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

List of Figures

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

1.1 Bar Chart

Figure 1.1.

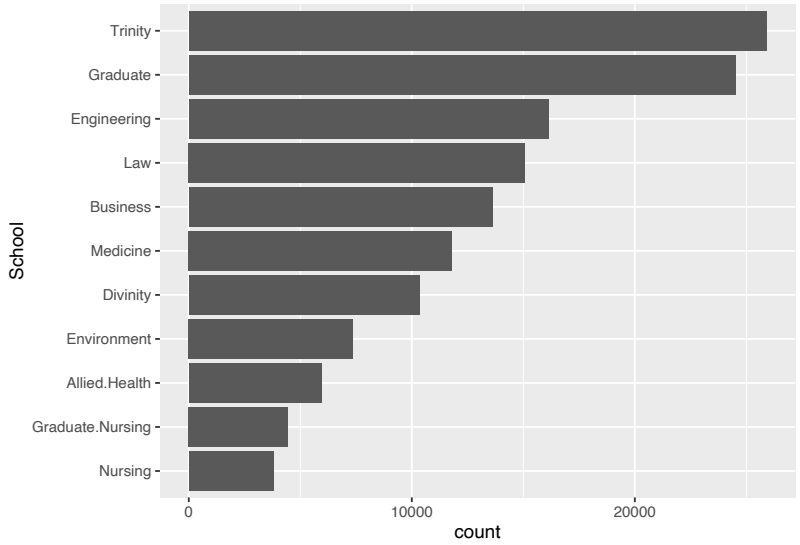
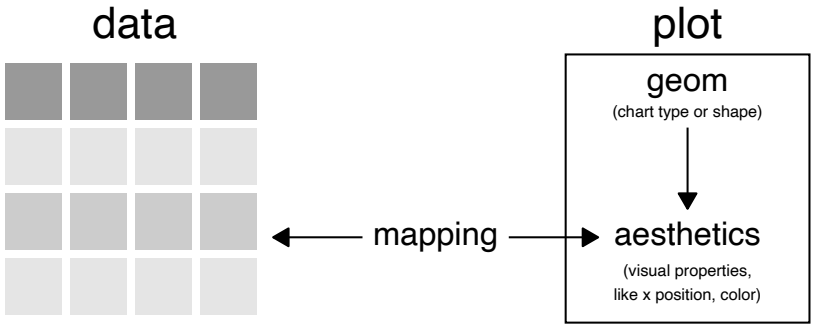


FIGURE 1.1: Total Duke Enrollment by School

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.

Coral Resilience Data

Protecting coral reefs[2]

Figure A.1.

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

[1] https://doi.org/10.7924/r4db82p1j
[2] https://doi.org/10.7924/G8348HFP

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

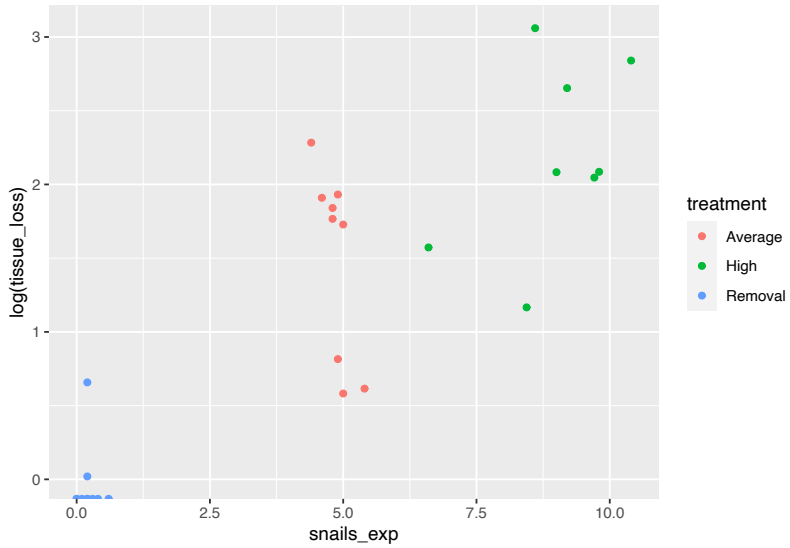


FIGURE A.1: Log of tissue loss by snail density

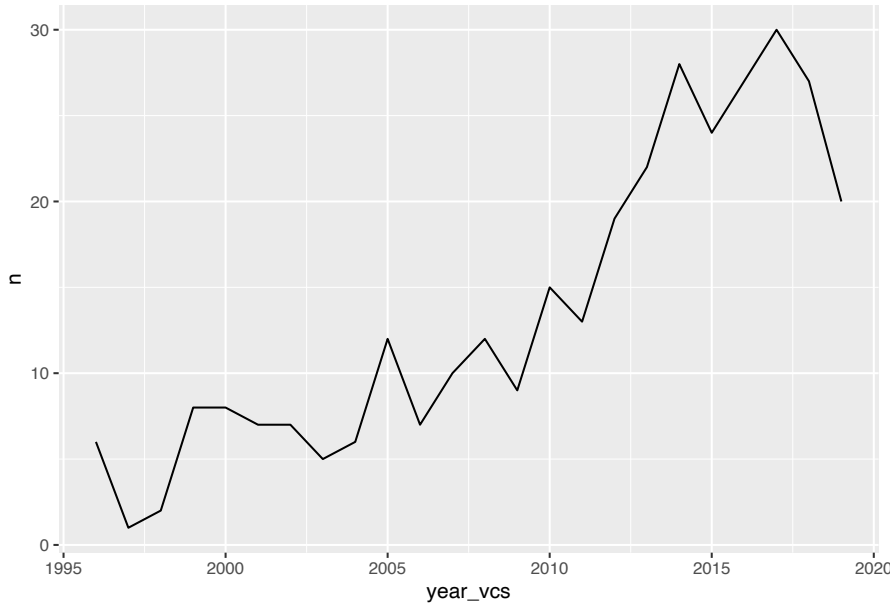
Git Experience

A Behavioral Approach to Understanding the Git Experience[3]

```
# won't work without aggregation
#ggplot(git_experience, aes(x=year_vcs)) +
#  geom_line()

ggplot(git_experience %>% count(year_vcs), aes(x=year_vcs, y=n)) +
  geom_line()
```

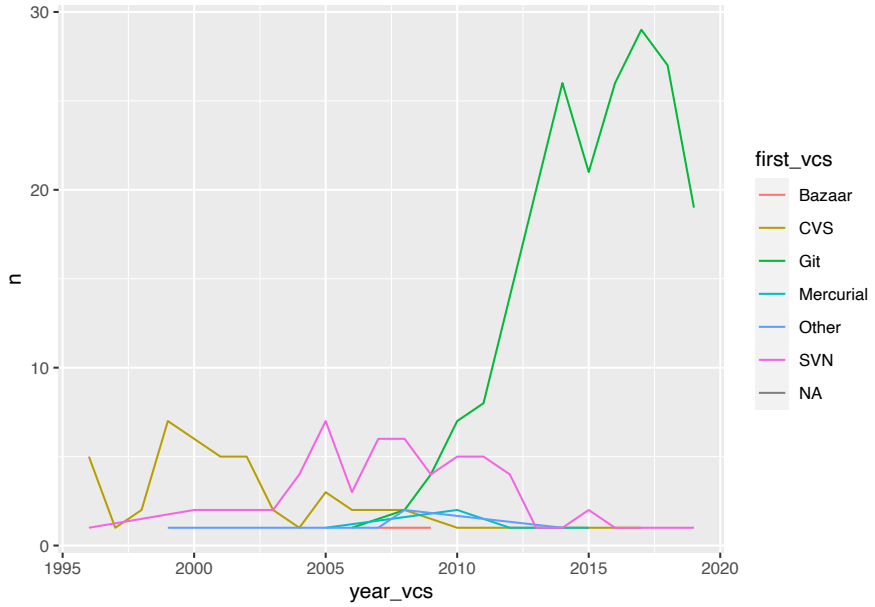
```
## Warning: Removed 1 row(s) containing missing values
## (geom_path).
```

[3] https://osf.io/57tb8/




```
ggplot(git_experience %>% count(year_vcs, first_vcs), aes(x=year_vcs, y=n, color=first_vcs)) +
  geom_line()
```

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

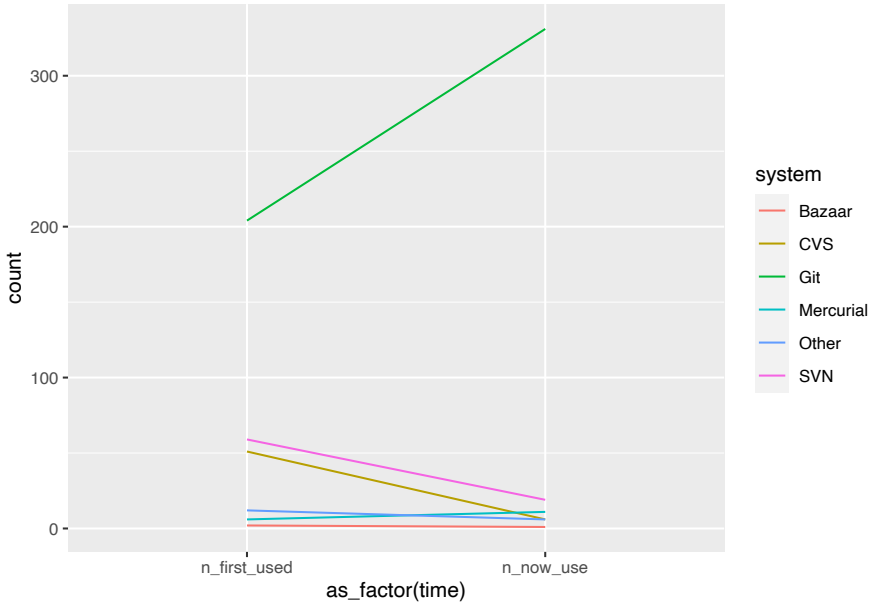



```
# First use vs. now use

first_used <- git_experience %>%
  count(first_vcs) %>%
  rename("n_first_used" = "n", "system" = "first_vcs")

now_use <- git_experience %>%
  pivot_longer(cols=c(first_bazaar,first_cvs,first_git,
                      first_hg,first_monotone,first_svn, first_other),
               names_to = "system") %>%
  dplyr::filter(value == TRUE) %>%
  count(system) %>%
  rename("n_now_use" = "n") %>%
  mutate(system = case_when(
    system == "first_bazaar" ~ "Bazaar",
    system == "first_cvs" ~ "CVS",
    system ==  "first_git" ~ "Git",
    system ==  "first_hg" ~ "Mercurial",
    system ==  "first_monotone" ~ "Monotone",
    system ==  "first_svn" ~ "SVN",
    system ==  "first_other" ~ "Other"
  ))
```

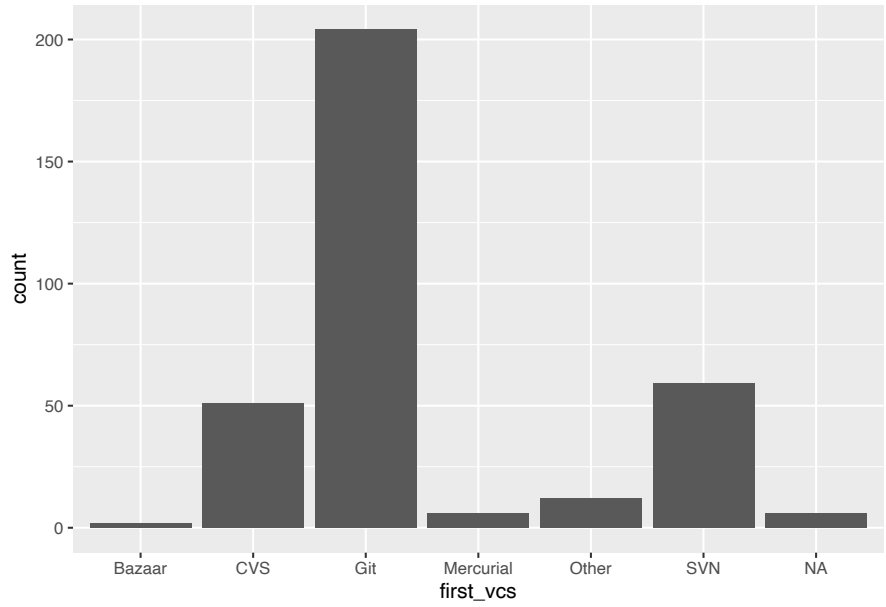
```
combined <- first_used %>% left_join(now_use) %>%
  pivot_longer(cols = c(n_first_used, n_now_use),
               names_to = "time",
               values_to = "count") %>%
  drop_na()
```

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

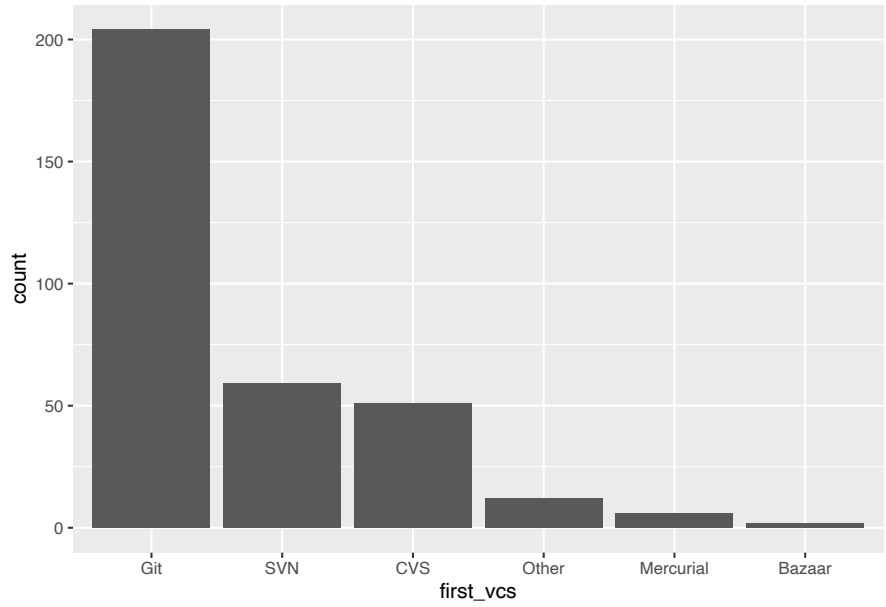
```
ggplot(combined, aes(x=as_factor(time), y=count, group=system, color=system)) +
  geom_line()
```



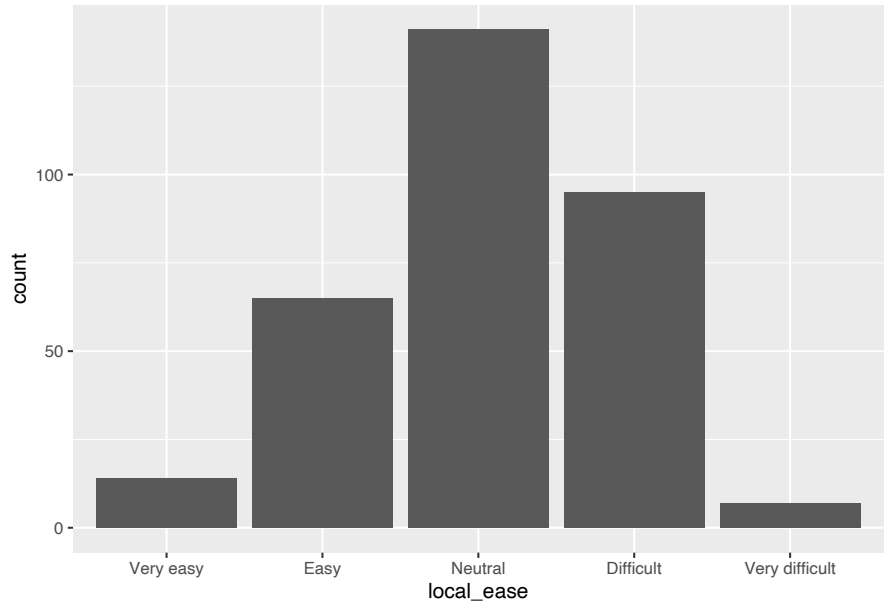

```
ggplot(git_experience, aes(x=first_vcs)) +
  geom_bar()
```



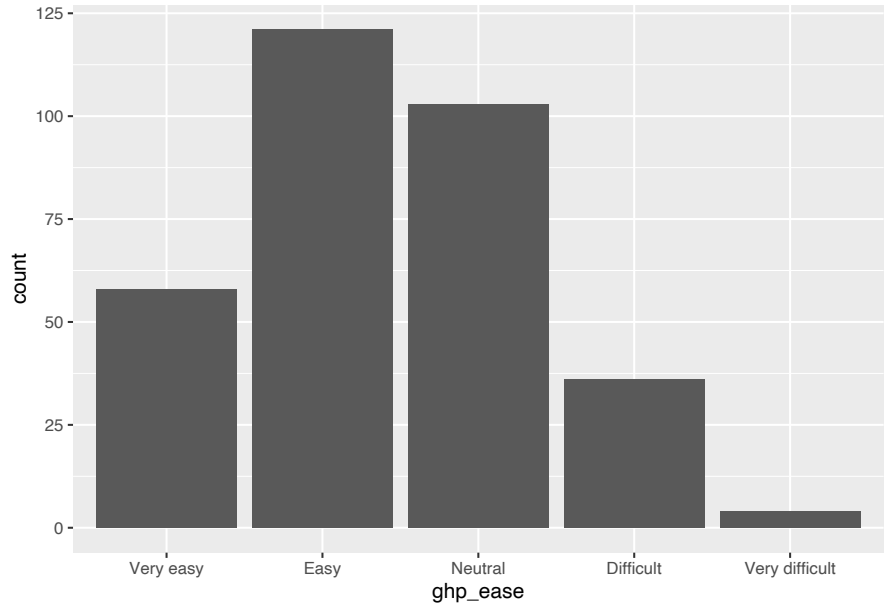

```
ggplot(git_experience %>%
         drop_na(first_vcs) %>%
         mutate(first_vcs = as_factor(first_vcs) %>%
             fct_infreq()),
         aes(x=first_vcs)) +
  geom_bar()
```



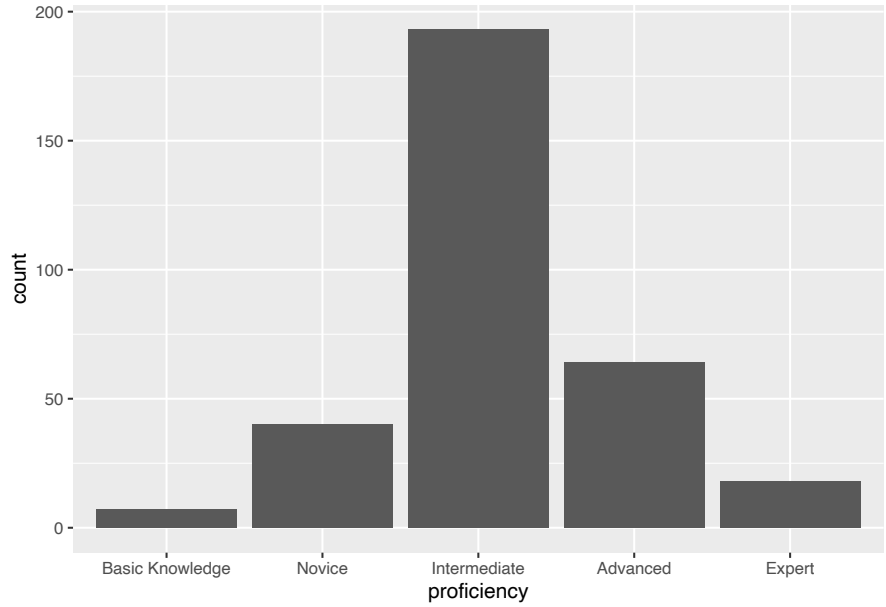

```
#How difficult was it for you to learn how to use git on your local computer?
# 1 - Very easy (1)
# 2 - Easy (2)
# 3 - Neutral (Neither easy nor difficult) (3)
# 4 - Difficult (4)
# 5 - Very difficult (5)

ggplot(git_experience %>%
         drop_na(local_ease) %>%
         mutate(local_ease = as_factor(local_ease) %>%
             fct_recode("Very easy"="1",
                        "Easy"="2", "Neutral"="3",
                        "Difficult"="4", "Very difficult"="5")),
       aes(local_ease)) +
  geom_bar()
```



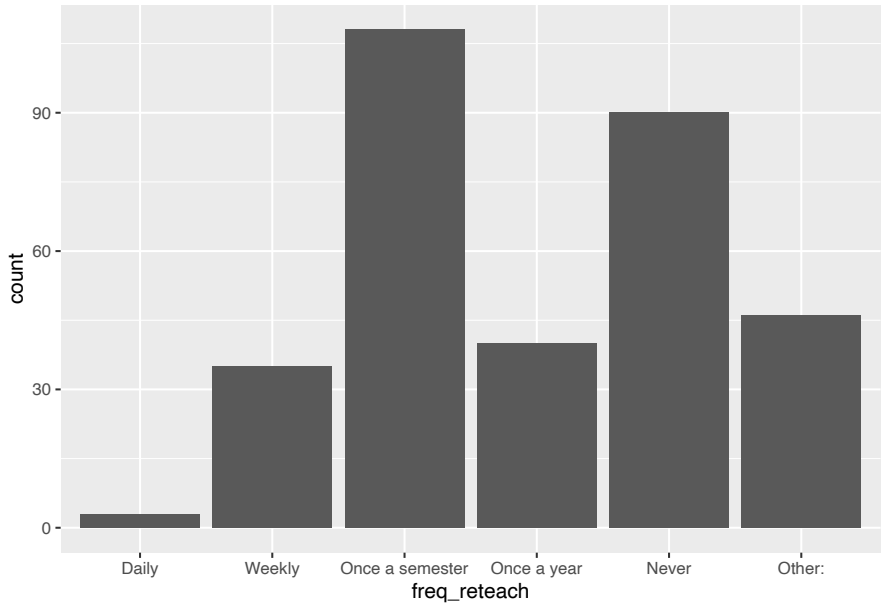

```
#How difficult was it for you to learn how to use the git hosting platform (e.g. GitLab, GitHub, e
# 1 - Very easy (1)
# 2 - Easy (2)
# 3 - Neutral (Neither easy nor difficult) (3)
# 4 - Difficult (4)
# 5 - Very difficult (5)
ggplot(git_experience %>%
         drop_na(ghp_ease) %>%
         mutate(ghp_ease = as_factor(ghp_ease) %>%
             fct_recode("Very easy"="1",
                        "Easy"="2", "Neutral"="3",
                        "Difficult"="4", "Very difficult"="5")),
       aes(ghp_ease)) +
  geom_bar()
```



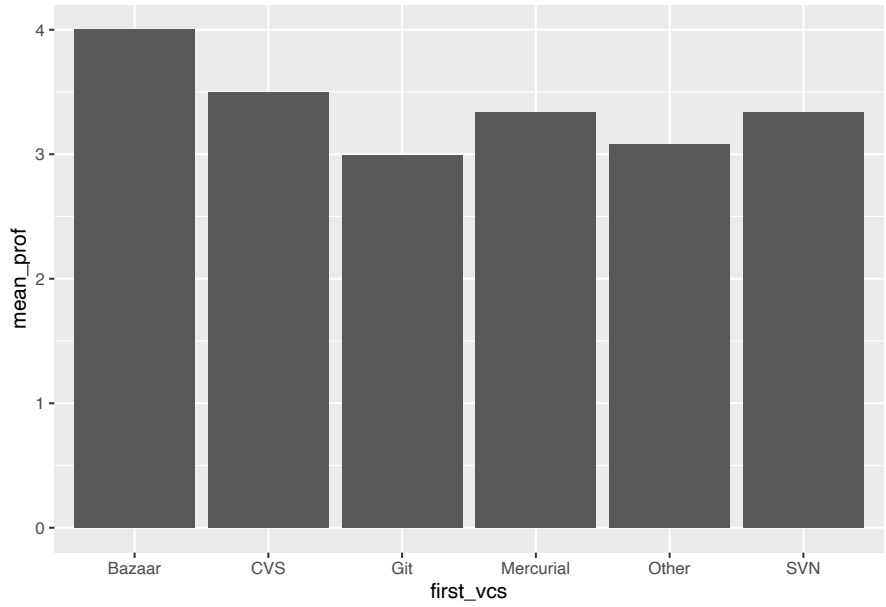

```
# How proficient do you think you are with git?
# 1 - Fundamental Awareness (basic knowledge) (1)
# 2 - Novice (limited experience) (2)
# 3 - Intermediate (practical application) (3)
# 4 - Advanced (applied theory) (4)
# 5 - Expert (recognized authority) (5)
ggplot(git_experience %>%
         drop_na(proficiency) %>%
         mutate(proficiency = as_factor(proficiency) %>%
             fct_recode("Basic Knowledge"="1",
                        "Novice"="2", "Intermediate"="3",
                        "Advanced"="4", "Expert"="5")),
       aes(proficiency)) +
  geom_bar()
```



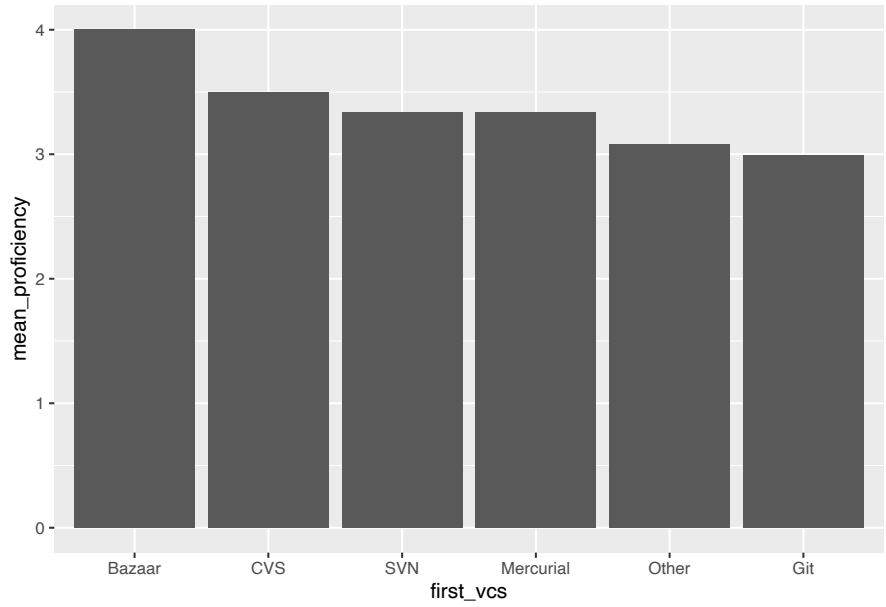

```
# How frequently do you have to reteach yourself git?
# Daily (1)
# Weekly (2)
# Once a semester (3)
# Once a year (4)
# Never (5)
# Other: (6)
ggplot(git_experience %>%
         drop_na(freq_reteach) %>%
         mutate(freq_reteach = as_factor(freq_reteach) %>%
             fct_relevel("Daily", "Weekly", "Once a semester",
                         "Once a year", "Never", "Other:")),
       aes(freq_reteach)) +
  geom_bar()
```




```
git_experience %>% drop_na(first_vcs) %>%
  group_by(first_vcs) %>%
  summarise(count = n(), mean_prof = mean(proficiency, na.rm=TRUE)) %>%
  ggplot(aes(x=first_vcs, y=mean_prof)) +
    geom_col()
```

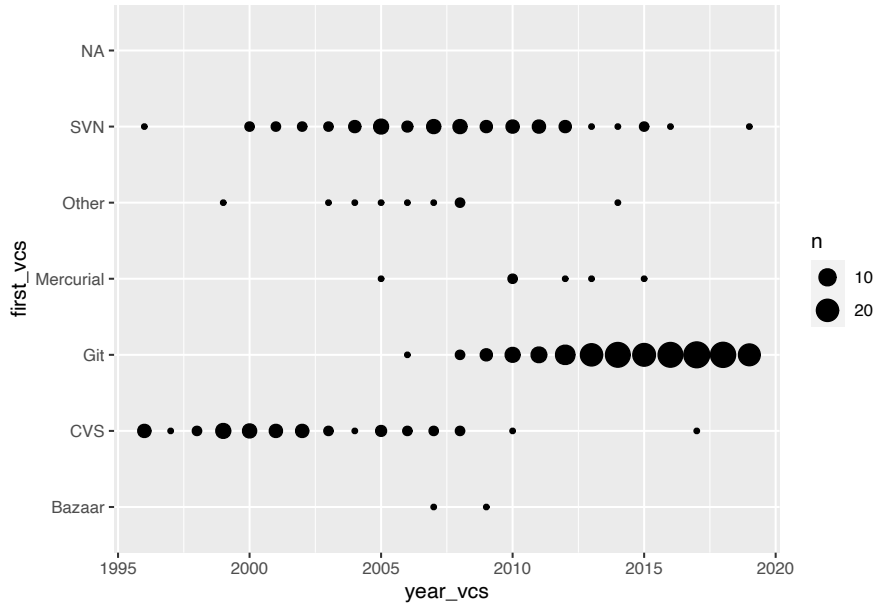



```
git_experience %>%
  drop_na(first_vcs) %>%
  group_by(first_vcs) %>%
  summarise(count = n(), mean_proficiency = mean(proficiency, na.rm=TRUE)) %>%
  mutate(first_vcs = as_factor(first_vcs) %>%
           fct_reorder(mean_proficiency) %>% fct_rev()) %>%
  ggplot(aes(x=first_vcs, y=mean_proficiency)) +
    geom_col()
```



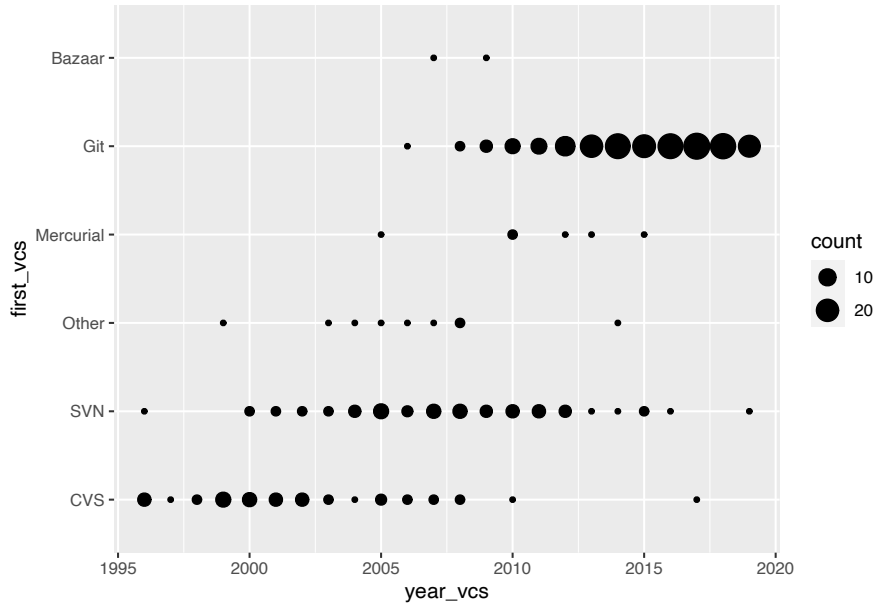

```
ggplot(git_experience, aes(x=year_vcs, y=first_vcs)) +
  geom_count()
```

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



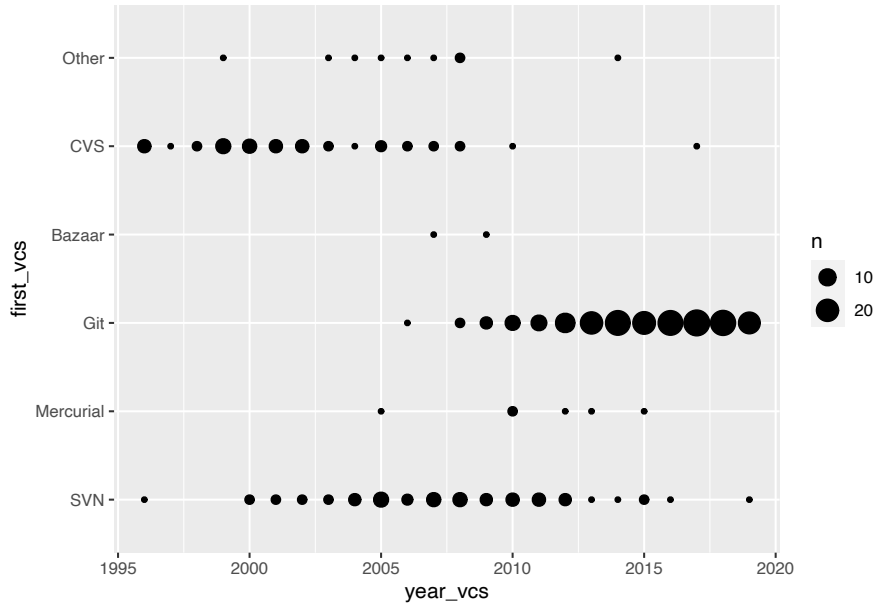

```
git_experience %>%
  drop_na(year_vcs, first_vcs) %>%
  group_by(year_vcs, first_vcs) %>%
  summarize(min_year = min(year_vcs),
            count=n()) %>%
  mutate(first_vcs = as_factor(first_vcs) %>%
           fct_reorder(min_year)) %>%
  ggplot(aes(x=year_vcs, y=first_vcs, size=count)) +
  geom_point()
```

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



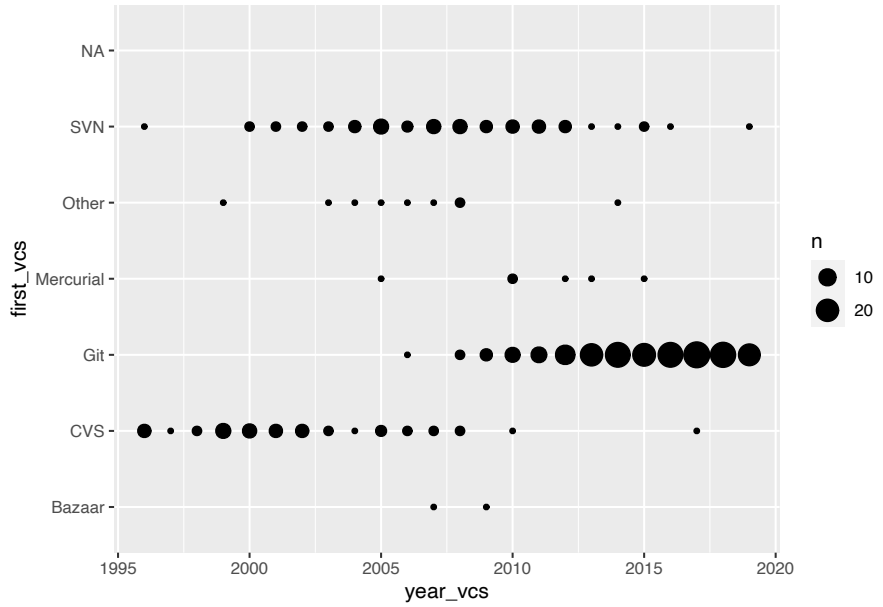

```
git_experience %>%
  drop_na(first_vcs) %>%
  mutate(first_vcs = as_factor(first_vcs) %>%
           fct_reorder(year_vcs, .fun=min)) %>%
  ggplot(aes(x=year_vcs, y=first_vcs)) +
  geom_count()
```

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



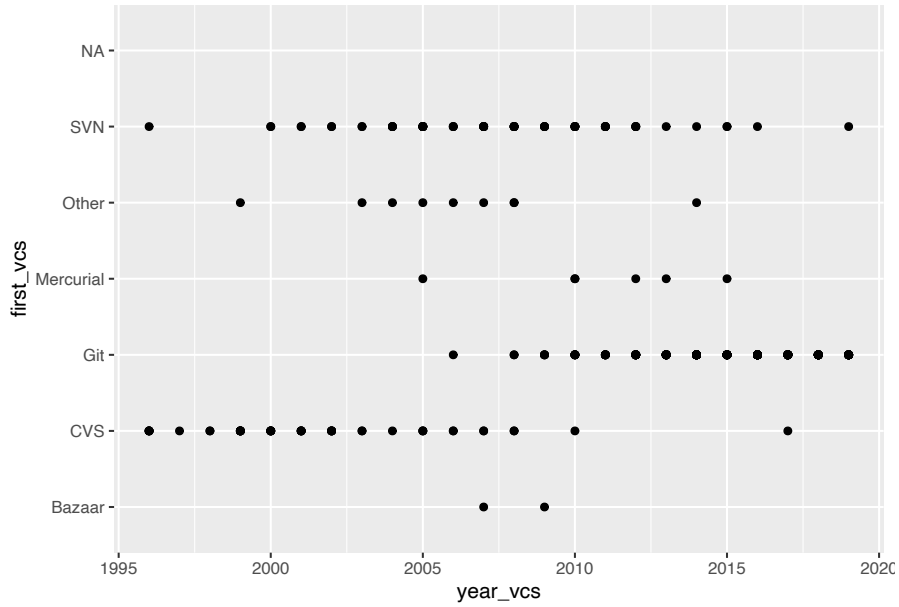

```
ggplot(git_experience %>% count(year_vcs, first_vcs),
       aes(x=year_vcs, y=first_vcs, size=n)) +
  geom_point()
```

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



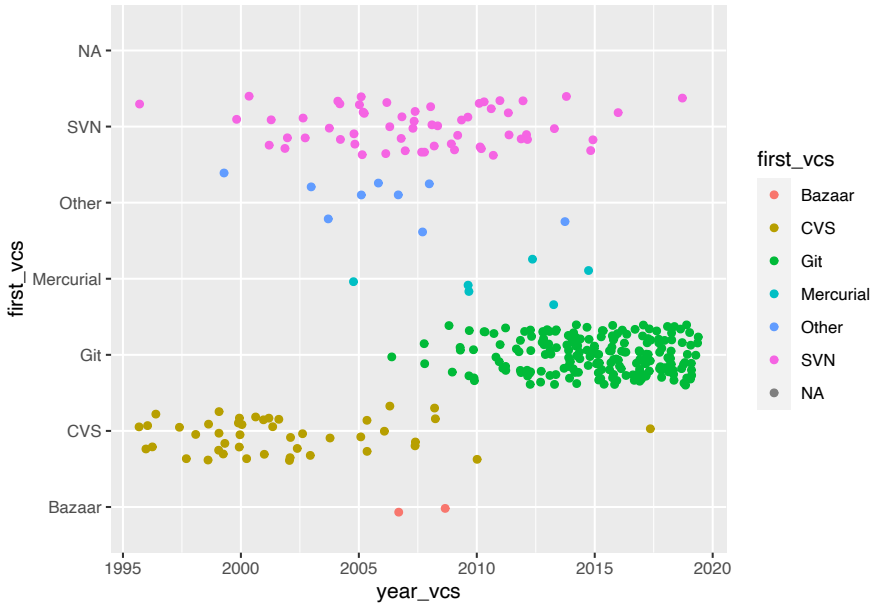

```
ggplot(git_experience, aes(x=year_vcs, y=first_vcs)) +
  geom_point()
```

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



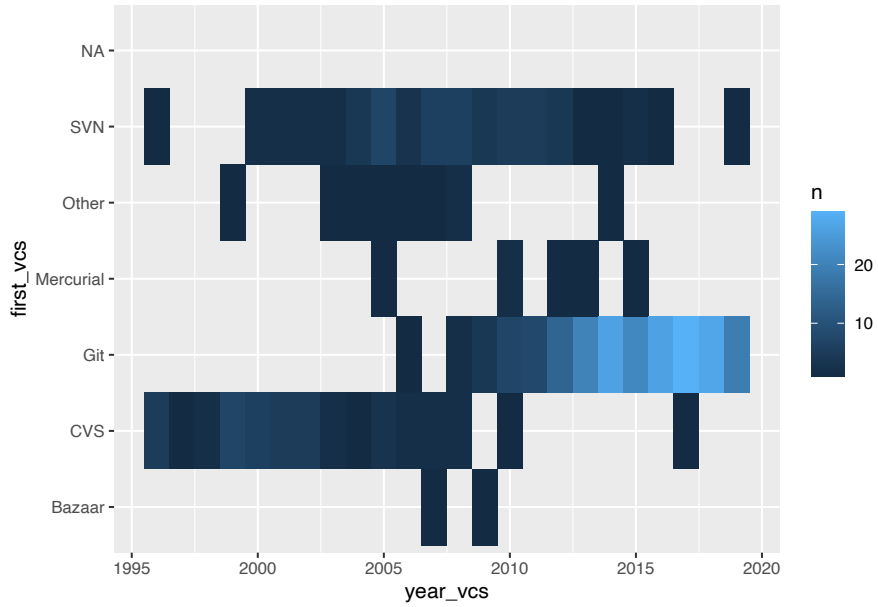

```
ggplot(git_experience, aes(x=year_vcs, y=first_vcs, color=first_vcs)) +
  geom_jitter()
```

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

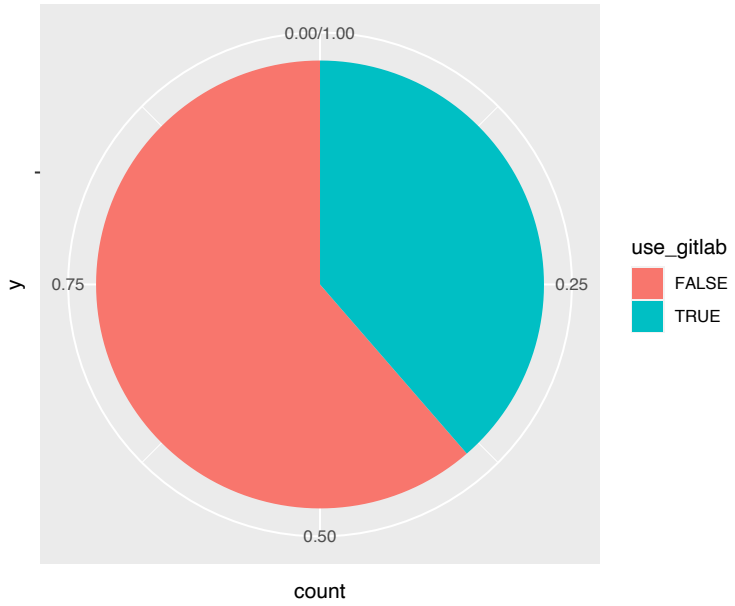



```
ggplot(git_experience %>% count(year_vcs, first_vcs),
       aes(x=year_vcs, y=first_vcs, fill=n)) +
  geom_tile()
```

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




```
ggplot(git_experience %>% drop_na(use_gitlab), aes(y="",fill=use_gitlab)) +
  geom_bar(position=position_fill()) +
  coord_polar()
```

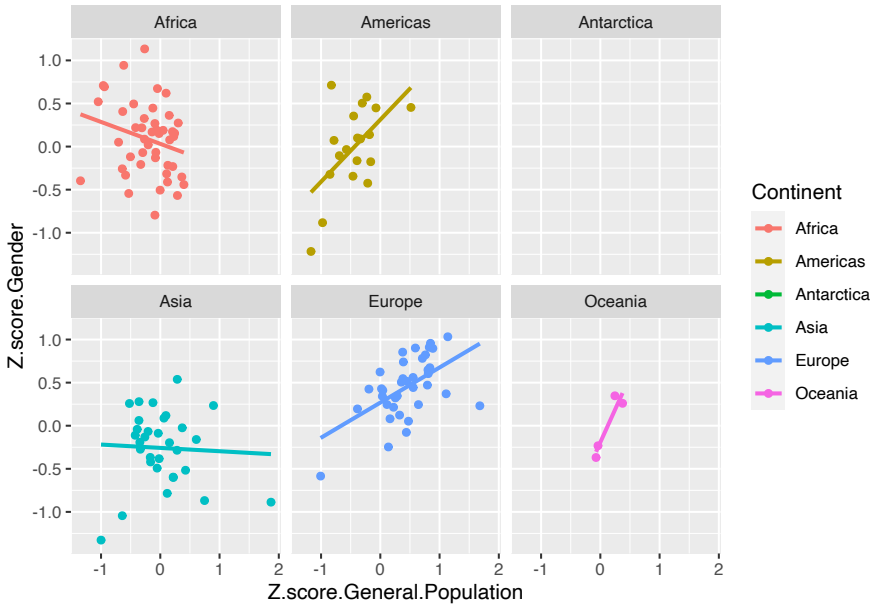


Inclusiveness Index

Inclusiveness Index[4]

```
ggplot(inclusiveness_index,
       aes(x = Z.score.General.Population,
           y = Z.score.Gender,
           color = Continent)) +
  geom_point() +
  geom_smooth(method="lm", se=FALSE) +
  facet_wrap(vars(Continent))
```

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

## Warning: Removed 115 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 115 rows containing missing values
## (geom_point).
```



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

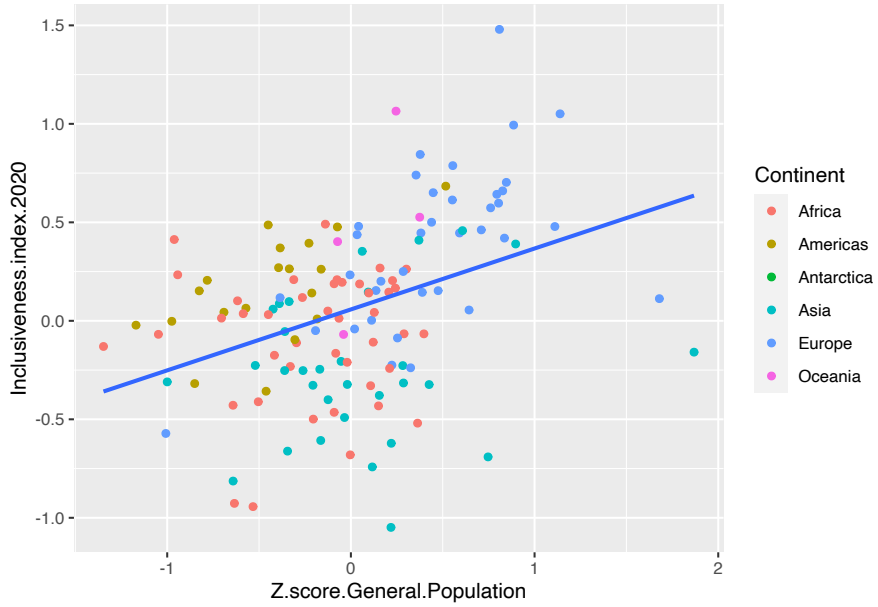
```
ggplot(inclusiveness_index,
       aes(x = Z.score.General.Population,
           y = Z.score.Sexual.Orientation)) +
  geom_point(aes(color = Continent)) +
  geom_smooth(method="lm", se=FALSE)
```

```
## `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).
```


```
ggplot(inclusiveness_index,
       aes(x = Z.score.General.Population,
           y = Inclusiveness.index.2020)) +
  geom_point(aes(color = Continent)) +
  geom_smooth(method="lm", se=FALSE)
```

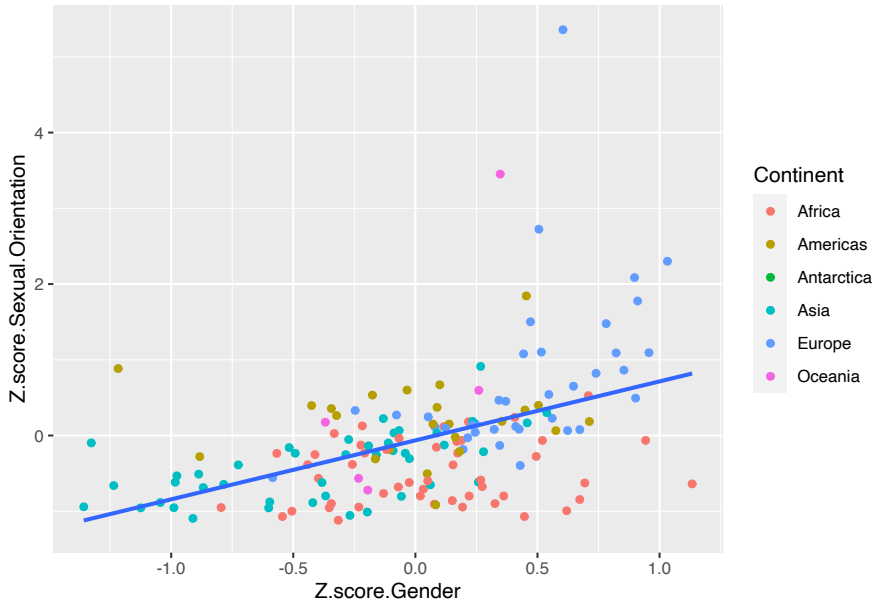
```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 117 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 117 rows containing missing values
## (geom_point).
```



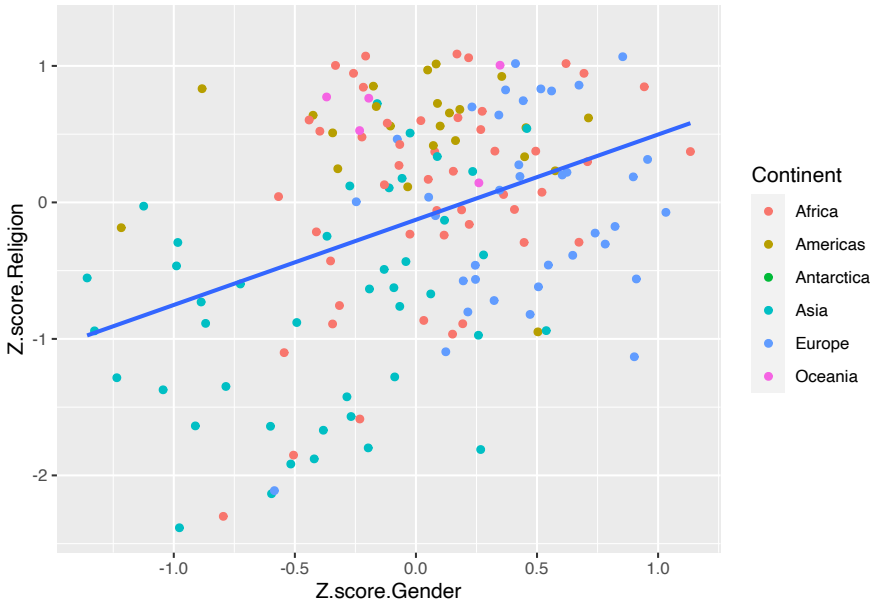

```
ggplot(inclusiveness_index,
       aes(x = Z.score.Gender,
           y = Z.score.Sexual.Orientation)) +
  geom_point(aes(color = Continent)) +
  geom_smooth(method="lm", se=FALSE)
```

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

## Warning: Removed 94 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 94 rows containing missing values
## (geom_point).
```




```
ggplot(inclusiveness_index,
       aes(x = Z.score.Gender,
           y = Z.score.Religion)) +
  geom_point(aes(color = Continent)) +
  geom_smooth(method="lm", se=FALSE)
```

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

## Warning: Removed 94 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 94 rows containing missing values
## (geom_point).
```

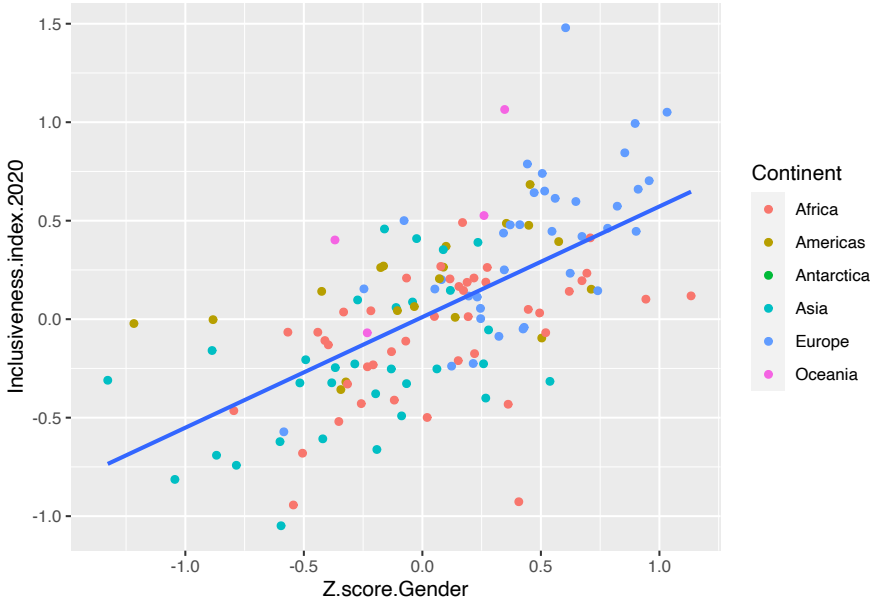



```
ggplot(inclusiveness_index,
       aes(x = Z.score.Gender,
           y = Inclusiveness.index.2020)) +
  geom_point(aes(color = Continent)) +
  geom_smooth(method="lm", se=FALSE)
```

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

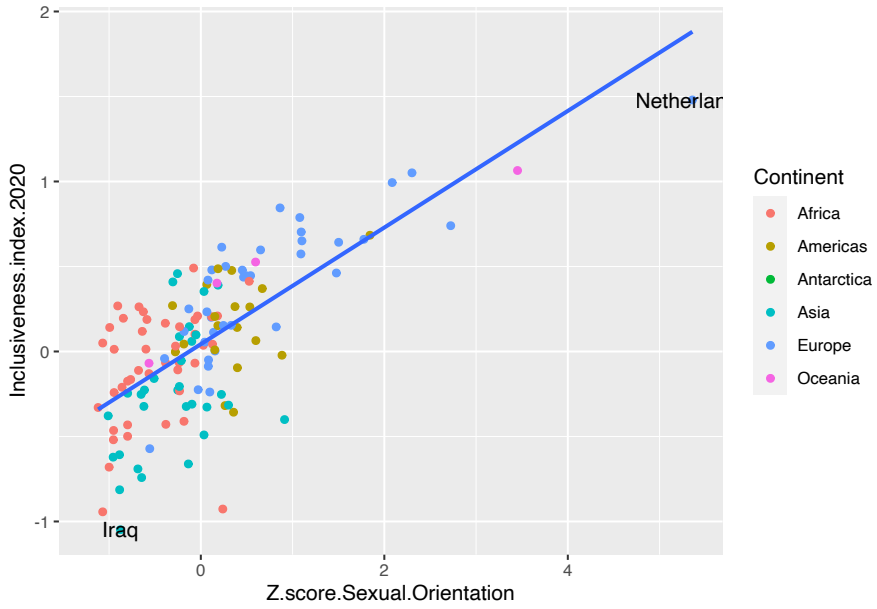
```
## Warning: Removed 117 rows containing non-finite values
## (stat_smooth).
```

```
## Warning: Removed 117 rows containing missing values
## (geom_point).
```



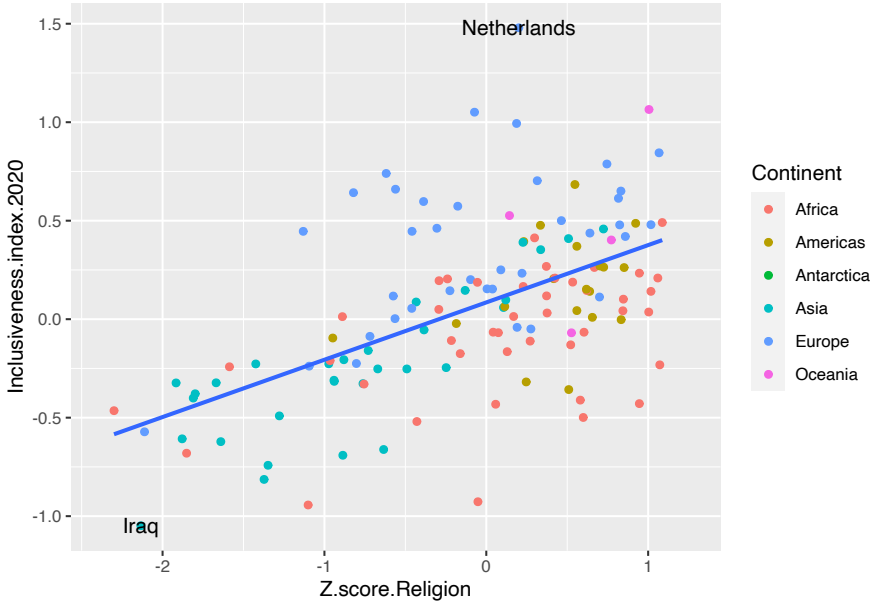

```
ggplot(inclusiveness_index,
       aes(x = Z.score.Sexual.Orientation,
           y = Inclusiveness.index.2020)) +
  geom_point(aes(color = Continent)) +
  geom_text(data = inclusiveness_index %>%
              dplyr::filter(Inclusiveness.index.2020 == max(Inclusiveness.index.2020, na.rm=TRUE)
  geom_smooth(method="lm", se=FALSE)
```

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

## Warning: Removed 117 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 117 rows containing missing values
## (geom_point).
```



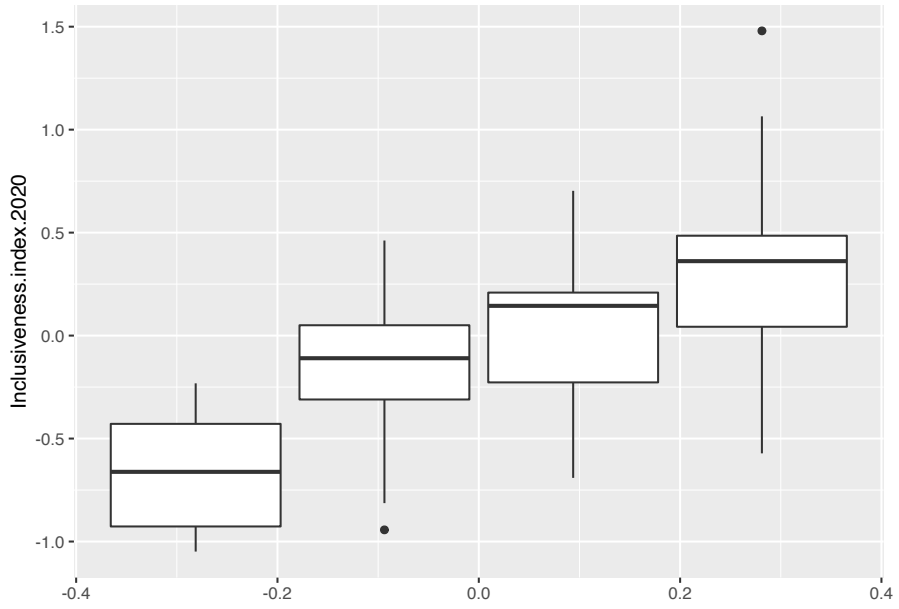

```
### USE THIS ONE? ###
ggplot(inclusiveness_index,
       aes(x = Z.score.Religion,
           y = Inclusiveness.index.2020)) +
  geom_point(aes(color = Continent)) +
  geom_text(data = inclusiveness_index %>%
              dplyr::filter(Inclusiveness.index.2020 == max(Inclusiveness.index.2020, na.rm=TRUE)
  geom_smooth(method="lm", se=FALSE)
```

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

## Warning: Removed 117 rows containing non-finite values
## (stat_smooth).

## Warning: Removed 117 rows containing missing values
## (geom_point).
```




```
ggplot(inclusiveness_index,
       aes(group = Z.score.Disability,
           y = Inclusiveness.index.2020)) +
  geom_boxplot()
```

```
## Warning: Removed 117 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