Shiny for All: Making Statistics and Data Science Accessible

Neil J. Hatfield

Land Acknowledgments

The Pennsylvania State University campuses are located on the original homelands of the Erie, Haudenosaunee (Seneca, Cayuga, Onondaga, Oneida, Mohawk, and Tuscarora), Lenape (Delaware Nation, Delaware Tribe, Stockbridge-Munsee), Monongahela, Shawnee (Absentee, Eastern, and Oklahoma), Susquehannock, and Wahzhazhe (Osage) Nations. As a land grant institution, we acknowledge and honor the traditional caretakers of these lands and strive to understand and model their responsible stewardship. We also acknowledge the longer history of these lands and our place in that history.

Metro Toronto Convention Centre events take place on land traditionally inhabited and cared for by First Nations. Toronto is the traditional territory of many First Nations, including the Mississaugas of the Credit, the Anishnabeg, the Chippewa, the Haudenosaunee and the Wendat peoples.

Outline

1. How I came to Shiny Apps
2. Shiny Overview
3. Web Accessibility
    A. Web Standards
    B. Tools
4. Examine a Few Key Areas
5. An Invitation

https://github.com/neilhatfield/JSM2023_Shiny_for_All

How I came to Shiny Apps

Book of Apps for Statistics Teaching (BOAST)

https://shinyapps.science.psu.edu

Freely available collection of Shiny apps.

Currently, the book is divided into three sections.

1. Introductory Apps
    1. Data Gathering
    2. Data Description
    3. Basic Probability
    4. Statistical Inference
2. Upper Division Apps
    1. Probability/ Math-Stat
    2. Regression
    3. ANOVA
    4. Time Series
    5. Sampling
    6. Categorical Data
    7. Data Science
    8. Stochastic Processes
3. Other Disciplines
    1. Biology
4. Coming Soon!
    1. Nonparametrics
    2. Bayesian Analysis
    3. Game Theory
    4. Neural Nets
    5. Bootstrapping
    6. *k*-Means Clustering

Shiny App Project

An important aspect of BOAST is that almost all of the apps are created *by* students *for* students.

The Shiny App project is a summer/fall experience for undergraduate students majoring in Statistics and/or Data Science.

The goals for the project include

- Supporting the participants in building their R coding experience,
- Supporting the participants in constructing more productive understandings of various statistical topics,
- Provide the participants with experience working on a coding team that uses GitHub and follows a centralized Style Guide,
- Update (and improve) existing apps in BOAST, and
- Create new apps for BOAST.

Since 2017, 61 students have completed the project. This year we have five awesome students.

Shiny App Project

The Shiny App project is a summer/fall experience for undergraduate students majoring in Statistics and/or Data Science.

Summer Component

- A week of training that introduces the students to {shiny} and other related packages, the Style Guide, etc.
- They then adopt an existing app (or several) to work on improving
- They also begin to plan a brand new app.
- Typically, this is a paid experience.

Fall Component

- Students may opt to take a 1 credit seminar in the Fall semester.
- Continue work from the summer
- We require them to present their work in some forum
- Plan and carryout some research

Shiny Overview

What is Shiny?

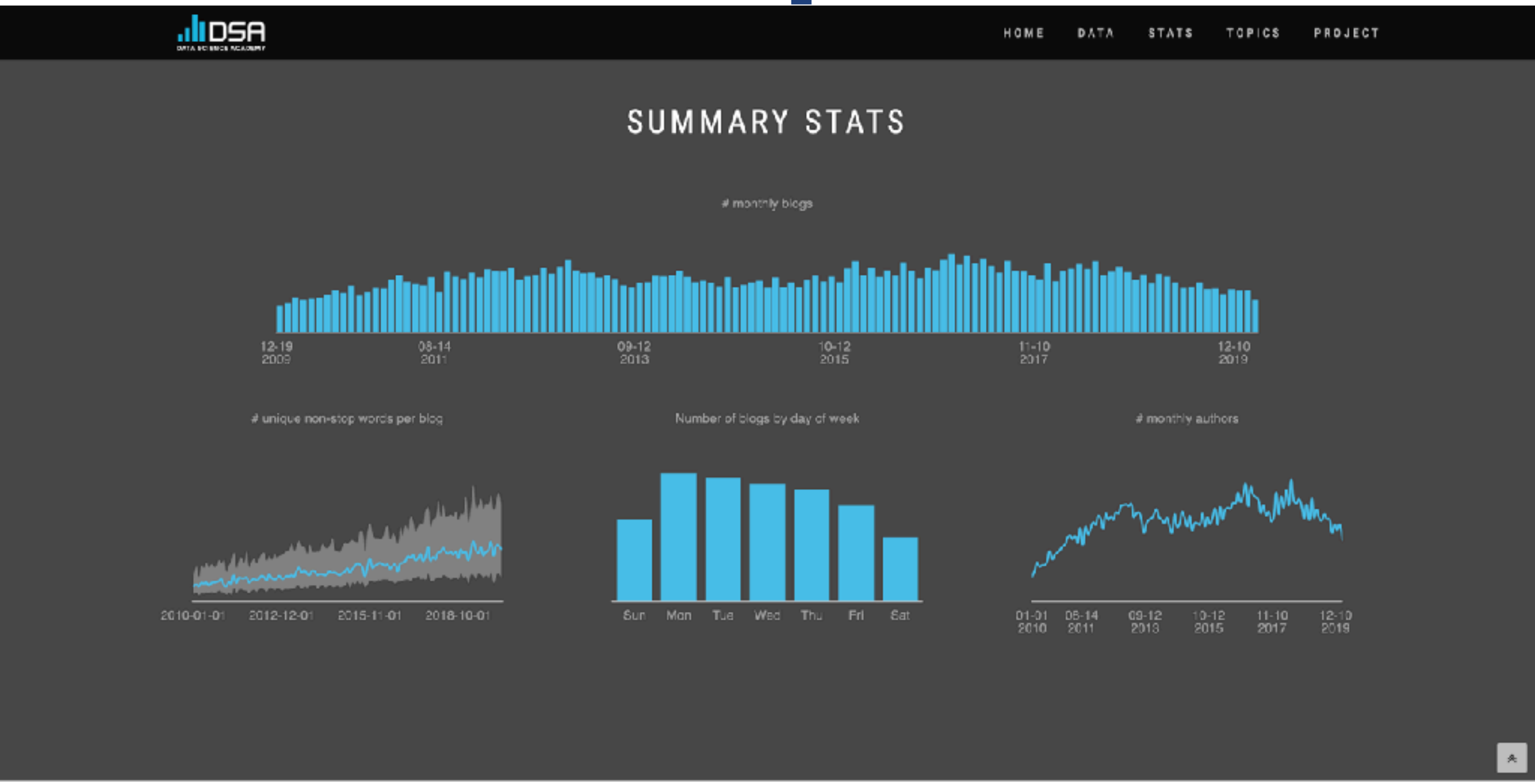
Shiny is an open-source package for R (there is a Python version now).

The goal is to provide a way for analysts to turn their analyses into interactive web applications, with them needing to know HTML, CSS, or JavaScript.

In other words, all a person needs to know is R, not how the web works.

The functions of Shiny will translate all of the R code into the appropriate HTML when the app is launched (either locally or on a remote server).

Examples



2020 Grand Prize Winner

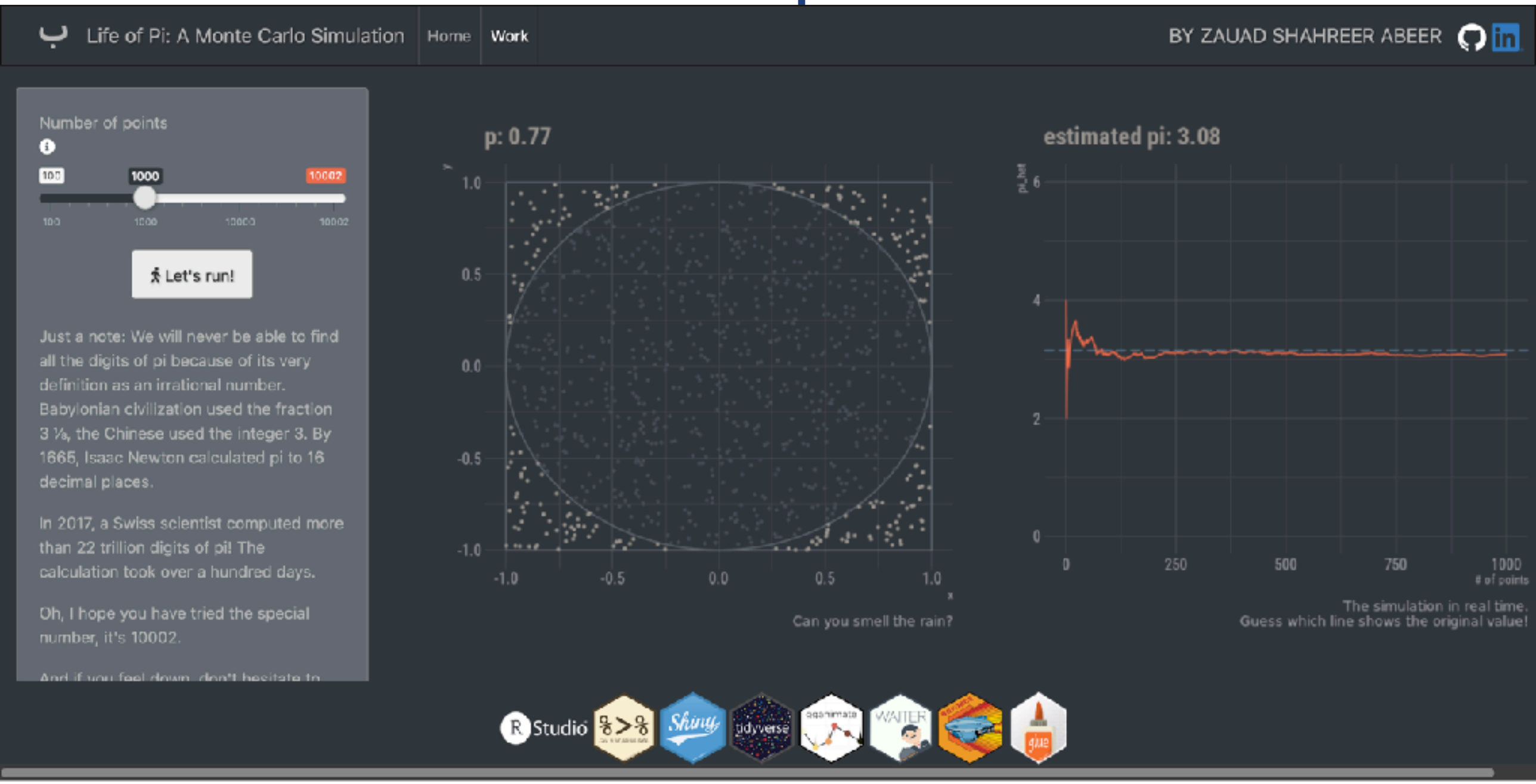
[Blog Explorer by Stefan Schleps]

11 Accessibility Errors

69 Contrast Errors

14 Accessibility Alerts

Examples

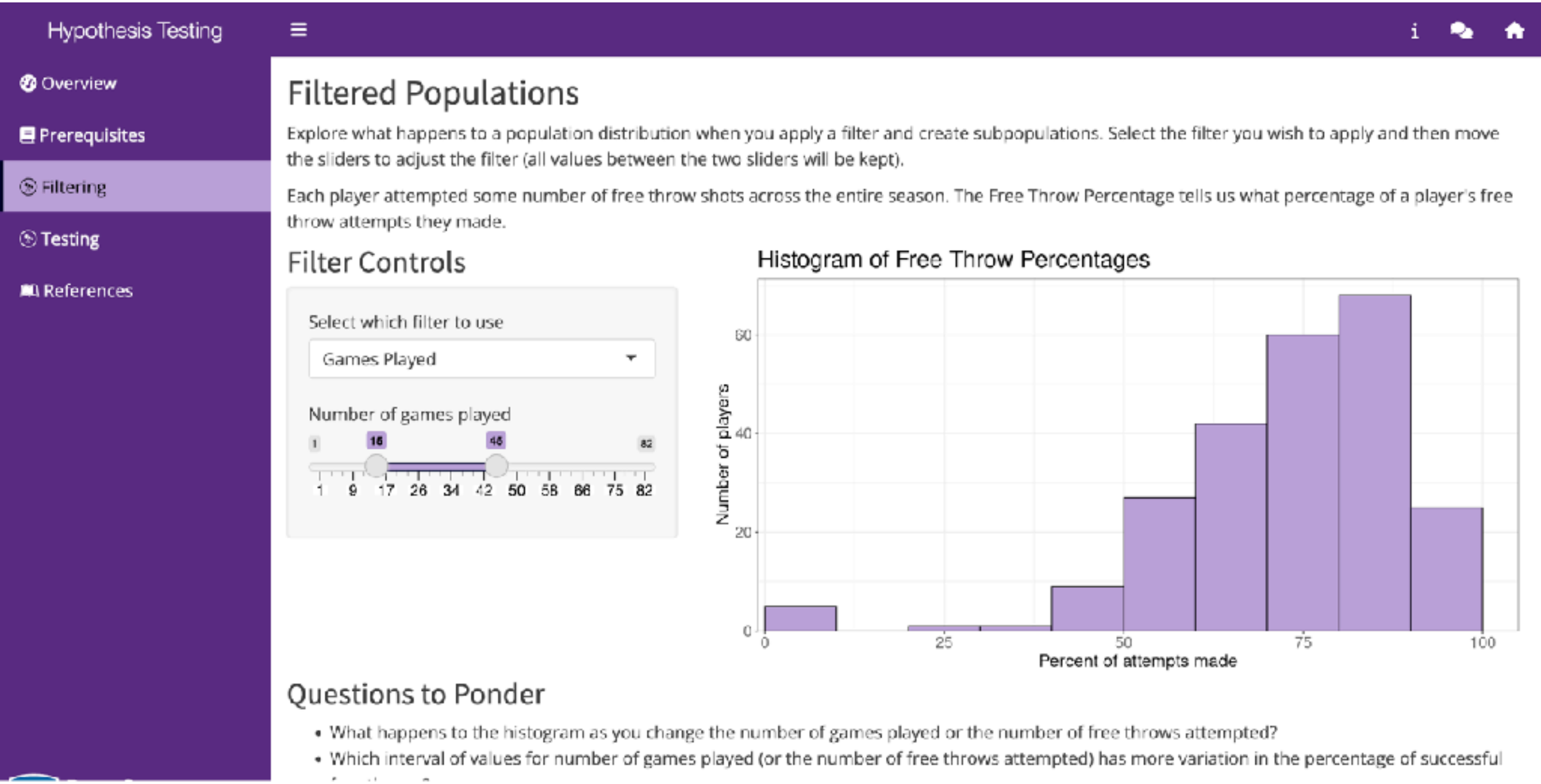


2020 Honorable Mention

[Life of Pi by Zauad Sharer Abeer]

17 Accessibility Errors
11 Contrast Errors
8 Accessibility Alerts

Examples



[Hypothesis Testing](#)
BOAST Project

3 Accessibility Errors
0 Contrast Errors
12 Accessibility Alerts

Web Accessibility

Web Accessibility Standards

Web Content Accessibility Guidelines (WCAG)

These guidelines are put together by the World Wide Web Consortium (W3C) and are based upon many rounds of discussion and research.

Current Guidance is WCAG 2.1: https://www.w3.org/TR/WCAG21/

The W3C is currently soliciting feedback from their advisory board on WCAG 2.2.

Within the guidelines there are three levels of conformance:

- A—the lowest,
- AA—midrange, and
- AAA—highest.

An underlying principle for WCAG is that web accessibility will improve the web and the usability of web content for *ALL* users.

Tools

WAVE, the Web Accessibility Evaluation Tool by WebAim, https://wave.webaim.org/

- WAVE provides a set of tools including an online checker and browser plugins.
- The plugins are great as they let you check accessibility issues in your app without having to publish/deploy your app.

The shinya11y Package, https://github.com/ewenme/shinya11y

- The shinya11y package provides an interface you can deploy in your app that will check some accessibility issues.

Coloring for Colorblindness, https://davidmathlogic.com/colorblind/

- This page will let you set your own color palettes and see how the work for different forms of colorblindness.

Key Areas
(Not a Complete List)

Creating Accessible Shiny Apps

Creating accessible Shiny apps—for any purpose—requires rejecting Posit's premise that "no web development skills are required".

The best way to create an accessible Shiny app is to start in your design process to think about accessibility.

The principles of Universal Design can help guide your process.

As you debug and test your app, make checking accessibility part of the debugging process.

In what follows, I'll talk about some of the most common accessibility errors that I see in the context of Shiny apps.

Document Structure

Page Title and Language

https://www.jumpingrivers.com/blog/accessible-shiny-standards-wcag/

HTML Tags—The Browser's Instructions

Headings (h1, h2, h3,...)

Headings are not only for stylistic choices. Their most important function is to create a structure for your content.

Paragraphs (p)

This where the majority of your text content will live.

Lists (ol, ul) and list items (li)

All list items must be in a list environment.

Inputs and Tables

Inputs

Inputs allow you to make any app dynamic as this is how you can empower your user to explore and test things out.

To make an input accessible, we must be sure that we include a meaningful/useful label.

👍 Every input should have a label argument set to concise, descriptive text (no empty—""—and no nulls).

Tables

Tables and Content Layout (🛑 Don't do this)

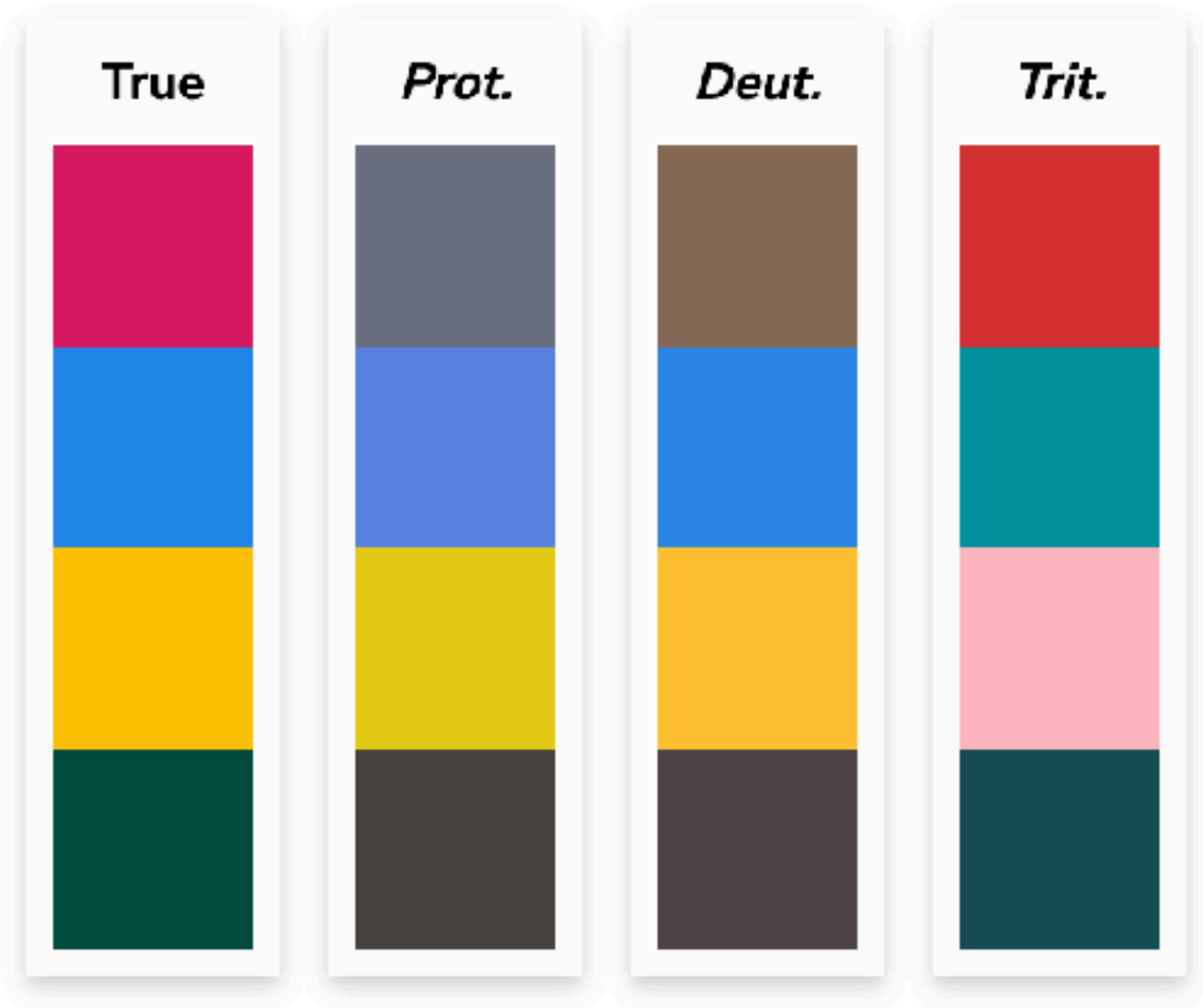
👍 Make sure to include a table caption.

👍 Use column headers and row headers when appropriate. (scope="col" and scope="row")

Check out W3C's Table Tutorial at https://www.w3.org/WAI/tutorials/tables/

Using Color

Color usage is complicated and so is color blindness.



Protanopia refers to when a person has less sensitivity to reds.
Dueteranopia refers to when a person has less sensitivity to greens.
Tritanopia refers to when a person has issues with short wavelength cones.

Color Contrast Ratios

Ratio of Text to Background

Normal Text at the AA level: 4.5:1

Large Text at the AA level: 3:1

Normal Text at the AAA level: 7:1

Large Text at the AAA level: 4.5:1

Check out WebAim's Contrast Checker

Large text is either at least [14pt and bold] or [18pt].

Ratio of Graphic and UI Components 3:1

Includes form input boarders and background

Text that is purely decorative (no info, no action) has no required contrast ratio. Logos and brand names don't either.

👍 Avoid solely conveying information through the use of color.

We'll treat the logo as being content for this exercise.

Dark Mode

Something to keep in mind is that dark mode isn't not some kind of magical solution for color accessibility.

While a large proportion of individuals might prefer dark mode, and it can provide some solutions (e.g., photophobia), this approach can bring its own set of accessibility challenges.

Halation is the spreading of light beyond boundaries, creating a hazy, foggy, or blurry effect.

According to one estimate, ~1 in 3 people in the United States has astigmatism. Astigmatism can cause the halation effect.

👍 Avoid using white text on a pure black background.

👍 If using dark mode, provide users with a toggle option.

Alt Text

How do we provide a faithful description of an image that is useful?

Statistics and data science (overly?) rely upon visualizations to convey thoughts and ideas.

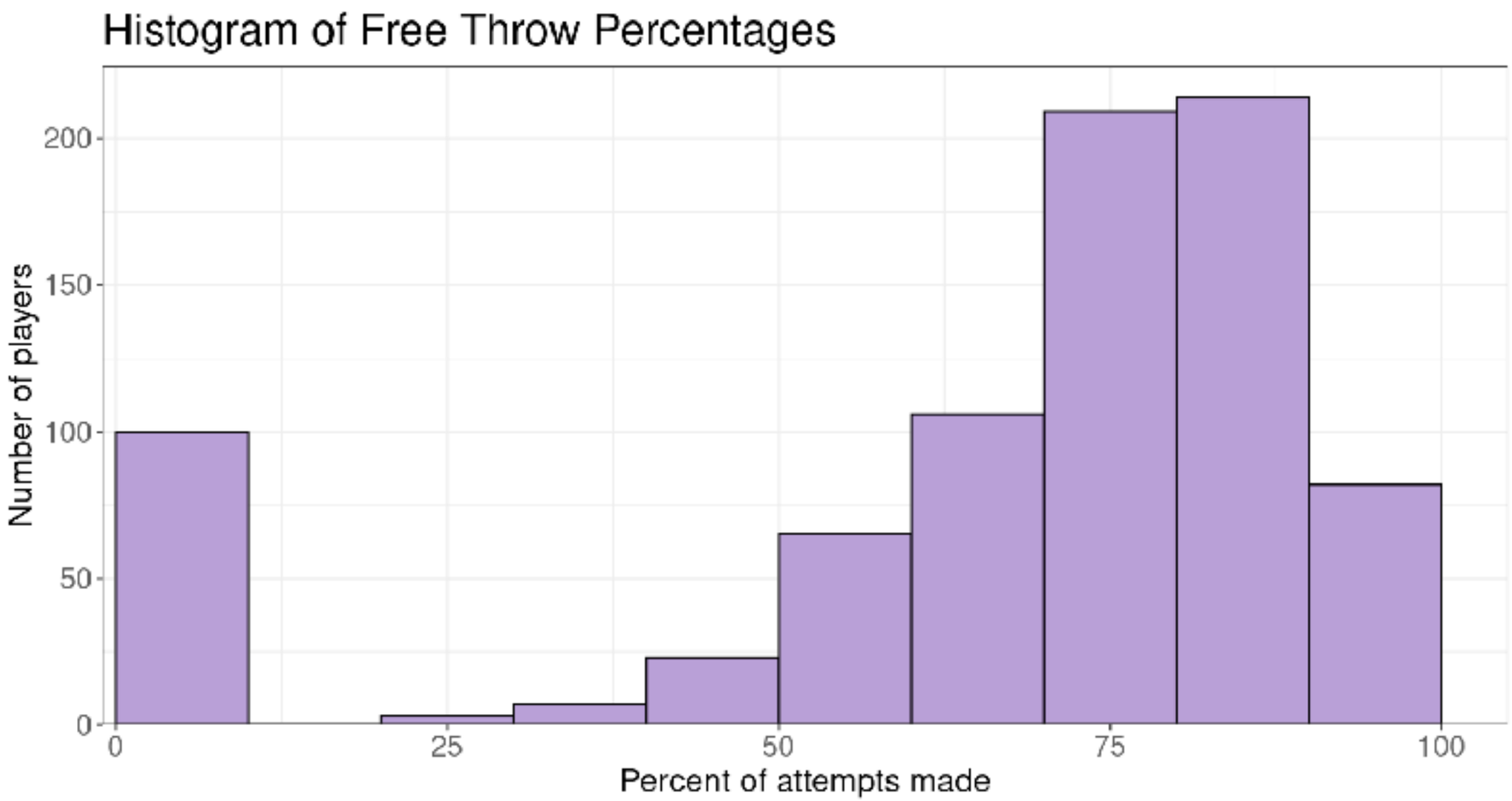
Shiny's History with Alt Text for Plots

- Pre-Version 1.6: No built in support for alt text in renderPlot
- Version 1.6: The creation of the alt argument to renderPlot
- Current: The alt argument isn't a required argument and will default to problematic text (i.e., "Plot object")

Writing Alt Text

The strict length for alt text is 100 character; use them wisely!

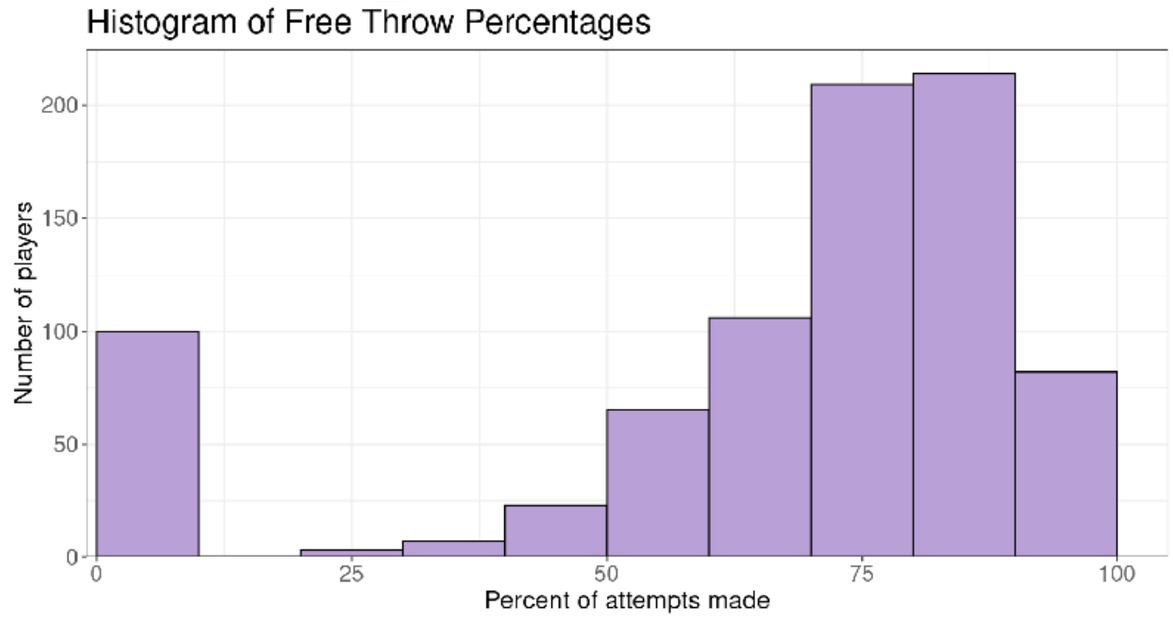
You can go up to ~150 characters but may get flagged.



There is a large number of players with a zero for their percent, then a gap between them and the next group of players. The graph is left-skewed with an average of around 67%.

A Package to Help

The BrailleR package is a promising tool that can help in generating descriptions of graphics generated in R. However, not all graphing packages and plot types are supported.



A geom_bar will be vertical or horizontal rectangular bars. They can be of varying widths. Depending on the type of bar chart they may be touching or not and could even be stacked on top of each other. geom_histogram, geom_bar and geom_col all create the same ggplot object just with different settings to get different results. geom_histogram divides a single continuous variable into bins and counts the number of observation in each bin then displays the counts with bar heights. geom_bar makes the height proportional to the number of observation in each group. geom_col make the height represent values in the data. A sighted user will be using either the height or area of the bar to determine the proportion/count of a certain group.

Writing Alt Text

When you are dealing with complicated figures as well as any figure that you need more than 100 characters to describe, keep in mind that EVERYONE is going to need help.

Accessible Rich Internet Applications (ARIA) can help.

Write your longer description as text on the page (in a p tag) and assign that element a unique id.

Then for the figure, provide a short bit of alt text and use the aria-describedby and pass the id of the description. This will create a link for screenreader technology.

Reactivity and Alt Text

One the most powerful elements of Shiny apps is the reactivity. This allows users to explore ideas, observe changes over time, and engage in What If scenarios to see what happens.

Much of this reactivity has a visual component which means that the alt text for one instance of the plot may not be appropriate for another.

Something that I've made use of is the fact that the alt text argument is part of the reactive environment. Thus, I can update the alt text view a switch or by selecting the appropriate row of a table of alt text.

You can also pass values of your inputs into the alt text for updates.

An Invitation

An Invitation

Whenever you create something to communicate with others, think about accessibility.

JSM 2023's theme of One Community is an excellent reminder that as we seek to better our communities and world, empower people, we should take whatever actions we can to ensure that everyone has access.

- **Do what you can to support the continued development of tools and resources to emphasize accessibility.**
- **When working with others on websites, documents, apps, etc., ask about what accessibility steps they are taking.**
- **Treat the absence of accessibility as a breaking bug in your code.**

Thank You!

njh5464@psu.edu