DATA VISUALISATION PROJECT REFLECTION

2024-HS1-COS30045-Data Visualisation-H1

JUNE 9, 2024

STUDENT ID: 104653505
Student Name: Anh Vu Le

Word counts (2998)

Contents

1. Introduction

1.1 Brief Overview

This reflection summarizes my learning journey and contributions in the "Data Visualization" course. My partner and I developed an interactive project, applying principles and practices of information visualization to create user-friendly visual representations of COVID-19 data.

1.2 Purpose of Reflection

This reflection covers:

1. **Learning and Application of Data Visualization Concepts**: I critically evaluated existing visualizations, applied structured design processes, and used sketching and prototyping. Our final project included a website with visualizations of COVID-19 death rates, vaccination rates, and their correlations in Australia.
2. **Programming and Technical Skills**: I analyzed my programming skills, particularly in D3.js, and addressed coding challenges to create interactive visualizations, such as a choropleth map with hover effects.
3. **Project Contributions and Teamwork**: I focused on visualizing deaths and vaccination data, ensuring accessibility and comprehension, and contributed to our process book, documenting design decisions and coding practices.
4. **Adherence to Tufte's Principles and Best Practices**: We incorporated principles like data-ink ratio, graphical excellence, and layering into our design, demonstrated by our minimalist choropleth map.
5. **Personal and Professional Growth**: The project enhanced my technical skills, collaboration abilities, and critical thinking about data representation, contributing to my academic and professional development.

This reflection demonstrates my learning, contributions to the project, and the course's impact on my development as a data visualization practitioner.

2. Data Visualization Concepts

2.1 Learning Process

Over the course of the unit, I learned to effectively demonstrate data through various charts and visualizations. Weekly labs and lectures provided a structured approach to mastering these skills. Key topics included:

- **Introduction to Data Visualization**: Understanding the basics and historical context.
- **Data Visualization Design Guidelines**: Learning about graphical integrity and effective design principles.
- **Visual Variables and Perception**: Delving into how human perception affects data comprehension.
- **User Research and Interaction**: Incorporating user feedback and interactive elements to improve visualization efficacy.

1. **Data-Ink Ratio (Edward Tufte)**
    - **Description**: Tufte emphasizes maximizing the data-ink ratio, which means using minimal ink to display the data clearly. As Tufte asserts in *The Visual Display of Quantitative Information*, “Above all else, show the data.”
    - **Application**: In our project, we applied this principle by stripping down unnecessary grid lines, labels, and embellishments from our visualizations. For instance, in our choropleth map visualizing COVID-19 deaths across Australia, we ensured the map was clean and focused solely on conveying the death rates with clear, distinct color gradations.
    - **Outcome**: This resulted in a more readable and less cluttered map that effectively communicated the severity of COVID-19's impact.
2. **Chartjunk Elimination (Edward Tufte)**
    - **Description**: Tufte argues against "chartjunk" – non-data ink or redundant visuals that do not contribute to understanding the data. As he noted, “Graphical excellence requires telling the truth about the data.”

- **Application**: We avoided using 3D effects, excessive shading, and irrelevant images. In our bar charts representing health expenditure, we used simple, flat bars with meaningful color differentiation to represent different states.
- **Outcome**: The clean and straightforward design made it easier for users to compare health expenditures across states without distraction.

3. **Small Multiples (Edward Tufte)**
    - **Description**: Tufte's small multiples technique involves using a series of similar graphs or charts using the same scale and axes to compare different pieces of data. As Tufte puts it, "Small multiples, whether tabular or pictorial, move to the heart of the matter: repeated comparisons."
    - **Application**: For our vaccination visualization, we used small multiples to display vaccination rates over time across different states. Each small multiple had the same axes, making it easy to compare trends and patterns.
    - **Outcome**: This approach provided a clear comparative view, allowing users to quickly understand how vaccination rates evolved across different regions.
4. **Color Usage and Perception (Noah Iliinsky)**
    - **Description**: According to Iliinsky in *Designing Data Visualizations*, color should be used purposefully to highlight important data points and facilitate comprehension. As Iliinsky notes, "Color is one of the most powerful tools available to the designer for both functional and aesthetic reasons."
    - **Application**: In our project, we used color strategically. For example, red was used to represent COVID-19 deaths, as it psychologically attracts attention and signifies danger. Blue, the opposite of red, was used to represent vaccination rates, signifying their role in combating the pandemic.
    - **Outcome**: The deliberate color choices made the visualizations intuitive. Additionally, we ensured accessibility by running our color schemes through tools like davidmathlogic for colorblind users.
5. **Choosing Suitable Charts for the Project**

Description: Throughout the unit, we became familiar with a variety of chart types, including bar charts, line charts, scatter plots, and choropleth maps. Choosing the most suitable chart for each aspect of our project was crucial for effective data communication.

Application:

- For the health expenditure visualization, we chose a bar chart combined with a line chart to represent the expenditure of OECD countries. The bar chart allowed for easy comparison of expenditure between countries, while the line chart overlaid on it helped to show the rank and trend of Australia's expenditure over time. This combination made it easier to see both the individual amounts and the relative ranking.
- We used two bar charts to visualize vaccination rates. One bar chart displayed the vaccination rates across different states, allowing for straightforward comparison. The other bar chart focused on vaccination rates among different age groups within Australia, highlighting disparities or progress among these groups.
- The choropleth map was selected for visualizing COVID-19 deaths because it provided a geographical perspective, making it easy to see the impact of the pandemic across different regions of Australia.

Outcome: The combined bar and line chart effectively communicated both the magnitude and the ranking of health expenditures, facilitating quick comparisons. The bar charts for vaccination rates made it simple to compare across states and age groups, while the choropleth map provided a powerful visual representation of the geographical spread and intensity of COVID-19 deaths.

6. **Data Type Considerations**
   - **Description**: Different data types require different visualization techniques to be effectively communicated.
   - **Application**: We visualized categorical data using bar charts, ordinal data using color gradients on our choropleth map, and continuous data through line charts for trends over time. For example, transforming

health expenditure data from millions to billions made the data more comprehensible and manageable.
    - **Outcome**: Tailoring the visualization technique to the data type improved clarity and ensured the audience could quickly grasp the essential insights.
7. **User Interaction and Feedback**
    - **Description**: Engaging users through interactive elements can significantly enhance the usability and comprehension of visualizations.
    - **Application**: Initially, we implemented hover effects to display data upon mouseover, but this proved inconvenient for users who had to keep the mouse stationary to view information. We revised this approach to click-to-view details, improving the user experience.
    - **Outcome**: The revised interaction model was more user-friendly and allowed users to explore the data at their own pace without inconvenience.

2.2 Conclusion

Applying the principles of data visualization from experts like Edward Tufte and Noah Iliinsky significantly enhanced the quality of our project. By focusing on clarity, eliminating unnecessary elements, and using colour and interaction strategically, we created effective and engaging visualizations. This reflection highlights the depth of my learning and the practical application of these principles to achieve a high standard in our final project.

3. Data Programming Concepts

3.1 Solution and Approaches Used

In software development, a step-by-step approach is crucial for managing complexity and ensuring robustness. Edward Tufte's principle, "Above all else, show the data," guided our focus on clarity and precision in each step of our project. Loading the map data presented our first challenge. Using d3.json('aus.json'), I efficiently loaded geographic data. Initial issues with reading and formatting datasets for deaths and vaccinations were resolved using d3.csv. Creating map elements like state boundaries and applying a color scale based on death rates proved complex. Incorrect color mapping was a hurdle, which I addressed by accurately calibrating our color domain. Adding interactivity posed performance issues due to excessive DOM manipulations. We optimized interactions by refining event listeners, ensuring smooth user interactions. This methodical approach, though challenging, ensured each component was robust and well-integrated.

Evidence of Research: book named "*Interactive Data Visualization for the Web: An Introduction to Designing with D3" by* Murray, S.

Code of color adjustment:

```
// Set color scale for the map based on death totals
var colorScaleOrdinal = d3.scaleThreshold()
    .domain([0, 1000, 2000, 3000, 4000, 5000, 6000]) // Define thresholds for color scale
    .range(["#fee5d9", "#fcbba1", "#fc9272", "#fb6a4a", "#ef3b2c", "#cb181d", "#a50f15", "#67000d"]); // Define color range
```

Defines a color scale for the map based on death totals, using D3's scaleThreshold() function to create a color scale with defined thresholds and a range of colors.

3.2 Loading Map

When first loading the map, the aus.json map data from online sources did not work because D3 cannot read JSON files where geometry elements are in one line. D3 requires each state to be on a separate line. After systematically researching online sources, including forums and D3 documentation, I learned that the JSON structure needed to be adjusted. I manually went through the JSON data and separated each state into different lines to make it compatible with D3. This allowed the map to load correctly. Research on Stack Overflow and D3 documentation provided insights into the required JSON structure.

Evidence of Research: Tutorials on D3's official site and practical examples from the D3 community guided the implementation of this solution

Code of reading map:

```
// Loading the map data with properly structured JSON
d3.json('script/aus.json').then(function(geoData) {
    var projection = d3.geoMercator().fitSize([width, height], geoData);
    var path = d3.geoPath().projection(projection);
    svg.selectAll("path")
        .data(geoData.features)
        .enter().append("path")
        .attr("d", path)
        .attr("class", "state");
});
```

Explanation: The code correctly loads the map data using D3's d3.json() function after restructuring the JSON file to meet D3's requirements.

3.3 Libraries Used

Libraries provide pre-built functions and tools that streamline development and ensure consistency. Noah Iliinsky in *Designing Data Visualizations* notes, “Libraries are like

having pre-fabricated pieces in a complex structure – they help build efficiently and reliably." We primarily used D3.js, facilitating dynamic visualizations. Initial challenges in loading and processing JSON data were overcome using d3.json. Labeling states with abbreviations instead of full names required modifying JSON data. Misaligned labels initially caused clutter, resolved by adjusting the projection function. Using D3.js, I created state borders and applied color scales for the choropleth map. Calibration issues led to inaccurate color scales, fixed by adjusting the color domain.

Code for labelling state and coloring statewise:

```
.style("fill", function (d) {
    var stateName = d.properties.name;
    var stateCode = StateCodes[stateName];
    return colorScaleOrdinal(deathTotals[stateCode]); // Set fill color based on death totals
})
    return StateCodes[d.properties.name]; // Add stateCode as attribute for easy reference
})
```

Sets the fill color of each state on the map based on its death totals, using the previously defined color scale. Also adds the state code as an attribute for easy reference.

3.4 Loading and Processing Data

Proper data loading and processing are essential for accurate and responsive visualizations. Initially, using d3.csv() to read multiple CSV files led to complications and data collisions, making the data handling process cumbersome and prone to errors. Edward Tufte's principle of maximizing the data-ink ratio influenced our approach to efficient data handling. Post-cleaning, we faced a problem with the default bar chart due to duplicated data. Using d3.csv, we ensured proper data handling but identified that duplication caused inflated totals, which we fixed by dividing totals by two. Additionally, updating bar charts based on user interactions required refining data

binding for smooth transitions. By researching online, particularly on MDN Web Docs and examples in the D3 community, I discovered Promise.all(). This function allows multiple datasets to be loaded simultaneously, simplifying the process and reducing errors. MDN Web Docs provided clear guidance on using Promise.all() for handling multiple asynchronous operations.

Evidence of Research: MDN Web Docs provided clear guidance on using Promise.all() for handling multiple asynchronous operations.

Code in deaths.js:

```
// Load the data files (CSV and JSON)
Promise.all([
    d3.csv("dataset/deaths.csv"),
    d3.csv("dataset/vaccination.csv"),
    d3.json("script/aus.json")
]).then(function (files) {
    var deathData = files[0];
    var vaccinationData = files[1];
    var geoData = files[2];
```

Loads the data files (CSV and JSON) asynchronously using Promise.all() to handle multiple file loading. It then assigns the loaded data to variables for further use.

3.5 Interactivity

Interactive elements play a crucial role in engaging users and enhancing the usability of visualizations. Reflecting on Tufte's principle that "Graphical excellence consists of complex ideas communicated with clarity, precision, and efficiency," When I started implementing interactivity, we encountered several challenges. Hover and click effects initially caused significant performance issues, which was frustrating. To tackle this, I focused on optimizing interactions by refining event listeners and minimizing DOM updates. This approach led to noticeable performance improvements. One specific issue was the overlapping of map elements, which I resolved by carefully adjusting

their positioning and layering. Adding tooltips was another enhancement I made to provide additional information on hover, significantly improving the user experience.

Evidence of Research: D3 forums and examples provided insights into managing z-index and event listeners to ensure tooltips functioned correctly.

Code in deaths.js:

```
// Add mouseover when mouse hover on state
    .on("mouseover", function (event, d) {
        var stateName = d.properties.name;
        var stateCode = StateCodes[stateName];
        d3.select(this)
            .transition()
            .duration(200)
            .style("fill", function() {
                return stateCode === selectedState ? "grey" : "black"; // Change color on hover
            });
        showTooltip(event, d); // Show tooltip on hover
    })
```

Adds a mouseover event listener to each state on the map, changing the fill color of the state and showing a tooltip when the mouse hovers over it.

3.6 Creating and Updating Default Bar Charts

Default views provide an initial overview of data, dynamically updated based on user interactions. Guided by Tufte's principle of "showing comparisons," we focused on creating default bar charts for total deaths and vaccinations. Initially, we encountered data processing errors that led to incorrect totals. By adjusting calculations to prevent

duplication, we ensured accuracy. Updating bar charts based on user interactions required refining data binding for seamless transitions.

Moreover, we faced significant challenges when implementing interactive features for the bar charts. Clicking on a state would either fail to display any bar charts or cause multiple charts to appear simultaneously, cluttering the interface. Additionally, there was an issue where bar charts did not disappear correctly upon clicking again. To resolve these issues, I researched online resources, including tutorials and examples on D3's official site. By implementing a function to remove existing bar charts before displaying new ones, I ensured that only the relevant charts were displayed and resolved the issue of charts not disappearing correctly.

Evidence of Research: Tutorials on D3's official site and practical examples from the D3 community guided the implementation of this solution.

Code of Updating bar charts:

```
function updateBarCharts(deathState, vaccinationState, stateName) {
    removeBarCharts(); // Clear previous charts

function createDefaultBarChart(deathData, vaccinationData) {
removeBarCharts(); // Clear previous charts

// Add function click on state
        .on("click", function (event, d) {
            var stateName = d.properties.name;
            var stateCode = StateCodes[stateName];
            if (selectedState === stateCode) {
                selectedState = null; // Deselect state
                createDefaultBarChart(deathData, vaccinationData); // Reset to default bar
chart
```

Defines functions for updating and creating bar charts, including removing previous charts and handling click events on states to display bar charts accordingly.

3.7 Debugging

Debugging ensures code is error-free and functions as intended. Donald Knuth states, “The most important thing in debugging is to learn how to ask the right questions.” Throughout the project, debugging identified and resolved issues like data duplication, incorrect color mapping, and performance problems. By asking the right questions, we pinpointed and fixed data processing logic. Verifying color scales and data properties resolved color mapping issues. Performance optimization was achieved by refining event listeners and minimizing DOM manipulations. Integrating JavaScript and CSS elements initially conflicted, resolved by using SVG for consistent styling.

Code of editing style in text and bar charts:

```
    .style("font-weight", "bold") // Set font weight
    .style("fill", "black") // Set font color
  .attr("width", xScaleDeath.bandwidth()) // Width of each bar
  .attr("height", function (d) { return barHeight - yScaleDeath(d); }) // Height of each bar
  .attr("fill", "maroon"); // Bar color
```

4.Teamwork and Peer Assessment

At the beginning of the project, my role involved several key tasks: finding the dataset, filling out the outline of the process book, and collaborating with my teammate to finalize the project topic. My teammate was responsible for finding the topic, outlining the website, and contributing to the process book. To keep the team on track, we implemented a strategy of completing weekly tasks before our Monday lab sessions. Initially, we aimed to finish a chart each week, but this approach proved ineffective. We then reallocated tasks: my teammate focused on the expenditure and line bar charts, while I took on the choropleth map of deaths in Australia and the bar chart for deaths and vaccinations over the year. This adjustment helped us progress more efficiently.

Midway through the project, we encountered several challenges, particularly in weeks 9 and 10. Our primary conflict arose from differing strategies: I wanted to complete the map first and then integrate it into the index.html file, while my teammate preferred developing the code directly within the index.html file. Initially, I resisted her approach, leading to conflicts. Eventually, we adjusted our strategy, with me focusing on developing the complete website and implementing all charts, and my teammate providing the base code, which I adapted and enhanced with CSS and other elements. This compromise allowed us to move forward more effectively.

Throughout this experience, I learned the importance of flexibility and listening to others' opinions, especially when working in a team. I realized my stubbornness and improved my collaborative skills. For instance, after the conflict, I consistently offered help and sought feedback from my teammate, which improved our teamwork. When I completed my tasks, I would send updates to my teammate and offer assistance if needed. This proactive communication helped in situations where she needed help with datasets, process book entries, or website enhancements. This approach not only resolved conflicts but also strengthened our teamwork.

Despite my teammate's strong data visualization skills, there were times when I doubted her ability to complete tasks due to past conflicts. For instance, she made changes to the website without informing me, leading to confusion and misunderstandings. However, we addressed these issues by establishing clear communication and task separation. We assigned tasks separately with clear deadlines and expectations and implemented regular "project stand-ups" to ensure tasks were progressing smoothly. This experience taught me the value of adaptability in collaborative environments. Being open to change and willing to adjust strategies

is key to successful teamwork, and this project helped me develop these crucial skills.

New Skill Acquired: Adaptability in Collaborative Environments One significant skill I developed was adaptability in a collaborative environment. This skill is crucial for integrating diverse viewpoints and working harmoniously within a team. For instance, when our initial strategies failed, I adapted by taking on new responsibilities and supporting my teammate more actively. This flexibility not only resolved conflicts but also enhanced the overall quality of our project.

5.Written Communication

Logical Structure and Flow

I ensured a logical progression of ideas by structuring the document into clear sections, each addressing a specific aspect of teamwork and peer assessment. Each paragraph contributes meaningfully to demonstrating the learning acquired during the semester. Transitions between sections are smooth, guiding the reader seamlessly from one point to the next.

Clarity and Precision

I used precise language and clear examples to communicate ideas effectively. For instance, describing specific tasks and strategies, detailing conflicts, and outlining resolutions all contribute to a comprehensive understanding of the team dynamics and my role.

Overall Excellence

The document reflects excellence in written expression by presenting a well-rounded view of my contributions, challenges faced, and lessons learned. Feedback from peers and instructors was incorporated to refine the content, ensuring it meets high standards of written communication.

6.Conclusion

In summary, this project was a valuable learning experience in data visualization and programming, significantly enhancing my technical skills, teamwork abilities, and understanding of effective data representation. By applying principles from experts like Edward Tufte and Noah Iliinsky, we created a clean, informative, and user-friendly visualization of COVID-19 data in Australia. Through challenges and conflicts, particularly in programming and team collaboration, I developed adaptability and problem-solving skills, which are crucial for future projects. This reflection highlights the comprehensive learning journey, illustrating the application of theoretical principles to practical challenges and underscoring my growth as a data visualization practitioner.

7. References

Murray, S. (2020). *Interactive Data Visualization for the Web: An Introduction to Designing with D3*

Wattenberger, A., & Murray, N. (Eds.). (2020). *Fullstack D3 and Data Visualization: Build beautiful data visualizations with D3.* Fullstack.io.

Jackson, P. (2016*). Mastering D3.js: Bring your data to life by creating and deploying complex data visualizations with D3.js*. Packt Publishing.

Iliinsky, N., & Steele, J. (2011). *Designing Data Visualizations*. O'Reilly Media.

Steele, J., & Iliinsky, N. (2010). *Beautiful Visualization: Looking at Data through the Eyes of Experts*. O'Reilly Media.

Tufte, E. R. (2001). *The Visual Display of Quantitative Information*. Graphics Press.

Tufte, E. R. (2006). *Beautiful Evidence*. Graphics Press.

Tufte, E. R. (1997). *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press.

TutorialsTeacher. (n.d.). Loading data from file in D3.js. TutorialsTeacher. Retrieved June 9, 2024, from https://www.tutorialsteacher.com/d3js/loading-data-from-file-in-d3js

MDN Web Docs. (n.d.). Promise.all. MDN Web Docs. Retrieved June 9, 2024, from https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise/all

D3.js. (n.d.). Community. Retrieved from https://d3js.org/community

Shaffer, J. (2016, April 20). 5 Tips on Designing Colorblind-Friendly Visualizations. *Tableau*. https://www.tableau.com/blog/examining-data-viz-rules-dont-use-red-green-together

Plotly. (2017, December 11). Maximizing the Data-Ink Ratio in Dashboards and Slide Deck. *Medium*. https://medium.com/plotly/maximizing-the-data-ink-ratio-in-dashboards-and-slide-deck-7887f7c1fab

D3.js. (n.d.). Community. Retrieved from https://d3js.org/community

Iliinsky, N., & Steele, J. (2011). *Designing Data Visualizations*. O'Reilly Media.