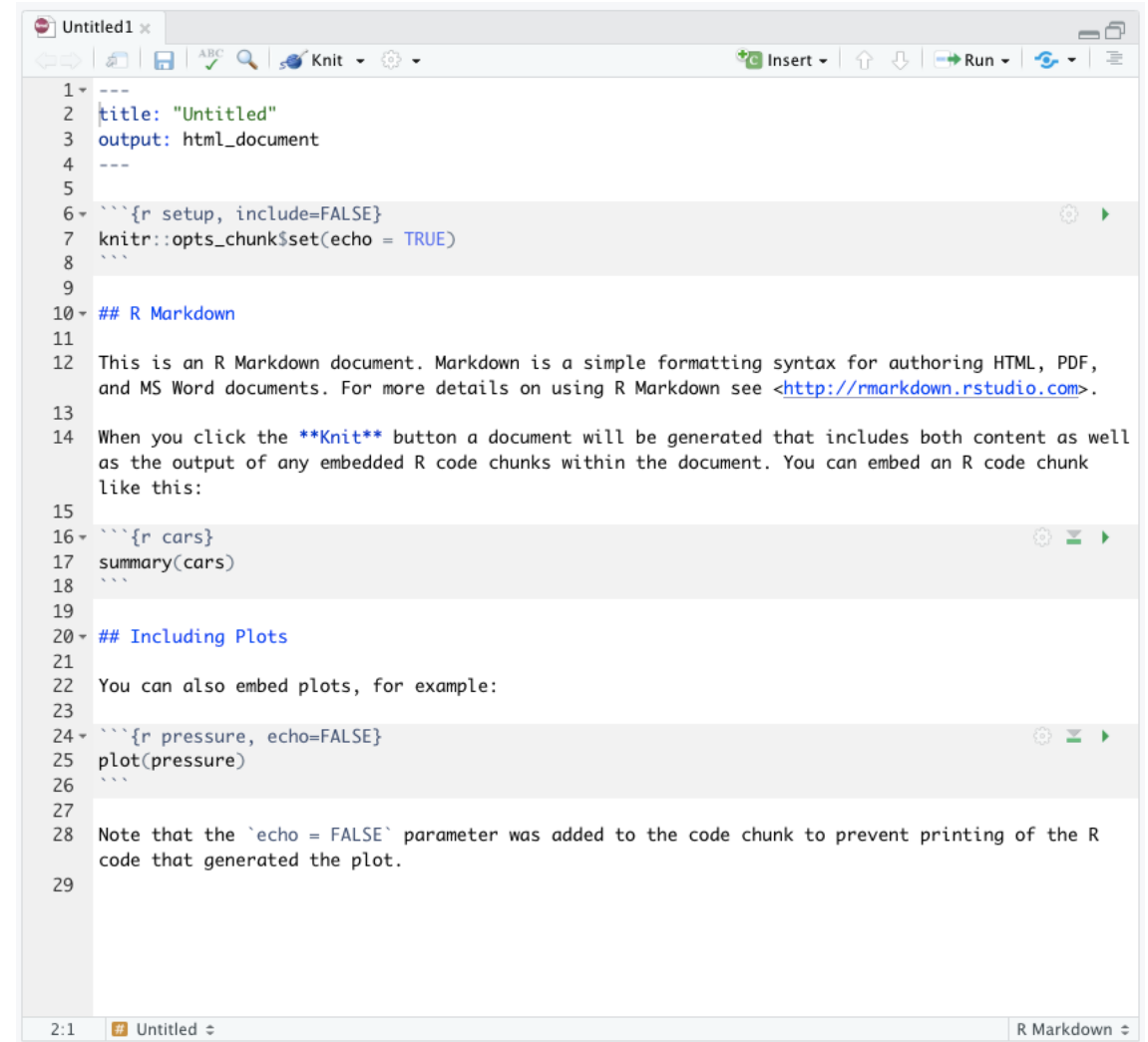
Questions to ask:

- Are you already using R? Why switch?
- Are you going to have to share this process or reproduce it? Try R!
- Is it a quick project, or will others work on it? Maybe Excel is fine.
- Do you need to try a bunch of charts quickly, build interactive components, etc.? Tableau might be more powerful and faster.

Working in RStudio

# R Markdown files

- Blend “normal” text (using Markdown syntax for formatting) with code chunks and their output
- Can be compiled (“knit”) into other formats (HTML, Word, PDF)
- Similar to Jupyter Notebooks for Python
- NB: The next generation of R Markdown is [Quarto](#)



```
1 ---
2 title: "Untitled"
3 output: html_document
4 ---
5
6 ```{r setup, include=FALSE}
7 knitr::opts_chunk$set(echo = TRUE)
8 ```
9
10 ## R Markdown
11
12 This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF,
13 and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
14
15 When you click the Knit button a document will be generated that includes both content as well
16 as the output of any embedded R code chunks within the document. You can embed an R code chunk
17 like this:
18
19 ```{r cars}
20 summary(cars)
21 ```
22
23 ## Including Plots
24
25 You can also embed plots, for example:
26
27 ```{r pressure, echo=FALSE}
28 plot(pressure)
29 ```
30
31 Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R
32 code that generated the plot.
```

# Why R Markdown?

- Plots show up inline
- Easier to incorporate explanatory text and materials
- Like to be able to easily run one chunk at a time

Caution: Running things out of order can mean your code won't work again later. Clear your environment often and run code chunks in order to be safe.

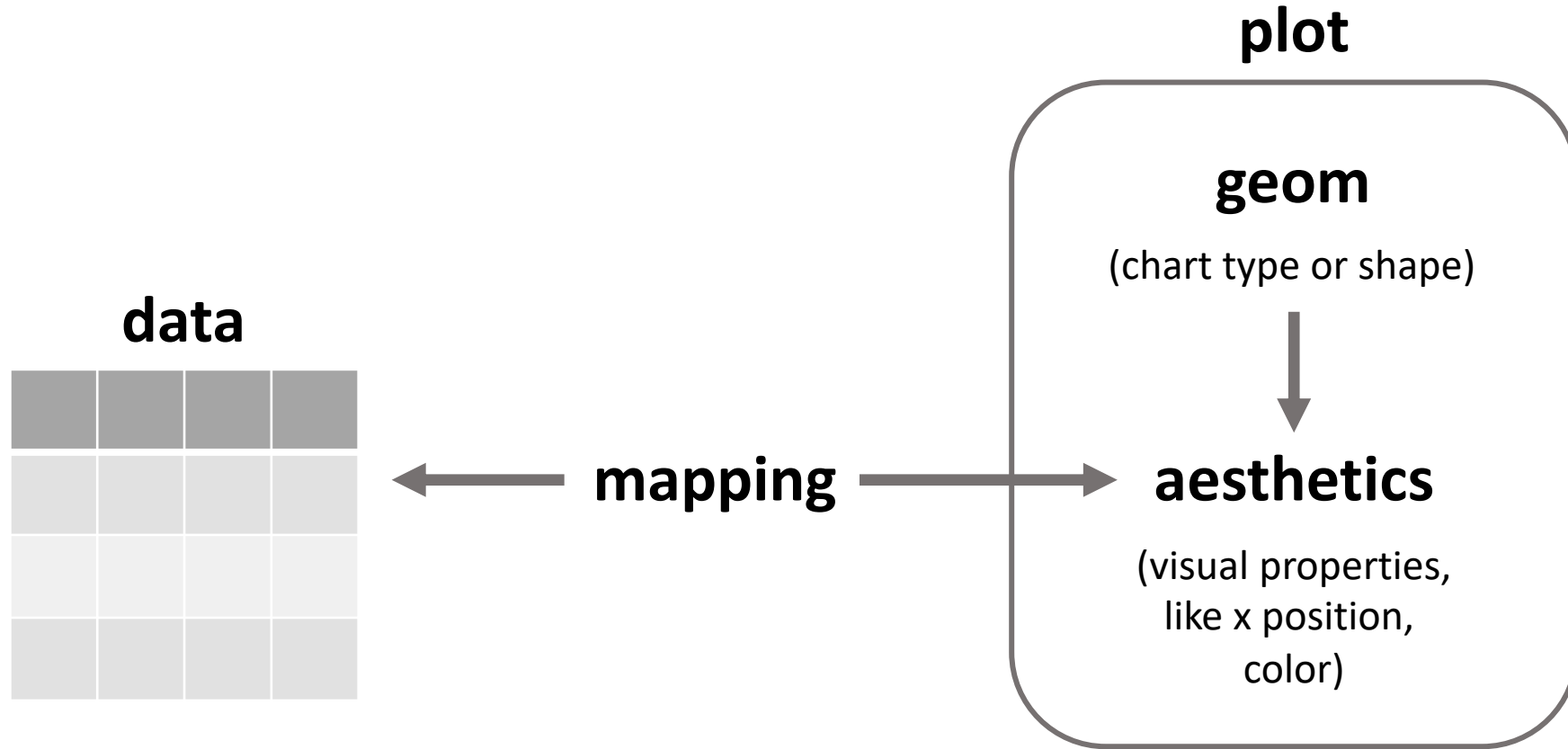
# R Markdown test

- File → New File → R Markdown
- Click OK to accept defaults
- Type inside the first few lines to edit the YAML header (edit title, add author, etc.)
- Add a new R code chunk at the end of the file using Insert → R
- Type some R code inside the code chunk:  
**library(tidyverse)**
- Run the new code chunk

```
29  
30 ```{r}  
31  
32 library(tidyverse)|  
33  
34 ```  
35
```

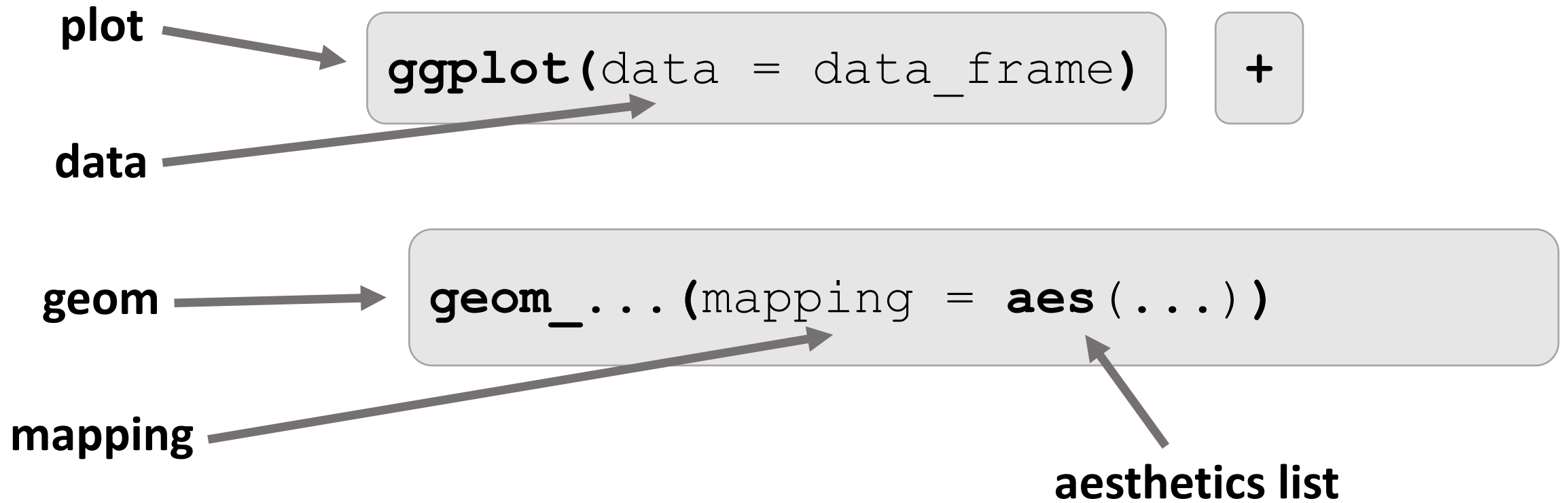
ggplot2: making a basic plot

# Basic elements in any ggplot2 visualization





# Template for a simple plot

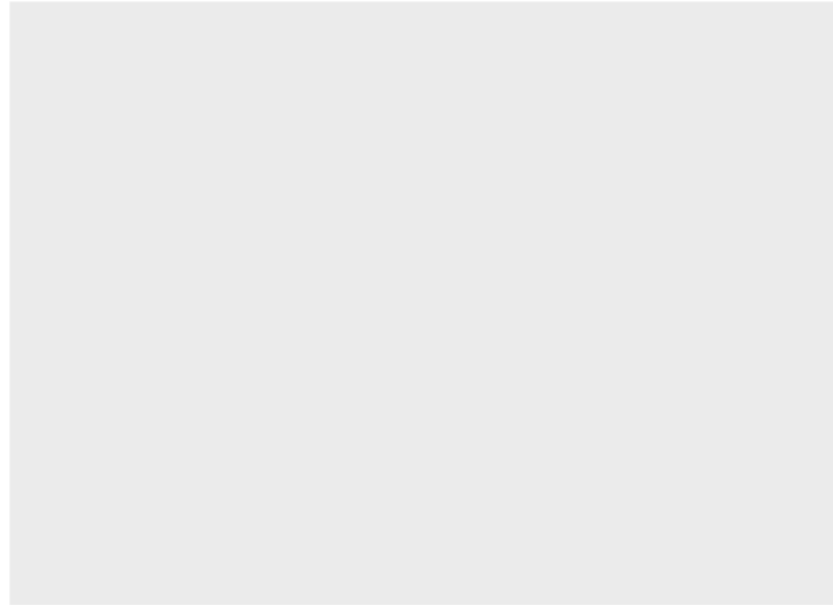


# 1. Set the data

“iris”

Petal.Width	Petal.Length	Species
0.3	1.4	setosa
1.3	4.0	versicolor
2.1	5.7	virginica

```
ggplot(data=iris)
```



## 2. Choose a shape layer

“iris”

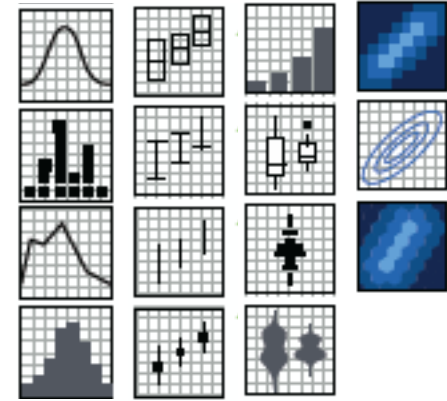
Petal.Width	Petal.Length	Species
0.3	1.4	setosa
1.3	4.0	versicolor
2.1	5.7	virginica

```
ggplot(data=iris) +  
  geom_point()
```

Error: geom\_point requires  
the following missing  
aesthetics: x and y

# Types of geoms

- `geom_bar()`
- `geom_point()`
- `geom_histogram()`
- `geom_map()`
- etc.



<http://bit.ly/ggplot2-cheatsheet>

### 3. Map variables to aesthetics

“iris”

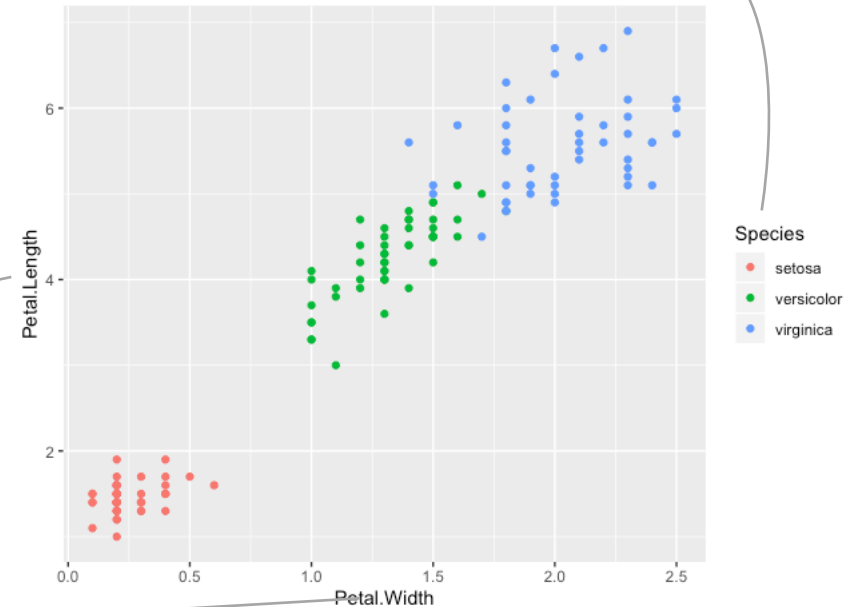
Petal.Width	Petal.Length	Species
0.3	1.4	setosa
1.3	4.0	versicolor
2.1	5.7	virginica

x position

y position

color

```
ggplot(data=iris) +  
  geom_point(  
    mapping=aes(x=Petal.Width,  
                 y=Petal.Length,  
                 color=Species))
```

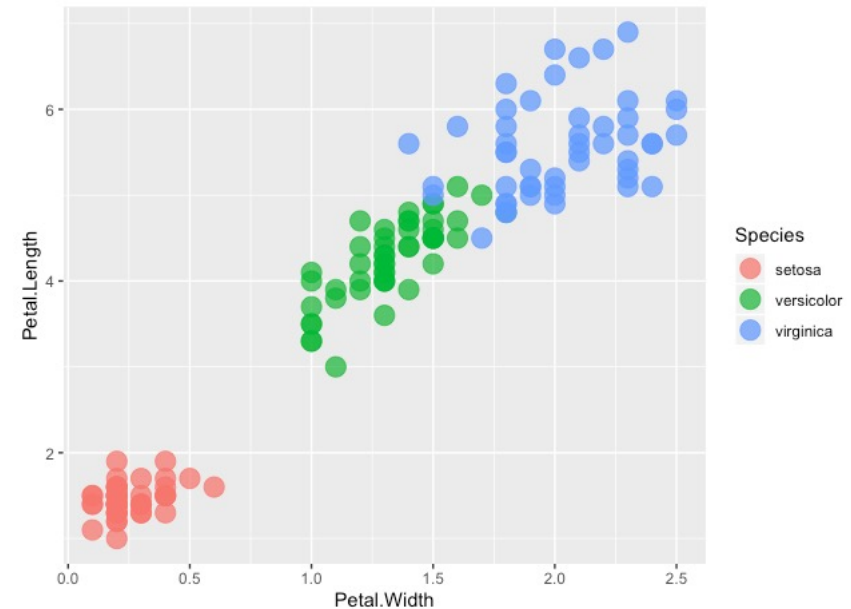


## 4. Add non-variable adjustments

“iris”


Petal.Width	Petal.Length	Species
0.3	1.4	setosa
1.3	4.0	versicolor
2.1	5.7	virginica

```
ggplot(data=iris) +  
  geom_point(  
    mapping=aes(x=Petal.Width,  
                 y=Petal.Length,  
                 color=Species),  
    size=5, alpha=.75)
```



# Fixing Errors

# Debugging code

- Start simple
  - If you see an error:
    - read error message for hints
    - check for problems with spelling/punctuation marks
  - Get code to run without errors
  - Check result to see if it makes sense
- 
- Add a small change
  - Get code to run without errors
  - Check result to see if it makes sense
  - etc.



# RStudio built-in help documentation

- In console, type  
`?<function or package name>`
- In help tab (not help menu), type into main search box
- In package tab, click on package name
- In help menu, use the Cheat Sheets submenu to download cheat sheet PDFs


# ggplot2 Cheat Sheet

Help →

Cheatsheets →

## Data Visualization with ggplot2

### Data Visualization with ggplot2 : : CHEAT SHEET



#### Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data set**, a **coordinate system**, and **geoms**—visual marks that represent data points.

To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.

Complete the template below to build a graph.

```
ggplot(data = <DATA>) +  
  <GEOM FUNCTION> (mapping = aes(<MAPPINGS>))  
  stat = <STAT>, position = <POSITION> +  
  <COORDINATE FUNCTION> +  
  <SCALE FUNCTION> +  
  <THEME FUNCTION>
```

ggplot(data = mpg, aes(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

qplot(x = cty, y = hwy, data = mpg, geom = "point") Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

last\_plot() Returns the last plot

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5" x 5" file named "plot.png" in working directory. Matches file type to file extension.

#### Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

##### GRAPHICAL PRIMITIVES

```
a <- ggplot(economics, aes(date, unemploy))  
b <- ggplot(seals, aes(x = long, y = lat))  
a + geom_blank()  
b + geom_curve(aes(yend = lat + 1, xend = long + 1, curvature = z))  
a + geom_path(lineend = "butt", linejoin = "round", linemitre = 1)  
a + geom_polygon(aes(group = group))  
b + geom_rect(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1))  
a + geom_ribbon(aes(ymin = unemploy - 900, ymax = unemploy + 900))
```

##### LINE SEGMENTS

```
b + geom_abline(aes(intercept = 0, slope = 1))  
b + geom_hline(aes(yintercept = lat))  
b + geom_vline(aes(xintercept = long))  
b + geom_segment(aes(yend = lat + 1, xend = long + 1))  
b + geom_spoke(aes(angle = 1:1155, radius = 1))
```

##### ONE VARIABLE continuous

```
c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)  
c + geom_area(stat = "bin")  
c + geom_density(kernel = "gaussian")  
c + geom_dotplot()  
c + geom_freqpoly()  
c + geom_histogram(binwidth = 5)  
c2 + geom_qq(aes(sample = hwy))
```

##### discrete

```
d <- ggplot(mpg, aes(f))  
d + geom_bar()
```

##### TWO VARIABLES

###### continuous x, continuous y

```
e <- ggplot(mpg, aes(cty, hwy))  
e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE)  
e + geom_jitter(height = 2, width = 2)  
e + geom_point()  
e + geom_quantile()  
e + geom_rug(sides = "bl")  
e + geom_smooth(method = lm)  
e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE)
```

###### discrete x, continuous y

```
f <- ggplot(mpg, aes(class, hwy))  
f + geom_boxplot()  
f + geom_dotplot(binaxis = "y", stackdir = "center")  
f + geom_violin(scale = "area")
```

##### discrete x, discrete y

```
g <- ggplot(diamonds, aes(cut, color))  
g + geom_count()
```

##### THREE VARIABLES

```
sealsSz <- with(seals, sqrt(delta_long^2 + delta_lat^2))  
i <- ggplot(seals, aes(long, lat))  
i + geom_contour(aes(z = z))  
i + geom_raster(aes(fill = z), interpolate = FALSE)  
i + geom_tile(aes(fill = z))
```

##### continuous bivariate distribution

```
h <- ggplot(diamonds, aes(carat, price))  
h + geom_bin2d(binwidth = c(0.25, 500))  
h + geom_density2d()  
h + geom_hex()
```

##### continuous function


```
i <- ggplot(economics, aes(date, unemploy))  
i + geom_area()  
i + geom_line()  
i + geom_step(direction = "hv")
```

##### visualizing error

```
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)  
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))  
j + geom_crossbar(fatten = 2)  
j + geom_errorbar()  
j + geom_linerange()  
j + geom_pointrange()
```

##### maps

```
data <- data.frame(murder = USArrests$Murder, state = tolower(rownames(USArrests)))  
map <- map_data("state")  
k <- ggplot(data, aes(fill = murder))  
k + geom_map(aes(map_id = state), map = map)  
k + expand_limits(x = map$long, y = map$lat)  
k + geom_tile(aes(fill = murder))
```



RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more at <http://ggplot2.tidyverse.org> • ggplot2 2.1.0 • Updated: 2016-11

<https://www.rstudio.com/resources/cheatsheets/#ggplot2>

# Get workshop files

URL: <https://github.com/amzoss/RVis-2Day>

## On GitHub:

- Click green “Code” button and select “Download ZIP”
- Unzip files on your computer
  - Windows: Double-click, then look for “Extract Files” at the top
  - Mac: Double-click
- Note: have noticed some issues when using OneDrive to store files

## In RStudio:

- Project → New project...
- Existing directory
- Select unzipped folder
- Create Project

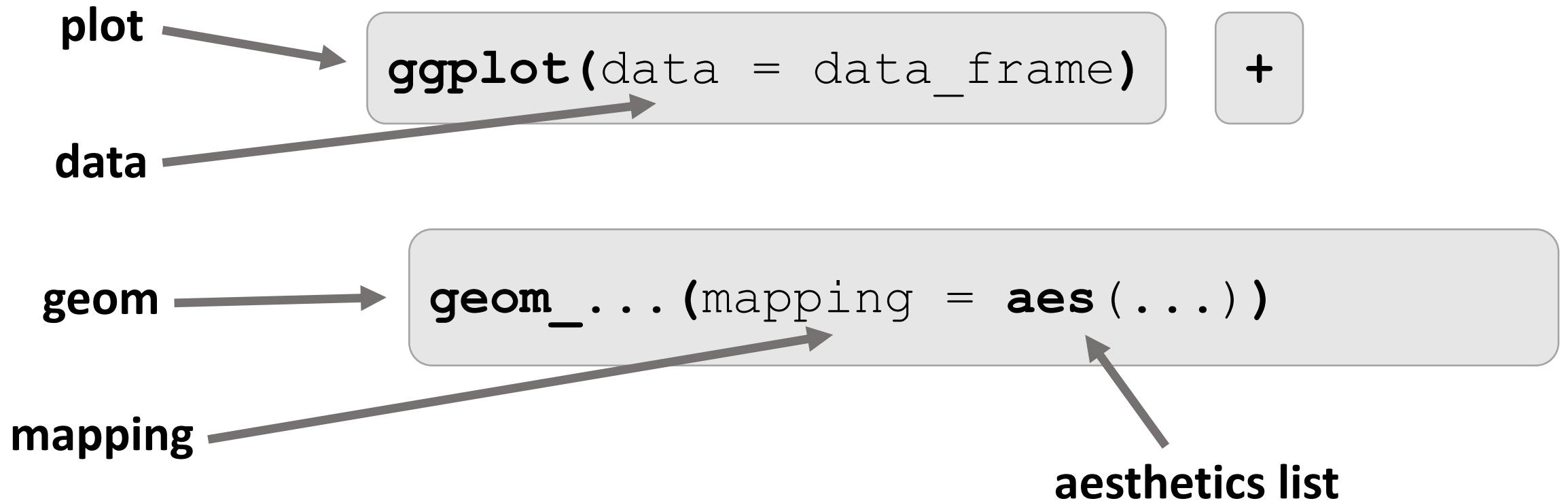
Morning Break

# Exercise 1:

# Inclusiveness Index

<https://belonging.berkeley.edu/inclusivenessindex>

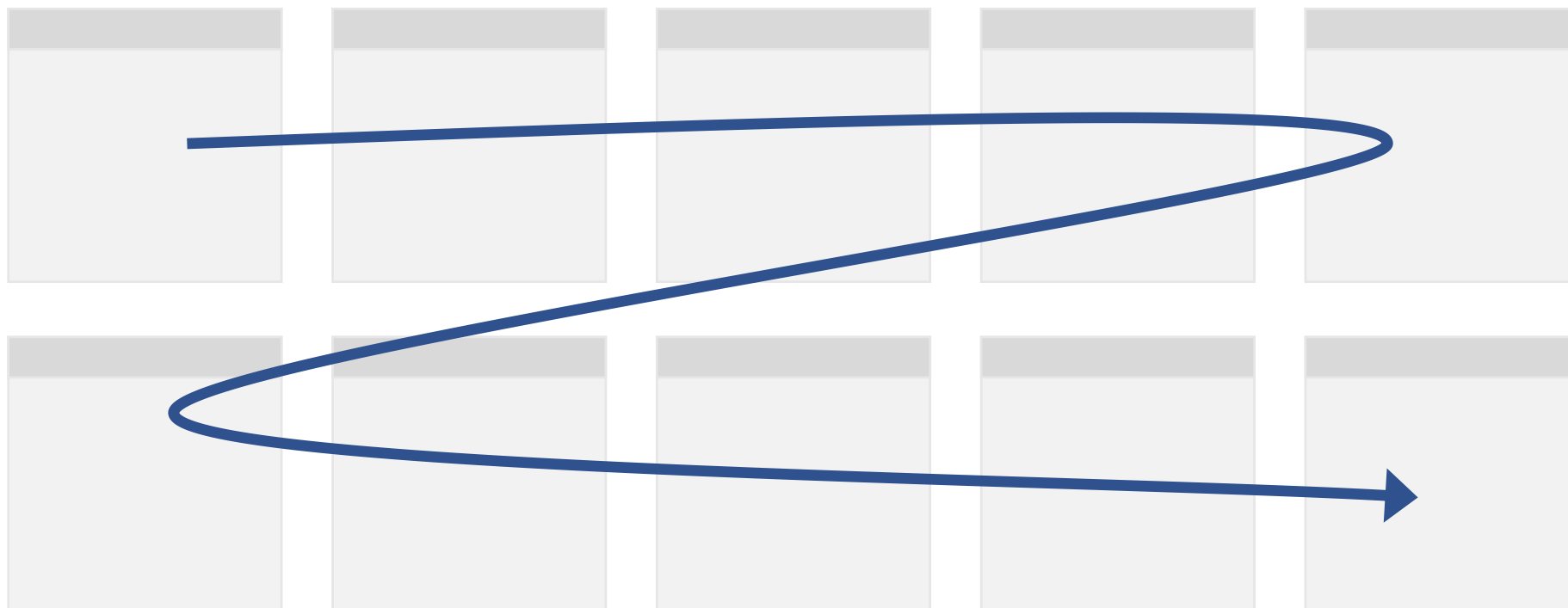
# Template for a simple plot



Creating repeated charts

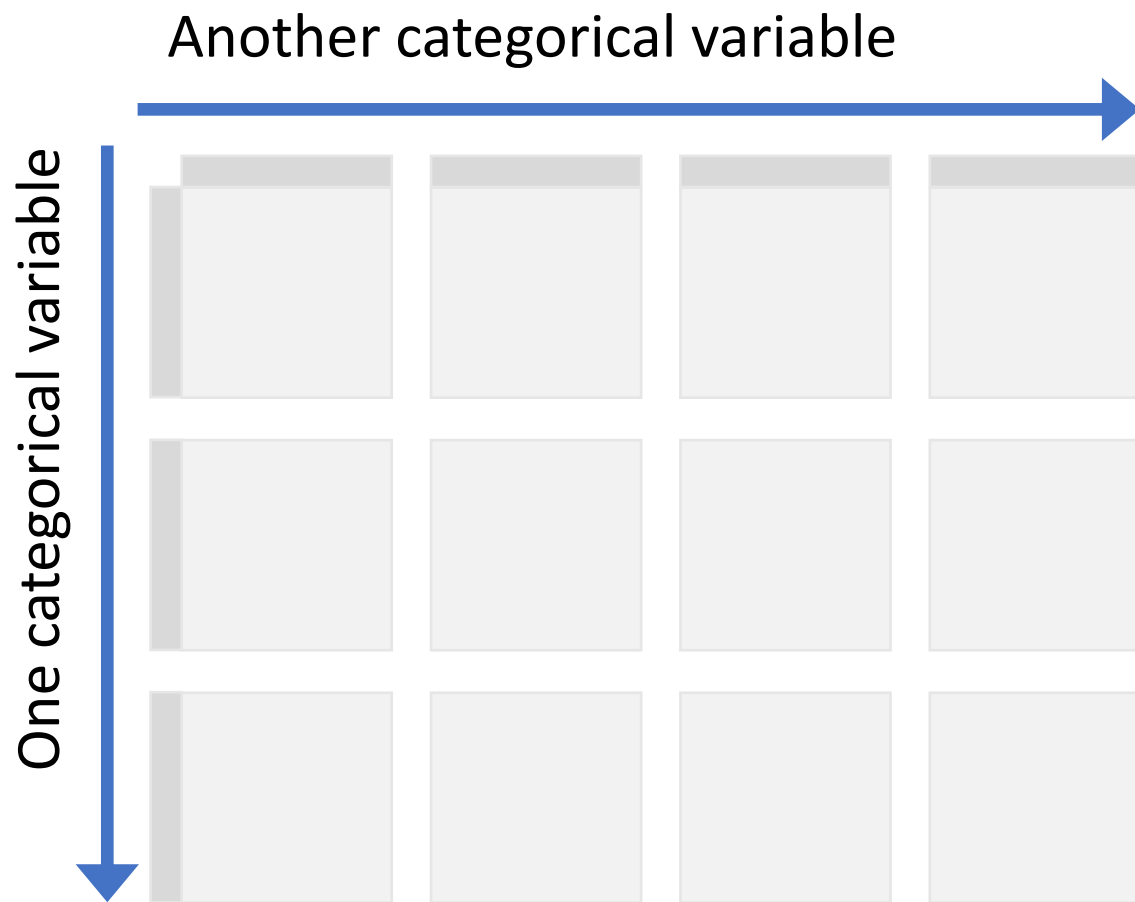
# facet\_wrap()

```
+ facet_wrap(vars(variable))
```





# facet\_grid()

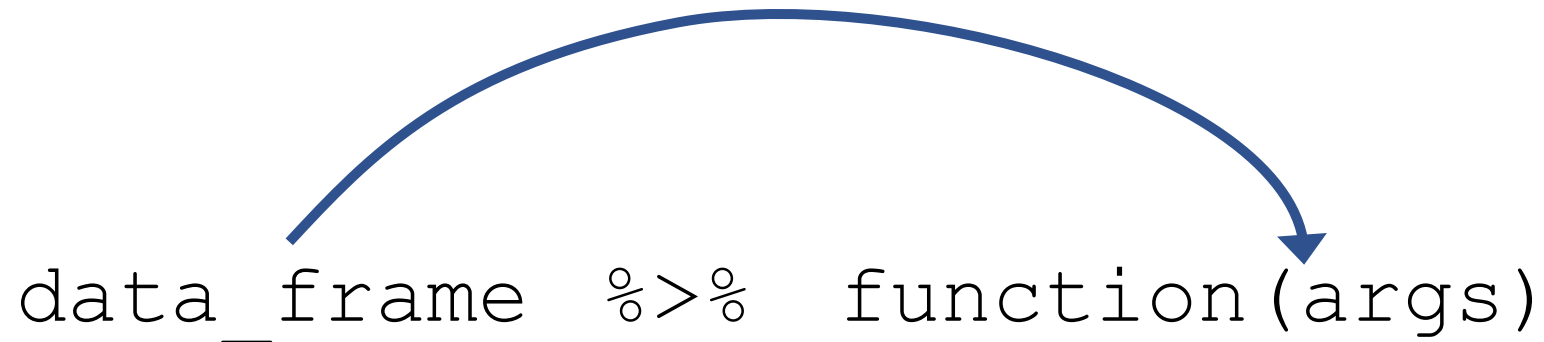


```
+ facet_grid(rows=vars(yvar,  
                    cols=vars(xvar))
```

Helpful data manipulation

# Note: about %>%

- Loads automatically with tidyverse
- Used throughout tidyverse (except for ggplot2)
- Pushes data from the left into the function on the right



# filter

Select a subset of rows

```
data %>% dplyr::filter(name == "John")
```

same as

```
dplyr::filter(data, name == "John")
```

<https://www.rstudio.com/resources/cheatsheets/#dplyr>

# select

Select a subset of columns (many options!)

```
data %>% dplyr::select(id, name, age)
```

```
data %>% dplyr::select(-count)
```

<https://www.rstudio.com/resources/cheatsheets/#dplyr>

# drop\_na

Remove rows with NA values, either in any column or in specified columns

```
data %>% drop_na()
```

```
data %>% drop_na(age)
```

<https://www.rstudio.com/resources/cheatsheets/> (Data Import with Tidyr Cheatsheet)

# count

Take a dataset, group it by one or more variables, and count the number of rows grouped. Count will be stored in a variable called “n”.

```
data %>% count(sex)
```

sex	n
m	23
f	45

```
data %>% count(sex, marital_status)
```

sex	marital_status	n
m	married	18
m	unmarried	5
f	married	31
f	unmarried	14

count is same as group\_by -> summarise

count() is shorthand for grouping by the categorical variable and then summarizing by the number of rows in each group.

```
data %>% count(sex)
```

sex	n
m	23
f	45

```
data %>% group_by(sex) %>%  
  summarise(n = n())
```

sex	n
m	23
f	45



# Pipe data into ggplot

When doing data manipulation, can be easier to pipe results to ggplot

```
data_frame %>% ggplot()
```

same as

```
ggplot(data = data_frame)
```

Lunch

# Exercise 2: Customizing charts

Accessibility

All graphics need alternative text for screen reader users.

alt= "**Chart type** of **type of data**  
where **reason for including chart**"

Include a **link to data source**  
somewhere in the text

[Writing alt text for data visualization/](#)

# Alternative Text in R and R Markdown

- ggplot2 now has [alt option in labs\(\)](#); gets read by shiny but not knitr
- in the meantime, use [fig.alt](#) in code chunk (new, just for HTML output)
  - can use [fig.cap](#) in code chunk as a backup, but will display in page
- embedded images in the Markdown:  
`![combined alt text and figure caption](path/to/image)`

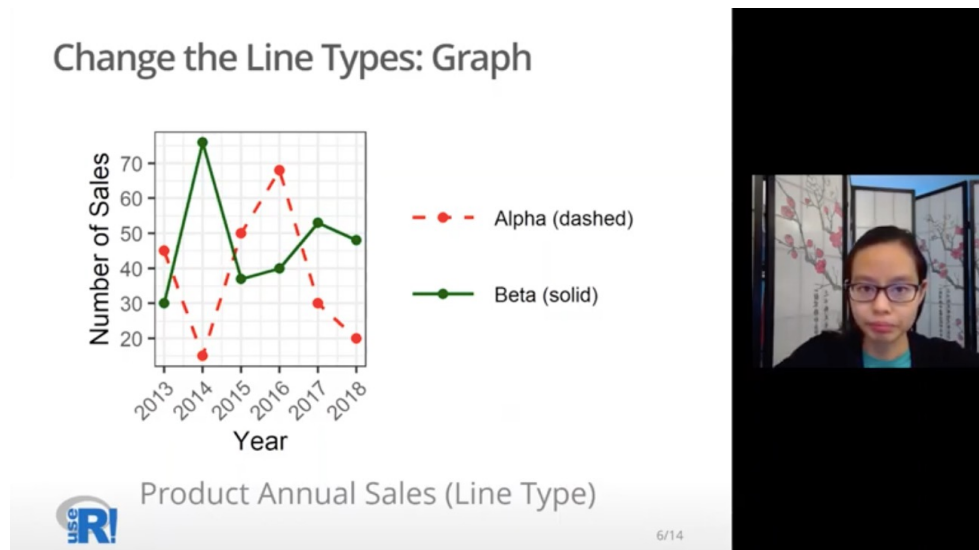
Note: Alt text should be relatively short.

For longer descriptions, use the [savonliquide package](#)

# Color Vision Deficiency

## Use dual encoding (never just color)

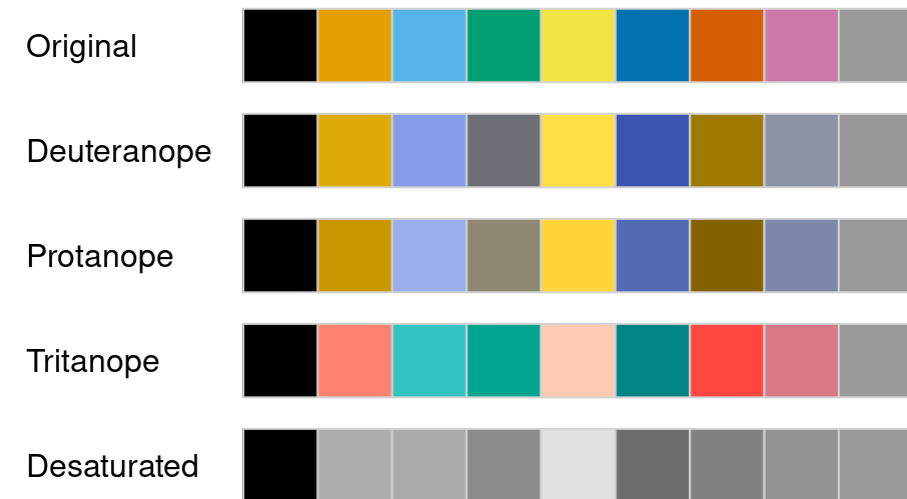
- Line color – also vary line type
- Point color – also vary point shape



[https://www.youtube.com/watch?v=mbi\\_JVC1arM](https://www.youtube.com/watch?v=mbi_JVC1arM)

## Use safe color palettes

- [colorspace package](http://colorspace.r-forge.r-project.org/index.html)



<http://colorspace.r-forge.r-project.org/index.html>

# Low Vision

- High color contrast
  - Both marks/text on background and labels on marks
  - Check with [savonliquide package](#)
- Large text
  - See [“output-examples” file](#) for more sample code
  - Will cover in a later session



# Converting graphics to sound, touch, text

- sonify package
- tactileR package
- [BrailleR package](#)
  - Note: set plot title, subtitle, caption using labs()

# Accessibility Resources

- [savonliquide package](#)
- [Making betterR figures: Accessibility and Universal Design](#)
- [Highlights from the DVS accessibility fireside chat](#)

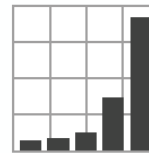
# Scales

- Scales control how an aesthetics mapping displays in the chart, e.g.:
  - the labels that show up on the axis
  - the number of example sizes in a size legend
  - the colors used for a “fill” or “color” mapping
- Modify these properties by adding a scale layer to the chart

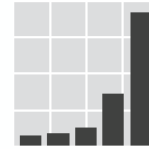
```
scale_x_continuous()  
scale_y_log10()  
scale_fill_discrete()
```

# Themes

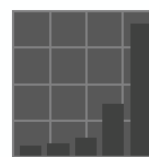
- Themes control properties of various visual elements, including:
  - Axis titles, text, ticks, lines
  - Plot colors, margins, text
  - Legend colors, margins, text
- Can add built-in themes as new layers, override specific theme elements, or build your own custom theme



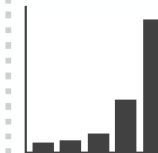
**r + theme\_bw()**  
White background with grid lines.



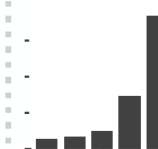
**r + theme\_gray()**  
Grey background (default theme).



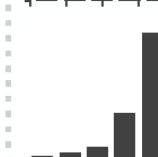
**r + theme\_dark()**  
Dark for contrast.



**r + theme\_classic()**  
**r + theme\_light()**



**r + theme\_linedraw()**  
**r + theme\_minimal()**  
Minimal theme.



**r + theme\_void()**  
Empty theme.

<https://github.com/rstudio/cheatsheets/blob/master/data-visualization-2.1.pdf>

# geom vs. scale vs. theme

Adding something that will appear  
inside the **chart coordinate space**?

You will (almost always) be adding a **geom**!

Changing the way a **variable is displayed**?  
(e.g., different axis breaks, different color mapping)

You will be adding a **scale**!

Changing the **look and feel** of the chart?

You will be adding or making changes to a **theme**!

# More practice: Advanced ggplot2 workshop

[Workshop video](#)

[Workshop materials](#)

Afternoon Break

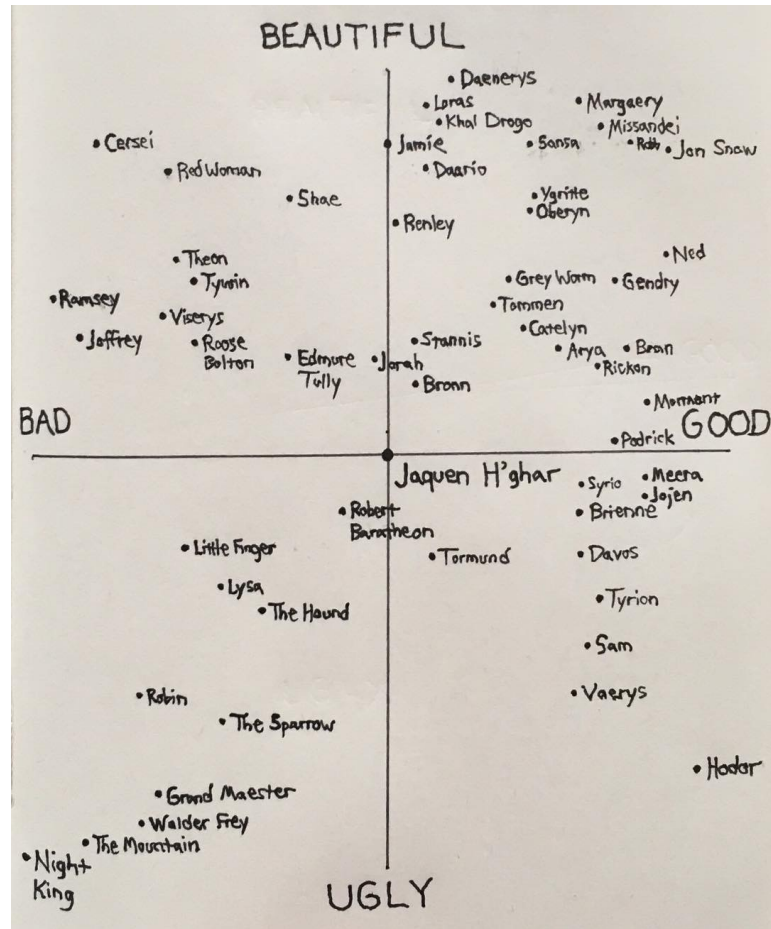
# Exercise 3:

## Game of Thrones character ratings

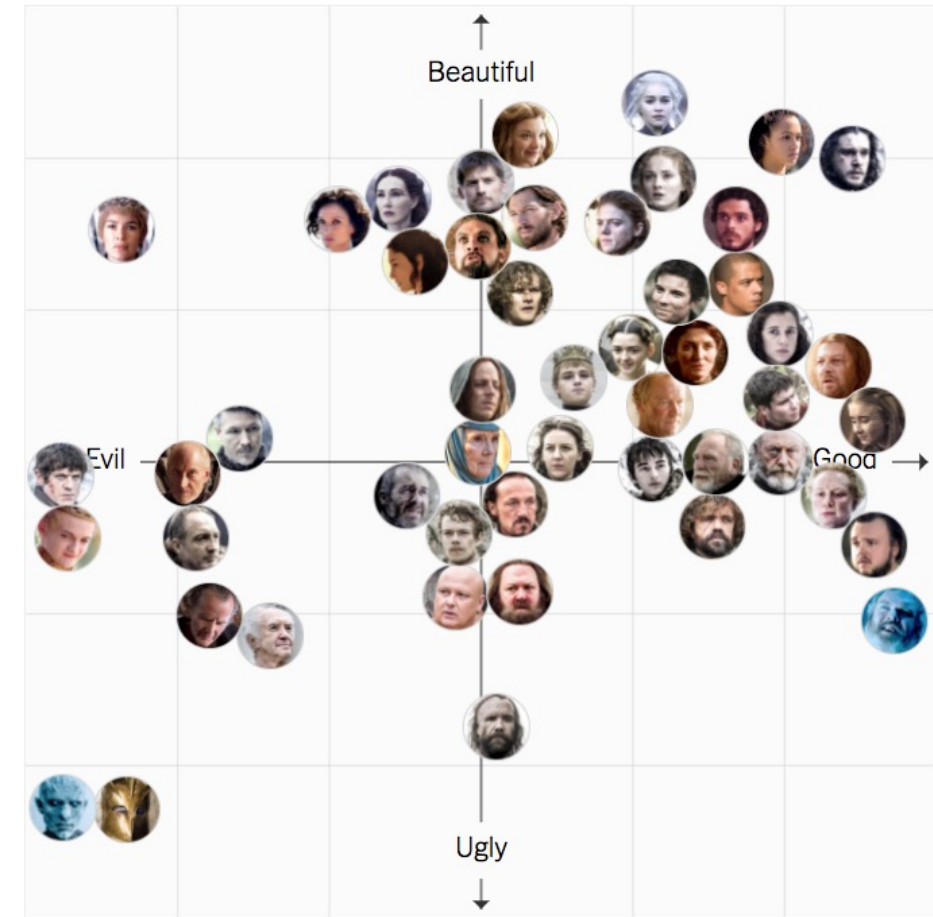
<https://www.nytimes.com/interactive/2017/08/09/upshot/game-of-thrones-chart.html>



# Game of Thrones character ratings



<https://www.instagram.com/p/BWnn-YogX1n/>



<https://www.nytimes.com/interactive/2017/08/09/upshot/game-of-thrones-chart.html>

ggplot2: inheritance

# Template for a simple plot (review)

**main plot  
function**

```
ggplot(data = data_frame)
```

+

**shape  
layer**

```
geom_...(mapping = aes(...),  
           non-variable adjustments)
```

# Expanded template

**main plot  
function**

```
ggplot(data = data_frame,  
        mapping = aes(...))
```

+

**shape  
layer**

```
geom_...(data = data_frame,  
           mapping = aes(...),  
           non-variable adjustments)
```

# Inheritance

data and aesthetics will carry through  
from main function to shape layers

main plot  
function

```
ggplot(data = data_frame,  
       mapping = aes(...))
```

shape  
layer

```
geom_...(data = data_frame,  
         mapping = aes(...),  
         non-variable adjustments)
```

shape  
layer

```
geom_...(data = data_frame,  
         mapping = aes(...),  
         non-variable adjustments)
```

+

+

Advanced topics:  
Mapping examples

# Mapping resources

- [tigris](#) for downloading TIGER/Line shapefiles
- [sf](#) (simple features) for spatial tables
  - [Spatial Data Science book](#)
  - [Spatial Data Science in the tidyverse slides](#)
  - [Spatial Data Science in the tidyverse video](#)

# Other helper packages

- [gganonymize](#) to randomize text in ggplot2 figures
- [visdat](#) to visualize variable classes and missing data
- [ggthemes](#) for additional themes and scales, especially ones that match software defaults (e.g., Tableau)
- [esquisse](#) for building ggplot2 charts interactively
- [colorblindr](#) for simulating color vision deficiency
- [ggpubr](#) for publication-ready plots



# ggplot2 Resources

- General ggplot2 information  
<http://ggplot2.tidyverse.org/>
- R Graphics Cookbook (recipes for plots)  
<http://www.cookbook-r.com/Graphs/index.html>
- R for Data Science (online book that includes ggplot2)  
<http://r4ds.had.co.nz/>
- ggplot2: Elegant Graphs for Data Analysis (book by Hadley Wickham)  
<http://ggplot2.org/book/>
- ggplot2 cheatsheet (also in RStudio)  
<http://bit.ly/ggplot2-cheatsheet>
- [Data Carpentry lesson on ggplot2](#)
- [Data Visualization: A Practical Introduction](#), by Kieran Healy
- [RStudio “Visualize Data” Primer](#)

# Thanks for your feedback!

[angela.zoss@duke.edu](mailto:angela.zoss@duke.edu)