*Angela M. Zoss, Ph.D.*

# *Visualization for Data Science with R*

To my family.

I'm so grateful for your support.

# *Contents*

**List of Tables** **v**

**List of Figures** **vii**

**Proposal** **ix**

**About the Author** **xiii**

**1 Overview of common visualizations and how to read them** **1**

1.1 Visualization components . . . . . 1

1.2 Bar Chart . . . . . 2

1.2.1 Variations . . . . . 5

1.3 Scatter Plot . . . . . 11

1.3.1 Variations . . . . . 11

1.4 Line Chart . . . . . 11

1.5 Pie Chart . . . . . 13

1.6 Heat Map . . . . . 13

1.7 Histogram . . . . . 14

1.8 Box Plot . . . . . 15

1.9 Maps . . . . . 16

1.10 Movies . . . . . 18

**2 Building basic visualizations with ggplot2** **23**

2.1 Basic ggplot2 syntax . . . . . 23

**3 Working with textual data in ggplot2** **25**

**4 Customizing the design of ggplot2 visualizations 27**

**5 Avoiding unethical design practices 29**

**6 Building ggplot2 visualizations into print publications 31**

**7 Basic accessibility for static visualizations 33**

7.1 Low Vision . . . . . 33

7.2 Color Vision Deficiency . . . . . 33

7.2.1 Dual encoding (never just color) . . . . . 33

7.2.2 Color palettes . . . . . 33

7.3 Alternative Text for Screen Readers . . . . . 34

7.4 Converting graphics to sound, touch, text . . . . . 34

7.5 Accessibility Resources . . . . . 34

**8 Exploring interactivity in visualizations with plotly and crosstalk 37**

**9 Using RMarkdown to build websites for projects 39**

**10 Using RMarkdown to build dashboards for projects 41**

**11 Basic usability for interactive visualizations 43**

**12 Teacher's guide 45**

**Appendix 47**

**A Datasets 47**

A.1 Bar Chart . . . . . 47

**Bibliography 69**

**Index 71**

. . . . . 71

# *List of Tables*

A.1 A sample from the Duke Enrollment By School dataset. . . . . 47

# *List of Figures*


1 Angela M. Zoss, Ph.D. . . . . xiii

1.1 Visualization components, labeled. . . . . 2
1.2 A sample bar chart. . . . . 3
1.3 A sample bar chart with both positive and negative values. . . 4
1.4 A sample bar chart, with the bars oriented horizonally. . . . . 4
1.5 A sample lollipop plot. . . . . 5
1.6 A sample lollipop plot with both positive and negative values. . 6
1.7 A sample stacked bar chart. . . . . 7
1.8 A sample filled stacked bar chart. . . . . 7
1.9 A sample grouped bar chart. . . . . 8
1.10 A grouped bar chart focusing on two continents. . . . . 9
1.11 The same grouped bar chart with a different arrangement of the categorical variables. . . . . 9
1.12 A dumbbell plot comparing two continents. . . . . 10

A.1 Total Duke Enrollment by School . . . . . 48
A.2 Log of tissue loss by snail density . . . . . 49

# *Proposal*

**Note: This book is a work in progress, with a full draft expected in April of 2022.**

This book combines instruction on writing R code with building basic graphic design skills in a way that is unusual in data science literature. The book will guide readers through a series of projects, each designed to cover both how visualizations work in R and how visualizations can be designed to have the greatest impact. Far more than a "do this, then this" checklist, this book will focus on building understanding, confidence, and the ability to transfer skills to other tools and design contexts. It will avoid technical jargon that our target audience is unlikely to have encountered before. To accommodate learners who don't have time to work through an entire book, each chapter will operate independently, covering a specific set of tasks that all make sense together as part of a visualization project. For those who would like extra practice, there will be several types of hands-on exercises, from those that are entirely prescribed to those that allow readers to apply new techniques to problems in their own areas.

The book will have solutions (in the form of completed code and sample output) for all exercises. While not a textbook, the book will also include a brief teacher's guide for courses that might want to use one or more chapters to structure lessons in a course. The book will also have a website, including links to Open Access content, solutions, and related resources like video tutorials.

The target audience of this book would be professionals who are having to learn data science techniques on the job, likely at an under-resourced organization or company. These newly minted data professionals may feel comfortable in Excel but have only just started to learn R for processing data. They have never used a programming language to build a visualization before, and even creating charts in Excel has often been a frustrating and mystifying process. They appreciate that R is freely available and are able to get started on a data science project, but the idea of creating publication-quality visualizations using only code is daunting.

Increasingly, programs of study with a focus on preparing students for professional careers in under-resourced fields, like public policy and even management, include courses on data analysis and communication using freely available software. This book, while not a textbook, could easily be used for a semester-long course, titled something like "Practical data visualization for

the modern workforce." A chapter could be covered each week, and larger projects could help learners synthesize chapters into a complete set of analyses and communication materials.

## Why read this book

This book will be:

- Written for non-academics, beginning programmers
- Each chapter stands alone
- Covers pressing modern issues, like accessibility and ethics
- Focuses on freely available software
- Combines hands-on exercises with basic graphic design principles

## Structure of the book

- Chapter 1: Overview of common visualizations and how to read them
- Chapter 2: Building basic visualizations with ggplot2
- Chapter 3: Working with textual data in ggplot2
- Chapter 4: Customizing the design of ggplot2 visualizations
- Chapter 5: Avoiding unethical design practices
- Chapter 6: Building ggplot2 visualizations into print publications
- Chapter 7: Basic accessibility for static visualizations
- Chapter 8: Exploring interactivity in visualizations with plotly and crosstalk
- Chapter 9: Using RMarkdown to build websites for projects
- Chapter 10: Using RMarkdown to build dashboards for projects
- Chapter 11: Basic usability for interactive visualizations
- Chapter 12: Teacher's guide

## Software information and conventions

I used the **knitr** package (Xie, 2015) and the **bookdown** package (Xie, 2021) to compile my book. My R session information is shown below:

```
xfun::session_info()
```

```
## R version 4.1.0 (2021-05-18)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Locale: en_US.UTF-8 / en_US.UTF-8 / en_US.UTF-8 / C / en_US.UTF-
8 / en_US.UTF-8
##
## Package version:
##   base64enc_0.1.3   bookdown_0.23
##   compiler_4.1.0    digest_0.6.27
##   evaluate_0.14     glue_1.4.2
##   graphics_4.1.0    grDevices_4.1.0
##   highr_0.9         htmltools_0.5.1.1
##   jquerylib_0.1.4   jsonlite_1.7.2
##   knitr_1.33        magrittr_2.0.1
##   markdown_1.1      methods_4.1.0
##   mime_0.11         rlang_0.4.11
##   rmarkdown_2.10    rstudioapi_0.13
##   stats_4.1.0       stringi_1.7.3
##   stringr_1.4.0     tinytex_0.33
##   tools_4.1.0       utils_4.1.0
##   xfun_0.25         yaml_2.2.1
```

Package names are in bold text (e.g., **rmarkdown**), and inline code and filenames are formatted in a typewriter font (e.g., `knitr::knit('foo.Rmd')`). Function names are followed by parentheses (e.g., `bookdown::render_book()`).

Angela Zoss

# *About the Author*

**FIGURE 1:** Angela M. Zoss, Ph.D.

Angela is the Assessment & Data Visualization Analyst[1] in the Assessment & User Experience Department[2] in the Duke University Libraries[3]. She has many years of experience in teaching and training, predominantly focusing on teaching data visualization to university students, faculty, and staff. She is also active in several open source development projects, including FOLIO[4] and Wax[5].

[1] https://library.duke.edu/about/directory/staff/angela.zoss
[2] https://library.duke.edu/about/depts/assessment-user-experience
[3] https://library.duke.edu/
[4] https://github.com/folio-org/
[5] https://github.com/minicomp/wax

# 1

# *Overview of common visualizations and how to read them*

This book will cover how to create a variety of visualizations using R. One of the first things you should do to improve your skills creating visualizations is to become familiar with the kinds of visualizations that are possible and the different features of each.

Effective visualization design relies on a solid understanding of how data properties, visualization types, and audience characteristics interact to help people make sense of a visualization. In this chapter, we'll look at a series of common visualization types, and we'll break down how each is meant to be read. Understanding these basic visualization types will create a solid foundation for communicating your data science work to a broad audience.

## 1.1 Visualization components

As we discuss different visualizations, we will also be talking about different components within the visualizations. In figure 1.1 below, the major components of the visualization are labeled: the main title, the subtitle, the x axis title, the y axis title, the panel, the horizontal and vertical gridlines, and the axis labels and tick marks for both axes. Almost all of the visualizations we cover in this book will use these basic components.

In this set of basic visualization components, we see two components labeled as an axis. These axes are called the x and y axes, and they always appear in these positions: the x axis always goes left to right, and the y axis always goes up and down.

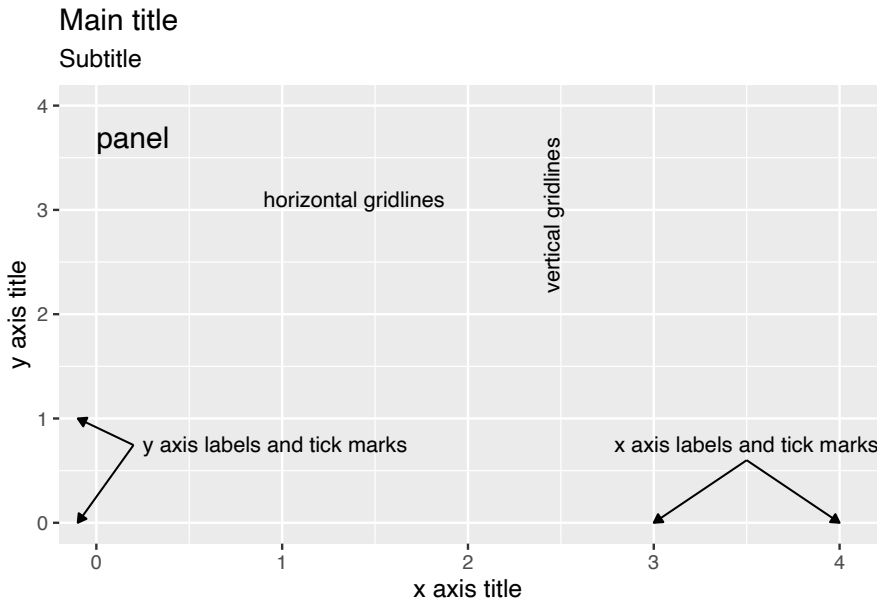



**FIGURE 1.1:** Visualization components, labeled.

## 1.2 Bar Chart

The bar chart is possibly the most common type of visualization. In this type of visualization, the basic shape being used to represent data values is a rectangle. In a traditional bar chart, each rectangle (or bar) has exactly the same width, and the height of the bar is representative of some data value. To create a simple bar chart, the data set should have one column that contains textual (or categorical) data and one column that contains numerical data. A common way to create these two columns is to start with one categorical data column and count the number of records for each category to create the numerical column.

In the sample bar chart above, the categorical variable is displayed on the x axis and the numerical variable is displayed on the y axis. This results in a classic style of bar chart where each bar has the same width and the heights are proportionate to a data value. Each bar has a starting value of zero on the y axis. Each bar travels upward from zero and stops at the correct data value.

When reading this visualization, we are comparing the lengths of bars in order to understand patterns within the numerical data values from our data set.

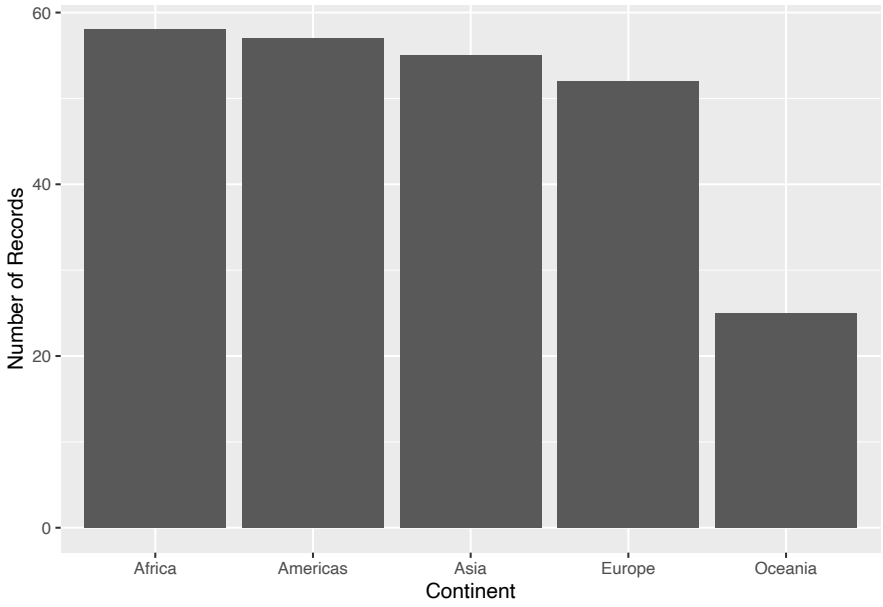



**FIGURE 1.2:** A sample bar chart.

The power of the bar chart lies in how precisely we can detect differences in the end points of the bars. This is something that people naturally do quite well when all of the bars start at the same lowest point, or baseline.

Bar charts are especially effective if the bars that have small differences in lengths appear close to each other. In the above chart, this is accomplished by arranging the bars so they appear with the highest data values on the left and the lowest data values on the right.

Here is another example of a bar chart where the data values include both positive and negative values. For data values that are negative, the bar travels downward from zero and stops at the correct data value. The x axis title and labels appear at the bottom of the panel, below the lowest data values.

For stylistic reasons, bar charts may also appear with the bars oriented horizontally instead of vertically. In that case, each bar will have the same height, and the widths of the bars will vary based on the data values. The text (or categorical) axis will then be the y axis, and the numerical axis will be the x axis.

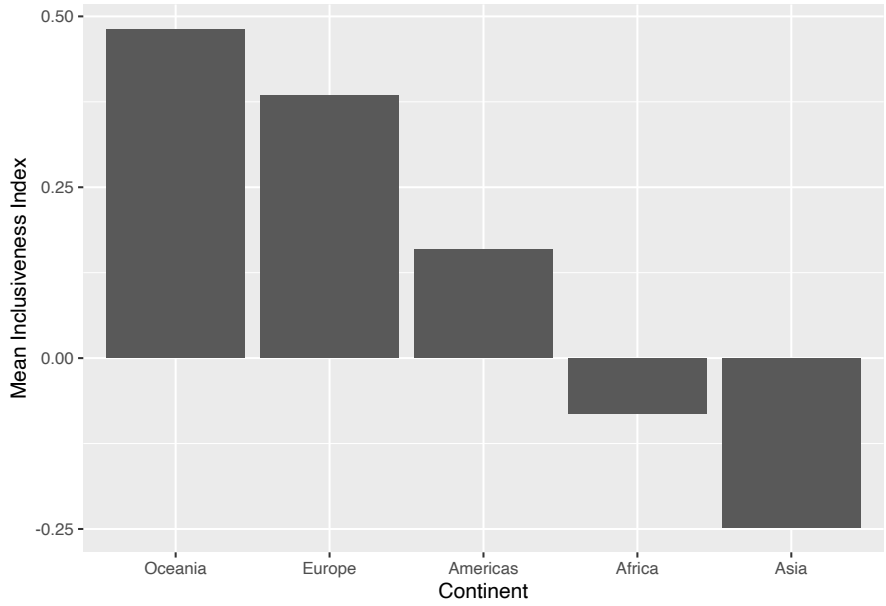



**FIGURE 1.3:** A sample bar chart with both positive and negative values.

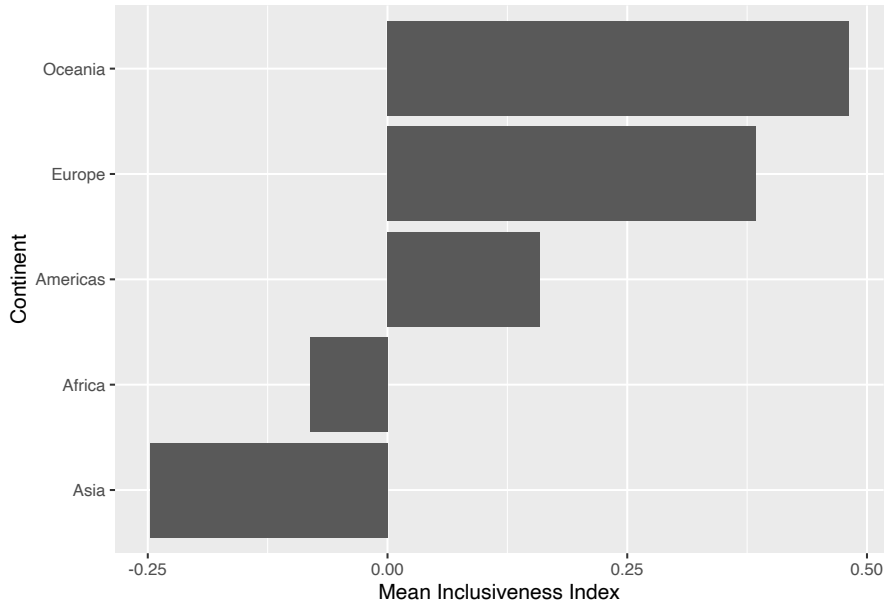



**FIGURE 1.4:** A sample bar chart, with the bars oriented horizonally.

### 1.2.1 Variations

The following charts are variations on a bar chart. They either incorporate additional variables, change the basic shape of the chart, or both.

#### 1.2.1.1 Lollipop plot

One quick variation of the bar chart is called the lollipop plot In a lollipop plot, the bars are replace by a long line with a circle at the end, creating something that looks like a lollipop. Apart from the different shapes used, the lollipop plot works just like a bar chart. The circles draw attention to the data value, but the lines extending to the axis reinforce the length comparisons.

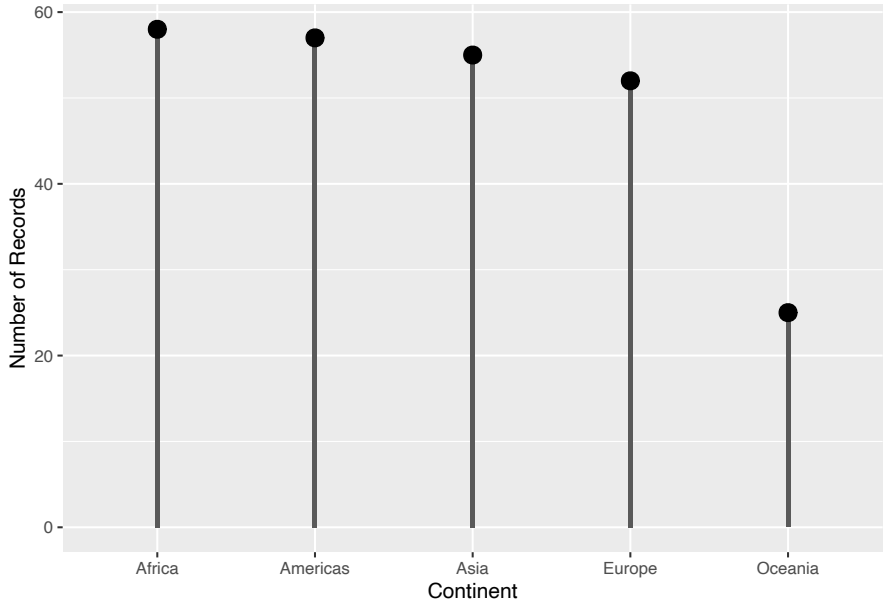



**FIGURE 1.5:** A sample lollipop plot.

#### 1.2.1.2 Bar charts with color

A simple bar chart includes one categorical variable and one numerical variable. Sometimes, however, it is useful to explore the patterns in relation to a second categorical variable. Adding another categorical variable to a bar chart usually means using color to represent the extra variable.

With a stacked bar chart, the additional variable is used to segment the bars

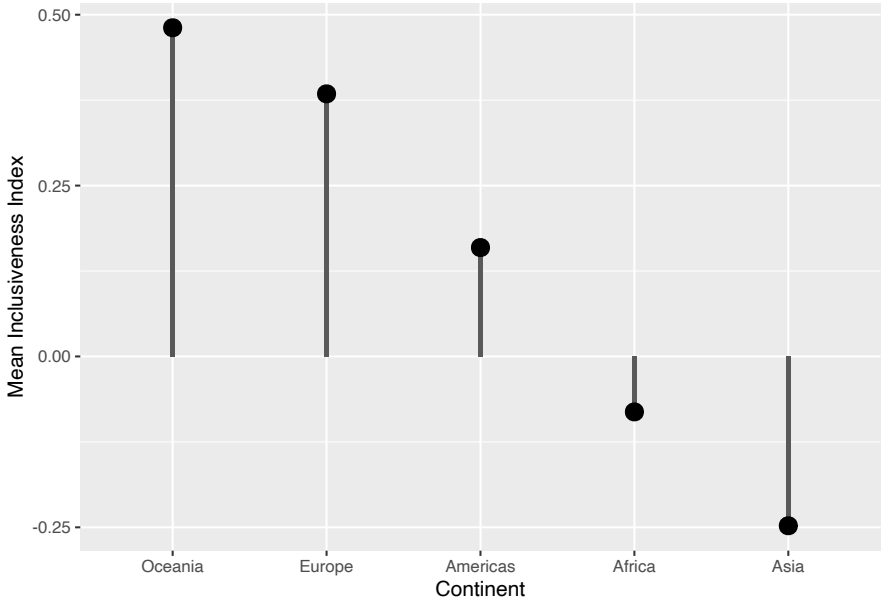



**FIGURE 1.6:** A sample lollipop plot with both positive and negative values.

into separate, colored regions. In this example, the bar for each continent is subdivided into groups of countries based on their values for Inclusive Index.

With this stacked bar chart, you can still see the total number of records for each Continent, but what happens when you try to compare the different segments inside the bar chart? Starting at the bottom, it seems to work out okay. All of the bars for the "No data" category start at the same baseline (the x axis), and we can read these segments like a normal bar chart. But what happens with the "Low" segments right above them? And the "Medium-Low" segments above those? Every time we have a group of segments that aren't lined up with each other, we have to try to guess how tall the bar is in comparison to the other bars in the group. The farther apart the segments are, the harder it is to make that comparison.

Another variation of the stacked bar chart is the "filled" stacked bar chart. Instead of using the raw counts to determine the lengths of the bars, in the filled stacked bar chart, the full bars are all stretched to have the same height, and each segment becomes the percentage of the records in each bar. (Notice how the y axis changes from "Number" to "Percentage.") This is useful if the percentages matter more than the raw counts, but it doesn't fix any of the concerns with comparing different segments without a common baseline.

An alternative to the stacked bar chart is called the grouped bar chart. In the

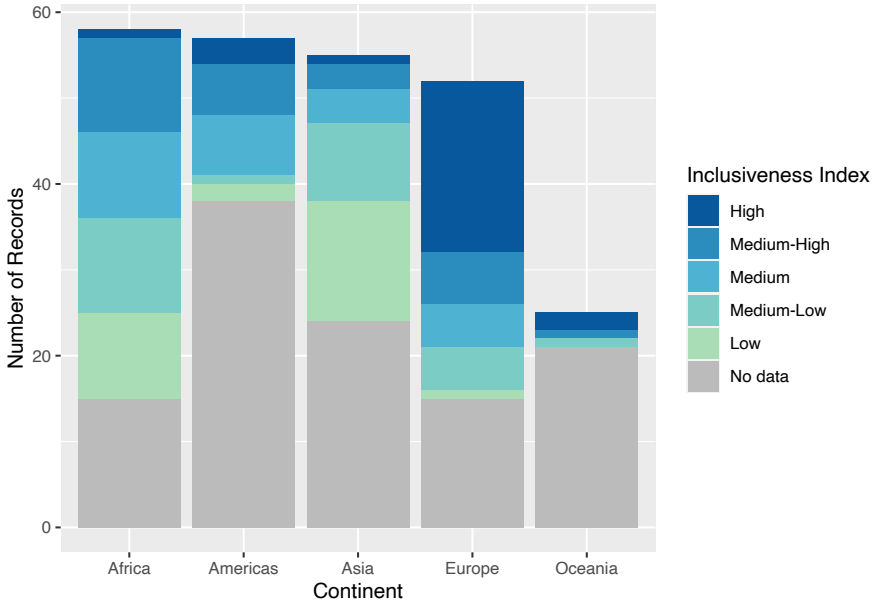



**FIGURE 1.7:** A sample stacked bar chart.

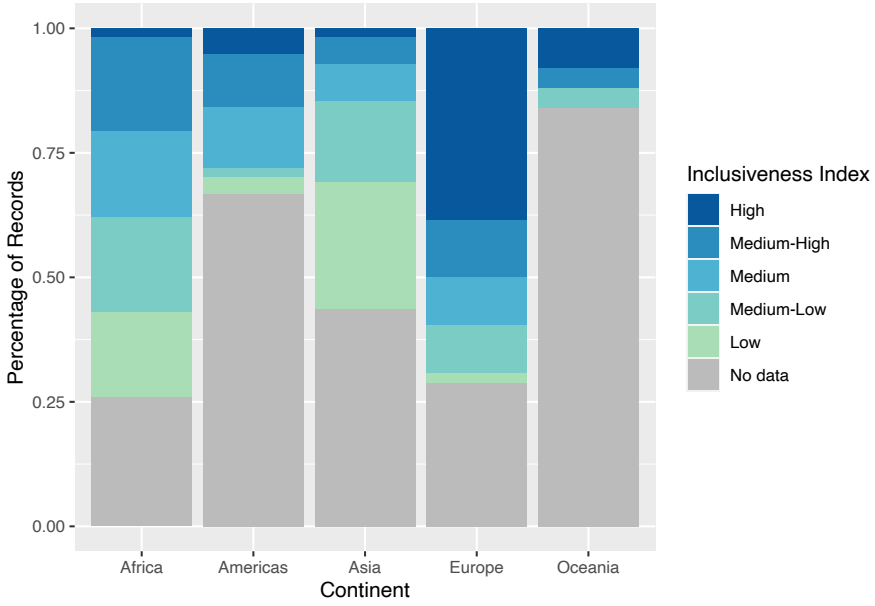



**FIGURE 1.8:** A sample filled stacked bar chart.

grouped bar chart, every segment starts from the x axis. Each continent forms a group of bars, and each option of the Inclusiveness Index is a separate bar.

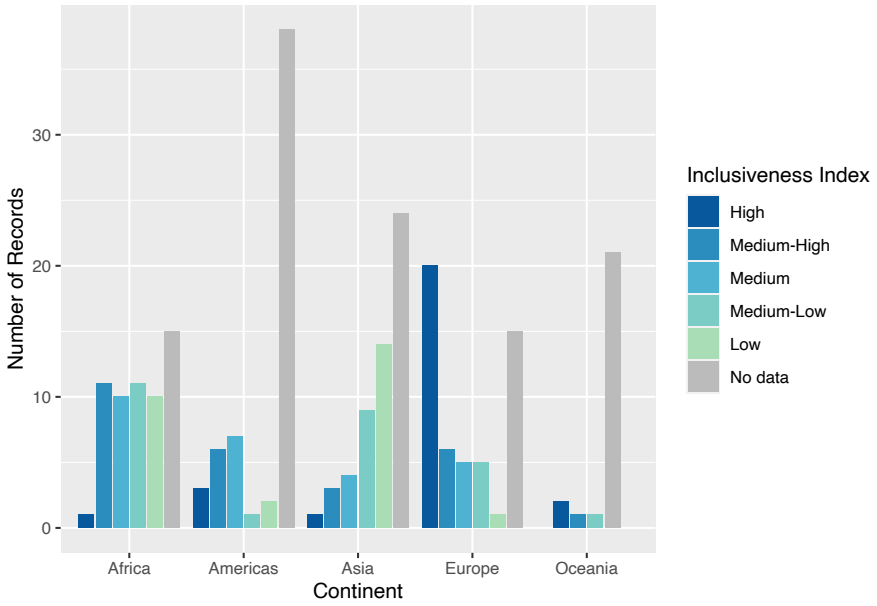



**FIGURE 1.9:** A sample grouped bar chart.

#### 1.2.1.3 Dumbbell plot

Stacked and grouped bar charts show some of the limitations of bar charts for making complex comparisons. The rectangles in the bar chart take up a large amount of space. Think back to the lollipop plot, where it's the circles that directly represent the data value. Converting bars to something like circles opens up the ability to make more direct comparisons.

For example, let's say we want to compare two continents more directly: Asia and Europe.

This chart groups all of the segments by continent, which makes it easy to compare different Index categories within a single continent. What if we want to bring more attention to the difference between continents for each category? We could always switch which category is the primary division on the x axis and which is represented by color.

This improves our ability to compare the continents directly because the bars are directly next to each other. The amount of space the bars take up is still pretty large, though. If we combine this chart with something like a lollipop plot, we get one last variation: a dumbbell plot.

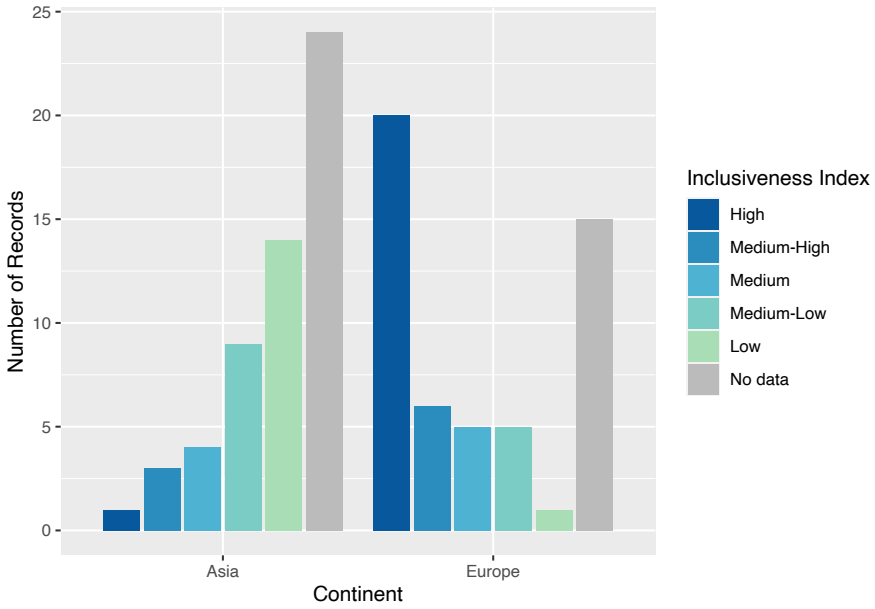



**FIGURE 1.10:** A grouped bar chart focusing on two continents.

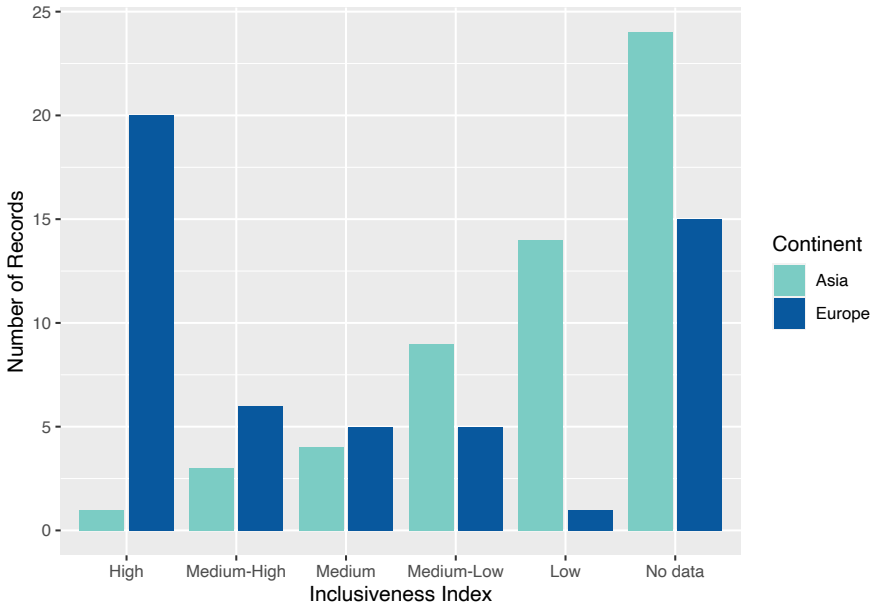



**FIGURE 1.11:** The same grouped bar chart with a different arrangement of the categorical variables.

With a dumbbell plot, we use a circle to represent the data values, just like the lollipop. Instead of having a line that extends all the way to the axis, though, we use a line to connect the two dots in each category of Inclusiveness Index.

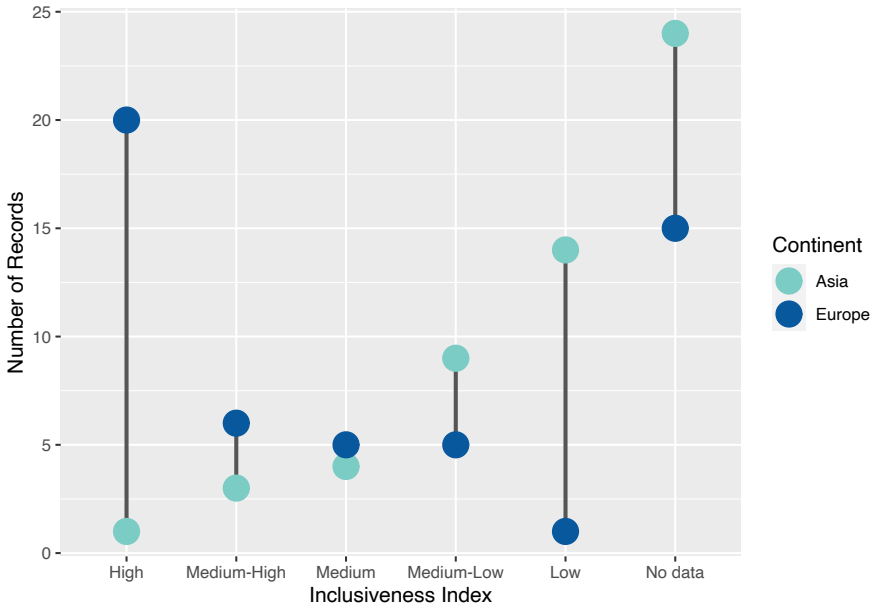



**FIGURE 1.12:** A dumbbell plot comparing two continents.

With a dumbbell plot, there are three main trends we can explore in the chart. We can focus on the green circles to see the pattern in the Asia data values. We can focus on the blue circles to see the pattern in the Europe data values. Finally, we can focus on the lengths of the lines connecting the circles to compare the continents at each level of the Inclusiveness Index. This chart time is an efficient way to compare these data values, but remember that it can be difficult to compare the lengths of shapes when they don't have the same baseline.

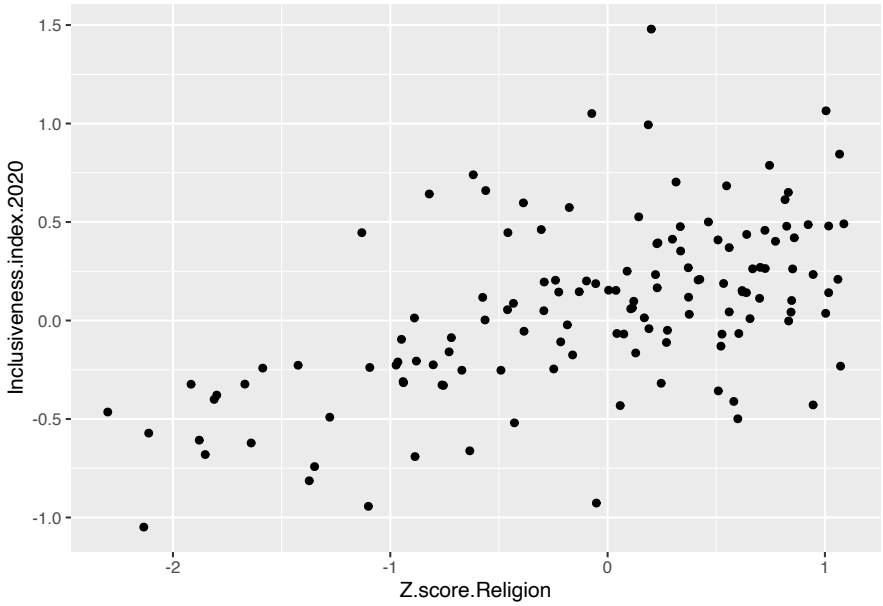
## 1.3 Scatter Plot

### 1.3.1 Variations

#### 1.3.1.1 Scatter plot with color

#### 1.3.1.2 Bubble chart

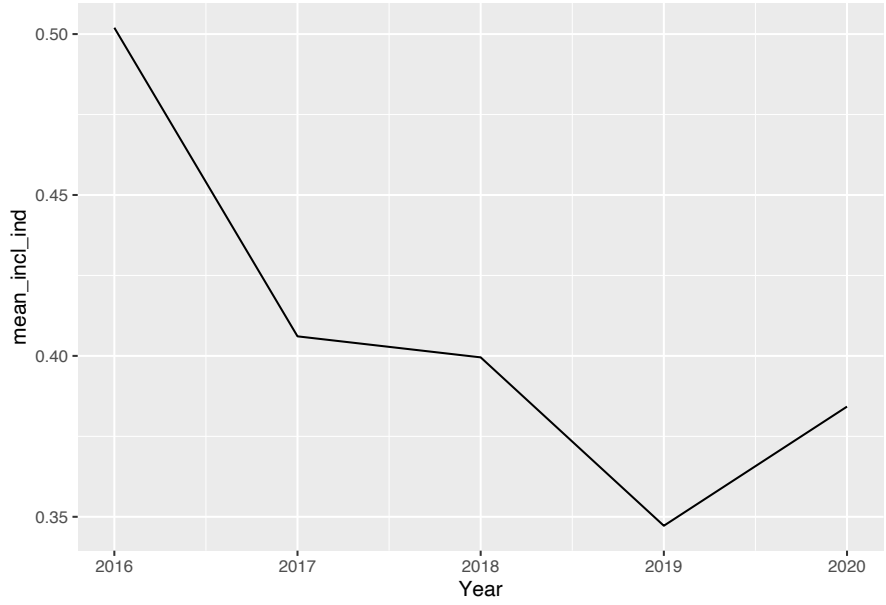
#### 1.3.1.3 Countour plot



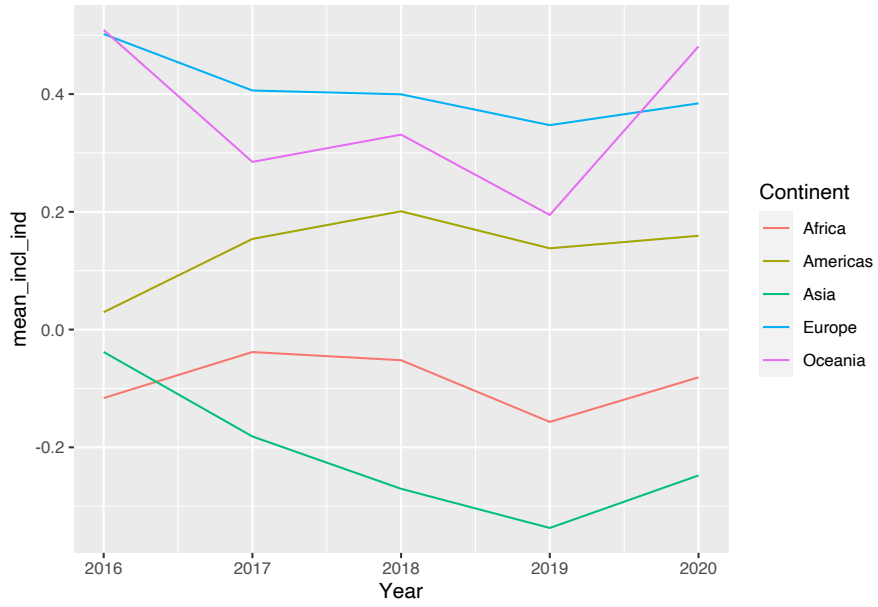

## 1.4 Line Chart

line chart / area chart

```
## `summarise()` has grouped output by 'Year'. You can override using the `.groups` argument.
```

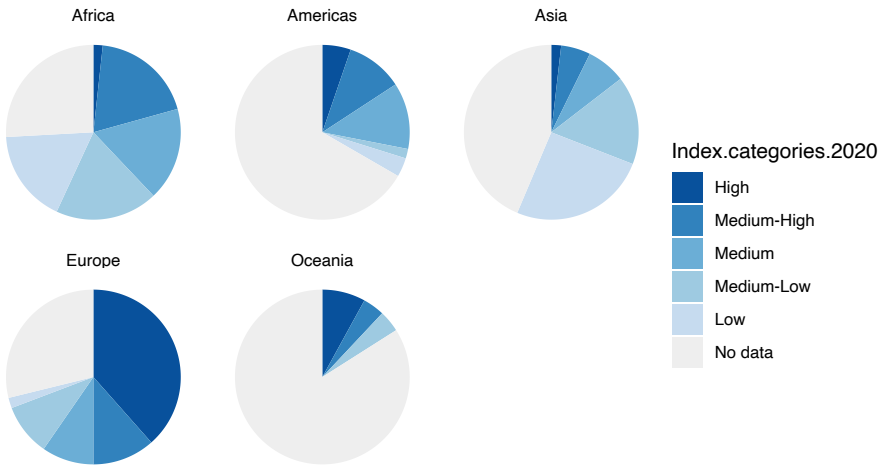



```
## `summarise()` has grouped output by 'Year'. You can override using the `.groups` argument.
```



## 1.5 Pie Chart

pie chart / donut chart

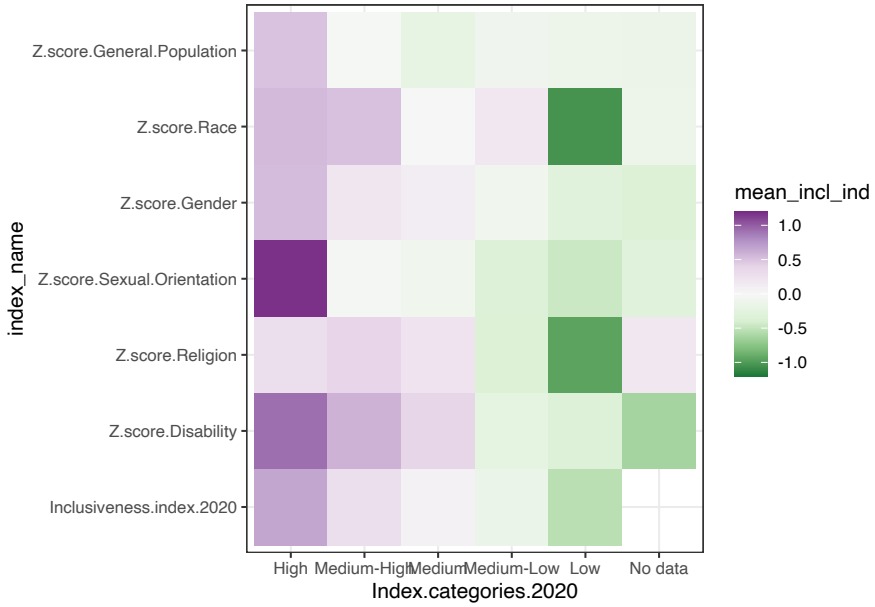



## 1.6 Heat Map

heat map / matrix / circles with color and size

```
## `summarise()` has grouped output by 'Index.categories.2020'. You can override using the `.groups` ar
```

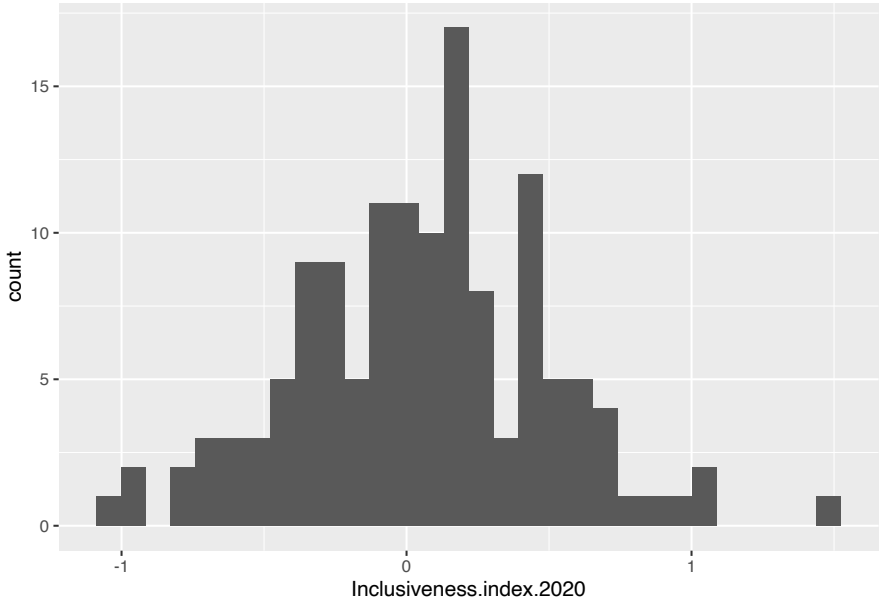
```
## Warning: Unknown levels in `f`:
## Inclusiveness.index.2020
```




## 1.7 Histogram

histogram / density

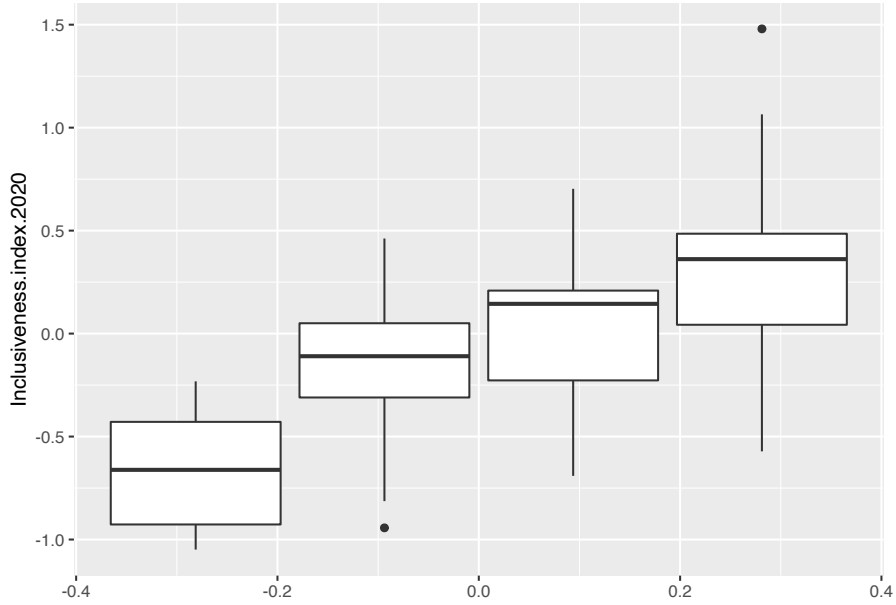
```
## `stat_bin()` using `bins = 30`. Pick better value
## with `binwidth`.
```




## 1.8 Box Plot

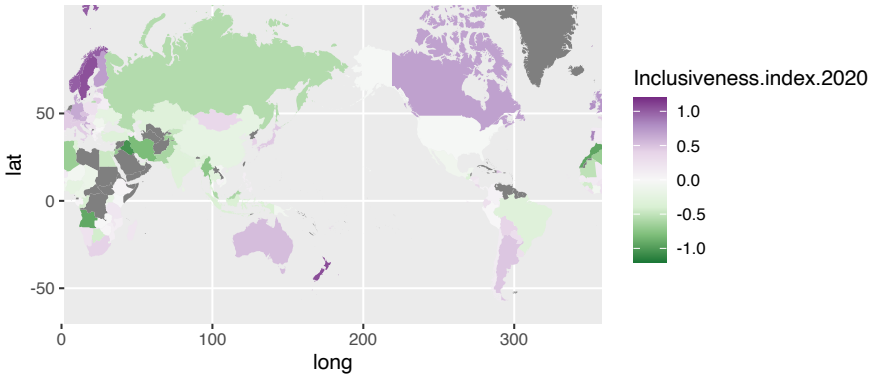
box plot / violin plot / bee swarm

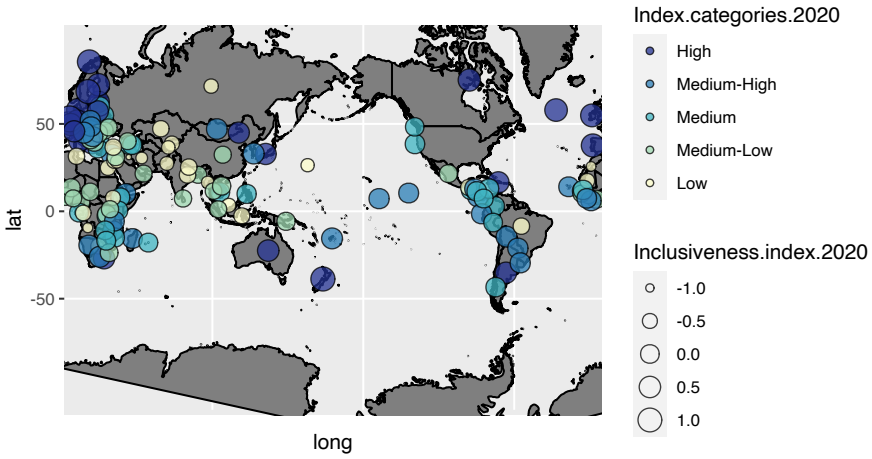
```
## Warning: Removed 113 rows containing non-finite values
## (stat_boxplot).
```



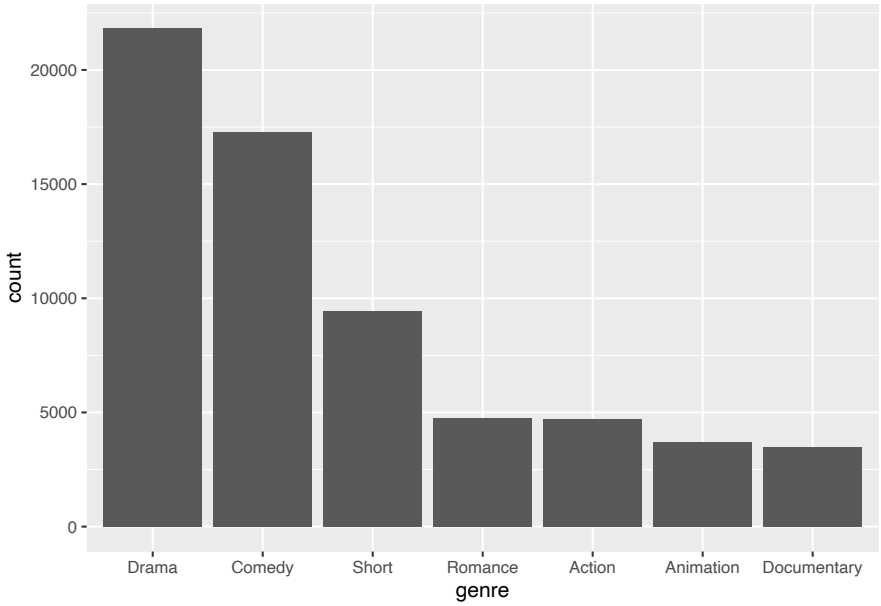

## 1.9 Maps

choropleth / proportional symbol map

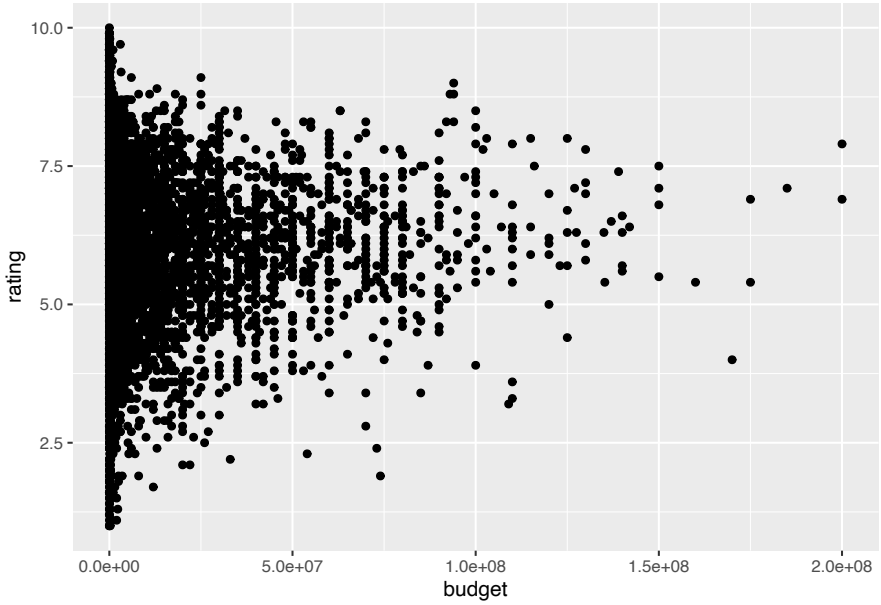

lat
50
0
-50
0
100
200
300
long
Inclusiveness.index.2020
1.0
0.5
0.0
-0.5
-1.0


lat
50
0
-50
long
Index.categories.2020
High
Medium-High
Medium
Medium-Low
Low
Inclusiveness.index.2020
-1.0
-0.5
0.0
0.5
1.0

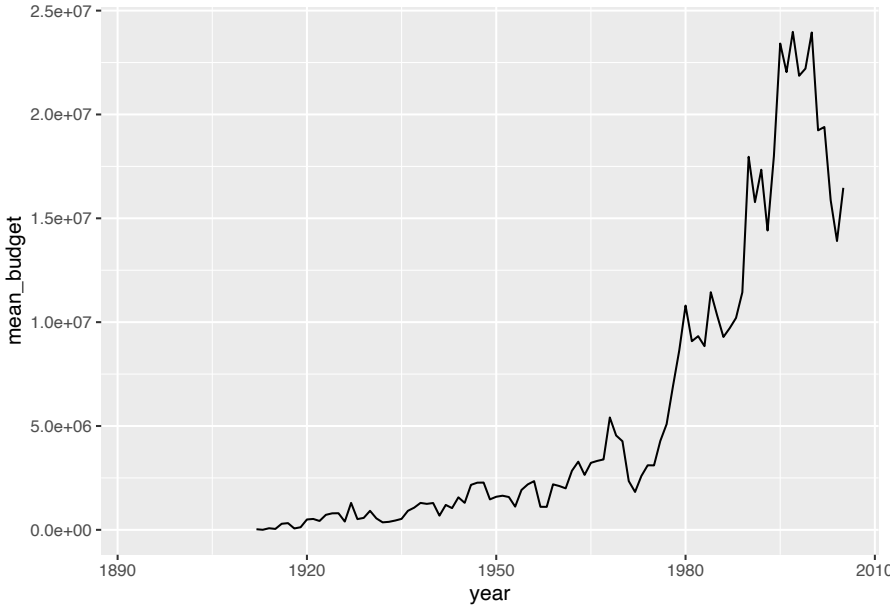
## 1.10 Movies

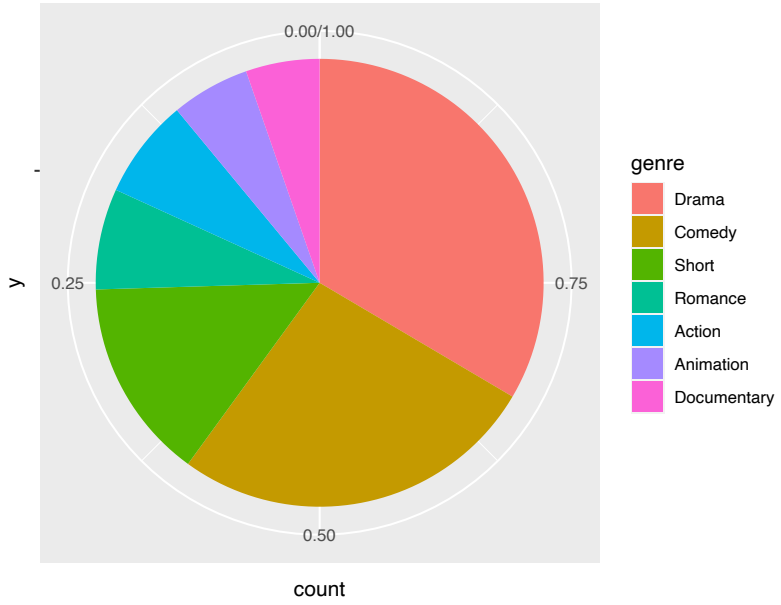



```
## Warning: Removed 53573 rows containing missing values
## (geom_point).
```

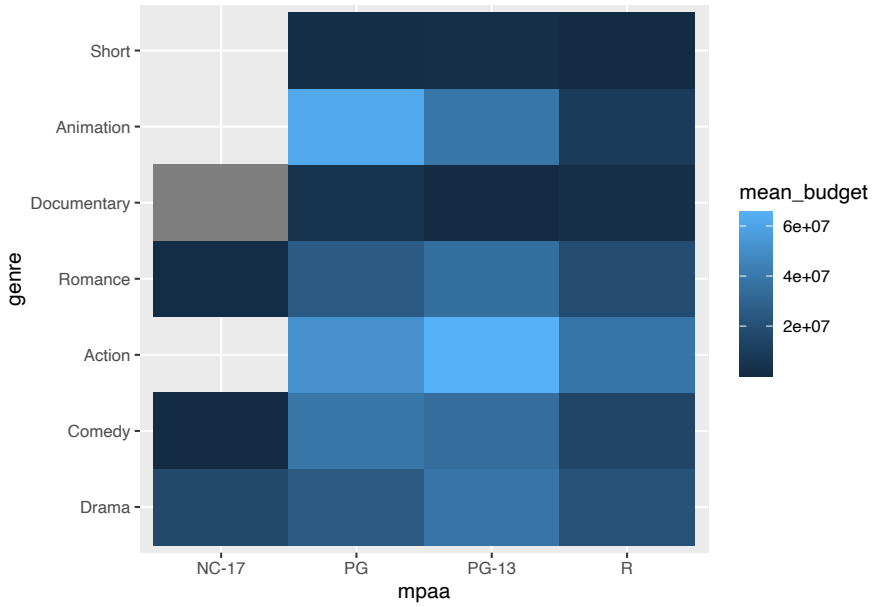



```
## Warning: Removed 10 row(s) containing missing values
## (geom_path).
```

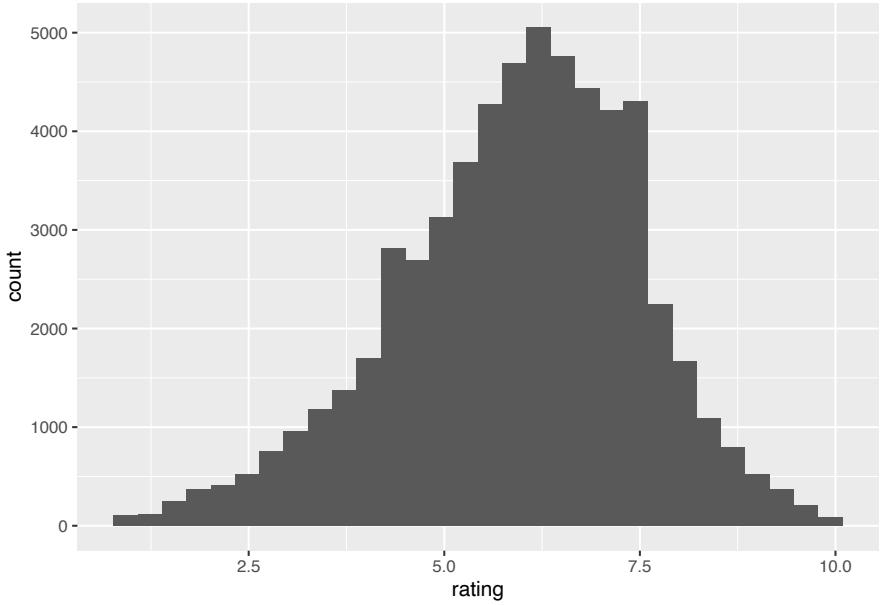


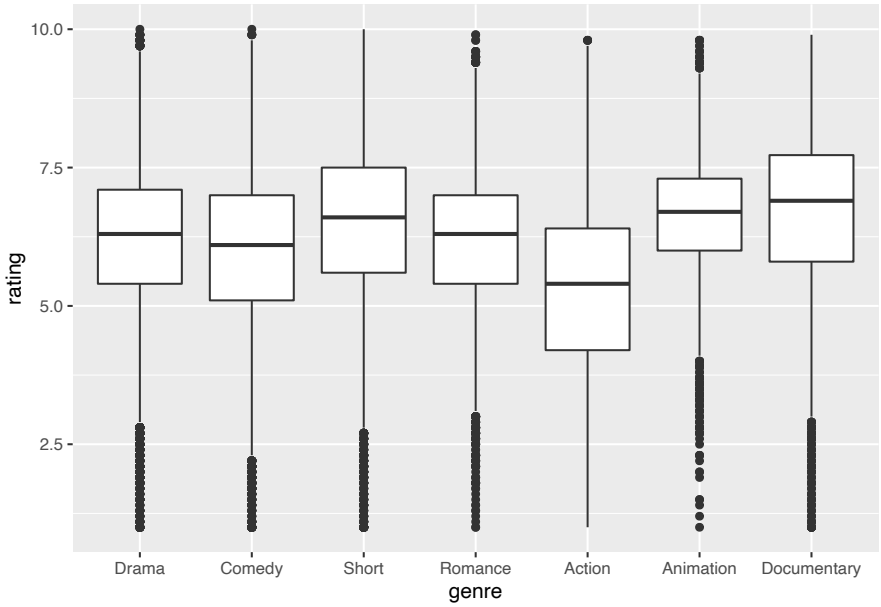



```
## `summarise()` has grouped output by 'mpaa'. You can override using the `.groups` argument.
```




```
## `stat_bin()` using `bins = 30`. Pick better value
## with `binwidth`.
```



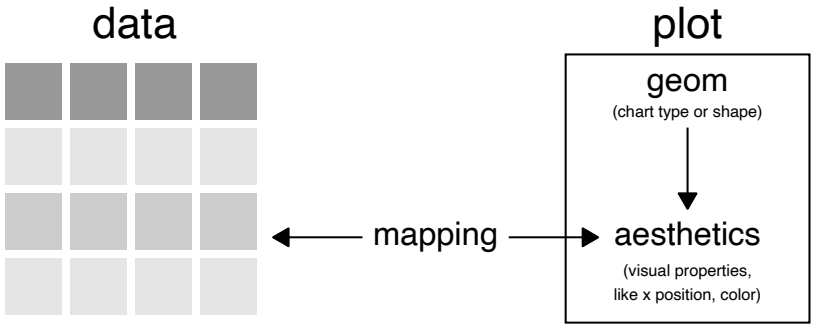




(No maps, but maybe that's okay)

# 2

# *Building basic visualizations with ggplot2*

## 2.1 Basic ggplot2 syntax



# 3

# *Working with textual data in ggplot2*

*sample text*

Cleaning data: use duke_enrollment (either by status or school) to talk about factors. Have Semester, which is really a time-based variable. Need to combine with Year to get the real sequence of enrollment.

# 4

# *Customizing the design of ggplot2 visualizations*

*sample text*

We talk about the *FOO* method in this chapter.

# 5

# *Avoiding unethical design practices*

*sample text*

We talk about the *FOO* method in this chapter.

# 6

# *Building ggplot2 visualizations into print publications*

*sample text*

We talk about the *FOO* method in this chapter.

# 7

# *Basic accessibility for static visualizations*

## 7.1 Low Vision

- Large text
  - "output-examples" file[1]
- High color contrast
  - Both marks/text on background and labels on marks
  - Check with savonliquide package[2]

## 7.2 Color Vision Deficiency

### 7.2.1 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

### 7.2.2 Color palettes

- colorspace package[3]

[1] https://github.com/amzoss/RVis-2Day/blob/master/Day%201/templates/output-examples.md

[2] https://github.com/feddelegrand7/savonliquide

[3] http://colorspace.r-forge.r-project.org/index.html

## 7.3 Alternative Text for Screen Readers

In R, R Markdown:

- fig.alt[4] in code chunk (new, just for HTML output)
- fig.cap[5] in code chunk as backup
- embedded images: write alt text between square brackets
- New: ggplot2 v3.3.4 adds alt option in labs()[6], with plans to propagate to Rmd, Shiny

Writing good alt text for visualizations[7]

Longer descriptions: savonliquide package[8]

## 7.4 Converting graphics to sound, touch, text

- sonify package
- tactileR package
- BrailleR package[9]
  - Note: set plot title, subtitle, caption using labs()

Accessible Data Science for the Blind Using R[10]

## 7.5 Accessibility Resources

- savonliquide package[11]

[4] https://blog.rstudio.com/2021/04/20/knitr-fig-alt/
[5] https://bookdown.org/yihui/rmarkdown/r-code.html
[6] https://ggplot2.tidyverse.org/reference/labs.html
[7] https://nightingaledvs.com/writing-alt-text-for-data-visualization/
[8] https://github.com/feddelegrand7/savonliquide
[9] https://r-resources.massey.ac.nz/BrailleRInAction/GGPlot.html
[10] https://jooyoungseo.com/post/ds4blind/
[11] https://github.com/feddelegrand7/savonliquide

- Making betteR figures: Accessibility and Universal Design[12]
- Highlights from the DVS accessibility fireside chat[13]

[12] https://bookdown.org/ybrandvain/Applied-Biostats/betteRfigs.html#accessibility-and-universal-design

[13] https://nightingaledvs.com/highlights-from-the-dvs-accessibility-fireside-chat/

# 8

# *Exploring interactivity in visualizations with plotly and crosstalk*

*sample text*

We talk about the *FOO* method in this chapter.

# 9

# *Using RMarkdown to build websites for projects*

*sample text*

We talk about the *FOO* method in this chapter.

# 10

# *Using RMarkdown to build dashboards for projects*

*sample text*

We talk about the *FOO* method in this chapter.

# 11

# *Basic usability for interactive visualizations*

*sample text*

We talk about the *FOO* method in this chapter.

# 12

# *Teacher's guide*

*sample text*

We talk about the *FOO* method in this chapter.

# A

# *Datasets*

## Duke Enrollment

Duke enrollment[1]

Sample of Duke Enrollment By School dataset, Table A.1.

## A.1 Bar Chart

Figure A.1.

[1] https://doi.org/10.7924/r4db82p1j

**TABLE A.1:** A sample from the Duke Enrollment By School dataset.

| Year | Semester | Origin | Region | Sex | School | Count |
|---|---|---|---|---|---|---|
| 1970 | Fall | Alabama | United States | Female | Trinity | 11 |
| 1970 | Fall | Alabama | United States | Female | Graduate | 7 |
| 1970 | Fall | Alabama | United States | Female | Divinity | 1 |
| 1970 | Fall | Alabama | United States | Female | Law | 1 |
| 1970 | Fall | Alaska | United States | Female | Trinity | 1 |
| 1970 | Fall | Alaska | United States | Female | Graduate | 1 |

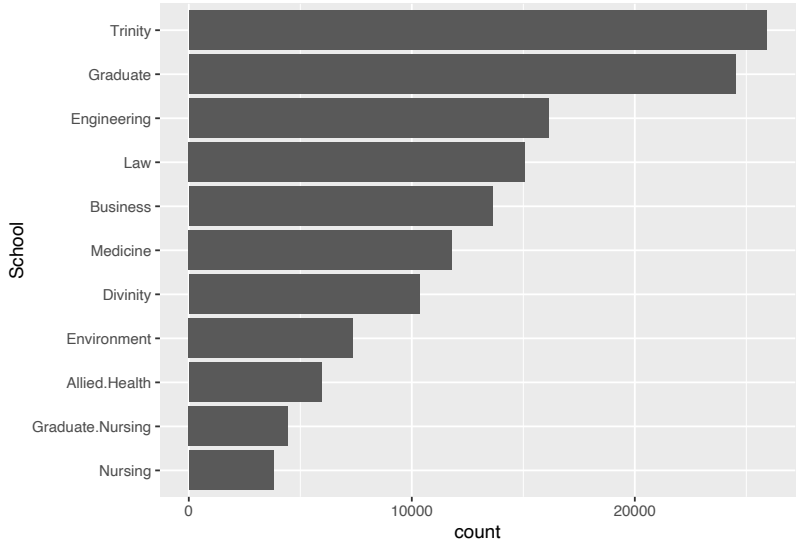



**FIGURE A.1:** Total Duke Enrollment by School

## Coral Resilience Data

Protecting coral reefs[2]

Figure A.2.

```
## Warning: Removed 1 rows containing missing values
## (geom_point).
```

## Git Experience

A Behavioral Approach to Understanding the Git Experience[3]

```
## Warning: Removed 1 row(s) containing missing values
## (geom_path).
```

[2] https://doi.org/10.7924/G8348HFP
[3] https://osf.io/57tb8/

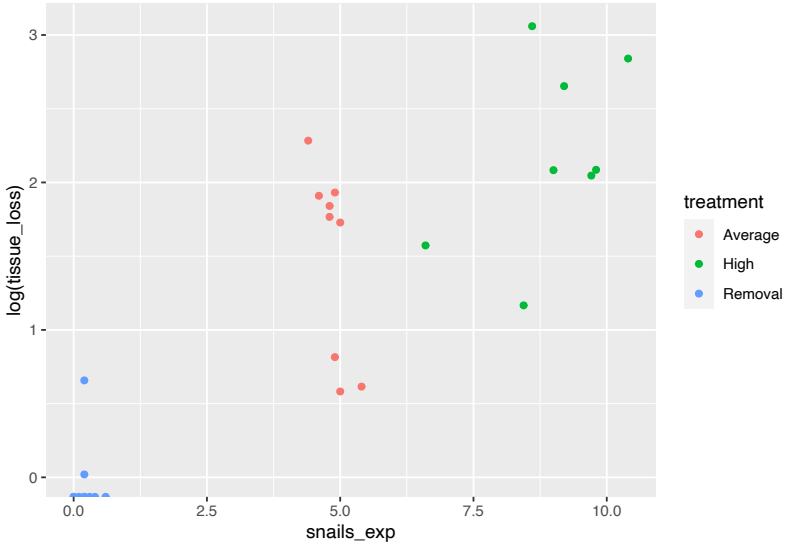
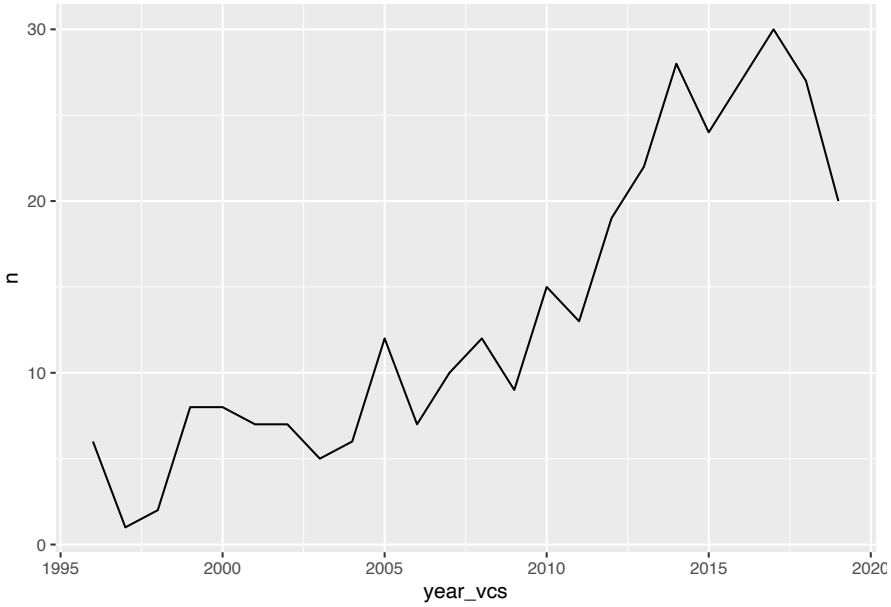
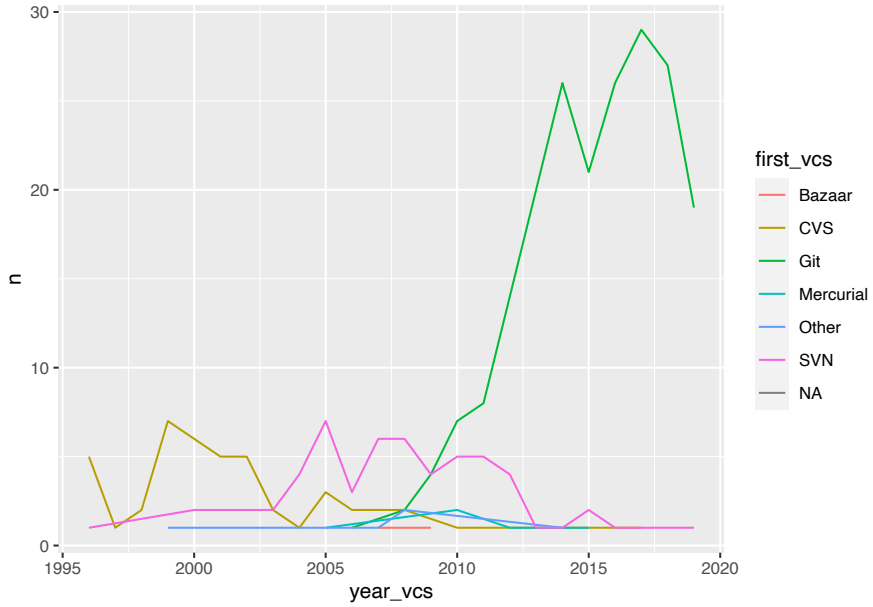


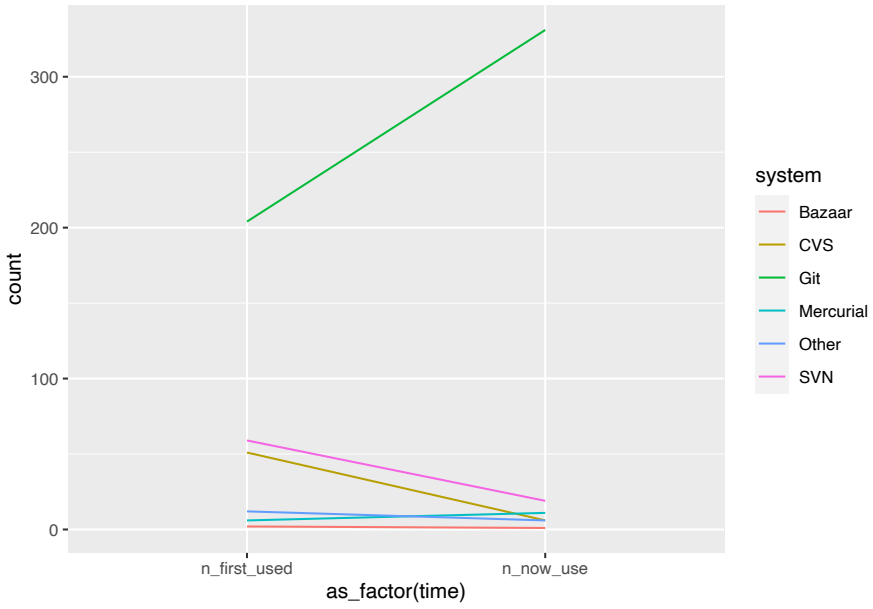

**FIGURE A.2:** Log of tissue loss by snail density

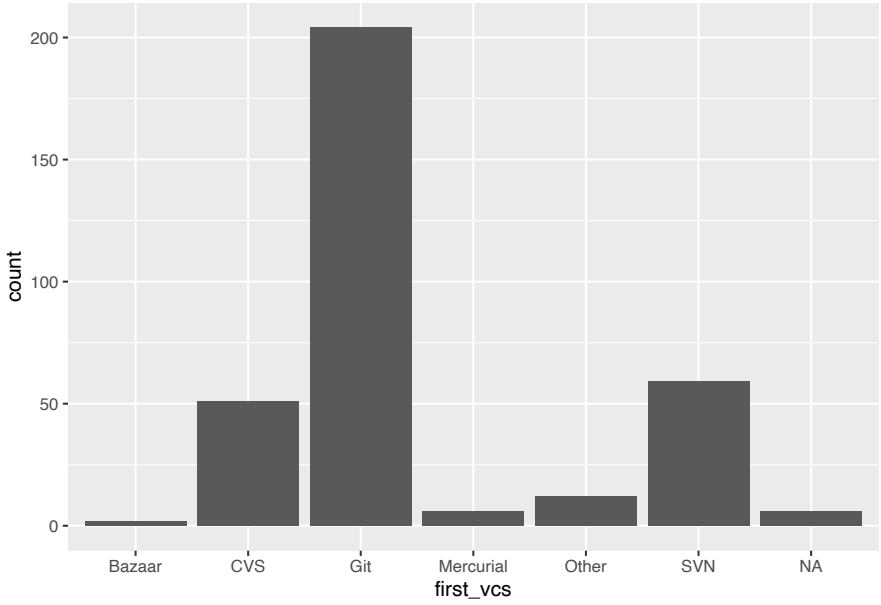


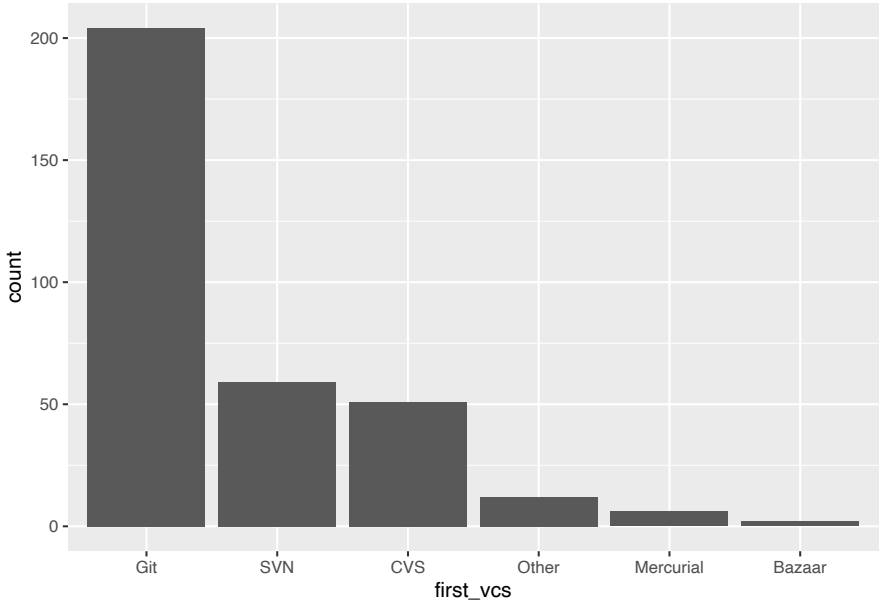

```
## Warning: Removed 3 row(s) containing missing values
## (geom_path).
```

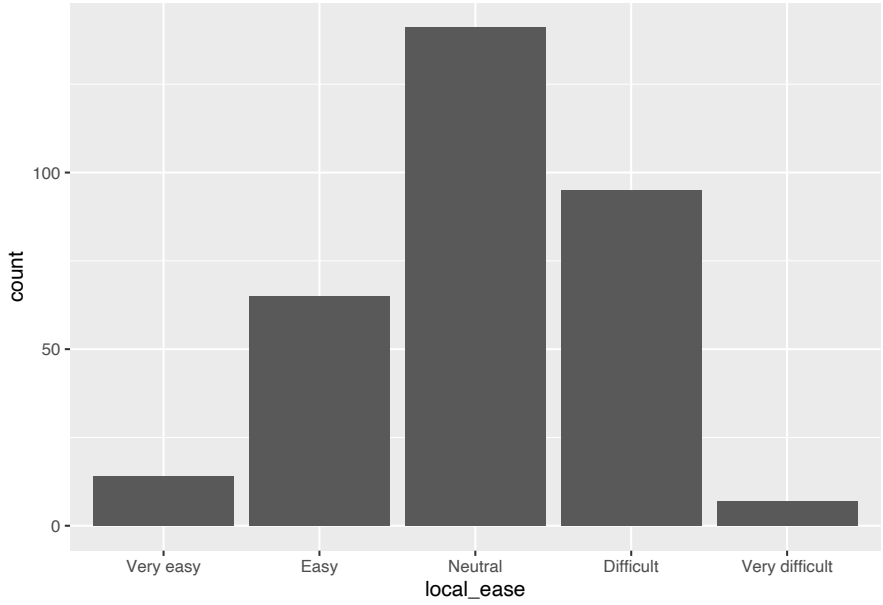


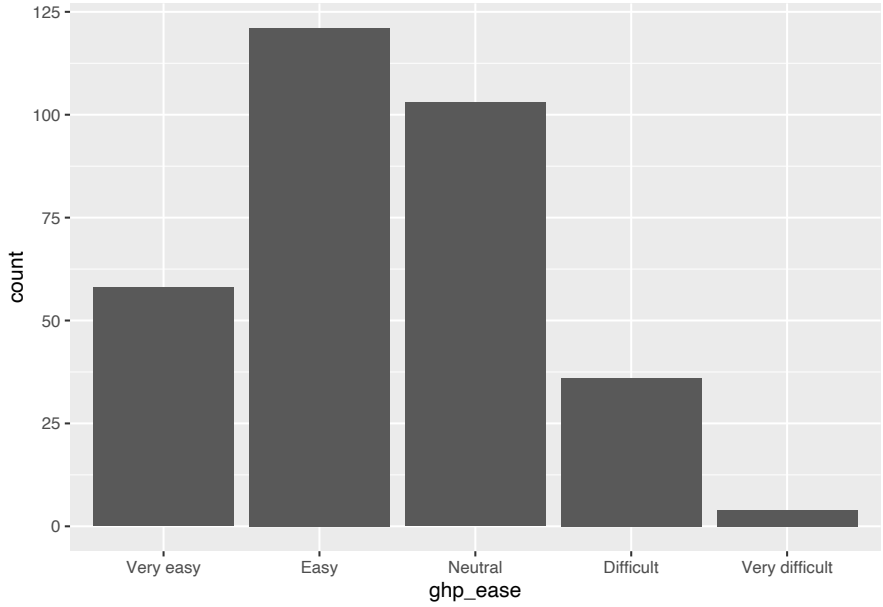

```
## Joining, by = "system"
```

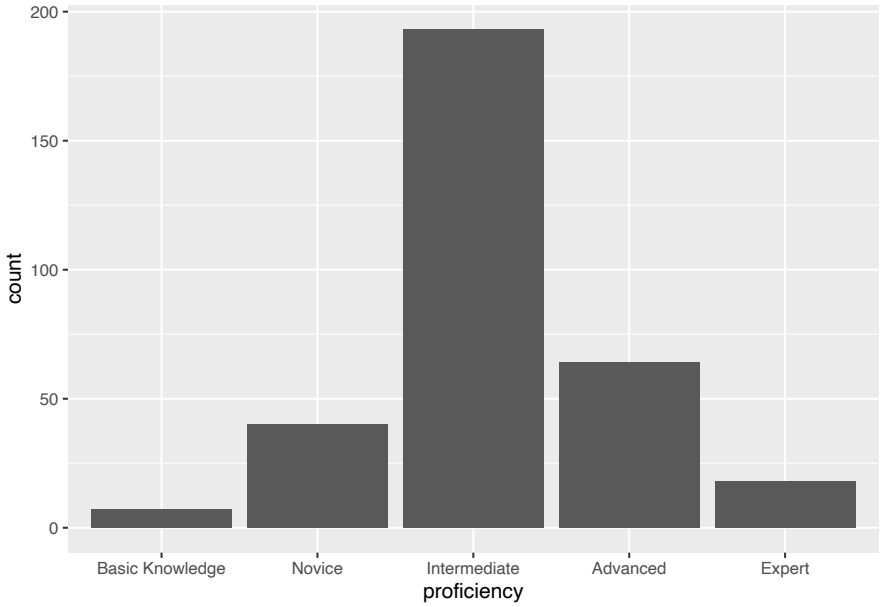


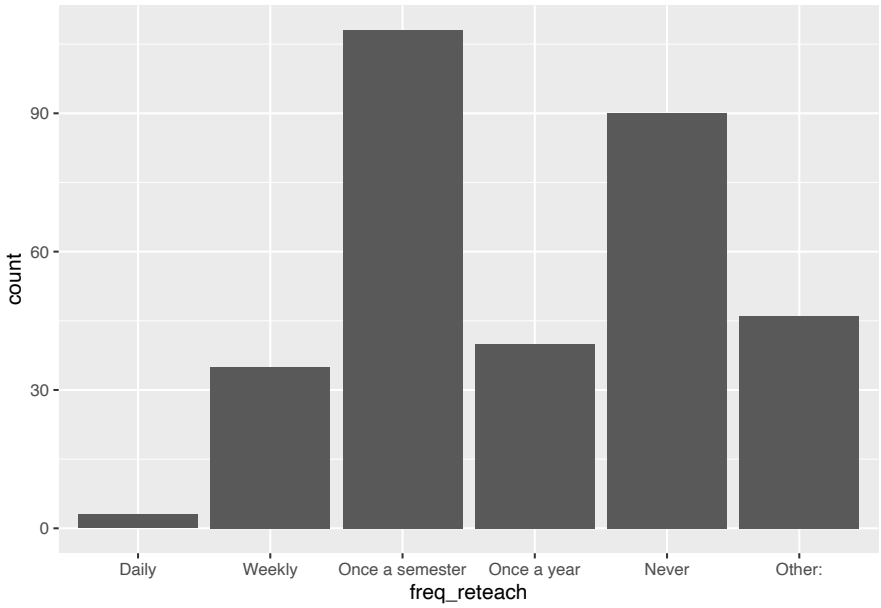

200
150
100
50
0
count
Bazaar
CVS
Git
Mercurial
Other
SVN
NA
first_vcs

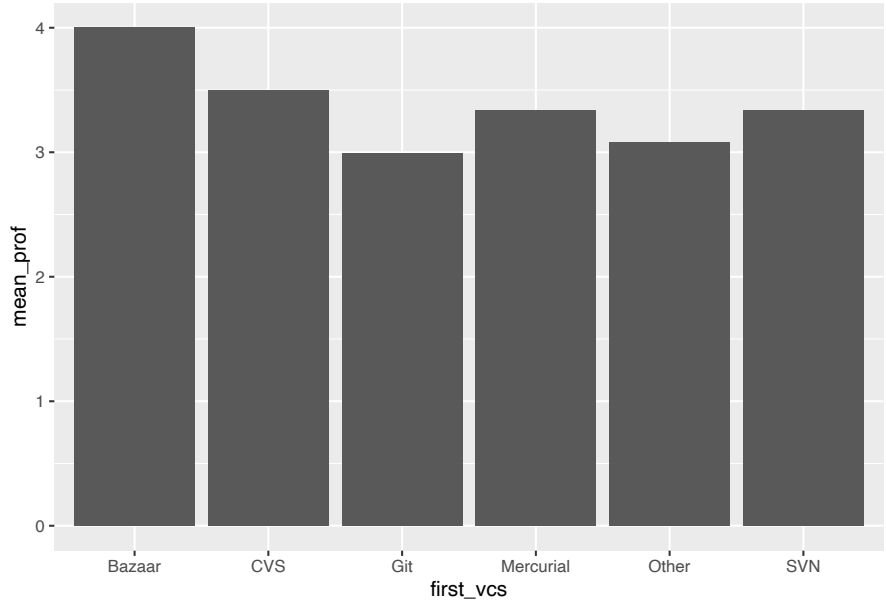

200
150
100
50
0
count
Git
SVN
CVS
Other
Mercurial
Bazaar
first_vcs

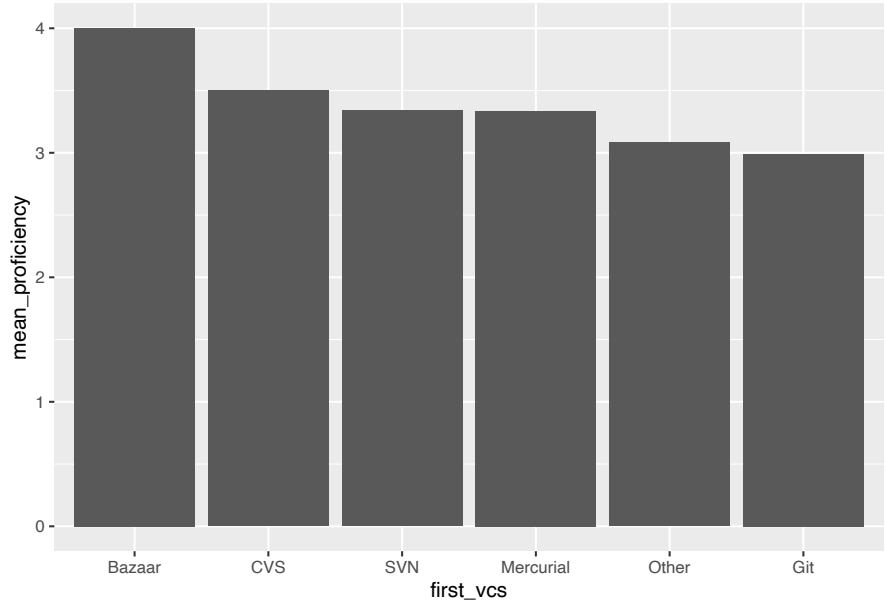

count
100
50
0
Very easy
Easy
Neutral
Difficult
Very difficult
local_ease


count
125
100
75
50
25
0
Very easy
Easy
Neutral
Difficult
Very difficult
ghp_ease

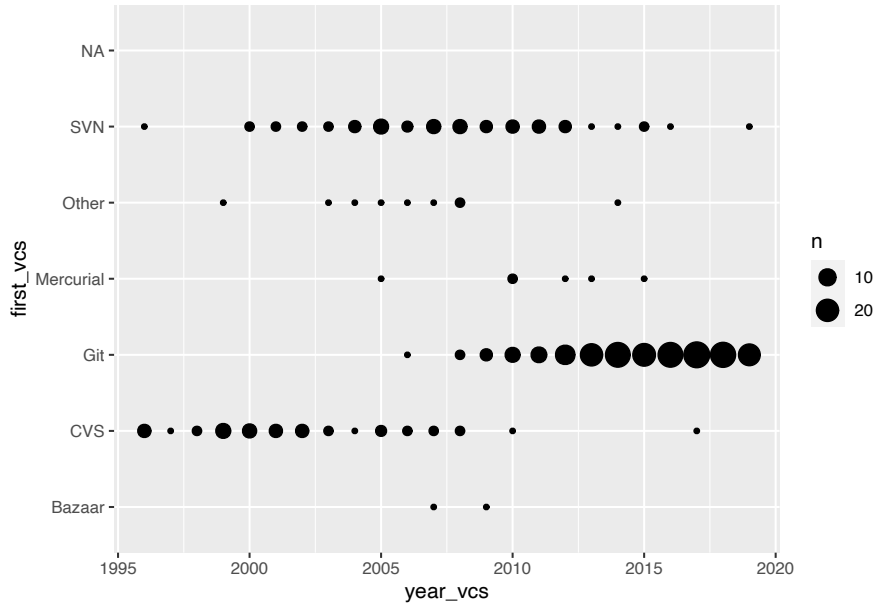

200
150
100
50
0
count
Basic Knowledge
Novice
Intermediate
Advanced
Expert
proficiency


90
60
30
0
count
Daily
Weekly
Once a semester
Once a year
Never
Other:
freq_reteach

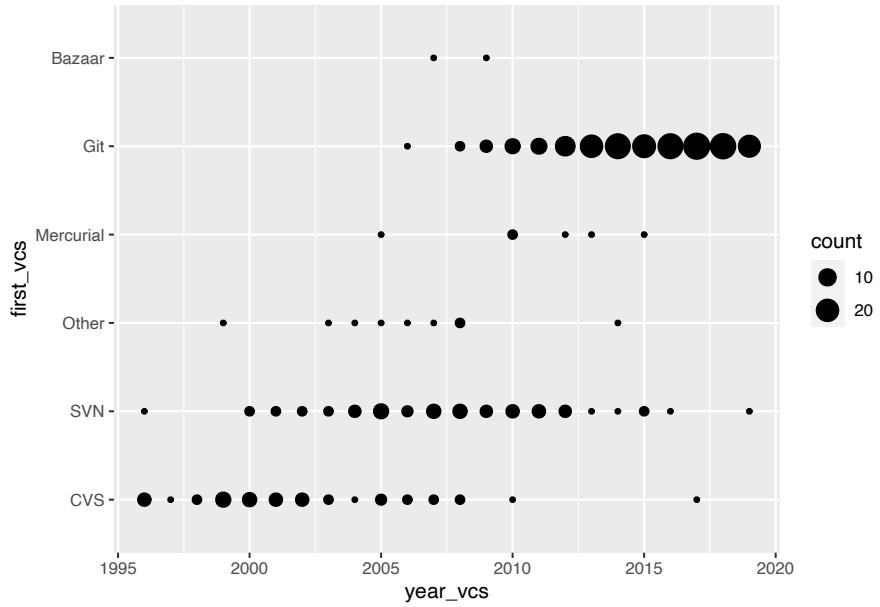


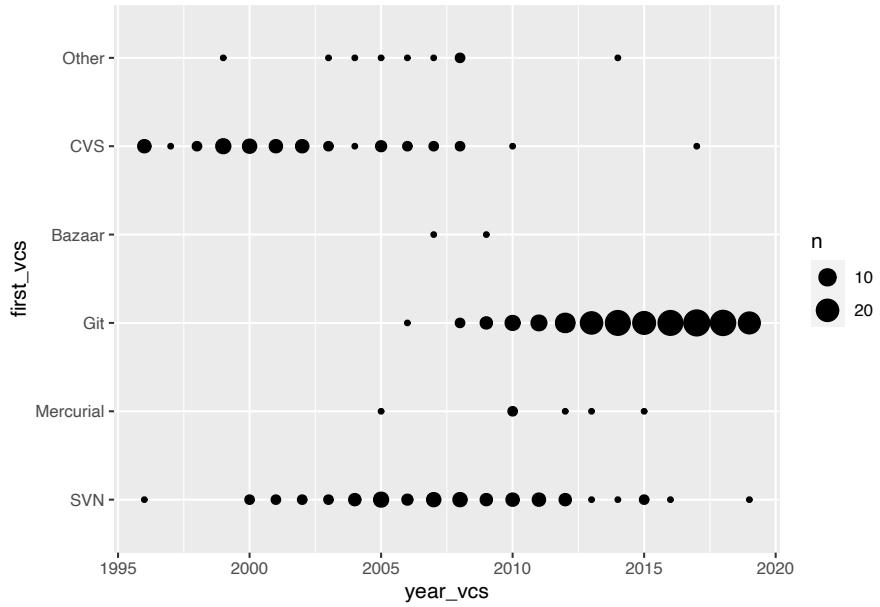

```
## Warning: Removed 15 rows containing non-finite values
## (stat_sum).
```



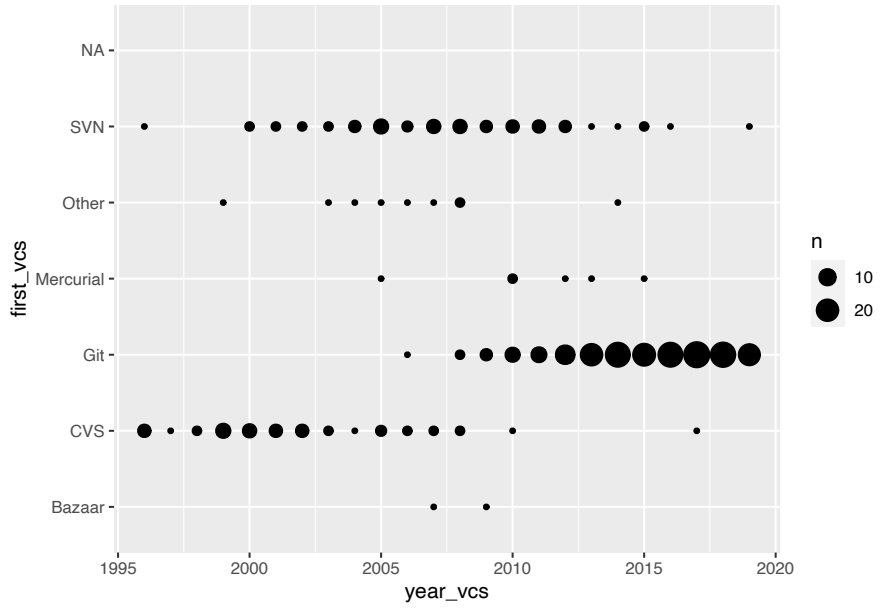

```
## `summarise()` has grouped output by 'year_vcs'. You can override using the `.groups` argument.
```




```
## Warning: Removed 9 rows containing non-finite values
## (stat_sum).
```

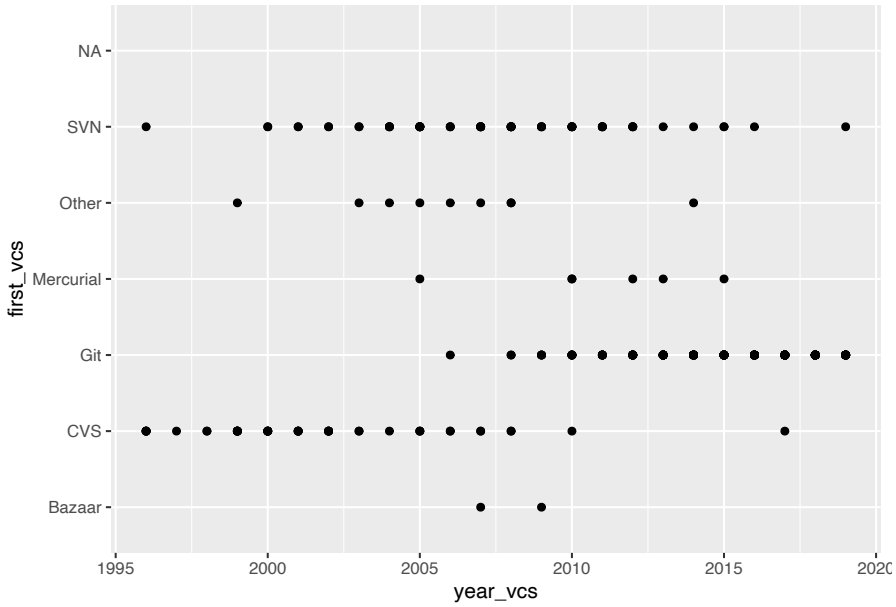



```
## Warning: Removed 3 rows containing missing values
## (geom_point).
```

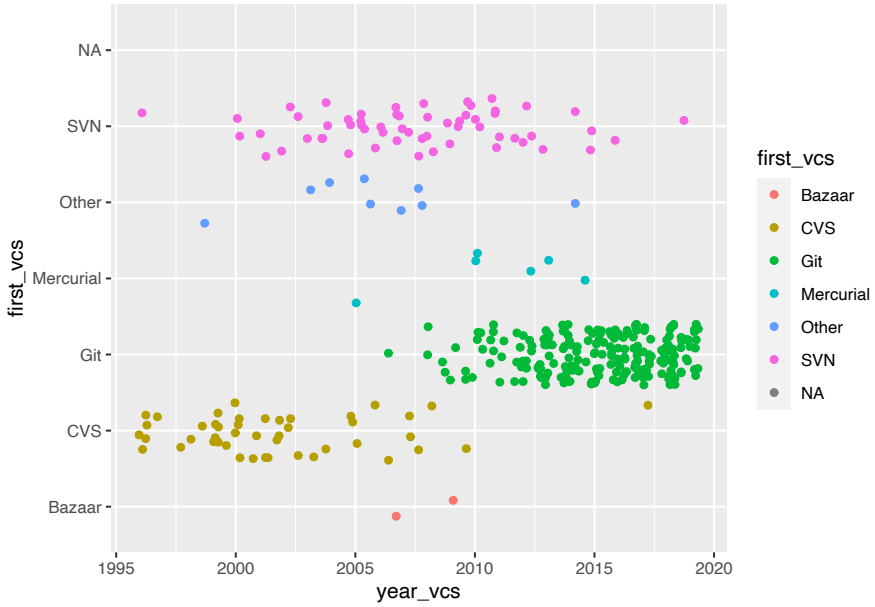



```
## Warning: Removed 15 rows containing missing values
```

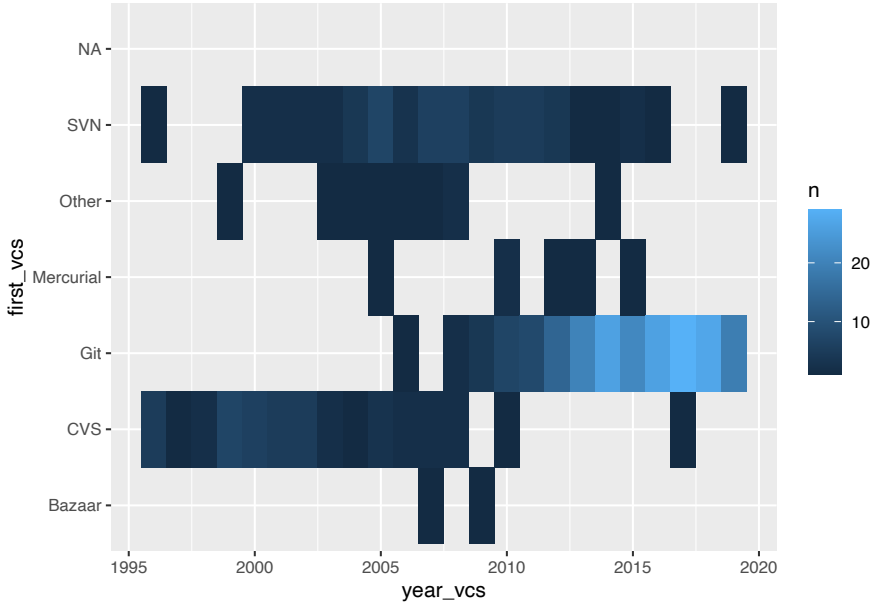
```
## (geom_point).
```

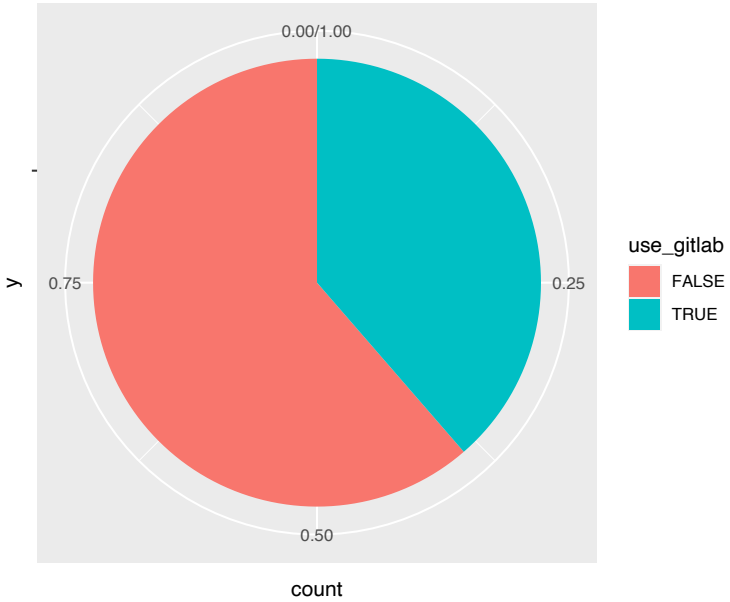



```
## Warning: Removed 15 rows containing missing values
## (geom_point).
```



```
## Warning: Removed 3 rows containing missing values
## (geom_tile).
```



## Inclusiveness Index

Inclusiveness Index[4]

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 111 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 111 rows containing missing values
## (geom_point).
```

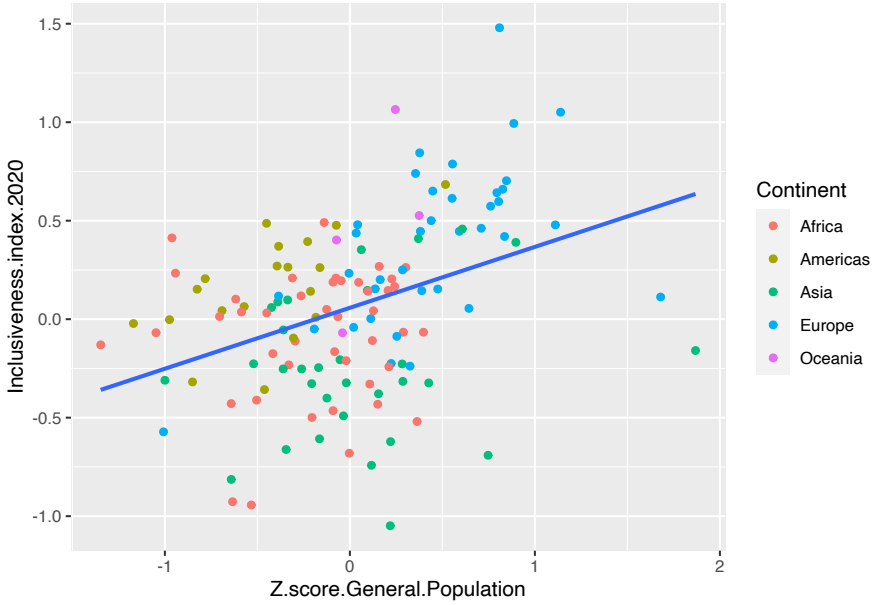
```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 109 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 109 rows containing missing values
## (geom_point).
```

[4] https://belonging.berkeley.edu/inclusivenessindex/data-and-resources

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 113 rows containing non-finite values
## (stat_smooth).
```

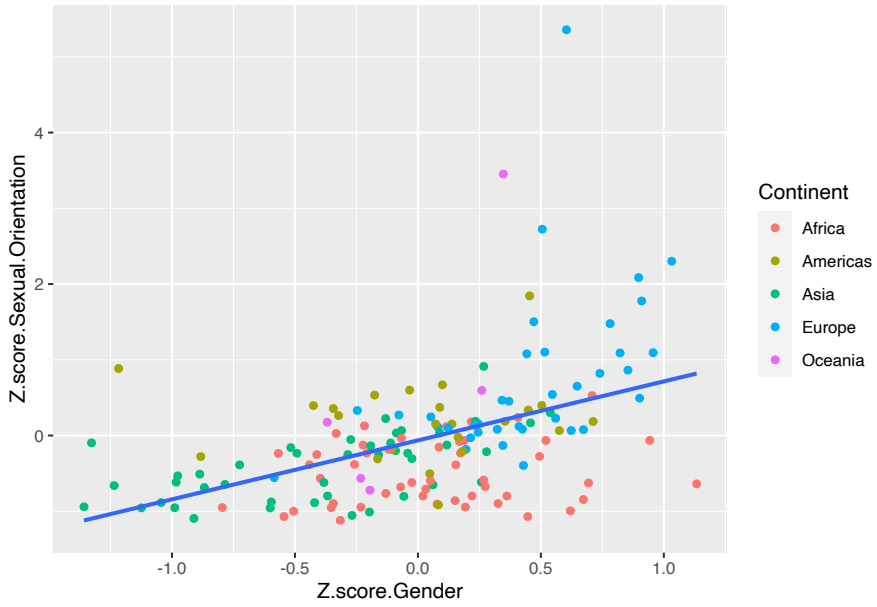
```
## Warning: Removed 113 rows containing missing values
## (geom_point).
```



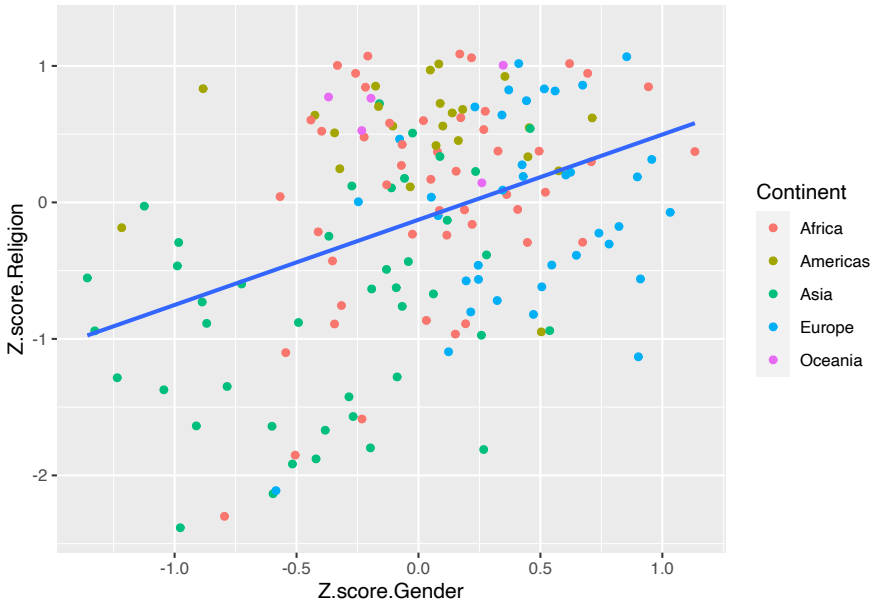

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 90 rows containing non-finite values
## (stat_smooth).
```

```
## Warning: Removed 90 rows containing missing values
## (geom_point).
```



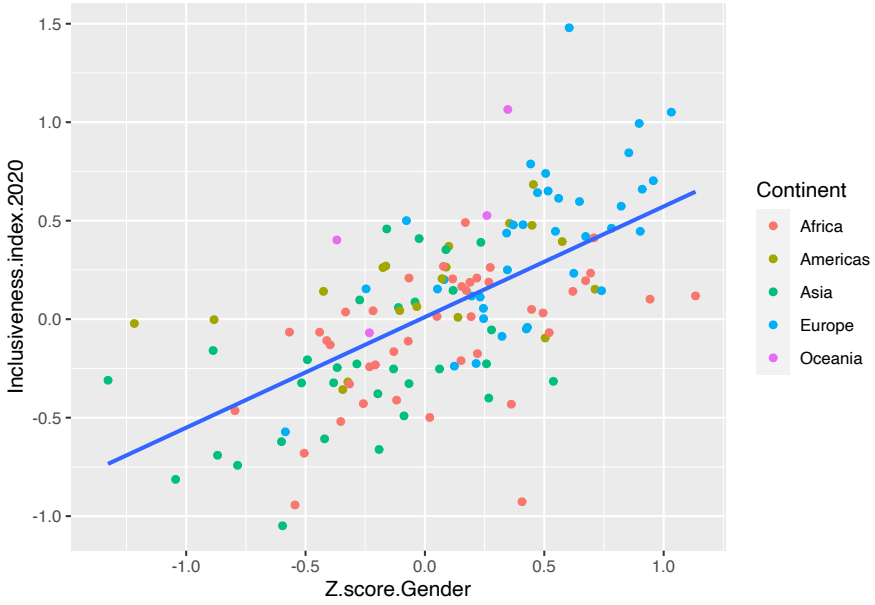

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 90 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 90 rows containing missing values
## (geom_point).
```




```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 113 rows containing non-finite values
## (stat_smooth).
```

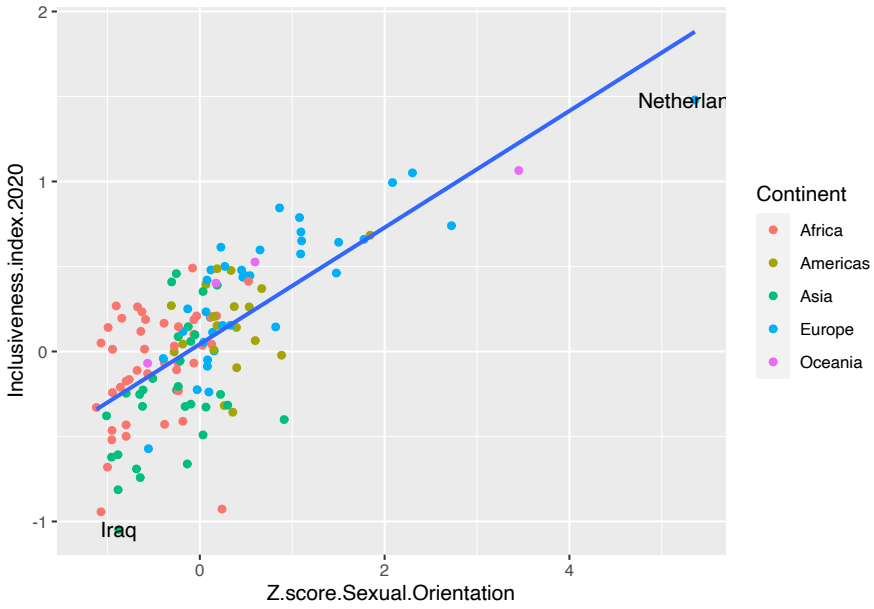
```
## Warning: Removed 113 rows containing missing values
## (geom_point).
```



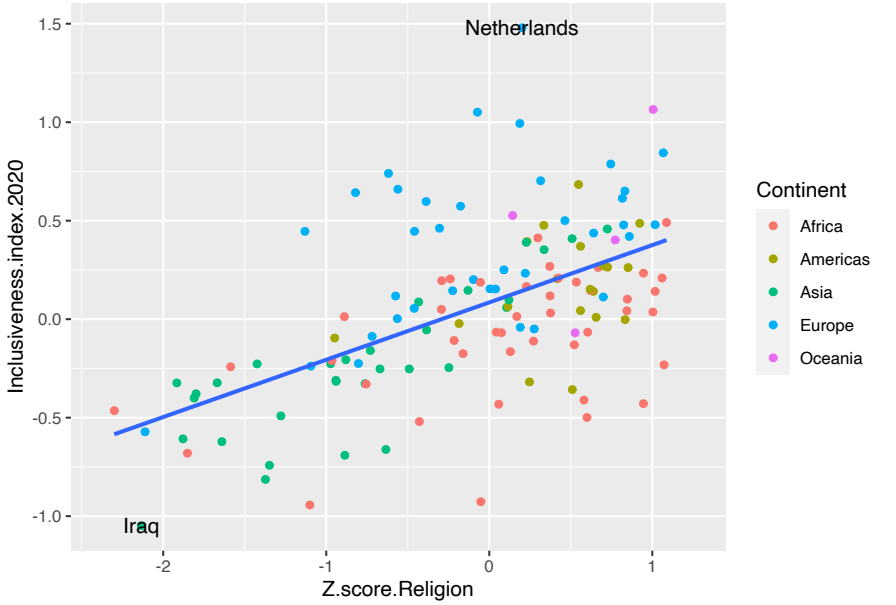

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 113 rows containing non-finite values
## (stat_smooth).
```

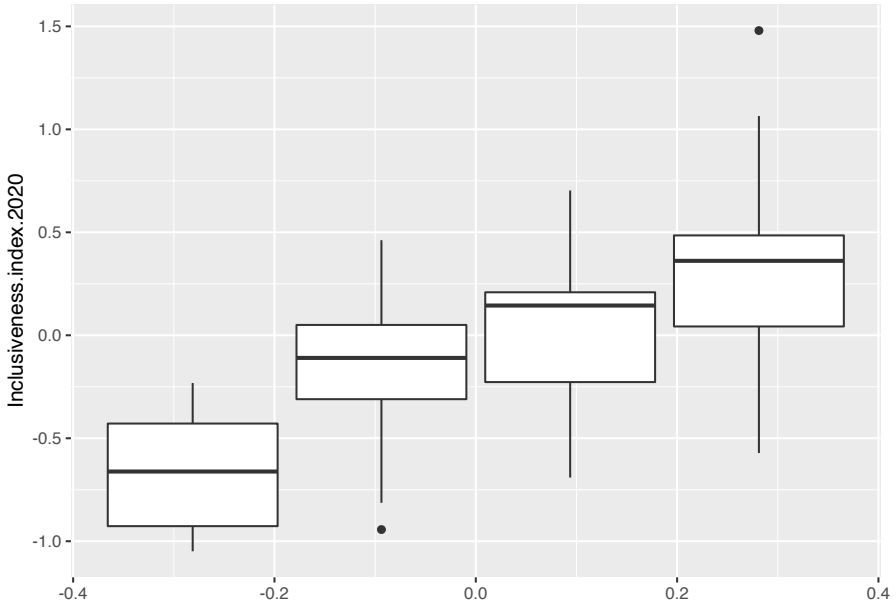
```
## Warning: Removed 113 rows containing missing values
## (geom_point).
```




```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 113 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 113 rows containing missing values
## (geom_point).
```




```
## Warning: Removed 113 rows containing non-finite values
## (stat_boxplot).
```



## Candidate Demographics

Candidate Demographics[5]

Includes State, Candidate Name, Candidate Party, Office Name, White/Non-White, Race, Gender, Race/Gender Category, Office Level; 4 years (2012, 2014, 2016, 2018), over 40k records

## Affinity Spending

Affinity Spending[6]

[5] https://wholeads.us/research/rising-tide-ballot-demographics/
[6] https://github.com/OpportunityInsights/EconomicTracker

# *Bibliography*

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2021). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.23.

# *Index*

bookdown, x

FOO, 27, 29, 31, 37, 39, 41, 43, 45

knitr, x