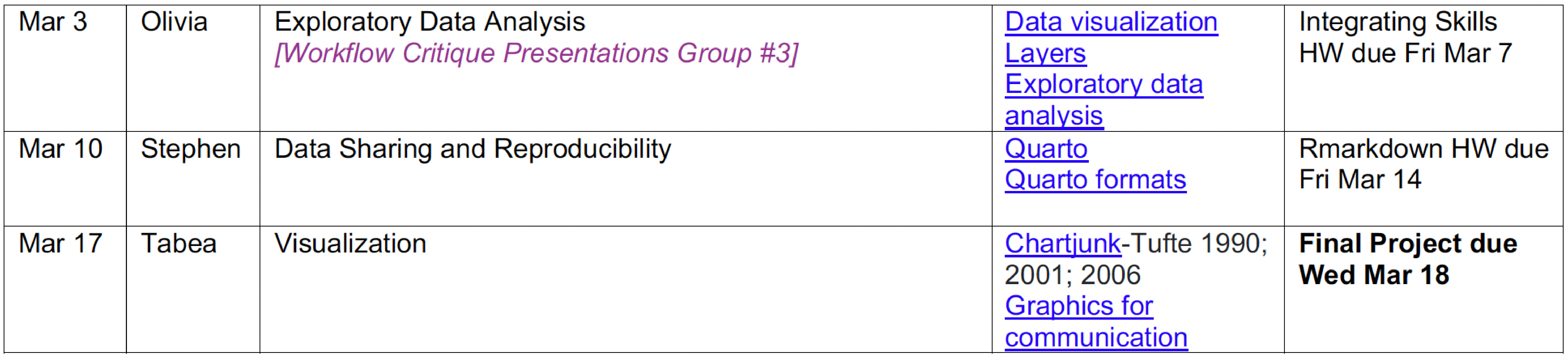
PSYC 259: Principles of Data Science

Week 6: Exploratory Data Analysis

Announcements

1. Remainder of the quarter



Mar 3	Olivia	Exploratory Data Analysis *[Workflow Critique Presentations Group #3]*	Data visualization Layers Exploratory data analysis	Integrating Skills HW due Fri Mar 7
Mar 10	Stephen	Data Sharing and Reproducibility	Quarto Quarto formats	Rmarkdown HW due Fri Mar 14
Mar 17	Tabea	Visualization	Chartjunk-Tufte 1990; 2001; 2006 Graphics for communication	**Final Project due Wed Mar 18**

2. A note on course and instructor evaluations

Instructor & Course Evaluations

- Although course evaluations open today, please do not fill them out until next week
- Tuppett (Psych Dept Chair) will be emailing with further instructions on how to do so, given the circumstances of this course
- Stephen will give you time in class next week (3/10) to fill them out

Plan for Today

1. Lecture
   - Confirmatory + exploratory data analysis
   - Tools for data checking
   - ggplot2 basics
2. Tutorial (exploratory data analysis)
3. BREAK
4. Workflow presentations (group #3)

Exploratory Data Analysis

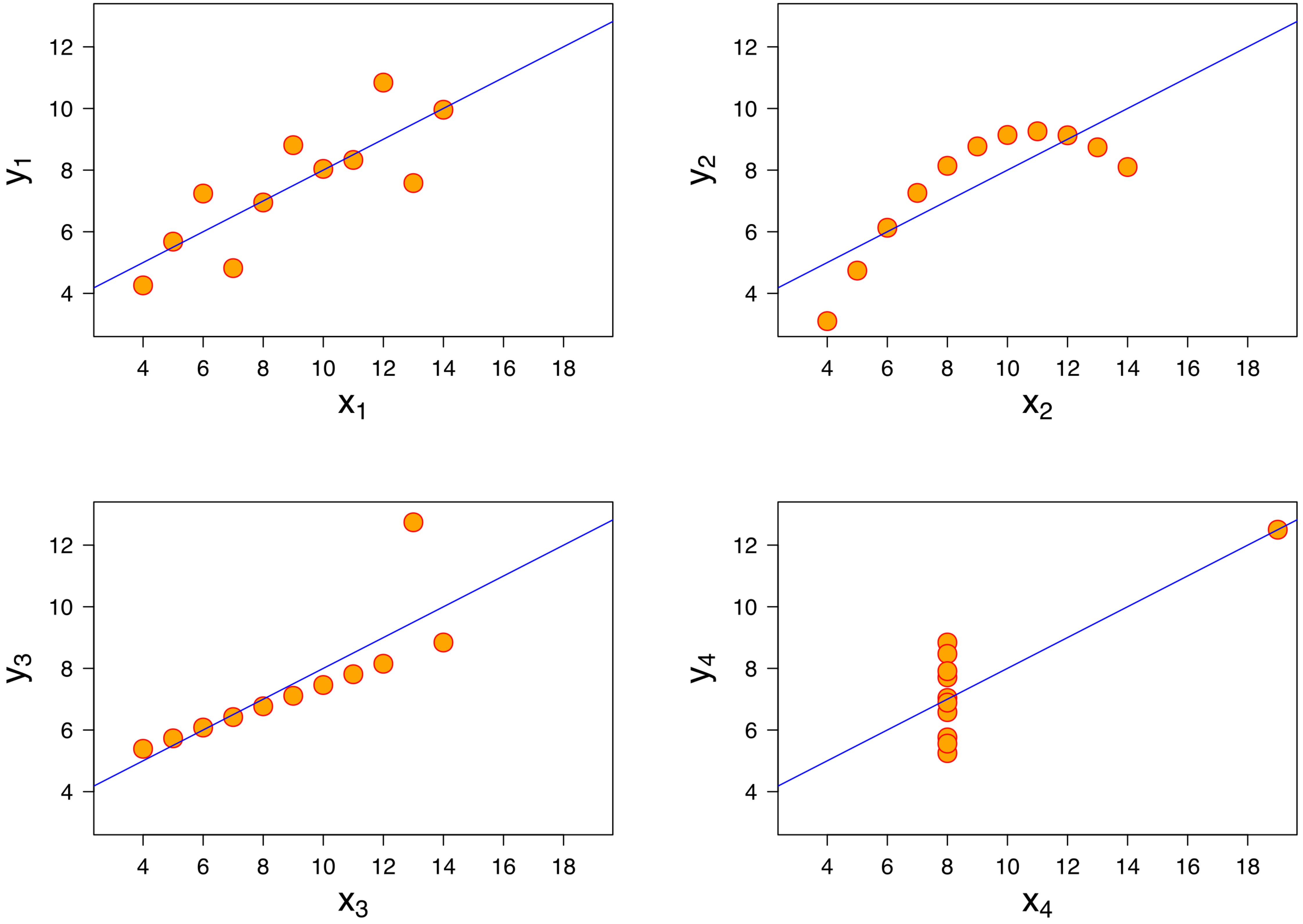
CDA vs. EDA

- Goals of Confirmatory Data Analysis
  - Hypothesis testing, probabilistic modeling, inference
- Goals of Exploratory Data Analysis (Tukey)
  - Understanding the patterns in the data
  - Generating hypotheses **(to be tested in other datasets)**
  - Checking your assumptions about data quality
  - “To find the unexpected, to avoid being fooled, and to develop rich descriptions” (Behrens & Yu, 2003)

Why do we need EDA? (Behrens & Yu, 2003)

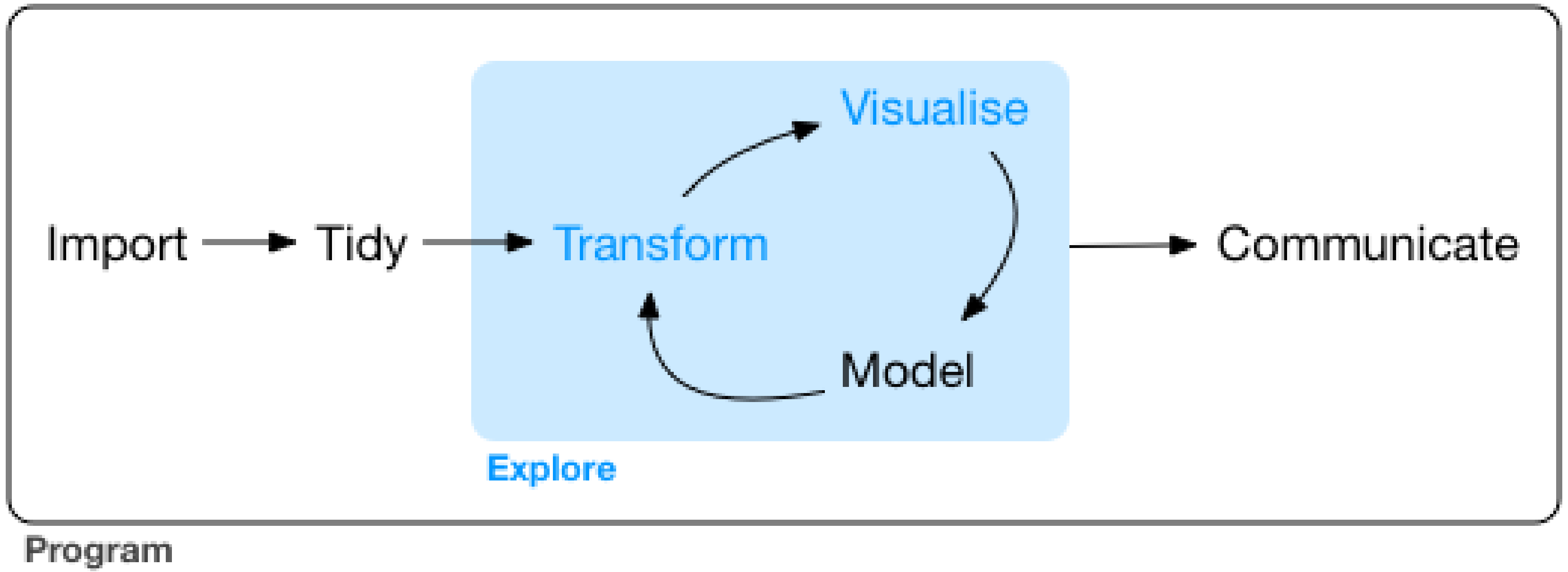
- Summarization = a loss of information
  - If you first look at summarized data (across trials, across participants, etc.), you may miss important patterns that exist at the raw data level
- Statistics lie, so you need graphics
  - Correlations without looking at the scatterplot
  - Means without examining outliers/distribution
  - Statistical tests without examining *n*

Anscombe’s quartet



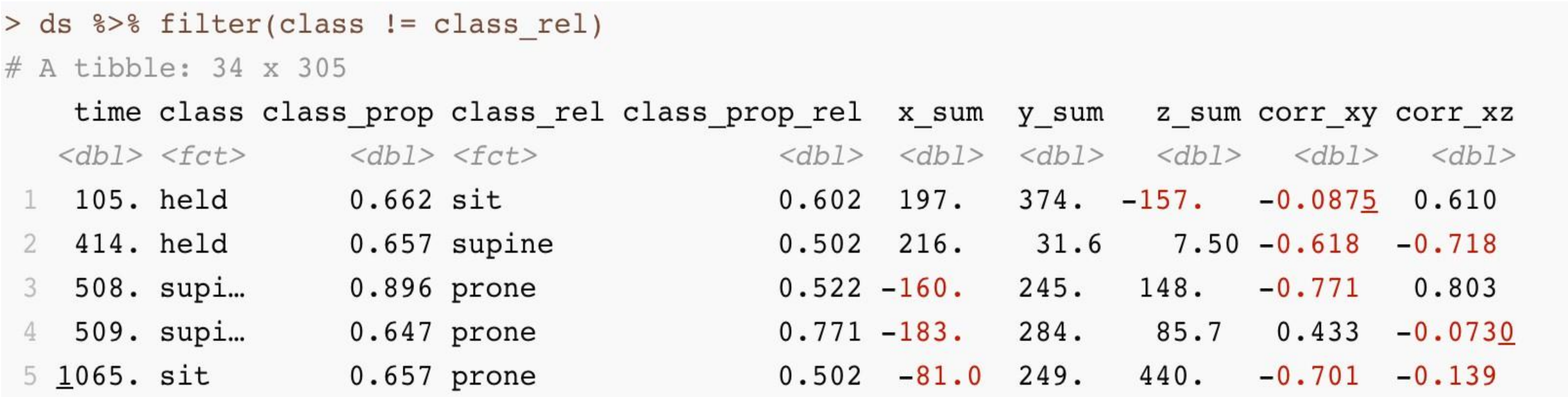
Where does wrangling stop and EDA begin?

- Data need to be minimally read in, appropriately labelled, and tidied to check and visualize
- EDA can reveal errors or redundancies that require new data wrangling steps



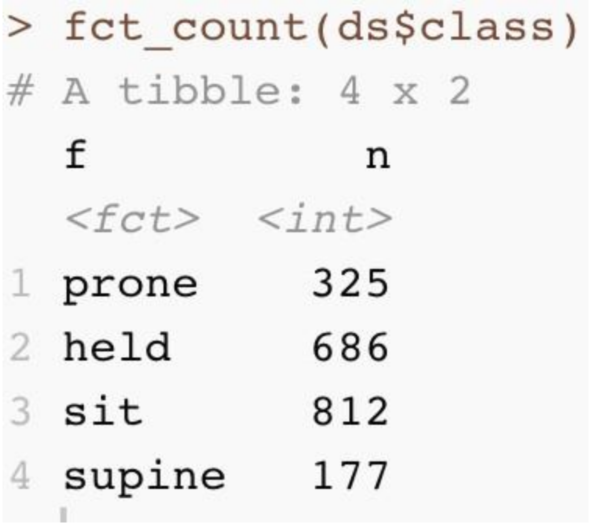
Tools for datachecking that we have already covered

- filter with logical statements



```
> ds %>% filter(class != class_rel)
# A tibble: 34 x 305
   time class class_prop class_rel class_prop_rel  x_sum  y_sum   z_sum corr_xy corr_xz
  <dbl> <fct>      <dbl> <fct>              <dbl>  <dbl>  <dbl>   <dbl>   <dbl>   <dbl>
1  105. held       0.662 sit                0.602  197.   374.  -157.   -0.0875   0.610
2  414. held       0.657 supine             0.502  216.    31.6    7.50 -0.618   -0.718
3  508. supi…      0.896 prone              0.522 -160.   245.   148.   -0.771    0.803
4  509. supi…      0.647 prone              0.771 -183.   284.    85.7   0.433   -0.0730
5 1065. sit        0.657 prone              0.502  -81.0  249.   440.   -0.701   -0.139
```

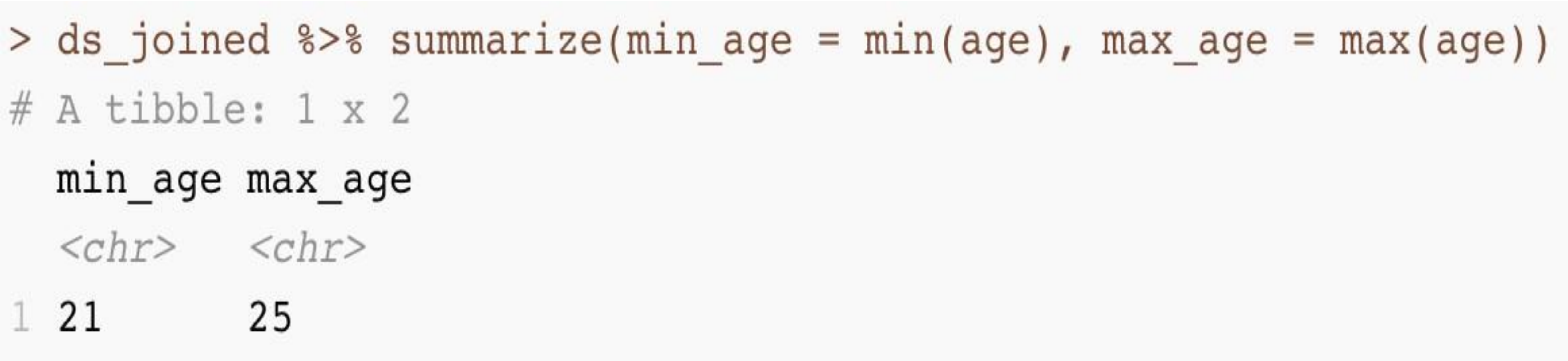
- fct_count to check factor frequencies



```
> fct_count(ds$class)
# A tibble: 4 x 2
  f          n
  <fct>  <int>
1 prone    325
2 held     686
3 sit      812
4 supine   177
```

Tools for datachecking that we have already covered

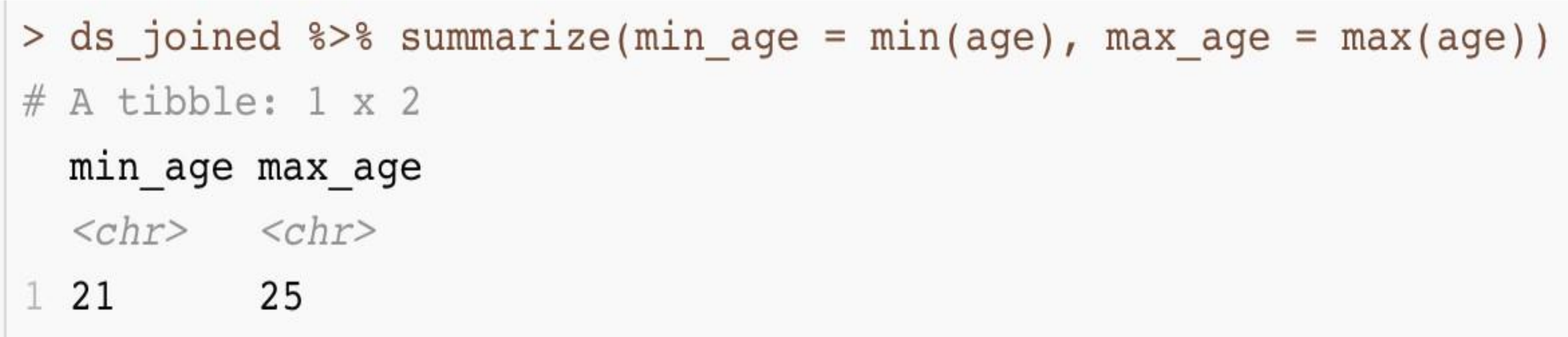
- summaries (with the right statistics/groupings)



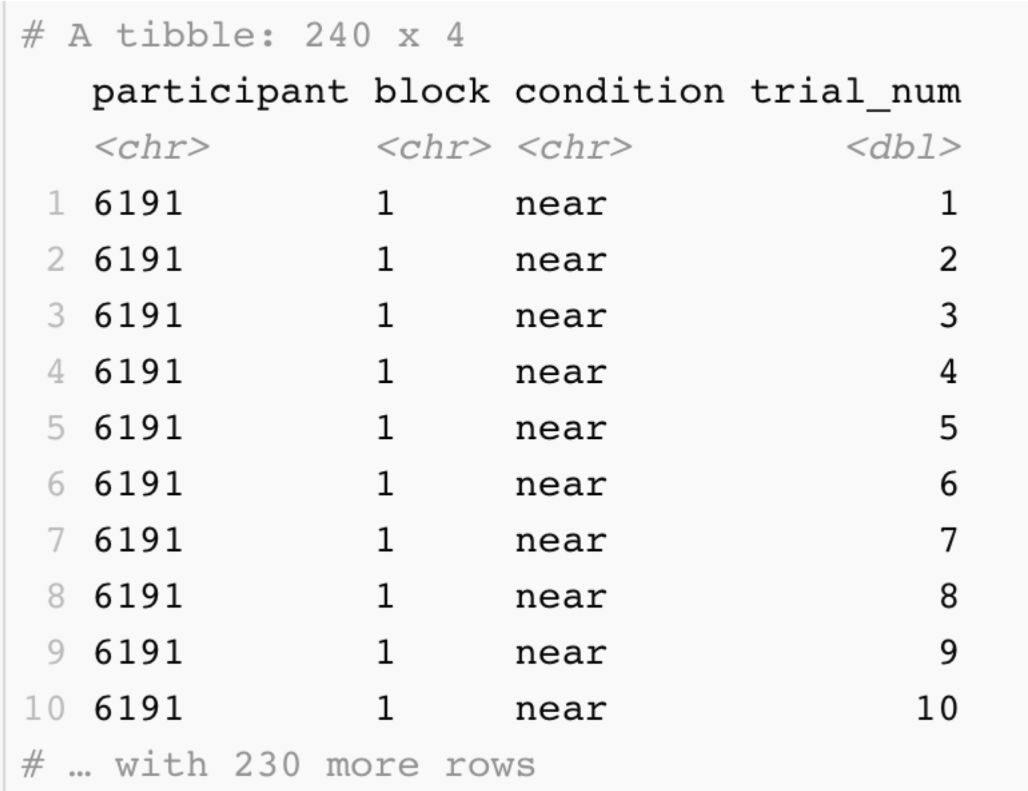
```
> ds_joined %>% summarize(min_age = min(age), max_age = max(age))
# A tibble: 1 x 2
  min_age max_age
  <chr>   <chr>
1 21      25
```

Tools for datachecking that we have already covered

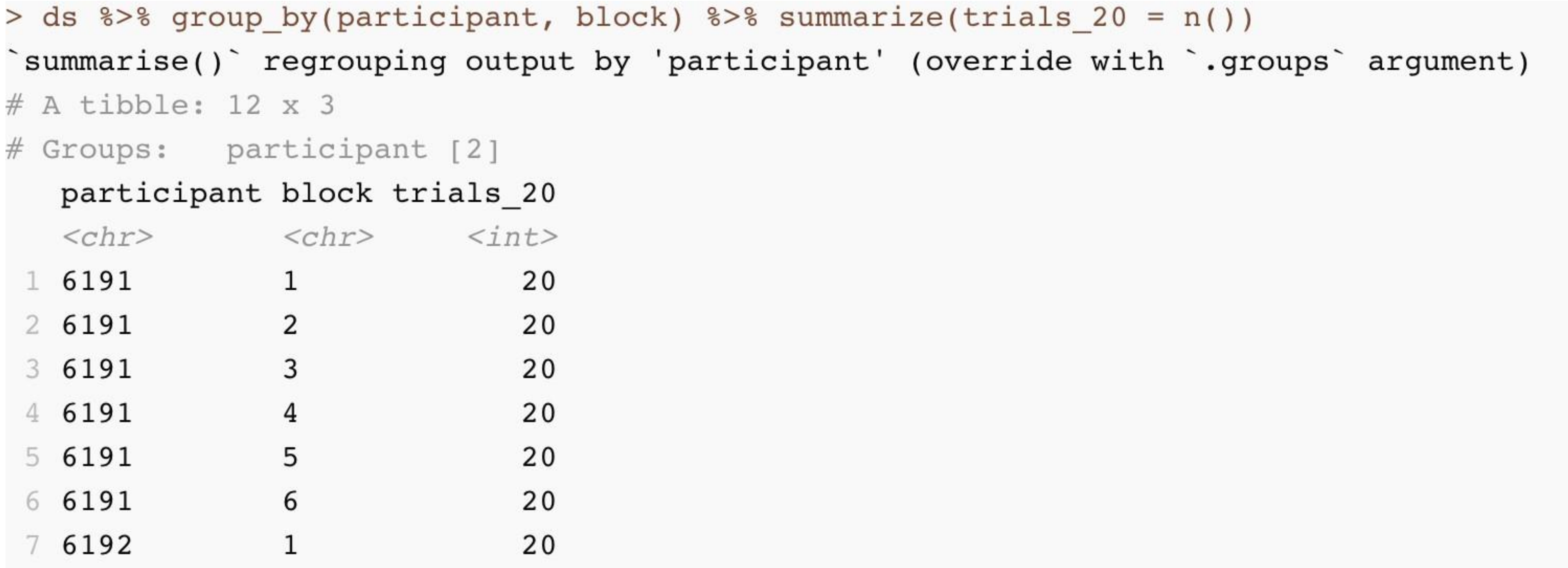
- summaries (with the right statistics/groupings)



```
> ds_joined %>% summarize(min_age = min(age), max_age = max(age))
# A tibble: 1 x 2
  min_age max_age
  <chr>   <chr>
1 21      25
```



```
# A tibble: 240 x 4
   participant block condition trial_num
   <chr>       <chr> <chr>         <dbl>
 1 6191        1     near              1
 2 6191        1     near              2
 3 6191        1     near              3
 4 6191        1     near              4
 5 6191        1     near              5
 6 6191        1     near              6
 7 6191        1     near              7
 8 6191        1     near              8
 9 6191        1     near              9
10 6191        1     near             10
# … with 230 more rows
```



```
> ds %>% group_by(participant, block) %>% summarize(trials_20 = n())
`summarise()` regrouping output by 'participant' (override with `.groups` argument)
# A tibble: 12 x 3
# Groups:   participant [2]
  participant block trials_20
  <chr>       <chr>     <int>
1 6191        1            20
2 6191        2            20
3 6191        3            20
4 6191        4            20
5 6191        5            20
6 6191        6            20
7 6192        1            20
```

Tools for data checking that we have already covered

- Automation
  - EDA means taking a detailed approach to look at data on different levels (participant/condition/wave/etc.)
  - Running multiple filters/checks, plotting multiple figures, etc. can get overwhelming without automation

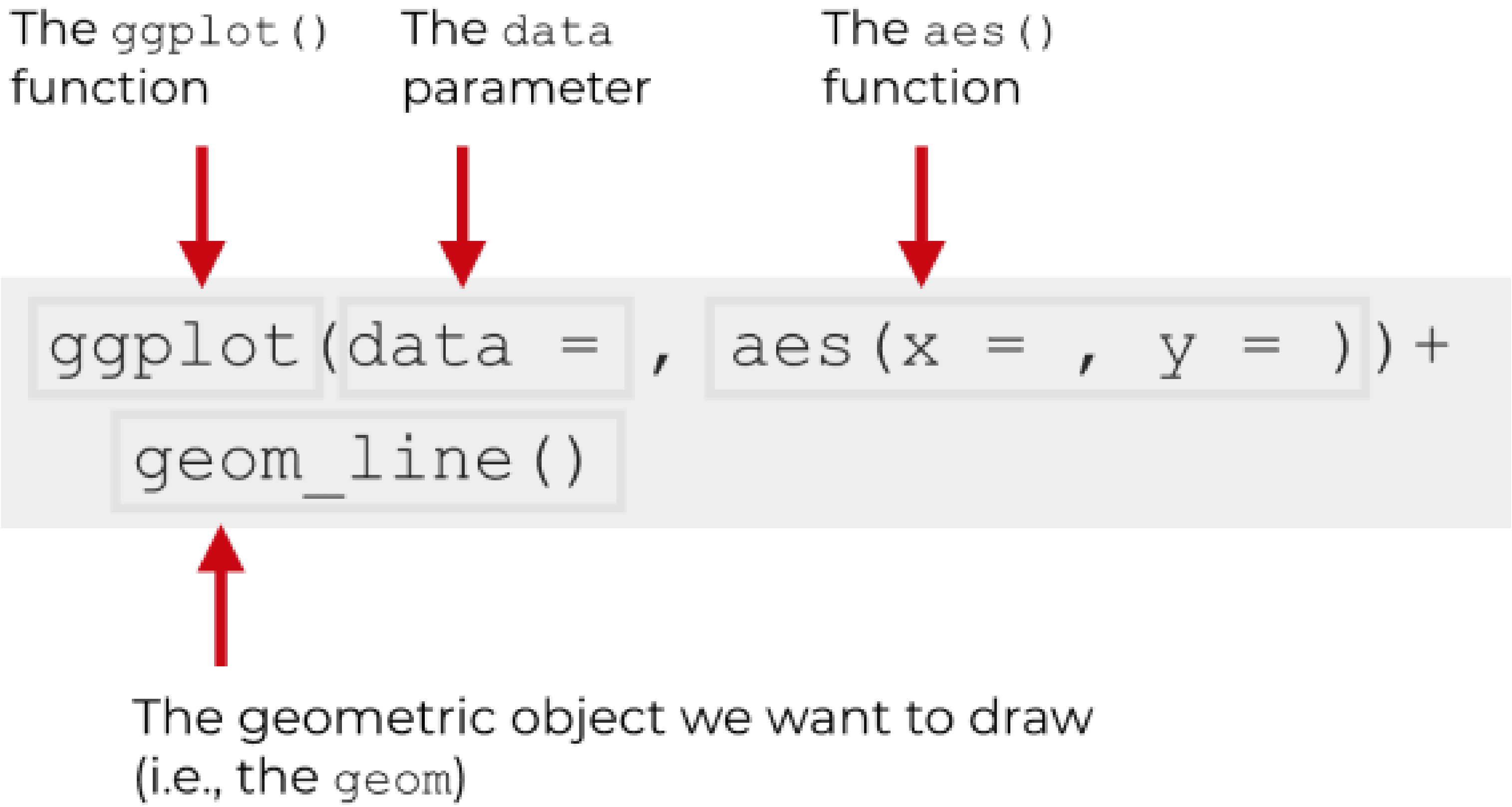
New tools for EDA - Visualizations

- DataExplorer package
  - Brute force, first glance methods
  - plot_histogram() of every continuous variable
  - plot_bar() counts of every categorical variable
- VisDat package
  - vis_miss() to plot missing values
  - vis_expect() to plot conditionals

New tools for EDA - Visualizations

- ggplot2 package (part of tidyverse)
  - Create any type of graph
  - Today we'll talk about making quicker plots for eda using geom_histogram, geom_point, geom_boxplot, and a few others
  - Week 8, Tabea will talk about making publication-ready plots to communicate effects

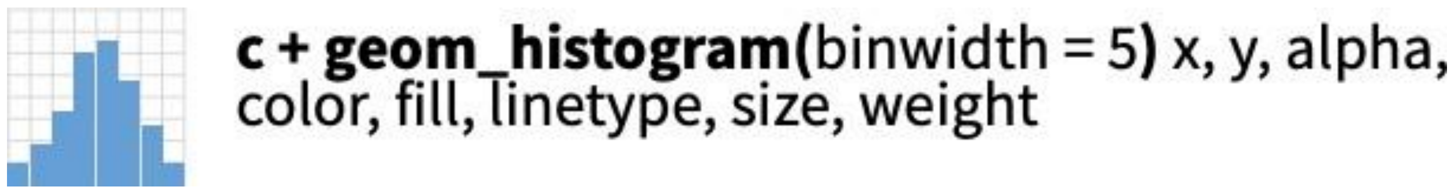
Anatomy of a ggplot call



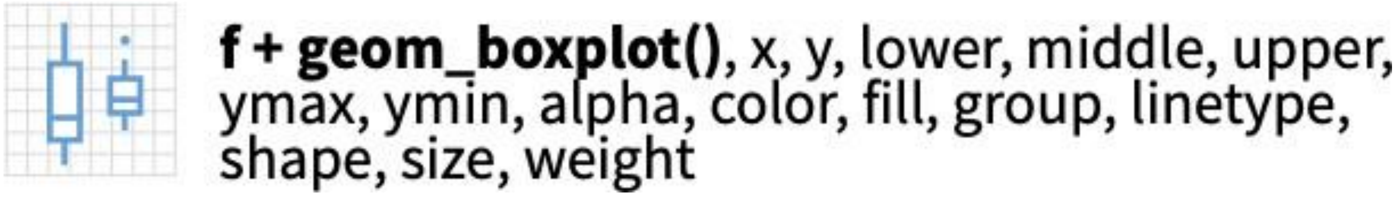
```
ggplot(data = , aes(x = , y = ))+
  geom_line()
```

- define the dataset we are using (long format)
- define the mapping of variables to *aes*thestics
- Add (+) geoms, graphical elements like histograms, lines, points, bars, boxplots, and many others
- Optional arguments to change the overall look

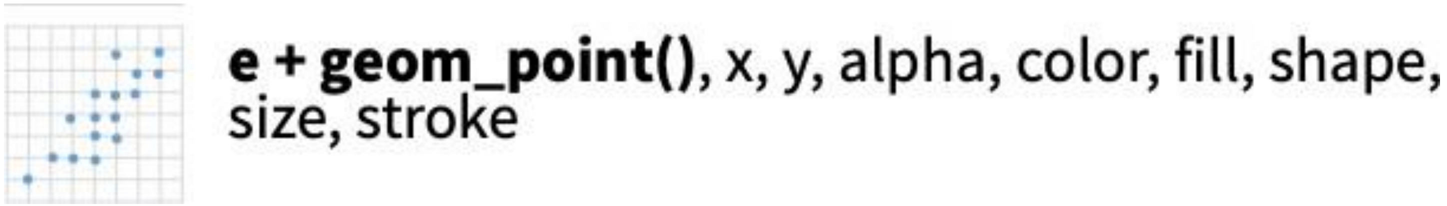
Each type of geom has different aesthetics that can be mapped



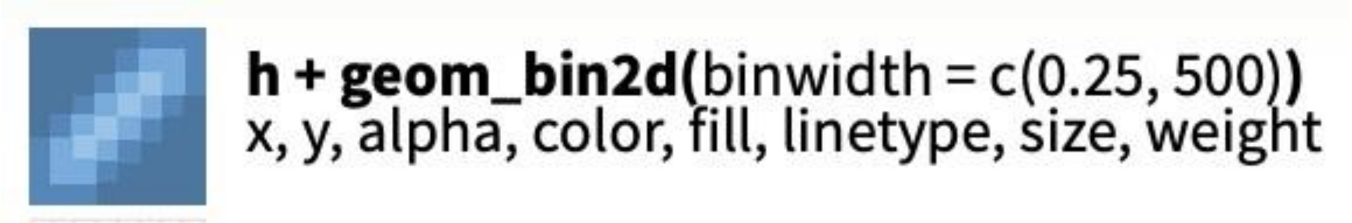
c + geom_histogram(binwidth = 5**)** x, y, alpha, color, fill, linetype, size, weight



f + geom_boxplot(), x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight



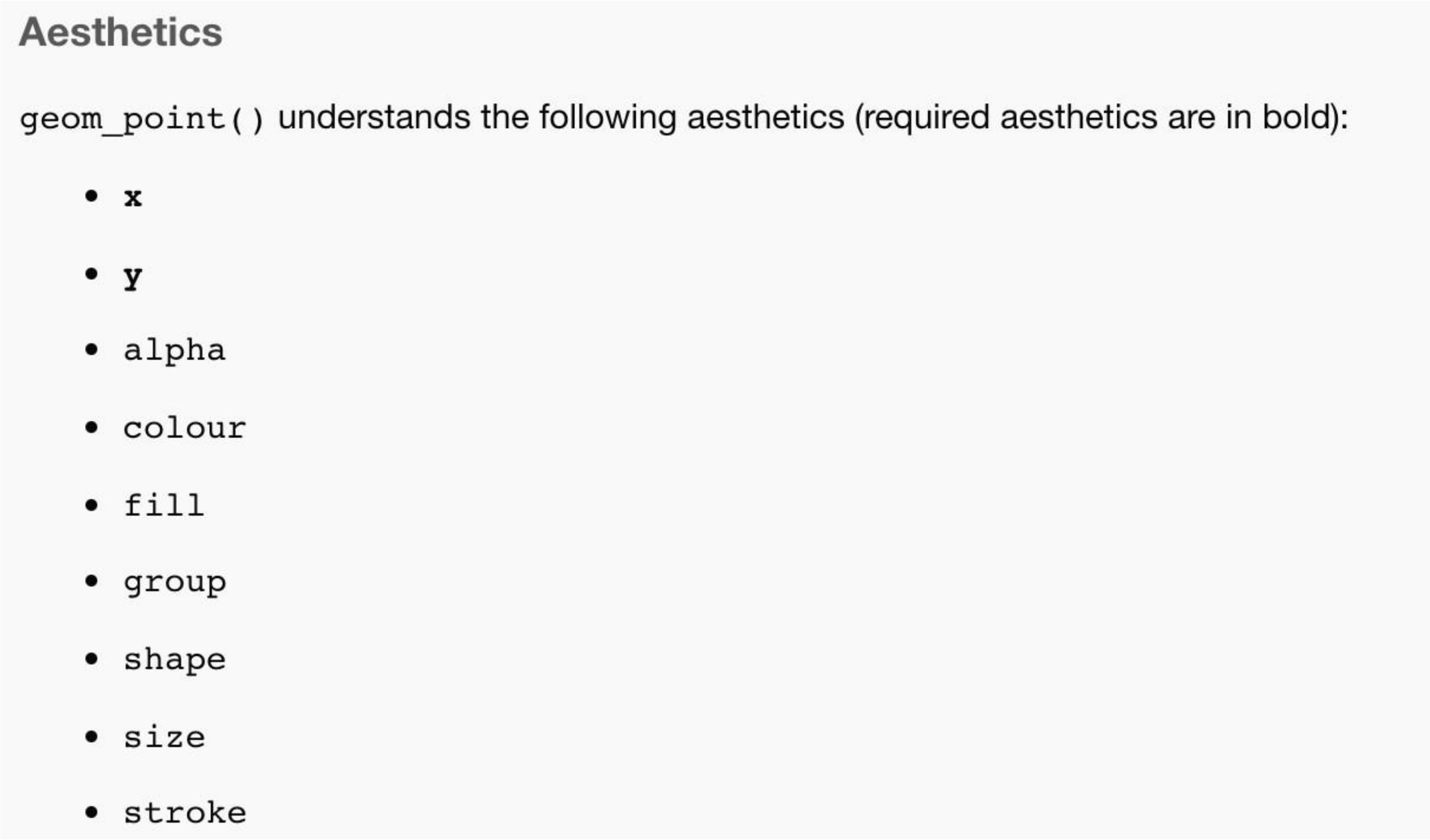
e + geom_point(), x, y, alpha, color, fill, shape, size, stroke



h + geom_bin2d(binwidth = c(0.25, 500)**)** x, y, alpha, color, fill, linetype, size, weight

What aes values are required for each geom?

- Check the help page to see required mappings in bold



Aesthetics

`geom_point()` understands the following aesthetics (required aesthetics are in bold):

- **x**
- **y**
- alpha
- colour
- fill
- group
- shape
- size
- stroke

Adding elements to graphs

- ggplot() + geom_X() +….
- Add (+) other modifications to the plot
  - xlim(lower_bound, upper_bound) or ylim
  - hline(yintercept = X) or vline
  - xlab(“x label”)
  - titles, custom scales, other geoms
- Make sure that plus is on the previous line, lines that start with + will throw an error

Tutorial: Exploratory data analysis