




TGVE: a tool for analysis and visualization of geospatial data

L. Hama¹  R. Beecham^{1,2}  N. Lomax^{1,2} 

¹Leeds Institute for Data Analytics, University of Leeds. ²School of Geography, University of Leeds.

Abstract

We introduce the Turing Geovisualisation Engine (TGVE), a web-based, open-source tool for interactive visualization and analysis of geospatial data. Built on ReactJS and R, TGVE is designed to support a variety of users, including data scientists and stakeholders who wish to engage the wider public with geospatial data. In this short paper, we provide an overview of TGVE's features and capabilities, including its ability to publish data and customize visualization settings using URL parameters. We highlight the potential impact of TGVE for geospatial research and offer examples of its use in practice. Additionally, we discuss current limitations of the tool and outline future work, such as improving compatibility with other geospatial data formats and addressing performance issues for large datasets.

CCS Concepts

• *Information systems* → *Geographic information systems; Web applications*; • *Human-centered computing* → *Visualization toolkits*;

1. Introduction

The complexities of analysing attribute rich data across both space and time are well-documented [BDHL21]. Add to this the requirement for a flexible web application [BBJ*20, CGE13] that can process research datasets, a framework for design innovation and the need for a bespoke and open-source tool becomes apparent. This paper describes the development of such a tool, the Turing Geovisualization Engine (TGVE).

The TGVE is designed with three objectives in mind: (1) enable non-expert users to generate and explore visualizations from geospatial data, (2) enable expert users to integrate the TGVE into their applications and (3) enable production-ready geospatial applications with a minimum amount of code. In this paper, we describe the tool, the technology used to develop it, and provide examples of how the TGVE meets our three objectives. To our best knowledge given certain features of TGVE there are no alternatives for objectives (1) and (3) and the TGVE provides a novel approach to objective (2). Using the TGVE non-experts can publish geospatial data with little effort, whilst expert users can customise and integrate it into their applications.

2. Background

Research teams develop interactive web applications for various purposes to visualize geospatial data from a single source or within a single domain. Because of their specificity, these tools are usually good for one dataset, but their scalability and generalizability can be limited. Recent examples of such tools include Covid-19 data dashboards [Ang20], tools for assessing distributions of disease

risk [HGFF14], the Australian Cancer Atlas [DCA*19] and the opportunity Atlas [CFH*18]. Widely used corporate solutions are generalizable tools offered as proprietary software such as Tableau, PowerBI and GIS packages such as ArcGIS. There are also a limited number of non-proprietary software like QGIS. A third domain is a range of open-source programming language packages (e.g. ggplot2 [Wic16] in R, vega [SRHH15], vega-lite [SMWH16] and d3.js [Zhu13]). Such packages provide users with varying flexibility to build data visualizations from scratch. These packages, however, are limited in generating interactive maps and require a strong degree of technical knowledge.

The closest npm package to the TGVE is KeplerGL [He18]. The difference between the two is the focus of the development: whilst we see KeplerGL as focused on corporate solutions (the developers spun out a company called Unfolded), the TGVE is developed from academic research needs. There are technical differences as well as how the front-end is used within a data science environment. Another open source visualization tool related to the TGVE is Microsoft Sanddance [DF15]. Although Sanddance incorporates DeckGL via Vega, geospatial visualization is not the main focus of Sanddance. Both the TGVE and Sanddance utilize URL parameters as a data source. The TGVE is focused on geospatial visualization and analysis.

3. TGVE

3.1. Tech stack overview

The TGVE is built on low-level open-source packages with focus on representation and interaction [YaKSJ07, RÇD*17, Mac04]. As

a two-tier client-server architecture, we have used R language as the back-end and React (JavaScript) as the front-end; the choice of these two technologies is based on the interaction and representation in data science workflows. The rationale for this approach focuses on picking the right tool for the right job: R for the data processing and React for the UI (User Interface) and visualization. Using both R and React means the application has access to tens of thousands of R packages [CRA21] and over half a million npm packages [AOMS20]. For this two-tier architecture, the TGVE relies on the R package `plumber` [AS19] to implement a RESTful Application Programming Interface (API). The low-level C library used by `plumber` is the same library that powers NodeJS (>v0.9.0) [Mar15] and Python package `Pyuv` [Cor17]. Utilizing low level libraries, the TGVE automates tasks that would otherwise need to be done manually having advanced geospatial skills such as combining datasets.

The current release of the TGVE is made of three components: (1) an npm (Node Package Manager) package called `tgvejs`, (2) a React application using the npm package, and (3) an R package that uses the React app. The React app (see <https://github.com/tgve/app>) can be used for various purposes including Python web applications as shown in Figure 1. Using the app on the code-sharing platform GitHub, users can publish their data using GitHub pages (see section 4.2) or on their own hosting solutions. The React application uses build tools from the ReactJS community [Fre21] which means we can also incorporate other libraries developed around such tools. For instance, we can generate a native (desktop) application using tools like `Tauri` [Tau20] from the TGVE app. Generating native apps using tools like `Tauri` enables a boost to performance as the ReactJS code is compiled to WebAssembly [HCLH18, HRS*17].

3.2. Source code and availability

The project home page lives on GitHub: <https://github.com/tgve>. The TGVE front-end is released as an "npm" scoped package on www.npmjs.com (called the '@tgve/tgvejs'), which can be integrated into other JS applications. For instance, it has been integrated into the R package "tgver" which can be used interactively in R, for instance to visualize "sf" objects (for "sf" objects please refer to [PB18]). For production and reproducibility, there are Dockerfiles [And15] which make it possible to use the application with minimal deployment customisation required in a production environment. This is how both showcases mentioned below were deployed.

3.3. Data formats

Geospatial data are inherently large data and Geographic Information Systems continue to dominate the tools used to process such data. Some standards are widely used and built into programming languages such as R and JavaScript. For instance GeoJSON [BDD*16] is an Open Geospatial Consortium (OGC [OGC18]) standard and being a JSON (JavaScript Object Notation) it is also parsed in JavaScript and other languages.

The TGVE is currently limited to reading JSON (GeoJSON), ESRI shapefiles and Comma Separated Value (CSV) files. The CSV

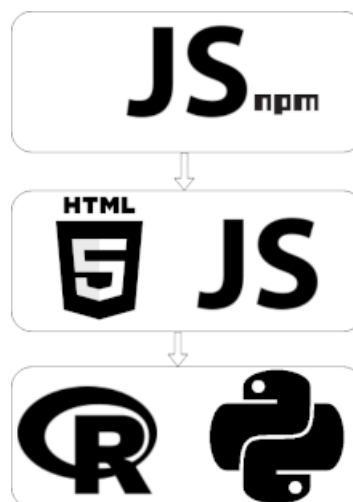


Figure 1: The TGVE is built on practices of modular design both in terms of the choice of the technology used and how they are brought together. The figure shows the three components of the TGVE; starting with JS (npm package) and building towards relevant libraries like CRAN [CRA21] in R.

files can contain location columns and are converted to GeoJSON using a library developed by Mapbox [Map]. However, the TGVE has been used to visualise compact Radar format HDF5 ('.h5') data [Kor11] for entomological research [Bio21], utilising a JS package [USN20] in the browser. The TGVE can use JS libraries to read Shapefiles [ESR97], TopoJSON, ProtoBuf (GeoBuff) and other text/binary formats. We are also aware that such formats can be more compact than GeoJSON. The rationale for adopting Shapefiles is their wide use by the community and the UK's Open Geoportal service [ONS17]. Ideally, the TGVE would be limited to spatial formats that are OGC standards.

3.4. WebGL for Geospatial Visualization

The main geospatial visualization library used in the TGVE is a library called DeckGL [Wan19]. DeckGL uses WebGL (Web Graphics Library) for creating 2.5D/3D views projected over MapboxGL [ER15] map tiles as the base for a world view with primitive shapes such as columns, flat or extruded polygons [Che20]. The use of the Graphics Library (GL) can also be extended to conventional charting libraries. DeckGL provides wrapper functions around WebGL Shading Language (GLSL) to facilitate generating custom visualizations draped on MapboxGL maps (see Figure 2) including triangulated irregular networks (TIN) such as Digital Terrain Models (DTM) [WLHA*98]. It is also possible to generate views that are separate from the Mercator and other geographic projections.

3.5. Flexible views

The main User Interface (UI) of the TGVE is made up of three main interconnected views: 2D slippy maps, DeckGL powered 2.5D visualization layer(s) and a sidebar that includes interactive "widgets"

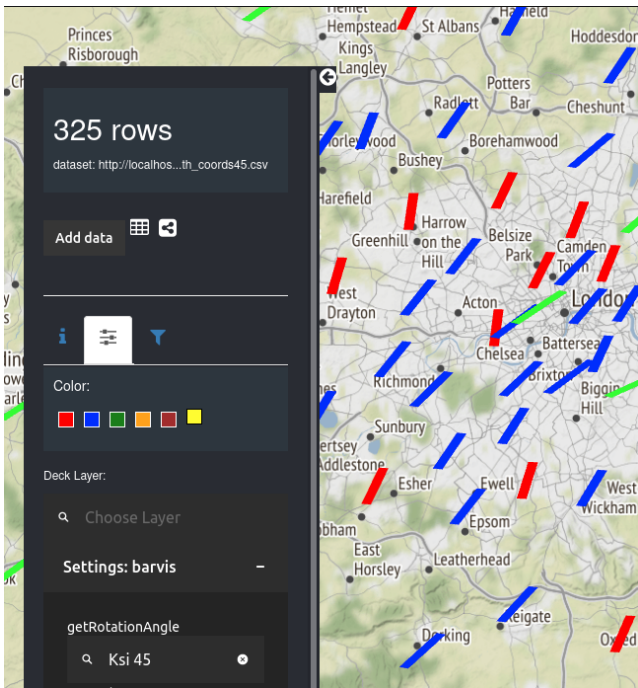


Figure 2: Custom *DeckGL* layer depicting points (longitude and latitude pairs) as bars for a UK road safety sample dataset. The red, green and blue bars represent three categorical values mapped from a column from the dataset. The bars tilted (up to 45°) represent with rotation degrees showing a column of the data which includes values from 0 to 45.

and charts (statistics about the loaded dataset) generated from the active dataset. The TGVE allows users to change the type of visualization and generate charts for column(s) of the dataset loaded. This changes as the data are filtered.

Browser-based interaction "widgets" such as range sliders, search and select and drop-down menus are generated dynamically and can be customised based on the data types in a dataset column, thanks to React and component libraries such as *BaseWeb* [Ube21]. This makes it easier to generate interaction for other components of the TGVE. For instance, to assign column values from a dataset to a particular *DeckGL* layer or visualization feature (e.g. a line width), the TGVE automatically generates the necessary UI elements so that the user can map data columns to the layer generated. In the latest version (1.4.2) of the package, the TGVE also allows hiding/showing specific parts of the application, for example the sidebar that can be seen in Figures 2 and 3.

3.6. Overview, filter and detail

Certain *DeckGL* layers ("grid layers") are aggregation layers that first enable the user to see an overview of the spatial extent of the dataset, before drilling down into the details as needed. In the TGVE, mouse hovering on areas of the visualization can be used to generate line charts for a subset of the data (see 4.1). The slippy maps in general (vector or raster tiles) are hierarchies of overview-



Figure 3: A customised version of Turing Geovisualization Engine showing UK road casualty data. The default view of the a deployment of the SaferActive project shows an area of south London with columns of the dataset featured on the sidebar of the TGVE to focus on and filter the data based on those columns (accident severity, casualty travel mode, etc).

detail by design with zoom levels determining how much detail is shown on a particular map view.

Filtering of data in the TGVE is based on exact matches for values in a particular column of a dataset. Geospatial filtering can be done using slippy map view boundary (screen view) or in the case of origin-destination data (lines) by selecting an origin or a destination. For temporal filtering, currently exact match is used but this is under active development for alternative (range) filtering. Overall, the tool aims to give an overview of any dataset loaded, provide filtering to individual data points working with *DeckGL* that generates the visualization.

4. Using the TGVE

As mentioned in 3.1 the three components of the TGVE makes it possible to use it in some distinct ways. Firstly the *npm* package is there for JavaScript users (React users) to use the TGVE in their applications. The aim of the R package is to use the TGVE in data science workflows, for example in Rmarkdown (like Jupyter notebooks in R) and also offline for interactive data analysis and visualization. The third component which is the template repository on GitHub and GitHub pages can be used in these ways: (a) embed the TGVE in notebooks just by using the URL API values such as adding source of data, type of visualization etc.; (b) ad-hoc visualization of remote datasets; (c) publishing data on GitHub on

user and organisations spaces on GitHub (see section 4.2); and most importantly (d) go-to TGVE instance to load data from local machines using the TGVE's user interface from local machines. The TGVE user interface enables users to read either a file that includes geography or a CSV file with a corresponding geography file. We believe use-cases (b) and (d) meet our objective (1).

So far, the application has been used to visualise a range of geospatial datasets. The npm package (see 3.1) can be integrated into other React applications, and the `tgver` is developed to be used in R data workflows. The app hosted on GitHub is designed to be used anywhere where the web is used. That includes Jupyter notebooks and Rmarkdown documents as the application takes data sources as URL parameters (see more in 4.2).

The TGVE has been used for two research projects to analyse large geospatial data: SaferActive and SPENSER [Ham20, LSA*22]. In both projects, the tool is used as a "full-stack" production-ready solution to serve national datasets and as front-end analytics. The Docker-based deployment means both projects can be deployed on hosting infrastructure with the capacity to scale. Due to space constraints we focus on how the TGVE was used in the SaferActive project, and briefly outline the data pipeline.

4.1. SaferActive showcase

SaferActive is a UK Department for Transport (DfT) funded research project which aims to generate a model of road casualty data for the prioritisation of investment in traffic calming measures for vulnerable road users. Part of the project visualises years of national road casualty data. In this showcase, there are three main datasets: (1) STATS19 "raw" point data from the DfT; (2) aggregated UK level statistics at Local Authority District (LAD) and Police Force geographies of the UK of Killed or Seriously Injured (KSI); and (3) UK level route-network raster tiles generated from kilometers cycled. A fork of the TGVE was used to accommodate the tiles but this was a minor change. This is an example of how objective (2) is met where expert users can customise the TGVE to visualize geospatial research output.

The pipeline of the point dataset uses an R package called `stats19`, whilst the R package `plumber` is used to build the API to serve the data. The backend of the showcase is a `plumber` R script that serves the three datasets mentioned above. This can be found at the project's organisation space on GitHub [ITS20].

The point data (dataset 1) is processed in the same script. The steps of the processing can be summarised as: limiting the years of the dataset (2015 to 2019) using the R package `stats19`, limit the number of columns served to the front end, combining accident and casualty tables of the dataset, and using a more efficient way of spatial filtering rather than using `sf` R package (using `R.data.table`). The R script (backend) serves one API endpoint which returns a GeoJSON output of the accidents within a bounding box. At the front-end, the STATS19 point data is processed by DeckGL layers, and various DeckGL layers can be used to generate pin maps, heatmaps, square grids and more depending on the layer chosen.

In this showcase, the application is a two tier architecture which was hosted (for the duration of the project) on a commercial hosting

provider using a Docker image for scalability and reproducibility. This showcase illustrates how the TGVE is used to visualize the research outputs and the application source code is production-ready using the TGVE. SaferActive also showcases that the TGVE is best suited for multivariate analyses such as the attribute-rich dataset of STATS19 [Hai95]. Figure 3 shows the latest version of the project.

4.2. Publishing data using TGVE

GitHub Pages is a service provided by GitHub to deploy "websites" for code repositories. As the TGVE is a web application, users can publish data using the TGVE on GitHub pages. There is documentation [Ham21] guiding users on how to set up forks of the TGVE app by cloning the template repository (<https://github.com/tgve/app>).

Another feature of the TGVE is its ability to accept data sources and other configurations as URL (Unified Resource Locator) parameters. For example, the app at <https://tgve.github.io/app> can be customised just by passing parameters like: `'tgve.github.io/app?defaultURL=domain.com/file.csv&dark=false'`. This removes the need for having a local instance or the need to "install" the TGVE. This makes it possible to also run the React app in a Jupyter notebook using this feature by just adding HTML's 'iframe' element. A novelty in the TGVE compared with other tools, to our knowledge, is the ability to combine data with geography on the fly using the same URL parameters approach (see [TGV21]).

5. Conclusion and future work

The TGVE is a web-based geovisualization tool built with R and React, which can be used in different environments to enhance and facilitate data science workflows. The novelty lies in creating a flexible tool, which draws on the relative strengths of R + React for geospatial data analysis. The TGVE makes adding data sources, defining flexible views and filtering using URL parameters easy, meaning it suitable for non-expert users who want to generate visualizations from geospatial data and explore the structure of those data. The R package specifically is aimed at expert R users who may wish to use the tool offline without any development toolchain. The utility of the application is demonstrated in this paper through a showcase example, where it has been used to analyse and visualize UK national-level datasets.

The project is evolving and the current focus is on prioritising areas for development. One important area is to develop more layers for the TGVE for multivariate data visualization. For example further utility could be offered by allowing a user to pick 3D (2.5D) shapes to represent data and map columns to such shapes. Other areas for development include converting ESRI shapefiles [ESR97] and the development of the R package to assist data scientists in undertaking exploratory analysis of data from spatial databases.

6. Acknowledgement

This work was supported by Towards Turing 2.0 under the EPSRC Grant EP/W037211/1 & The Alan Turing Institute.

References

- [And15] ANDERSON C.: Docker [software engineering]. *Ieee Software* 32, 3 (2015), 102–c3. 2
- [Ang20] ANGUS C.: Covid plots and analysis. *University of Sheffield* 10 (2020). 1
- [AOMS20] ABDALKAREEM R., ODA V., MUJAHID S., SHIHAB E.: On the impact of using trivial packages: An empirical case study on npm and pypi. *Empirical Software Engineering* 25, 2 (2020), 1168–1204. 2
- [AS19] ALLEN J., S B.: plumber: An api generator for r, 2019. URL: <https://www.rplumber.io>. 2
- [BBJ*20] BREUNIG M., BRADLEY P. E., JAHN M., KUPER P., MAZROOB N., RÖSCH N., AL-DOORI M., STEFANAKIS E., JADIDI M.: Geospatial data management research: Progress and future directions. *ISPRS International Journal of Geo-Information* 9, 2 (2020), 95. 1
- [BDD*16] BUTLER H., DALY M., DOYLE A., GILLIES S., HAGEN S., SCHAUB T., ET AL.: The geojson format. *Internet Engineering Task Force (IETF)* (2016). 2
- [BDHL21] BEECHAM R., DYKES J., HAMA L., LOMAX N.: On the use of ‘glyphmaps’ for analysing the scale and temporal spread of covid-19 reported cases. *ISPRS International Journal of Geo-Information* 10, 4 (2021), 213. 1
- [Bio21] BIODAR: Visualizing radar data with tgve, 2021. URL: <https://github.com/biodar/bioAtlas>. 2
- [CFH*18] CHETTY R., FRIEDMAN J. N., HENDREN N., JONES M. R., PORTER S. R.: *The opportunity atlas: Mapping the childhood roots of social mobility*. Tech. rep., National Bureau of Economic Research, 2018. 1
- [CGE13] CASTRONOVA A. M., GOODALL J. L., ELAG M. M.: Models as web services using the open geospatial consortium (ogc) web processing service (wps) standard. *Environmental Modelling & Software* 41 (2013), 72–83. 1
- [Che20] CHEN X.: deck.gl technical deep dive, February 2020. 2
- [Cor17] CORRETTÉ S.: 2017. URL: <https://github.com/saghul/pyuv>. 2
- [CRA21] CRAN: Cran: Contributed packages, 2021. URL: <https://cran.r-project.org>. 2
- [DCA*19] DUNCAN E. W., CRAMB S. M., AITKEN J. F., MENGERSEN K. L., BAADE P. D.: Development of the australian cancer atlas: Spatial modelling, visualisation, and reporting of estimates. *International journal of health geographics* 18, 1 (2019), 21. 1
- [DF15] DRUCKER S., FERNANDEZ R.: A unifying framework for animated and interactive unit visualizations. *Microsoft Research* (2015). 1
- [ER15] ERIKSSON O., RYDKVIST E.: An in-depth analysis of dynamically rendered vector-based maps with webgl using mapbox gl js, 2015. 2
- [ESR97] ESRI: Esri shapefile technical description, July 1997. 2, 4
- [Fre21] FREEMAN A.: Creating a react app. In *Essential TypeScript 4*. Springer, 2021, pp. 473–495. 2
- [Hai95] HAIGNEY D.: Stats19. *INROADS* 17, 2 (1995). 4
- [Ham20] HAMA L.: University of leeds spenser visualization using tgve, 2020. URL: <https://geospenser.com>. 4
- [Ham21] HAMA L.: Tgve app readme, 2021. URL: <https://github.com/tgve/app>. 4
- [HCLH18] HERRERA D., CHEN H., LAVOIE E., HENDREN L.: Web-assembly and javascript challenge: Numerical program performance using modern browser technologies and devices. *University of McGill, Montreal: QC, Technical report SABLE-TR-2018-2* (2018). 2
- [He18] HE S.: From beautiful maps to actionable insights: Introducing kepler.gl, 2018. URL: <https://eng.uber.com/keplergl/>. 1
- [HGFF14] HANSELL A. L., GHOSH R., FORTUNATO L., FECHT D.: *The environment and health atlas for England and Wales*. Oxford University Press, USA, 2014. 1
- [HRS*17] HAAS A., ROSSBERG A., SCHUFF D. L., TITZER B. L., HOLMAN M., GOHMAN D., WAGNER L., ZAKAI A., BASTIEN J.: Bringing the web up to speed with webassembly. In *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation* (2017), pp. 185–200. 2
- [ITS20] ITS: Saferactive: prioritising investment in traffic calming measures for vulnerable road users, 2020. 4
- [Kor11] KORANNE S.: Hierarchical data format 5: Hdf5. In *Handbook of open source tools*. Springer, 2011, pp. 191–200. 2
- [LSA*22] LOMAX N., SMITH A. P., ARCHER L., FORD A., VIRGO J.: An open-source model for projecting small area demographic and land-use change. *Geographical Analysis* 54, 3 (2022), 599–622. 4
- [Mac04] MACEACHREN A. M.: *How maps work: representation, visualization, and design*. Guilford Press, 2004. 1
- [Map] MAPBOX: csv2geojson. URL: <https://github.com/mapbox/csv2geojson>. 2
- [Mar15] MARATHE N.: An introduction to libuv. *Nikhilm. github. io* (2015). 2
- [OGc18] OGC: Simple feature access - part 1: Common architecture, 2018. URL: <https://www.ogc.org/standards/sfa>. 2
- [ONS17] ONS: April 2017 open geography portal frequently asked questions, April 2017. 2
- [PB18] PEBESMA E., BIVAND R.: sf: Simple features for r. r package version 0.6-3. URL: cran.r-project.org/package=sf (2018). 2
- [RÇD*17] ROTH R. E., ÇÖLTEKIN A., DELAZARI L., FILHO H. F., GRIFFIN A., HALL A., KORPI J., LOKKA I., MENDONÇA A., OOMS K., ET AL.: User studies in cartography: opportunities for empirical research on interactive maps and visualizations. *International Journal of Cartography* 3, sup1 (2017), 61–89. 1
- [SMWH16] SATYANARAYAN A., MORITZ D., WONGSUPHASAWAT K., HEER J.: Vega-lite: A grammar of interactive graphics. *IEEE transactions on visualization and computer graphics* 23, 1 (2016), 341–350. 1
- [SRHH15] SATYANARAYAN A., RUSSELL R., HOFFSWELL J., HEER J.: Reactive vega: A streaming dataflow architecture for declarative interactive visualization. *IEEE transactions on visualization and computer graphics* 22, 1 (2015), 659–668. 1
- [Tau20] TAURI: Build smaller, faster, and more secure desktop applications with a web frontend, 2020. URL: <https://tauri.studio>. 2
- [TGV21] TGVE: Tgve npm package on www.npmjs.com, 2021. URL: <https://www.npmjs.com/package/@tgve/tgve.js>. 4
- [Ube21] UBER ENGINEERING: Base web react ui framework, 2021. URL: <https://baseweb.design/>. 3
- [USN20] USNIST: jsfive: A pure javascript hdf5 file reader, 2020. URL: <https://github.com/usnistgov/jsfive>. 2
- [Wan19] WANG Y.: Deck.gl: Large-scale web-based visual analytics made easy. *arXiv preprint arXiv:1910.08865* (2019). 2
- [Wic16] WICKHAM H.: *ggplot2: elegant graphics for data analysis*. springer, 2016. 1
- [WLHA*98] WOOD L., LE HORS A., APPARAO V., BYRNE S., CHAMPION M., ISAACS S., JACOBS I., NICOL G., ROBIE J., SUTOR R., ET AL.: Document object model (dom) level 1 specification. *W3C recommendation 1* (1998). 2
- [YaKSJ07] YI J. S., AH KANG Y., STASKO J., JACKO J. A.: Toward a deeper understanding of the role of interaction in information visualization. *IEEE transactions on visualization and computer graphics* 13, 6 (2007), 1224–1231. 1
- [Zhu13] ZHU N. Q.: *Data visualization with D3.js cookbook*. Packt Publishing Ltd, 2013. 1