Information Visualisation - Survey Group 2: Responsive Data Visualisation

Valentin Adler, Ledio Jahaj, Markus Petritz, and Pooja Yeli

706.057 Information Visualisation SS 2021
Graz University of Technology

10 May 2021

Abstract

Typical data visualisations are static in the sense that they usually have a fixed layout and size. Viewing such visualisations can be problematic on some devices, which is where responsive data visualisations come into play. These types of visualisations adapt to the device they are viewed on to make better use of the available screen area, so that the user can view it as if it was made for their device. In this survey, we first provide an introduction to responsive data visualisation, where we also define responsive visualisation patterns that describe responsive behaviour. This will be followed by a showcase of charts using some of these patterns. We then give an overview of current software libraries that support responsive behaviour to some extent. Finally, we look at a software editor for making responsive data visualisations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Data visualisations usually have fixed dimensions or a fixed layout, which might be fine if they are viewed on a regular computer, but the user experience might suffer if viewed on a mobile phone. A smaller screen size usually means that one has to either scale down the visualisation, so that it can be viewed entirely, or zoom in and only look at small parts of it. A better option would be if the visualisation adapts itself to the device's limitations, so that it can be viewed regardless of what device the user has. The aim of responsive data visualisations is to facilitate that.

1.1 Responsive Web Design

Responsive Web Design (RWD) is a web design principle, where one flexible page design is used for all devices. This means that instead of having various dedicated designs for different devices, the main design simply adapts itself to the screen size. Usually RWD makes use of breakpoints based on the viewport width (e.g. with CSS media queries), so that depending on the screen width different CSS rules might be active. This enables flexible components that can move around the page, scale up or down, or maybe even disappear entirely, so that the available screen space is used more efficiently. Additionally, there might be support for different forms of interaction, which are not available on all devices (e.g. touch interactions).

1.2 Responsive Data Visualisation

Responsive Data Visualisation (RDV) makes use of the same principles as RWD: flexible components that can be rearranged or rescaled to fit the available area. It is usually done with web-based solutions, as they are both cross-platform and already support typical responsive techniques.

However, as Andrews [2018] says, there is an important distinction between *scalable* and *responsive*. Responsive visualisations should not just scale, ideally they should go further:

- Layout: should adapt to the available display space.
- Display Density: should adapt to the available display space, for example by sampling and displaying fewer data points on narrower screens.
- Interaction: support different interaction modalities provided by devices, such as keyboard, touch, or motion.

1.3 Responsive Visualisation Patterns

In RDV, there are common patterns, which can be used either alone or in combination to facilitate responsive behaviour.

1.3.1 RP1: Rotating Axis Tick Labels

Axis tick labels can be rotated to better fit the available space around them. The labels could start out in a horizontal position, then rotate so that they are tilted at 45 degrees, and finally end up in a vertical position.

1.3.2 RP2: Removing Axis Ticks

Axis ticks could be thinned out or removed entirely, as sometimes there simply is not enough space to display all markings. The ticks should be removed in such a way, so that the scale of the axis is still made clear to the user.

1.3.3 RP3: Shortening Strings

Some strings could be shortened if there is not enough space. Good candidates for these would be axis tick labels, the chart title, or strings in the legend.

1.3.4 RP4: Flipping 90 Degrees

If the chart takes up too much space horizontally, then rotating it by 90 degrees could improve the user experience. Especially on phones it is normal behavior to just scroll down, while horizontal scrolling is unusual. The chart does no have to flip entirely, it can be sufficient to simply swap the axes.

1.3.5 RP5: Reposition Chart Components

Some chart components could be repositioned to different areas. For example, a chart's legend might be to the right of the chart on wider screens, but move beneath it on narrower screens, freeing up more width for the actual chart.

1.3.6 RP6: Interactive Zoom

Interactive zoom might not be *responsive* by itself, but can still help greatly with the user experience. Maps often have this feature, so that the user can zoom in to get a more detailed view of a specific area. The same idea could be used for e.g. scatter plots in order to have a more detailed look at a smaller part of the rendered data.

1.3.7 RP7: Interactive Filtering

Interactive filtering might also not be *responsive* by itself. Enabling the user to filter out data or maybe deselect entire datasets, can influence the data visualisation, as having less data to show means there could be more available space for the remaining data.

1.3.8 RP8: Adjust Data Density

On smaller screens there might not be enough space to properly render all data points. In such a situation, it could be helpful to thin out (sample) or group data points into clusters.

1.3.9 RP9: Changing the Chart Type

Changing the chart type to some degree (or maybe entirely) can also be useful in some circumstances. A scatter plot with a large number of data points on a small screen might be hard to analyse if the majority of the data points are closely together. If a heat map were used instead, then the colour of a tile could indicate how many points there are within that area. Another example is a line chart which turns into a sparkline on narrower displays.

Chapter 2

Examples

In this chapter a closer look is taken at some examples of the patterns that were introduced in earlier chapter.

2.1 Responsive Line Chart

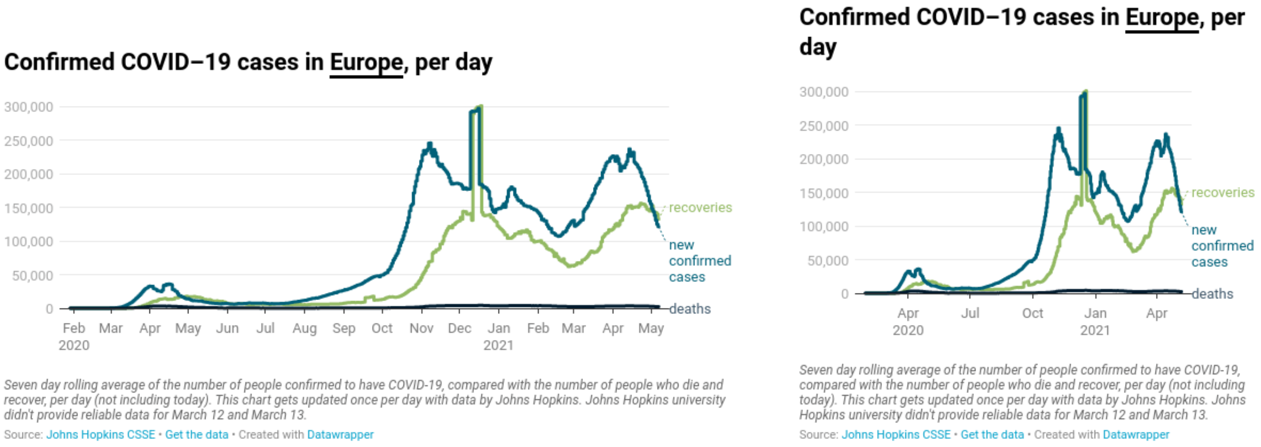
Figure 2.1 depicts pattern “RP2: Removing Axis Ticks”. Figure 2.1a shows the wide window, where the x axis has the full range of months as ticks. In Figure 2.1b it can be seen how the chart adapts itself to a narrower window. Some of the x-axis ticks get removed, while still making the scale clear to the viewer. A video showcasing this behavior can be seen at `https://youtu.be/lkBUAQbnAvU`.

2.2 Responsive Bar Chart

Figure 2.2 depicts pattern “RP4: Flipping 90 Degrees”, “RP5: Reposition Chart Components” and “RP7: Interactive Filtering”. The wide window shows a detailed bar chart, as shown in Figure 2.2a. The chart is made responsive to adapt for narrow window, as shown in Figure 2.2b. In the narrow window, the chart is incorporated with many changes like rotating the chart by 90 degrees and both the axes and the legend are repositioned. An user can also interact with the chart by deselecting the year to remove bars or to add the bars back to the chart as shown in Figure 2.2c. The video at `https://youtu.be/Tbo5sJUR0cY` shows this behavior.

2.3 Responsive Stacked Area Chart

Figure 2.1 depicts pattern “RP2: Removing Axis Ticks”. Figure 2.3a has the full range of months as axis ticks for the x-axis. Whereas in Figure 2.3b some of the x-axis labels with ticks are removed to adapt for narrower windows. This behavior is shown in a video at `https://youtu.be/ci3_2Xyu0Aw`.



Confirmed COVID–19 cases in Europe, per day

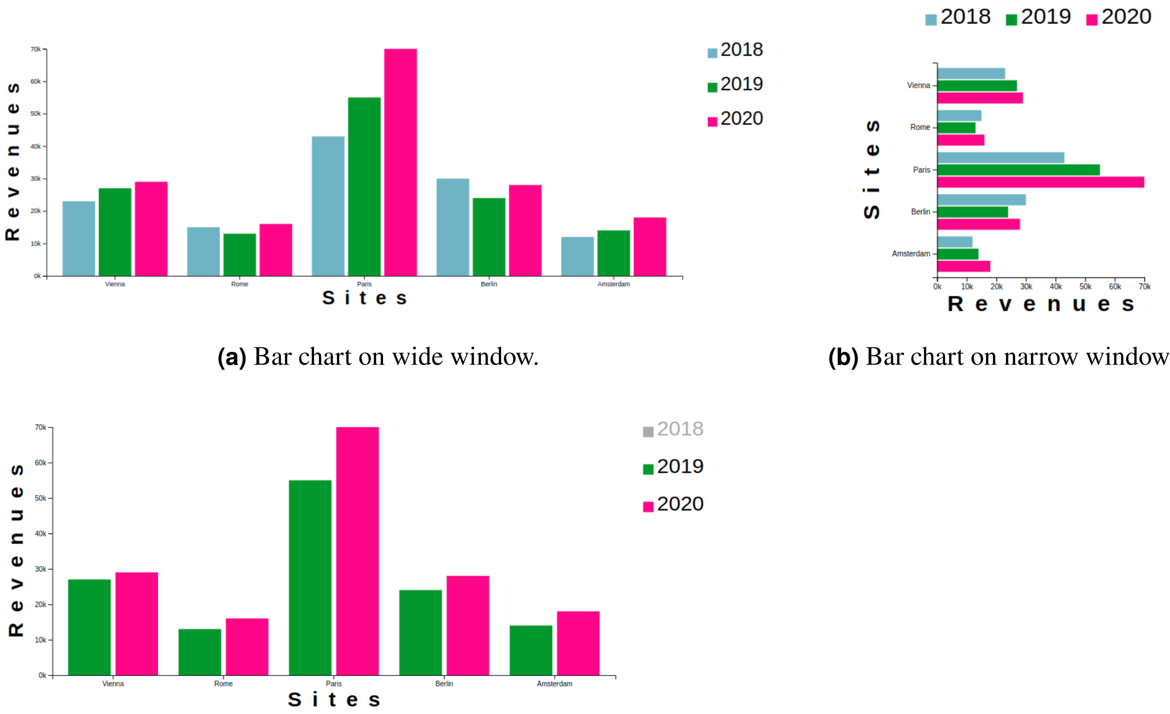
Seven day rolling average of the number of people confirmed to have COVID-19, compared with the number of people who die and recover, per day (not including today). This chart gets updated once per day with data by Johns Hopkins. Johns Hopkins university didn't provide reliable data for March 12 and March 13.

Source: Johns Hopkins CSSE • Get the data • Created with Datawrapper

(a) Line chart on wide window.

(b) Line chart on narrow window.

Figure 2.1: An example of a responsive line chart, adapted from Rost [2020a] and Rost [2020b].

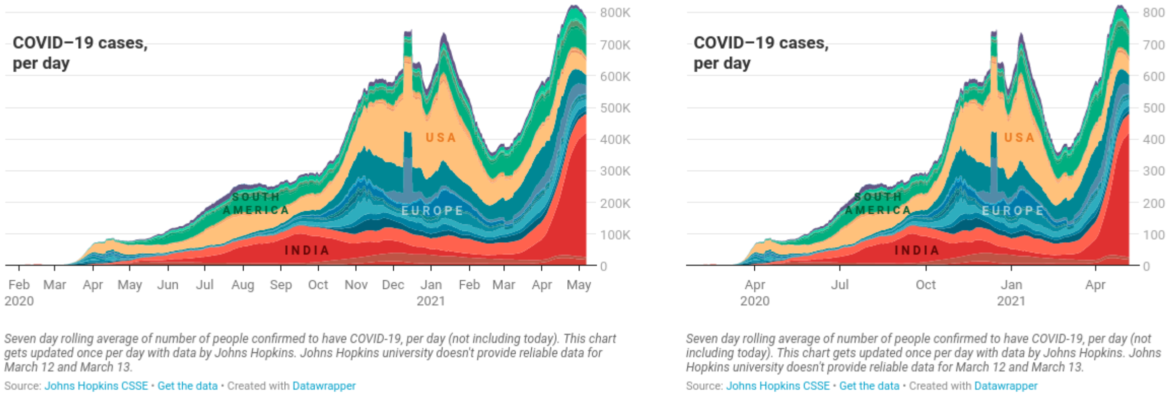


(a) Bar chart on wide window.

(b) Bar chart on narrow window.

(c) Bar chart depicting interactive filtering.

Figure 2.2: An example of responsive bar chart, adapted from Oberrauner [2021c].



Seven day rolling average of number of people confirmed to have COVID-19, per day (not including today). This chart gets updated once per day with data by Johns Hopkins. Johns Hopkins university doesn't provide reliable data for March 12 and March 13.

Source: Johns Hopkins CSSE • Get the data • Created with Datawrapper

Seven day rolling average of number of people confirmed to have COVID-19, per day (not including today). This chart gets updated once per day with data by Johns Hopkins. Johns Hopkins university doesn't provide reliable data for March 12 and March 13.

Source: Johns Hopkins CSSE • Get the data • Created with Datawrapper

(a) Stacked area chart on wide window.

(b) Stacked area chart on narrow window.

Figure 2.3: An example of responsive stacked area chart, adapted from Rost [2020a] and Rost [2020c].

2.4 Responsive Scatter Plot

Figure 2.4 is an example for patterns "RP2: Removing Axis Ticks" and "RP6: Interactive Zoom". Figure 2.4a depicts a detailed view of the scatter plot in a wide window. The scatter plot is adapted to narrow screens by removing some x-axis ticks as shown in Figure 2.4b. The user can interact with the scatter plot by selecting an area to zoom into as shown in Figure 2.4c. This behavior is showcased in a video at `https://youtu.be/oxK8a6ohtYI`.

2.5 Responsive Radial Scatter Plot

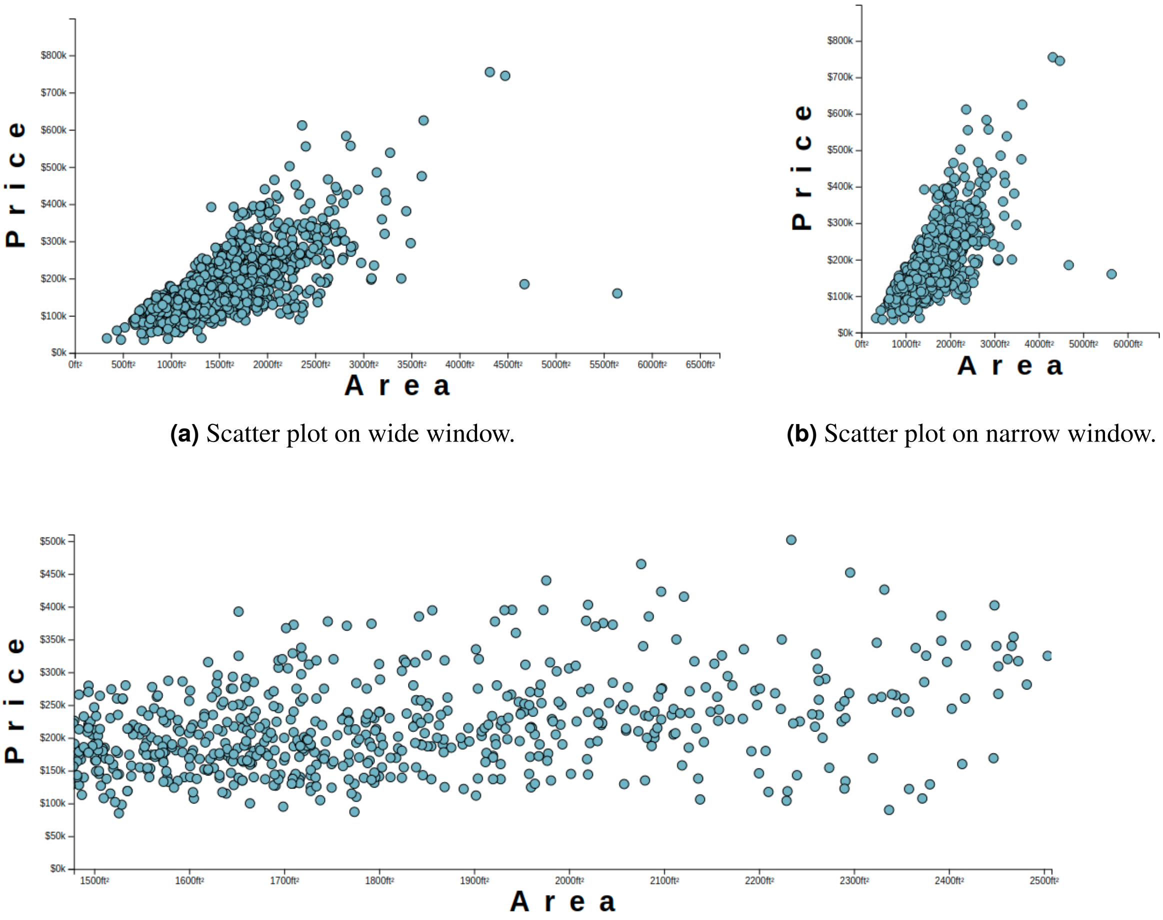
Figure 2.5 shows an example of "RP2: Removing Axis Ticks" and "RP5: Reposition Chart Components". A detailed view of the chart in a wide window is shown in Figure 2.5a. To adapt the chart for narrow screens the chart incorporates changes like repositioning of legend and removing some axis labels with ticks as shown in Figure 2.5b. A video showing this responsive behavior can be seen at `https://youtu.be/VTHyg6NGSCQ`.

2.6 Responsive Choropleth Map

Figure 2.6 shows an example of "RP6: Interactive Zoom". Figure 2.6a depicts an overview of the map. A user can zoom-in a part of map to get a detailed view of the area as shown in the Figure 2.6b. The video at `https://youtu.be/-DVtkiACZno` shows this behavior.

2.7 Responsive Node-Link Chart

Figure 2.7 depicts an example for patterns "RP4: Flipping 90 Degrees" and "RP5: Reposition Chart Components". Figure 2.7a shows a detailed view of the chart with images followed by names of the people and also shows links that depicts who follows whom on Twitter. As the chart is complex, to adapt this chart for narrow screens, it is rotated by 90 degrees, only names of the people are displayed, the chart components are repositioned and follows/followed by is displayed by clicking the respective person's name, as shown in Figure 2.7b. Shown in the video at `https://youtu.be/IYaXWP8CD54`.

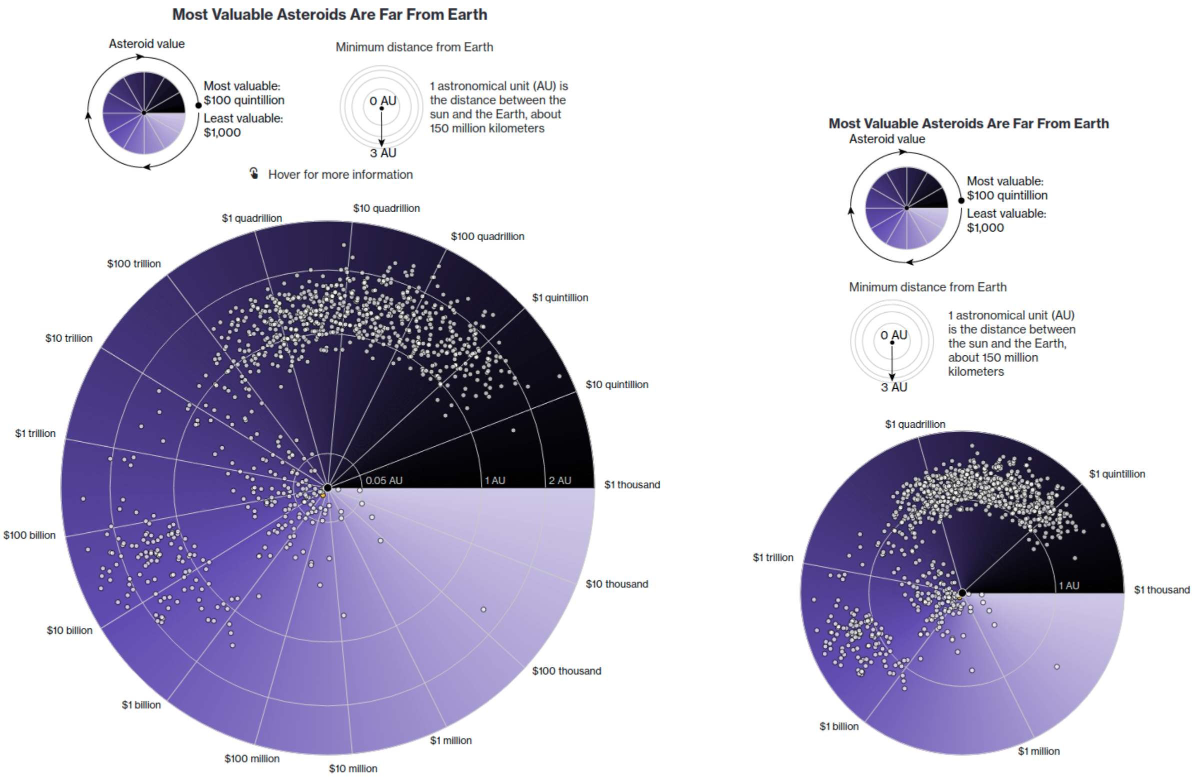


(a) Scatter plot on wide window.

(b) Scatter plot on narrow window.

(c) Scatter plot depicting interactive zoom.

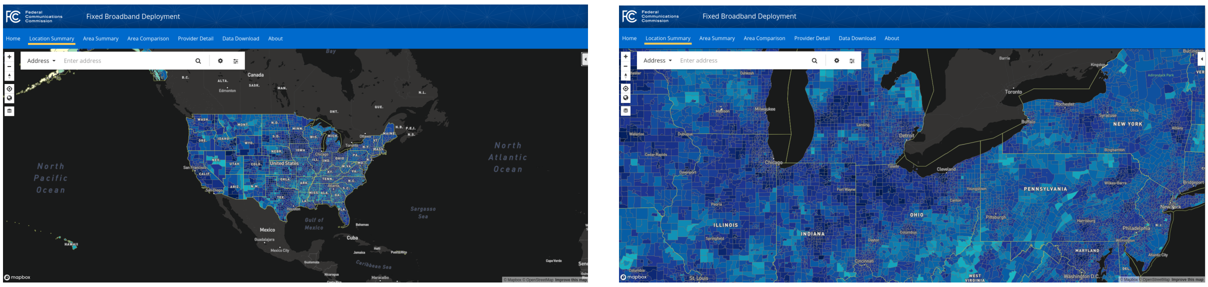
Figure 2.4: An example of responsive scatter plot, adapted from Oberrauner [2021b].



(a) Radial scatter plot on wide window.

(b) Radial scatter plot on narrow window.

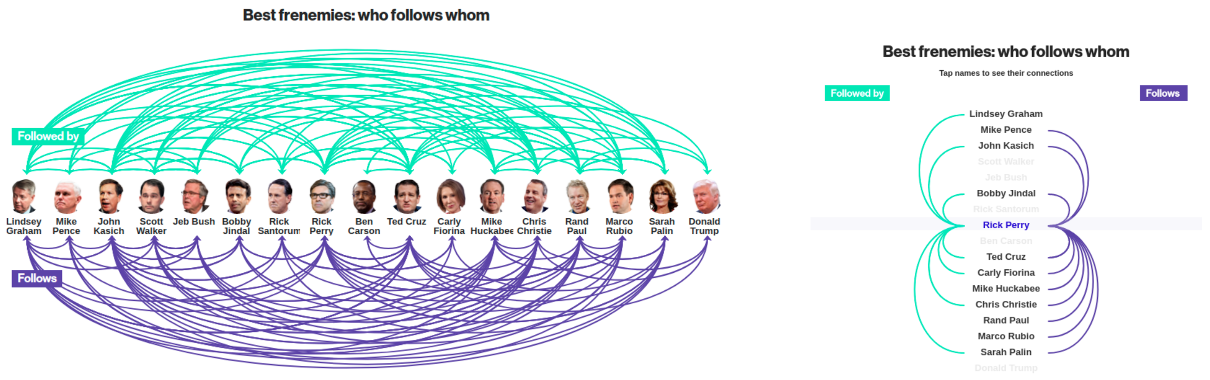
Figure 2.5: An example of responsive radial scatter plot, adapted from Barton and Recht [2018].



(a) Choropleth map on wide window.

(b) Choropleth map showing interactive zoom on wide window.

Figure 2.6: An example of responsive choropleth map, adapted from FCC [2021].



(a) Node-Link chart on wide window.

(b) Node-Link chart on narrow window.

Figure 2.7: An example of node-link chart, adapted from Yaccino et al. [2015].

Chapter 3

Software Libraries

Seven software libraries which claim to produce responsive data visualization are described and reviewed. The results show the lack of responsiveness as discussed throughout this survey and may motivate library makers to include our patterns.

In order to meaningfully compare them, the same simple bar chart was recreated with each library. The bar chart has 10 bars, each of the same colour, representing the price of gold for the last 10 years in January, extracted from IndexMundi [2021]. The data is shown in Table 3.1.

Tables 3.2 thtough 3.8 give a condensed overview of the seven software libraries we looked at.

Jan 2021	1866.98
Jan 2020	1560.67
Jan 2019	1291.75
Jan 2018	1331.30
Jan 2017	1192.10
Jan 2016	1097.91
Jan 2015	1250.75
Jan 2014	1244.27
Jan 2013	1671.85
Jan 2012	1654.05
Jan 2011	1360.46

Table 3.1: The gold price in US dollars per Troy Ounce in January from 2011 until 2021 from IndexMundi [2021].

Name	RespVis
Author	Peter Oberrauner
Description	Rendering SVG using D3 and FaberJS
Language	TypeScript
License	MIT
Website	respvis.netlify.app

Table 3.2: Information Sheet on the RespVis Framework.

Name	rdv
Author	Nick Rabinowitz
Description	Extending D3 to provide new responsive features
Language	JavaScript
License	MIT
Website	github.com/nrabinowitz/rdv

Table 3.3: Information Sheet on the rdv Library.

Name	Chart.js
Author	Multiple
Description	Rendering charts in <canvas>
Language	JavaScript
License	MIT
Website	github.com/chartjs/Chart.js

Table 3.4: Information Sheet on the Chart.js Library.

Name	HighCharts
Author	Highsoft
Description	SVG-based charting library
Language	JavaScript
License	Paid, Creative Commons Attribution-Non-Commercial
Website	highcharts.com

Table 3.5: Information Sheet on the HighCharts Library.

Name	ApexCharts
Author	Ideare Inc
Description	JS charting library; renders in SVG integrations for popular frameworks like Angular, Vue, React, ...
Language	JavaScript
License	MIT
Website	github.com/apexcharts/apexcharts.js

Table 3.6: Information Sheet on the ApexCharts Library.

Name	Pancake
Author	Rich Harris
Description	Charting library for Svelte rendered with HTML and CSS
Language	JavaScript, Svelte
License	MIT
Website	github.com/Rich-Harris/pancake

Table 3.7: Information Sheet on the Pancake Library.

Name	RGraph
Author	Richard Heyes
Description	JS charting library, supports canvas and SVG rendering
Language	JavaScript
License	MIT
Website	github.com/heyesr/rgraph

Table 3.8: Information Sheet on the RGraph Library.

3.1 RespVis

RespVis [Oberrauner 2021a] is a project from Graz University of Technology Master's student Peter Oberrauner, who is supervised by Keith Andrews. It resulted from the lack of responsive patterns available in other libraries and therefore has implemented the most patterns of all our reviewed libraries. Currently, it is still in its infancy, but soon it will support more chart types. The render time is a bit slow, but the responsiveness is great and works very well. To try it out you have to compile it yourself as there is yet no CDN available.

It is mainly written in TypeScript, which gives it robustness and maintainability. As we skipped through the code we found traces of D3 [Bostock 2021] libraries, with many of the imports commented out, so we can see ambitions to completely remove them. Otherwise the responsiveness was written from scratch. For the layouting we could see that FaberJS [FusionCharts 2021] was used to mimic CSS grids. What we found very positive was the usage of SVG in the frontend, as it allows for a better user experience as it just works in the browser, adapts to variable view sizes and pixel densities as well as scaling without pixelation due to the vector nature.

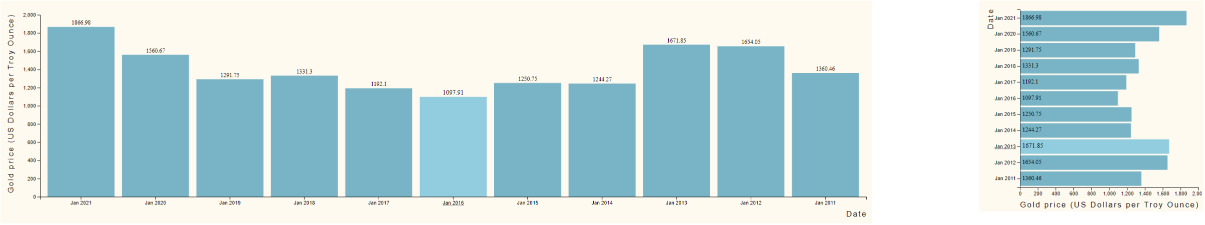
As time of writing, the following chart types are supported by the library and examples provided:

- Bar Chart
- Grouped Bar Chart
- Stacked Bar Chart
- Scatterplot Chart
- Brush Zoom

All of the given examples utilize one or more responsive patterns, so this library is highly appreciated to create meaningful responsive data visualizations.

We implemented a simple bar chart using RespVis library as seen in Figure 3.1. The following responsive patterns can be observed:

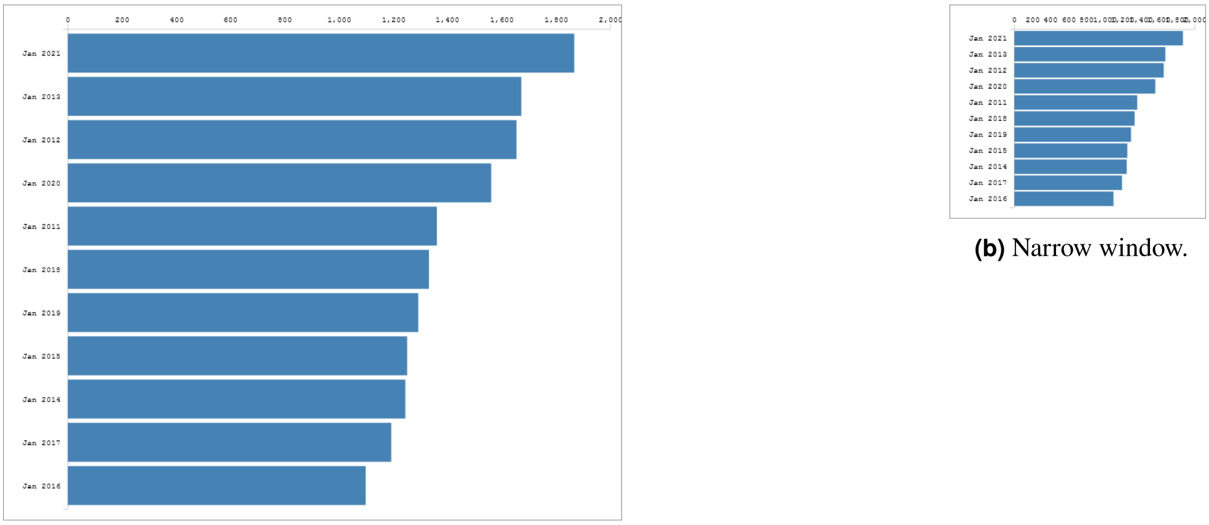
- RP2: Removing Axis Ticks
- RP3: Flipping 90 Degrees



(a) Wide window.

(b) Narrow window.

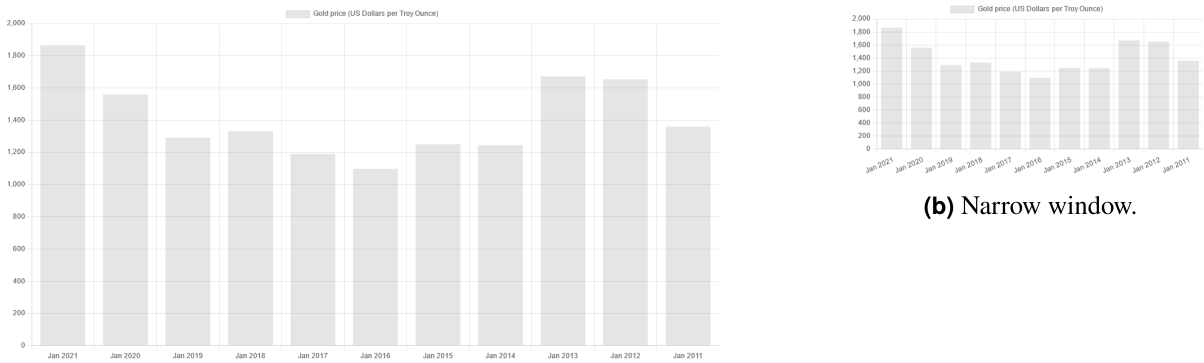
Figure 3.1: The example bar chart implemented with the RespVis library.



(b) Narrow window.

(a) Wide window.

Figure 3.2: The example bar chart implemented with the rdv library.



(b) Narrow window.

(a) Wide window.

Figure 3.3: The example bar chart implemented with the Chart.js library.

3.2 rdv

This small library was included because it had implemented a very creative way to adapt a scatterplot for smaller devices by converting it to a heat map. rdv was written by Nick Rabinowitz [Rabinowitz 2021] a few years ago using D3 [Bostock 2021] in plain JavaScript. It is very simple but very effective in its presentation.

Due to the old date and the heavy usage on D3, any further development on this library seems to be not worthwhile. However this idea of turning a scatterplot into a heatmap could be implemented in a more sophisticated library such as RespVis as mentioned in Section 3.1.

We could find only two chart types being supported:

- Bar Chart
- Scatterplot Chart

Each of the given example utilized one of our patterns, but as described before they seem more niche than being generally usable. Figure 3.2 shows our implementation.

3.3 Chart.js

Of all our reviewed libraries, Chart.js [Chart.js 2021] is probably the simplest one when considering the code. The rendering is done using the HTML canvas element and therefore JavaScript is required to display anything. It is our least favorite library when it comes to responsiveness since it only supports very basic patterns such as thinning out the axis ticks. We wont list each chart type as it has a wide range of supported types, examples of which can be viewed on the official website.

Figure 3.3 shows our implementation of a bar chart with Chart.js. The following patterns can be observed:

- RP1: Rotating Axis Tick Labels
- RP2: Removing Axis Ticks

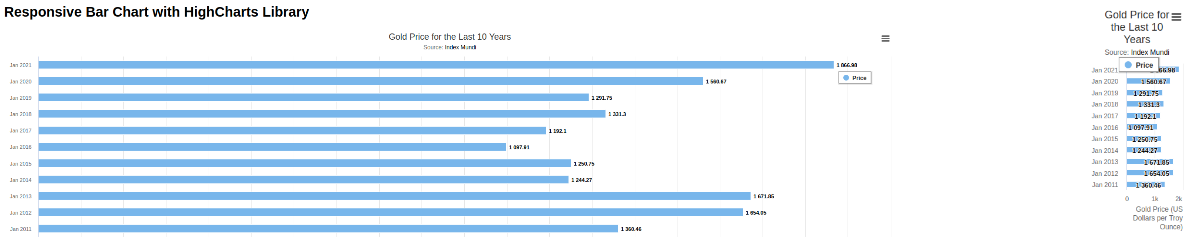
3.4 HighCharts

Highcharts is a software library for charting [*HighCharts* 2021] . It is written in pure JavaScript and based on SVG charting. It was first released in 2009. It was created by Highsoft in Vik, Norway by Torstein Honsi. With Highcharts one can create a lot of different types of responsive charts such as Bar and Column Charts, Area charts, scatter and bubble charts, Pie charts and many more. It is even possible to have combinations of chart types. We implemented a simple bar chart using Highcharts library, which can be seen in Figure 3.4. It is fairly responsive and has the following patterns:

- RP2: Removing Axis Ticks
- RP3: Shortening Strings

3.5 ApexCharts

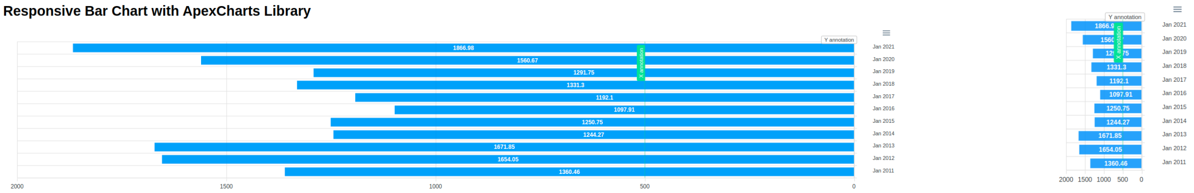
ApexCharts is a modern JavaScript charting library [ApexCharts 2021]. It can be used to built a wide range of responsive charts. Its API is very simple to use. This library also include wrappers so it can be integrated with some frameworks like Vue.js, Angular and React. We implemented a simple "Reversed Bar Chart" using ApexCharts, as seen in Figure 3.5, and we can say that does a very good job in responsiveness. This chart includes RP2: Removing Axis Ticks pattern.



(a) Wide window.

(b) Narrow window.

Figure 3.4: The example bar chart implemented with the HighCharts library.



(a) Wide window.

(b) Narrow window.

Figure 3.5: The example bar chart implemented with the ApexCharts library.

3.6 Pancake

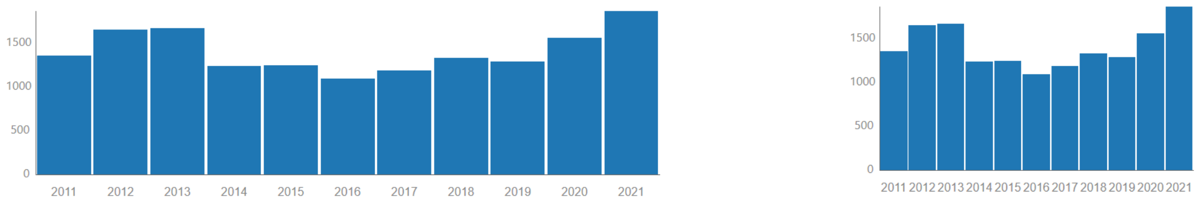
Pancake was created by Rich Harris [Harris 2021], who is a Graphics Editor working for The New York Times. He was upset with the chart library landscape because they all require JavaScript to work. For his new article he wanted to see if it works without it and he succeeded. In a blog article [Harris 2020], he describes his thought process as well as the technical specification underlying it. To sum it up, instead of relying on SVG with the obvious responsiveness bottlenecks, he build the chart using HTML and CSS for the percentage-based system to glue a scalable HTML layer and the SVG graph layer together. To further simplify the process he wrote a Svelte [Svelte 2021] component - a framework also by him - that not only helps with the creation of such graphs but also adds interactivity, if the user decides to load JS or their internet connection allows it.

Svelte is a new reactive framework that compiles to very minimal JavaScript files. Due to this compiling nature only the items that have to be reactive are in fact reactive. This saves valuable client-side computing time as no Virtual DOM has to be built. To connect the dots with Pancake, one can compile a Svelte component into HTML and JS, serve the HTML along with the page and instantly allow the browser to render the graph while in the meantime the JS, depending on the internet speed, is slowly loading and then hydrating the existing HTML with interactive components and building the full user experience.

The library is still experimental and mostly used by himself, but he still managed to built some great examples:

- Line Chart
- Stacked Bar Chart
- Treemap

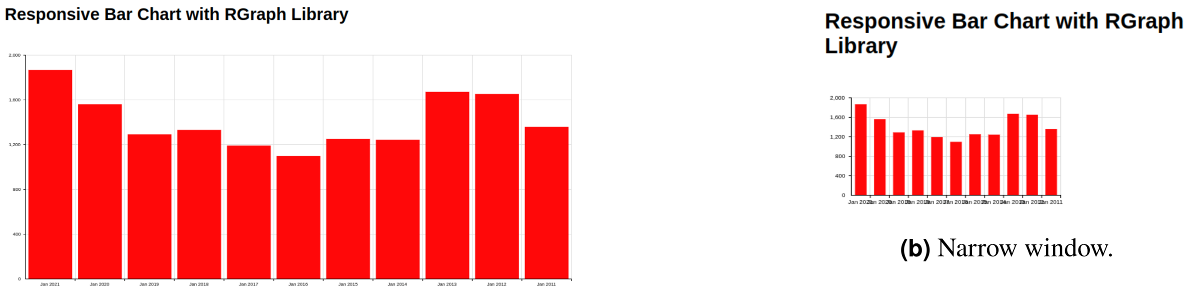
Figure 3.6 shows our implementation of a bar chart. Unfortunately it only scales with the container width.



(a) Wide window.

(b) Narrow window.

Figure 3.6: The example bar chart implemented with the Pancake library.



(a) Wide window.

(b) Narrow window.

Figure 3.7: The example bar chart implemented with the RGraph library.

3.7 RGraph

RGraph is an open-source JavaScript charts library for websites [Heyes 2021]. It initially supported only Canvas drawing of the charts. Now it supports also SVG and has the ability to create responsive charts also. The library has the capability to import many data sources like CSV files, Google Sheets and AJAX. This sources can be used as data sets for the charts instead of writing the data manually in the JavaScript file. This library can be used to create 2D and 3D charts.

We implemented a simple bar chart example using RGraph as seen in Figure 3.7. The library is good for creating charts but not so good in making them responsive. Our example is a semi-responsive charts as it scales when the window size becomes small but it does not have the patterns that we want. The bar chart is also adjustable. Meaning that with mouse clicks, the user can adjust the bars but even this does not comply to our patterns. Although RGraph has a "responsive" function built in, it has a lot of room for improvements.

Chapter 4

Software Editors

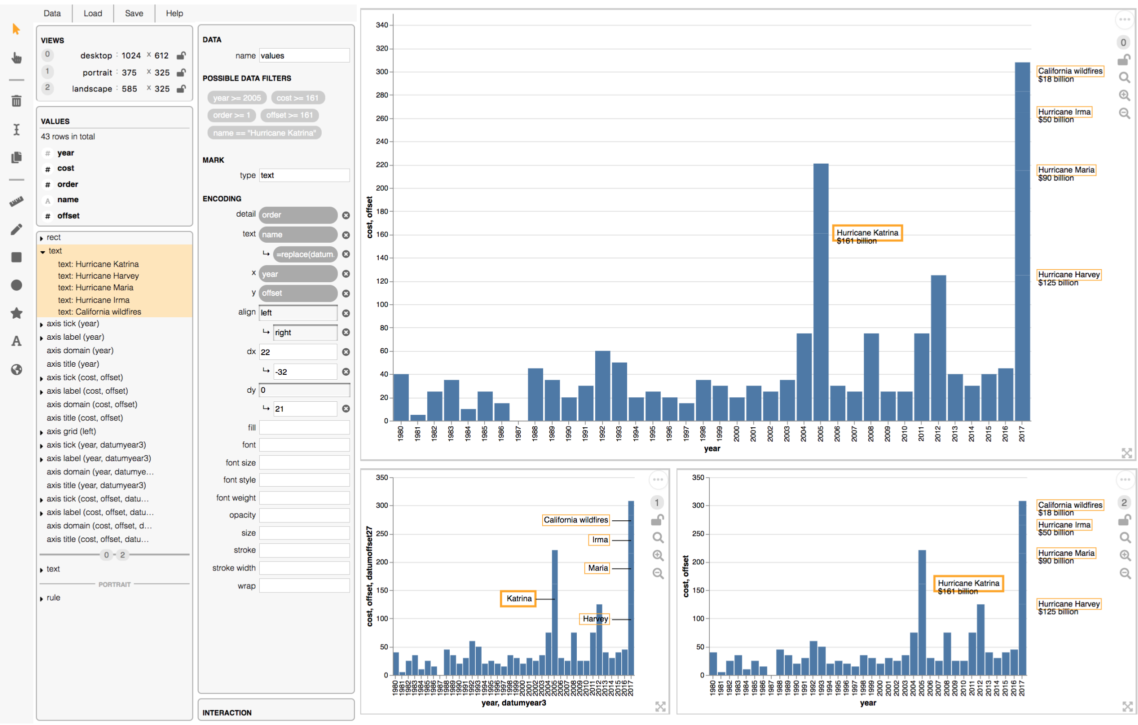
In this section of our survey, we are looking for software editors which can be used to create responsive visualizations. The editor should have a well constructed GUI (Graphical User Interface) that is user friendly. This tool should be able to visually create responsive graphs that work on different platforms and web browsers. Since the field of editors for responsive visualizations is a new area so far, we were able to find only one such editor.

4.1 Hoffswell's Editor

Jane Hoffswell, Wilmot Li, and Zhicheng Liu [Hoffswell et al. 2020] created a system that can be used as a GUI to built responsive visualizations for different devices like desktop and mobile. It is built on top of VEGA and VEGA-Lite [Vega 2021]. Their paper won the Best Paper Award at CHI 2020.

In Figure 4.1 we see an image of the UI of the Hoffswell's editor. On the left side, we have the marks toolbar, where a user can choose a mark for their charts. To the right, we can see the Views panel where the user can choose the number of views he wants to create charts for. In the current example there are three views for desktop, mobile portrait and mobile landscape. Below, we can see the Values panel. Here we upload our database and choose the attributes to be displayed in the axis we choose to. We can edit the axis to show whichever information we want. We can create the text for the titles, label and shape the bars etc. On the right panel, the user can edit the data of the chart, filter the data, edit the text and charts size, font, colors and choose the encoding. Lastly, we have the main panel where the visualizations are shown for each view. Every change made is reflected in real time in this panel. We can edit the views also by clicking the labels and text in the graph visualizations. If a user wants to edit just one of the views, he can click on the lock icon located in the right hand-side of each graph. It locks that view and no changes affects it. We could not get the opportunity to try this system ourselves, but there is a video from Jane Hoffswell making a bar chart with this system [Hoffswell 2020].

This system gives the opportunity to create a visual representation of some data in different sizes and shapes for different platforms, but it does not make one of these charts responsive in it's own platform. For example if one resizes the window of the browser, the chart will not be responsive to this resizing. In their research, they tried to tackle the lack of an UI program to give the user the opportunity to create cross-platform charts simultaneously and see the progress in real time.



VIEWS

0	desktop	1024	×	612
1	portrait	375	×	325
2	landscape	585	×	325

VALUES

43 rows in total

- # year
- # cost
- # order
- A name
- # offset

- rect
- text
  - text: Hurricane Katrina
  - text: Hurricane Harvey
  - text: Hurricane Maria
  - text: Hurricane Irma
  - text: California wildfires
- axis tick (year)
- axis label (year)
- axis domain (year)
- axis title (year)
- axis tick (cost, offset)
- axis label (cost, offset)
- axis domain (cost, offset)
- axis title (cost, offset)
- axis grid (left)
- axis tick (year, datumyear3)
- axis label (year, datumyear3)
- axis domain (year, datumye…
- axis title (year, datumyear3)
- axis tick (cost, offset, datu…
- axis label (cost, offset, datu…
- axis domain (cost, offset, d…
- axis title (cost, offset, datu…

0 - 2

- text

PORTRAIT

- rule

DATA

name: values

POSSIBLE DATA FILTERS

- year >= 2005
- cost >= 161
- order >= 1
- offset >= 161
- name == "Hurricane Katrina"

MARK

type: text

ENCODING

detail	order
text	name
↳	=replace(datum.
x	year
y	offset
align	left
↳	right
dx	22
↳	-32
dy	0
↳	21
fill	
font	
font size	
font style	
font weight	
opacity	
size	
stroke	
stroke width	
wrap	

INTERACTION

Figure 4.1: Hoffswell et al's editor system for responsive, cross-platform data visualization. [Copyright © 2020 Association of Computing Machinery.]

Chapter 5

Concluding Remarks

In this survey we have given a short introduction and overview to responsive data visualisations, while providing a list of responsive visualisation patterns, which describe and define responsive behavior. This list is not meant as a complete definition, as there are surely more responsive patterns that we have not defined. Adding more to this list could potentially help with actually implementing such features in existing visualisations.

The examples of responsive data visualisations only included some of the patterns we defined in the preceding chapter. This could imply that some of the patterns are rarely used in existing charts, or not yet supported by commonly used tools.

The software libraries we listed have varying levels of support for responsive visualisation patterns. Some of these libraries are in an experimental state and very limited when it comes to the amount of supported features. Other libraries might have a high range of available visualisations, but their support of responsive behavior is very limited. This is an area of interest for further development, as there is currently no software library that has both a wide range of supported visualisation types, while also employing many different responsive visualisation patterns. Having such a library can potentially help increase the usability of data visualisations on the web, especially in a time with so many different types of devices that can be used to view such visualisations.

The fact that we could only find one software editor for responsive data visualisations is another indicator for an area of potential improvement. The editor is not publicly available, so there are effectively none that can be freely used. While the software libraries we listed support responsive behavior, they might not be easy to use for everyone. A software editor could help greatly with making responsive data visualisations. This could improve the user experience of both the users creating the charts and those viewing them on all kinds of devices.

Bibliography

Andrews, Keith [2018]. *Responsive Visualisation*. MobileVis '18 Workshop at CHI 2018 (Montreal, QC, Canada). 21 Apr 2018, pages 1–6. https://mobilevis.github.io/assets/mobilevis2018_paper_4.pdf (cited on page 1).

ApexCharts [2021]. *ApexCharts*. https://apexcharts.com/ (cited on page 15).

Barton, Susanne and Hannah Recht [2018]. *The Massive Prize Luring Miners to the Stars*. 08 Mar 2018. https://bloomberg.com/graphics/2018-asteroid-mining/ (cited on page 9).

Bostock, Mike [2021]. *D3: Data-Driven Documents*. https://d3js.org/ (cited on pages 13, 15).

Chart.js [2021]. *Chart.js*. https://github.com/chartjs/Chart.js (cited on page 15).

FCC [2021]. *FCC Fixed Broadband Deployment*. https://broadbandmap.fcc.gov/#/location-summary?version=jun2020&tech=acfosw&speed=25_3 (cited on page 9).

FusionCharts [2021]. *FaberJS*. https://github.com/fusioncharts/faberjs (cited on page 13).

Harris, Rich [2020]. *A New Technique for Making Responsive, JavaScript-Free Charts*. https://dev.to/richharris/a-new-technique-for-making-responsive-javascript-free-charts-gmp (cited on page 16).

Harris, Rich [2021]. *Pancake*. https://github.com/Rich-Harris/pancake (cited on page 16).

Heyes, Richard [2021]. *RGraph*. https://rgraph.net/ (cited on page 17).

HighCharts [2021]. https://highcharts.com/ (cited on page 15).

Hoffswell, Jane [2020]. *Walkthrough - Total Cost of Major Natural Disasters*. 08 Jan 2020. https://youtu.be/wIvh6UBfMW0 (cited on page 19).

Hoffswell, Jane, Wilmot Li, and Zhicheng Liu [2020]. *Techniques for Flexible Responsive Visualization Design*. Proc. Conference on Human Factors in Computing Systems (CHI 2020) (Honolulu, Hawaii, USA (Virtual)). ACM, 25 Apr 2020, pages 1–13. doi:10.1145/3313831.3376777. https://jhoffswell.github.io/website/resources/papers/2020-ResponsiveVisualization-CHI.pdf (cited on page 19).

IndexMundi [2021]. *Gold Futures End of Day Settlement Price Apr 2011 - Mar 2021*. 09 May 2021. https://indexmundi.com/commodities/?commodity=gold&months=120 (cited on page 11).

Oberrauner, Peter [2021a]. *RespVis*. https://github.com/AlmostBearded/respvis (cited on page 13).

Oberrauner, Peter [2021b]. *RespVis - Brush Zoom*. https://respvis.netlify.app/examples/scatter.html (cited on page 8).

Oberrauner, Peter [2021c]. *RespVis - Grouped Bar Chart*. https://respvis.netlify.app/examples/grouped-bar.html (cited on page 6).

Rabinowitz, Nick [2021]. *rdv*. https://github.com/nrabinowitz/rdv (cited on page 15).

Rost, Lisa Charlotte [2020a]. *17 (or so) responsible live visualizations about the coronavirus, for you to use*. 06 Mar 2020. https://blog.datawrapper.de/coronaviruscharts/ (cited on pages 6–7).

Rost, Lisa Charlotte [2020b]. *Confirmed COVID–19 cases in Europe, per day*. 06 Mar 2020. https://www.datawrapper.de/_/gjz3C/ (cited on page 6).

Rost, Lisa Charlotte [2020c]. *New confirmed COVID-19 cases, per day*. 06 Mar 2020. https://www.datawrapper.de/_/NMZfH/ (cited on page 7).

Svelte [2021]. *Svelte*. https://svelte.dev (cited on page 16).

Vega [2021]. *Vega & Vega-Lite*. https://vega.github.io/ (cited on page 19).

Yaccino, Steven, Jeremy Scott Diamond, and Mira Rojanasakul [2015]. *This is Who Republican Presidential Contenders Follow on Twitter*. 19 Mar 2015. https://bloomberg.com/politics/graphics/2015-who-republican-candidates-follow/ (cited on page 10).