STAT 679 Exercises

Contents

1 Design Principles **1**

2 Visualization with R **5**

3 Visualization with D3 **21**

4 Spatial and Temporal Data **26**

5 Network and Hierarchical Data **34**

6 High-Dimensional and Text Data **36**

7 Model Visualization **42**

1 Design Principles

1. [Formulating Questions] Pick a dataset from TidyTuesday, Data Is Plural, Kaggle Datasets, Google Dataset Search, or your own personal studies that you would like to visualize this semester.
    a. What makes you interested in this dataset? Please provide a link, so we can use it to create future exercises.
    b. Using whatever software that you are familar with, prepare a visualization to answer one question you have based on the data. This will not be evaluated except for effort – the purpose of the problem is to create a record your data visualization skills at the start of the course. Submit both your visualization and code used to create it.
2. [Data Types] For the following parts, specify whether the data field is Nominal, Ordinal, or Quantitative.
    a. `BuildingArea` in the Melbourne Homes dataset
    b. `Sport` in the 2012 Olympics dataset
    c. `age_group` in the Language Learning dataset
3. [Critique - Infographic] Figure 1 below is from the Information is Beautiful Infographic "Food Waste is a Big Climate Problem We Can Actually Solve." It shows the breakdown of sources of food waste in the UK. Give an example of a question that this visualization helps answer. How easy / difficult is it to answer based on the current design? What is one design choice that you like? What is one thing you would do differently if you were redesigning it?
4. [Critique - Science Paper] Figure 2 below is from the scientific article *Both consumptive and non-consumptive effects of predators impact mosquito populations and have implications for disease transmission* by Russell et al. (2022). It describes the predators of mosquitos at different points in their life cycles. Give an example of a question that this visualization helps answer. How easy / difficult is it to answer based on the current design? What is one design choice that you like? What is one thing you would do differently if you were redesigning it?
5. [Critique - Science Paper] Figure 3 below is from the scientific article *Genetic basis and dual adaptive role of floral pigmentation in sunflowers* by Todesco et al. (2022). While all sunflower petals look the

Figure 1: Sources of food waste, from Information is Beautiful infographic.

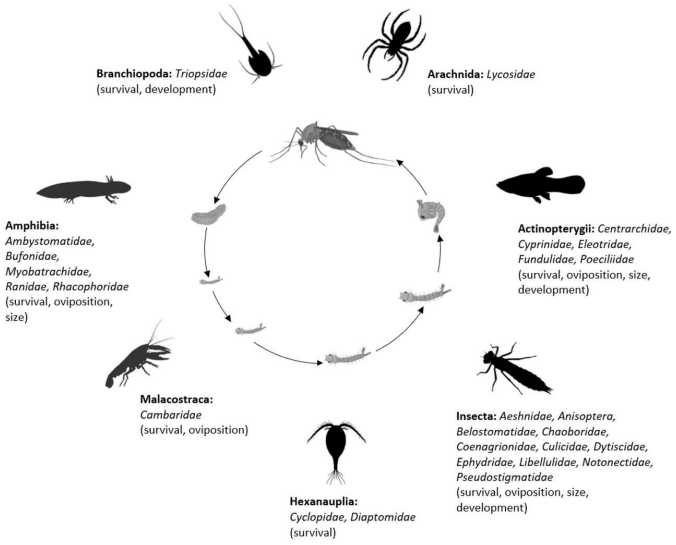


Figure 2: Animals that eat mosquitos, depending on the age of the mosquito, from Russell et al. (2022).

same to us (they look yellow), they actually vary quite a bit in how they reflect ultraviolet light, which is visible to pollinators. The figure studies the variation in this ultraviolet proportion (LUVp) across populations of sunflowers. Give an example of a question that this visualization helps answer. How easy / difficult is it to answer based on the current design? What is one design choice that you like? What is one thing you would do differently if you were redesigning it?

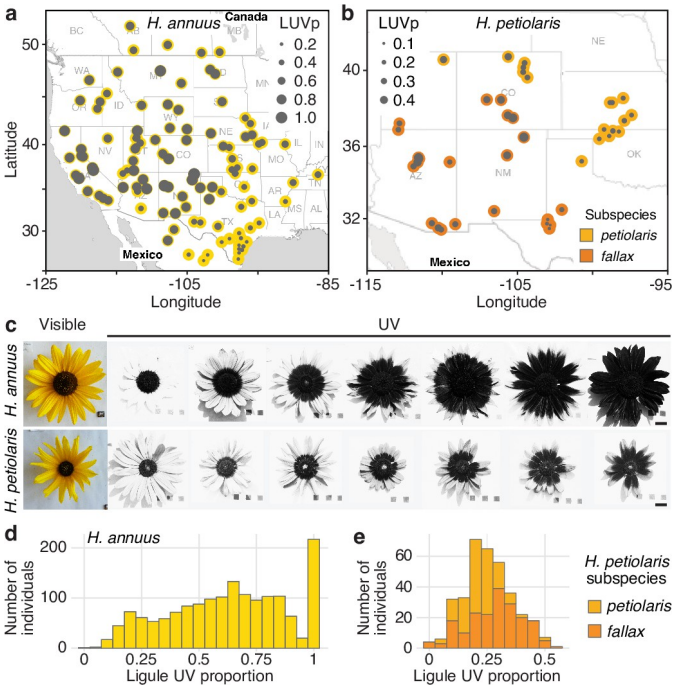


Figure 3: Variation in amount of ultraviolet proportion of sunflower petals, from Todesco et al. (2022).

6. [`ggplot2` reflection] Compare and contrast the `ggplot2` package with an alternative interface for constructing visualizations with which you have past experience (e.g., base R's `plot`, the `lattice` package, `matplotlib` in python, Tableau, or Excel). What characteristics are common between the two interfaces, and what is unique to each? In which situations does it make more sense to use one approach vs. the other?

7. [Beauty and functionality] There are a surprising number of controversies in data visualization, but few can get as heated as the beautify vs. functionality debate. In each pair of articles below, we have one representative from each school of thinking.

   - The Creative Pace of the 20th Century's Greatest Authors, Visualized vs. Redesigning Visualizations
   - Poor op/ed data graphic in New York Times vs. The Power of Visualization's "Aha!" Moments

   Respond to the positions taken by one of these pairs of articles (you are free to choose). Which of the arguments did you find most / least convincing? How might you evaluate the beauty or functionality of your own visualizations?

8. [Reading Response] This problem asks you to reflect on one of the readings from this week. Prepare a brief (1 paragraph) response. For example, you may discuss any of the following points,

   a. Are there points from the reading that you strongly agree or disagree with?
   b. Are there lessons from the reading that you think you might incorporate into your future projects or plans?
   c. How would you explain the main ideas of the reading to a friend who is not technically trained in visualization?

9. [Visual Redesign] In this exercise, you will find a visualization you have made in the past and redesign it using the skills you have learned in this course.

    a. Identify one of your past visualizations for which you still have data. Include a screenshot of this past visualization.
    b. Comment on the main takeaways from the visualization and the graphical relationships that lead to that conclusion. Is this takeaway consistent with the intended message? Are there important comparisons that you would like to highlight, but which are harder to make in the current design?
    c. Comment on the legibility of the original visualization. Are there aspects of the visualization that are cluttered or difficult to read?
    d. Propose and implement an alternative design. What visual tasks do you prioritize in the new design? Did you have to make any trade-offs? Did you make any changes specifically to improve legibility.

10. [Concept Map] Prepare a concept map to summarize the last week of reading material. An effective submission will include a thorough coverage of both abstract concepts and practical examples, summarized concisely into short phrases. It will also include well-justified links between nodes, with text explanations for edges whose interpretation might not be obvious at first.

11. [Mini-presentation] Prepare a short presentation to summarize the last week of reading material. Make sure to include coverage of both important concepts and practical implementation details. Prepare either a short set of slides or an original code notebook. When you present your material, make sure that all team members have an opportunity to speak.

12. [New Data Application] Create a version of a visualization included in this week's readings, lecture notes, or in-class exercises, but applied to new dataset of your choice.

    a. Provide a reference to the visualization that you will be adapting for your application.
    b. Identify a dataset that you will use. What makes this dataset interesting, and how was it gathered / generated?
    c. Generate a version of the visualization from (a) to the dataset from (b). Interpret the result in context.

13. [Transition Taxonomy] In "Animated Transitions in Statistical Graphics," Heer and Robertson introduce a taxonomy of visualizations transitions. These include,

    - View Transformation: We can move the "camera view" associated with a fixed visualization. This includes panning and zooming, for example.
    - Filtering: These transitions remove elements based on a user selection. For example, we may smoothly remove points in a scatterplot based on a dropdown menu selection.
    - Substrate Transformation: This changes the background context on which points lie. For example, we may choose to rescale the axis in a scatterplot to show a larger range.
    - Ordering: These transitions change the ordering of an ordinal variable. For example, we may transition between sorting rows of a heatmap alphabetically vs. by their row average.
    - Timestep: These transitions smoothly vary one plot to the corresponding plot at a different timestep. For example, we might show "slide" a time series to the left to introduce data for the most recent year.
    - Visualization Change: We may change the visual encoding used for a fixed dataset. For example, we may smoothly transition from a bar chart to a pie chart.
    - Data Scheme Change: This changes the features that are displayed. For example, we may smoothly turn a 1D point plot into a 2D scatterplot by introducing a new variable. In this problem, we will explore how these transitions arise in practice and explore how they may be implemented.

    a. Pick any visualization from the New York Times Upshot, Washington Post Visual Stories, the BBC Interactives and Graphics, or the Guardian Interactives pages. Describe two transitions that it implements. Of the 7 transition types given above, which is each one most similar to? Explain your choice.

b. For any transition (which may or may not be one of those you chose in (a)), identify the types of graphical marks used to represent the data. How would you create this type of mark in SVG?
c. To achieve the transition effect, how do you expect that the SVG elements would be modified / added / removed? Specifically, if elements are modified, what SVG `attrs` would be changed, and if elements are added or removed, how would the enter-exit-update pattern apply? You do not need to look at the code implementing the actual visualization, but you should give a plausible description of how the transition could be implemented in D3.

14. [Dissecting a visualization] Pick two static views of visualizations from any of the following sites: 1, 2, 3. For each view, answer the questions below,

    a. What graphical marks are used in the display? What data attribute does each mark encode?
    b. Are there visual queries that their design makes especially straightforward to answer? Are there queries that are more difficult relative to a simpler alternative?
    c. Why do you think the authors chose the visual encodings that they did?

15. [Analyzing a Demo] Watch one of the following interactive demos,

    or choose a comparable one from your own visualization interests.

    a. Analyze the transitions or interactivity are shared in the demo. Specifically, list all transition or interactive elements of the visualization and summarize the designer's motivation for including them.
    b. For one of the transition or interactivity elements, imagine an alternative design that accomplishes a similar purpose. Compare and contrast your proposed implementation with that in the original display.

16. [Guided literature search] This exercise will help you search for references to support the literature review for your course project.

    a. Reflect on the problem your project is trying to solve. List 6 - 10 data visualization related keywords that should bring up literature relevant to that problem. These keywords can be related to the type of data you have or the visual tasks your work is supposed to support.

    b. Use these keywords to search for 10 - 15 papers relevant papers. You may find it useful to (i) search using Google Scholar, (ii) skim the bibliographies of the first few papers you find that seem relevant (especially if they are review papers), (iii) browse abstracts in major journals / conferences (like TVCG, PacificVis, IEEEVis, EuroVis or VAST, or (iv) go to the home pages of researchers chairing conference sessions related to your keywords.

    c. Skim the abstracts from the papers on your list. Based on your quick reading, group your identified papers into overall themes. Prepare a short description of each theme. How are the groups of papers similar / different from one another?

17. [Code Analysis] Pick one code example from this week's lecture notes. Provide a 3 - 4 sentence summary of the overall implementation strategy. Then, add comments every 10 - 15 lines describing what each section of code is doing. If there are parts that you do not understand, add a ??? next to it (we will review these in the next lecture).

2 Visualization with R

1. [Ikea Furniture] The dataset below shows prices of pieces of Ikea furniture. We will compare prices of different furniture categories and label the (relatively few) articles which cannot be bought online.

```
ikea <- read_csv("https://uwmadison.box.com/shared/static/iat31h1wjg7abhd2889cput7k264bdzd.csv")
```

    a. Make a plot that shows the relationship between the `category` of furniture and `price` (on a log-scale). Show each `item_id` as a point – do not aggregate to boxplots or ridgelines – but

make sure to jitter and adjust the size the points to reduce the amount of overlap. *Hint: use the* `geom_jitter` *layer.*

b. Modify the plot in (a) so that categories are sorted from those with highest to lowest average prices.

c. Color points according to whether they can be purchased online. If they cannot be purchased online, add a text label giving the name of that item of furniture.

2. [City Temperatures] Let's create versions of Figure 2.3 and 2.4 from the reading this week. The command below reads in the data. We've filtered to a slightly different set of cities (Barrow is in Alaska, Honolulu is in Hawaii), but we should still be able to study changes in temperature over time.

```
temperature <- read_csv("https://raw.githubusercontent.com/krisrs1128/stat479_s22/main/data/tempera
```

a. Make a version of Figure 2.3 using a line mark (`geom_line`). Make at least one customization of the theme to make the plot more similar to the version in Figure 2.3. *Hint: To group the lines by city, use the* `group =` *aesthetic mapping.*

b. Using the `group_by` + `summarise` pattern, compute the mean temperature for each month in each city.

c. Using the data generated in (b), Make a version of Figure 2.4 using a tile mark (`geom_tile`). Try either (i) adding the `scale_fill_viridis_c(option = "magma")` scale to match the color scheme from the reading or (ii) adding `coord_fixed()` to make sure the marks are squares, not rectangles.

d. Compare and contrast the two displays. What types of comparisons are easier to make / what patterns are most readily visible using Figure 2.3 vs. Figure 2.4, and vice versa?

3. [Penguins] The data below measures properties of various Antarctic penguins.

```
penguins <- read_csv("https://uwmadison.box.com/shared/static/ijh7iipc9ect1jf0z8qa2n3j7dgem1gh.csv"
```

Create a single plot that makes it easy to answer both of these questions, (i) How is bill length related to bill depth within and across species? (ii) On which islands are which species found?

Read about Simpson's paradox and summarize it in your own words. Then, explain how part (i) provides a real-world example of this paradox.

4. [Student Exercise Responses] The data here give an (anonymized) summary of responses to the [Formulating Questions] exercise, given by students in Stat 679 / 992 in Fall 2022. It describes the dataset topic, its source, the submitted visualization type, and the software used to create it.

a. What types of topics seem to interest students in this class?

b. What types of visualization techniques and software to students seem most familiar with? How should teaching be adapted to reflect this?

c. When were submissions being made? Are the in-class exercises actually being solved in class?

5. [When2Meet Queries] In addition to being a scheduling tool, When2Meet polls provide a heatmap visualization of respondent availability. In this exercise, use the results from our office hour poll to evaluate the properties of this visualization.

a. What are two concrete questions that the heatmap visualization is effective at answering? Why do you think this?

b. What are two concrete questions that the heatmap visualization is *not* effective at answering? Why do you think this?

c. Describe an alternative static or interactive visual design that is better suited to one of the questions you identified in (b).

d. The dataset at this link include responses from our poll. Columns 1 - 35 represent students, and 1/0 denotes whether the student is or is not available. Implement a version of your proposed visual design from (c).

6. [London Olympics] The data at this link describes all participants in the London 2012 Olympics.

a. Create a layered display that shows (i) the ages of athletes across sports and (ii) the average age within each sport.
b. Sort the sports from lowest to highest average age.
c. Develop one new question based on these data. What makes you interested in it? Provide a visualization that supports the comparisons needed to arrive at an answer.

```
olympics <- read_csv("https://uwmadison.box.com/shared/static/rzw8h2x6dp5693gdbpgxaf2koqijo12l.csv"
```

7. [Language Learning] This problem will look at a simplified version of the data from the study *A critical period for second language acquisition: Evidence from 2/3 million English speakers*, which measured the effect of the the age of initial language learning on performance in grammar quizzes. We have downloaded the data from the supplementary material and reduced it down to the average and standard deviations of test scores within (initial learning age, the `Eng_start` variable) $\times$ (current age-group, the `age_group` variable) combinations. We have kept a column `n` showing how many participants were used to compute the associated statistics. The resulting data are available here.

   a. Define two new fields, `low` and `high`, giving confidence intervals for the means in each row. That is, derive new variables according to $\hat{x} \pm 2 * \frac{1}{\sqrt{n}}\hat{\sigma}$.

   b. Design and implement a visualization that helps answer the following questions,

   - How does test score vary as a function of current age group?
   - How does test score vary as a function of initial learning age? Is this relationship the same across all age groups?
   - For which populations are we least certain about true language learning ability?

   c. Provide answers for each question from (b) within context.

8. [Homelessness] Take a static screenshot from any of the visualizations in this article, and deconstruct its associated visual encodings.

   a. What do you think was the underlying data behind the current view? What where the rows, and what were the columns?
   b. What were the data types of each of the columns?
   c. What encodings were used? How are properties of marks on the page derived from the underlying abstract data?
   d. Is multi-view composition being used? If so, how?

9. [Plant Growth Experiment] This problem will give you practice with tidying a dataset so that it can be easily visualized. The data describe the height of several plants measured every 7 days. The plants have been treated with different amounts of a growth stimulant. The first few rows are printed below – `height.x` denotes the height of the plant on day `x`.

```
plants <- read_csv("https://uwmadison.box.com/shared/static/qg9gwk2ldjdtcmmmiropcunf34ddonya.csv")
```

   a. Propose an alternative arrangement of rows and columns that conforms to the tidy data principle.

   b. Implement your proposed arrangement from part (a).

   c. Using the dataset from (b), design and implement a visualization showing the growth of the plants over time according to different treatments.

10. [California Wildfires] In this problem, we will interactively visualize a dataset giving statistics of recent California wildfires. The steps below guide you through the process of building this visualization.

    a. (Static) Plot the day of the year that each fire started against the county within which it was located. Use the size of the mark to encode the number of acres burned in each fire.

    b. (Interactive) Provide a way for the viewer to interactively highlight or reveal details about subsets of fires. For example, introduce a slider to interactively highlight selected years, a tooltip to

highlight the name of a fire, or a select to search for counties, or a slider to filter by fire size.

c. Introduce at least one other UI output. For example, print a table of the selected fires, interactively print summary statistics, or show a histogram of fire sizes. Sketch the reactivity graph associated with your application.

11. [Pokemon] This problem gives practice in deriving new variables to improve a faceted plot. The data below give attack and defense statistics for Pokemon, along with their types. We will build a visualization to answer the question – how do the different types of Pokemon vary in their attack and defense potential?

```
pokemon <- read_csv("https://uwmadison.box.com/shared/static/hf5cmx3ew3ch0v6t0c2x56838er1lt2c.csv")
```

a. Derive a new column containing the attack-to-defense ratio, defined as $\frac{\text{Attack}}{\text{Defense}}$.

b. For each `type_1` group of Pokemon, compute the median attack-to-defense ratio.

c. Plot the attack vs. defense scores for each Pokemon, faceted by `type_1`. Use the result of (b) to ensure that the panels are sorted from types with highest to lowest attack-to-defense ratio.

d. Propose, but do not implement, a visualization of this dataset that makes use of dynamic queries. What questions would the visualization answer? What would be the structure of interaction, and how would the display update when the user provides a cue?

12. [Melbourne Missingness] Consider the plot of missingness in the Melbourne Homes dataset in Figure 4. Select all the true conclusions.

- The `Landsize` variable is always missing.
- The variable with the highest number of missing values is `BuildingArea`.
- There are 560 rows where exactly one of `CouncilArea`, `YearBuilt`, or `BuildingArea` are missing.
- In more than 2/3 of the rows where `BuildingArea` is missing, `YearBuilt` is also missing.

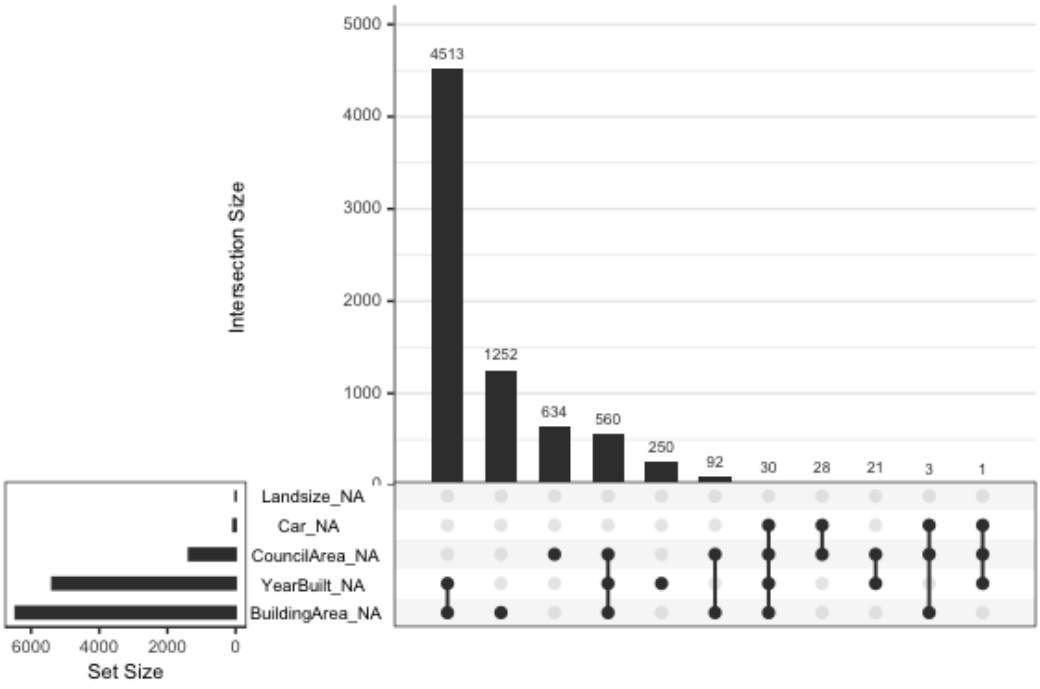


Figure 4: Upset plot of missingness in the Melbourne Homes dataset.

13. Study the visualization at this link. Its design reflects which of the following visualization techniques? (select all that apply).

    - Faceting
    - Details-on-demand
    - Ridgelines
    - Overview + detail
    - Tidy data

14. [Yu-Gi-Oh!] This question asks you to tidy a Yi-Gi-Oh! dataset.

    a. Write code that provides more meaningful attribute and type columns. The result of your transformation should look like this.
    b. Count the number of attribute and type combinations. Derive new features that count the fraction of each combination that are on the trading and official card game banlists (Banned trading and official cards have non-NA values in the tcg_ban and ocg_ban columns, respectively). The result of your transformation should look like this.
    c. Reshape the dataset so that the trading and official card game proportions are listed in a single column and the Spell and Trap cards are at the top. The result of your transformation should look like this.

15. [Reshaping practice] The following parts ask you to provide R code that transforms one dataset (I) into another (II).

    a. Regional murder rates: Turn I into II.
    b. Antiobitics and bacteria. Turn I into II

16. [Sales summaries] Consider the sales data below.

```
##   region quarter sales
## 1      A      Q1     6
## 2      A      Q2     5
## 3      A      Q3     3
## 4      A      Q4     2
## 5      B      Q1     4
## 6      B      Q2     8
## 7      B      Q3     2
## 8      B      Q4     6
```

    a. Provide code to compute the total sales for each quarter, across both regions. The result should look like the table below.

```
## # A tibble: 4 x 2
##   quarter total
##   <chr>   <dbl>
## 1 Q1         10
## 2 Q2         13
## 3 Q3          5
## 4 Q4          8
```

    b. Provide code to compute the proportion of each quarter's sales that came from each region. The result should look like the table below.

```
## # A tibble: 8 x 4
##   region quarter sales  prop
##   <chr>  <chr>   <dbl> <dbl>
## 1 A      Q1          6 0.6
## 2 A      Q2          5 0.385
## 3 A      Q3          3 0.6
```

```
## 4 A       Q4           2 0.25
## 5 B       Q1           4 0.4
## 6 B       Q2           8 0.615
## 7 B       Q3           2 0.4
## 8 B       Q4           6 0.75
```

17. [Interactive Penguins] This problem visualizes the Penguins dataset interactively.

    a. Create two static scatterplots: one of of bill length vs. depth and another of flipper length vs. body mass.
    b. Create selection menus that can be used to highlight the island that the penguin came from, along with its gender.
    c. Create selection brushes that can be used to link the two plots together. That is, highlighting points in a brush for one view should highlight the corresponding penguins in the other.
    d. Provide one takeaway from the interactive visualization that is not possible to make using the original static version.

18. [Olympics Derivations] This problem gives some practice with deriving and visualizing new variables.

    a. Create new columns for the city and country of birth for each athlete in the London 2012 Olympics dataset.
    b. Compute the standard deviation of athlete age within each sport. Which sport has widest SD in age?
    c. Make a visualization of sports against participant age. Sort sports by age variance.

```
olympics <- read_csv("https://uwmadison.box.com/shared/static/rzw8h2x6dp5693gdbpgxaf2koqijo12l.csv"
```

19. The following questions refer to the NYC flights dataset. The first few lines are printed below.

```
library(nycflights13)
flights %>% select(carrier, air_time, distance)
```

```
## # A tibble: 336,776 x 3
##    carrier air_time distance
##    <chr>      <dbl>    <dbl>
##  1 UA           227     1400
##  2 UA           227     1416
##  3 AA           160     1089
##  4 B6           183     1576
##  5 DL           116      762
##  6 UA           150      719
##  7 B6           158     1065
##  8 EV            53      229
##  9 B6           140      944
## 10 AA           138      733
## # ... with 336,766 more rows
```

    a. Provide code to create a new column giving the average speed of the flight: `speed` := $\frac{\texttt{distance}}{\texttt{air\_time}}$.
    b. Is there a large variation in flight speed across carriers? Design and sketch code for a visualization that could be used to answer this question (you may assume the output of (a)).

20. [Birds Code Review] This exercise asks you to conduct an imaginary code review. These are often used in data science teams to,

    - Catch potential bugs
    - Make sure code is transparent to others
    - Create a shared knowledge base

    It is important to be perceptive but friendly.

- Can the code be made more compact?
- Are there visual design choices / encodings that could be refined?
- If your colleague did something well, say so!

They can also be a great way to learn new functions and programming patterns. Unlike standard code-reviews, I ask you to give an example implementing your recommendations.

Specifically, in this review, suppose you are working with a scientific colleague on your study of bird egg properties and their evolutionary implications. They have written the code below. Provide your code review as a set of bullet points, and include code giving an example implementation of your ideas. The original data are from this study.

```
library(ggrepel)

birds <- read_csv("https://raw.githubusercontent.com/krisrs1128/stat479_s22/main/exercises/data/bir
  separate(Species, c("genus", "species2"))

bird_summaries <- birds %>%
  group_by(genus) %>%
  summarise(across(c("Asymmetry", "Ellipticity", "AvgLength (cm)", "Number of images", "Number of e
  arrange(-Ellipticity_MEAN)

ggplot(bird_summaries) +
  geom_point(aes(Asymmetry_MEAN, Ellipticity_MEAN)) +
  geom_text_repel(aes(Asymmetry_MEAN, Ellipticity_MEAN, label = genus))
```

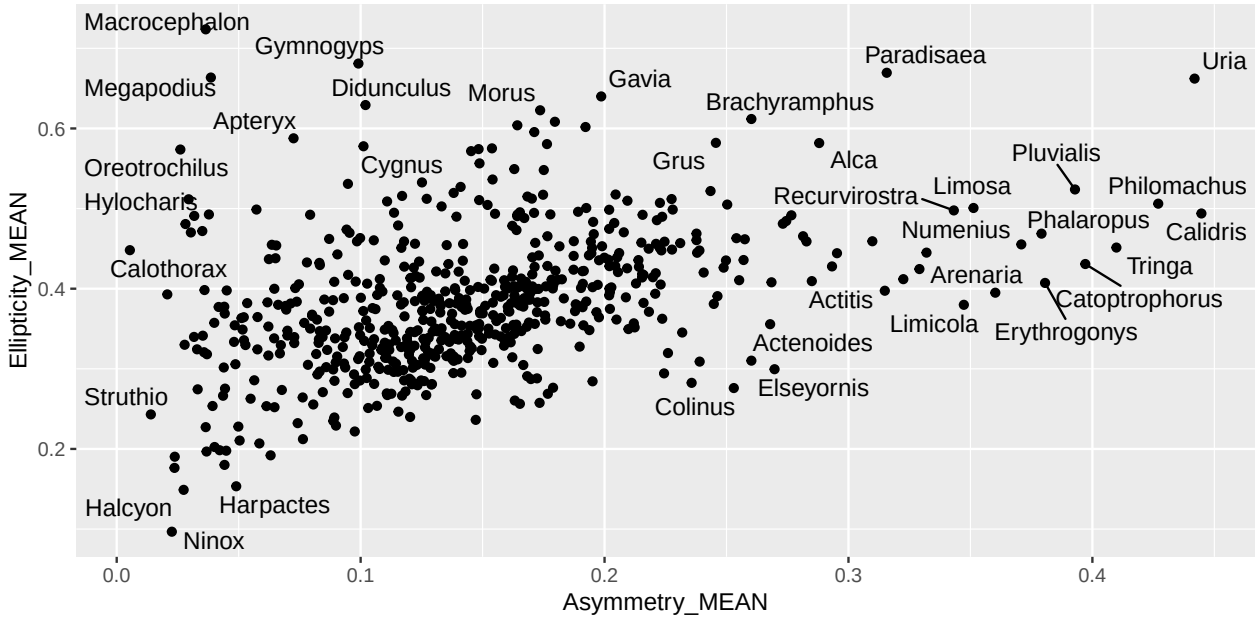


Figure 5: An example figure for code review.

21. [Soccer Code Review] This exercise asks you to conduct an imaginary code review. These are often used in data science teams to,

    - Catch potential bugs
    - Make sure code is transparent to others
    - Create a shared knowledge base

    It is important to be perceptive but friendly.

- Can the code be made more compact?
- Are there visual design choices / encodings that could be refined?
- If your colleague did something well, say so!

They can also be a great way to learn new functions and programming patterns. Unlike standard code-reviews, I ask you to give an example implementing your recommendations.

Specifically, in this review, suppose you are working on a sports blog, and your colleague is soccer interested in which teams won the most games in a few European leagues over the last few years. They have written the code below. Provide your code review as a set of bullet points, and include code giving an example implementation of your ideas. The original data are from this link.

```
win_props <- read_csv("https://raw.githubusercontent.com/krisrs1128/stat479_s22/main/exercises/data
  group_by(team, year) %>%
  summarise(n_games = n(), wins = sum(wins) / n_games)

best_teams <- win_props %>%
  ungroup() %>%
  slice_max(wins, prop = 0.2) %>%
  pull(team)

win_props %>%
  filter(team %in% best_teams) %>%
  ggplot() +
  geom_point(aes(year, team, size = n_games, alpha = wins))
```

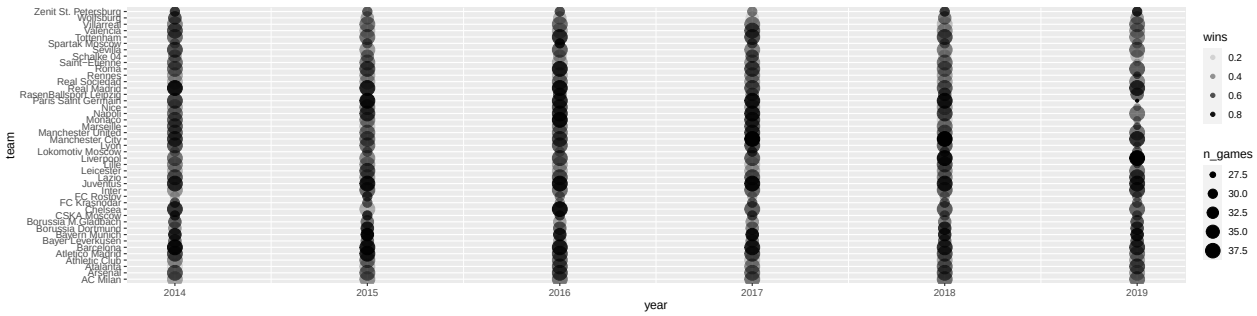


Figure 6: An example figure for code review.

22. [Multiple Weather Variables] In this problem, we will construct a compound figure to simultaneously visualize several weather variables in one display. The block below reads a dataset that includes temperature, precipitation, and wind measurements for New York City and Seattle over the course of several years. It then then averages each variable for each day (1 - 365) in the year.

```
weather <- read_csv("https://raw.githubusercontent.com/krisrs1128/stat479_s22/main/exercises/data/w
  mutate(day_in_year = lubridate::yday(date)) %>%
  group_by(location, day_in_year) %>%
  summarise(across(precipitation:wind, mean))
```

   a. Construct three base `ggplot2` plots based on the variables that are provided. For example, you may construct a line plot (`geom_line`) of average precipitation, a histogram (`geom_histogram`) of per-city windspeeds, or a ribbon plot (`geom_ribbon`) of temperature ranges over time. Make sure to display at least two weather variables from among the four that are provided.
   b. Design a compound figure based on the base plots in part (a). Why did you lay out the subplots in the locations that you did? Ensure that consistent visual encodings are used throughout, that legends are collected, and that clear but unobtrusive annotation is provided.

c. Discuss, but do not implement, an alternative compound figure for the same data, with either different base plots or overall layout (or both). For which visual comparisons are either designs best suited?

23. [Faceting rules-of-thumb] Circle all the true statements for compound and faceted figures.

    a. Whenever possible, each panel in a faceted plot should be made to have its own y-axis scale.

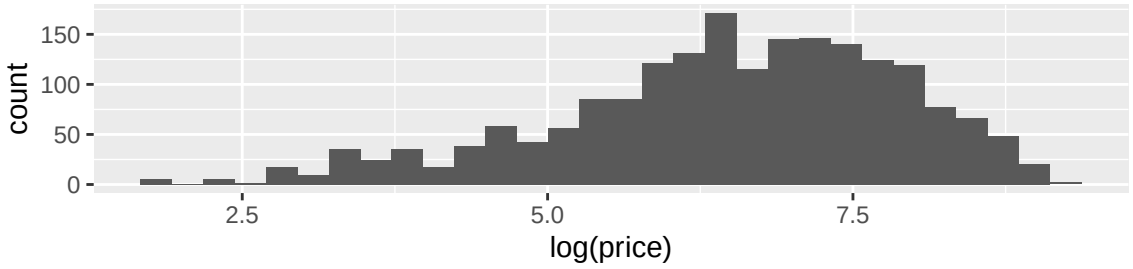
    b. When using the `patchwork` package, two plots `p1` and `p2` can be placed side-by-side using `p1 / p2`.

    c. When using the `patchwork` package, the `plot_layout` function can be used to collect legends.

    d. Relative to interactively filtered displays, faceted plots are less taxing on the reader's memory.

24. [Ikea Faceting] The code below generates a histogram of prices for a subset of items sold at Ikea, a furniture outlet.

```
ikea <- read_csv("https://uwmadison.box.com/shared/static/iat31h1wjg7abhd2889cput7k264bdzd.csv") %>
  filter(category %in% c("Tables", "Bookcases", "Beds", "Cabinets", "Sofas", "Wardrobes"))
ggplot(ikea) +
  geom_histogram(aes(log(price)))
```



The four plots (i - iv) are generated by faceting this base plot by furniture category (the variable `category`). On the blank lines below, put the number of the plot that matches the corresponding faceting command, or leave the line empty if the corresponding plot does not appear.

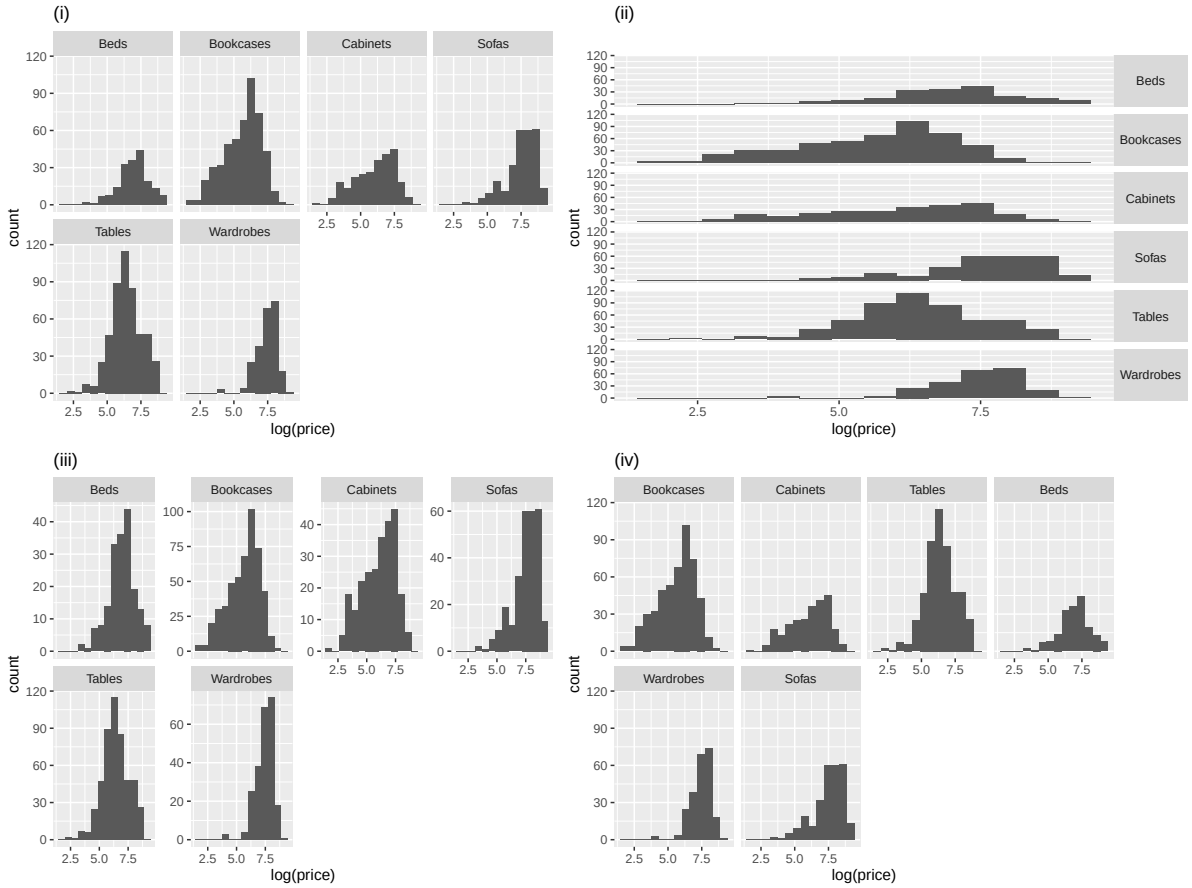
___ a. `facet_grid(category ~ .)`

___ b. `facet_wrap(~ category)`

___ c. `facet_wrap(~ category, scales = "free_y")`

___ d. `facet_wrap(~ category, axes = "separate")`

___ e. `facet_wrap(~ reorder(category, price))`



25. [Antibiotics Comparison] Below, we provide three approaches to visualizing species abundance over time in an antibiotics dataset.

```
antibiotic <- read_csv("https://uwmadison.box.com/shared/static/5jmd9pku62291ek20lioevsw1c588ahx.cs
antibiotic
```

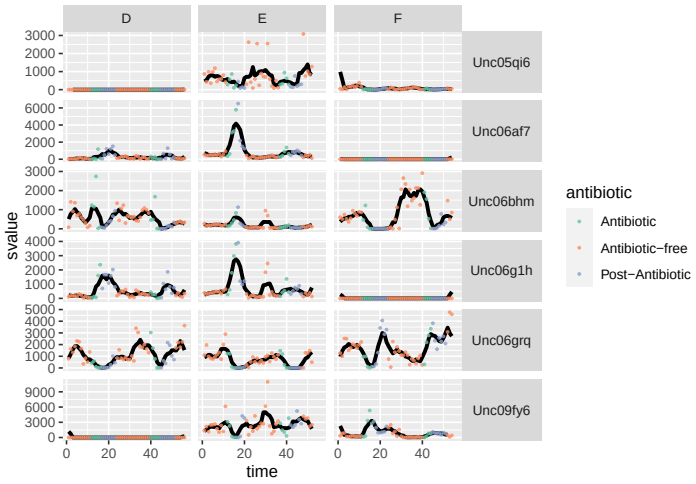
```
## # A tibble: 972 x 7
##    species  sample value ind    time svalue antibiotic
##    <chr>    <chr>  <dbl> <chr> <dbl>  <dbl> <chr>
##  1 Unc05qi6 D1         0 D         1   NA   Antibiotic-free
##  2 Unc05qi6 D2         0 D         2   NA   Antibiotic-free
##  3 Unc05qi6 D3         0 D         3    0   Antibiotic-free
##  4 Unc05qi6 D4         0 D         4    0   Antibiotic-free
##  5 Unc05qi6 D5         0 D         5    0   Antibiotic-free
##  6 Unc05qi6 D6         0 D         6    0.2 Antibiotic-free
##  7 Unc05qi6 D7         0 D         7    0.2 Antibiotic-free
##  8 Unc05qi6 D8         1 D         8    0.2 Antibiotic-free
##  9 Unc05qi6 D9         0 D         9    0.2 Antibiotic-free
## 10 Unc05qi6 D10        0 D        10    0.2 Antibiotic-free
## # ... with 962 more rows
```

For each approach, describe,

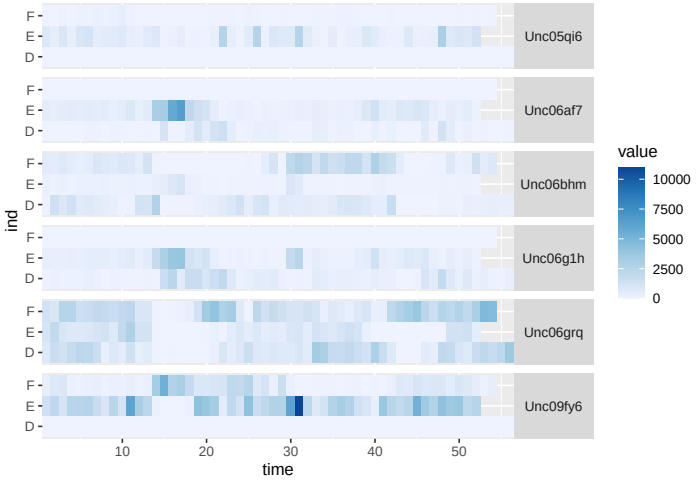
- One type of visual comparison for which the visualization is well-suited.
- One type of visual comparison for which the visualization is poorly-suited.

Make sure to explain your reasoning.

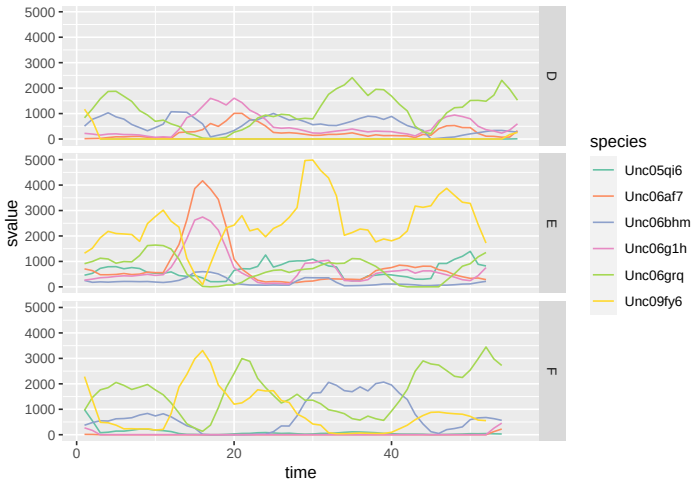
a. Approach 1



b. Approach 2



c. Approach 3



d. Sketch code that could be used to make one of the three visualizations above.

26. [Name app bugs] The following versions of the Name app all have errors that will raise an error if you

try running them in R. For each part, isolate the line(s) that contain the bug. Provide a conceptual explanation for why the error occurred and how they could be prevented more generally.

a. Program (a).

```
library(shiny)

ui <- fluidPage(
  titlePanel("Hello!"),
  textInput("name", "Enter your name"),
  textOutput("printed_name")
)

server <- function(input, output) {
  input$name <- paste0("Welcome to shiny, ", input$name, "!")
  output$printed_names <- renderText({ input$name })
}

app <- shinyApp(ui, server)
```

b. Program (b).

```
library(shiny)

ui <- fluidPage(
  titlePanel("Hello!"),
  textInput("name", "Enter your name"),
  textOutput("printed_name")
)

server <- function(input, output) {
  output$printed_names <- renderText({
    paste0("Welcome to shiny, ", input$name, "!")
  })
}

app <- shinyApp(ui, server)
```

c. Program (c).

```
library(shiny)

ui <- fluidPage(
  titlePanel("Hello!"),
  textInput("name", "Enter your name"),
  textOutput("printed_name")
)

server <- function(input, output) {
  output$printed_name <- renderDataTable({
    paste0("Welcome to shiny, ", input$name, "!")
  })
}

app <- shinyApp(ui, server)
```

d. Program (d).

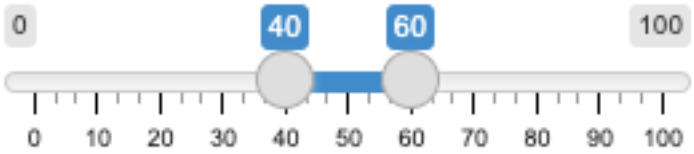
```r
library(shiny)

ui <- fluidPage(
  titlePanel("Hello!")
  textInput("name", "Enter your name")
  textOutput("printed_name")
)

server <- function(input, output) {
  output$printed_name <- renderText({
    paste0("Welcome to shiny, ", input$name, "!")
  })
}

app <- shinyApp(ui, server)
```

27. [UI Options] Look into the Shiny and shinyWidgets documentation to determine which UI functions can be used to generate the types of inputs described below. Example figures are also shown. Provide a minimal app with similar input functionality (just leave the server empty).

    a. An input to select a date range.

    b. A yes / no toggle that uses an icon for the options.

    c. A slider that can be used to select a range of values.



28. [Improving an app] Make the following apps more concise, modular, and readable by using reactive expressions to capture duplicated computation and / or externalizing complex computations into functions which are defined outside of the server.

    a. Program (a)

```r
library(shiny)

ui <- fluidPage(
  titlePanel("Calculator"),
  numericInput("x", "Enter the value of x", 0),
  textOutput("f1"),
  textOutput("f2"),
```

```
  textOutput("f3")
)

server <- function(input, output) {
  output$f1 <- renderText({ 3 * input$x ^ 2 + 3})
  output$f2 <- renderText({ sqrt(3 * input$x ^ 2 + 3) - 5})
  output$f3 <- renderText({ 30 * input$x ^ 2 + 30})
}

shinyApp(ui, server)
```

b. Program (b)

```
penguins <- read_csv("https://uwmadison.box.com/shared/static/ijh7iipc9ect1jf0z8qa2n3j7dgem1gh.csv"
islands <- unique(penguins$island)
species <- unique(penguins$species)

ui <- fluidPage(
  titlePanel("Penguins Plot"),
  selectInput("species", "Species", species, multiple = TRUE),
  selectInput("island", "Island", islands, multiple = TRUE),
  selectInput("var1", "First Variable", colnames(penguins)[3:6]),
  selectInput("var2", "Second Variable", colnames(penguins)[3:6]),
  plotOutput("scatterplot"),
  plotOutput("histogram1"),
  plotOutput("histogram2"),
)

server <- function(input, output) {
  output$scatterplot <- renderPlot({
    current_data <- penguins %>%
      filter(
        island %in% input$island,
        species %in% input$species
      )
    ggplot(current_data) +
      geom_point(aes(.data[[input$var1]], .data[[input$var2]]))
  })

  output$histogram1 <- renderPlot({
    current_data <- penguins %>%
      filter(
        island %in% input$island,
        species %in% input$species
      )
    ggplot(current_data) +
      geom_histogram(aes(.data[[input$var1]]))
  })

  output$histogram2 <- renderPlot({
    current_data <- penguins %>%
      filter(
        island %in% input$island,
        species %in% input$species
```

```
      )
    ggplot(current_data) +
      geom_histogram(aes(.data[[input$var2]]))
  })

}

shinyApp(ui, server)
```

29. [Reactivity Graphs] Manually sketch out the reactivity graph associated with any of the "Shiny Demos" in the official Shiny Gallery. Make sure to distinguish between input, output, observe, and reactive nodes. For a challenge, try the same exercise for one of the apps in the Shiny User Showcase. *Note: In many cases, the UI and server functions have been split into two files, `ui.R` and `server.R`. The code can nonetheless be read as if the functions were in a single `app.R` file.*

30. [Olympics Interactive App] The code below sets up a Shiny app for interactively visualizing athlete weight and heights in the 2012 London Olympics. We would like to have an interactive scatterplot of `Weight` vs. `Height, cm` that updates which points (athletes) are highlighted depending on which sports have been selected by a dropdown menu. Code for generating the scatterplot is provided in the function `scatterplot`.

```
library(shiny)
library(tidyverse)
olympics <- read_csv("https://uwmadison.box.com/shared/static/rzw8h2x6dp5693gdbpgxaf2koqijo12l.csv"

#' Scatterplot with highlighted points
#'
#' Assumes a column in df called "selected" saying whether points should be
#' larger / darker
scatterplot <- function(df) {
  ggplot(df) +
    geom_point(
      aes(Weight, `Height, cm`,
          alpha = as.numeric(selected),
          size = as.numeric(selected))
      ) +
    scale_alpha(range = c(0.05, .8)) +
    scale_size(range = c(0.1, 1))
}

ui <- fluidPage(
  selectInput("dropdown", "Select a Sport", choices = unique(olympics$Sport), multiple = TRUE),
  plotOutput("scatterplot")
)

server <- function(input, output) {
  # fill this in...
}
```

    a. Provide server code which would allow the scatterplot to update the highlighted athletes depending on the currently selected sports.

    b. We have been asked to also print a table of the selected athletes. Assume the UI has the form,

```
ui <- fluidPage(
  selectInput("dropdown", "Select a Sport", choices = unique(olympics$Sport), multiple = TRUE),
```

```
  plotOutput("scatterplot"),
  dataTableOutput("table")
)
```

Describe changes to your solution to (a) to meet the new requirements. How would you minimize code duplication? Be as specific as possible.

31. [NYC Rentals] In this problem, we'll create a visualization to dynamically query a dataset of Airbnb rentals in Manhattan in 2019. The steps below guide you through the process of building this visualization.

    a. Make a scatterplot of locations (Longitude vs. Latitude) for all the rentals, colored in by `room_type`.

    b. Design a plot and a dynamic query so that clicking or brushing on the plot updates the points that are highlighted in the scatterplot in (a). For example, you may query a histogram of prices to focus on neighborhoods that are more or less affordable.

    c. Implement the reverse graphical query. That is, allow the user to update the plot in (b) by brushing over the scatterplot in (a).

    d. Comment on the resulting visualization(s). If you had a friend who was interested in renting an Airbnb in NYC, what would you tell them?

32. [Birds Brushed Scatterplot]. The `birds` and `bird_summaries` datasets defined below give characteristics of the eggs laid by different species of birds. In this problem, you will build an interactive app to compare egg `Asymmetry` and `Ellipticity` across species and genera.

```
birds <- read_csv("https://raw.githubusercontent.com/krisrs1128/stat479_s22/main/exercises/data/bir
  separate(Species, c("genus", "species2"))

bird_summaries <- birds %>%
  group_by(genus) %>%
  summarise(across(c("Asymmetry", "Ellipticity", "AvgLength (cm)", "Number of images", "Number of e
  arrange(-Ellipticity_MEAN)
```

    a. Create a scatterplot of `Asymmetry` vs. `Ellipticity` which supports brushing. Specifically, whenever the user brushes over a set of points in the scatterplot, they should be shown a table of the selected species. Also, change the color of the currently brushed points, so that they stand out more clearly.

    b. Design and implement an additional genus-level graphical input using the `bird_summaries` data. For example, you could allow the user to highlight species belonging to taxa with high average egg asymmetry.

33. [Shiny free exploration] Pick one of the following datasets, or propose one of your own interest: London Olympics, Pokemon, Chicago Traffic Accidents, Australian Pharmaceuticals.

    a. Based on your understanding of how the data were collected and what they represent, prepare four questions that you are interested in. Choose questions that reflect different aspects of the data and whose answers will require more nuanced analysis.

    b. Design and implement a Shiny app whose interactivity helps to answer the questions you identified in (a).

    c. Sketch the reactivity graph of your app's final implementation.

34. [Reactivity Discussion] Consider the `reactive()` and `observeEvent()` functions implemented by the `shiny` package. What are common use cases for the two functions? Be as specific as possible.

3 Visualization with D3

1. [Javascript Array Manipulation] These are some exercises to prepare you for more complex array and object manipulation that you may need in more advanced D3 visualizations. You will likely find the following resources useful: [1, 2, 3, 4].

   a. Find the minimum, maximum, and mean of the following array.

```
let x = d3.randomUniform(100);
```

   b. Extract the first 5 elements of `x` and then concatenate them to the last 5 elements of `y`. Your result should look like `[0, 1, 2, 3, 4, 15, 16, 17, 18, 19]`.

```
let x = d3.arrange(100),
    y = d3.arrange(20);
```

   c. Given the object below, how would you create the array `['apple', 'orange', 'banana']` of just the object's keys?

```
let fruits = {apple: 32, orange: 14, banana: 10}
```

   d. Repeat (c), but to create the array [32, 14, 10] of just the object's values.

   e. Given an array, how would you create a new array of just its unique values? For example, given an array `[1, 3, 3, 3, 4, 1, 2, 1]`, how would you create `[1, 3, 4, 2]`. Your solution doesn't need to preserve the order of appearance in the original array.

2. [Penguins Array Manipulation] We will use the `penguins` dataset to practice common types of javascript array operations. The javascript file here defines two objects, `penguins` and `penguins2`, each storing the first 10 rows of the penguins dataset, but in slightly different formats.

   a. For each of `penguins` and `penguins2`, describe whether the data structure is (i) an object of arrays, (ii) an array of objects, or (iii) neither. In case (iii), specify the structure.

   b. Using `penguins2`, compute the mean flipper length.

   c. Using `penguins`, define a new array whose elements are `true` or `false` depending on whether flipper length is larger than 187.

   d. Create a new array containing all information from just two penguins: The ones with the smallest and largest flipper lengths.

3. [Draw a Smiley Face] This exercise gives practice in drawing SVG elements from scratch and styling them.

   a. Using only SVG elements, draw a black-and-white smiley face.

   b. Associate each of your SVG elements with either a class or an ID. Use these newly defined classes / IDs together with a CSS file to style your SVG elements.

   c. Using d3's `clone` function, create 50 copies of your smiley face, located at random locations on the SVG canvas.

4. [Modifying Selections] This exercise gives practice making selections on an HTML page and applying changes using either CSS or D3. We will work with this HTML page, which contains a variety of SVG elements grouped and labeled according to different classes and IDs.

   a. Draw the HTML tree structure of the provided page.

   b. Using a CSS rule, change all circles to blue.

   c. Using D3, move the circle with the ID `part_c` to location (100, 100)

   d. Using D3, change all the squares of class `part_d` to red.

   e. Using CSS or D3, change all circles of class `part_e` to red, but only if they are contained within the group with label `group_part_e`.

5. [Debugging JS Code] Imagine that you are helping your colleagues debug their javascript programs. They have explained their implementation goals and the difficulties they have encountered. Your job is

to both help them resolve their specific bug and to briefly explain the concepts that will help them avoid these issues in the future. Specifically, for each part, explain propose a solution to your colleague's problem and briefly explain the concept that will help avoid similar bugs.

a. Goal: I want to try to get the average of the "temp_max" array in "forecast" below.

Attempt: Here is my HTML and JS file. This is the part that I think is relevant,

```
let forecast = {temp_max: [22, 24, 28, 21], temp_min: [15, 14, 21, 24]}
d3.mean(forecast[0])
```

Problem: The console prints the error message, `Uncaught TypeError: t is not iterable at t.mean (d3-array@3:2:13861) at in_class4-2a.js:3:4`

b. Goal: I want to change the color of all the circles on my HTML page to blue.

Attempt: Here is my HTML and JS file. This is the part that I think is relevant,

```
d3.select("circle")
  .attr("fill", "blue")
```

Problem: Only the color of the one circle is changing.

c. Goal: I want to change the color of all the class "highlighted" squares.

Attempt: Here is my HTML and JS file. This is the part that I think is relevant,

```
d3.selectAll("highlighted")
  .attr("fill", "red")
```

Problem: Nothing is changing.

d. Goal: I want to move the circle to location 100, 200 on the SVG canvas.

Attempt: Here is my HTML and JS file. This is the part that I think is relevant,

```
d3.select("#my_circle")
  .attrs({ x: 100, y: 200 })
```

Problem: It is not moving.

6. [Random Point Transitions] This exercise will give practice implementing transitions on simulated data. The code below generates a random set of 10 numbers,

```
let generator = d3.randomUniform();
let x = generator(10);
```

   a. Encode the data in `x` using the x-coordinate positions of 10 circles.
   b. Animate the circles. Specifically, at fixed time intervals, generate a new set of 10 numbers, and smoothly transition the original set of circles to locations corresponding to these new numbers.
   c. Extend your animation so that at least one other attribute is changed at each time step. For example, you may consider changing the color or the size of the circles. Make sure that transitions remain smooth (e.g., if transitioning size, gradually increase or decrease the circles' radii).

7. [Simple Bar Chart] Consider the following list of numbers,

```
let generator = d3.randomUniform();
let x = d3.range(10).map(generator);
```

   a. Create a bar chart that has one bar per number, with the height of that bar determined by the value in the array. There is no need to add labels or axes.

   b. Turn the bar chart from (a) on its side. The horizontal length of each bar should now encode each of the original numbers.

8. [Bar Chart Transitions] This problem continues [Simple Bar Chart] above. We will create a bar chart that adds and removes one bar each time a button is clicked. Specifically, the function below takes an initial array `x` and creates a new array that removes the first element and adds a new one to the end. Using D3's generate update pattern, write a function that updates the visualization from [Simple bar chart] every time that `update_data()` is called. New bars should be entered from the left, exited from the right, and transitioned after each click. Your solution should look (roughly) like this example.

```
let bar_ages = [],
generator = d3.randomUniform(0, 500),
id = 0;

function update() {
  bar_ages = bar_ages.map(d => { return {id: d.id, age: d.age + 1, height: d.height }})
  bar_ages.push({age: 0, height: generator(), id: id});
  bar_ages = bar_ages.filter(d => d.age < 5)
  id += 1;
}
```

9. [Marching Circles] This problem gives practice using D3's general update pattern. The code at this link updates an array of objects each time a button is clicked. You will build a visualization that draws circles whose positions are updated each time the underlying data changes.

   a. What does the `update` function do? Give a brief explanation of each line.
   b. Using the general update pattern, create a drawing of circles where the x-coordinate position encodes the age (in terms of button clicks) of each circle. Older circles should be on the right.
   c. Exit all circles whose age is larger than 5.
   d. Modify your implementation so that circles enter from the top left corner and exit to the bottom right.
   e. Modify your implementation so that (i) circles are only exited after their age exceeds 10 and (ii) the y-coordinate positions of the circles move along a sine curve.

10. [Colorful squares] The code below draws a square on an otherwise blank canvas.

    a. Modify the code so that, every time the square is clicked, it changes to a new, randomly chosen color from the `d3.schemeCategory10` color palette.

    b. Introduce at least one new transition or interactive element. Be creative! For example, you can let the user create new squares, have the squares fall off of the canvas, change in size, or automatically change color.

11. [A revised "general update pattern"] When D3 was introduced, it recommended creating, modifying, and deleting elements using a "general update pattern", and this is still found in most introductions to D3, e.g., 1, 2, or 3. However, recent versions of D3 support a new `selection.join()` pattern to streamline updates. This exercise provides practice with this new pattern, following the documentation available here.

    a. Consider the join pattern in the second block in the documentation,

    ```
    svg.selectAll("text")
      .data(randomLetters())
      .join("text")
        .attr("x", (d, i) => i * 16)
        .text(d => d);

    yield svg.node();
    await Promises.tick(2500);
    ```

    What would happen if instead of `.join("text")`, we had used `.append("text")`? What can you conclude about the behavior of `.join()`?

b. Consider the join pattern in the third block in the documentation,

```
svg.selectAll("text")
 .data(randomLetters(), d => d)
 .join(
   enter => enter.append("text")
           .attr("fill", "green")
           .text(d => d),
   update => update
           .attr("fill", "gray")
 )
```

Every time new random letters are generated using `randomLetters()`, there is a chance that some letters are removed, others are added, and some overlap between the two calls. In each of the three cases above, how does the letter appear in the visualization after the update? Does it depend on the original location of that letter within the array?

c. Rewrite one example either from the D3 for R users book or from our own course lecture notes so that, instead of using the earlier general update pattern (based on `.enter()`, `.exit()`, `.merge()`, or `.update()`) it uses only `.join()`.

12. [Interactive Penguins II] The code at this link creates a scatterplot of body mass against flipper length in the penguins dataset. Your goal in this exercise is to provide a version that responds to user queries.

    a. Skim through `penguins.csv`, which can be downloaded by clicking the small paperclip symbol on the right hand side of the interface. Based on the data available, discuss a potential user query that might be worth supporting. What makes the query interesting?
    b. Describe two types of interactivity available in D3 that could be used to support this query. For each, explain (i) the types of input the user provides to specify the query and (ii) the changes in the appearance of graphical marks that is initiated by the query.
    c. Implement one of the two interactivity possibilities that you described in the previous part. You may extend the current online implementation. In your submission, include a short video of the interactivity, the code you used, and a brief explanation of the logic behind your implementation.

13. [General Update Pattern Reflection] The general update pattern is one of the central, but also most complex, concepts relevant to D3 programming. This exercise asks you to reflect on your understanding of this concept.

    a. Explain one situation where the general update pattern might be used. Your application may be to an animation or interactive visualization, for example. Be as specific as possible.
    b. Sketch the pseudocode that implements a version of the general update pattern that could potentially be used in the example you outlined in part (a). Give an explanation of the main parts of your pseudocode.
    c. What is one aspect of D3's general update pattern that you think might be most confusing to your peers?

14. [Random Walk with Links] This problem gives practice with the general update pattern in an animation inspired by this example. The code at this link creates a data structure that stores the locations and velocities of all the circles. You will add SVG circle and line elements that reflect updates to this data structure.

    a. What does the `update()` function do? Be specific.
    b. Append the circles in their initial state. The result should be a static plot of circles at the `x` and `y` coordinate positions specified by `circles`.
    c. Animate the circles by calling the `update` function repeatedly. Their locations should reflect the changes in the underlying `x` and `y` property values.
    d. Create links between pairs of circles that are within a prespecified distance of one another. Make sure to enter, exit, and update the paths to reflect the motion of the circles. Hint: For paths with

only two endpoints, you may use the `line` SVG element.

e. What would happen if you did not exit the lines when the circles drifted too far apart? Try this in your visualization and explain what you see.

15. [Urbanization Linked Plots I] In this problem, we will create a static version of the linked bar chart + slope graph visualization from Nadieh Bremer's Urbanization in East Asia visualization. Starter HTML, CSS, and JS code is available at these links [1, 2, 3]. Note that the group elements for the slope graph and barchart have already been translated, so each individual component can assume that the x-axis starts at pixel 0.

    a. Describe the structure of the data that are available for the visualization. How will be able to create a set of bars for both 2000 and 2010?
    b. What types of scales can be used to create the bar chart? Note that the country-level population densities vary from 0 to 10000. Create these scales.
    c. Append the rectangles in the bar chart. The colors from the original visualization are #858483 and #da6761 for 2000 and 2010, respectively.
    d. Annotate the bar chart with country names.
    e. What types of scales should be used for the slope graph? Note that the city-level population densities vary from 0 to 35000. Create these scales.
    f. Draw the circles and lines needed for the slope graph. Provide labels for the cities whose 2010 population density is above 19600.

16. [Urbanization Linked Plots II] This problem continues [Urbanization Linked Plots I], adding interactivity to the scaffolding from the previous version. Starter HTML, CSS, and JS code is available at these links [1, 2, 3]. This provides a static visualization over which we will add our interactivity.

    a. Add an event listener to the rectangles in the bar chart. Print the name of the currently hovered country every time a new bar is hovered.
    b. Update the slope graph based on bar chart interaction. Specifically, show only cities within the currently hovered country. You do not need to return to displaying all cities when the mouse moves off of a country.
    c. Add an event listener to the lines and circles in the slope graph. Print the name of the currently hovered city every time a new slope is hovered.
    d. Update the appearance of the slope graph depending on user interactions. Specifically, if the user has hovered over either the line or the circle associated with a city, highlight that city and make the rest more transparent.

17. [Interactive Census] The code below creates a bar chart of US female population by age groups from the 1900 census. The x-axis is age group, and the y-axis is number of people. We will create an interactive version of this display that allows us to see how the age structure changes over time. [This exercise is adapted from a visualization by Tongshuang (Sherry) Wu at the University of Washington.]

    a. Create an interactive version of the original display that allows you to toggle between age groups in 1900 and 1910. New bars should be introduced for people born between 1900 and 1910. The remaining bars should be shifted to the right by 10 years, and their heights should be adjusted to reflect the new size of that age group.

    b. Generalize your solution in (a) to allow the user to click forward or backward across all the decades in the original dataset.

    c. Allow the user to switch between the genders available in the dataset. When a selection is made, apply a smooth transition to the heights of the bars.

18. [Icelandic Population Analysis] In this problem, we will analyze the design and implementation of this interactive visualization of Iceland's population.

    a. Explain how to read this visualization. What are two potential insights a reader could takeaway from this visualization?
    b. The implementation uses the following data join,

```
rect = rect
  .data(data.filter(d => d.year === year), d => `${d.sex}:${d.year - d.age}`)
```

What does this code do? What purpose does it serve within the larger visualization?

c. When the bars are entered at `Age = 0`, they seem to "pop up," rather than simply being appended to the end of the bar chart. How is this effect implemented?

d. Suppose that you had comparable population-by-age data for two countries. What queries would be interesting to support? How would you generalize the current visualization's design to support those queries?

19. [D3 free exploration] Pick one of the following datasets, or propose one of your own interest: London Olympics, Pokemon, Chicago Traffic Accidents, Australian Pharmaceuticals.

a. Based on your understanding of how the data were collected and what they represent, prepare four questions that you are interested in. Choose questions that reflect different aspects of the data and whose answers will require more nuanced analysis.

b. Design and implement a D3 visualization whose interactivity helps to answer the questions you identified in (a). Make sure to include the general update pattern in some part of your implementation.

c. Explain the way the general update pattern is used in your visualization. Describe why you do or do not use transitions within your implementation.

4 Spatial and Temporal Data

1. [Matching Autocorrelation Functions] The purpose of this problem is to build further intuition about auto-correlation. We'll simulate data with known structure and then see what happens to the associated autocorrelation functions.

a. The code below simulates a sinusoidal pattern over the course of 2020. Extend the code so that `date` and `y` are contained in a tsibble object.

```
library(lubridate)
date <- seq(from = as_date("2020-01-01"), to = as_date("2020-12-31"), by = 1)
x <- seq(0, 12 * 2 * pi, length.out = length(date))
y <- sin(x) + rnorm(length(x), 0, .4)
```

b. Using the tsibble object, calculate and visualize the induced autocorrelation function. Use a maximum lag of 50, and interpret the resulting plot.

c. Write a function to simulate a version of the tsibble above, but with a linear trend from 0 to `z`, where `z` is an argument to the function.

d. Using the function from (c), generate 5 datasets with linear trends of varying magnitudes. Plot the associated autocorrelation functions and comment on the relationship between the strength of the trend and the shape of the function.

2. [German Traffic] We will use compare and contrast two encodings to visualize traffic volumes across a set of cities. The data are read in below,

```
traffic <- read_csv("https://uwmadison.box.com/shared/static/x0mp3rhhic78vufsxtgrwencchmghbdf.csv")
```

Each row is a timepoint of traffic within a city in Germany.

a. Make a plot of traffic over time, within each of the cities.

b. Describe two takeaways from the resulting figure.

c. Propose an alternative encoding of the same data. What are the strengths of the second approach? What are the strengths of the original approach?

3. [Interactive German Traffic] This problem will revisit the previous problem from an interactive point of view. We will build a visualization that helps users explore daily traffic patterns across multiple German cities, using interactivity to help users navigate the collection. We will need additional features related to the day of the week for each timepoint, created by the `wday` function below,

```
library(lubridate)
traffic <- read_csv("https://uwmadison.box.com/shared/static/x0mp3rhhic78vufsxtgrwencchmghbdf.csv")
 mutate(day_of_week = wday(date))
```

   a. Design and implement a Shiny app that allows users to visualize traffic over time across selected subsets of cities. Make sure that it is possible to view data from more than one city at a time. It is not necessary to label the cities within the associated figure.
   b. Introduce new inputs to allow users to select a contiguous range of days of the week. For example, the user should have a way of zooming into the samples taken within the Monday - Wednesday range.
   c. Propose, but do not implement, at least one alternative strategy for supporting user queries from either part (a) or (b). What are the tradeoffs between the different approaches in terms of visual effectiveness and implementation complexity?

4. [Spotify Time Series I] In this problem, we will study music streaming on Spotify in 2017. We'll start by looking at some characteristics of the most streamed song, and then will practice how to extract features from across the collection of most streamed songs.

   a. Let's look at the most streamed song of 2017, which was "Shape of You." The dataset here contains the number of streams for this song across regions, for each day in which it was in the Spotify 100 most streamed songs for that region. Create a `tsibble` object from this dataset, keying by `region` and indexing by `date`.

   b. Filter to `region == "global"`, and make a `gg_season` plot by month. Comment on the what you see.

   c. Provide a scatterplot showing the relationship between the number of streams of this song in the US and in Canada. Do the same between the US and Japan. Briefly comment. **Hint**: Use `pivot_wider` to spread the time series for each region across columns of a reshaped dataset.

   d. The dataset here contains similar data, but for all songs that appeared in the Spotify 100 for at least 200 days in 2017. We have filtered to only the global totals. Read these data into a tsibble, keyed by `artist:track_name` and extract features of the `streams` time series using the `features` function in the feasts library. It is normal to see a few errors reported by this function, it just means that some of the statistics could not be calculated.

   e. Which tracks had the highest and lowest `trend_strength`'s? Visualize their streams over the course of the year.

   f. Discuss one other feature that would be interesting to extract from these time series. Provide a non-technical explanation about what songs with high values of this feature would potentially look like.

5. [Spotify Time Series II] The code below provides the number of Spotify streams for the 40 tracks with the highest stream count in the Spotify 100 dataset for 2017. This problem will ask you to explore a few different strategies that can be used to visualize this time series collection.

```
spotify_full <- read_csv("https://uwmadison.box.com/shared/static/xj4vupjbicw6c8tbhuynw0pll6yh1w0d.
top_songs <- spotify_full %>%
  group_by(track_name) %>%
  summarise(total = sum(streams)) %>%
  slice_max(total, n = 40) %>%
  pull(track_name)
```

```
spotify <- spotify_full %>%
  filter(region == "global", track_name %in% top_songs)
```

   a. Design and implement a line-based visualization of these data.

   b. Design and implement a horizon plot visualization of these data.

   c. Building from the static views from (a - b), propose, but do not implement, a visualization of this dataset that makes use of dynamic queries. What would be the structure of interaction, and how would the display update when the user provides a cue? Explain how you would use one type of D3 selection or mouse event to implement this hypothetical interactivity.

6. [Bike Demand] This problem asks you to visualize a dataset of hourly bikeshare demand. Provide your code and make sure it is readable.

   a. Make a line plot of bike demand (`count`) by hour, faceted out across the 7 days of the week (`weekday`).
   b. Create a new summary data.frame giving the 25 and 75 percent quantiles of demand (`count`) for each hour (`hr`) by day of the week (`weekday`) combination, separately within each year (`yr`) that the data was collected.
   c. Using a ribbon plot, overlay the quantities from (b) onto your plot from part (a). Use color to distinguish between the ribbons for the first and second year that the data were collected.
   d. Provide a brief description of some takeaways from the final visualization.

7. [Australian Pharmaceuticals I] The PBS dataset contains the number of orders filed every month for different classes of pharmaceutical drugs, as tracked by the Australian Pharmaceutical Benefits Scheme. It is read in below,

```
pbs <- read_csv("https://uwmadison.box.com/shared/static/fcy9q1uleqru7gcs287q903y0rcnw2a2.csv") %>%
  mutate(Month = as.Date(Month))
```

   a. In the text box below, provide code that will transform the data into a tsibble object, with month as the index and ATC2_desc as the key.
   b. Using the feasts package, extract features for this time series collection. The series with the largest trend_strength is...
   c. Perform a PCA on the following subset of extracted features (these are just trend_strength, features that start with "seasonal", or features containing the string "acf"), Make sure to normalize all features first.

```
"trend_strength", "seasonal_strength_week", "seasonal_peak_week", "seasonal_trough_week",
"stl_e_acf1", "stl_e_acf10", "acf1", "acf10", "diff1_acf1",
"diff1_acf10", "diff2_acf1", "diff2_acf10", "season_acf1",
"pacf5", "diff1_pacf5", "diff2_pacf5", "season_pacf"
```

   d. Make a plot of the top two principal components. Show the 17 features for each component, and order the features according to the magnitude of their contribution. Provide a brief interpretation of the second principal component direction.

8. [Australian Pharmaceuticals II] The code below takes the full PBS dataset from the previous problem and filters down to the 10 most commonly prescribed pharmaceutical types. This problem will ask you to implement and compare two approaches to visualizing this dataset.

```
pbs_full <- read_csv("https://uwmadison.box.com/shared/static/fcy9q1uleqru7gcs287q903y0rcnw2a2.csv"
    mutate(Month = as.Date(Month))

top_atcs <- pbs_full %>%
  group_by(ATC2_desc) %>%
  summarise(total = sum(Scripts)) %>%
  slice_max(total, n = 10) %>%
```

```
  pull(ATC2_desc)

pbs <- pbs_full %>%
  filter(ATC2_desc %in% top_atcs, Month > "2007-01-01")
```

   a. Implement a stacked area visualization of these data.

   b. Implement an alluvial visualization of these data.

   c. Compare and contrast the strengths and weaknesses of these two visualization strategies. Which user queries are easier to answer using one approach vs. the other?

9. [Polio incidence] In this problem, we will use a heatmap to visualize a large collection of time series. The data, prepared by the Tycho Project, contain weekly counts of new polio cases in the United States, starting as early as 1912 (for some states).

   a. Pivot the raw dataset so that states appear along rows and weeks appear along columns. Fill weeks that don't have any cases with 0's.

```
polio <- read_csv("https://uwmadison.box.com/shared/static/nm7yku4y9q7ylvz5kbxya3ouj2njd0x6.csv")
```

   b. Use the `superheat` package to make a heatmap of the data from (a). Have the color of each tile represent `log(1 + cases)`, rather than the raw counts. Reorder the states using a hierarchical clustering by setting `pretty.order.rows = TRUE`.

   c. Supplement the view from part (b) with a barchart showing the US total incidence during every given week. Interpret the resulting visualization. *Hint: use the* ***yt*** *argument of superheat.*

   d. Describe types of annotation would improve the informativeness of the plot made in part (c). Also, describe one advantage and one disadvantage of visualizing a collection of time series as a heatmap, instead of as a collection of line plots.

10. [Chicago Traffic Accidents] The dataset here is a preprocessed version of a dataset downloaded from Chicago's official traffic accidents database. In this problem, we will use a heatmap to understand what times during the week and over the year are associated with the highest number of traffic accidents.

    a. Use a heatmap to visualize the accident counts over time. Specifically, one dimension of the heatmap should index the 168 hours in the week, and the other should index every week from 2013-03-03 to 2022-03-27.

    b. Enrich your visualization with information about the total number of accidents in each of the 168 hours in the week. *Hint: Use the* ***yt*** *or* ***yr*** *arguments to* ***superheat****.*

    c. Provide a brief interpretion of your final visualization.

11. [Air Pollution Time Searcher] The dataset at this link measures hourly PM2.5 concentration in Beijing for one year (2014). We will implement a simplified version of a TimeSearcher visualization to help us discover patterns in daily pollution trends. We will start by creating a line plot before implementing brush interactivity.

    a. We will visualize the year's worth of data as a collection fo 365 separate lines. How do the data need to be restructured in order support this visualization? Implement a function that reshapes the data. You may use either R or javascript.

    b. Build a static lineplot visualization of these data. Make sure to provide x and y-axes.

    c. Introduce a brush so that all series that pass through the brush are highlighted.

    d. Following the discussion on this page, modify your solution to part (c) so that multiple brushes can be drawn simultaneously. Filter down to series that pass simultaneously through all brushes.

e. Comment on the shapes of the series that you can discover with your multi-brush visualization. What are the most common trends? Did you discover any outlying shapes?

f. Bonus: Link your time searcher visualization with a table. Consider using an existing javascript tables library, like one of those discussed on this page.

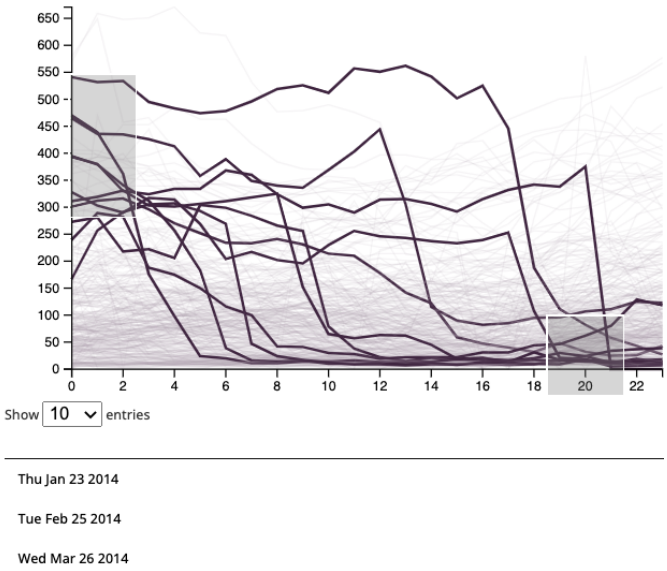


Figure 7: An example solution to [Air Pollution Time Searcher].

12. [Women, Business and the Law] The World Bank's Women, Business, and the Law program has curated a dataset about gender equality across countries since 1971. Since there are more than 30 variables summarizing each country for each year, we will use clustering to observe the general trends across all measures.

    a. The code below reads in the data, simplifies column names (saving the original names in the `question_mapping` data.frame), and converts text Yes / No responses to numerical 0 / 1. Widen the dataset so that each row corresponds to a single country × year combination.

```
# read in and create a codebook
survey <- read_csv("https://github.com/krisrs1128/stat479_s22/blob/main/_slides/week9/exercises/Viz
question_mapping <- survey %>%
  select(Question, `Question Category`) %>%
  distinct() %>%
  mutate(q_id = str_c("Q", row_number()))

recode_vector <- setNames(question_mapping$Question, question_mapping$q_id)
survey <- survey %>%
  mutate(
    Question = fct_recode(Question, !!!recode_vector),
    answer = as.numeric(ifelse(`Text Answer` == "Yes", 1, 0))
  ) %>%
  select(Country, `WBL Report Year`, Question, answer)

survey
```

```
## # A tibble: 332,500 x 4
##    Country     `WBL Report Year` Question answer
```

```
##    <chr>                  <dbl> <fct>     <dbl>
##  1 Afghanistan             1971 Q1            0
##  2 Afghanistan             1971 Q2            1
##  3 Afghanistan             1971 Q3            0
##  4 Afghanistan             1971 Q4            1
##  5 Afghanistan             1971 Q5            0
##  6 Afghanistan             1971 Q6            1
##  7 Afghanistan             1971 Q7            1
##  8 Afghanistan             1971 Q8            1
##  9 Afghanistan             1971 Q9            0
## 10 Afghanistan             1971 Q10           1
## # ... with 332,490 more rows
```

b. Apply K-means to the question responses (you may choose K). Visualize the centroids and briefly interpret each cluster in context, using the original text of the questions stored in `question_mapping`.

c. Visualize change in cluster sizes over time. Provide a brief interpretation (for example, in which types of questions is there the most / least progress towards equality?).

13. [NYC Trees] In this problem, we'll use vector data to enrich a visualization of trees in New York City. In the process, we'll practice reading in and generating summaries of geospatial data.

a. The data at this link include a subset of data from the New York City Tree Census. Make a scatterplot of the locations of all trees in the data, coloring in by tree species group and faceting by health.

b. Suppose we wanted to relate these data to characteristics of the built environment. We have curated public data on roads and buildings within the same neighborhood. Read these data into `sf` objects using `read_sf`. For both datasets, report (i) the associated CRS and (ii) the geometry type (i.e., one of point, linestring, polygon, multipoint, multilinestring, multipolygon, geometry collection).

c. Generate a version of the plot in (a) that has the roads and buildings in the background.

14. [Himalayan Glaciers] In this problem, we'll apply the reading's discussion of raster data to understand a dataset containing Landsat 7 satellite imagery of a Himalayan glacier system.

a. Read the data into a `brick` object. What is the spatial extent of the file (that is, within what geographic coordinates do we have data)? How many layers of sensor measurements are available?

b. Generate an RGB image of this area. In Landsat 7, the first three layers (B1, B2, and B3) provide the red, green, and blue channels.

c. Make a plot of the slopes associated with each pixel within this region.

15. [CalFresh Enrollment I] In this problem, we will investigate spatial and temporal aspects of enrollment in CalFresh, a nutritional assistance program in California.

a. The code below reads in the CalFresh data. We've filtered out February 2019, since benefits were distributed differently in this month, leading to outliers for most counties. Extract features of the `calfresh` time series using the `features` function in the feasts library.

```
library(tsibble)
calfresh <- read_csv("https://uwmadison.box.com/shared/static/rduej9hsc4w3mdethxnx9ccv752f22yr.csv"
  filter(date != "2019 Feb") %>%
  mutate(date = yearmonth(date)) %>%
  as_tsibble(key = county, index = date)
```

b. Visualize CalFresh enrollment over time for the counties with the highest and lowest `seasonal_strength_year`.

c. The code below reads in a vector dataset demarcating the county boundaries in California. Join in the features dataset from (a) with this these vector data. Use this to produce a map with each county shaded in by its `seasonal_strength_year`.

```
library(sf)
counties <- read_sf("https://uwmadison.box.com/shared/static/gropucqxgqm82yhq13do1ws9k16dnxq7.geojs
```

d. Propose, but do not implement, a visualization of this dataset that makes use of dynamic queries. What questions would the visualization answer? What would be the structure of interaction, and how would the display update when the user provides a cue?

16. [CalFresh Enrollment II] In this problem, we will develop an interactively linked spatial and temporal visualization of enrollment data from CalFresh, a nutritional assistance program in California. We will use D3 in this problem.

    a. Using a line generator, create a static line plot that visualizes change in enrollment for every county in California.
    b. On the same page as part (a), create a choropleth map of California, shaded in by the average enrollment for that county over the full time window of the dataset.
    c. Propose one interactive, graphical query the links the combined spatial + temporal view from (a - b). For example, you may consider highlighting time series when a county is hovered, highlighting counties when specific series are brushed, or updating choropleth colors when a subsetted time range is selected.
    d. Implement your proposal from part (c).

17. [Geospatial Datasets] For each of the datasets below, specify whether it is in a vector or raster data format. If it is in a vector data format, explain which types of geometries it contains (e.g., a point or linestring). Explain your reasoning.

    a. NYC Building Footprints
    b. Africa Population 2020
    c. Himalayan Glacial Lakes
    d. US EV Charging
    e. Zion Elevation

18. [Geospatial Commands] For each of the commands below, explain (i) whether it can be applied to vector data, raster data, or both and (ii) discuss a typical use case for the function.

    a. `read_sf`
    b. `raster`
    c. `as.data.frame(img, xy = TRUE)`
    d. `geom_sf`
    e. `coord_sf`

19. [Temporal and Geospatial Commands] For each of the commands below, explain what role it serves in temporal or spatial data visualization. Describe a specific (if hypothetical) situation within which you could imagine using the function.

    a. `geom_stream`
    b. `tm_lines`
    c. `rast`
    d. `geom_horizon`

20. [`tmap` Exploration] There are many example maps on the `tmap` package's homepage – see the vignettes, tutorials, or presentations, for example. In this problem, you will recreate and deconstruct one example visualization from this webpage.

    a. Pick one figure from the page linked above. Describe the data used to generate this figure. Is the data in a vector or raster format?
    b. Regenerate the figure and write comments in the code that describe the purpose for each layer (or related groups of layers).
    c. Have you ever worked with a dataset where the type of visualization you just created might have been helpful? If so, describe how. If not, imagine a hypothetical data analysis project where you might use this code.

21. [Glacial Lakes]. The data at this link contain labels of glacial lakes the Hindu Kush Himalaya, created during an ecological survey in 2015 by the International Centre for Integrated Mountain Development.

    a. How many lakes are in this dataset? What are the latitude / longitude coordinates of the largest lakes in each Sub-basin?

    b. Plot the polygons associated with each of the lakes identified in step (a). _Hint: You may find it useful to split lakes across panels using the `tmap_facets` function. If you use this approach, make sure to include a scale with `tm_scale_bar()`, so that it is clear that lakes have different sizes.

    c. Visualize all lakes with latitude between 28.2 and 28.4 and with longitude between 85.8 and 86. Optionally, add a basemap associated with each lake.

22. [Population Density] The problem below visualizes population data from the `afrilearndata` package.

    a. What are the dimensions (in pixels) of the population dataset? What is its coordinate reference system?

    b. Convert the raster data to a `data.frame`. Omit rows without any population data.

    c. Visualize the data. What are some regions with the lowest and highest population densities?

23. [Masking Temperature Data] This problem will visualize the temperature data downloaded from this site from the Australian Bureau of Meteorology. It presents some new ideas related to raster masking.

    a. Read in the raster data, directly convert it to a data.frame, and use a `geom_raster` layer to visualize raw temperature data.

    b. The data includes temperature over the ocean! We really only want to visualize the temperatures over land. Using the `mask` function from the `raster` package, clip the original raster data to a polygon containing the Australia country boundaries. These boundaries are read in the code block below,

```
library(rnaturalearth)
australia <- ne_countries(country = "australia")
# mask(... fill this in...)
```

    c. Regenerate the plot from (a), but using the masked raster from (b). Ensure that the masked NA values are left out of the visualization. Bonus: Use a binned color scale (e.g., `scale_fill_viridis_b`) to make the temperature variations even clearer.

24. [Interactive NYC Trees] This problem walks through the creation of an interactive map of the NYC Trees Dataset.

    a. Filter down to only those trees whose diameter at breast height (`tree_dbh`) is larger than 10. This will help reduce overplotting.

    b. Use `leaflet` and the `addCircles` command to create a map with the trees kept in part (a). Have the size of the circles reflect the `tree_dbh` value.

c. Add a basemap that highlights features of the city without being too busy. For example, you can use `addProviderTiles(providers$CartoDB.Positron)`.

d. Color each tree's circle according to its species group. *Hint: Create a mapping from species groups to colors using the `colorFactor` function in Leaflet. For example, `pal <- colorFactor(brewer, filtered_trees$species_group)` creates an objet `pal` that can be called in the `color` argument to `addCircles`.*

25. [Job Interview] Imagine that you are interviewing job candidates for a data science position that involves visualizing {This Part Filled in Class}. Prepare an interview question that test candidates' knowledge of / experience with these topics. Also prepare examples of strong responses. An good question will either (i) ask the interviewee to describe a specific time in the past where they used the concept or (ii) present the interviewee with a type of challenge related to what is commonly encountered and see how they would address it.

26. [Visualization for Everyday Decisions] We routinely encounter spatial and temporal data in everyday life, from the from the dates on a concert calendar to the layout of buttons in an elevator. This problem asks you to critically reflect on the way these data are represented.

    a. Describe one type of spatial or temporal dataset (loosely defined) that you encounter in a nonacademic context. What visual encodings are used? Does it implement any interactivity?

    b. What questions does the visualization help you answer? How easily can you arrive at an accurate answer?

    c. In the spirit of the re-imagined parking sign project, propose an alternative representation of these data (or an alternative way of interacting with the same representation). Why did you propose the design that you did? What advantages do you think it has?

5 Network and Hierarchical Data

1. [Political Book Recommendations] In this problem, we'll study a network dataset of Amazon bestselling US Politics books. Books are linked by an edge if they appeared together in the recommendations ("customers who bought this book also bought these other books").

    a. The code below reads in the edges and nodes associated with the network. The edges dataset only contains IDs of co-recommended books, while the nodes data includes attributes associated with each book. Build a `tbl_graph` object to store the graph.

```
edges <- read_csv("https://uwmadison.box.com/shared/static/54i59bfc5jhymnn3hsw8fyolujesalut.csv", c
nodes <- read_csv("https://uwmadison.box.com/shared/static/u2x392i79jycubo5rhzryxjsvd1jjrdy.csv", c
```

    b. Use the result from part (a) to visualize the network as a node-link diagram. Include the book's title in the node label, and shade in the node according to political ideology.

    c. Create the analogous adjacency matrix visualization. Provide examples of visual queries that are easy to answer using one encoding but not the other (i.e., what is easy to see in the node-link view vs. what is easy to see in the adjacency matrix).

2. [Network Vis T/F] Select all the true statements about network visualization.

    - For large, dense networks, node-link diagrams are preferable to adjacency matrix visualizations.
    - For tasks related to local topology, node-link diagrams are preferable to adjacency matrix visualizations.
    - It is possible to visually encode edge weight in either node-link or adjacency matrix visualizations.
    - It is possible to visually encode node category in either node-link or adjacency matrix visualizations.

3. [Movie Genres] How do movie genres relate to one another? Is romance + comedy a more common combination than fantasy + adventure? We will use the dataset here to answer these questions, using a node-link diagram inspired by (but much simpler than) the film flowers project by Shirley Wu.

a. Build a `tbl_graph` containing the movies and their links.

b. Create a node-link diagram to represent the connections between movies and their genres.

c. A list of all the genres in this network is available here. Design a strategy to distinguish between genres and movies, and justify your choice of visual encodings. *Hint: Mutate the nodes in the* `tbl_graph`.

4. [COVID-19 Phylogeny] We will visualize a phylogenetic tree of SARS-CoV-2 genetic sequences. Each sequence has been annotated with the date and location it was found, and we will try to see how genetic variation covaries with these measurements.

   a. Build a `tbl_graph` using the links and nodes associated with the phylogenetic tree. Note that many nodes do not have variables associated with them. This is because annotations are only available for the leaves of the tree, and nodes are unobserved common ancestor viruses.

   b. Visualize the phylogenetic tree using the data from (a).

   c. Visually encode the date or location at which each sequence was found (for a challenge, try encoding both). Justify your choice of encodings – what are some advantages / downsides of the approach you decided on?

   d. Discuss the resulting visualization. For example, how much genetic diversity do different countries have? Where are different waves located genetically?

5. [Climate Crisis Youtube recommendations] We will study a dataset giving links between Youtube videos related to the climate crisis. The data were gathered by simulating a user browsing through recommendations, after having initially searched using a set of climate-related seed terms. Each node is a video, which includes features like the number of views and the channel it belongs to. Each edge provides an algorithmically generated recommendation.

   a. Build a `tbl_graph` from the provided node and edge data.

   b. Visualize the connections between videos as a node-link diagram. Color nodes in by the simulated browser session.

   c. Visualize the connections between videos as an adjacency matrix.

   d. Compare and contrast the visualizations in parts (b) and (c). What are some of the trade-offs associated with using one view vs. the other?

   e. Select a small number of channels of interest (e.g., "The Daily Show with Trevor Noah") For either parts (b) or (c), find a way to visually distinguish nodes belonging to these channels from the rest.

6. [UK Energy Flow] We will visualize how energy is produced and consumed in the United States, using a dataset from the UK's Department of Energy and Climate Change. Nodes and edges are used to classify types of activity that produce or consume energy. For example, the edge

   ```
   Wind    Electricity grid    289
   ```

   means that 289 KwH produced by wind power were sent to the country's electricity grid.

   a. Build a `tbl_graph` from the provided edges. Ensure that the resulting graph object has directed edges.

   b. Use `ggraph` to visualize the flow of energy across nodes in this network. Ensure nodes are annotated.

   c. Experiment with several layouts. Which do you find most useful? Is there an alternative layout you would prefer? Explain your reasoning.

7. [Stack Overflow Tag Network] Questions on Stack Overflow are given tags. This makes it easy to search through all the questions on a certain topic. The co-occurrence of tags appearing on certain questions also makes it natural to build a network between these tags – we draw a weighted edge

between co-occurring tag, with weights representing the number of questions within which the pair appears. We will look at only a small subset of tags, originally presented in this post. In the process, we will practice creating network visualizations and encoding detected cluster memberships.

a. Build a `tbl_graph` object based on node and link data that were generated using a web scraper.

b. Create a node-link visualization associated with the network.

c. Cluster nodes and encode modify your visualization from (b) to encode cluster membership. *Hint: Mutate the graph using the* ***`group_louvain()`*** *or* ***`group_infomap()`*** *functions from* ***`tidygraph`****.*

d. Comment on the visualization from (c). What do the clusters seem to correspond to?

8. [Coltrane Network] We will visualize the network of musicians who share a recording with John Coltrane, obtained from this source. Two musicians are linked with one another if they appeared on an album together, at least as recorded in the MusicBrainz database. To visualize these data, we will practice creating network visualizations and encoding detected cluster memberships.

a. Build a `tbl_graph` object based on the node and link data.

b. Create a node-link visualization associated with the network.

c. Cluster nodes and modify your visualization from (b) to encode cluster membership. *Hint: Mutate the graph using the* ***`group_louvain()`*** *or* ***`group_infomap()`*** *functions from* ***`tidygraph`****.*

d. Comment on the visualization from (c). How would you interpret the resulting clustering?

6 High-Dimensional and Text Data

1. [Colony Collapse] In this problem, we will study a dataset describing factors that might be leading to colony collapse disorder among bees. Since there are multiple stressors for each state × timepoint combination, we will use clustering to create a summarized "ecological stress" profile.

a. The code below reads in data from its original Tidy Tuesday source and replaces NAs with 0's. We will summarize each state × timepoint combination by its profile of `stress_pct` across each stressor. Reshape the data so that `stressor` appears along columns, with the associated percentage contained within the table. Each row is therefore a profile of the amount of different stressors at a particular location and time.

```
stressor <- read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/202
  mutate(stress_pct = replace_na(stress_pct, 0))
```

b. Apply K-means to the profiles constructed in part (a). You may choose K. Visualize and briefly interpret the resulting centroids.

c. Design and implement a visualization to show the change in cluster memberships over time. Briefly interpret your results in context.

d. [Bonus] The code below reads in spatial data associated with each state. Design and implement a visualization that encodes spatial variation across clusters, either at a single timepoint or over all time. An example result is shown in Figure 8.

```
library(rnaturalearth)
library(sf)
states <- ne_states("United States of America") %>%
  st_as_sf() %>%
  rmapshaper::ms_simplify() %>% # reduce size for easy plotting
  filter(!(name %in% c("Alaska", "Hawaii"))) %>%
  select(name, geometry) %>%
  rename(state = name)
```

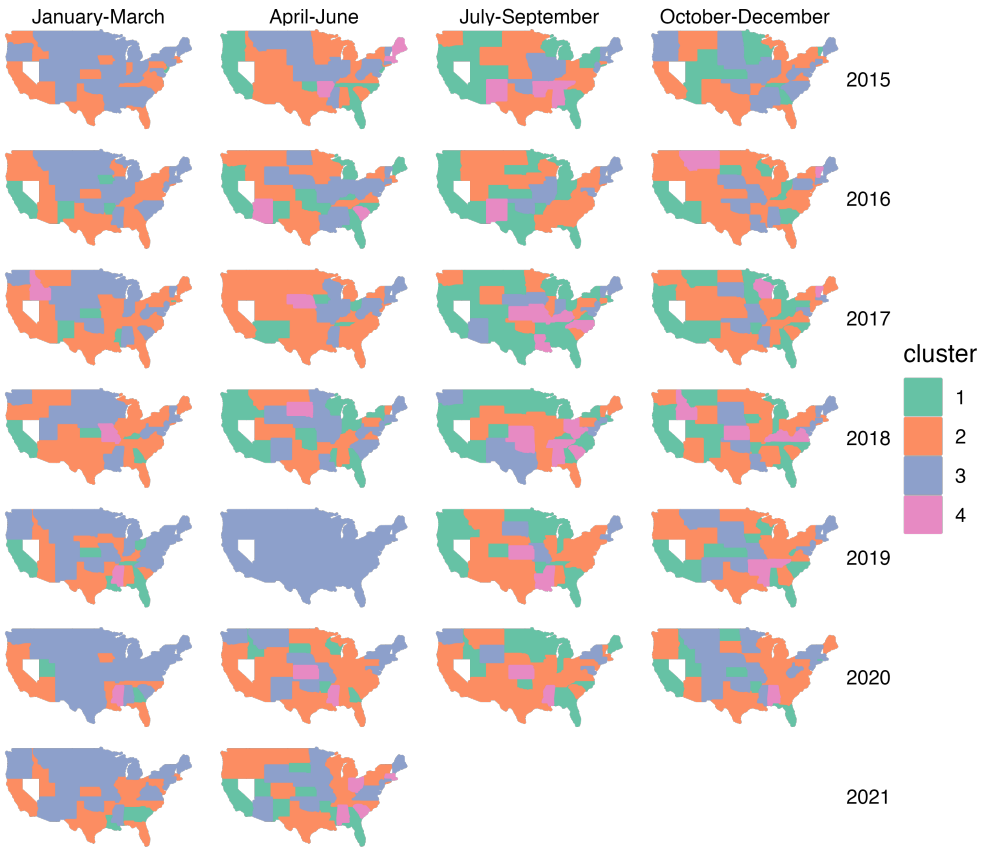


Figure 8: Spatial distribution of colony collapse clusters over time.

2. [Hierarchical Clustering T/F] Figure **??** shows a subset of a large hierarchical clustering tree. Select all the true statements below.

   - Samples C and D are more similar to one another than A and B.
   - If this subtree were cut to form K = 5 clusters, then C and D would belong to the same cluster.
   - A and B are more similar to each other than B and C.
   - If this subtree were cut to form K = 2 clusters, then A and B would belong to the same cluster.

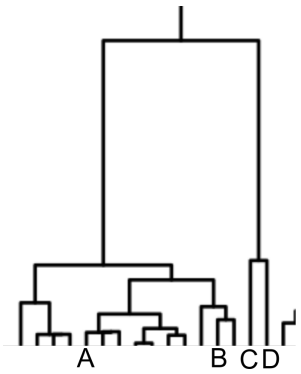


Figure 9: A subset of a large hierarchical clustering tree

3. [Taxi Trips] In this problem, we will use hierarchical clustering to find typical taxi trip trajectories in Porto, Portugal. The data are a subset from the the ECML / PKDD 2015 Challenge – the link provides a complete data dictionary. We have preprocessed it into two formats. The first (`wide`) includes each taxi trip on its own row, with latitude and longitude coordinates along the journey given as separate columns (`x_0, y_0` is the origin of the trip and `x_15, y_15` is the destination). The second (`long`) format spreads each point of the journey into a separate row.

   a. Filter the `long` form of the data down to trip `1389986517620000047` and plot its trajectory as a sequence of points.

```
trips <- read_csv("https://uwmadison.box.com/shared/static/098cjaetm8vy0mufq21mc8i9nue2rr2b.csv", c
trips_wide <- read_csv("https://uwmadison.box.com/shared/static/cv0lij4d9gn3s8m2k98t2ue34oz6sbj5.cs
```

   b. We could hierarchically cluster rows in either the `wide` or the `long` format datasets. How would the interpretation of the results differ between the two approaches?

   c. Compute a hierarchical clustering of the `wide` format data, using only the columns starting with `x` or `y` as features.

   d. Cut the hierarchical clustering tree so that 8 clusters are produced. Visualize the trajectories of the taxi trips either colored or faceted by their cluster.

4. [Support for Animal Research] This problem utilizes response data from a survey of UW–Madison students and faculty by Sandgren and Streiffer (2019). The survey assessed respondents' support for the use of non-human animals in scientific research. Questions were categorized into various groups. The heatmap in figure **??** shows each respondent's average response by question type. Higher-valued responses indicate more support for the use of non-human animals in scientific research.

   a. The definition of "strong support" is not given. However, one cluster seems to correspond to the strong support group. Which one, and why?
   b. True or False: There appears to be a weak relationship between support for non-human animal research across question types.
   c. Cluster 3 contains the fewest respondents. Discuss potential reasons why this may have been the case.

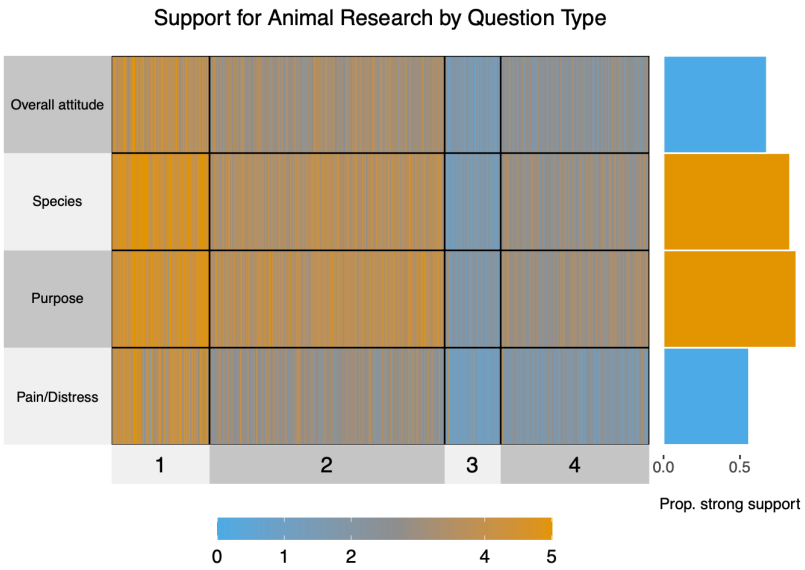


Figure 10: Responses from Sandgren and Streiffer (2022).

5. [Interpreting Clustering] Imagine that you are a statistical consultant working with a scientist / sports team / journalist / sales division head (pick your favorite or make up your own example). At one point in your study, you found it useful to apply various types of clustering methods. In this problem, you are asked to provide a non-technical explanation of how to interpret your clustering output, assuming that your audience is familiar with their data but not statistical methodology.

    a. You have run K-means with $K = 5$ clusters across their dataset, which includes 50 different measurements for each row. Explain to your audience what each centroid represents.

    b. At another point, you applied hierarchical clustering to the same dataset. You include a visualization of the hierarchical clustering tree in your report. Provide a brief, non-technical explanation of how to interpret the tree.

    c. Your audience finds the hierarchical clustering tree, but wonders why you needed a new algorithm to produce it. They think that you would have gotten similar results if you had simply run K-means with different values of K. Respond to their observations.

6. [Beijing Air Pollution] This problem studies daily air pollution and weather patterns in Beijing from 2010 to 2014. Each row in this dataset gives measurements for three variables, `pollution`, `temp`, and `wndspd` over time for a single day, indexed by `date`. For example the `temp_13` column gives the temperature at 1pm. Note that both `pollution` and `wndspd` have been $\log(1 + x)$ transformed.

    a. Cluster all the days using a hierarchical clustering. Be sure to scale the columns in advance, so that no single variable has undue influence on the result. Cut the tree so that we are left with 10 clusters, and provide a ribbon plot to visualize several quantiles associated with each cluster's typical pollution, temperature, and windspeed trajectories. visualization to understand the centroids of each cluster. Briefly interpret the main patterns you observe.

    b. Visualize the hierarchical clustering tree associated with your clustering from part (a). Bonus: For a single time of day, encode the average pollution level for all descendant nodes within your tree representation. To summarize a the pollution level of all descendants leaves in a tree, you may use the `map_local_dbl` function from `tidygraph`.

```
mean_fun <- function(neighborhood, ...) {
  cur_pollution <- neighborhood %>%
    filter(leaf) %>%
```

```
    pull(pollution_10) # average the 10am pollutions
  mean(cur_pollution, na.rm = TRUE)
}

G %>%
  mutate(avg_pollution = map_local_dbl(order = graph_order(), mode = "out", .f = mean_fun))
```

7. [Living wages] This problem will study a dataset from MIT's living wage calculator. Each row in this dataset gives living wages for state capitals within the continental US. Since living wages depend on a personal situation (e.g., number of children), this dataset is high-dimensional. The interpretation of each column is given here.

   a. Define a tidymodels `recipe` that normalizes all city features and specifies that PCA should be performed.

   b. Visualize the top 4 principal components. Based on the interpretation of each columns, what seems to distinguish capital cities with low vs. high values of PC2?

   c. Visualize the scores of each capital city with respect to the first two principal components. Make sure to annotate cities with their names. Pick a city on the graph and interpret its relative wage profile based on the principal components.

8. [Food nutrients] This problem will use PCA to provide a low-dimensional view of a 14-dimensional nutritional facts dataset. The data were originally curated by the USDA and are regularly used in visualization studies.

```
nutrients <- read_csv("https://uwmadison.box.com/shared/static/nmgouzobq5367aex45pnbzgkhm7sur63.csv
```

   a. Define a tidymodels `recipe` that normalizes all nutrient features and specifies that PCA should be performed.

   b. Visualize the top 6 principal components. What types of food do you expect to have low or high values for PC1 or PC2?

   c. Compute the average value of PC2 within each category of the `group` column. Give the names of the groups sorted by this average.

   d. Visualize the scores of each food item with respect to the first two principal components. Facet the visualization according to the `group` column, and sort the facets according to the results of part (c). How does the result compare with your guess from part (b)?

9. [Topics in *Pride and Prejudice*] This problem uses LDA to analyze the full text of *Pride and Prejudice*. The object `paragraph` is a data.frame whose rows are paragraphs from the book. We've filtered very short paragraphs; e.g., from dialogue. We're interested in how the topics appearing in the book vary from the start to the end of the book, for example.

```
library(tidytext)
paragraphs <- read_csv("https://uwmadison.box.com/shared/static/pz1lz301ufhbedzsj9iioee77r95xz4v.cs
paragraphs
```

```
## # A tibble: 1,092 x 2
##    text                                                              parag~1
##    <chr>                                                               <dbl>
##  1 "however little known the feelings or views of such a man may be on ~       1
##  2 "\"why, my dear, you must know, mrs. long says that netherfield is t~       2
##  3 "\"i see no occasion for that. you and the girls may go, or you may ~       3
##  4 "\"my dear, you flatter me. i certainly _have_ had my share of beaut~       4
##  5 "\"but consider your daughters. only think what an establishment it ~       5
##  6 "\"you are over-scrupulous, surely. i dare say mr. bingley will be v~       6
```

```
##  7 "\"i desire you will do no such thing. lizzy is not a bit better tha~       7
##  8 "mr. bennet was so odd a mixture of quick parts, sarcastic humour, r~       8
##  9 "mr. bennet was among the earliest of those who waited on mr. bingle~       9
## 10 "\"i honour your circumspection. a fortnight's acquaintance is certa~      10
## # ... with 1,082 more rows, and abbreviated variable name 1: paragraph
```

a. Create a Document-Term Matrix containing word counts from across the same paragraphs. That is, the i^{th} row of `dtm` should correspond to the i^{th} row of `paragraph`. Make sure to remove all stopwords.

b. Fit an LDA model to `dtm` using 6 topics. Set the seed by using the argument `control = list(seed = 479)` to remove any randomness in the result.

c. Visualize the top 30 words within each of the fitted topics. Specifically, create a faceted bar chart where the lengths of the bars correspond to word probabilities and the facets correspond to topics. Reorder the bars so that each topic's top words are displayed in order of decreasing probability.

d. Find the paragraph that is the purest representative of Topic 2. That is, if γ_{ik} denotes the weight of topic k in paragraph i, then print out paragraph i^* where $i^* = \arg\max_i \gamma_{i2}$. Verify that the at least a few of the words with high probability for this topic appear. Only copy the first sentence into your solution.

10. [Personality Types] This dataset contains a sample of posts from the Personality Cafe forum, together with labels giving the poster's (self-reported) Meyer-Briggs type (e.g., types starting with "I" are introverted, those with "E" are extroverted). It is time-consuming to go through all the posts manually, but to get a quick overview of the main topics that are discussed, we can use a Latent Dirichlet Allocation (LDA) model. This problem walks through the steps for (1) preparing a Document-Term matrix for LDA, (2) fitting the LDA model, and (3) interpreting the estimated topics and memberships.

    a. Transform the raw posts into a collection of per-post word counts. Remove stopwords from across all lexicons in `tidytext::stopwords`.

    b. Convert the `data.frame` from (a) into a topic models Document-Term matrix (i.e., an object of class `DocumentTermMatrix`). Fit an LDA model with 8 topics to the prepared object.

    c. Visualize the top 30 words within each of the fitted topics. Specifically, create a faceted bar chart where the lengths of the bars correspond to word probabilities and the facets correspond to topics. Reorder the bars so that each topic's top words are displayed in order of decreasing probability.

    d. Create a Structure plot displaying the topic memberships for each document. Sort documents according to their order on a hierarchical clustering tree, and facet documents according to personality type. Are there certain topics that appear to be more common in some personality types than others?

11. [Hotel Reviews] In this problem, we will practice using Latent Dirichlet Allocation to understand the topics that come up across hotel reviews from an online database. We will also study whether there are certain topics that are more common in positive vs. negative reviews.

    a. Transform the raw reviews into a collection of per-review word counts. Remove stopwords from across all lexicons in `tidytext::stopwords`.

    b. Convert the `data.frame` from (a) into a topic models Document-Term matrix (i.e., an object of class `DocumentTermMatrix`). Fit an LDA model with 8 topics to the prepared object.

    c. Create a heatmap of topic distributions for the 50 words d with the largest inter-quartile across topics, i.e., largest IQR for $(\beta_{dk})_{k=1}^{8}$). Provide an interpretation for one of the topics, based on the words that have high probability within it.

    d. Create a Structure plot displaying the topic memberships for each review. Sort reviews according to their order on a hierarchical clustering tree, and facet documents according to hotel rating. Are there certain topics that appear to be more common in negative vs. positive reviews?

12. [Single-Cell Genomics] In this problem, we will apply UMAP to a dataset of Peripheral Blood Mononuclear Cells (PBMC) released by 10X Genomics. The first column, `cell_tag`, gives an identifier for each cell in the dataset. All other columns are molecules that were detected in that cell. For example, CD74 is a molecule often found on the surface of T-cells.

    a. Define a tidymodels `recipe` that specifies that UMAP should be performed with the parameters `learn_rate = 0.1` and `neighbors = 5`. There is no need to normalize these data, as they have been normalized in advance using methods tailored to single-cell genomics data.

```
pbmc <- read_csv("https://uwmadison.box.com/shared/static/ai539s30rjsw5ke4vxbjrxjaiihq7edk.csv")
```

    b. Compute the UMAP embeddings across cells. Color points in by their value of the GNLY molecule.

13. [Phoneme Identification] The data read in below are a subset of about 4500 audio signals from the TIMIT speaker database. Each row is a processed version of an audio file describing how a single speaker said one of 5 sounds (e.g., "ao" is like the first vowel in "water" and "aa" is like the first vowel in "dark"). These 5 classes are stored in the column `g`. This dataset played a historically important role in sparking progress in the field of speech recognition.

```
phoneme <- read_csv("https://hastie.su.domains/ElemStatLearn/datasets/phoneme.data") %>%
  separate(speaker, c("split", "dialect_region", "speaker_id", "sentence_id"))
```

    a. Create a plot that shows each speaker's series (colnames `x.1` to `x.256`), faceted by phoneme. What features of these audio signals might you consider using if your goal were to distinguish between phonemes? *Hint: use `pivot_longer(x.1:x.256)` to reshape the data into tidy format.*

    b. Prepare a UMAP recipe that could be used to reduce the 256 sound dimensions to only two. Since all 256 audio features have the same interpretation, there is no need to normalize them. Extract the embeddings from the prepared recipe.

    c. Create a visualization of your UMAP embeddings from (b). Which phonemes seem the most similar to one another in the high-dimensional audio space?

14. [Political Blogs] In this problem, we will use topic modeling to see how the topics discussed across the political blogsphere evolved over course of 2008. We have fitted a model to a subsample of The CMU 2008 Political Blog Corpus, and the resulting topicmodels object is here.

    a. Use the discrepancy formula described in lecture,

$$D(k,l) := \beta_{kw} \log\left(\frac{\beta_{kw}}{\beta_{lw}}\right) + (\beta_{lw} - \beta_{kw})$$

    to identify 50 words that can be used to discriminate between the fitted topics.

    b. Make a heatmap that displays the β_{kw} probabilities of the words in part (b).

    c. Using the document metadata file, design and implement a visualization that shows variation in the memberships γ_{dk} either over time, across blogs, or in relation to political leaning. Suggest some interpretations for patterns that you see.

7 Model Visualization

1. [Silhouette Statistics Simulation] This problem uses simulation to build intuition about silhouette statistics. The function below simulates a mixture of three Gaussians, with means evenly spaced out between `(start, 0)` and `(end, 0)`. We will investigate what happens to the silhouette statistics for each point as the three clusters are made to gradually overlap.

```
library(tibble)
library(purrr)
library(ggplot2)
```

```r
mixture_data <- function(start = 0, end = 10, K = 3, n = 100) {
  mu <- cbind(seq(start, end, length.out = K), 0)
  map_dfr(1:K, ~ MASS::mvrnorm(n, mu[., ], diag(2)) %>% as_tibble())
}

ggplot(mixture_data()) +
  geom_point(aes(V1, V2)) +
  coord_fixed() +
  labs(x = "x", y = "y")
```

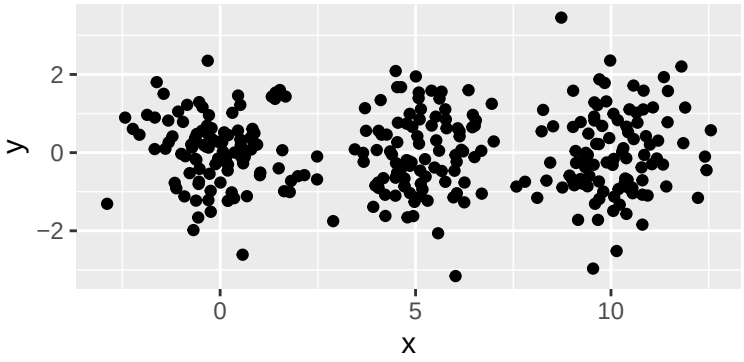


Figure 11: An example simulated dataset from problem 2.

a. Simulate versions of these data where the `end` argument is decreased from 10 to 0.

b. Write a function that performs K-means and computes the silhouette statistic given any one of the datasets generated in part (a). Use this function to compute the silhouette statistics for each point in the simulated datasets.

c. Visualize the silhouette statistics from part (b) overlaid onto the simulated data. Discuss the results.

2. [A Bikesharing Model] In this and the next problem, we will visualize models fitted to predict bikesharing demand in a subset of the Capitol Bikesharing dataset. We will see what types of features are learned by different types of models, whether there are any interactions between features, and whether linear and nonlinear approaches are substantively different.

   a. The code below fits a gradient boosting machine (GBM) to predict bikesharing demand (count) using all features available, except day of the year. Visualize the Ceteris Paribus profiles for the temperature and humidity variables, and provide a brief interpretation.

```r
library(caret)
bike <- read_csv("https://uwmadison.box.com/shared/static/aa91qdqehagag8wg8mqsm4z5b4g2hu0x.csv")
x <- dplyr::select(bike, -count, -dteday)
hyper <- data.frame(n.trees = 100, interaction.depth = 4, shrinkage = 0.1, n.minobsinnode = 10)
fit <- train(x = x, y = bike$count, method = "gbm", tuneGrid = hyper, verbose = FALSE)
```

   b. Has the GBM learned an interaction effect between the hour of day (`hr`) and weekend (`weekend`) features? Briefly justify your answer using a grouped CP plot.

3. [Contrastive Profiles for Bikesharing] This problem continues the exploration in the previous one. Here, we study whether the choice of GBM was critical, or whether any other model would have learned the same relationship between the predictors and bikesharing demand.

   a. The code below fits a lasso model to the same input and output dataset. Provide contrastive partial dependence profiles between the lasso and the GBM from above, focusing on the hour (`hr`),

humidity (`hum`), and temperature (`temp`) features. Comment on the result.

```
hyper <- data.frame(lambda = 0.001, alpha = 0)
fit_lm <- train(x = x, y = bike$count, method = "glmnet", tuneGrid = hyper)
```

b. Repeat part (a), but comparing the GBM with the CART model fitted below. Comment on the result.

```
library(caret)
hyper <- data.frame(cp = 0.01)
fit_cart <- train(x = x, y = bike$count, method = "rpart", tuneGrid = hyper)
```

4. [Gender Pay Gap] This problem uses CP profiles to investigate the gender gap in a Glassdoor dataset of employee salaries. It is helpful to use a model, because it allows us to control for multiple other factors – a direct plot of salary vs. gender could be criticized as not accounting for confounding variables. The code below trains a gradient boosting machine model on `BasePay` variable (yearly salary in USD), using all potential predictors in the dataset.

```
salary <- read_csv("https://github.com/krisrs1128/stat479_s22/raw/main/_slides/week12/exercises/Gla
library(caret)

x <- salary %>%
  select(Gender:Seniority) %>%
  mutate(across(where(is.character), as.factor)) # gbm needs chr -> factor
y <- salary$BasePay
fit <- train(x, y, method = "gbm", verbose = FALSE)
```

a. Before attempting to explain the model, it is helpful to consider its accuracy. Make a plot of the truth (`y`) against model predictions (`y_hat <- predict(fit)`) and comment on model performance.

b. Compute aggregate CP profiles grouped by gender and comment on the extent of the gender pay gap. According to the fitted prediction surface, is there more or less of a pay gap at certain ages or levels of seniority?

c. Show the analogous display without aggregating (i.e., `geom = "profiles"`). What is the interpretation of each line in this plot?

5. [Overfitting `sin(x)`] We can use contrastive PD plots to visually evaluate model over / underfitting. To illustrate, we will fit two K-nearest neighbors models on a simple dataset, simulated below.

```
N <- 100
df <- data.frame(x = runif(N, -pi, pi)) %>%
  mutate(y = sin(x) + rnorm(N, 0, 0.2))

ggplot(df) +
  geom_point(aes(x, y))
```

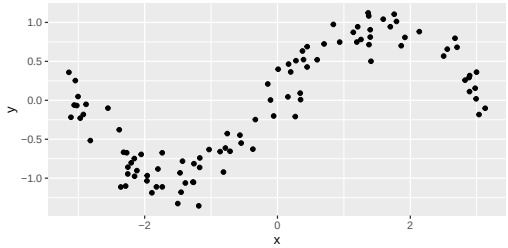


Figure 12: A simulated sine wave dataset, used to evaluate overfitting.

Below, we train three models to fit this curve, averaging over the 2, 10, and 50 nearest neighbors before making a prediction at a new location.

```
library(caret)
library(tidyverse)

hyper <- map(c(2, 10, 50), ~ data.frame(k = .))
fits <- map(
  hyper,
  ~ train(x = df %>% select(x), y = df$y, method = "knn", tuneGrid = .)
)
```

a. Create an `explainer` object for each of the fits above. Use these to create contrastive PD profiles between the three models. *Hint: It will be important to label each of the derived profile objects.*

b. Interpret the result from (a). Which models seem over / underfit? Which would you use in practice, and how did you draw this conclusion?

c. Repeat the same analysis with a random forest model (`method = "rf"` in caret). Instead of varying the `k` parameter, try several values of `mtry`. Is this class of models more or less sensitive to overfitting than K-nearest neighbors?

6. [Tensorflow Playground] This website provides an interactive visualization of a deep learning model trained on several toy datasets. This problem invites you to explore the visualization and reflect on its interpretation and design.

   a. Experiment with several datasets, input features, layer sizes, or network depths. For which setups does the model seem to converge to a solution most easily? When does the model struggle to converge? Describe how you drew your conclusions.

   b. Comment on the design of this visualization and the visualization concepts that it uses. What do you think are some of its more effective design decisions?

   c. Pose a question motivated by your experimentation with the interface. For example, this can be something you found confusing about the interface, a question it raised about neural networks more generally, or a thought about visual design.

7. [Histopathology Embeddings] This problem investigates the features learned by a residual network model trained trained to classify histopathology slides. Specifically, the script at this link was used to train a model to images from the Pcam benchmark dataset. Each image is a tissue slide. The class labels describe whether the center 32×32 patch within the image contains any cancerous cells.

   In the process, we will also practice using the `reticulate` package to read in numpy arrays produced by the python training script linked above. This language interoperability makes it possible to use the packages best suited to both modeling (`pytorch`) and visualization (`ggplot2`).

   a. We have hosted a subsample of the training images at this link. Their corresponding labels are stored as numpy arrays. Visualize the raw images corresponding to 10 images from each class. *Hint: To unzip these files from the command line, you can use* `tar -zxvf subset.tar.gz`

   b. For the subsample in part (a), we have saved the residual network features from the final pre-classification layer. They are available at this link. Generate UMAP embeddings for the images based on these features, and shade in each sample according to its class.

   c. Using `annotation_raster` layers from `ggplot2`, display the original images from (a) at the locations of the UMAP coordinates from (b). The correspondence between image filenames and features is given by this array. In particular, the i^{th} element of this array is the source image for the i^{th} row of the features matrix.

8. [Model Visualization Use Cases] The exercise below describes a few hypothetical situations where model visualization can be used. Discuss how you might approach them.

a. Your colleagues have trained a random forest model to predict whether activity observed on a corporate network might be the early signs of a cyberattack. The model is based on features like the average network data rate, the average packet size, and the port numbers of source traffic. Describe visualizations that could help your colleagues understand and improve their trained model.

b. You working with a client to analyze data from a crisis text helpline. They are interested in the topics that often arise during these support sessions. They are also curious whether some topics are are more easily resolved than others, as measured by a follow-up survey. Propose an analysis strategy and the types of visualizations you expect to be helpful.

c. You have built a deep learning model to detect changes in satellite imagery (e.g., before and after a natural disaster). Describe visualization techniques that can help you understand the latent image characteristics that may be learned by your model. # Uncertainty Visualization

9. [Household Radon Levels] This problem analyzes a dataset of household radon levels in Minnesota. The data were gathered by the EPA in an effort to understand which factors lead to increased exposure to this carcinogenic gas; e.g., it is known that radon levels are higher in basements than on the ground floor. A secondary goal of this problem is to illustrate the use of the `rstanarm` and `tidybayes` packages to extract prior and posterior predictive distributions. While these packages cannot be applied as generally as the methods discussed in lecture, they provide an elegant interface for specific types of Bayesian models. You will not need to write any code using these packages – we only ask that you visualize their outputs.

   a. Design and implement a visualization that describes the variation in `log_radon` levels from county to county.

```
radon <- read_csv("https://uwmadison.box.com/shared/static/3yn994tc1ft73z3br4ys18uri700gvik.csv")
```

   b. The code below simulates prior predictive data from two potential Bayesian models of `log_radon`. Both models have the form `log_radon ~ (1 | county) + floor`, which means each county - floor level combination gets its own mean. In the first model, a vague $\mathcal{N}(0, 1000)$ prior is used, while in the second, a somewhat more informative $\mathcal{N}(0, 10)$ is used instead.

```
library(rstanarm)
priors <- list(
  "vague" = stan_lmer(log_radon ~ (1 | county) + floor, prior = normal(0, 1000), data = radon, pri
  "informative" = stan_lmer(log_radon ~ (1 | county) + floor, prior = normal(0, 10), data = radon,
)
```

The code below extracts 5 simulated datasets from the prior predictive. Each row is a simulated house; the simulation number is denoted by `.draw`. The index to the original dataset is given by `.row`. The simulated `log_radon` levels are given in the `.prediction` column. The `prior` column describes which type of prior was used for that simulation.

```
library(tidybayes)
prior_preds <- map_dfr(priors, ~ add_predicted_draws(radon, .), .id = "prior") %>%
  filter(.draw > 3995)

head(prior_preds)
```

```
## # A tibble: 6 x 10
## # Groups:   floor, county, log_radon, log_uranium, .row [2]
##   prior floor county log_radon log_uranium  .row .chain .iteration .draw .pred~1
##   <chr> <dbl> <chr>      <dbl>       <dbl> <int>  <int>      <int> <int>   <dbl>
## 1 vague     1 Aitkin     0.833      -0.689     1     NA         NA  3996 -1360.
## 2 vague     1 Aitkin     0.833      -0.689     1     NA         NA  3997    85.8
## 3 vague     1 Aitkin     0.833      -0.689     1     NA         NA  3998  -826.
```

```
## 4 vague     1 Aitkin     0.833      -0.689     1     NA         NA  3999   906.
## 5 vague     1 Aitkin     0.833      -0.689     1     NA         NA  4000   391.
## 6 vague     0 Aitkin     0.833      -0.689     2     NA         NA  3996   280.
## # ... with abbreviated variable name 1: .prediction
```

Design and implement a visualization to compare the two types of priors. Which prior seems more plausible?

c. The code below draws posterior predictive samples from two new models, which either ignore or model variation from county to county (`lm` and `hierarchical` respectively). Both use information about whether the measurement came from a basement. Only the last 10 simulated runs are kept. Using these data, make boxplots of posterior predictive samples across counties, for each of the two models. What are the main differences between the two models? In what ways are the county level effects (as you studied in part (a)) accurately or inaccurately modeled?

```
library(tidybayes)
posterior_preds <- list(
  "lm" = stan_glm(log_radon ~ floor, prior = normal(0, 10), data = radon, chains = 1, refresh = 0),
  "hierarchical" = stan_lmer(log_radon ~ (1 | county) + floor, prior = normal(0, 10), data = radon,
) %>%
  map_dfr(~ add_predicted_draws(radon, .), .id = "model") %>%
  filter(.draw > 990)
```