Responsive Data Visualization

Yaryna Korduba, Stefan Schintler and Andreas Steinkellner

706.057 Information Visualisation SS 2022
Graz University of Technology

31 May 2022

Abstract

Ensuring responsiveness is a crucial task in designing web-based data visualisations, which should be accessible on multiple devices and contexts. This task is complex and requires customisations for each specific case.

This survey collects and generalises various approaches to make data visualisation responsive. We first define responsiveness in the context of data visualisation. Further, we define a set of responsive patterns, followed by examples for a set of chart types. Last but not least, we examine software tools for ensuring the responsiveness of visualisations.

Contents

List of Figures

Chapter 1

Introduction

The 21st century is often called the era of mobile. This era can be characterised by the extensive use of mobile devices, such as smartphones, tablets and smartwatches. According to the statistics from Statista [2022], during the fourth quarter of 2021 mobile traffic (excluding tablets and desktop devices) covered 54.4% of the web traffic worldwide. This indicates a significant increase in comparison to 31.16% of the web traffic in the 1st quarter of 2015 (see Figure 1.1). As the study of Adepu and R. F. Adler [2016] reports, such preference in the usage of mobile devices might be influenced by the ease of use, high device portability, continuous connectivity and touch-screen features.

Such a rapid change poses an important question for data visualisation: How to wisely adjust the information for displaying on mobile devices? In other words how to make visualisation for mobile devices responsive. This survey is focused on techniques to adjust desktop-first visualisations for smartphones and tablets. Another popular type of mobile device is a smartwatch. These gadgets have significantly smaller screen sizes, and they are mostly used to provide specific minimalist visualisations which can be read at a glance. Moreover, in contrast to tablets and smartphones, smartwatches vary in shapes: rectangular and round [Horak et al. 2021]. Due to the fact that smartwatch visualizations differ significantly from those adjusted to smartphone, tablet and laptop screens, smartwatch data representations are intentionally not taken into account in this survey.

1.1 Responsiveness

Responsiveness, in the context of data visualisation, indicates the ability of data visualisations to adapt and react to the changes in various contexts, such as the different devices, screen sizes, environment or size of data sets [Horak et al. 2021]. As Andrews [2018b] states, the notion of visualisation responsiveness is not interchangeable with the notion of scalability. Scalable visualisations simply shrink or grow to adjust to the screen space, and they retain their visual appearance in the process. In contrast, responsive visualisations employ wiser adjustments. In addition to simple scaling, they might also change their appearance depending on particular breakpoints. Furthermore, responsive visualisations might change display density and show only the most important data on smaller screens. Last but not least, responsive visualisations adjust to different ways of interactions, such as keyboard, mouse, and touch interaction [Andrews 2018b; Andrews 2018a].

Unfortunately, many web-based visualisations are still designed with desktop-screen in mind and it is hard to explore them on smaller devices. Recently, Wu et al. [2021] conducted an exploration where they investigated 374 visualisations from public sources. It appeared that 37.9% of this sample had no adjustments between the desktop and the mobile view. On 73.8% of the studied visualisations, the researchers experienced one or more issues when viewing them on a mobile screen [Wu et al. 2021].

Our research is mainly focused on the transition of already established desktop visualisations to mobile devices, and not vice versa. The reason for that is partly the vast amount of visualisations that were

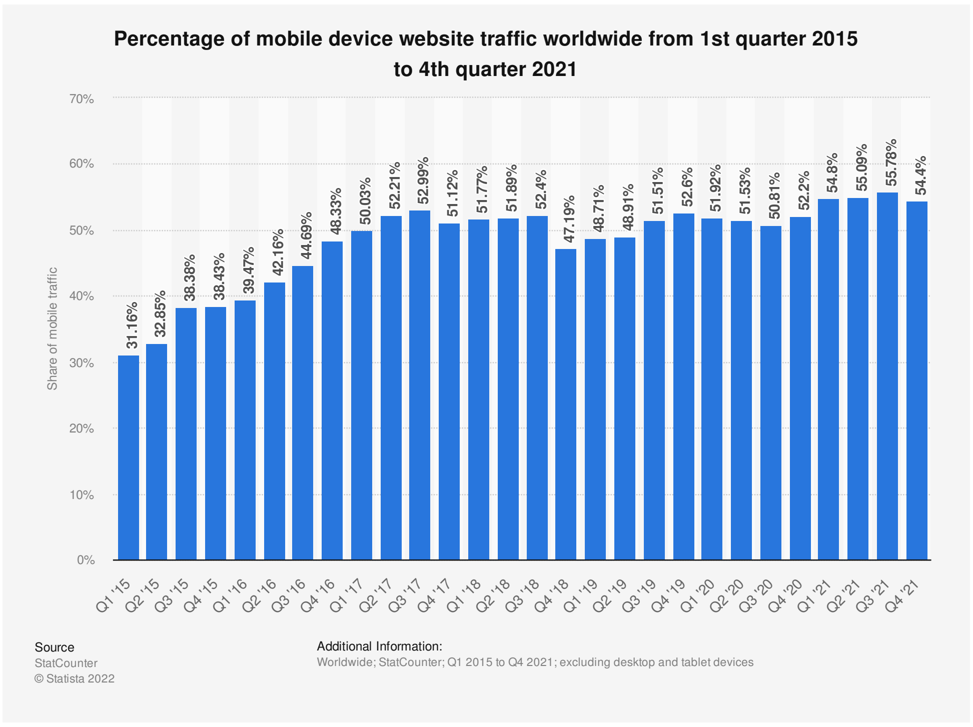

Figure 1.1: Percentage of mobile device website traffic (excluding desktop and tablet devices) worldwide from 1st quarter 2015 to 4th quarter 2021 [Visualisation exported from Statista [2022]. Used under § 42f.(1) of Austrian copyright law.]

created with the desktop view in mind. With that in mind, the necessity of designing and developing with responsiveness as the main focus comes naturally.

1.2 Mobile Advantages

Presenting data visualisation on mobile devices offers various advantages which are not available (or not as common) on desktop devices.

1.2.1 Widespread Availability

The obvious benefit of presenting data visualisations on mobile devices is their widespread availability. The demand for mobile devices and their availability are influenced by multiple factors. As Adepu and R. F. Adler [2016] say, one of the benefits of the mobile over the desktop devices is their high portability due to the small size. Some of the participants of a study conducted by Karlson et al. [2009], stated that they value having continuous phone-based access. Being always connected to the internet with their pocket-size devices enables users to receive multiple pieces of information on a daily basis no matter where they are. A lot of this information, for example, the weather forecast, stock rates, or health characteristics, is consumed through comprehensive responsive charts.

1.2.2 Touch Interaction Support

Users can perform gestures on mobile devices by using a stylus pen or fingers [Xiao et al. 2013]. As the documentation of Google [2022] states, the touch gestures can be divided into three categories: navigational, transforming and actionable. The navigational gestures include tapping, scrolling, swiping, dragging, and pinching. Transforming gestures include zooming in or out, rotating and positioning. The actionable gestures include double-tapping, pinching to zoom in or out, dragging to move elements on the screen and compound actions (such as pinching and moving one of the fingers to zoom in and rotate the view in parallel). Actionable gestures are used to invoke shortcuts or execute actions. They include but are not limited to swiping, tapping and long pressing [Google 2022]. Wisely selected and designed gestures can greatly improve the user experience in navigating and exploring the data.

1.2.3 Access to Sensor Data

Modern mobile devices are equipped with multiple sensors for measuring information. Some of these sensors are [Android Open Source Project 2022]:

- Accelerometer - measures the gravity and physical acceleration
- Gyroscope - measures the rotation level along the three axes
- Magnetometer - measures the ambient magnetic field
- Heart Rate sensor - measures the heart rate of a user
- Light sensor - measures the level of illumination

These powerful sensors can be utilised to enhance the visualisation experience on mobile devices by providing information about device rotation, ambient light, and much more.

1.3 Mobile Challenges and Limitations

While mobile devices present us with a set of advantages, they also come with their own set of unique challenges and limitations. Some of these limitations can be detrimental to the visualisation experience, if not taken into consideration.

1.3.1 Screen Size

One of the obvious challenges is the different screen sizes of desktop, tablet and mobile devices. As described by Giurgiu and Gligorea [2017], the popular width breakpoint for desktop devices is >= 992px for medium and >= 1200 for large screens. The widely used breakpoint for tablet screens is >= 768px. Small devices, like smartphones, usually have a viewport width of < 768px. Since pixels (px) are absolute units in CSS, it can often be helpful to set responsive breakpoints in relative units like em and rem, so that they scale with any changes in font sizing made by the user.

The small screen size of a mobile device might negatively influence user performance in comparison to desktop devices. In general, users have more difficulty consuming and understanding large amounts of information on smaller screens [Adepu and R. F. Adler 2016]. Therefore, the crucial question in designing responsive data visualisations is how to adjust a visualisation to the varying screen size without losing context and insights about the data, whilst retaining convenient interaction possibilities.

1.3.2 Low Performance

Every year large technological companies invest huge amounts of money into the development of smartphones, tablets and other mobile devices. Despite this fact, it is still hard to match the level of performance of mobile devices to the performance of traditional desktop devices. These challenges are connected to the size of the devices and the underlying battery technology. As Islam and Want [2014] say, the battery technology development has posed limitations on the energy requirements of computing hardware, such as graphical processing units or multi-core processor chips. Consequently, this limitation extends to the performance of heavy-load applications. Novel battery and chip solutions for mobile devices are slowly closing down the gap between desktop and mobile devices, but it is still a problem that needs to be addressed. At this point in time, interactive visualisations with heavy computational complexity and rendering requirements may be slow and inconvenient to view on mobile devices.

1.3.3 Interaction Limitations

On the one hand, touch screens in some way ease the usage of a device, as users do not need any additional gadgets, like a mouse or keyboard. On the other hand, this advantage comes with limitations. While most mouse events have a touch-screen alternative, some important interactions, such as hovering, do not have an equivalent on the touch screen devices. Müller et al. [2019] present a novel pre-touch technique for mobile devices, but these developments have not yet reached a mass market, as the technique they present in their research requires an additional motion controller. Many desktop-based visualisations employ hovering to provide an additional level of interaction. As an example, on a line chart, a tooltip with the information might be displayed on hover, while on click on a specific line, a line might become emphasised and the other data series might be greyed out.

Chapter 2

Patterns

In order to combat the various limitations of mobile devices and properly capitalise on their strengths, prior works have attempted to find ways to extract important patterns and techniques for introducing responsive elements to visualisations. While some research by Kim et al. [2021] and V. Adler et al. [2021] focuses more on the extraction of tools and patterns, others approach the problem by creating tools that help accelerate and simplify the process of conversion [Hoffswell et al. 2020b; Kim et al. 2022]. For our own set of patterns, we adopt the terminology of Kim et al. [2021], which groups patterns into categories of actions and targets. With this primary distinction of actions and targets in place, we define the following set of responsive visualisation patterns:

- P1: Scaling the Chart Down
- P2: Using Viewport-Specific Images
- P3: Data Generalisation
- P4: Semantic Zooming
- P5: Viewport-Specific Layouting
  - P5a: Automatic
  - P5b: Manual
- P6: Changing Components
  - P6a: Axes, Tick Marks, and Grid Lines
  - P6b: Labels
- P7: Using Different Charts

In the following, each section describes a pattern in detail, with a concrete usage example. Note that most patterns are not standalone and are often combined or used with general responsive web design techniques to create great web-based visualisations.

2.1 P1: Scaling the Chart Down

One of the most widely used patterns in making the visualisation responsive is scaling the chart down. This pattern is usually applied to guarantee that the chart elements do not fall out of the viewport, as illustrated in Figure 2.1. The general idea is to shrink the chart down by its height or width (or both of them) to match the viewport dimensions, as illustrated in Figure 2.2. Some modifications of this pattern are also possible. For example, one can adjust the height to the screen size while keeping the width larger than the viewport and providing a horizontal scroll. However, they are not as popular.

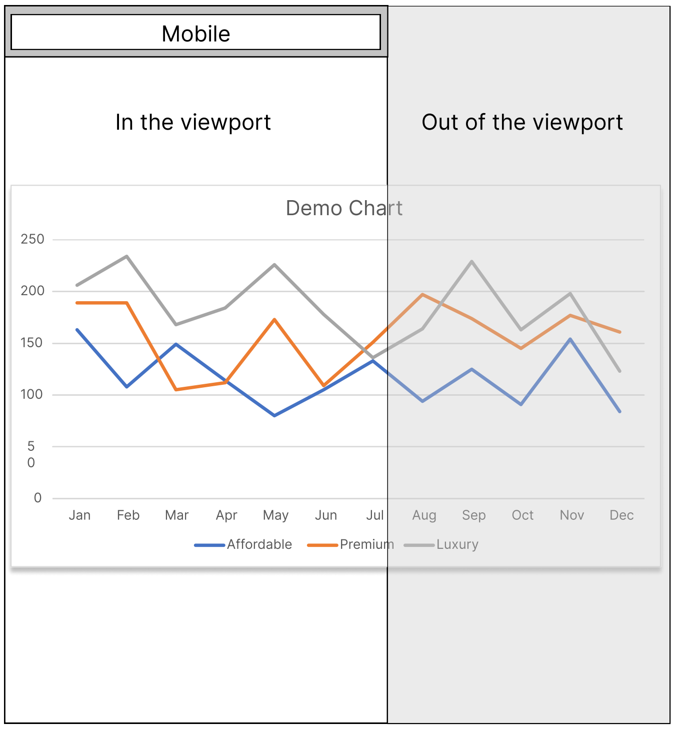

Figure 2.1: An example of a desktop visualisation, which falls out of the mobile device viewport, illustrating the necessity of pattern P1. [Image created by the authors.]

Scaling the chart down is a naïve and straightforward approach, but it can be rarely used on its own. This is because after shrinking the chart, the content and the representation itself often require careful adaptation [Horak et al. 2021]. For example, the tick labels might need adjustments to prevent overlapping and improve readability, or data points might need resizing for improved visibility.

2.2 P2: Using Viewport-Specific Images

Some visualisations can require a complex computational effort to generate. As discussed in section 1.3.2, performance on mobile devices is generally not on par with desktop computers. For that reason, rendering the visualisation as a static image on the smaller viewports instead of embedding the actual chart and doing complex computations for it in the runtime might make sense. Usually, standard screen sizes are rendered ahead of time, and CSS media queries, as recommended by W3C [2022], are used to select the right image for a particular viewport size.

Figure 2.3 displays a schematic example of this approach. The visualisation might be rendered on-demand on the desktop, whilst the mobile device always displays a pre-rendered image instead.

This approach is both easy to implement and beneficial for end-users, particularly on low-powered devices, like entry-level smartphones. One particular problem with this approach is the complete loss of interaction, making it only applicable to static visualisations, where additional user input is not supported.

2.3 P3: Data Generalization

A more transformative approach to the underlying data is used when applying data generalisation. In some cases, when large amounts of data influence visual complexity, a simple rescaling approach might not be enough. In order to display such complex data in a comprehensible way to the user, data points can be merged or even deleted. Ideally, this process leads to a chart with equal data representation, just with less visual complexity.

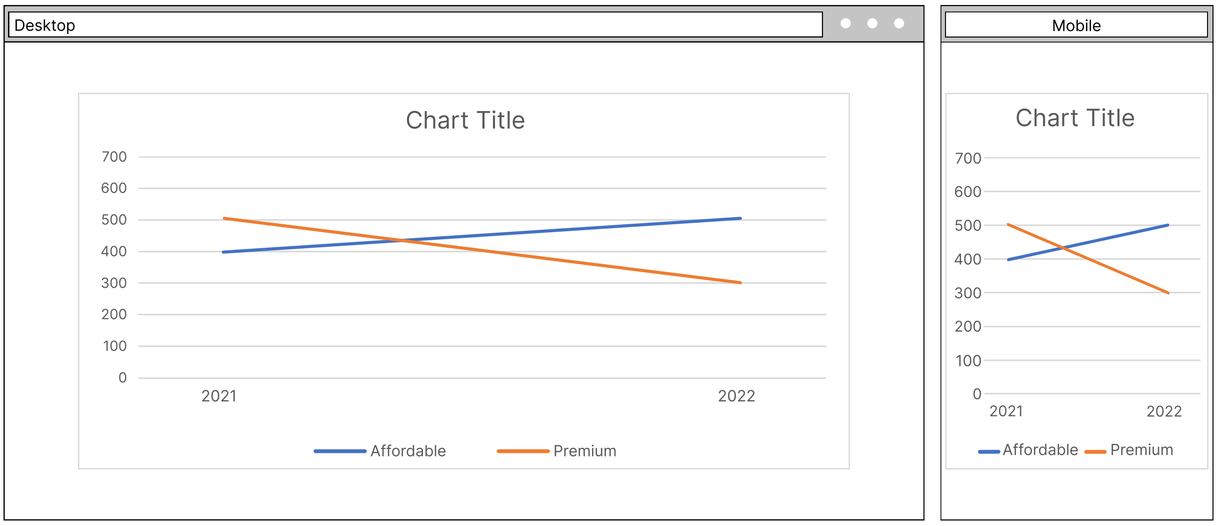

Figure 2.2: The desktop (left) and mobile (right) version of an example chart, illustrating pattern P1. The chart is scaled down according to the available width of the mobile device. The height of the visualisation is not updated since it fits into the viewport without further adjustments. [Image created by the authors.]

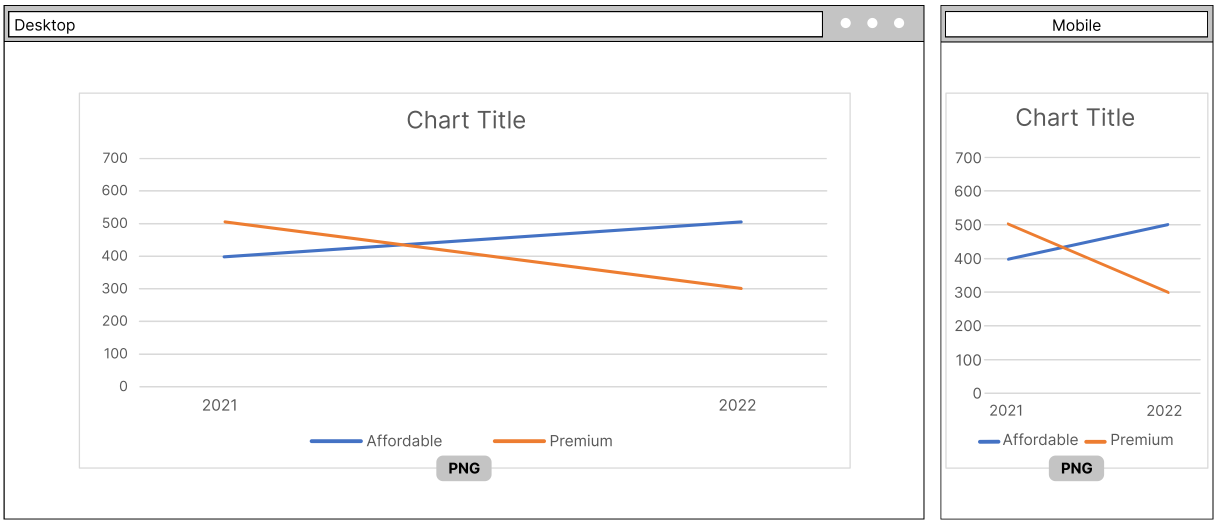

Figure 2.3: An example visualisation on desktop (left) and mobile (right) to illustrate the use of pattern P2. Depending on the available viewport dimensions, a different image is served. [Image created by the authors.]

In practice, however, generalisation comes at a cost. While a reduction in visual complexity sometimes perfectly satisfies responsive visualisation requirements, it can inadvertently skew the representation of the data since potentially significant trends or outliers might be removed or merged in the process. This unintentional transformation of the data creates the necessity to let users control the amount of generalisation, for example, by adding toggles or switches to reduce or increase the coarse-graining of the data.

In Figure 2.4, we demonstrate this approach on sample data. The visualisation on the left uses months to represent data on the horizontal axis, with one data point per line for each month. For the mobile version of the same visualisation, the months are grouped into seasons, averaging the data over three months into a single data point for that particular season. The insights obtained by looking at both of these charts might be entirely different. The mobile version of the chart suggests that there might be a downwards trend in affordable and premium real estate, while on the desktop version, it looks like

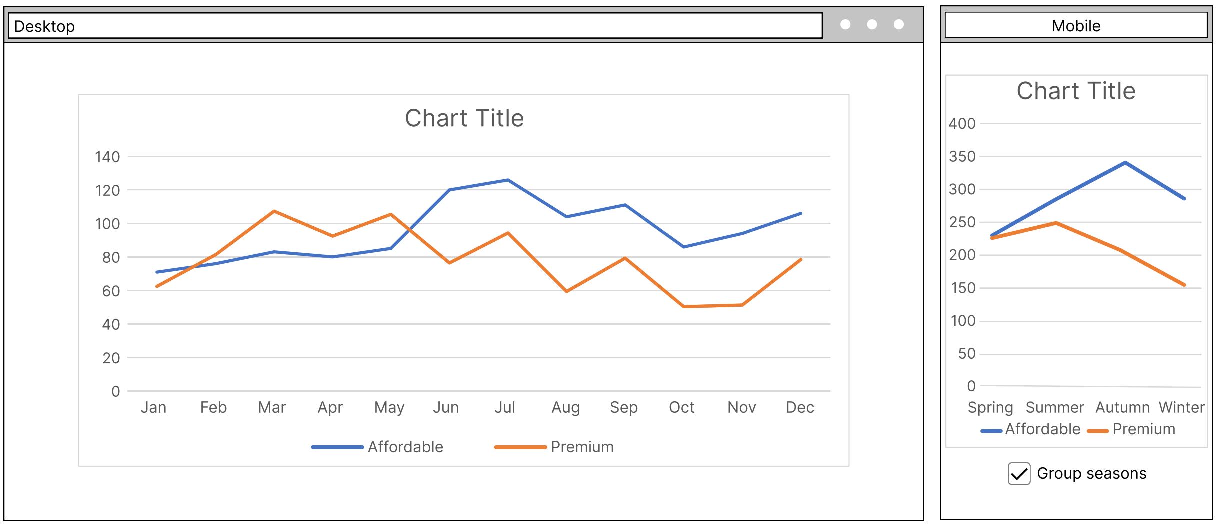

Figure 2.4: An example of a complex line chart views on desktop (left) and mobile (right) to illustrate the use of pattern P3. The mobile visualisation on the right introduces a seasonal grouping for months. [Image created by the authors.]

October and November are just low outliers. In order to let users control the amount of generalisation, the box below the chart toggles the grouping.

2.4 P4: Semantic Zooming

Instead of generalising the data directly, semantic zooming takes the currently visible amount of data into account and controls the generalisation dynamically based on the current zoom level. For users, this enables a very pleasing way of interacting with the data since the level of detail is continuously increased while zooming in and reduced while zooming back out.

This approach works well for map and graph-based visualisations since they can be intuitively navigated by zooming and panning. Depending on a chart, it might be less or more difficult to figure out how to implement zooming in a semantic way. Figure 2.5 demonstrates the difference between simple geometric zooming and semantic zooming in the medical domain [Skupin et al. 2013]. While simple geometric zooming only transforms the scaling of the data, semantic zooming changes the amount of data to display, as well as the representation.

2.5 P5: Viewport-Specific Layouting

Although a combination of other responsive patterns can already be enough to add responsiveness to a given visualisation, sometimes the layout of the data must be changed to fit the dimensions of the viewport. We call this approach viewport-specific layouting and introduce two techniques based on this idea.

2.5.1 P5a: Automatic

In visualisation, multiple charts are often combined into a set of small particles in order to gain insight into various related but slightly different sets of data [Tufte 1990]. On the desktop, these small particles are usually displayed in a grid-like format, with a set of multiple charts per row. On devices with smaller viewport size, like smartphones, this grid format is often not ideal due to width constraints.

In order to properly display the content on these devices, the arrangement of the individual charts must be updated. Instead of having rows of multiple charts arranged in a tabular structure, it is often beneficial

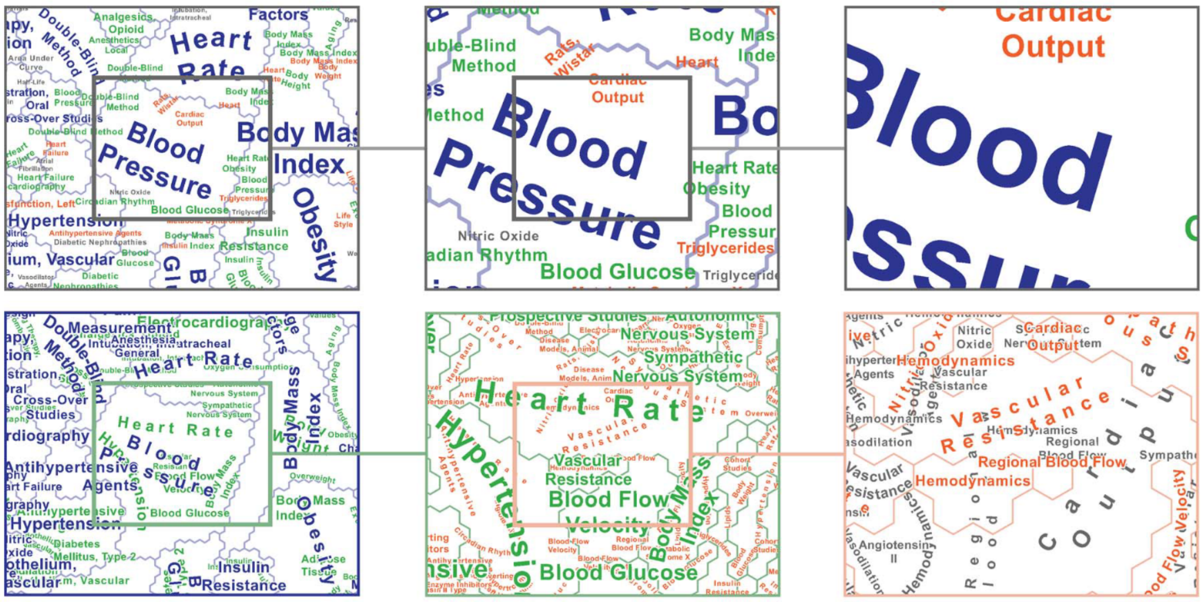

Figure 2.5: Difference between geometric zooming (on the top) and semantic zooming (on the bottom) in the medical domain, as published by Skupin et al. [2013]. Semantic zooming dynamically changes the data based on the current zoom level, whereas geometric zooming only magnifies the data. [Copyright © 2013 PLOS ONE and used under § 42f.(1) of Austrian copyright law.]

to display them in a scrollable list. Doing so allows the visualisation of a single chart to take up the entire screen width, which can significantly improve the readability of chart components. Figure 2.6 shows a set of small multiples arranged in a 2x4 grid. Instead of resizing the entire grid and fitting it to the available space, as proposed in pattern P1, the visualisations are displayed in a list on the mobile device. Users can navigate the list by scrolling, which is a basic movement on both desktop and mobile devices.

While this approach is generally straightforward to implement with common CSS media queries, it can reduce the readability and obfuscate potential insights that could be obtained if the data were set up in a grid. Instead of relying on users to correlate data points across charts by panning with their eyes, this approach can require additional forms of data annotation to convey all relevant information to users. Some horizontal relationships between data points might be entirely lost in the process.

2.5.2 P5b: Manual

Some visualisations display data in a novel and innovative format, which sometimes does not translate well to small screen width. In this case, using pattern P5a as an automatic approach is not an option, and the data needs to be repositioned manually. Often, this means radically changing the data layout of the entire visualisation to use more of the available height instead of growing horizontally.

This approach can be very time consuming since there is limited potential to automate this process. While tools as proposed by Hoffswell et al. [2020b], Kim et al. [2022] and Wu et al. [2021] and discussed in Chapter 4 might be used as assistance, some elements still require manual modifications to create an optimal fit.

In the example shown in the Figure 2.7, manual relayouting was used to constrain the visualisation width to the dimensions of a typical mobile device. Since automatic viewport-specific layouting is usually not an option for these custom visualisations, the layout was manually adapted for screens with smaller width.

This approach requires significantly more effort than its automatic counterpart we describe earlier, but it can lead to entirely novel visualisations, which, in some cases, can even improve data insight. Through seeing the data in a different layout, additional information might be obtained. A significant drawback

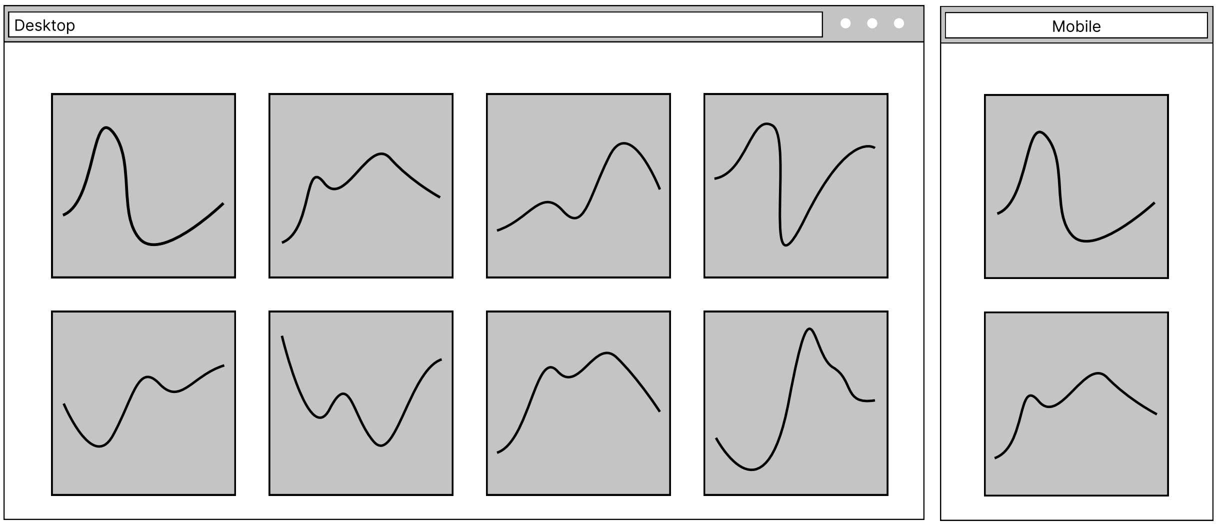

Figure 2.6: An example of a grid of small multiples on desktop (left) and mobile (right) to illustrate the use of pattern P5a. On mobile, the off-screen grid items are reached by vertical scrolling. [Image created by the authors.]

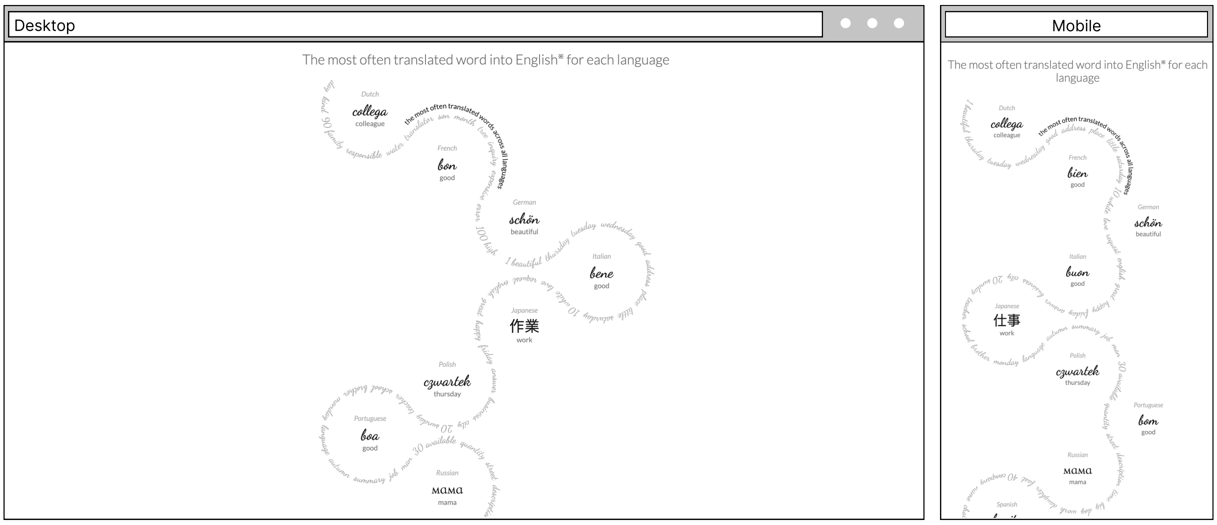

Figure 2.7: A visualisation of commonly translated words on Google, shown on both desktop (left) and mobile (right). The desktop visualisation would be too wide to properly display it on mobile devices, showcasing the necessity for pattern P5b. The layout is manually adapted to fit the available width on mobile devices, using more of the available height. [Visualisation created by Bremer [2022a]. Screenshot taken by the authors.]

of this approach is, that the relayouting needs to be done for a few generic screen sizes, and changing between them will usually be handled through media queries, as it is generally not feasible to create a new layout for all possible screen sizes.

2.6 P6: Changing Components

Sometimes changing the data or the visualisation layout is not possible due to various constraints. This is where changing components can be a useful aid in introducing responsiveness. This pattern utilises all proposed actions on the target category of "References / Layout", as introduced by Kim et al. [2021]. We distinguish between labels and other components because labels generally require a few different techniques.

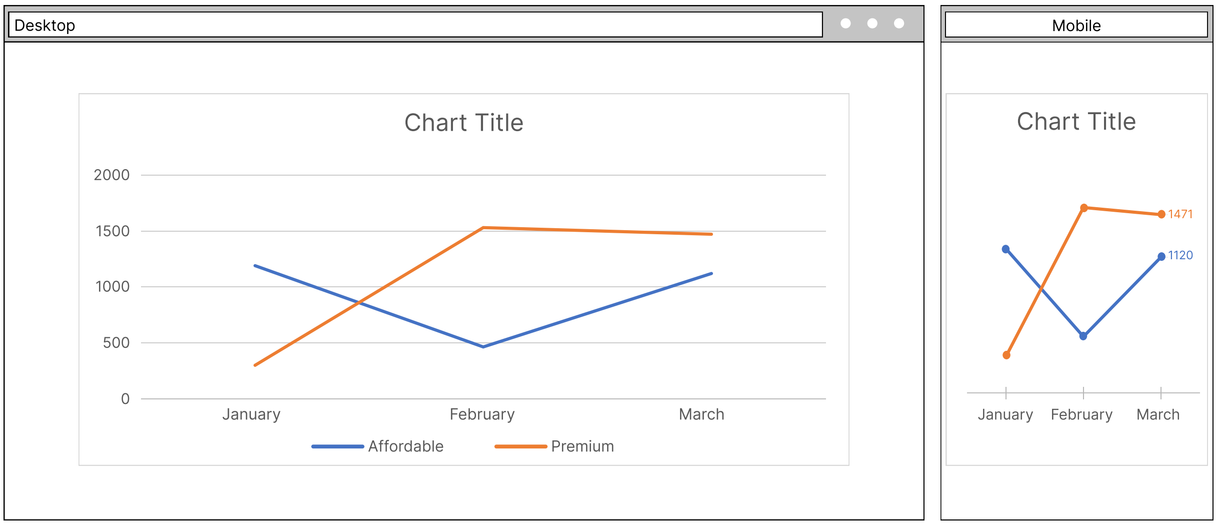

Figure 2.8: An example visualisation on desktop (left) and mobile (right) to illustrate the use of pattern P6a. On the mobile visualisation, the vertical axis is removed entirely, and an additional label is added to the last data point of each line segment. Additionally, the legend is removed. [Image created by the authors.]

2.6.1 P6a: Axes, Tick Marks, Grid Lines

First, we would like to look at axes, tick marks and grid lines. These can be repositioned or deleted to reduce clutter. Figure 2.8 illustrates this pattern on a visualisation, where on the desktop, the vertical axis is displayed normally, and an included legend helps users correlate the line colour to a set of data. On mobile, the vertical axis is removed entirely and instead, an additional label is added at the last data point of each line segment. Additionally, the legend is removed entirely.

Naturally, this approach can improve the readability of the visualisation. However, it might increase the amount of time a typical user needs to understand the data structure since some useful annotations might be removed. The process of deciding what to adapt or change is also a manual one for designers, so it can be quite time-consuming.

2.6.2 P6b: Labels

We clearly distinguish between labels and other chart components, as discussed in P6a, since text annotations, in general, can be adapted through additional techniques, such as abbreviations. When working with large numbers, it is generally widely accepted to use abbreviations for thousands (1.000 is displayed as 1K) or millions (1.000.000 is displayed as 1M) since they commonly occur in many tables and the shortening does not obfuscate the data. In articles, abbreviations could be introduced in the text and then used in the visualisation in order to reduce visual clutter.

Figure 2.9 illustrates the conversion of a desktop visualisation with the axes labels that are in the thousands to an abbreviated mobile version. Additionally, the text in the legend is also shortened.

Generally, this approach is simple to implement since the only tricky part is coming up with suitable abbreviations that are commonly understood. The era of phones has introduced a massive set of commonly used abbreviations due to character limitations when sending text messages or writing tweets [De Jonge and Kemp 2012]. While most of these are probably not suitable for professional visualisations, they can still be considered when space is limited.

The drawbacks to this method revolve around making sure that everyone understands every used abbreviation or text transformation. While some of them might be explained in the free text next to the visualisation, others might require an additional explanatory paragraph, which takes space away from the visualisation.

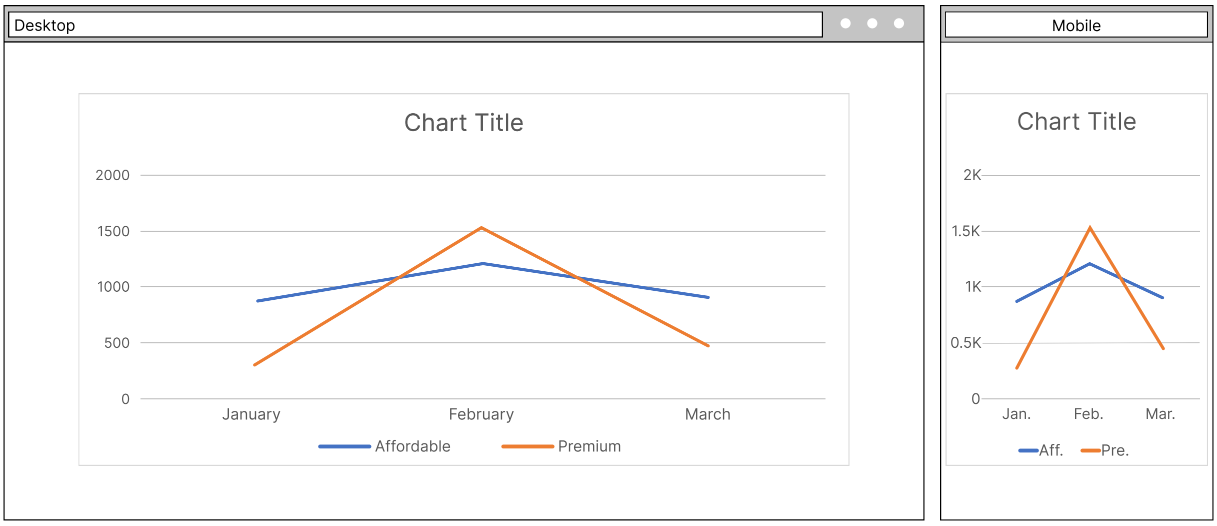

Figure 2.9: An example visualisation on desktop (left) and mobile (right) to illustrate the use of pattern P6b. On the mobile visualisation, all labels are abbreviated to decrease the visual clutter on the screen. [Image created by the authors.]

2.7 P7: Using Different Charts

In the extreme case where no other pattern is applicable or the results are not good enough, a more radical approach must be used. In these cases, the entire visualisation needs to be redone from scratch. Using a different chart type often requires a rework of the entire data structure to fit into smaller viewports. It is tough to automate and therefore requires manual, time-consuming intervention.

Figure 2.10 exemplifies this pattern. The desktop visualisation on the left uses an entirely different chart type that would not work on a mobile device. Therefore, a different chart type is exploited to embrace the smaller viewport size of a typical mobile device.

With this pattern, the visualisation is not transformed but entirely rewritten. While often not suitable due to the required time investment, this approach can lead to new insights into the data since the transformed visualisation usually displays the data from a different angle.

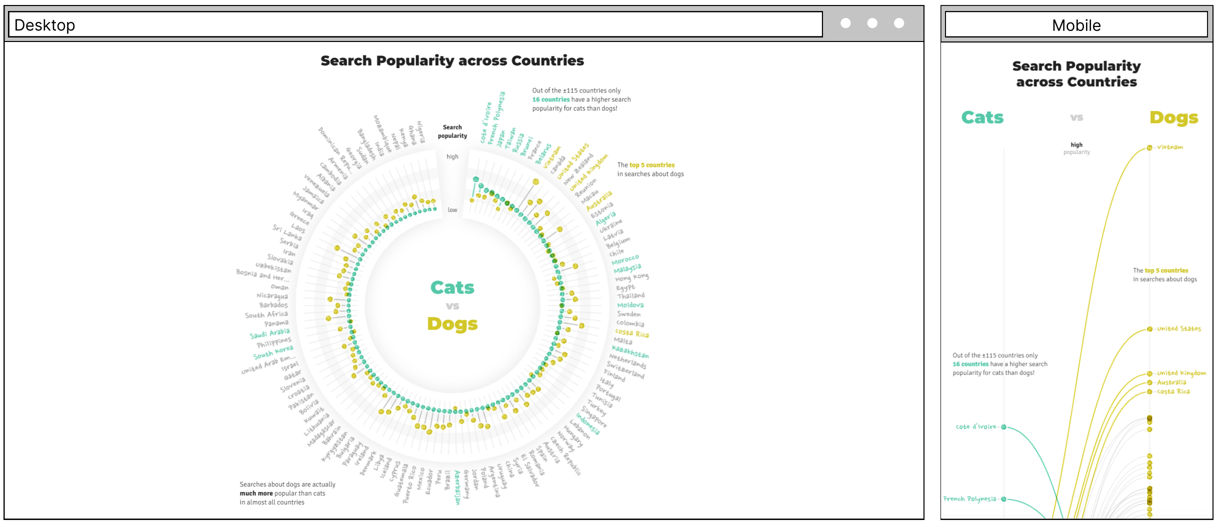
Desktop
Search Popularity across Countries
Search popularity
high
low
Cats
vs
Dogs
Out of the ±115 countries only 16 countries have a higher search popularity for cats than dogs!
The top 5 countries in searches about dogs
Searches about dogs are actually much more popular than cats in almost all countries
Mobile
Search Popularity across Countries
Cats
vs
Dogs
high popularity
Vietnam
The top 5 countries in searches about dogs
United States
United Kingdom
Australia
Costa Rica
Out of the ±115 countries only 16 countries have a higher search popularity for cats than dogs!
Cote d'Ivoire
French Polynesia

Figure 2.10: An example visualisation on desktop (left) and mobile (right), comparing the search popularity of cats and dogs to illustrate the use of pattern P7. On the desktop, countries are laid out radially around the outer circle, and a small bar chart visually compares the popularity of the search terms. On mobile, a scrollable layout is used instead of the circular arrangement, with key countries highlighted and annotated. [Visualisation created by Bremer [2022b]. Screenshot taken by the authors.]

Chapter 3

Chart Types and Examples

In this chapter, we would like to present some examples of visualisations that embrace some of the patterns we outlined in the previous section to achieve a responsive visualisation that works well on all screen sizes. We group the examples by chart type and always show a comparison between the desktop and mobile versions.

3.1 Bar Chart

Bar charts are a simple way to represent information, commonly used in media and readable by users without much effort by visually comparing bar lengths. Peter Oberrauner created an excellent example of a responsive stacked bar chart during his Master's Thesis at Graz University of Technology [Oberrauner 2022c], as shown in Figure 3.1. The bar chart responds to resize events and changes according to the available screen dimensions. In addition to scaling down and rotating the bars from a vertical to a horizontal orientation (P1), the visualisation also changes the axes (P6a) and labels (P6b) dynamically.

3.2 Line Chart

While bar charts are generally used to visualise data comparisons, line charts are usually used to indicate the change in data over a period of time. An interesting implementation of semantic zooming (P4) in a line chart was created by FRED [2022] and can be seen in Figure 3.2. Users can manually select the time span on both the desktop and mobile versions using the blue brush below the chart. On mobile, this selection is possible by using pinch the graph with two fingers.

3.3 Scatter Plot

Scatter plots generally benefit from the same techniques as the previous chart types. They represent data as points instead of lines or rectangles. Figure 3.3 is an example of a responsive scatter plot, utilising a combination of different patterns in order to embrace responsiveness and work appropriately on the mobile device. It was created during Peter Oberrauner's Master's Thesis [Oberrauner 2022c]. The chart is scaled down according to the available width (P1), some tick marks are removed on the axes (P6a), and the labels are abbreviated (P6b). We believe this chart could be improved further by introducing an optional grouping of points into similar clusters (P3).

3.4 Other Charts

After looking at some responsive examples of the most common chart types, in this section, we would like to showcase some examples of more non-traditional layouts which utilise some of the presented patterns in order to obtain responsiveness.

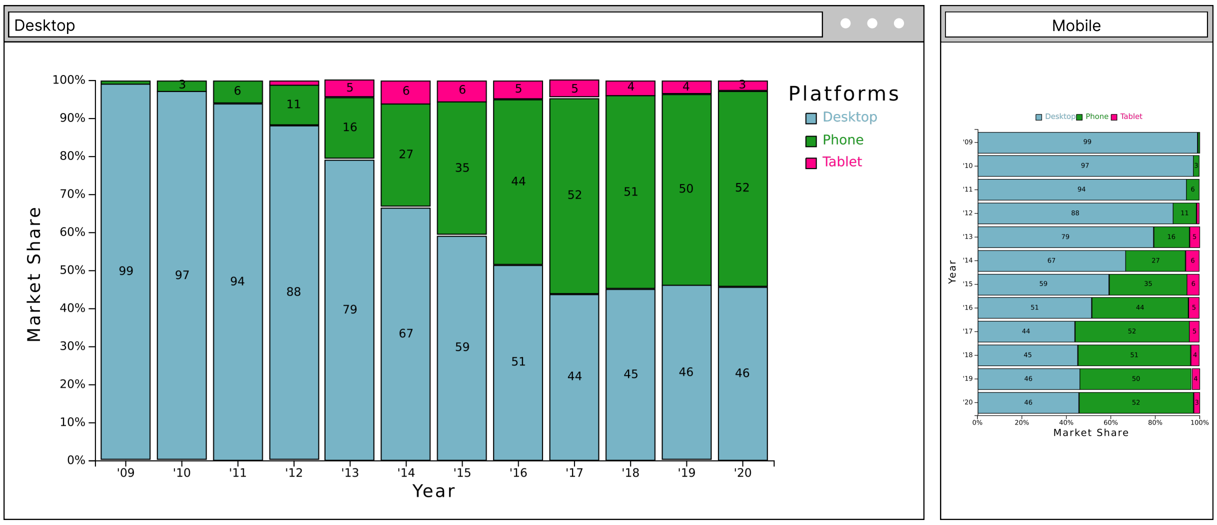

Figure 3.1: The desktop and mobile versions of the stack bar chart, with the axes rotated, legend moved, and labels abbreviated. [Visualisation created by Oberrauner [2022b], screenshot taken by the authors.]

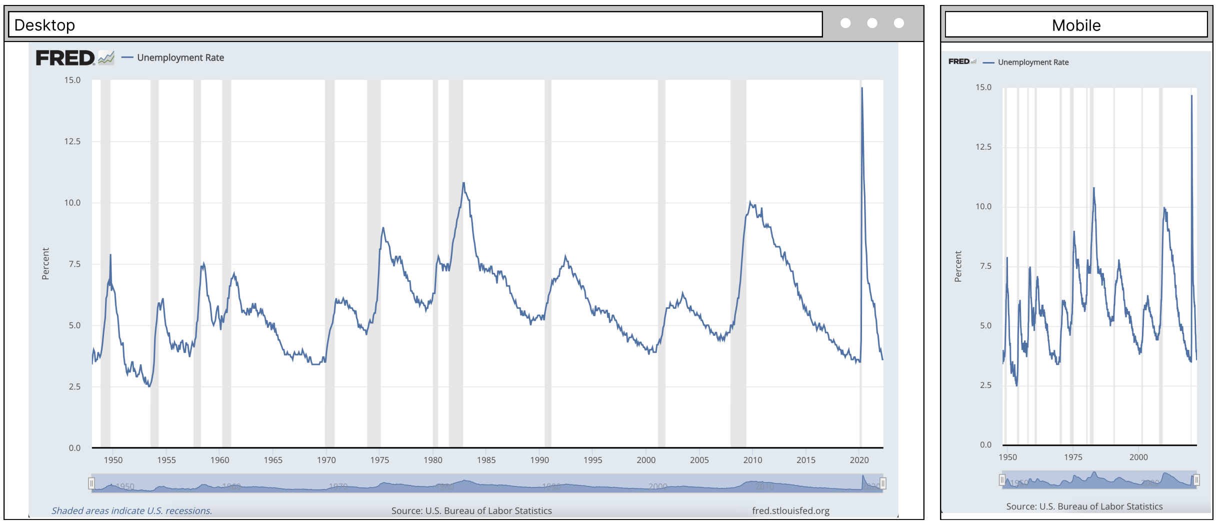

Figure 3.2: The desktop and mobile versions of the line chart. On the mobile version, the chart is scaled down by width, and there are fewer ticks displayed because of the lack of space (P6a). A user can use a mouse to select a time span in the blue brush below the chart and see the data for the selected range in higher granularity. On the mobile screen, this is possible by pinching the graph with two fingers. [Visualisation created by FRED [2022], screenshot taken by the authors.]

3.4.1 Voronoi Treemap

The first example is a Voronoi treemap visualisation of an information hierarchy. Nodes at each level in the tree are represented as polygons sized according to a metric, with polygon nesting indicating the hierarchical structure. This visualisation technique was first introduced by Andrews et al. [2002] and later named by Balzer and Deussen [2005]. It was presented as an improvement over existing rectangular treemap layouts, made possible by using an abstract polygonal data representation. In their taxonomy of treemap visualisation techniques, Scheibel et al. [2020] classify this technique as a “Space-filling Treemap”. Figure 3.4 shows an example of a Voronoi treemap generated with the tool VoroTree, developed by Christopher Oser as part of his Master’s Thesis [Oser 2022a]. It shows hierarchical information about the world’s gross domestic product. The visualisation automatically recalculates based on the available screen space, so any resizing of the browser window forces the entire graph to be redrawn, as illustrated on the mobile version. The main pattern used to make this visualisation responsive is automatic layouting,

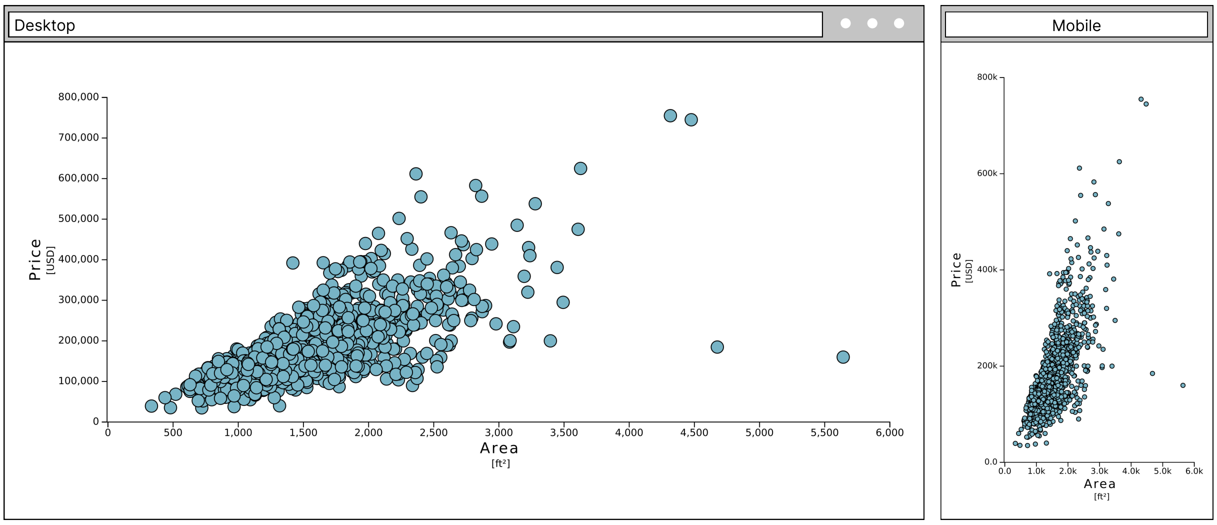

Figure 3.3: An example of a scatter plot on desktop (left) and mobile (right). On the mobile version, some tick marks are removed from the axes, and the labels are abbreviated. [Visualisation created by Oberrauner [2022a], screenshot taken by the authors.]

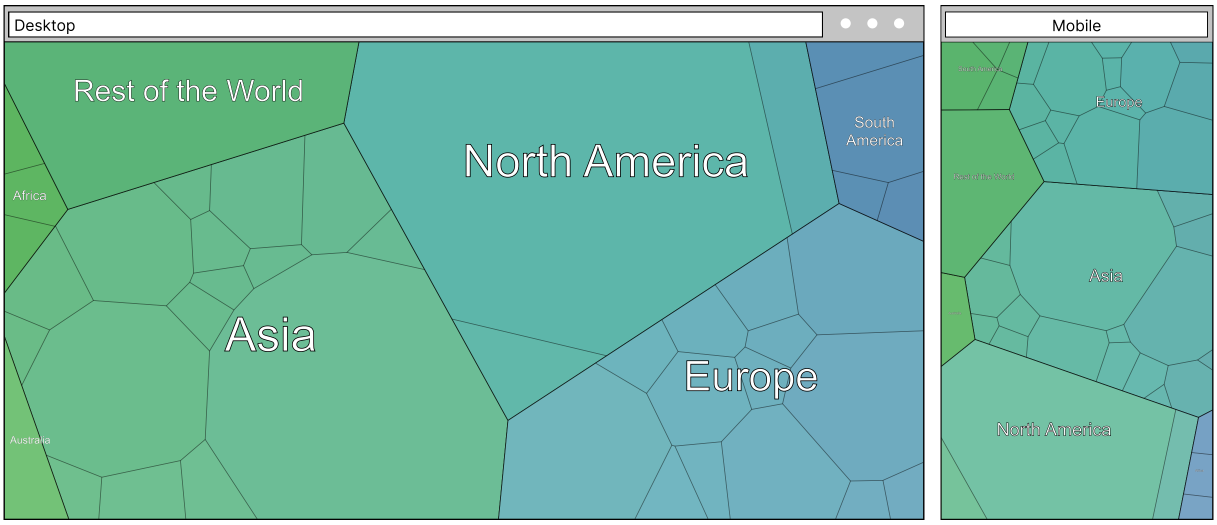

Figure 3.4: An example of a Voronoi treemap on desktop (left) and mobile (right). On both desktop and mobile devices, the visualisation automatically rescales based on the viewport dimensions. [Visualisation created by Oser [2022b], screenshot taken by the authors.]

as explained in pattern P5. Additionally, users can zoom in to obtain more detail - a technique covered in pattern P4.

3.4.2 Responsive Parallel Coordinates Plot

The second more exotic example is a responsive parallel coordinates chart, created and improved upon by a number of students under Keith Andrews' supervision and later polished by Aleš Smrdel [Andrews 2018a]. As shown in Figure 3.5, some dimensions are entirely removed from the data when the screen width is below a certain threshold. This displays a radical implementation of what we describe as a generalisation (P3). An additional button in the top left of the visualisation allows users to select the dimensions they want to be displayed on the chart manually. Additionally, the axes tick labels on the top are rotated based on the available width (P6b).

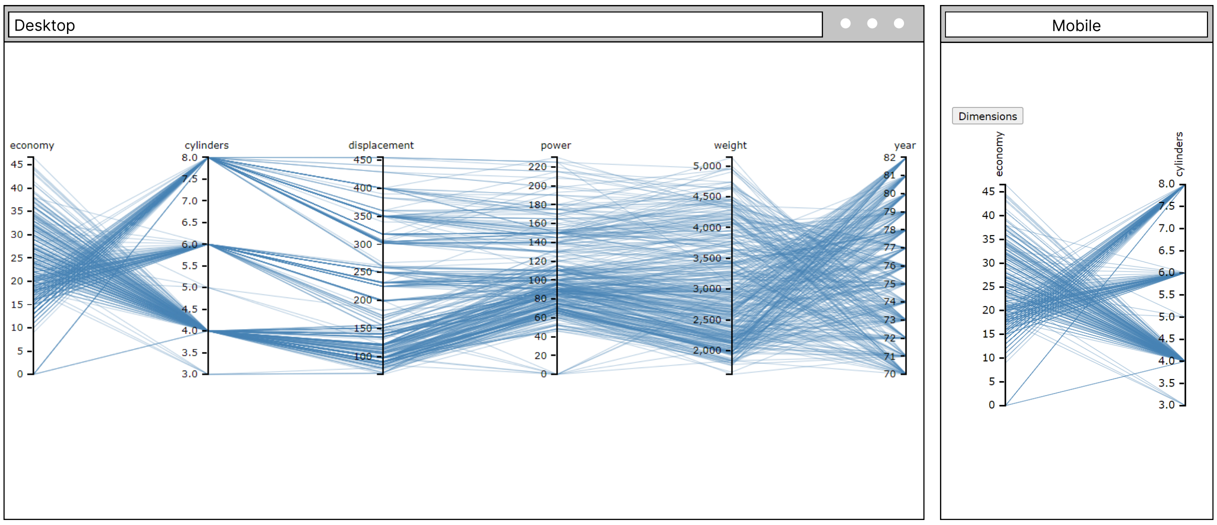

Figure 3.5: An example of a responsive parallel coordinates plot. Some dimensions are completely removed when going below a certain width threshold, displaying a radical approach of generalisation (P3). A button at the top of the mobile visualisations allows users to manually add back the dimensions. [Visualisation created by Andrews [2018a], screenshot taken by the authors.]

Chapter 4

Tools

Researchers have tried to develop tools to streamline adding responsiveness to existing charts. These tools mainly focus on bringing desktop visualisations to mobile devices but sometimes can be applied the other way around. As more of these tools are developed, the clear definition of design patterns, as outlined in Chapter 2 becomes even more important since they provide a starting point for implementation. The common denominator among all of the tools is that they generally utilise a specification, such as Vega-Lite, developed by the IDL [2022], to define the visualisation. Afterwards, a set of techniques is applied that transform the input to a desired (responsive) output specification.

4.1 Visualization Design Tool

In their paper, Hoffswell et al. [2020b] analyse 231 responsive news visualisations and come up with four design guidelines and a tool to help apply these guidelines. They then prove the functionality of their tool by introducing responsiveness to four example visualisations. Unfortunately, the tool is not available to the public and thus, could not be evaluated in the course of our research. The authors provide a short video in which they use their tool and an example page that lists their reworked examples [Hoffswell et al. 2020a]. Figure 4.1 shows the user interface of their tool in action.

4.2 MobileVisFixer

While the tool proposed by Hoffswell et al. [2020b] still relies upon the user to make adjustments, the tool MobileVisFixer by Wu et al. [2021] attempts to automate the entire process. This is implemented by scanning the chart elements and using a reinforcement learning framework, modelled as a Markov decision process, to add responsive elements to input charts. They describe a set of five design goals, namely:

- Simulate a trial-and-error process for manual repair
- Ensure the transparency and explainability of automation
- Remain as faithful as possible to the original visualisation
- Support compatibility with other algorithms
- Execute in browser rendering time

Figure 4.2 shows the transformation of two visualisations using the tool. Similar to the previously described tool, MobileVisFixer is currently not available to the public and thus, cannot be properly evaluated.

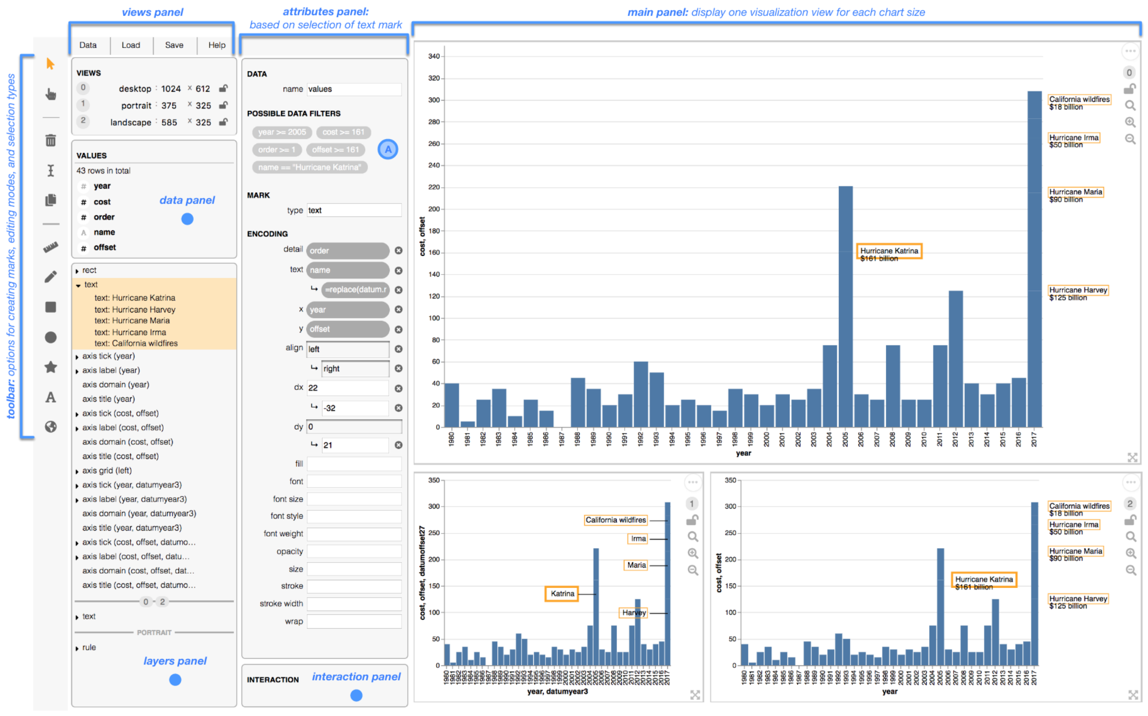
views panel
attributes panel: based on selection of text mark
main panel: display one visualization view for each chart size
toolbar: options for creating marks, editing modes, and selection types
data panel
layers panel
interaction panel
Data
Load
Save
Help
VIEWS
desktop 1024 × 612
portrait 375 × 325
landscape 585 × 325
VALUES
43 rows in total
year
cost
order
name
offset
DATA
name values
POSSIBLE DATA FILTERS
MARK
type text
ENCODING
INTERACTION
Hurricane Katrina $161 billion
Hurricane Harvey $125 billion
Hurricane Maria $90 billion
Hurricane Irma $50 billion
California wildfires $18 billion
year
cost, offset

Figure 4.1: A screenshot of the tool created by Hoffswell et al. [2020b], with added labels on the different panels that aid a designer in adding responsiveness to their visualisation. [Copyright © 2020 Association of Computing Machinery and used under § 42f.(1) of Austrian copyright law]

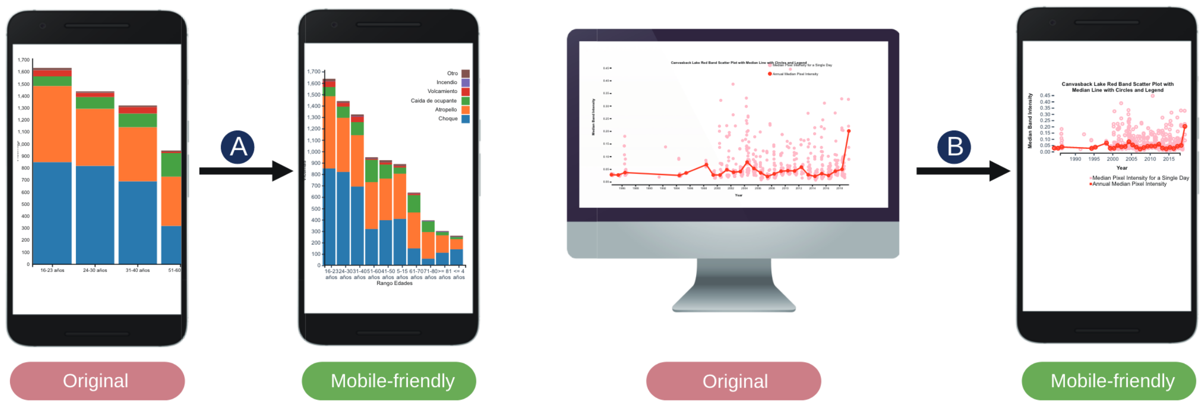

Figure 4.2: A schematic representation of MobileVisFixer, proposed by Wu et al. [2021], detailing the transformation from two desktop visualisations to mobile. [Copyright © 2021 IEEE and used under § 42f.(1) of Austrian copyright law]

4.3 Cicero Compiler

In their recent publication, Kim et al. [2022] propose an extension to the Vega-Lite grammar, allowing users to modify Vega-Lite nodes in a structured way. A typical Cicero node usually contains:

- Set of specifiers (target elements)
- Action
- Set of additional options

Typical use-cases for a tool like this would be queries such as: "For all tick marks, change the colour to red." The custom Cicero compiler then parses these additional specifications and generates a new Vega-Lite file in the process. Furthermore, the authors propose a recommender prototype system based on Answer Set Programming, suggesting multiple possible changes to a Vega-Lite visualisation using a heuristic cost-based approach. To identify important elements, they use a combination of established design patterns by Hoffswell et al. [2020b] and Kim et al. [2021] to create a grouping of suitable patterns for their system. Unfortunately, we could not evaluate this tool ourselves since it is not available to the public.

Chapter 5

Concluding Remarks

The age of mobile requires many novel solutions to embrace the portability of new devices. People want to carry their smartphone or tablet devices wherever they go, whilst retaining the ability to interact with complex data and rich visualisations. In a time where mobile device usage is still rising, it is more important than ever to think about easily applicable patterns and guidelines to help design visualisations with responsiveness in mind. Users should not be required to accept that some visualisations simply do not work well on mobile devices, and with the introduction of patterns and the development of new tools, this will hopefully soon be a thing of the past. Many successful companies have realised this shift towards mobile computing and properly emphasize the importance of designing with these devices in mind. As Google's chief executive Eric Schmidt said during an interview in 2010 [Ha 2010]:

> " What's really important right now is to get the mobile architecture right. Mobile will ultimately be the way you provision most of your services. The way I like to put it is, the answer should always be mobile first. You should always put your best team and your best app on your mobile app. "

Although we believe the idea of creating for mobile-first is great, the more generally applicable solution to many of these problems can be summarized as: **Responsive first**.

Bibliography

Adepu, Sushma and Rachel F. Adler [2016]. *A Comparison of Performance and Preference on Mobile Devices vs. Desktop Computers*. Proc. 7th Annual IEEE Ubiquitous Computing, Electronics, Mobile Communication Conference (UEMCON 2016). Oct 2016, pages 1–7. doi:10.1109/UEMCON.2016.7777808 (cited on pages 1–3).

Adler, Valentin, Ledio Jahaj, Markus Petritz, and Pooja Yeli [2021]. *Responsive Data Visualisation*. 706.057 Information Visualisation SS 2021 Survey Paper. Graz University of Technology, 10 May 2021. https://courses.isds.tugraz.at/ivis/surveys/ss2021/ivis-ss2021-g2-survey-resp-datavis.pdf (cited on page 5).

Andrews, Keith [2018a]. *Responsive Data Visualisation*. 2018. https://projects.isds.tugraz.at/respvis/ (cited on pages 1, 17–18).

Andrews, Keith [2018b]. *Responsive Visualisation*. CHI 2018 Workshop on Data Visualization on Mobile Devices (MobileVis 2018) (Montréal, Québec, Canada). 21 Apr 2018. https://mobilevis.github.io/assets/mobilevis2018_paper_4.pdf (cited on page 1).

Andrews, Keith, Wolfgang Kienreich, Vedran Sabol, Jutta Becker, Georg Droschl, Frank Kappe, Michael Granitzer, Peter Auer, and Klaus Tochtermann [2002]. *The InfoSky Visual Explorer: Exploiting Hierarchical Structure and Document Similarities*. Information Visualization 1.3/4 (Dec 2002), pages 166–181. doi:10.1057/palgrave.ivs.9500023 (cited on page 16).

Android Open Source Project [2022]. *Device Sensor Types*. 13 May 2022. https://source.android.com/devices/sensors/sensor-types (cited on page 3).

Balzer, Michael and Oliver Deussen [2005]. *Voronoi Treemaps*. Proc. IEEE Symposium on Information Visualization (InfoVis 2005) (Minneapolis, Minnesota, USA). IEEE Computer Society. Oct 2005, pages 49–56. doi:10.1109/INFOVIS.2005.40. http://graphics.uni-konstanz.de/publikationen/Balzer2005VoronoiTreemaps/Balzer2005VoronoiTreemaps.pdf (cited on page 16).

Bremer, Nadieh [2022a]. *Beautiful in English*. Visual Cinnamon, 13 May 2022. http://beautifulinenglish.com/ (cited on page 10).

Bremer, Nadieh [2022b]. *Why Do Cats and Dogs*. Visual Cinnamon, 13 May 2022. https://whydocatsanddogs.com/cats#chart-vs (cited on page 13).

De Jonge, Sarah and Nenagh Kemp [2012]. *Text-Message Abbreviations and Language Skills in High School and University Students*. Journal of Research in Reading 35.1 (Feb 2012), pages 49–68. ISSN 1467-9817. doi:10.1111/j.1467-9817.2010.01466.x (cited on page 11).

FRED [2022]. *Unemployment Rate*. Federal Reserve Bank of St. Louis, 25 May 2022. https://fred.stlouisfed.org/series/UNRATE (cited on pages 15–16).

Giurgiu, Luminita and Ilie Gligorea [2017]. *Responsive Web Design Techniques*. Proc. 23rd International Conference on Knowledge-Based Organization (KBO 2017) (Sibiu, Romania). Volume 23. 3. 15 Jun 2017, pages 37–42. doi:10.1515/kbo-2017-0153 (cited on page 3).

Google [2022]. *Material Design Gestures*. 13 May 2022. https://material.io/design/interaction/gestures.html#properties (cited on pages 2–3).

Ha, Anthony [2010]. *Eric Schmidt on Google's Mobile First Attitude, Weaknesses*. Venturebeat, 12 Apr 2010. https://venturebeat.com/2010/04/12/eric-schmidt-mobile-first/ (cited on page 23).

Hoffswell, Jane, Wilmot Li, and Zhicheng Liu [2020a]. *Example Gallery, Techniques for Flexible Responsive Visualization Design*. 2020. https://jhoffswell.github.io/website/resources/supplemental/responsive-supplemental/index.html (cited on page 19).

Hoffswell, Jane, Wilmot Li, and Zhicheng Liu [2020b]. *Techniques for Flexible Responsive Visualization Design*. Proc. 2020 Conference on Human Factors in Computing Systems (CHI 2020) (Honolulu, Hawaii, USA). ACM, 25 Apr 2020. ISBN 1450367089. doi:10.1145/3313831.3376777. https://jhoffswell.github.io/website/resources/papers/2020-ResponsiveVisualization-CHI.pdf (cited on pages 5, 9, 19–21).

Horak, Tom, Wolfgang Aigner, Matthew Brehmer, Alark Joshi, and Christian Tominski [2021]. *Responsive Visualization Design for Mobile Devices*. In: *Mobile Data Visualization*. Edited by Bongshin Lee, Raimund Dachselt, Petra Isenberg, and Eun Kyoung Choe. Chapman and Hall/CRC, 23 Dec 2021, pages 33–65. ISBN 0367534711. doi:10.1201/9781003090823-2. https://imld.de/cnt/uploads/Horak2021_MobileDataVisBook_Chap02_Responsive.pdf (cited on pages 1, 6).

IDL [2022]. *Vega-Lite Grammar*. University of Washington Interactive Data Lab, 12 May 2022. https://vega.github.io/vega-lite/ (cited on page 19).

Islam, Nayeem and Roy Want [2014]. *Smartphones: Past, Present, and Future*. IEEE Pervasive Computing 13.4 (Oct 2014), pages 89–92. ISSN 1558-2590. doi:10.1109/MPRV.2014.74 (cited on page 3).

Karlson, Amy, Brian Meyers, Andy Jacobs, Paul Johns, and Shaun Kane [2009]. *Working Overtime: Patterns of Smartphone and PC Usage in the Day of an Information Worker*. Proc. 7th International Conference on Pervasive Computing (Pervasive 2009) (Nara, Japan). Volume 5538. LNCS. Springer, 11 May 2009, pages 398–405. ISBN 3642015158. doi:10.1007/978-3-642-01516-8_27 (cited on page 2).

Kim, Hyeok, Dominik Moritz, and Jessica Hullman [2021]. *Design Patterns and Trade-Offs in Responsive Visualization for Communication*. Computer Graphics Forum 40.3 (29 Jun 2021), pages 459–470. ISSN 0167-7055. doi:10.1111/cgf.14321. https://mucollective.northwestern.edu/files/2021-responsive-vis.pdf (cited on pages 5, 10, 21).

Kim, Hyeok, Ryan Rossi, Fan Du, Eunyee Koh, Shunan Guo, Jessica Hullman, and Jane Hoffswell [2022]. *Cicero: A Declarative Grammar for Responsive Visualization*. Proc. 2022 Conference on Human Factors in Computing Systems (CHI 2022) (New Orleans, Louisiana, USA). ACM, 30 Apr 2022. doi:10.1145/3491102.3517455. https://arxiv.org/pdf/2203.08314.pdf (cited on pages 5, 9, 21).

Müller, Jonas, Lea Rieger, Ilhan Aslan, Christoph Anneser, Malte Sandstede, Felix Schwarzmeier, Björn Petrak, and Elisabeth André [2019]. *Mouse, Touch, or Fich: Comparing Traditional Input Modalities to a Novel Pre-touch Technique*. Proc. 18th International Conference on Mobile and Ubiquitous Multimedia (MUM 2019) (Pisa, Italy). ACM, 26 Nov 2019, pages 1–7. ISBN 145037624X. doi:10.1145/3365610.3365622. https://opus.bibliothek.uni-augsburg.de/opus4/frontdoor/deliver/index/docId/68414/file/2019_mum_fich.pdf (cited on page 4).

Oberrauner, Peter [2022a]. *RespVis - Point Chart*. 13 May 2022. https://respvis.netlify.app/examples/point.html (cited on page 17).

Oberrauner, Peter [2022b]. *RespVis - Stacked Bar Chart*. 13 May 2022. https://respvis.netlify.app/examples/stacked-bar.html (cited on page 16).

Oberrauner, Peter [2022c]. *RespVis: A Browser-Based, D3 Extension Library for Creating Responsive SVG Charts*. Master's Thesis. Graz University of Technology, Austria, 12 May 2022. 131 pages. https://ftp.isds.tugraz.at/pub/theses/poberrauner-2022-msc.pdf (cited on page 15).

Oser, Christopher [2022a]. *Responsive Voronoi Treemaps with VoroTree and VoroLib*. Master's Thesis. Graz University of Technology, Austria, 23 Feb 2022. 105 pages. https://ftp.isds.tugraz.at/pub/theses/coser-2022-msc.pdf (cited on page 16).

Oser, Christopher [2022b]. *VoroTree*. 13 May 2022. https://somestudentcoder.github.io/vorotree/ (cited on page 17).

Scheibel, Willy, Matthias Trapp, Daniel Limberger, and Jürgen Döllner [2020]. *A Taxonomy of Treemap Visualization Techniques:* Proc. 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (IVAPP 2020). Volume 3. Valletta, Malta: SCITEPRESS, 27 Feb 2020, pages 273–280. ISBN 9897584021. doi:10.5220/0009153902730280. https://publishup.uni-potsdam.de/opus4-ubp/frontdoor/deliver/index/docId/52469/file/pde008.pdf (cited on page 16).

Skupin, André, Joseph R. Biberstine, and Katy Börner [2013]. *Visualizing the Topical Structure of the Medical Sciences: A Self-Organizing Map Approach*. PLoS ONE 8.3 (12 Mar 2013). ISSN 1932-6203. doi:10.1371/journal.pone.0058779. https://cns.iu.edu/docs/publications/2013-skupin-pone.pdf (cited on pages 8–9).

Statista [2022]. *Share of Global Mobile Website Traffic 2015-2021*. 18 Feb 2022. https://statista.com/statistics/277125/share-of-website-traffic-coming-from-mobile-devices/ (cited on pages 1–2).

Tufte, Edward Rolf [1990]. *Envisioning Information*. Volume 126. Cheshire, CT: Graphics Press, 1990, page 67. ISBN 0961392118 (cited on page 8).

W3C [2022]. *Media Queries Level 3*. World Wide Web Consortium, 05 Apr 2022. https://w3.org/TR/mediaqueries-3/ (cited on page 6).

Wu, Aoyu, Wai Tong, Tim Dwyer, Bongshin Lee, Petra Isenberg, and Huamin Qu [2021]. *MobileVisFixer: Tailoring Web Visualizations for Mobile Phones Leveraging an Explainable Reinforcement Learning Framework*. IEEE Transactions on Visualization and Computer Graphics 27.2 (Feb 2021), pages 464–474. ISSN 1077-2626. doi:10.1109/TVCG.2020.3030423. https://microsoft.com/en-us/research/uploads/prod/2021/01/MobileVisFixer-InfoVis2020.pdf (cited on pages 1, 9, 19–20).

Xiao, Xiang, Teng Han, and Jingtao Wang [2013]. *LensGesture: Augmenting Mobile Interactions with Back-of-Device Finger Gestures*. Proc. 15th ACM on International Conference on Multimodal Interaction (ICMI '13). Sydney, Australia: ACM, Dec 2013, pages 287–294. ISBN 1450321291. doi:10.1145/2522848.2522850. https://people.cs.pitt.edu/~xiangxiao/lensgesture-icmi2013.pdf (cited on page 2).