Visualization for Data Science in R

Angela Zoss
Data Matters Fall 2024
https://www.angelazoss.com/RVis-2Day/

Welcome back!
We'll be starting soon!

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.

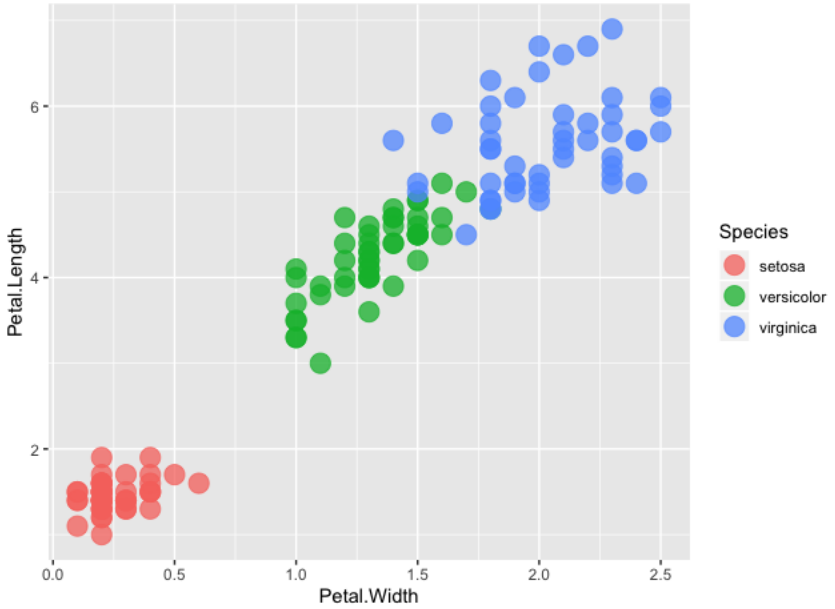
Day 1 Review

Example plot

"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)
```



General pattern

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)
```

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)

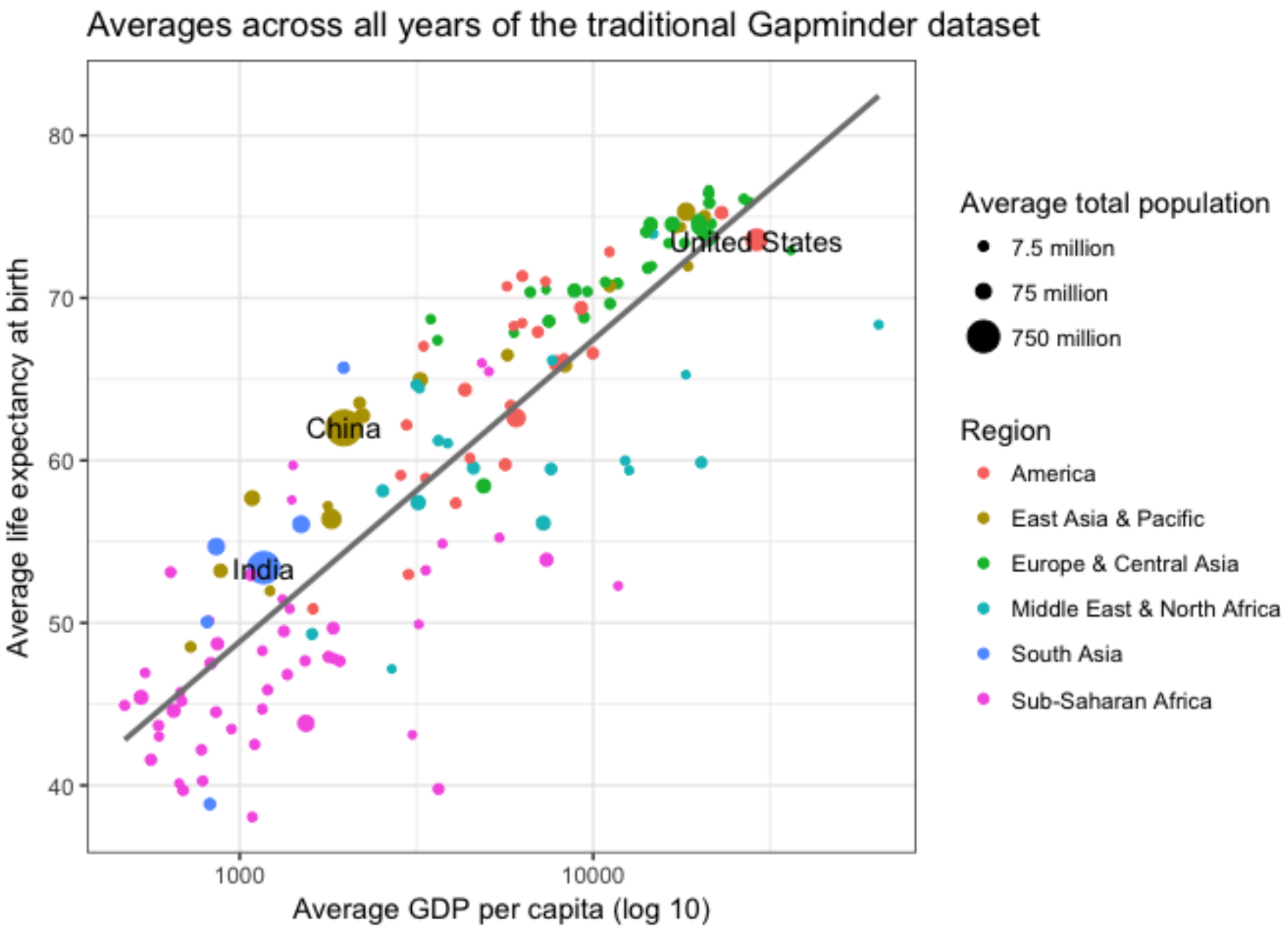
You will be adding a **scale**!

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

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

Exercise 1: Gapminder Data

http://www.gapminder.org/


Averages across all years of the traditional Gapminder dataset
Average life expectancy at birth
80
70
60
50
40
1000
10000
Average GDP per capita (log 10)
United States
China
India
Average total population
7.5 million
75 million
750 million
Region
America
East Asia & Pacific
Europe & Central Asia
Middle East & North Africa
South Asia
Sub-Saharan Africa

Saving charts out

Morning Break

Working with text variables

Text variables

In R, "character" variables

Gender	Age	Household Income	Education
Response	Response	Response	Response
Male	18-29		High school degree
Male	18-29	$0 - $24,999	Bachelor degree
Male	18-29	$0 - $24,999	High school degree
Male	18-29	$100,000 - $149,999	Some college or Associate degree
Male	18-29	$100,000 - $149,999	Some college or Associate degree
Male	18-29	$25,000 - $49,999	Bachelor degree
Male	18-29		High school degree
Male	18-29		High school degree
Male	18-29	$0 - $24,999	Some college or Associate degree
Male	18-29	$25,000 - $49,999	Some college or Associate degree
Male	18-29	$25,000 - $49,999	Bachelor degree
Male	30-44	$50,000 - $99,999	Graduate degree
Male	18-29		High school degree
Male	18-29	$0 - $24,999	Some college or Associate degree
Male	18-29	$50,000 - $99,999	Bachelor degree

Problems with text variables: Ordering

Factors

- Default ordering for categories: **alphabetical**
- Converting to factor allows you to:
  - Specify “levels” for a categorical variable
  - Specify the order of those levels
  - Specify whether the factor is “ordered”

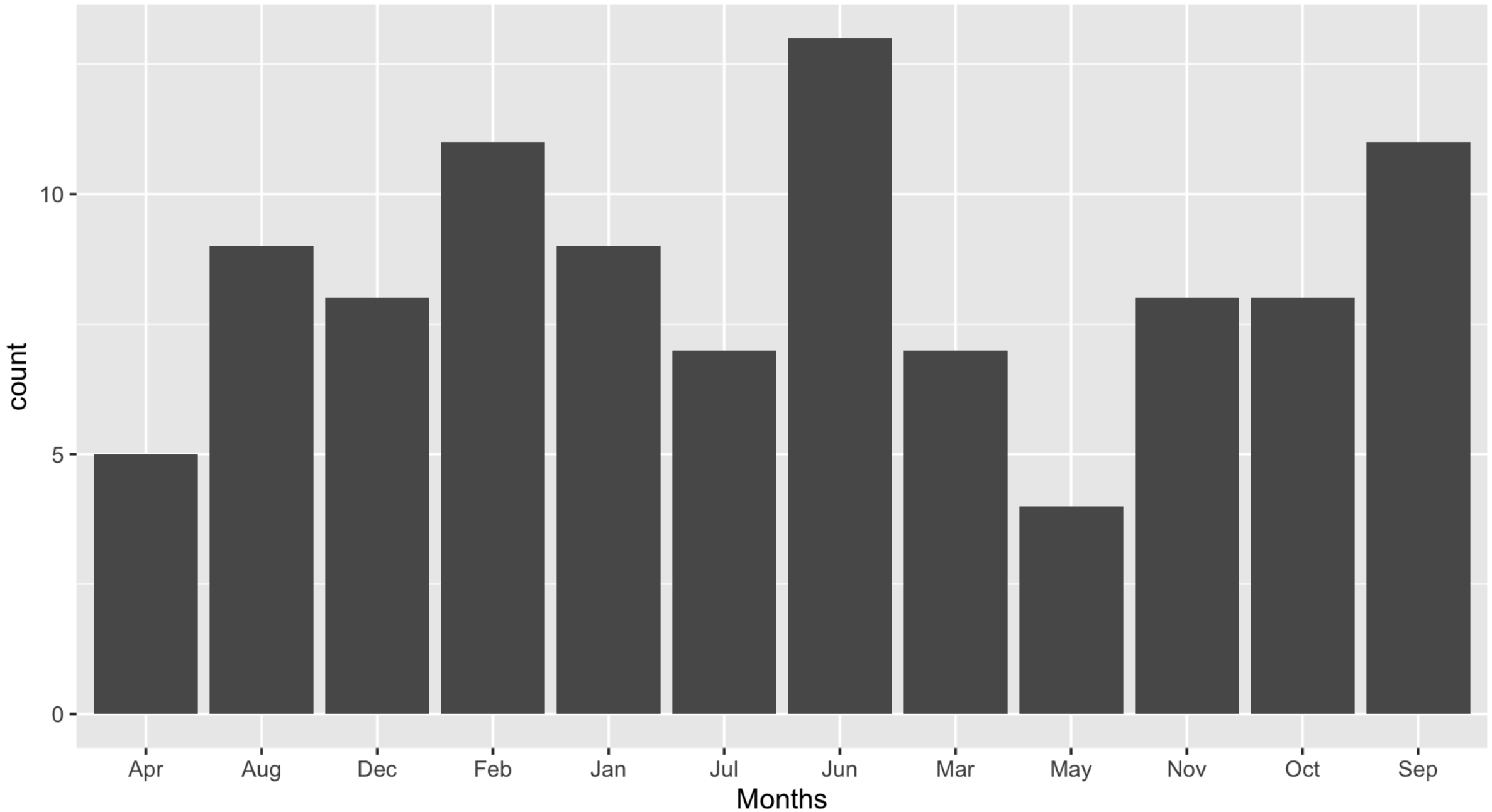
[R for Data Science: Factors]

```
> x1 <- c("Dec", "Apr", "Jan",
"Mar")

> factor(x1)
[1] Dec Apr Jan Mar
Levels: Apr Dec Jan Mar

> month_levels <- c( "Jan", "Feb",
"Mar", "Apr", "May", "Jun", "Jul",
"Aug", "Sep", "Oct", "Nov", "Dec" )

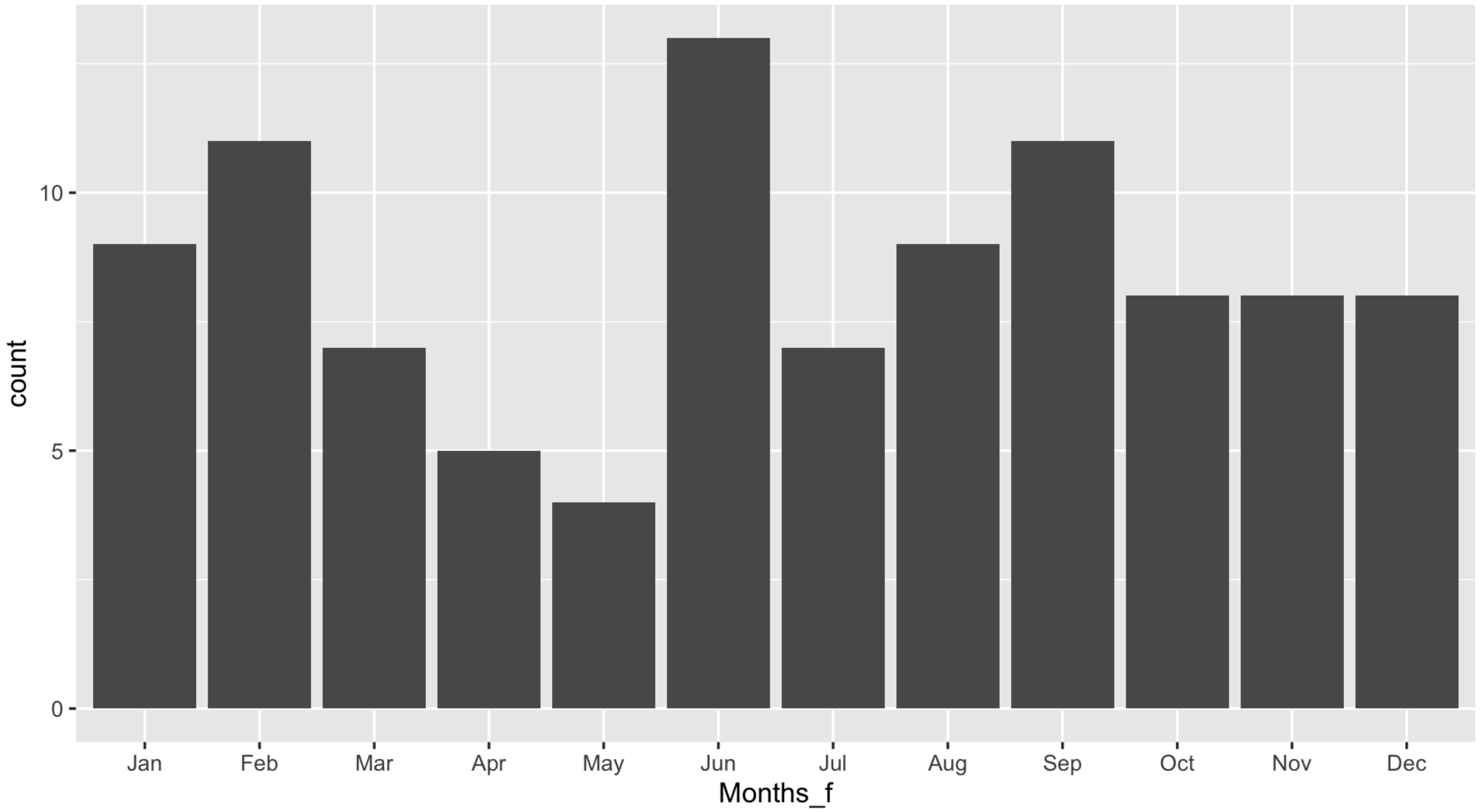
> y1 <- factor(x1,
             levels = month_levels)
> y1
[1] Dec Apr Jan Mar
Levels: Jan Feb Mar Apr May Jun Jul
Aug Sep Oct Nov Dec
```

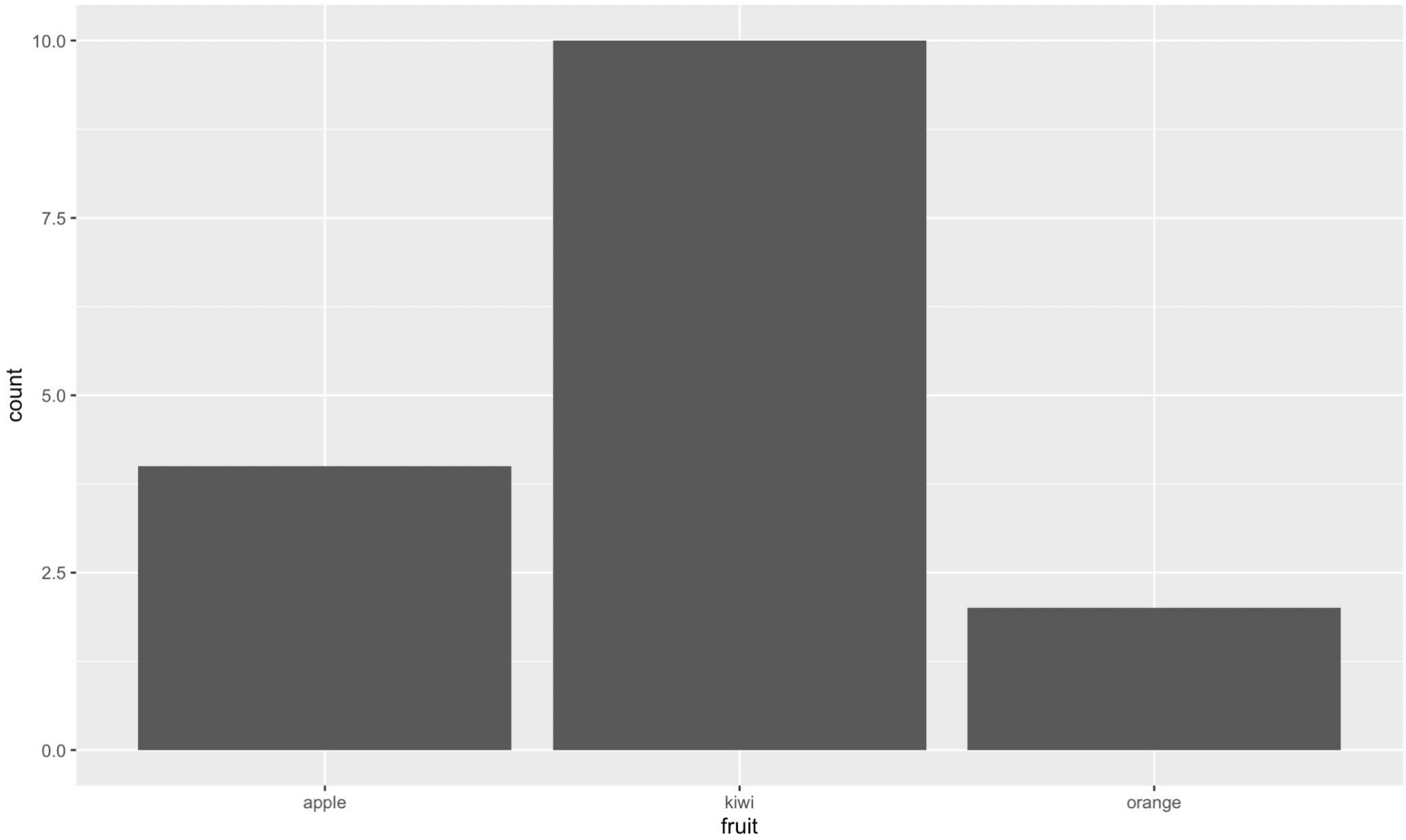

count
10
5
0
Apr
Aug
Dec
Feb
Jan
Jul
Jun
Mar
May
Nov
Oct
Sep
Months

Order by meaning

```
month_levels <- c( "Jan", "Feb", "Mar", "Apr",
"May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov",
"Dec" )


data <- data %>%
    mutate(Months_f = Months %>%
                        as_factor() %>%
                        fct_relevel(month_levels))
```

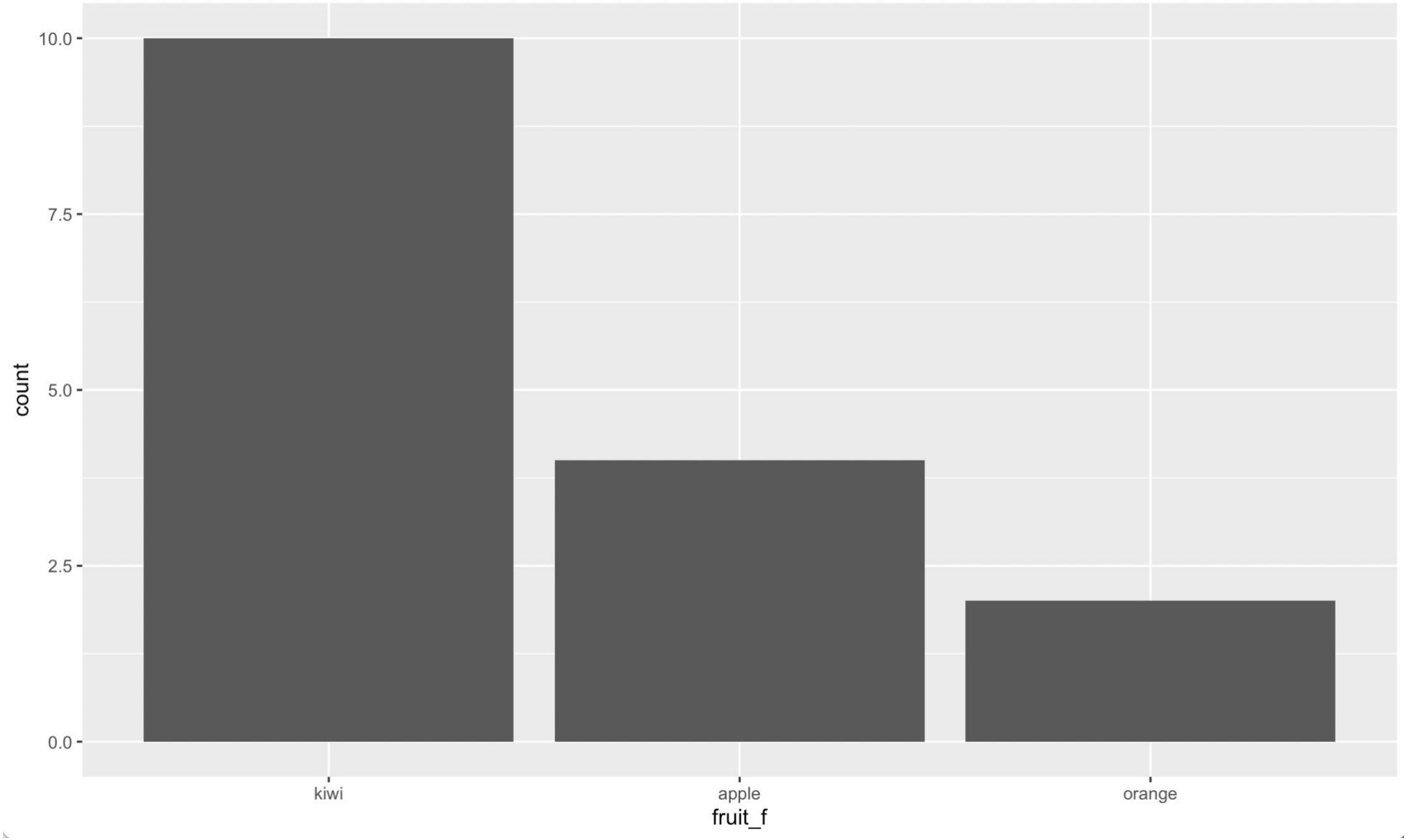

count
10
5
0
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Months_f


10.0
7.5
5.0
2.5
0.0
count
apple
kiwi
orange
fruit

Order by value (using forcats)

```
demo <- data %>%
     mutate(fruit_f = fruit %>%
                      as_factor() %>%
                      fct_infreq())


ggplot(data,
        aes(fruit %>%
             as_factor() %>%
             fct_infreq())) +
     geom_bar()
```


10.0
7.5
5.0
2.5
0.0
count
kiwi
apple
orange
fruit_f

forcats package: helpful functions

- `as_factor(char_var)`:
  convert a character variable to a factor
- `fct_infreq(factor)`:
  take factor levels and set the order according to (inverse) category frequency
- `fct_reorder(factor, num_var)`:
  sort factor levels by a second, numerical variable (like a pre-calculated count or average)

[forcats cheatsheet]

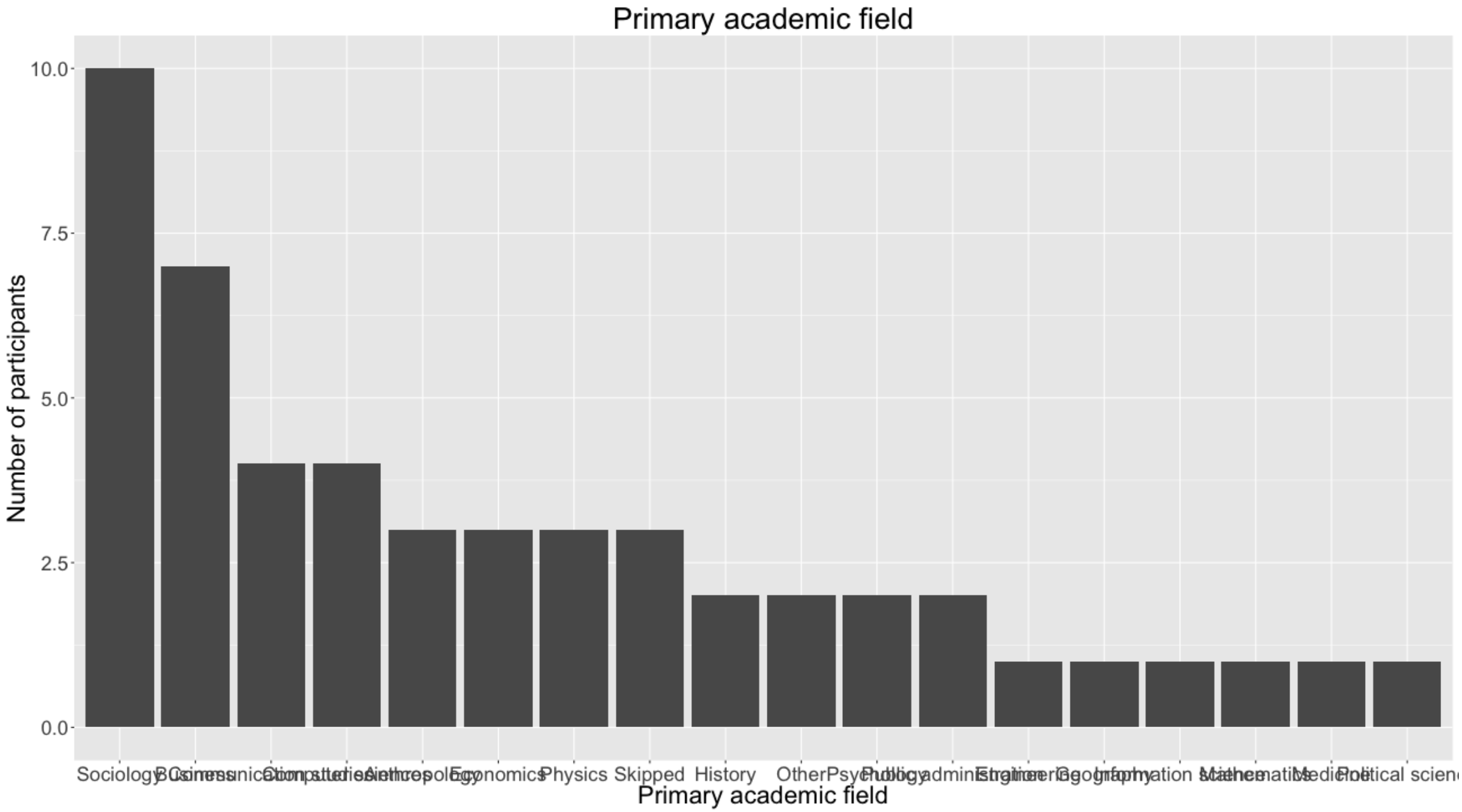
Note about read.csv (base R)

- Converts string variables to factors by default
- Can either:
  - Include `stringsAsFactors=FALSE`
  - Use `read_csv()` instead

Factoring resources from Amelia McNamara

- RStudioConf 2019: Working with Categorical Data in R Without Losing Your Mind (slides, video)
- Wrangling Categorical Data in R article
- Wrangling Categorical Data in R repository

Problems with text variables: Long category names

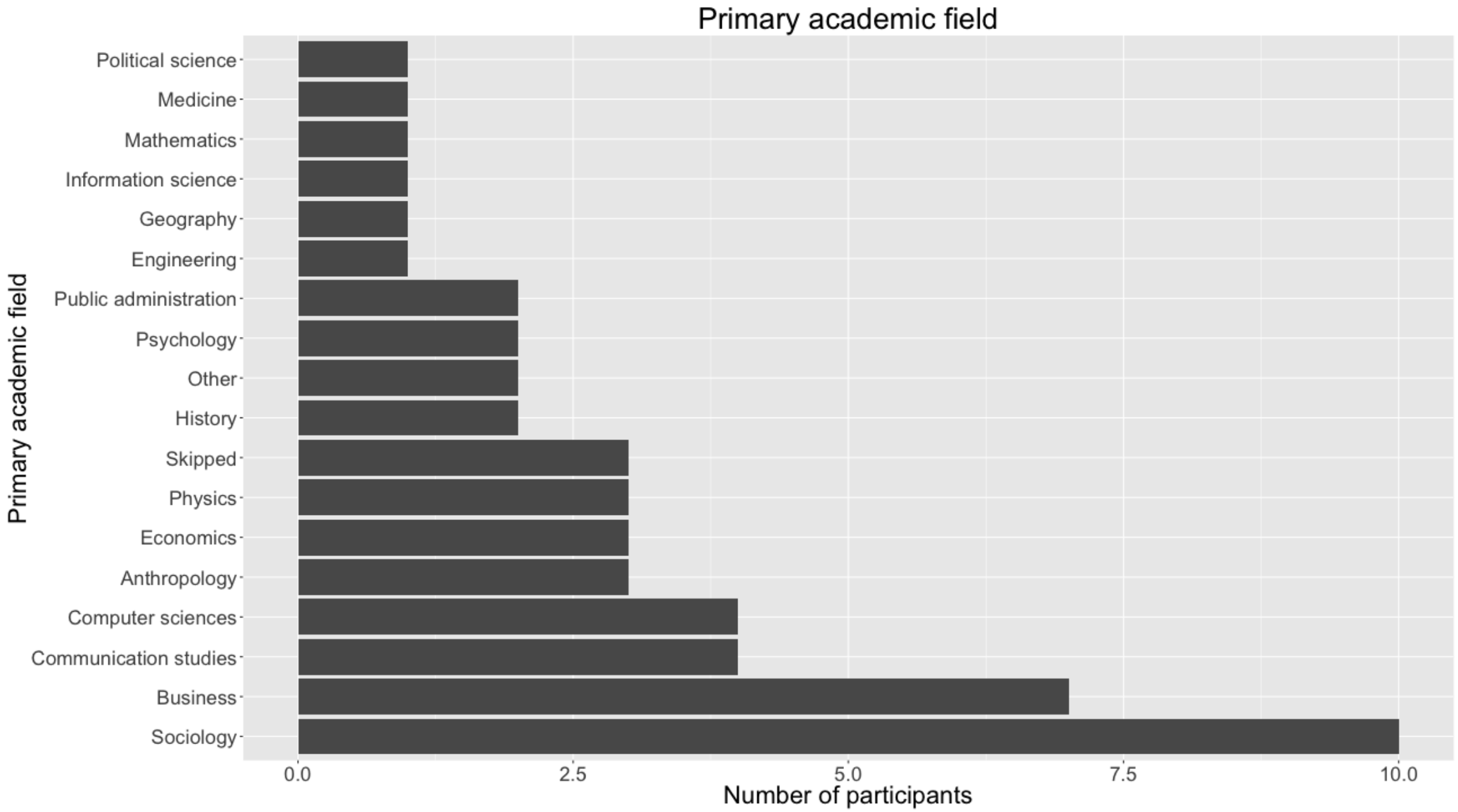

Primary academic field
Number of participants
10.0
7.5
5.0
2.5
0.0
Sociology
Business
Anthropology
Economics
Physics
Skipped
History
Other
Psychology
Primary academic field

In ggplot2, have to flip the axes

```
+ coord_flip()

or

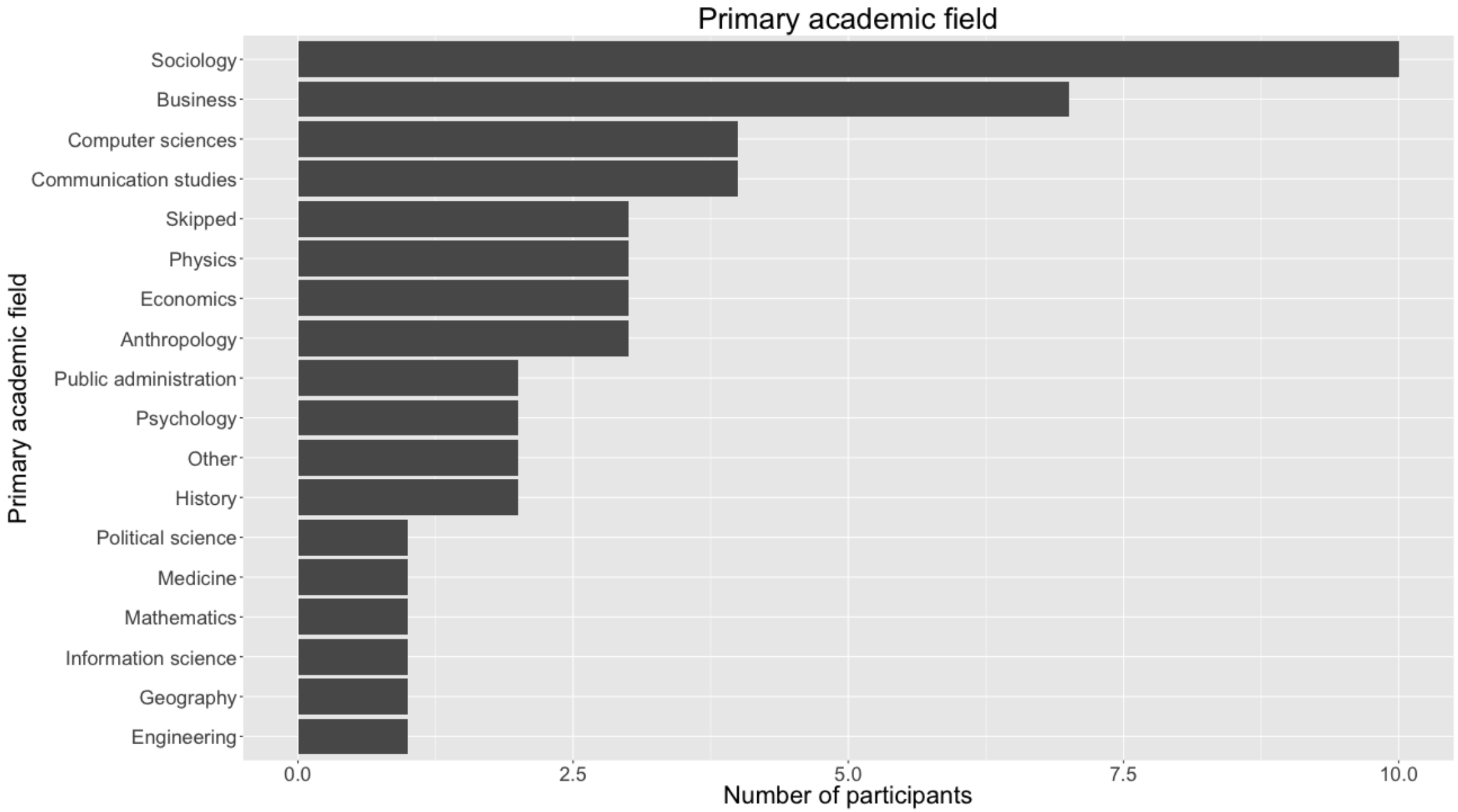
ggplot(df, aes(y=cat_variable)) +
    geom_bar()
```


Primary academic field
Political science
Medicine
Mathematics
Information science
Geography
Engineering
Public administration
Psychology
Other
History
Skipped
Physics
Economics
Anthropology
Computer sciences
Communication studies
Business
Sociology
Primary academic field
0.0
2.5
5.0
7.5
10.0
Number of participants

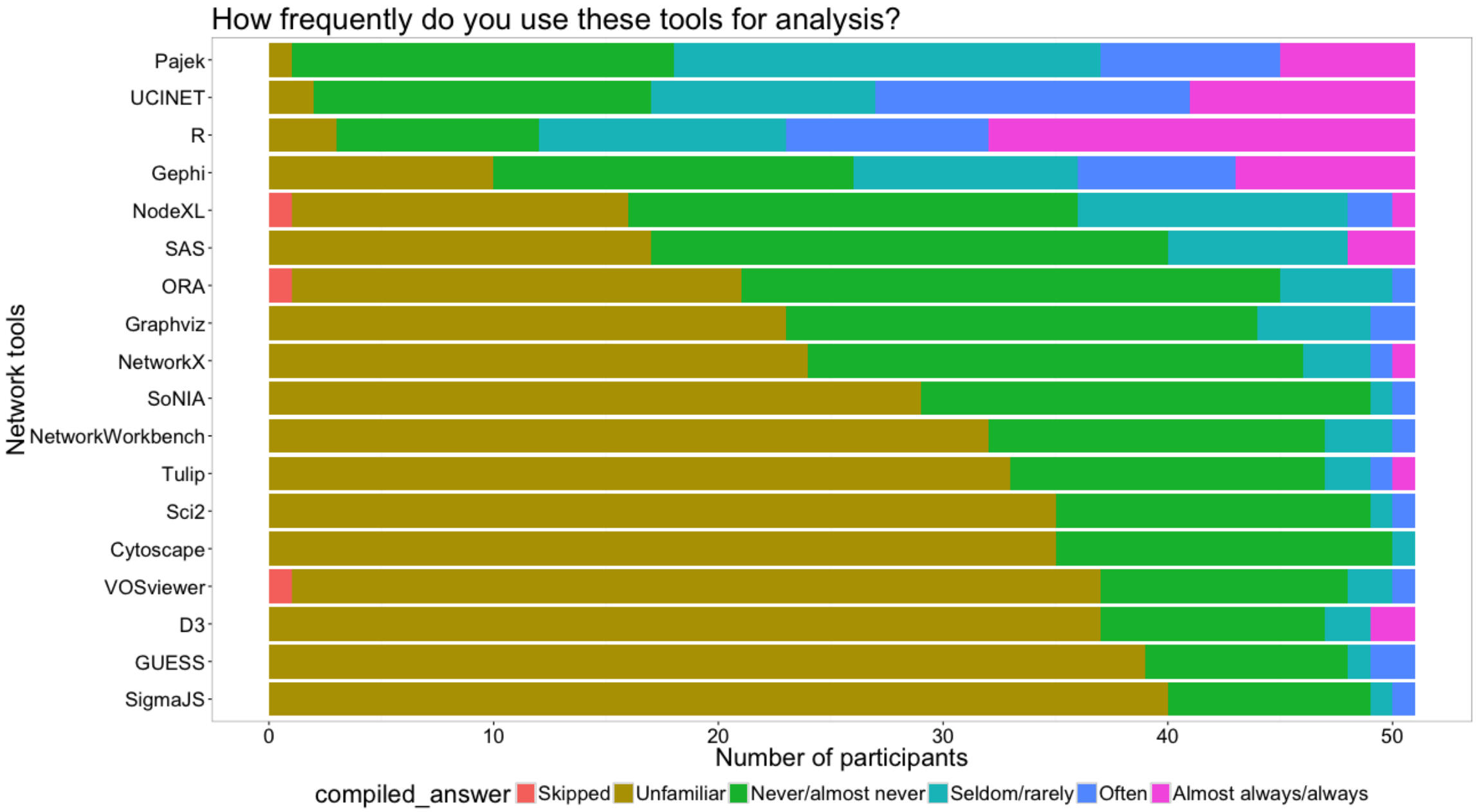
When you flip axes, you sort the other way

```
academic_field %>%
    as_factor() %>%
    fct_infreq() %>%
    fct_rev()
```

Have to reverse the order of the levels


Primary academic field
Sociology
Business
Computer sciences
Communication studies
Skipped
Physics
Economics
Anthropology
Public administration
Psychology
Other
History
Political science
Medicine
Mathematics
Information science
Geography
Engineering
Primary academic field
0.0
2.5
5.0
7.5
10.0
Number of participants

Problems with text variables: Arbitrary colors

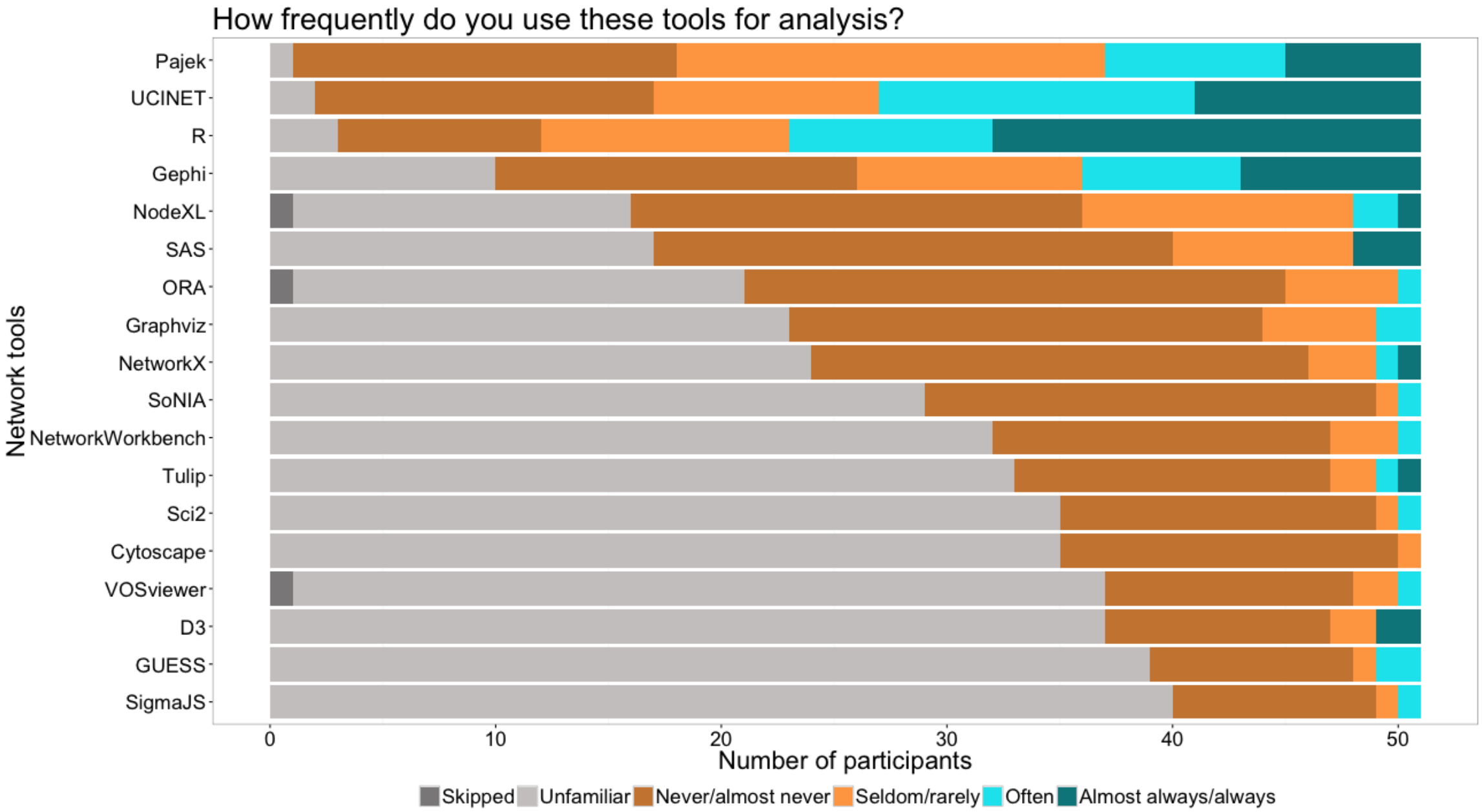

How frequently do you use these tools for analysis?
Network tools
Pajek
UCINET
R
Gephi
NodeXL
SAS
ORA
Graphviz
NetworkX
SoNIA
NetworkWorkbench
Tulip
Sci2
Cytoscape
VOSviewer
D3
GUESS
SigmaJS
0
10
20
30
40
50
Number of participants
compiled_answer
Skipped
Unfamiliar
Never/almost never
Seldom/rarely
Often
Almost always/always

Select colors manually, or use alternate palette

```
scale_fill_manual(
    values=c("snow4","snow3",
             "tan3","tan1",
             "turquoise2","turquoise4"))



# Also see package RColorBrewer
scale_fill_brewer(palette="BrBG")
```


How frequently do you use these tools for analysis?
Network tools
Pajek
UCINET
R
Gephi
NodeXL
SAS
ORA
Graphviz
NetworkX
SoNIA
NetworkWorkbench
Tulip
Sci2
Cytoscape
VOSviewer
D3
GUESS
SigmaJS
0
10
20
30
40
50
Number of participants
Skipped
Unfamiliar
Never/almost never
Seldom/rarely
Often
Almost always/always

Lunch

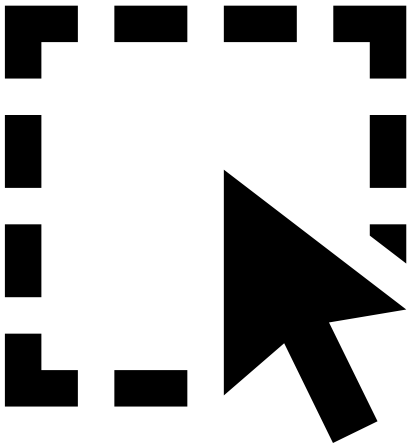
Designing tools for data exploration

Supporting data exploration

Output

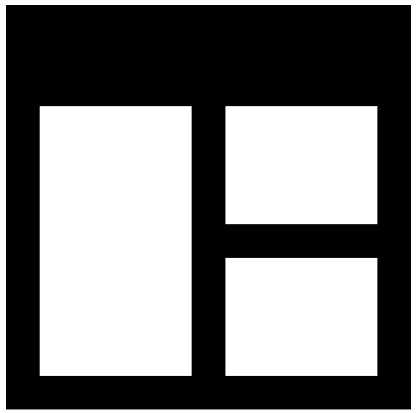
Picking the right visual elements

Input



Giving users the right controls

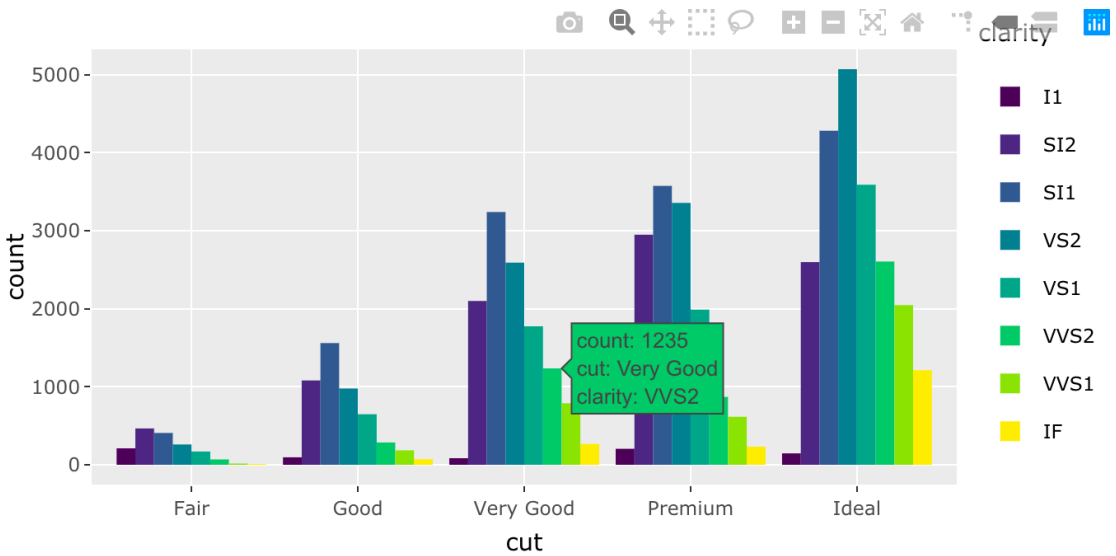
Layout



Arranging everything in the right place

Interactive components

- Start with the basic info
- Show more or less on demand

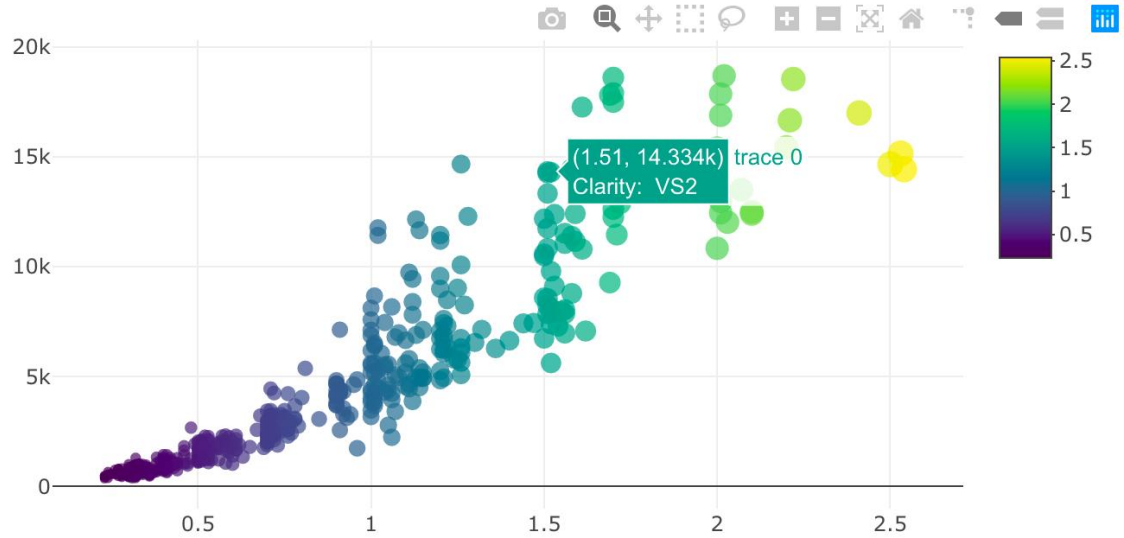


Show 5 entries Search:

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa

Showing 1 to 5 of 150 entries

Previous 1 2 3 4 5 … 30 Next

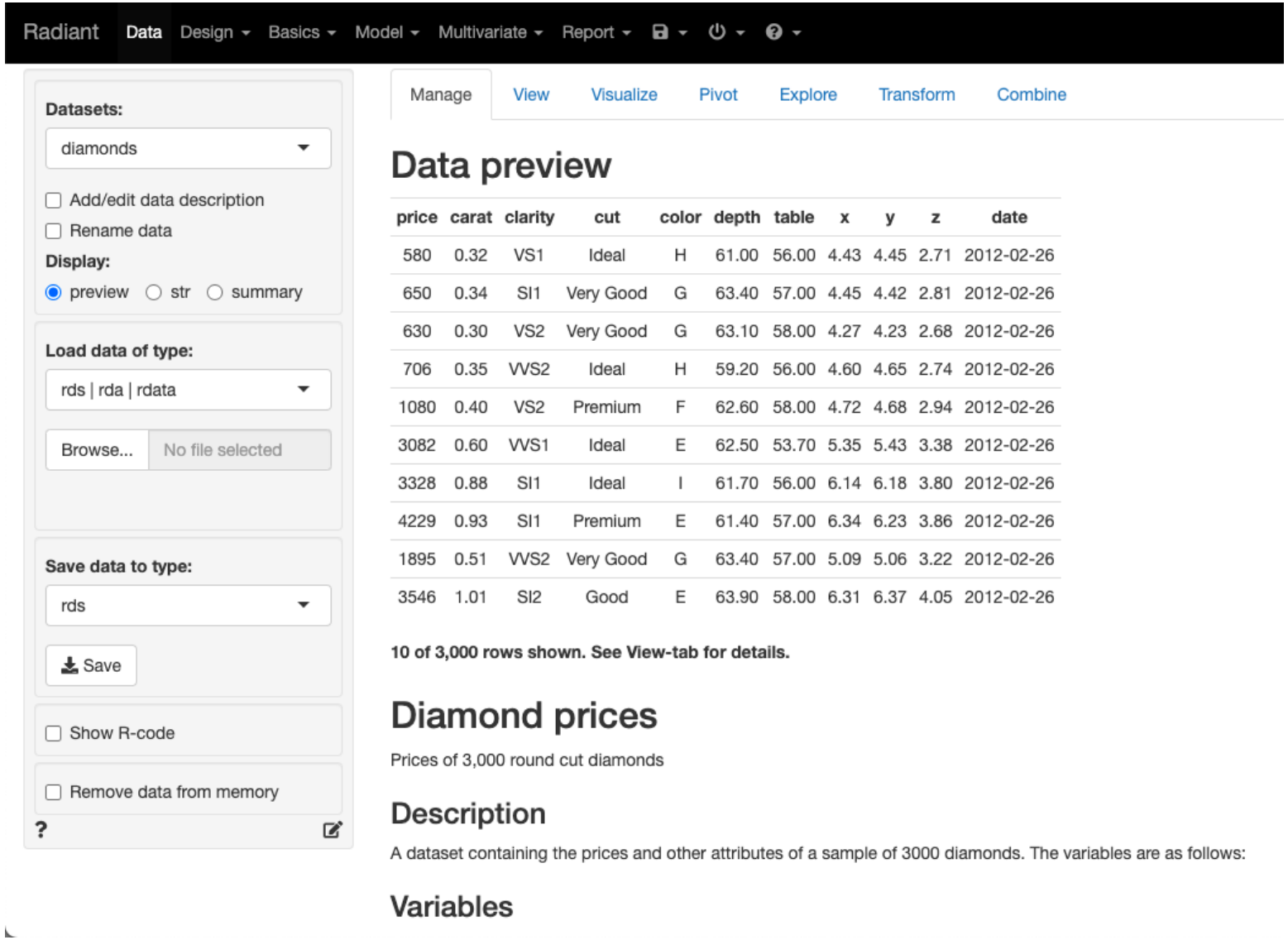


https://www.htmlwidgets.org/

http://gallery.htmlwidgets.org/

Responding to user input

- Generalized workflows
- Custom subsetting
- Changing parameters
- Personalizing output



https://shiny.rstudio.com/

https://shiny.rstudio.com/gallery/radiant.html

Interactive components

Why make charts interactive?

- Easier for data exploration
  - Drill-down to data subsets of interest
  - Details on demand
  - Customize look-and-feel of chart
- Can be more engaging for users

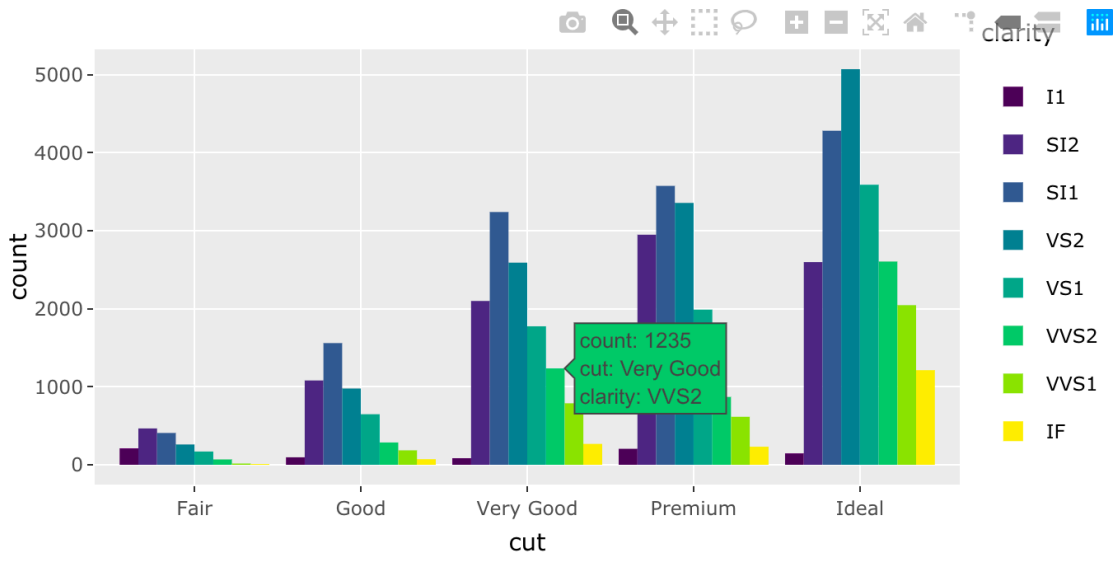
Visual information seeking mantra

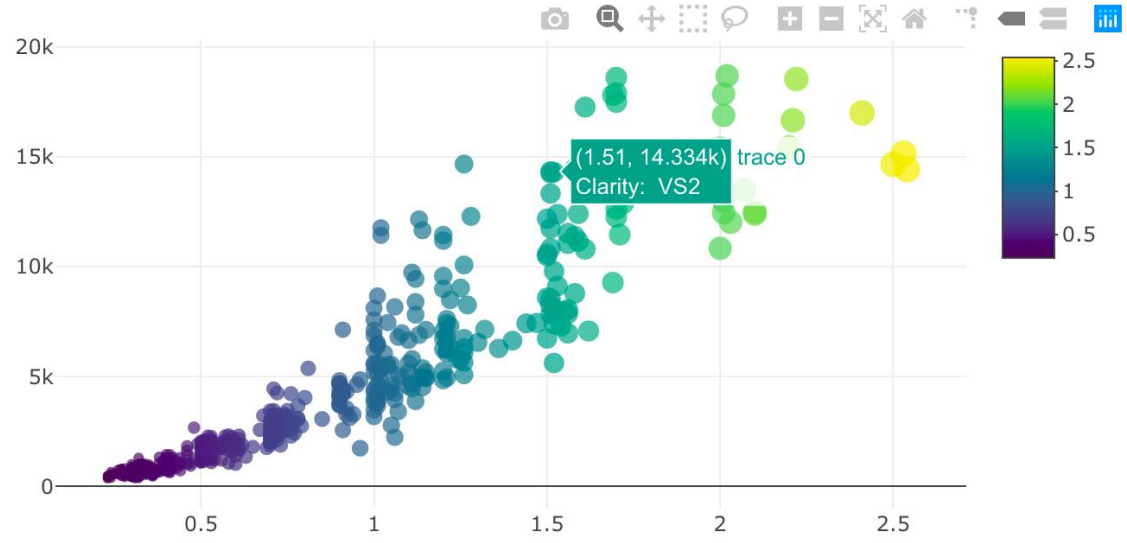
Overview first,
zoom and filter,
then details-on-demand

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualization. In *VL '96 Proceedings of the 1996 IEEE Symposium on Visual Languages.*

Interactivity in R Markdown

- R Markdown gets compiled into HTML
- Some R packages can create interactive elements by converting R output to HTML/JavaScript code in the final document
- We will use the **plotly** package to create interactive charts





[HTML widgets]

Other interactive chart packages

- ggiraph for extending ggplot2 with interactive geoms
- rCharts for an R version of Polycharts, NVD3, and MorrisJS
- rBokeh for an R version of Bokeh
- altair for an R version of Altair
- leaflet for interactive maps

Exercise 4: Make ggplot2 charts interactive

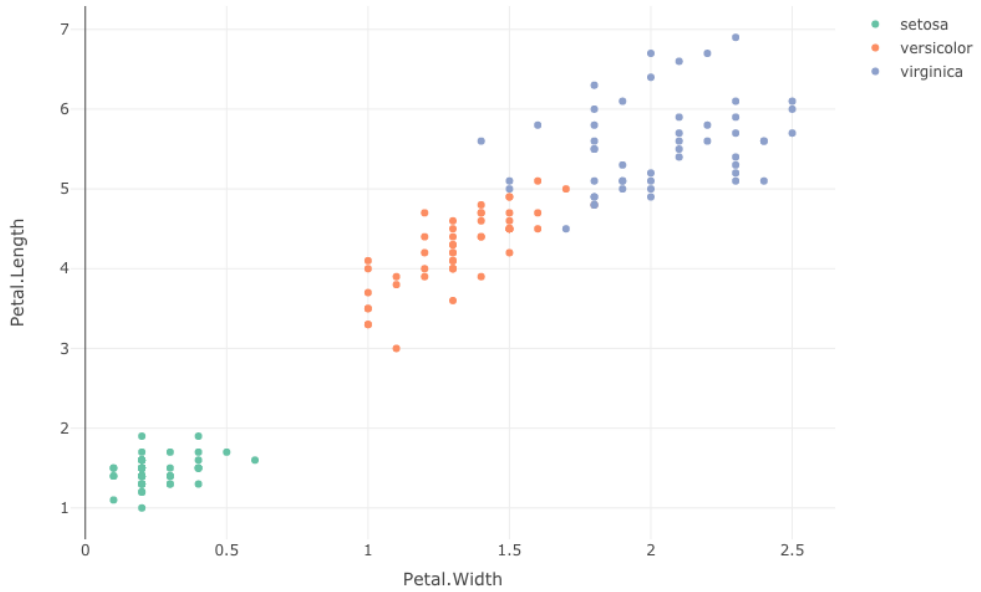
plotly

- Create plots that are interactive right away, either in R Markdown or in a website version
- Can either convert ggplot2 charts to plotly or build natively with plotly syntax

Basic plotly syntax

- Main plot function: plot_ly()
- Set the data: data = [data name]
- No aesthetics function, just list aesthetics pairings
- For each variable name, need "~" in front
- Default plot type is scatter; for others, add: type = "[plot type]"

```
plot_ly(data = iris,
    x = ~Petal.Width,
    y = ~Petal.Length,
    color = ~Species,
    type="scatter")
```



[Plotly for R Reference]

Publishing interactive plotly charts

- Write R Markdown in RStudio
- Make sure “output” at top is “html_document”
- Use knitr to knit to HTML
- Publish HTML to:
  - RPubs (click the “Publish” button in RStudio)
  - GitHub (setup a GitHub Pages repository and add the HTML files)
  - Any website you already have that can publish HTML

Afternoon Break

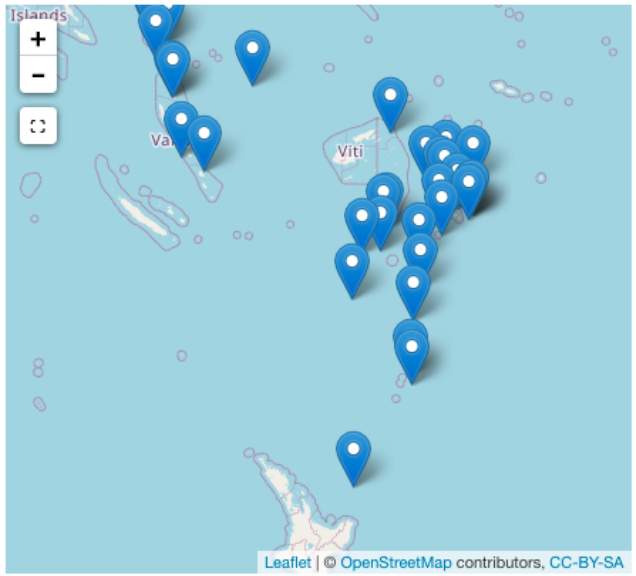
DT for interactive data tables

Coordinated Views

Views that share data

- Each view should be relatively simple, have a specific purpose
- Views can work together to explore complex interactions
- The **Crosstalk** package connects interactive components together

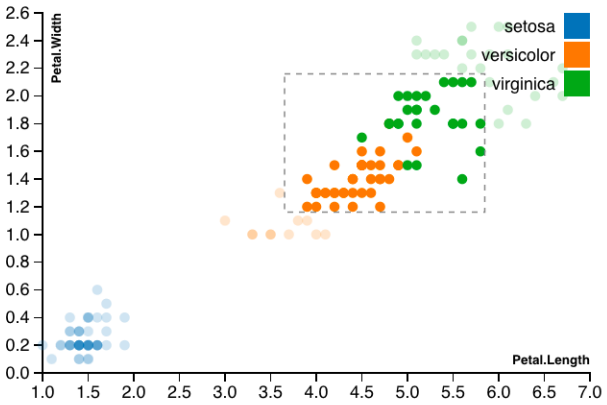
[crosstalk package]

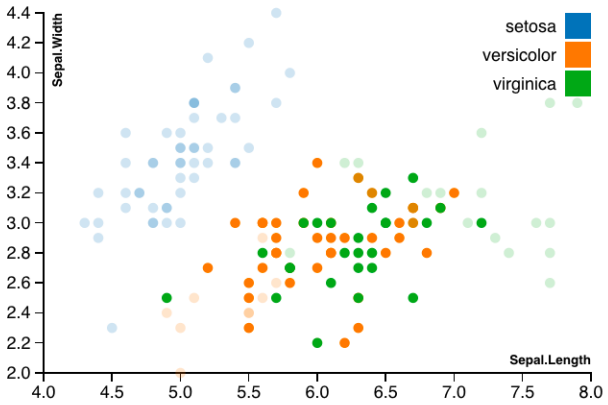


Search:

	lat	long	depth	mag	stations
308	-22	180.53	583	4.9	20
873	-11.02	167.01	62	4.9	36
277	-23.33	180.18	528	5	59
752	-21.29	185.77	57	5.3	69
352	-12.01	166.29	59	4.9	27
354	-30.17	182.02	56	5.5	68
168	-19.89	183.84	244	5.3	73
474	-10.79	166.06	142	5	40
338	-27.19	182.18	69	5.4	68

Showing 1 to 10 of 32 entries (filtered from 100 total entries)

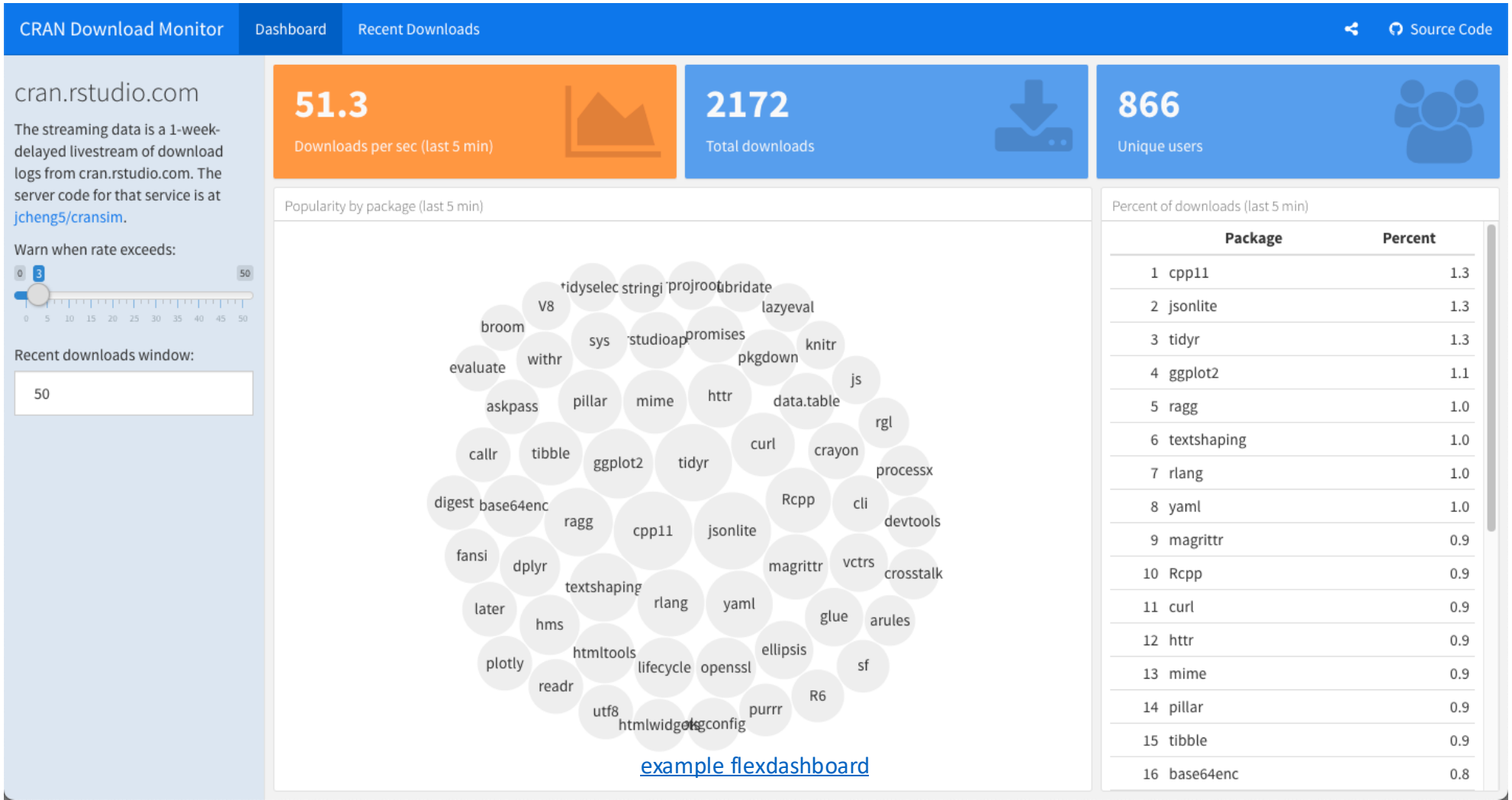




Combining interactive components with Crosstalk

Dashboards in R Markdown

What is a dashboard?



example flexdashboard

“Normal” R Markdown

- R Markdown elements like headings, text

```
# Heading 1
## Heading 2
Regular text
* Bulleted text
```

Note: Comments work like HTML
<!--HTML Comment style -->

- Code chunks

```
```{r}

```
```

Markdown for flexdashboards

Page

==========

Column (or Row)

Chart titles

Regular text
* Bulleted text

```{r}

```

Arrange various elements in flexdashboard

Using R Markdown for Slides

Several slide packages built into RStudio

Markdown Presentations

- File → New file → R Markdown…
- Select “Presentation”, then HTML (ioslides) or HTML (Slidy)

Alternate: Quarto Presentations

- File → New file → Quarto Presentation
- Under “Presentation”, select Reveal JS

Markdown for slides

Section header

New slide

--- (to start a new slide with no title)

Use standard R markdown for slide content:

Regular text
* Bulleted text
- Bulleted text
![image caption](image path)

```{r}

```

Presentation Resources

- ioslides Options
- Slidy Options
- Revealjs Options for Quarto

Using R Markdown for Websites

Using GitHub to host a website

GitHub Pages is a feature available on any GitHub repository.

GitHub Pages will look for HTML and markdown files in your GitHub repository and display them like a normal website.

Setup a repository with GitHub Pages

- Create a new GitHub repository (or start with an existing one)
- Make sure there is already a README.md file
- Click on “Settings”
- Look for “Pages” on the left
- “Source” can be “Deploy from a branch”
- Under “Branch,” change from “None” to “main”
- Leave the folder on “root” unless you plan to create a “docs” folder to store all of your pages
- Click Save

Upload website files into the repository

- You can upload .html files or .md files (or both) and GitHub should be able to display both
- Can’t just use the original .Rmd file, though. Want to knit those to HTML, then upload the HTML.
- The README file will normally be the main page people see when they go to the site. You can edit that file to include links to the other pages you have uploaded.

Example:

- https://github.com/amzoss/Apr2021VizTell (code/file view)
- https://www.angelazoss.com/Apr2021VizTell (website view)

What’s the URL of my website?

It should show up on the Pages section of Settings. Usually:

username.github.io/repository-name

Can also display it publicly:

- Go back to the “Code” view of the repository
- On the right, next to “About” click on the gear
- Under website, click the checkbox next to “Use your GitHub Pages website”

What’s the URL for the uploaded files?

- Add the filename (with extension) to the end of the normal URL

  username.github.io/repository-name/file-name.html

- For example, if you upload “flexdashboard.html” into your main directory, the URL would be

  username.github.io/repository-name/flexdashboard.html

Want a nicer URL? Make subdirectories.

- Instead of uploading flexdashboard.html, you can change the file name to index.html and put it in a folder called flexdashboard.
- When you have a file named index.html, you don’t need the filename in the URL.
- For example, if you put a file called “index.html” into a directory called “cool-project”, the URL to view that file would be:

username.github.io/repository-name/cool-project

Thanks for your time this week!

angela.zoss@duke.edu