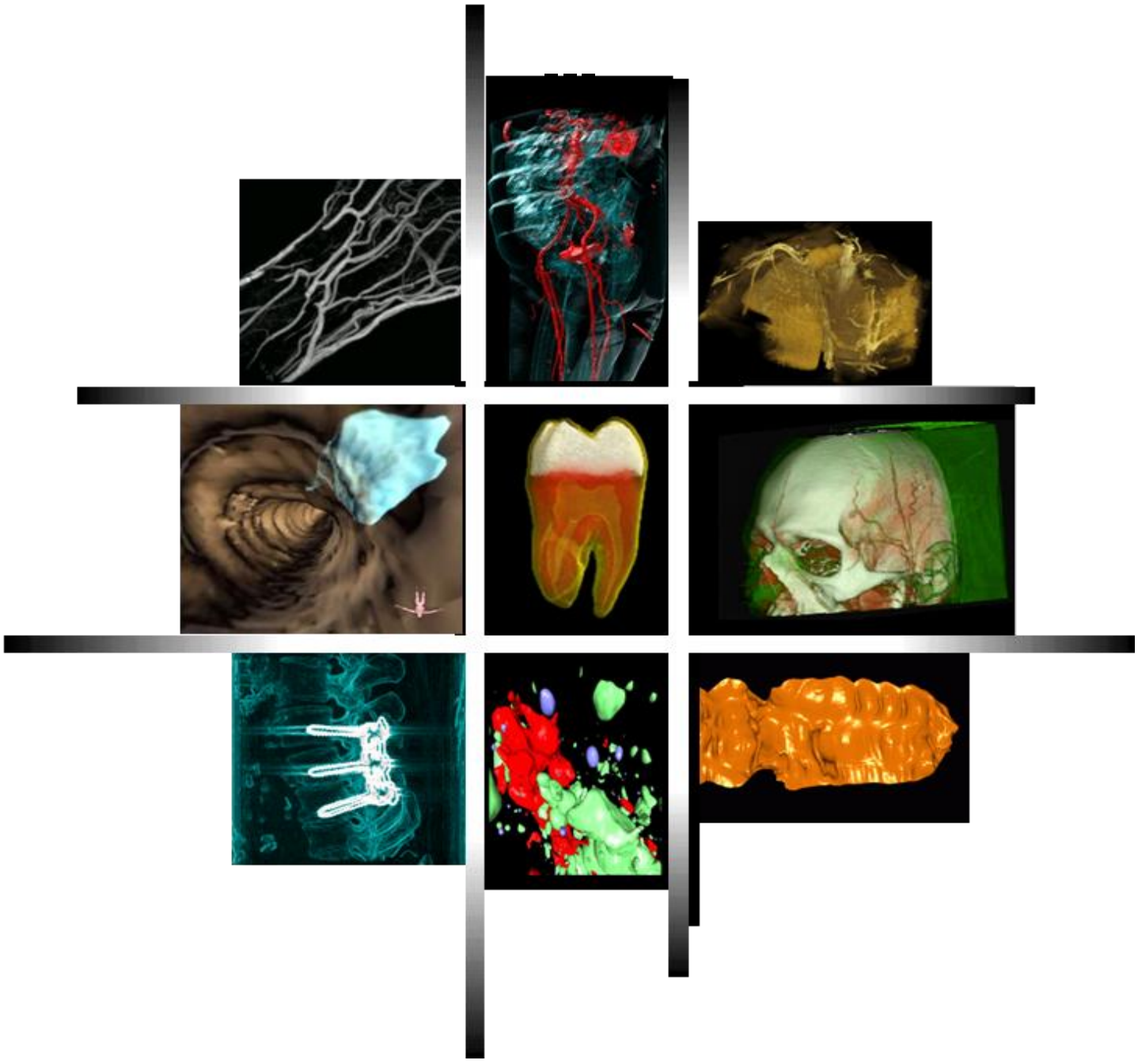
6171 UE Exercise:
Data Visualization

SoSe 25

Exercise – d3 Dashboard

Instruction-Sheet



General information

In this exercise, classic **InfoVis techniques** are implemented for the analysis of multivariate data and connected via interactive selections. We use the widely used **d3** library and **JavaScript** to generate Scalable Vector Graphics (**SVG**) on a web page.

The exercise comprises two parts: the first covers loading and parsing CSV data, followed by data representation in a tabular form. The same data should then be interactively examined via scatterplots and radar charts. The second part of the exercise draws upon the skills from part one and expands into developing a comprehensive visualization dashboard. The dashboard ideas will be discussed in an 1st presentation of the implementation concept. The goal is to select a new dataset which can be explained by the develop visualization- and interaction techniques.

For this exercise you will be provided with the following files:

(1) index.html
(2) style.css
(3) datavis.js
(4) dashboard.js
(5) 3 csv Datasets

In 1-3 is the necessary HMTL and JavaScript code to create empty axes for the scatterplot and radar chart using the JavaScript library d3 for part 1. For the menu jQuery UI is used. In 4 a basic javascript file is provided in which the dashboard function can be implemented.

Please note the versions used:

Library	Version
D3.js	7.x
jQuery	3.x
jQuery UI	1.13.x

Regarding d3, note that there are many online tutorials based on an older version of d3 which are partly not compatible anymore! It is recommended to use the official API reference of d3 version 7: https://github.com/d3/d3/blob/main/API.md .

The development environment that you use is up to you, we recommend e.g. Visual Studio IDE 2022 or JetBrains WebStorm.

Submission information

There are a total of **50 points to be achieved for this Exercise**.

Part 1:

- Data loading/preparation (1)
- Displaying data table (1)

- Adapting axes (1.5)
- Render scatterplot points (4)
- Scatterplot selection (2.5)
- Element Deselection (1)
- Radar Chart (3)
- Animated transitions (1)

Part 2:

- Dashboard Concept/Specification Sheet (10)
- Dashboard Implementation (25)
    - Rendering of at least 4 (non-static) charts
    - Arrangement of visualization techniques
    - Possibility to display non-numeric attributes
    - Calculation of new Values from Dataset
    - Adding Interaction and Transitions (Linking & Brushing, Tooltips)
    - Other methods/techniques from the Lecture
    - Final Presentation

Minimum requirement: The minimum requirement is achieved by implementing Part 1 with correct axes, correctly scaled and positioned points and implemented interactions. For the minimum requirement additional development in Part 2 must be provided (new visualization techniques, interactions, …).

Your Submission should include the following files:

- index.html (*Insert your names here in the place marked TODO*)
- style.css
- datavis.js
- dashboard.js (*if additional .js files are used add them*)
- data.csv (*your own dataset*)
- groupX_screenshot_part1.png (*your finished visualization*)
- groupX_video_part1.mp4 (*max 1min!*)
- groupX_screenshot_part2.png (*your finished dashboard*)
- groupX_video_part2.mp4 (*max 1min!*)

Part 1: Loading, Scatterplot and Radar Chart in d3

Tasks

Load data (1 points)

The data can be loaded via a FileUpload button. The functionality to parse the file is already implemented.

For this exercise, data can be loaded that matches the following format:

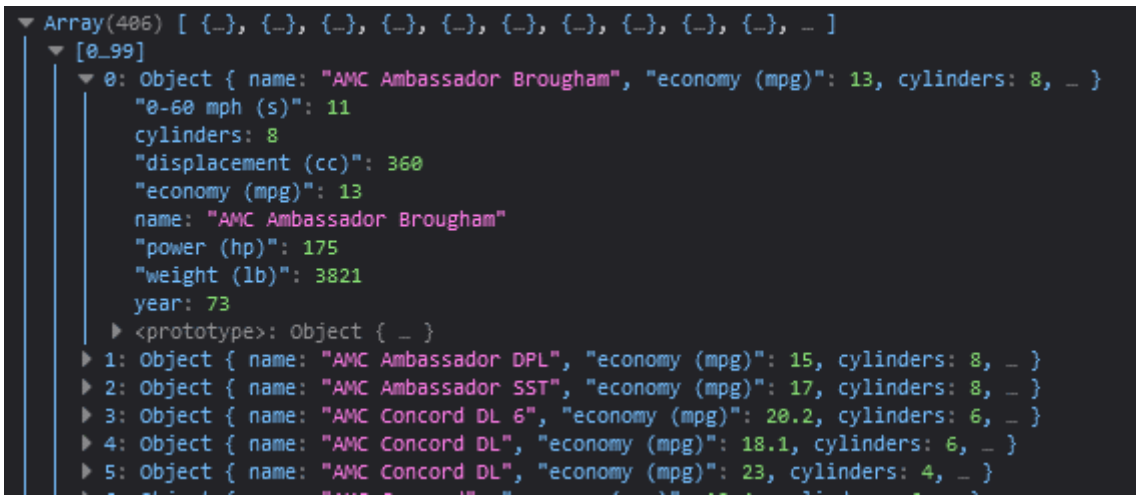
Name/ID	**Numeric attribute**	**Numeric attribute**	**…**
Label	1	2	…
…	…	…	…

In csv files, the titles of the columns are in the first row. We provide 3 widely used data sets:

- **cars.csv:** Car brands with attributes like year of manufacture, acceleration, consumption....
- **flowers.csv:** ("Iris dataset"): 3 types of iris plants with petal and sepal lengths.
- **nutritients.csv**: food with attributes like energy, protein, sugar, (Attention: a lot of data!)

TODO:

The data must be prepared in such a way that they can be used for visualization. A good way to do this is to use the d3-dsv (d3-csv) functionality. The result should be an array of JavaScript objects, like here for the cars dataset:



The names of the numerical attributes that should be visualized should be stored in the existing array "dimensions":

```
▼ Array(7) [ "economy (mpg)", "cylinders", "displacement (cc)", "power (hp)", "weight (lb)", "0-60 mph (s)", "year" ]
    0: "economy (mpg)"
    1: "cylinders"
    2: "displacement (cc)"
    3: "power (hp)"
    4: "weight (lb)"
    5: "0-60 mph (s)"
    6: "year"
    length: 7
  ▶ <prototype>: Array []
```

Display data table (1 points)

TODO:

Display the read file with all its attributes in a HTML table with the help of d3 on the first Tab (Data Loading). To be able to achieve the same appearance please look at the provided CSS styles.

Highlight the current position of the mouse (Hint: hover).

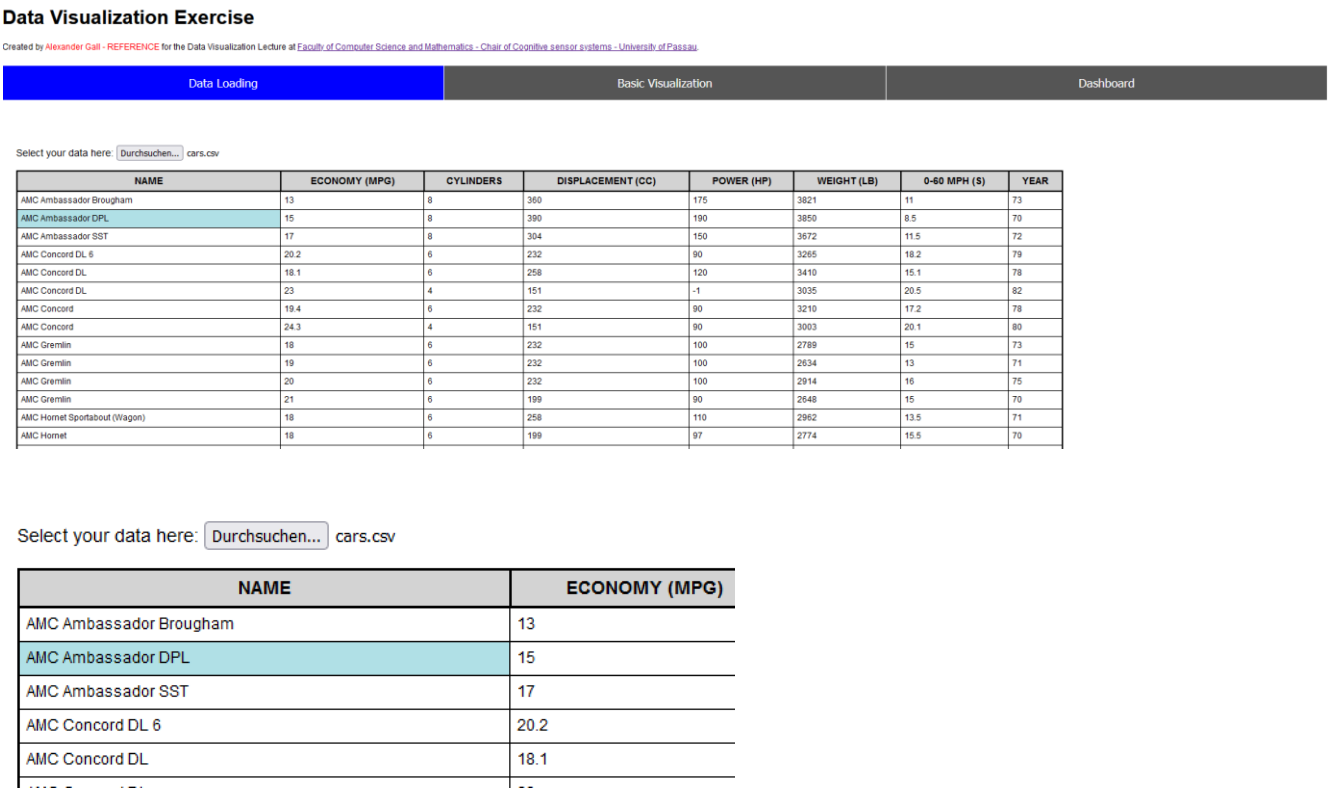


Figure 1

Adapt axes (1.5 points)

TODO:

Label all scatterplot and radar chart axes with the correct attributes. The attributes that are displayed in the scatterplot are selected from the menu. The Radar Chart should always show all numeric attributes. In the radar chart, draw also the **gridlines** (light gray lines in a circle around the axes).

Scale all axes according to the attributes to be displayed.

As an example, Figure 1 shows the result for the "cars" dataset (with economy and power axes selected in the scatterplot):

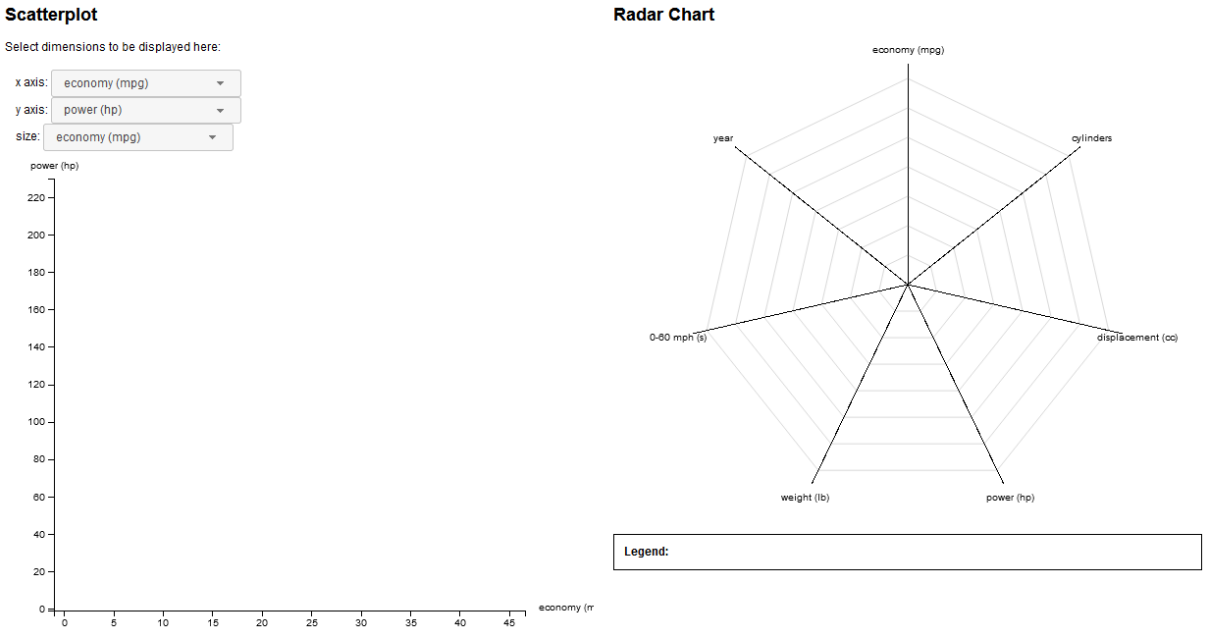


Figure 2

Render scatterplot points (4 points)

TODO:

For each item (i.e. line in the input file), **render** the points position and size according to the three attributes selected in the menu. Make sure that when you reselect the axes, the visual channels are adapted correctly.

Note: The color, transparency, etc. is up to you. Feel free to experiment with transparencies, shadows, or other effects.

Figure 2 shows the result for the "cars" dataset (with economy and weight axes selected in the scatterplot):

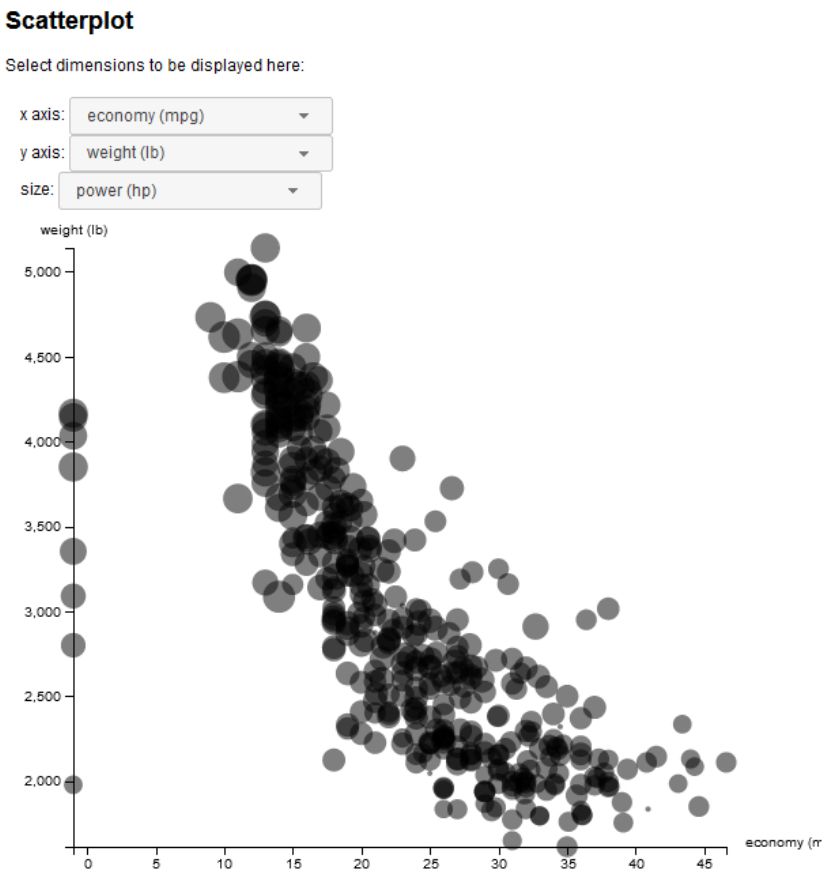


Figure 3

Scatterplot selection (2.5 points)

TODO:

By clicking on a point in the scatterplot, points should be selected and marked. Each selected point is assigned its own color and is displayed in the legend below the radar chart.

The legend should display the name or ID (i.e. the first, non-numeric, attribute in the dataset) of the selected point along with the assigned color. The way it is displayed is up to you.

The important thing is that the legend is easy to read, no colors are duplicated, and no more than a predefined number of items can be selected by you (5 to 10). When choosing colors, it is important that you choose a color scheme so that the individual categories can be easily distinguished.

In Figure 3 you can see the result for the "cars" dataset with 5 elements selected

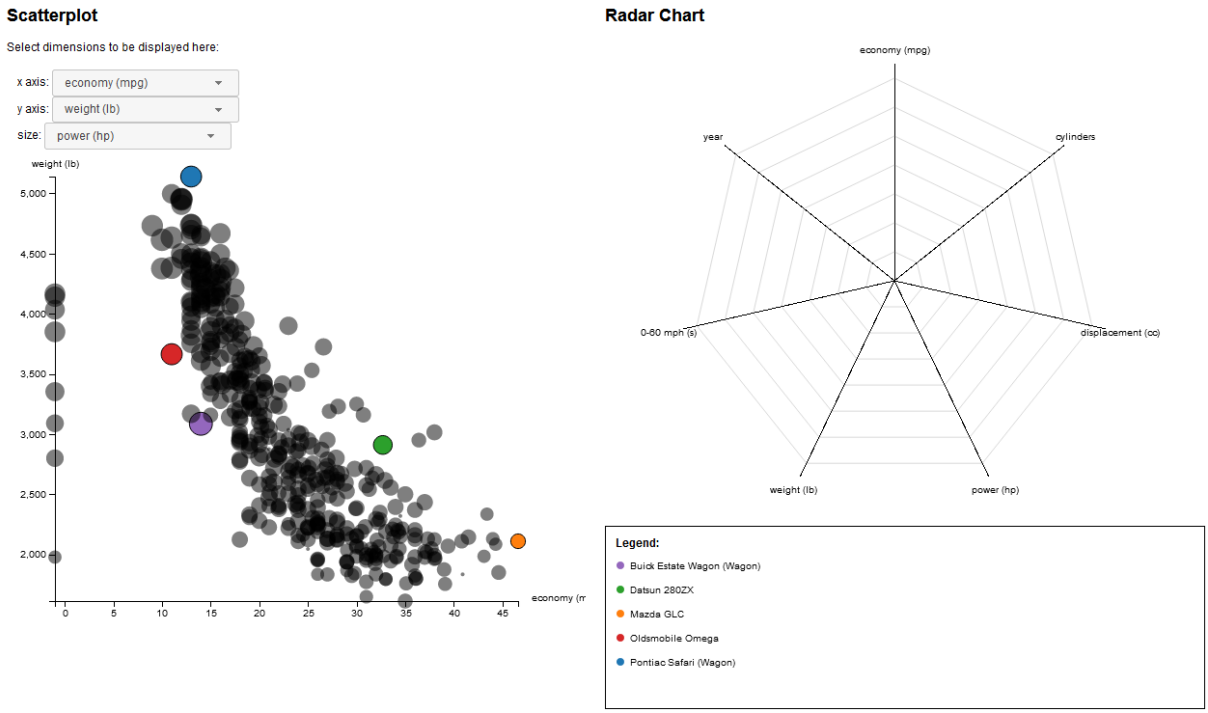


Figure 4

Deselection (1 point):

Add the option to also **deselect elements** via the legend (i.e. the element is no longer displayed in the radar chart and the color highlighting in the scatterplot is removed). Please also note the CSS styles that are already available!

Radar Chart (3 points)

TODO:

The scatterplot should be linked to the radar chart, i.e. the selected points in the scatterplot should be displayed as **lines in the radar chart**.

Make sure that the colors are assigned consistently (i.e. same color of the selected item in the scatterplot and the corresponding line in the radar chart) and correctly adapt or delete the lines when the user makes a new selection in the scatterplot.

Figure 4 shows the result for the "cars" dataset with 5 selected elements.

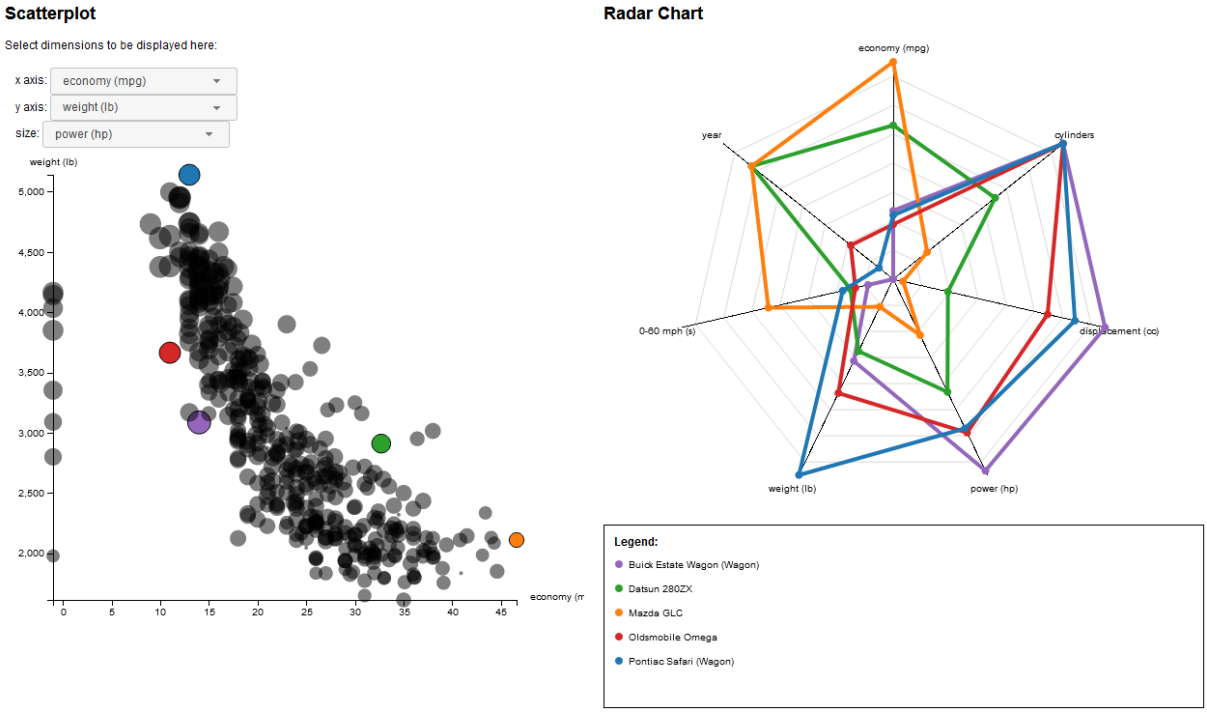


Figure 5

Animated transitions in the scatterplot (1 point)

D3 provides ways to **animate transitions** as the data changes. When the user changes the mapping of the attributes for the scatterplot via the menu, the change of the points (i.e. the position, the color, the size) should be animated.

Part 2: Dashboard

Dashboard Concept (10 points).

Select a multidimensional dataset of your choice that was **not provided by us** (see Task "Load data"). This dataset should be used for your Dashboard. The goal is to select a dataset (optionally several similar ones), which is then analyzed using appropriate visualization techniques. In the specification sheet your task is to describe the chosen dataset and which visualization techniques you need to analyze it efficiently. Provide Mockups of the arrangement, the Interactions, transition and everything the Dashboard will offer.

You can find possible datasets, for example, here:

- https://archive.ics.uci.edu/datasets (UCI Machine Learning Repository)
- https://www.kaggle.com/datasets (Kaggle Datasets)
- https://www.gapminder.org/data/ (Gapminder Datasets)

Possible visualization ideas can be found here, for example:

- https://observablehq.com/@d3/gallery (Examples from d3)
- https://datavizcatalogue.com/index.html (Examples of info vis types)

Important Note:
Please make sure that there are **no missing values** in your data set, as this is not handled in the given framework. If there are missing values in your data you need to adjust the code to handle these cases (gives you points) or add the missing values in a preprocessing step. Information about Data Imputation can be found here: https://towardsdatascience.com/6-different-ways-to-compensate-for-missing-values-data-imputation-with-examples-6022d9ca0779

Dashboard Implementation (25 points).

The main part of this exercise will focus on the **implementation of visualization techniques** for the dashboard. Here, the information obtained from the lecture and the concept presentation will be applied to implement a dashboard to analyze a data set exploratively or systematically (with hypothesis). This means that depending on the data select charts that can represent the underlying data structure (e.g., node-link diagrams, histograms, geographic maps,...), which are connected by interactions (e.g., selection of country in map and displays it in bubble chart,...) and that allows to explain the data set (e.g., less coal power yields less CO2 emissions). During the submission interview, you should be able to explain the data using the Dashboard.

An example of the dashboard page (with a not optimal encoding/design!) can be seen in figure 6.

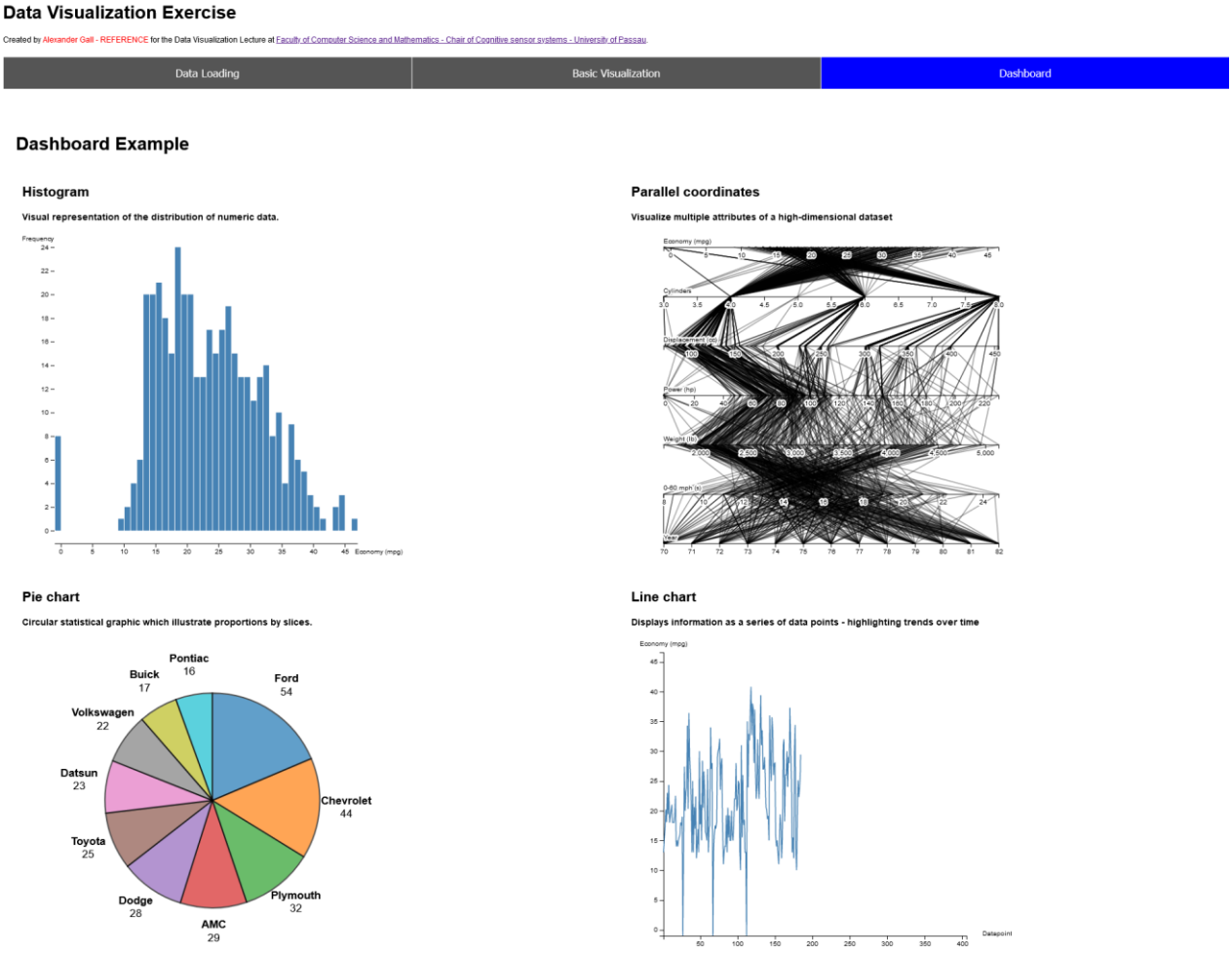


Figure 6

To get started, we have created the "Dashboard" tab on the webpage (see index.html) and the dashboard.js file. The dashboard.js file is intentionally kept relatively empty so that you can design your dashboard creatively. This means you can generate additional js files or add new methods or delete existing ones in the dashboard.js file. This also applies to the div containers in the HTML for the dashboard tab, where you can also make changes. It is also allowed, e.g. if you need more space, to use the tab only to open another HTML file for your dashboard. For the purpose of a better web interface, it is also allowed to use tools such as React, Angular, etc. for the dashboard. However, **d3 should still be used where possible**.

This does not apply to Part 1 - follow the specifications and do not use any additional libraries or similar!

Some criteria for evaluating the Dashboard:

- **Dashboard Implementation:** We a grading the implementation (should work during live demo and when we test it) and documentation of the code.

- **Dashboard Explanation:** As in the concept presentation as well as in the final presentation - explain the reasons for the chosen metaphors and techniques. Why was this technique chosen and are there other alternative methods?
- **Rendering of at least four effective visualizations:** Create 4 charts to analyze your dataset. More (useful) charts can increase the points.
- **Add interaction and transitions:** Enhance the implemented charts by visualization methods like Linking & brushing, filtering, selections, animations and other techniques. Try to use the interaction so that the charts support each other. Try to find at least one interactive method for each of your 4 visualizations.
- **Effective arrangement and encoding of visualization techniques:** Assess the encoding of the visualizations. Does the arrangement, encoding (color, geometry, …) facilitate intuitive data analysis?
- **Display non-numeric attributes and calculate derived values:** Additional points (and more Chart too choose from) are available if you extend the data loading of Part 1 to include non-numeric attributes or perform computations on the dataset (mean, variance, correlations,…).
- **Other methods/techniques from the lecture incorporated into the visualization:** Bonus points for creativity and innovation.

Final Presentation

Explain in a live demo (with some presentation slides) the reasons for the chosen metaphors and techniques, and possible alternatives. Explain the structure of your code and show us selected parts of your implementation and explain their functionality. All Team members should know and understand the code and used visualization techniques.

References:

- D3js.org with many examples (Attention: many use an older API version!) https://d3js.org/
- D3 API reference: https://github.com/d3/d3/blob/master/API.md
- More d3 examples from Mike Bostock, author of d3: https://bl.ocks.org/mbostock
- D3 tutorial: http://square.github.io/intro-to-d3/
- JQuery UI user interface: https://jqueryui.com/
- Parsing data with d3: https://github.com/d3/d3-dsv