Data Visualization

Sep. 23, 2021

Housekeeping

Asking for Help

Before emailing for help, try the following:

- Restarting R and rerunning your code.
- Read the error message.
- Search the internet.
- Read the assigned readings.
- Ask classmate(s) for help.

How to Ask for Help

If you have done your best to solve the problem and still need help, please email us. A good email will include:

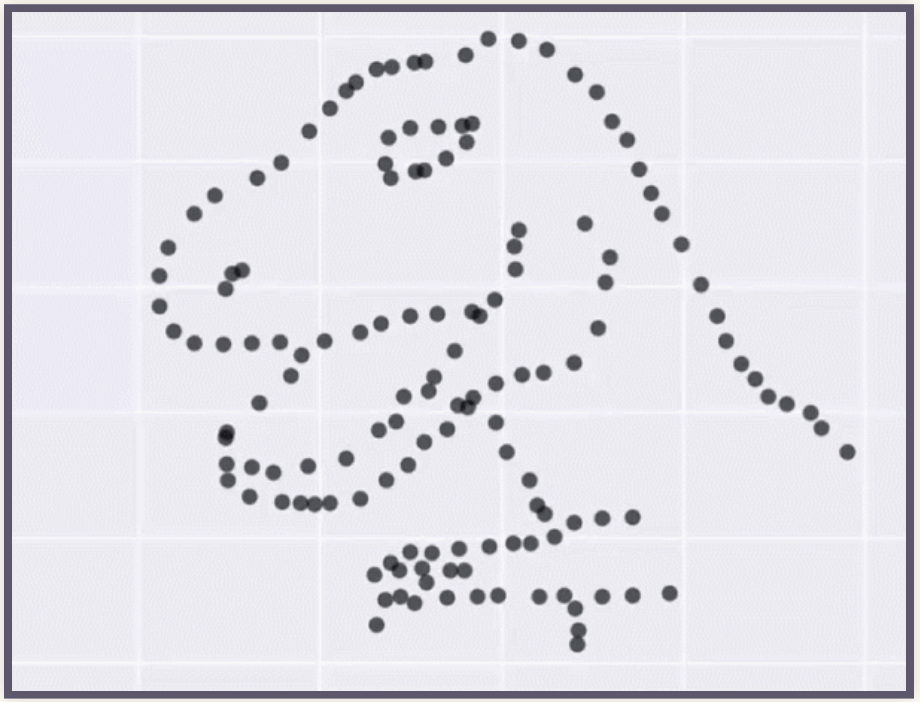
- Concise, detailed description of the problem.
- Attached script (with comments!).
- Email sent to both Rebecca and Zak.

Please do all you can to make it as easy as possible for us to help you. It takes a long time, and there are only two of us.

Today's Lab: Data Visualization

The Datasaurus Dozen

Different Datasets



Same Statistics!

- $M_x = 54.26$
- $M_y = 47.83$
- $SD_x = 16.76$
- $SD_y = 26.93$
- $r = -.06$

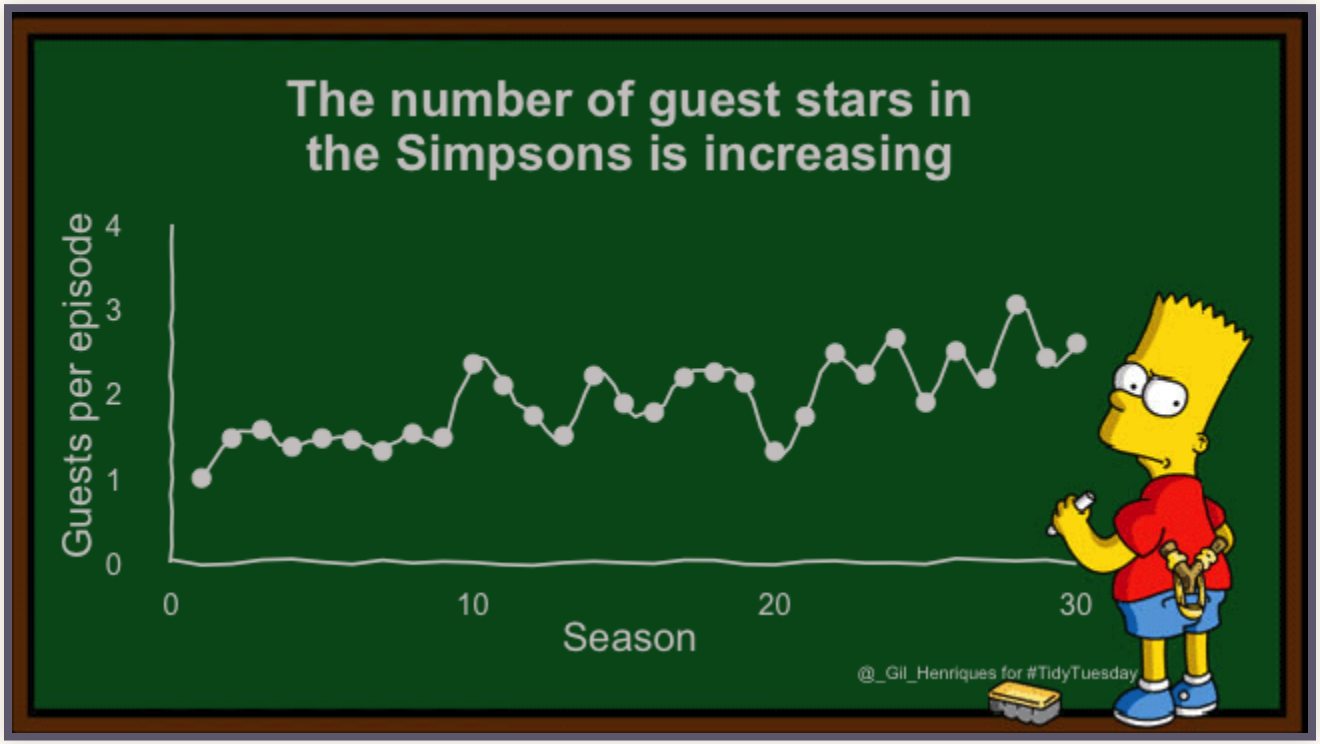
Learning Outcomes

After completing this lab, you should be able to:

1. Install and load R packages.
2. Understand why data visualization is a useful tool for science communication.
3. Know and apply the APA guidelines for figures in APA manuscripts.
4. Understand the basic "grammar of graphics" used by `ggplot2`.
5. Apply the basic "grammar of graphics" to visualize data in R.

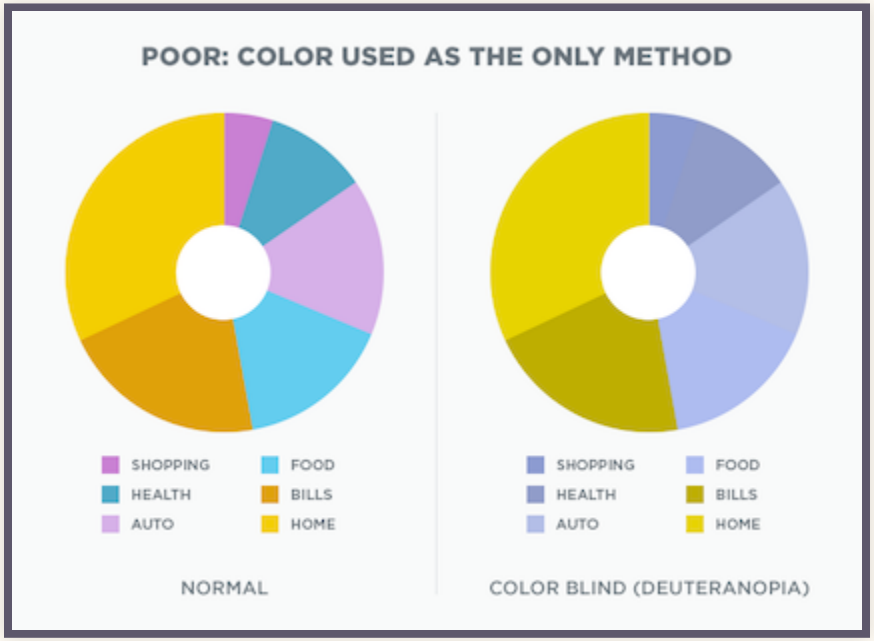
APA Guidelines for Figures

What not to do...



Guidelines

- Simple, legible font
- Axes labeled with units of measurement
- All features labeled and explained
- Figure notes appear below
- Accessible use of colour



APA Figure Example

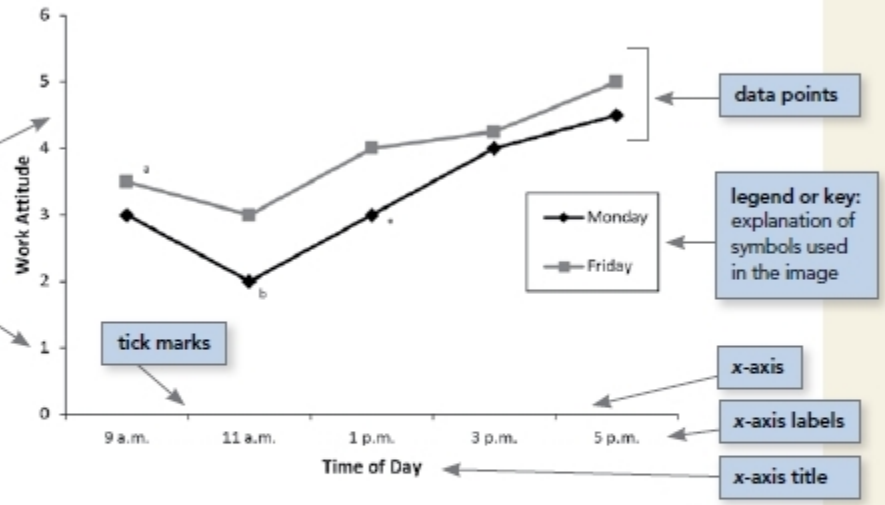
figure number

Figure 1

figure title

Changes in Work Attitude as a Function of Day and Time

image: graph, chart, drawing, map, plot, or photograph itself



data points

y-axis

y-axis title

y-axis labels

legend or key: explanation of symbols used in the image

tick marks

x-axis

x-axis labels

x-axis title

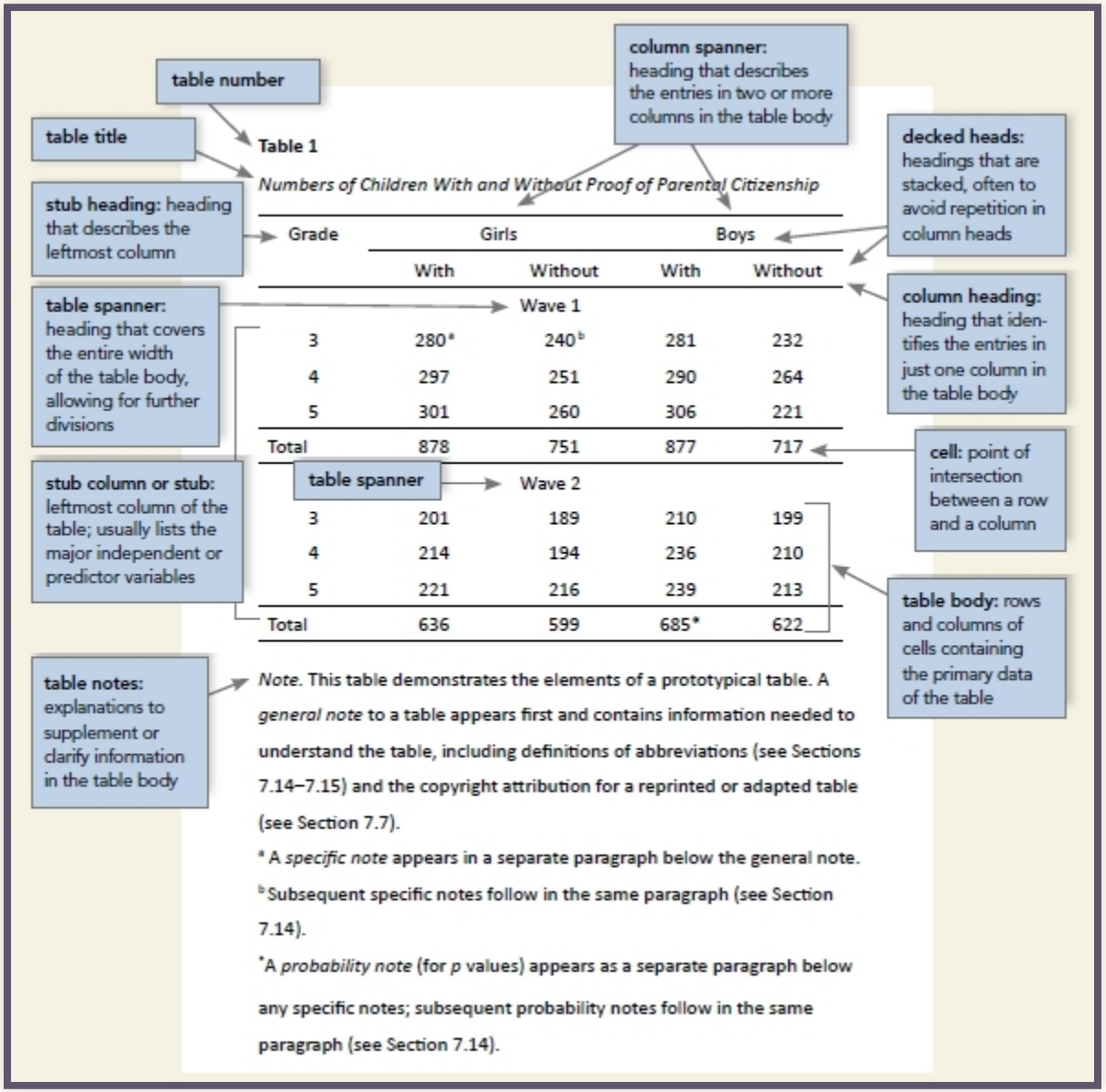
figure notes: explanations to supplement or clarify information in the image

Note. This figure demonstrates the elements of a prototypical figure. A *general note* to a figure appears first and contains information needed to understand the figure, including definitions of abbreviations (see Sections 7.15 and 7.28) and the copyright attribution for a reprinted or adapted figure (see Section 7.7).

[a]A *specific note* explains a particular element of the figure and appears in a separate paragraph below any general notes. [b] Subsequent specific notes follow in the same paragraph (see Section 7.28).

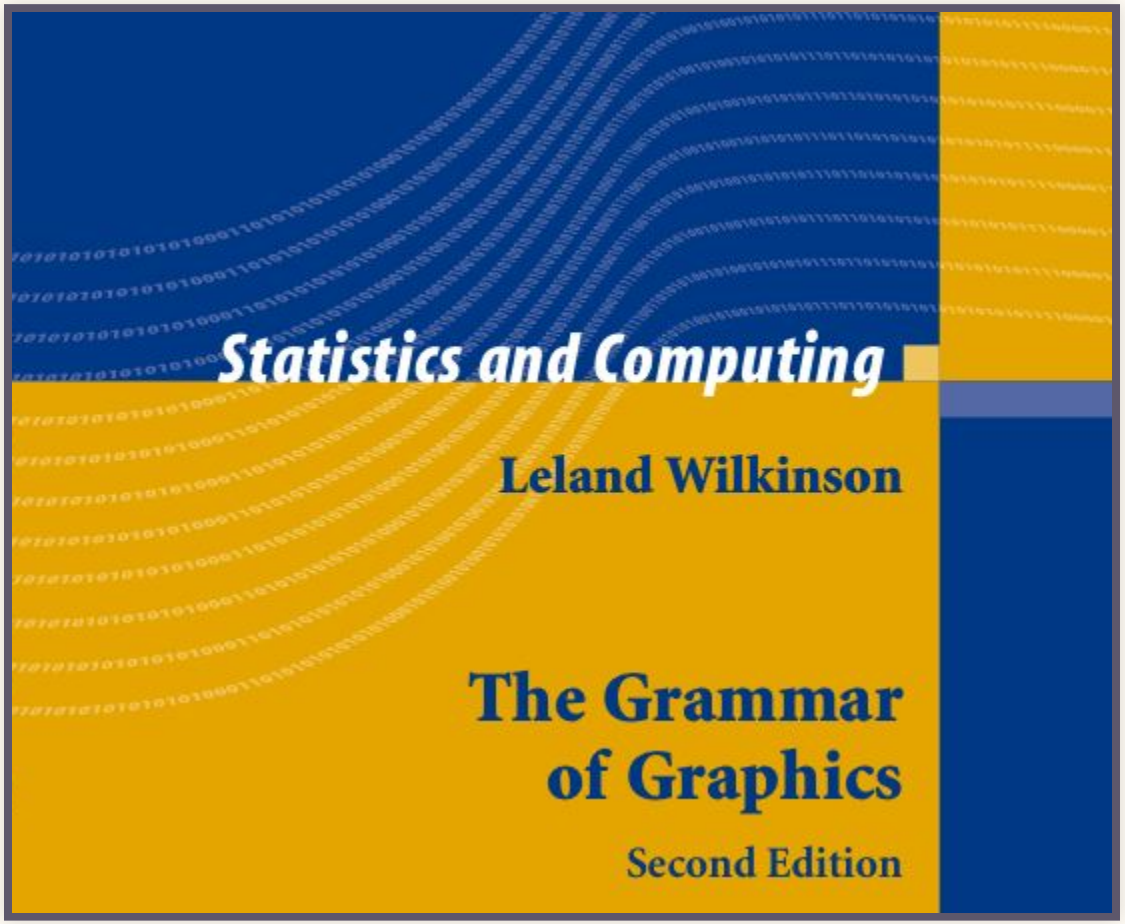
*A *probability note* (for *p* values) appears as a separate paragraph below any specific notes; subsequent probability notes follow in the same paragraph (see Section 7.28).

APA Table Example



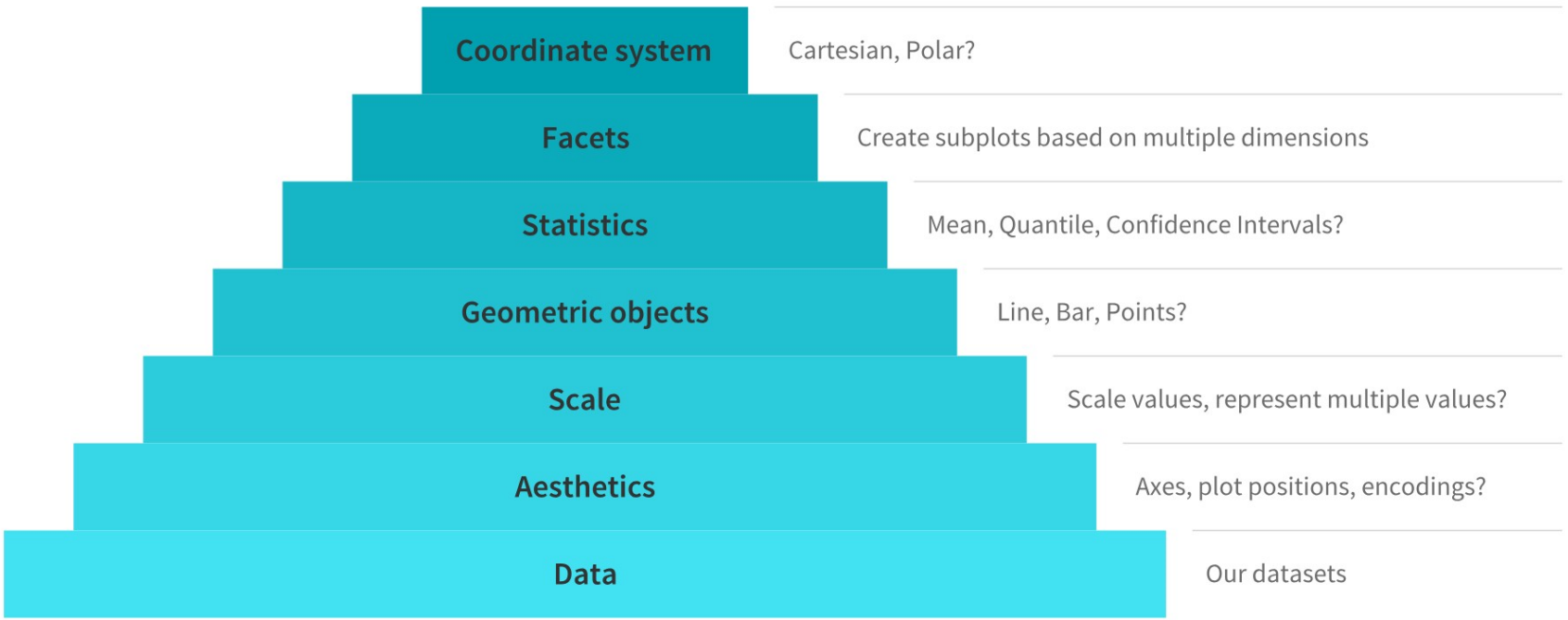
The Grammar of Graphics

The Grammar of Graphics



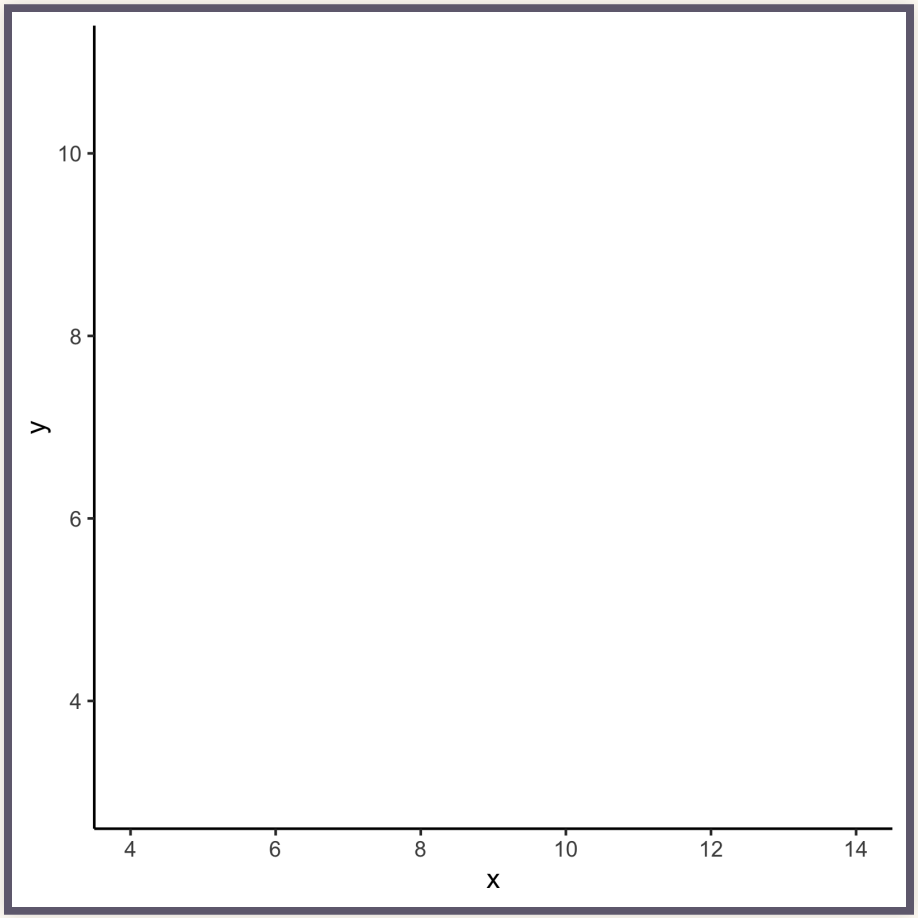
Layered Grammar of Graphics

Major Components of the Grammar of Graphics

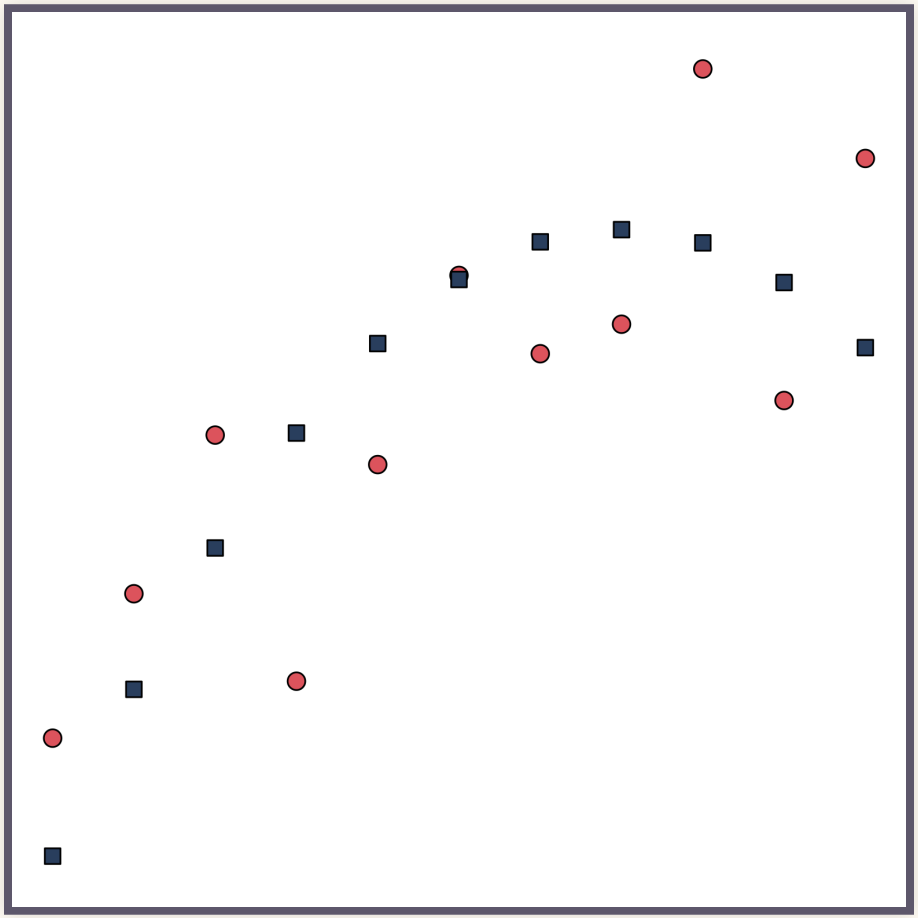


Example

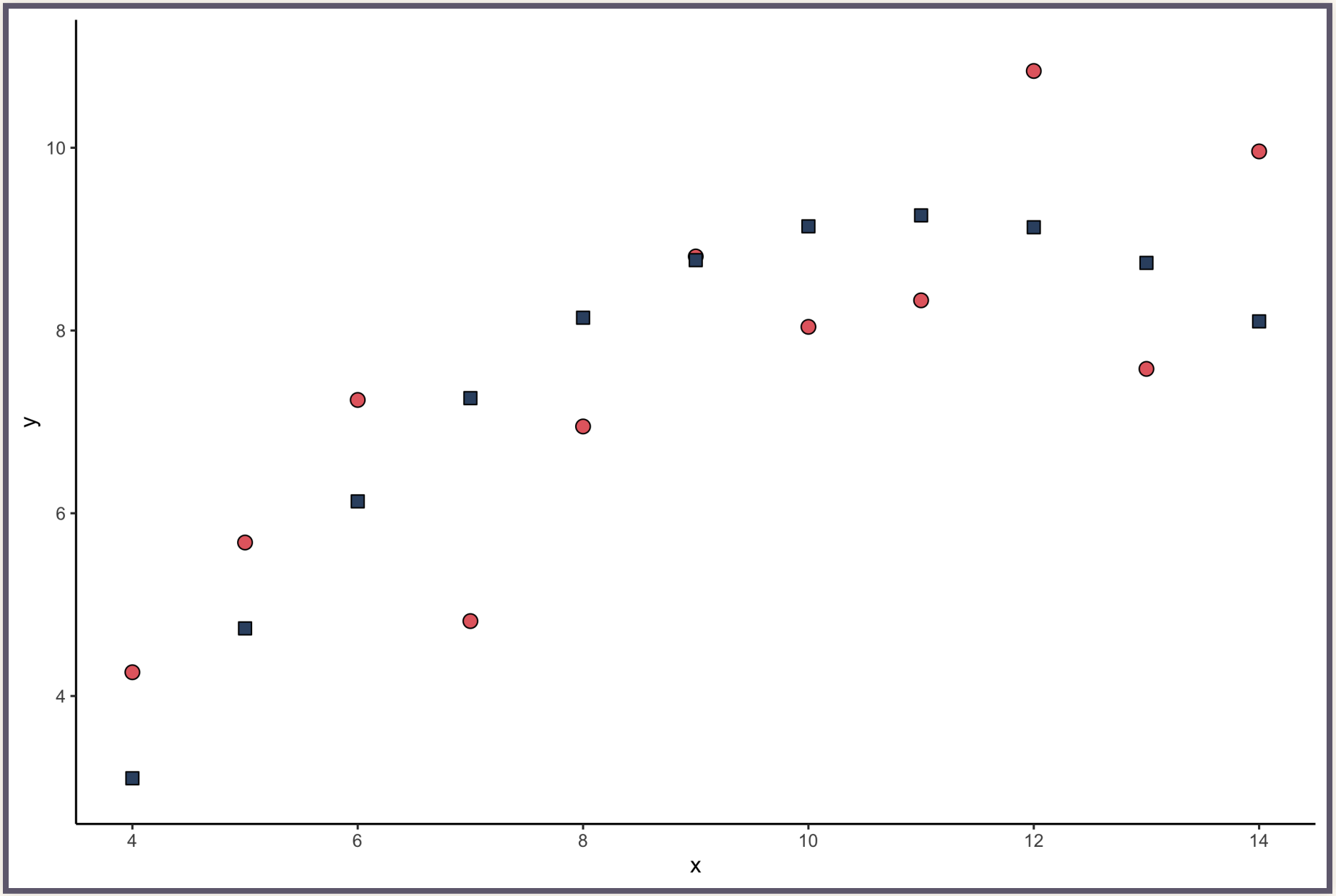
Scale and Coords



Geometric Objects



Example



R Packages

What is an R Package?

- Collection of R objects (mainly functions) that serve a unified purpose.
- Base packages are included with R.
- Add-on packages extends R's functionality.
  - Like downloadable content for a video game!
  - Or like purchasing a new Dungeons & Dragons adventure!

Base and Add-on Packages

Base R

- Installed with R.
- Only 14 of them (as of R 2.14.0).
- The basic functionality of R.
- By default, loaded in an R session.
- Built by the R Core Team.

Add-on

- Installed separately from R.
- 18224 just on the CRAN.
- Extend basic functionality of R.
- By default, unavailable in an R session.
- Built by R contributors.

Installing Packages: `install.packages()`

Use `install.packages()` to install a package from the CRAN.

- Only one required argument: `pkgs`.
- The name of the package in quotes.

tRy it! Install a Package from the CRAN

Use `install.packages()` to install `ggplot2` from the CRAN.

```
install.packages("ggplot2")
```

Accessing Installed Packages

- Running `install.packages()` will download the package and store it in a file on your computer.
- Like ordering a book to the library.
- Still needs to be checked out before you can use it.

Loading Packages: `library()`

Use `library()` to load packages.

- `library(package)`
- `package` can be a character string or not.

tRy it! Load `ggplot2`

Load the `ggplot2` package with `library()`

```
library(ggplot2)
```

R Packages: Best Practices

- Do **not** include calls to `install.packages()` in your scripts.
- Begin each script by loading all packages the script requires.
- Only load packages that the script requires.

R Packages: Review

- Base R packages are installed and loaded by default.
- Add-on packages must be installed and loaded separately.
- Install with `install.packages()`
  - Adds a package to your library.
  - Do this once.
  - Installed packages are available for checkout.
- Load with `library()`
  - `library()` checks out a package from your library.
  - Do this whenever a script requires the package.

The Data

tRy it! Data Import

Download “anscombe_long.csv” from the lab manual website and import it to R. Assign the resulting `data.frame` object the name `anscombe_long`.

```
anscombe_long <- read.csv("data/anscombe_long.csv")
```

tRy it! Convert to Factor

Convert to the column `dataset` to a factor with levels 1 = “I”, 2 = “II”, 3 = “III”, and 4 = “IV”.

```
anscombe_long$dataset <- factor(anscombe_long$dataset,
  levels = 1:4,
  labels = c("I", "II", "III", "IV")
)
```

The Data: Anscombe's Quartet

> *"Four x-y datasets which have the same traditional statistical properties (mean, variance, correlation, regression line, etc.), yet are quite different."*

The Data: Anscombe's Quartet

id	dataset	x	y
1	I	10	8.04
2	I	8	6.95
3	I	13	7.58
4	I	9	8.81
5	I	11	8.33
6	I	14	9.96
7	I	6	7.24
8	I	4	4.26
9	I	12	10.84
10	I	7	4.82
11	I	5	5.68
1	II	10	9.14
2	II	8	8.14
3	II	13	8.74

id	dataset	x	y
4	II	9	8.77
5	II	11	9.26
6	II	14	8.10
7	II	6	6.13
8	II	4	3.10
9	II	12	9.13
10	II	7	7.26
11	II	5	4.74
1	III	10	7.46
2	III	8	6.77
3	III	13	12.74
4	III	9	7.11
5	III	11	7.81
6	III	14	8.84
7	III	6	6.08
8	III	4	5.39
9	III	12	8.15
10	III	7	6.42

id	dataset	x	y
10	III	7	6.42
11	III	5	5.73
1	IV	8	6.58
2	IV	8	5.76
3	IV	8	7.71
4	IV	8	8.84
5	IV	8	8.47
6	IV	8	7.04
7	IV	8	5.25
8	IV	19	12.50
9	IV	8	5.56
10	IV	8	7.91
11	IV	8	6.89

Same Statistical Properties?

```
by(anscombe_long$x,
   INDICES = anscombe_long$dataset,
   FUN = mean
)
```

```
## anscombe_long$dataset: I
## [1] 9
## ------------------------------------------------------------
## anscombe_long$dataset: II
## [1] 9
## ------------------------------------------------------------
## anscombe_long$dataset: III
## [1] 9
## ------------------------------------------------------------
## anscombe_long$dataset: IV
## [1] 9
```

Same Statistical Properties?

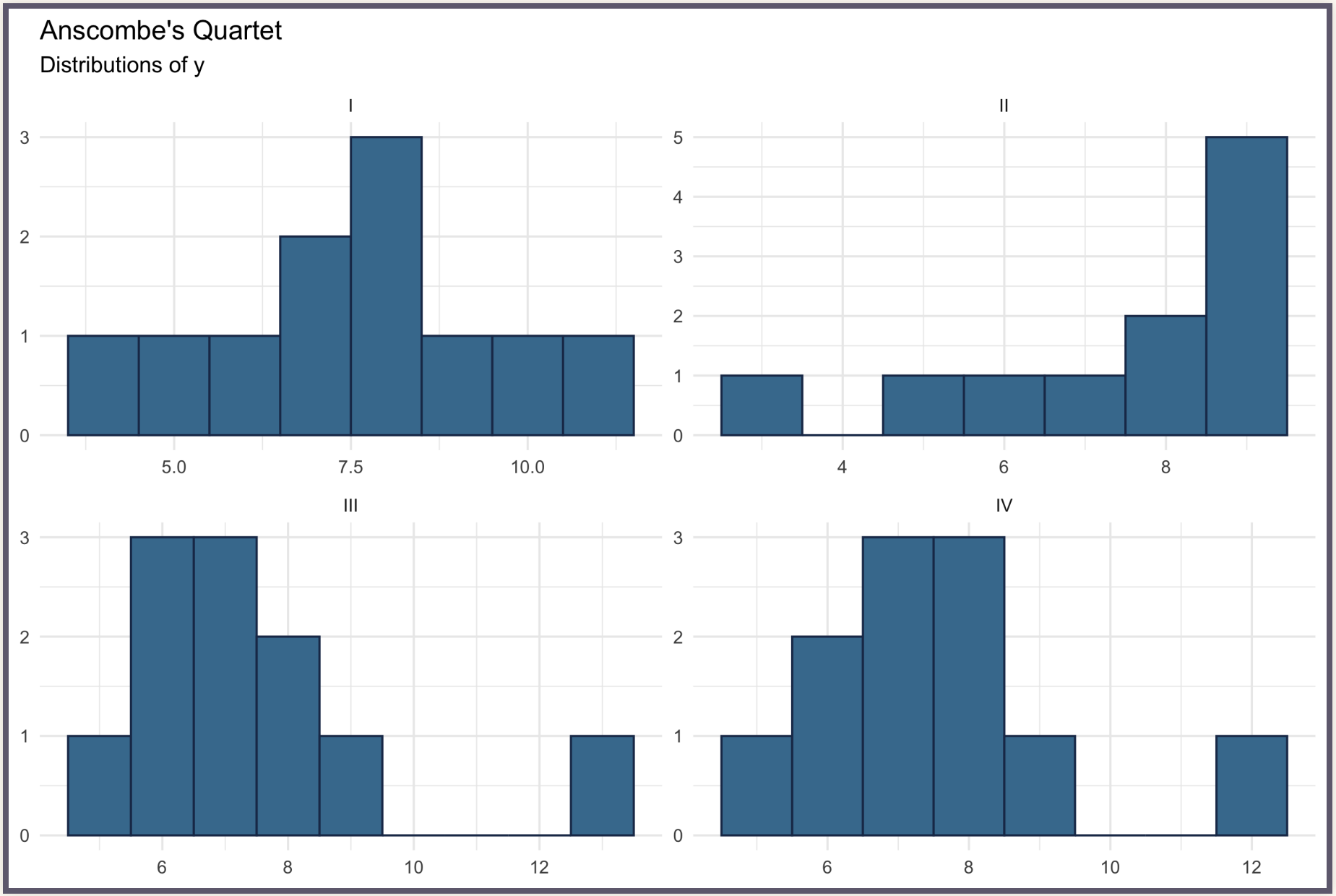
Table 1

Statistical Properties of x and y in Four Datasets

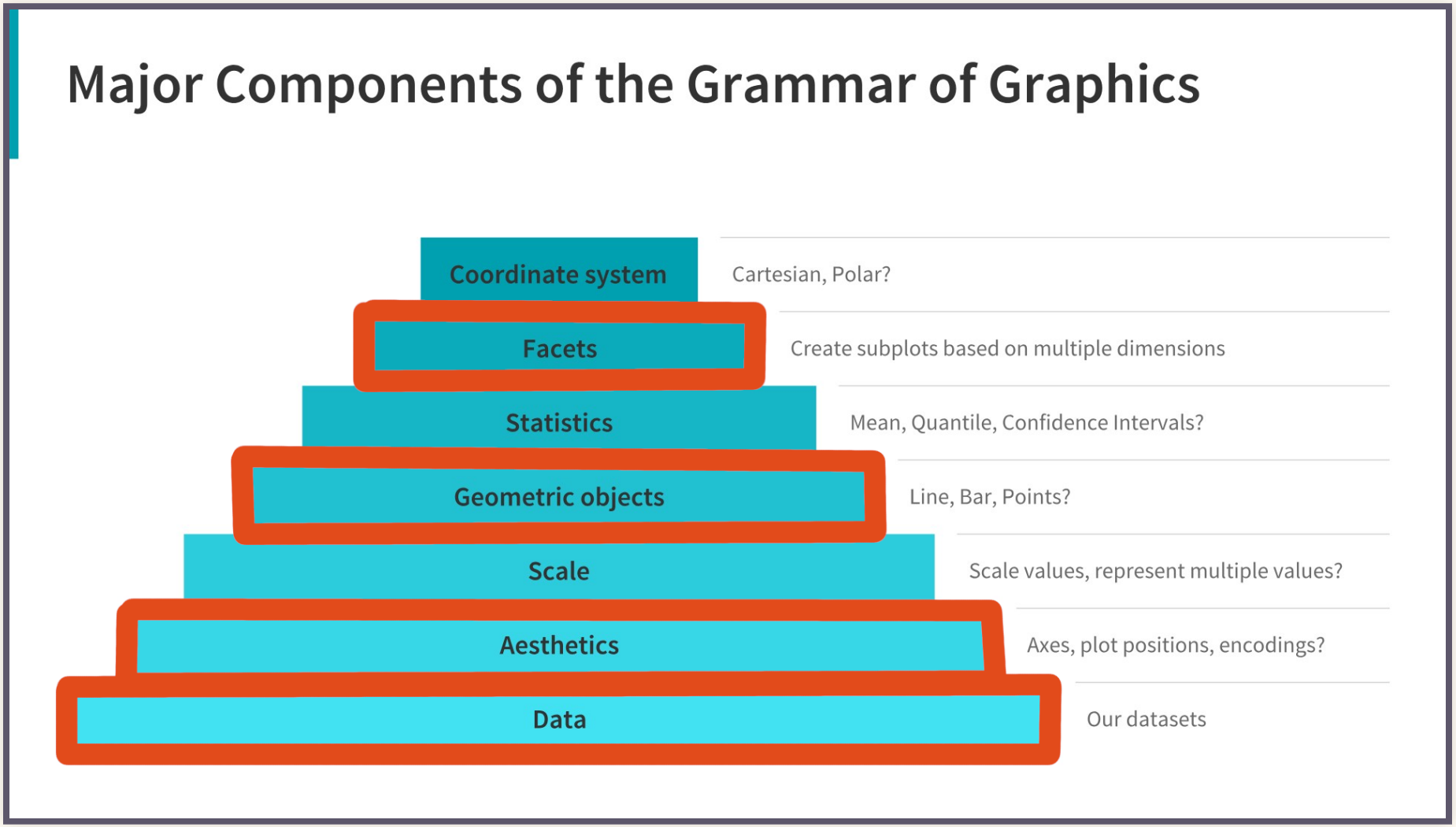
	Mx	SDx	My	SDy	cor(x, y)
I	9	3.32	7.5	2.03	0.82
II	9	3.32	7.5	2.03	0.82
III	9	3.32	7.5	2.03	0.82
IV	9	3.32	7.5	2.03	0.82

So… how are they different?

Plot 1

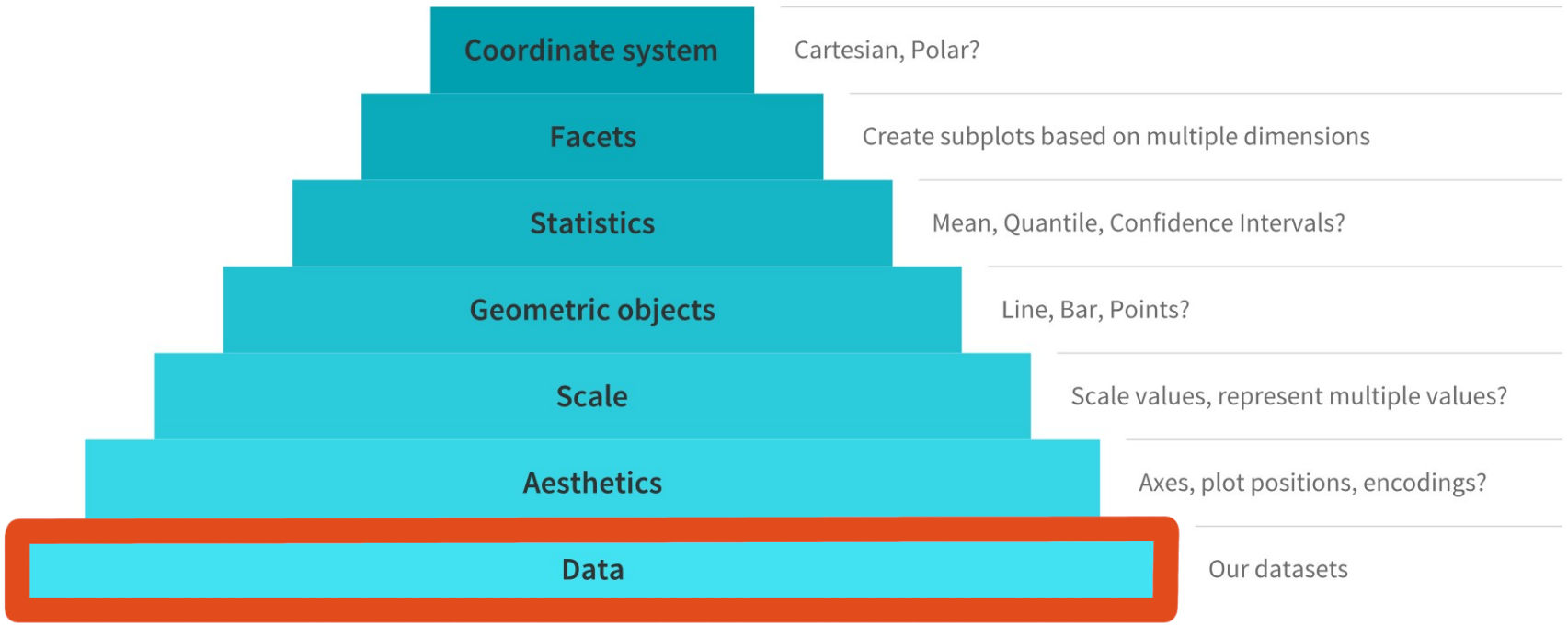

Anscombe's Quartet
Distributions of y
I
II
III
IV
0
1
2
3
4
5
5.0
7.5
10.0
6
8
10
12

Components Covered



Data

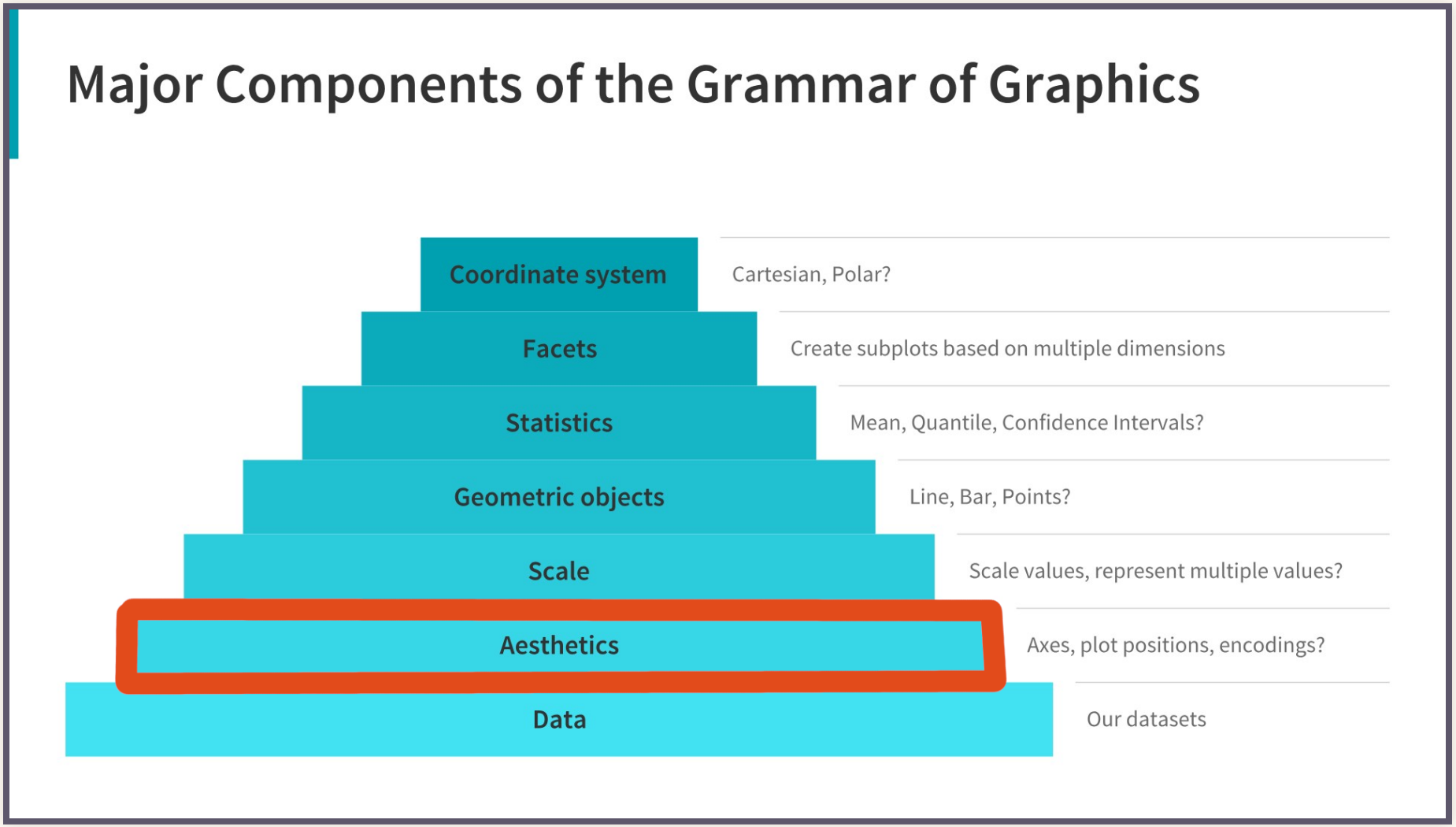
Major Components of the Grammar of Graphics



Plot 1: Data

```
ggplot(data = anscombe_long)
```

Aesthetics


Major Components of the Grammar of Graphics
Coordinate system
Cartesian, Polar?
Facets
Create subplots based on multiple dimensions
Statistics
Mean, Quantile, Confidence Intervals?
Geometric objects
Line, Bar, Points?
Scale
Scale values, represent multiple values?
Aesthetics
Axes, plot positions, encodings?
Data
Our datasets

What Are Aesthetics?

From the documentation for `ggplot2::aes()`:

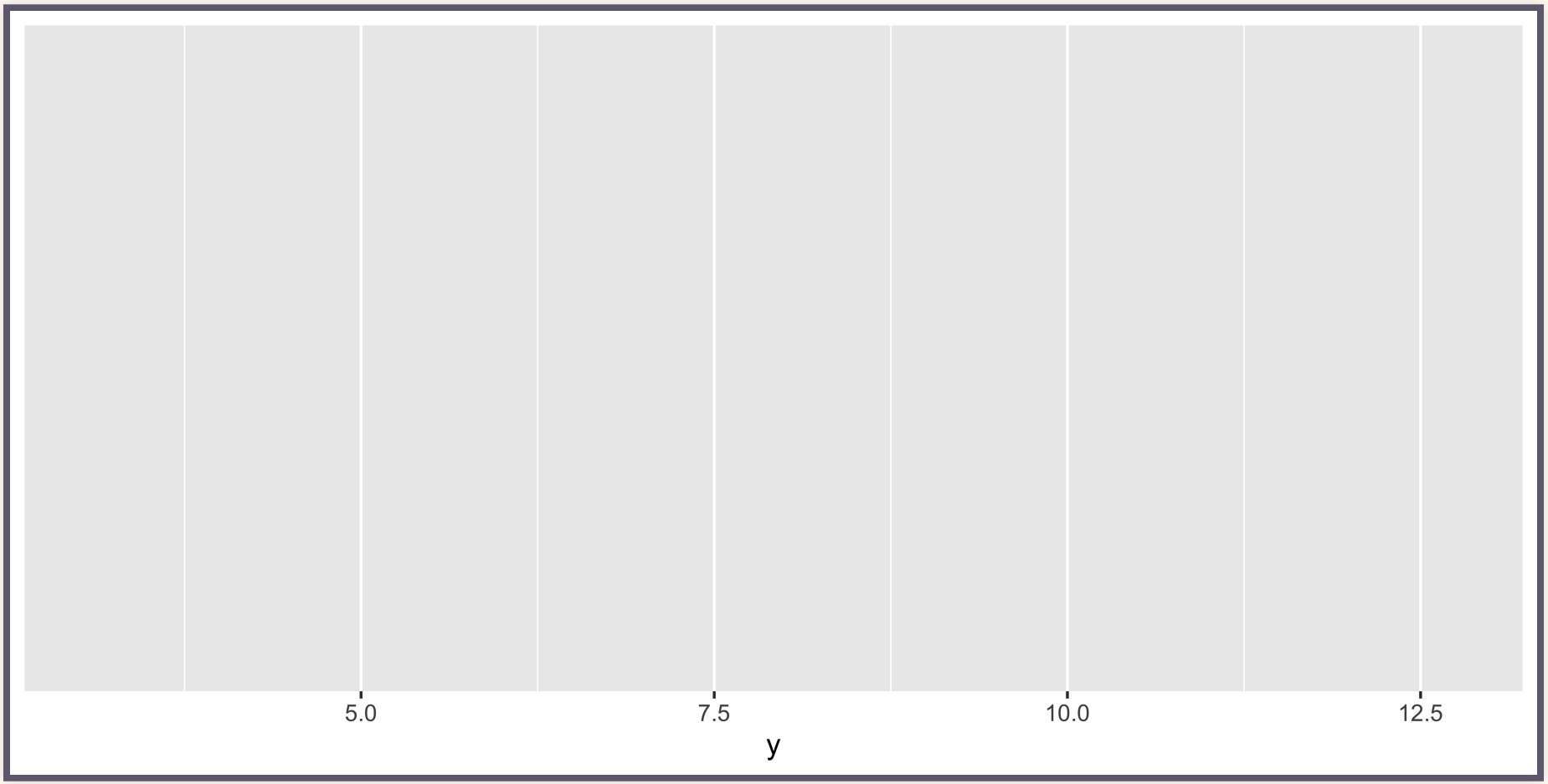
> *"Aesthetic mappings describe how variables in the data are mapped to visual properties (aesthetics) of geoms."*

Aesthetic Options

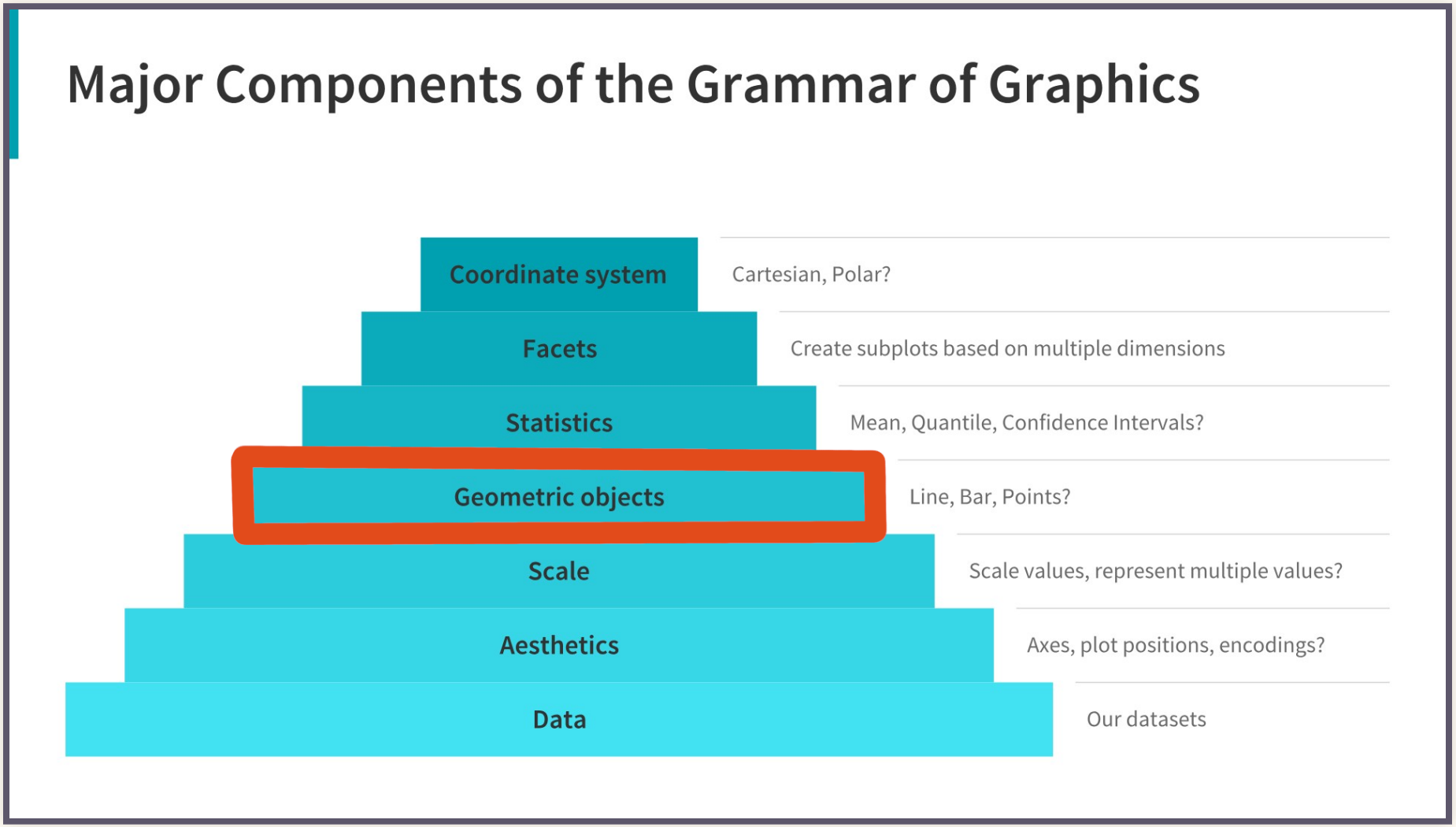
Argument	The Value is Mapped to:
x	Where the geom is placed along the x-axis.
y	Where the geom is placed along the y-axis.
colour	The line colour of the geom.
fill	The fill of the geom.
linetype	Different linetypes (e.g., solid, dashed, or dotted).
shape	Different shapes (e.g., square, circle, diamond).
size	The size of the geom.
alpha	The transparency of the geom.

Plot 1: Aesthetics

```
ggplot(data = anscombe_long, aes(x = y))
```



Geometric Objects



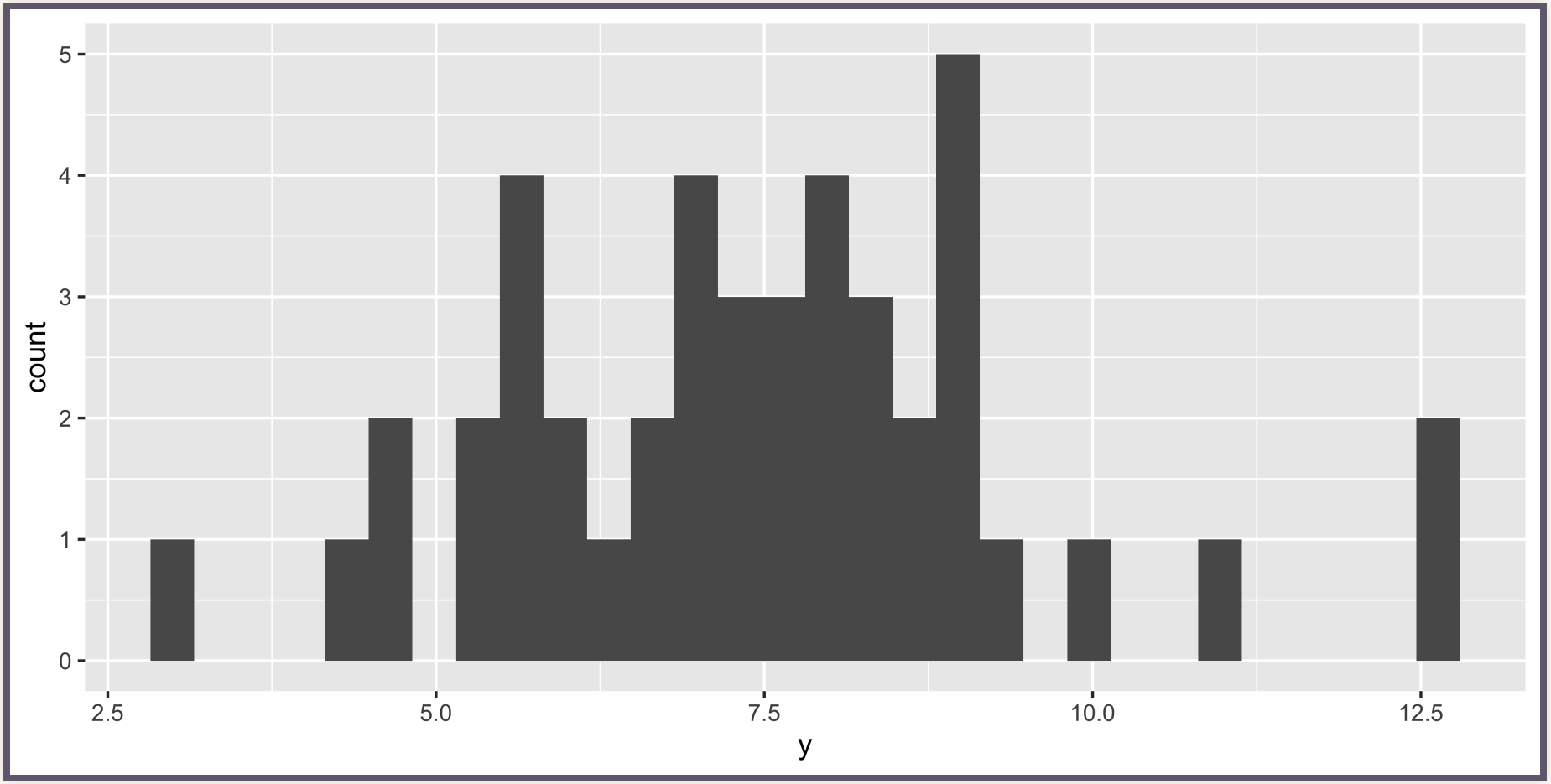
What are Geometric Objects?

There are many, and we'll learn more over the course of the lab. Today, we'll learn about these:

- geom_histogram()
- geom_point()
- geom_jitter()
- geom_smooth()
- geom_boxplot()

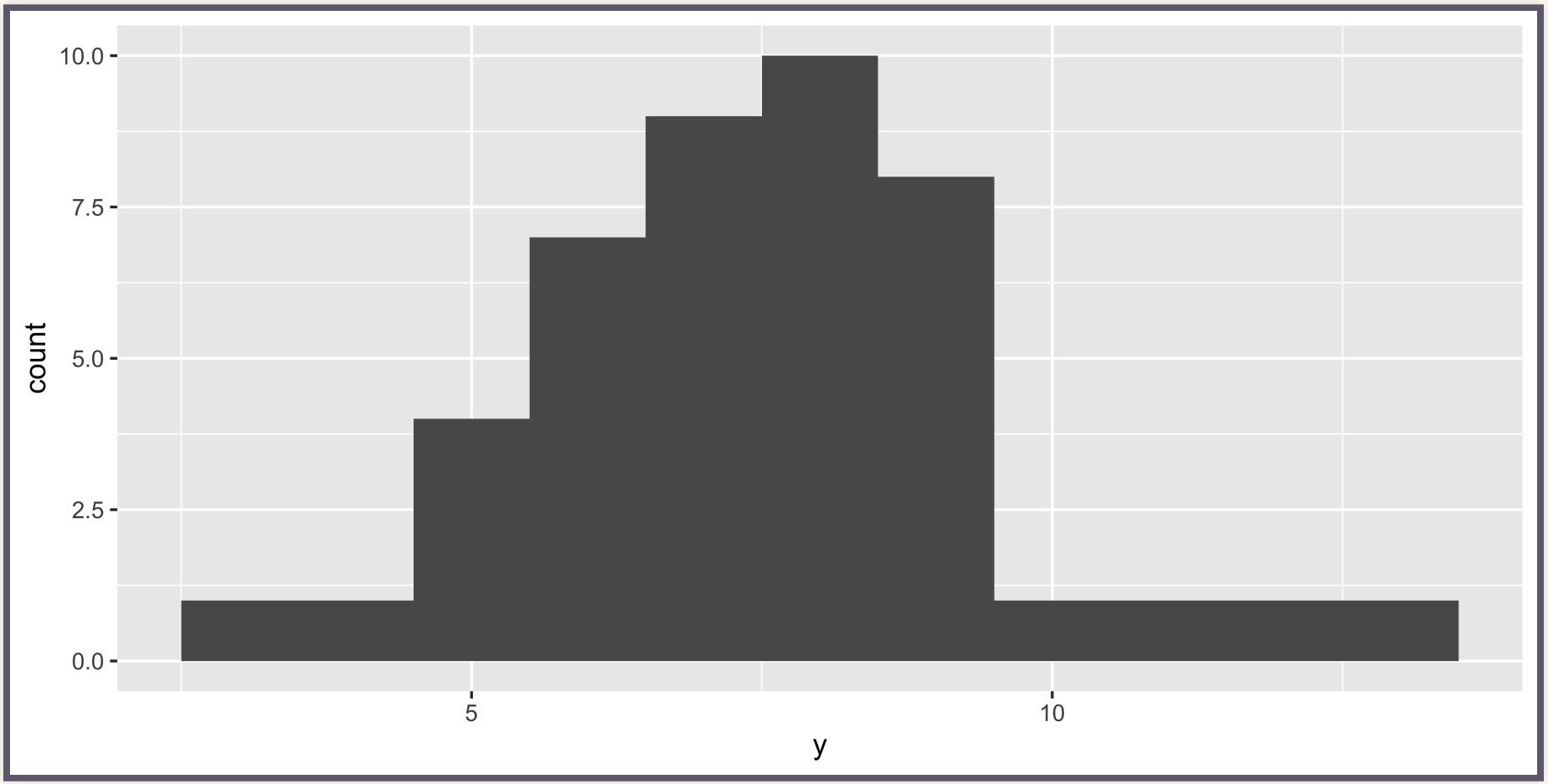
Plot 1: Geometric Objects

```
ggplot(data = anscombe_long, aes(x = y)) +
  geom_histogram()
```



Plot 1: Geometric Objects

```
ggplot(data = anscombe_long, aes(x = y)) +
  geom_histogram(binwidth = 1)
```



Learning About Geoms

Learn more about a geom by visiting the documentation for that geom. Let's start with `geom_histogram()`

```
?geom_histogram
```

The documentation will tell you required and optional aesthetics for a geom.

tRy it! Consult the Documentation

Scour the documentation for `geom_histogram()`. Which aesthetics can be mapped to `geom_histogram()`?

tRy it! Consult the Documentation (Answer)

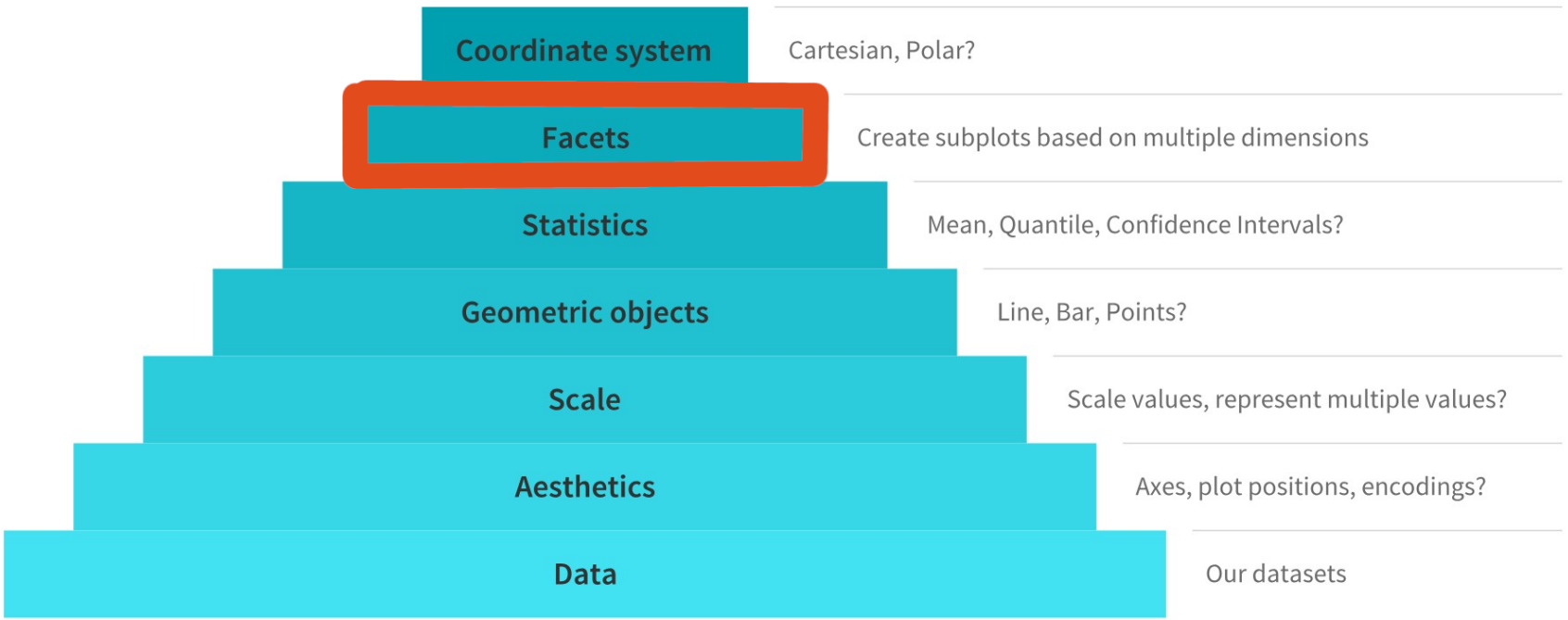
Aesthetics

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

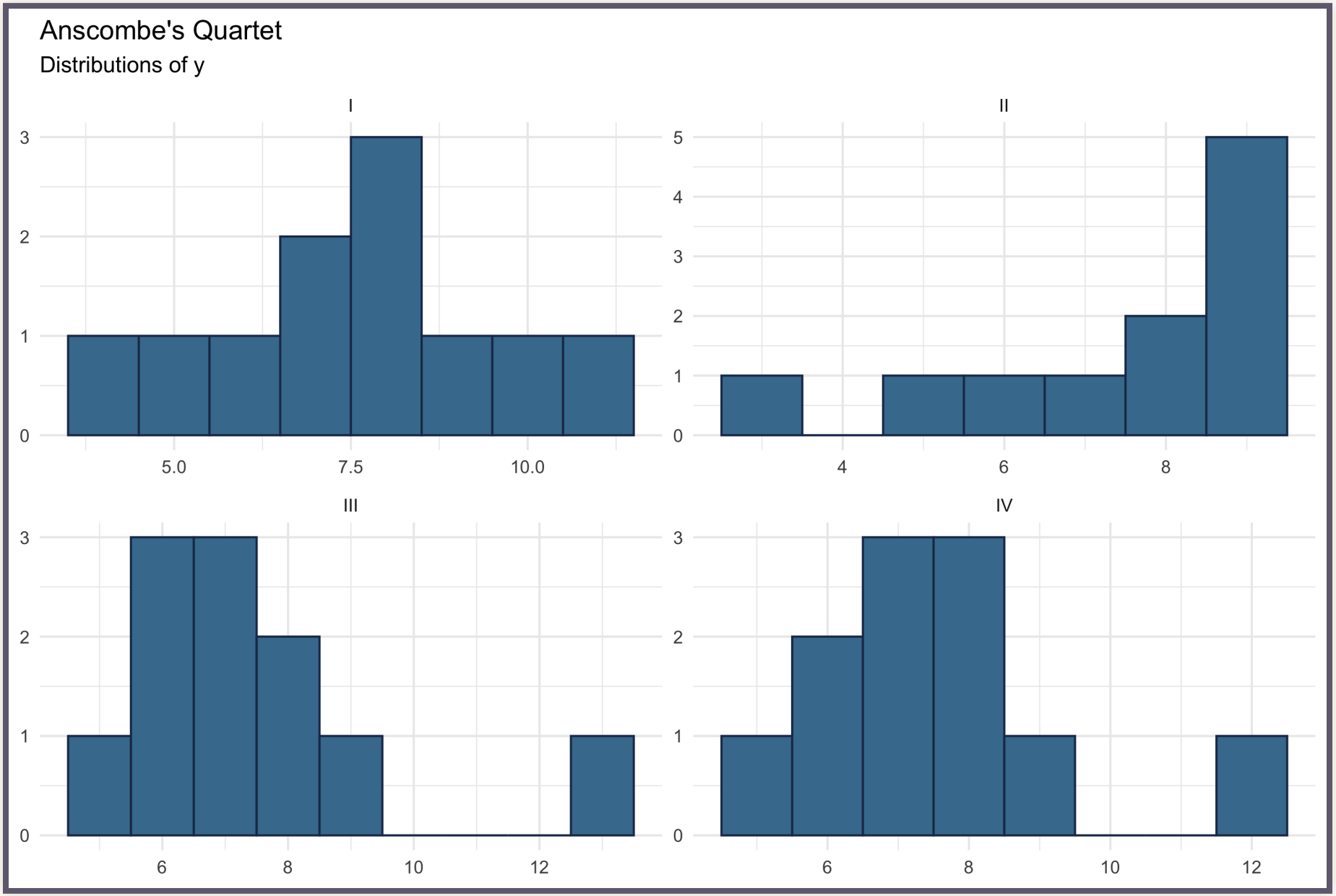
- **`x`**
- **`y`**
- `alpha`
- `colour`
- `fill`
- `group`
- `linetype`
- `size`

Facets

Major Components of the Grammar of Graphics

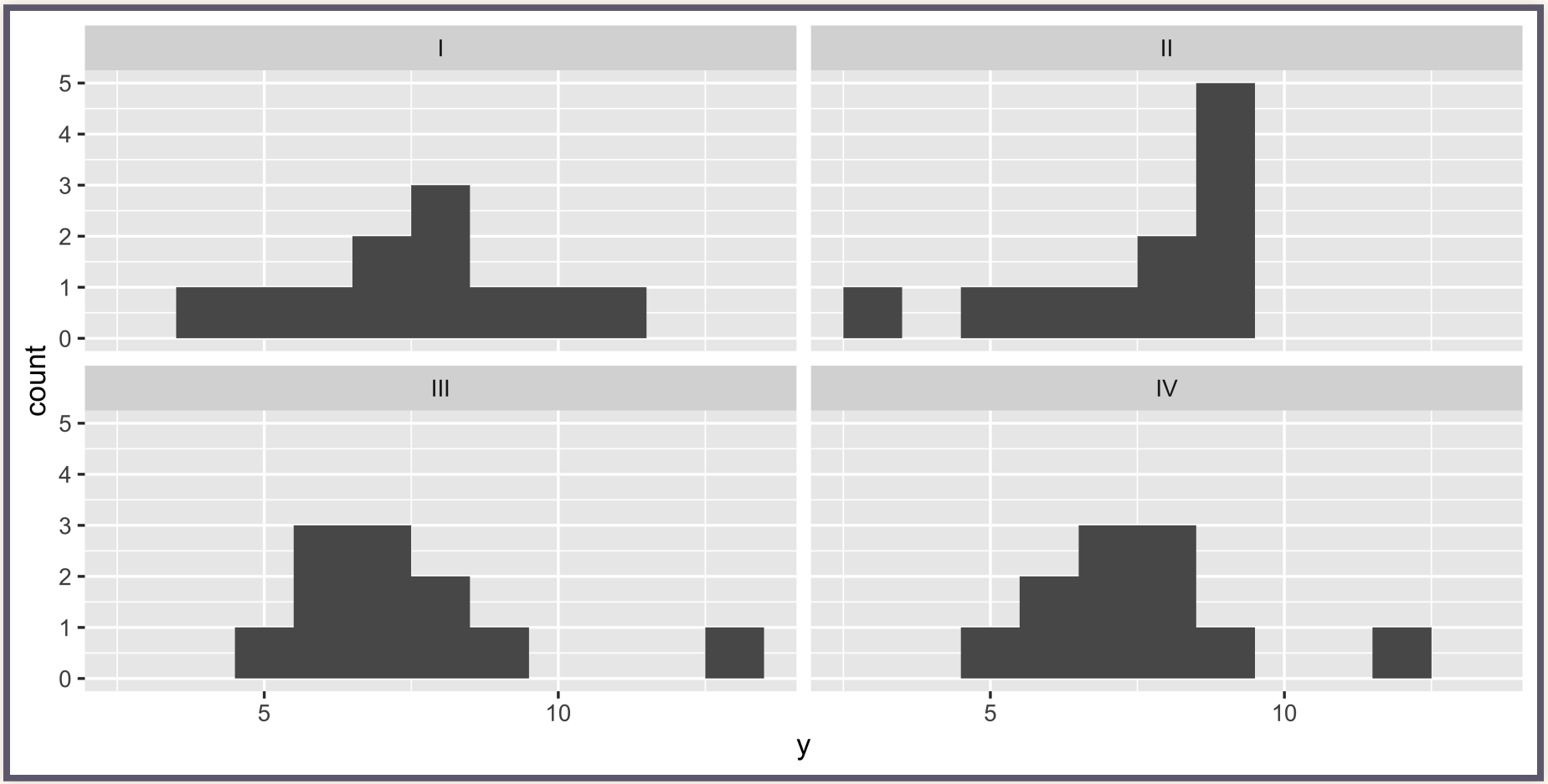


Plot 1: Facets



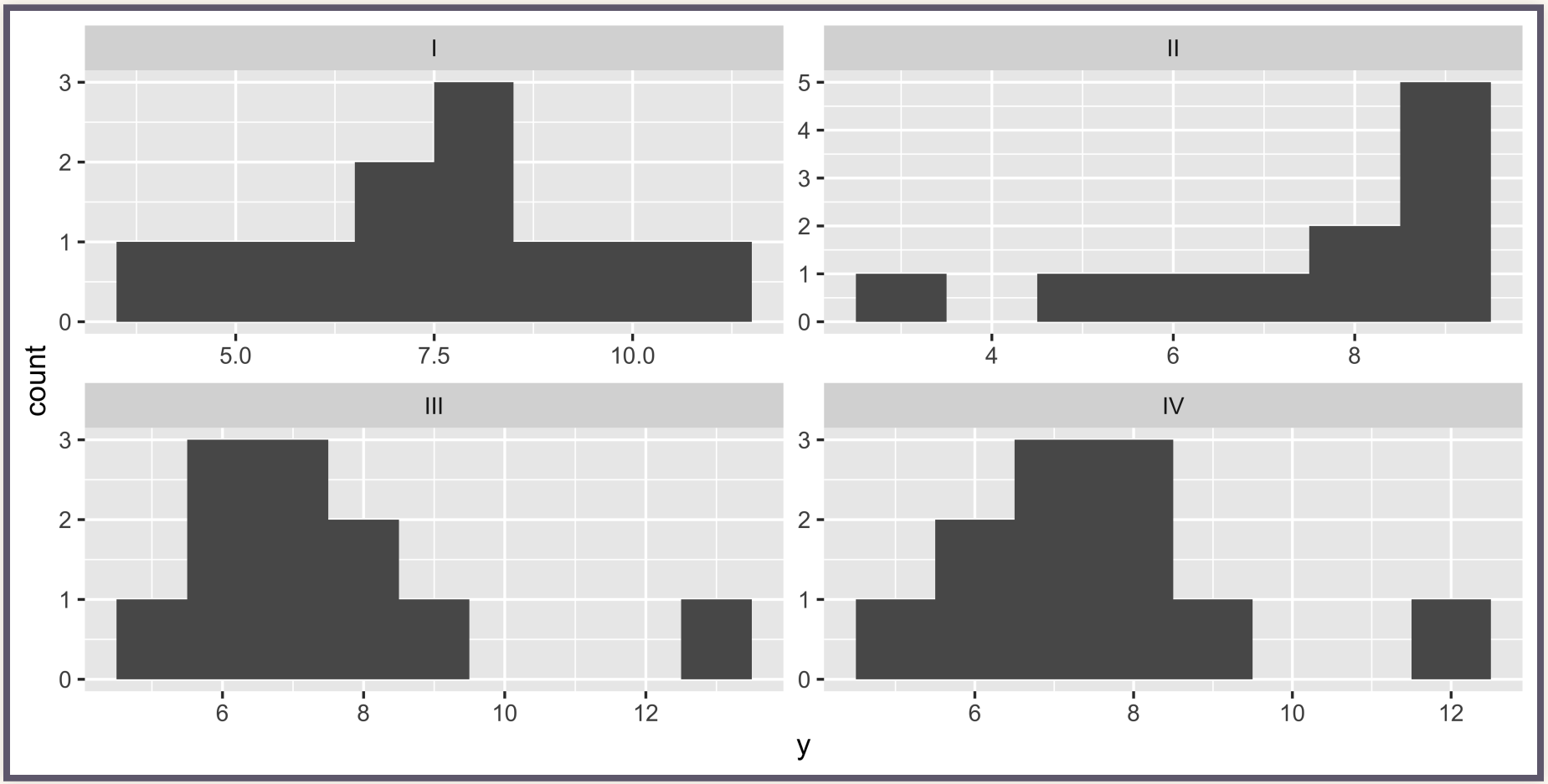
Adding Facets to Plot 1

```
ggplot(data = anscombe_long, aes(x = y)) +
  geom_histogram(binwidth = 1) +
  facet_wrap(facets = "dataset")
```



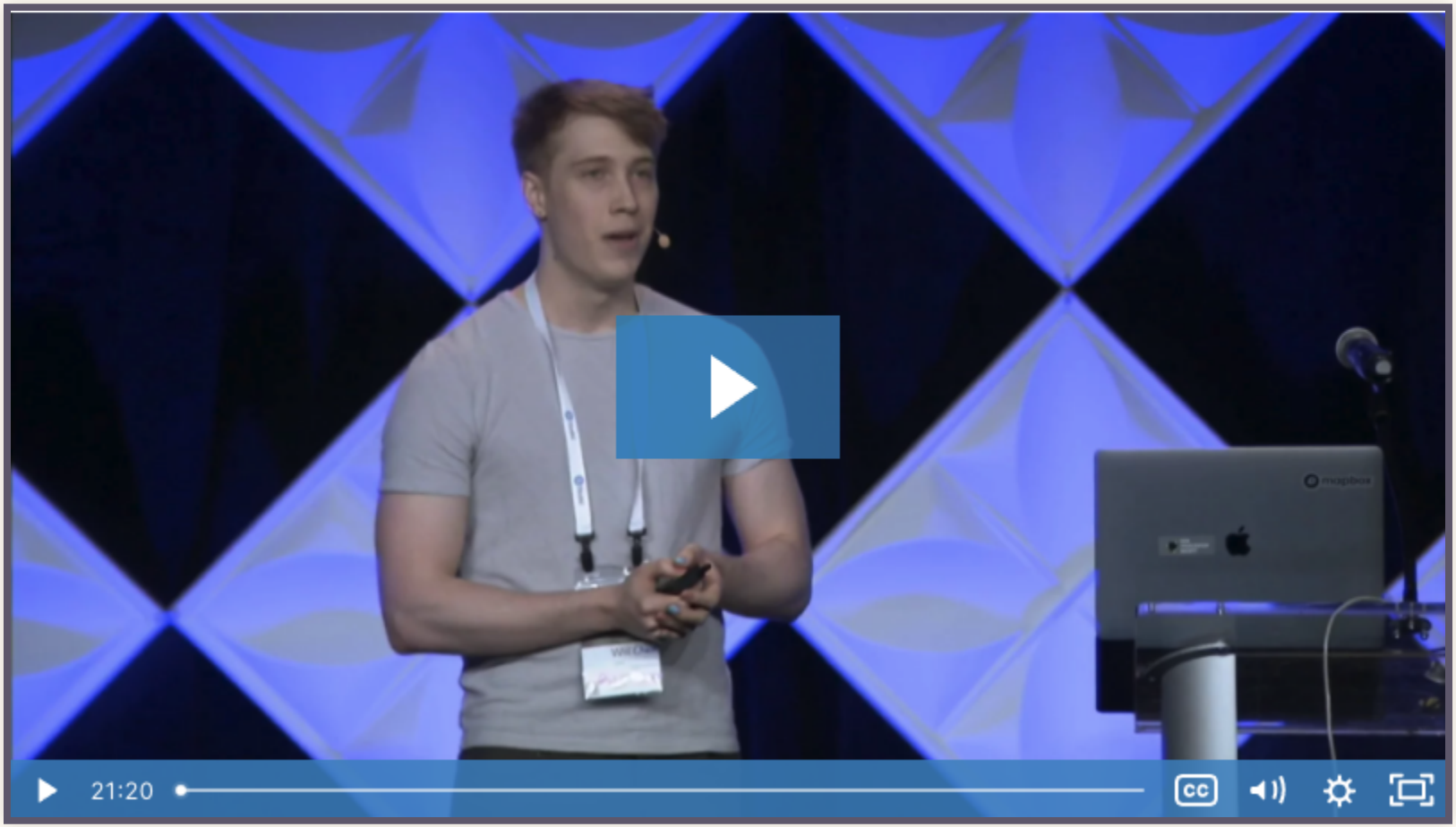
Adding Facets to Plot 1, Scales

```
ggplot(data = anscombe_long, aes(x = y)) +
  geom_histogram(binwidth = 1) +
  facet_wrap(facets = "dataset", scales = "free")
```



The Glamour of Graphics

From a conference talk given by William Chase.



Grammar vs. Glamour

Grammar of Graphics

"A tool that enables us to concisely describe the **components** of a graphic."

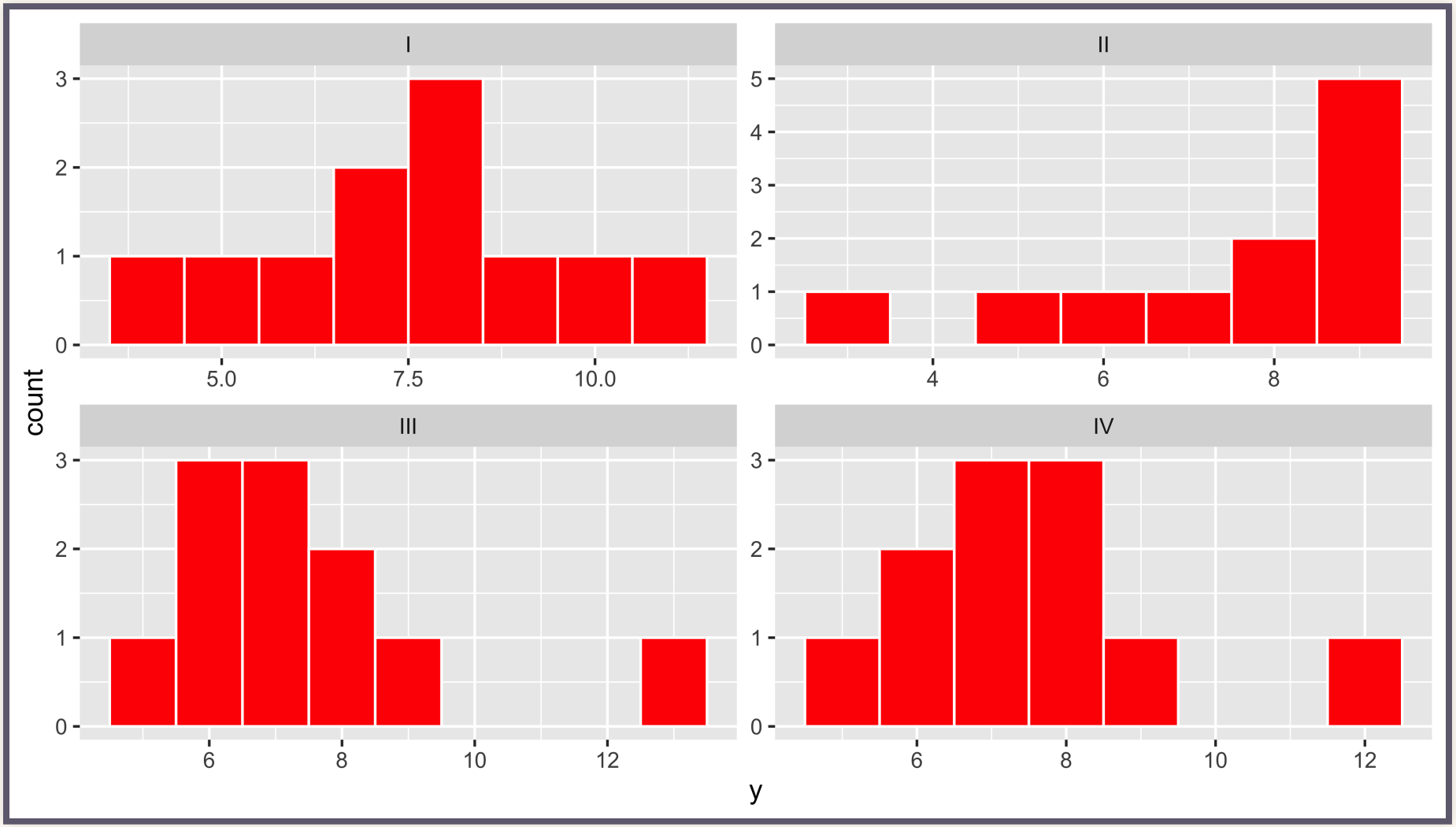
Glamour of Graphics

"A tool that enables us to concisely describe the **design** of a graphic."

Histogram Colour and Fill

```
ggplot(data = anscombe_long, aes(x = y)) +
  geom_histogram(binwidth = 1, colour = "white", fill = "red") +
  facet_wrap(facets = "dataset", scales = "free")
```

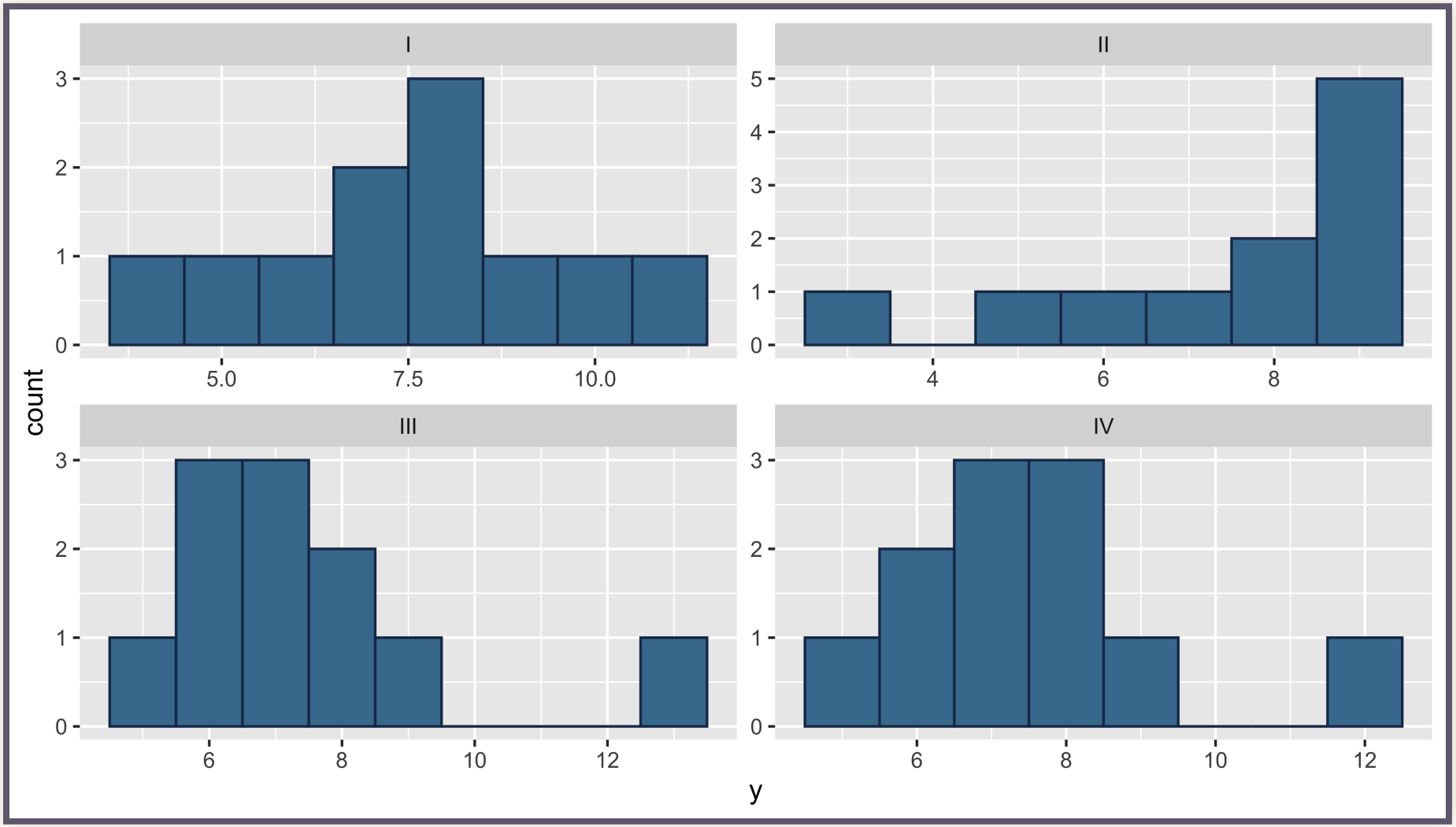
Histogram Colour and Fill (plot)



Specify Colour by Hex

```
ggplot(data = anscombe_long, aes(x = y)) +
  geom_histogram(binwidth = 1, colour = "#1d3557", fill = "#457b9d") +
  facet_wrap(facets = "dataset", scales = "free")
```

Specify Colour by Hex (plot)



Finding Colours for Plots

Colour Palettes

- https://coolors.co/
- http://colours.cafe/ for links to FB, Instagram, and Twitter.

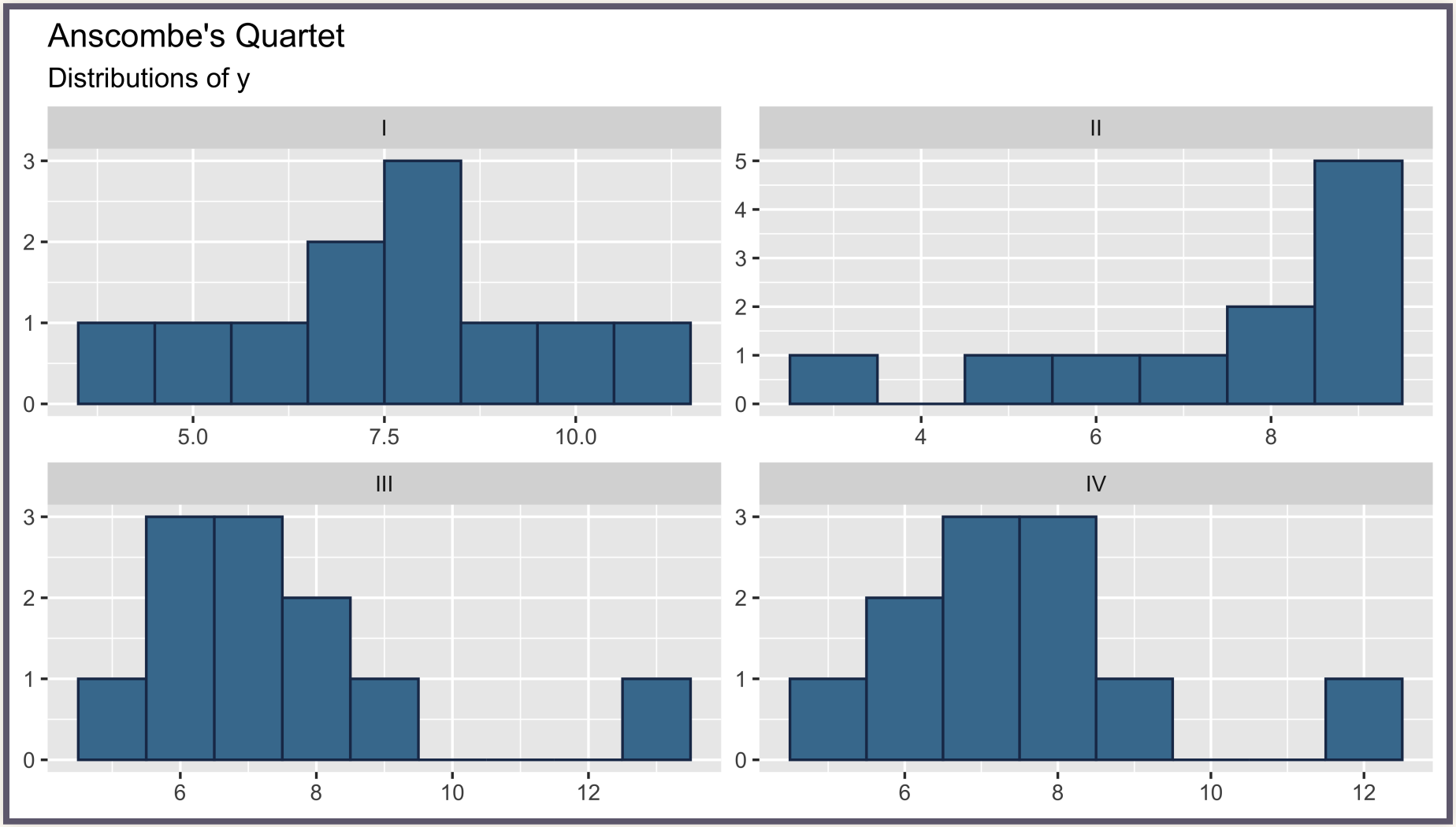
Colour Blind Filter

- https://www.toptal.com/designers/colorfilter/

Titles and Labels

```
ggplot(data = anscombe_long, aes(x = y)) +
  geom_histogram(binwidth = 1, colour = "#1d3557", fill = "#457b9d") +
  facet_wrap(facets = "dataset", scales = "free") +
  labs(
    title = "Anscombe's Quartet",
    subtitle = "Distributions of y",
    x = NULL,
    y = NULL
  )
```

Titles and Labels (plot)



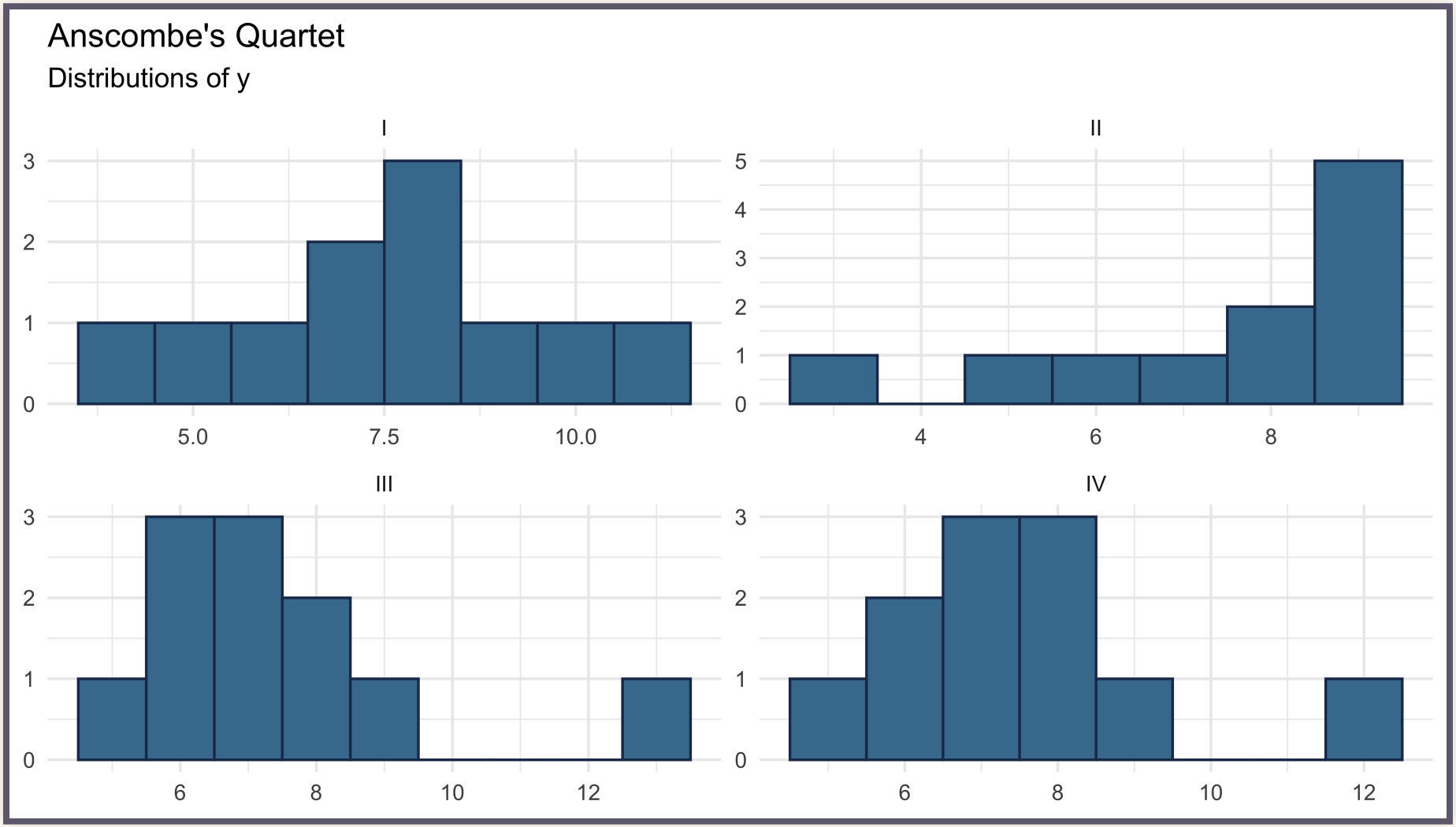
ggplot Themes

Use `ggplot` themes to change design elements of your plot. There are many and they all start with `theme_`. For example, our plot uses `theme_minimal()`, which is a good theme because it removes a lot of extraneous elements.

Add a Theme

```
ggplot(anscombe_long, aes(x = y)) +
  geom_histogram(binwidth = 1, colour = "#1d3557", fill = "#457b9d") +
  facet_wrap(facets = "dataset", scales = "free") +
  labs(
    title = "Anscombe's Quartet",
    subtitle = "Distributions of y",
    x = NULL,
    y = NULL
  ) +
  theme_minimal()
```

Add a Theme (plot)



We’re Done! Save it!

```
ggsave(filename = "plot1.png")
```

```
ggsave(filename = "plot1.png", width = 9, height = 6, dpi = 320)
```

Plot 2: Boxplot

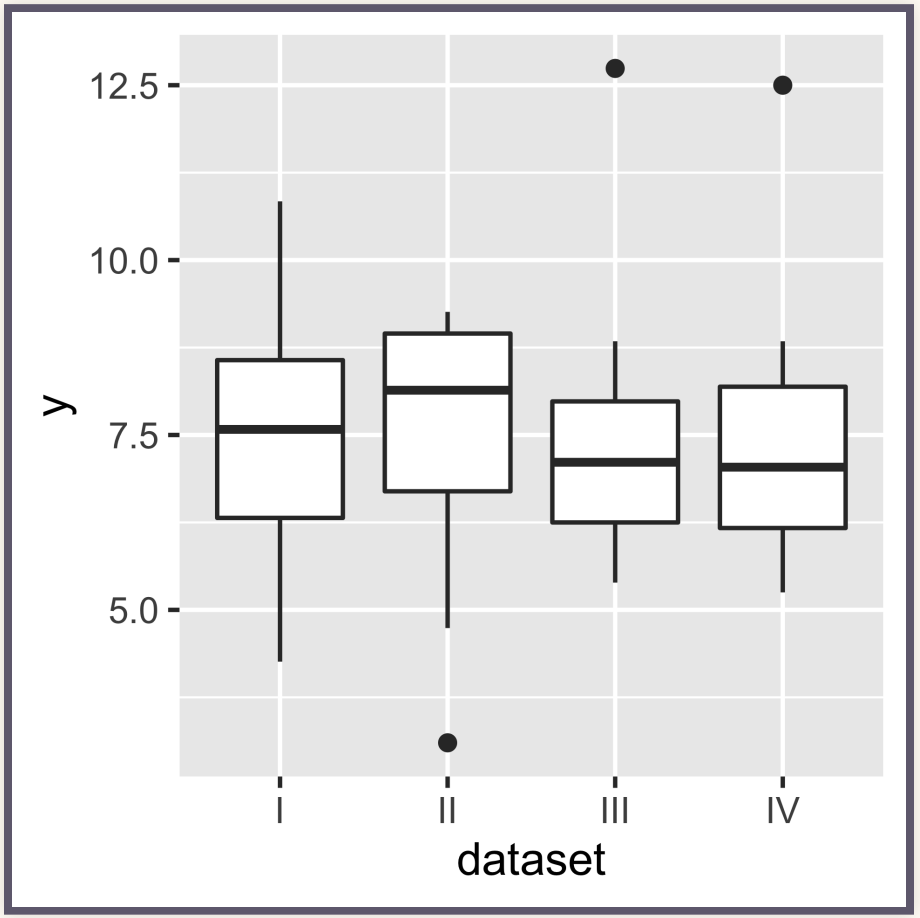
Boxplots

Five-number summary + outliers

1. Minimum (excluding outliers)
2. Maximum (excluding outliers)
3. Median
4. Lower quartile (median of the lower half of the dataset)
5. Upper quartile (median of the upper half of the dataset)
6. Outliers

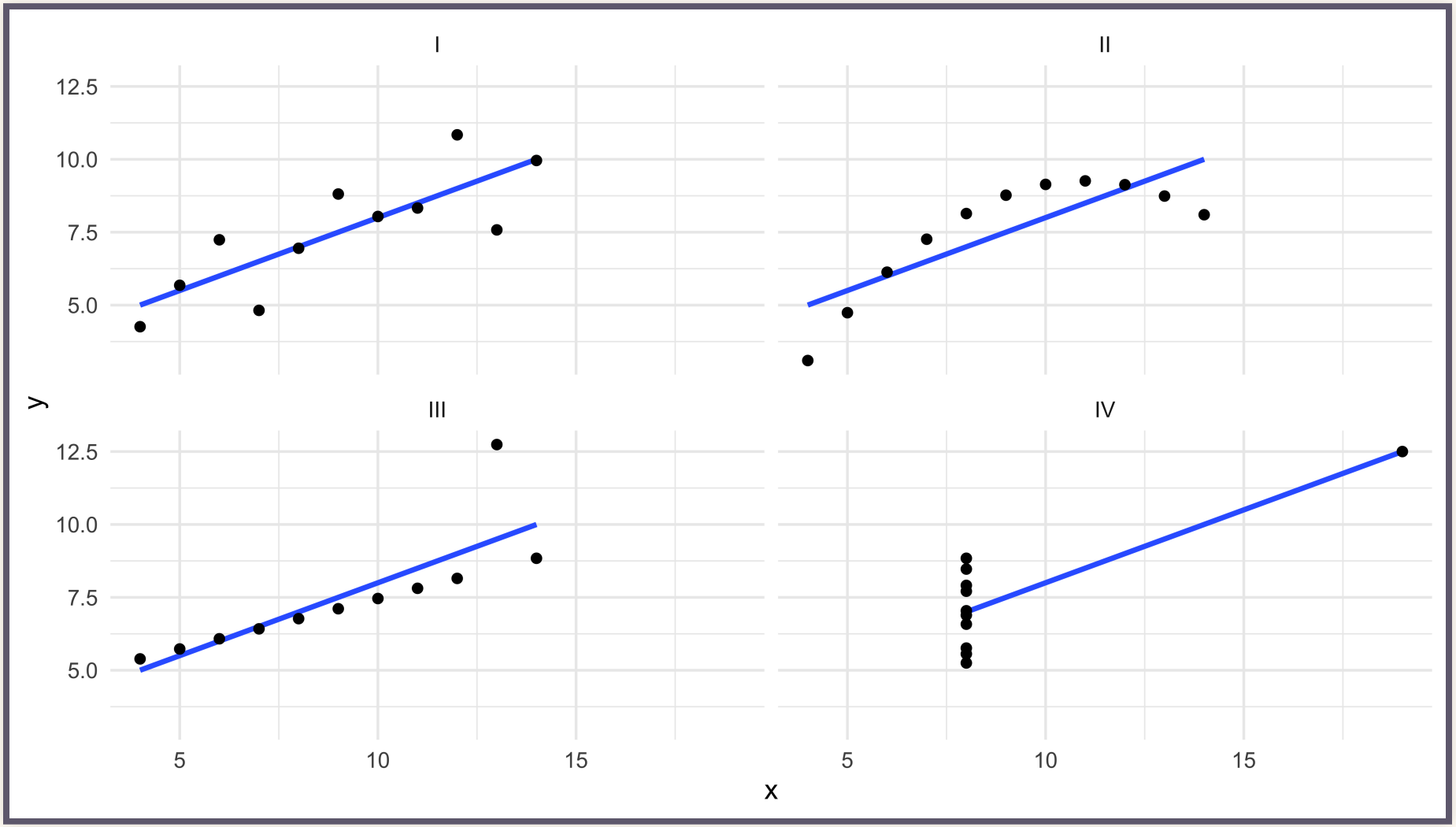
tRy it! Recreate the Example Boxplot

```
ggplot(anscombe_long, aes(x =
         dataset, y = y)) +
  geom_boxplot()
```



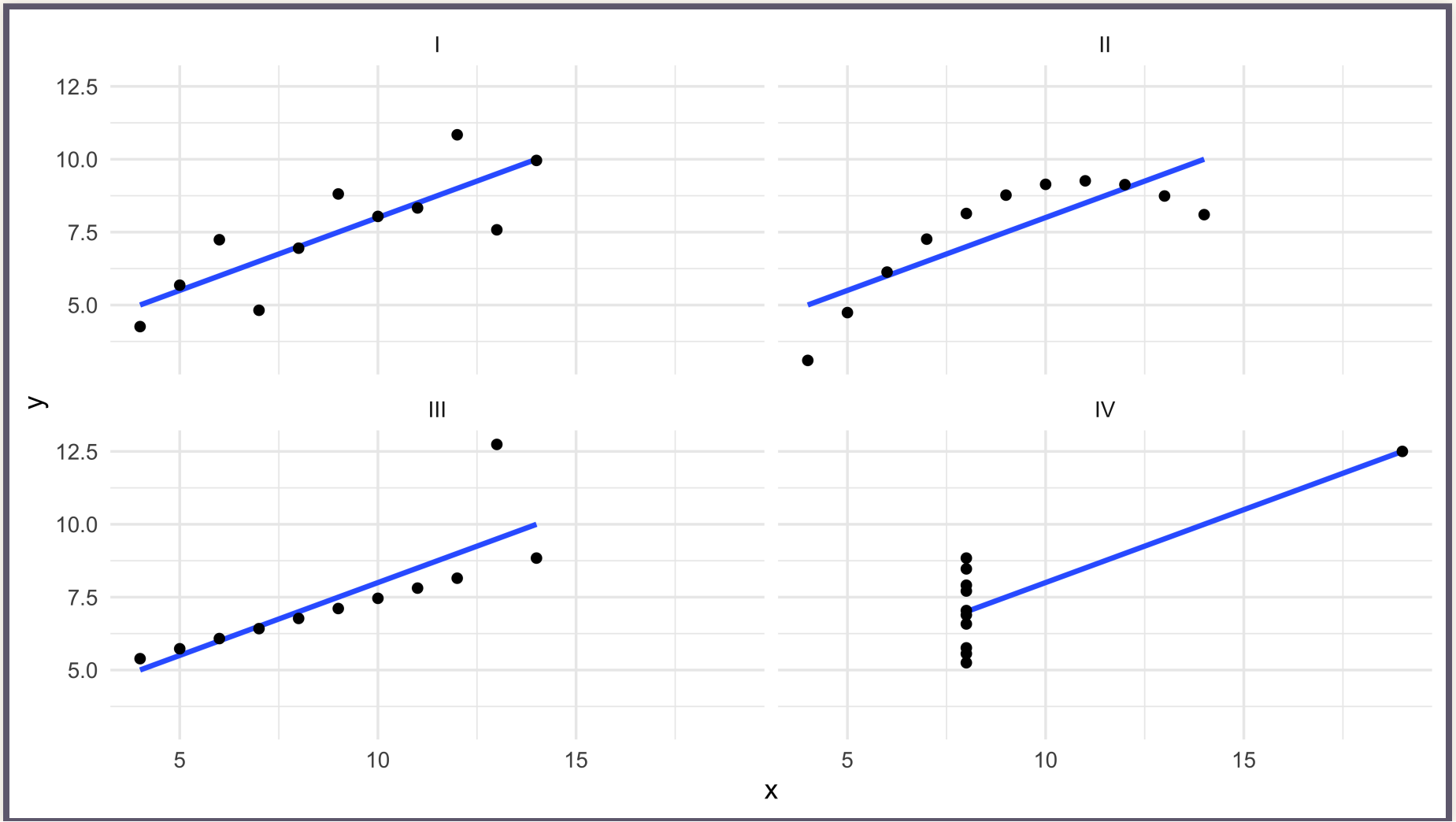
Scatter Plots with Trendlines

Plot 3



tRy it! Create Plot 3

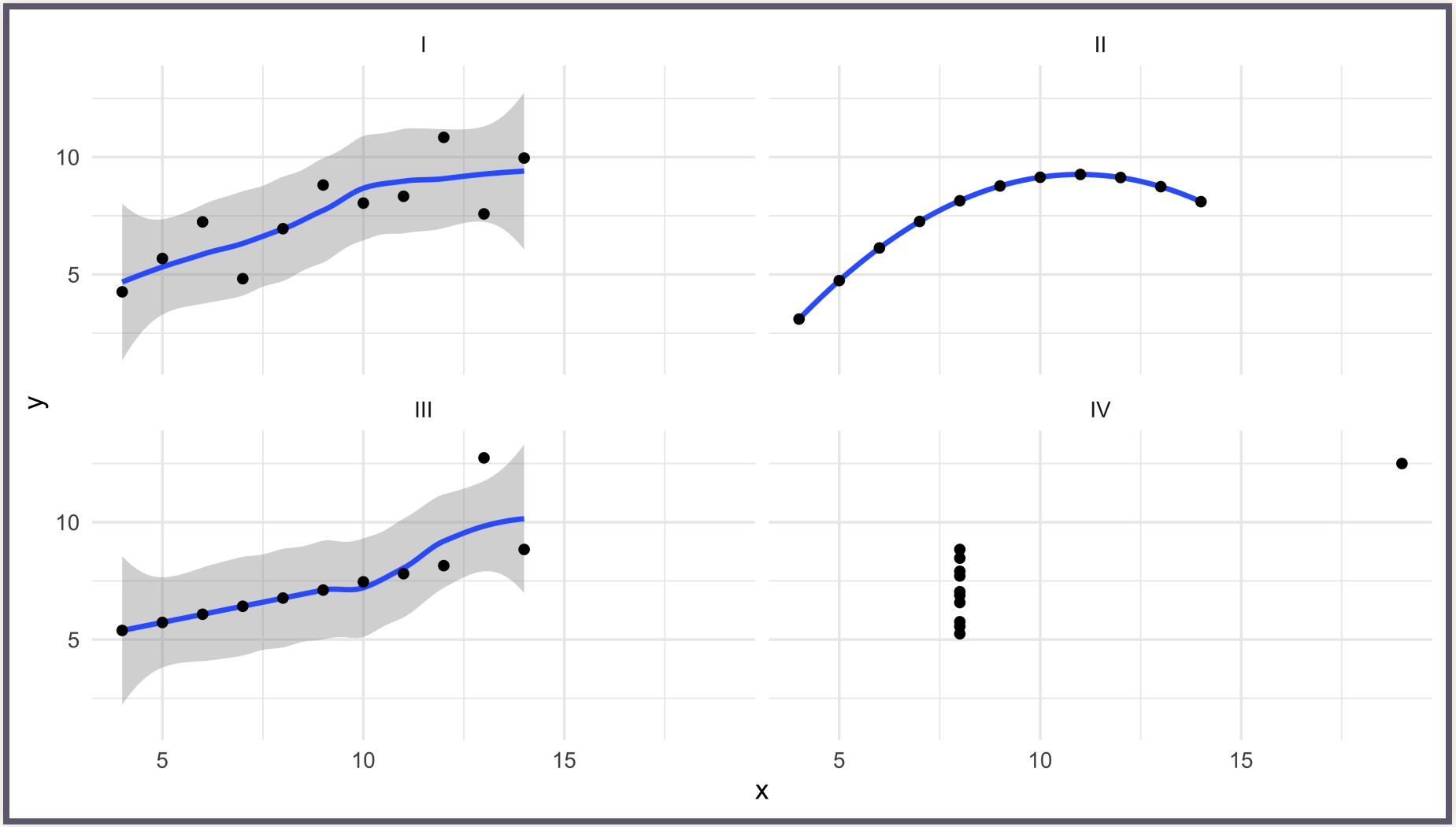
Plot 3 uses two new geoms: `geom_smooth()` and `geom_point()`.



Create Plot 3

```
ggplot(anscombe_long, aes(x = x, y = y)) +
  geom_smooth() +
  geom_point() +
  facet_wrap("dataset") +
  theme_minimal()
```

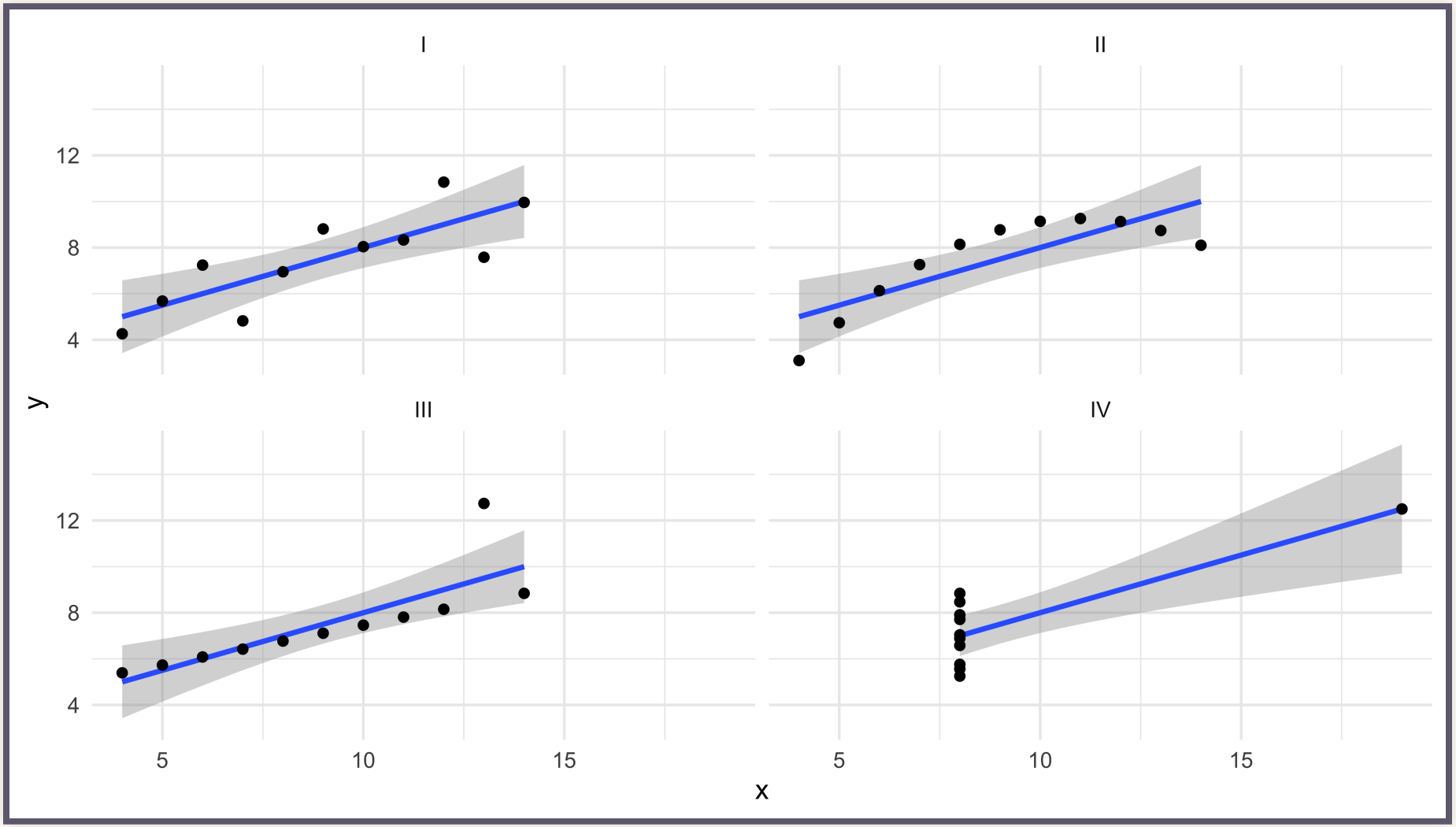
Create Plot 3



Adjust geom_smooth

```
ggplot(anscombe_long, aes(x = x, y = y)) +
  geom_smooth(method = "lm") +
  geom_point() +
  facet_wrap("dataset") +
  theme_minimal()
```

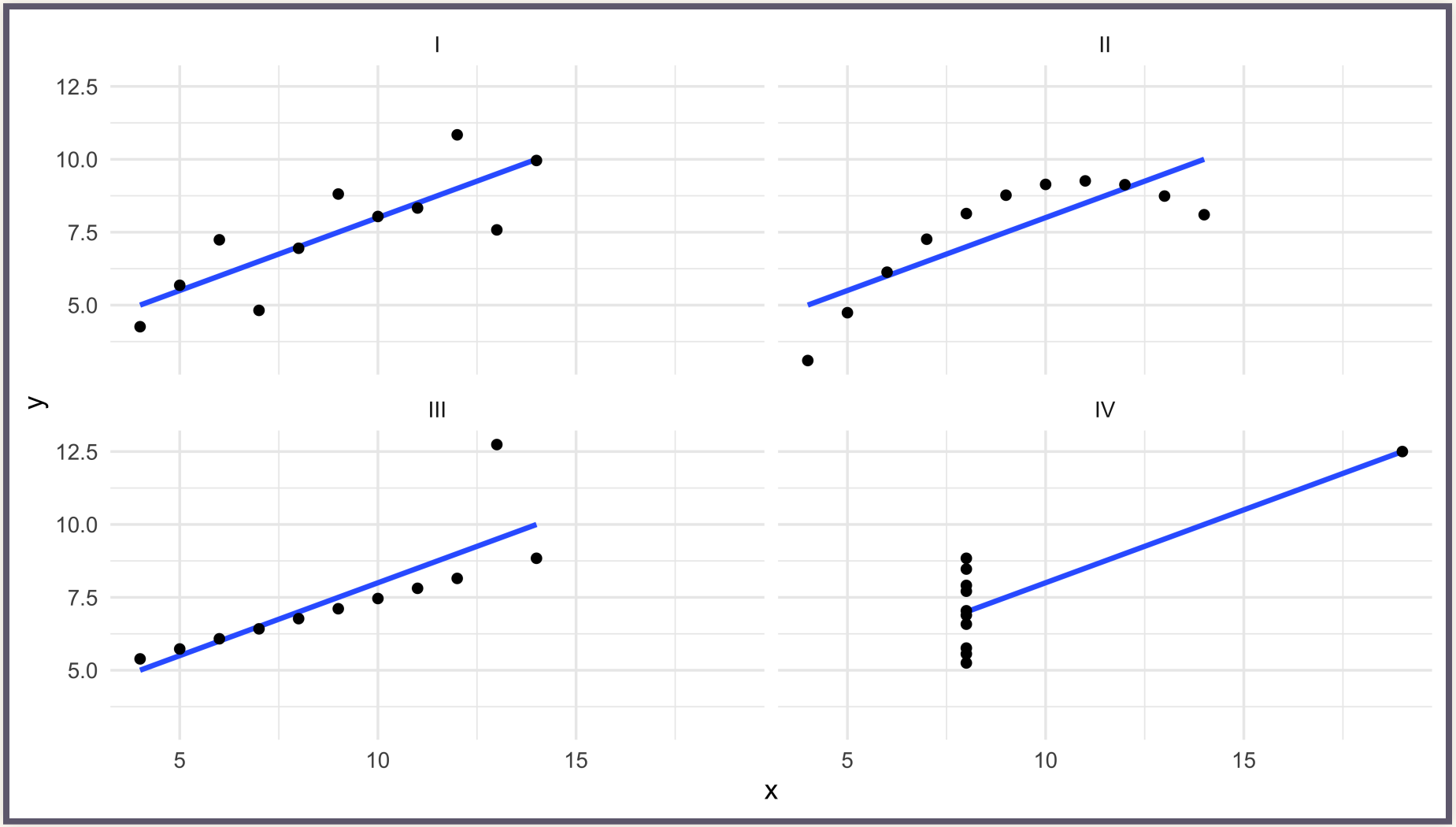
Adjust geom_smooth



Remove CI Around Smooth

```
ggplot(anscombe_long, aes(x = x, y = y)) +
  geom_smooth(method = "lm", se = FALSE) +
  geom_point() +
  facet_wrap("dataset") +
  theme_minimal()
```

Remove CI Around Smooth



Summary

What Did We Learn Today?

- Why does data visualization matter?
- What are APA guildelines for figures?
- Basics of `ggplot2` and layered grammar of graphics.
- Learned the geoms:
  - `geom_histogram()`
  - `geom_boxplot()`
  - `geom_point()`
  - `geom_smooth()`

See You Next Time!