Visualization

Winter Semester 2025/2026

Prof. Tobias Günther

Friedrich-Alexander-Universität
Technische Fakultät

Programming 3 - Data Aggregation and Bar Chart

1 Introduction

In previous exercises we created a visualization of countries, world graticules and bubbles, which represent incidents where migrants went missing or died. From those bubbles, we get a good overview of spatial locations where these incidents happen the most. In this exercise, we will add the time attribute to our visualization. For that, we will add the time attribute in a different view because most of our available channels are already used up in our world map. Furthermore, we want to avoid overloading of the world map since it already visualizes the bubbles on top of the world map and encoding more attributes on other channels might make it difficult to see anything at all. In the following tasks you will first aggregate the data and then visualize it using a bar chart.

Task 1: Data Aggregation (1 Pts)

The first step is to aggregate the data. For that, we must setup some helper functions and variables. You will find appropriate ToDos in your *bar_chart.js* file. Every ToDo for this first task can be found before and within the Histogram component. Start from the top until the ToDos refer you to this PDF. Please make sure to not remove the return statement for the placeholder rectangle before you proceed to the next task. Don't forget to include the *bar_chart.js* script in your *index.html.* The last ToDo for this task is the actual histogram computation. The following will provide you with a detailed explanation of the histogram computation inside the `Histogram` React component.
The `data` of this exercise comprises a discrete time series, given in tabular form. Each item of the data set has a time stamp and a quantity, representing a certain number of migrants who went missing. To provide a temporal overview, we will aggregate the quantitative data in a histogram, for which we use the function `d3.histogram()`. Since this is a utility function provided in D3, the parameters are set by successively calling setter functions. To define the histogram computation, we need to set three things:

- The histogram needs an accessor function that extracts the data attribute to be counted. The accessor is set with `value(xValue)`. In our case, we defined `xValue` earlier, which defines which attribute to map to the horizontal axis, i.e., the time axis.
- Then, the histogram is defined over a certain range of the x-axis. For this, we set the domain to `domain(xScale.domain())`.
- Lastly, we need to tell the time stamps that splits the time axis into bins. Those can be computed automatically using `d3.timeMonths`, which we assign to the histogram property `thresholds`, i.e., we set `thresholds(d3.timeMonths(start, stop))`.

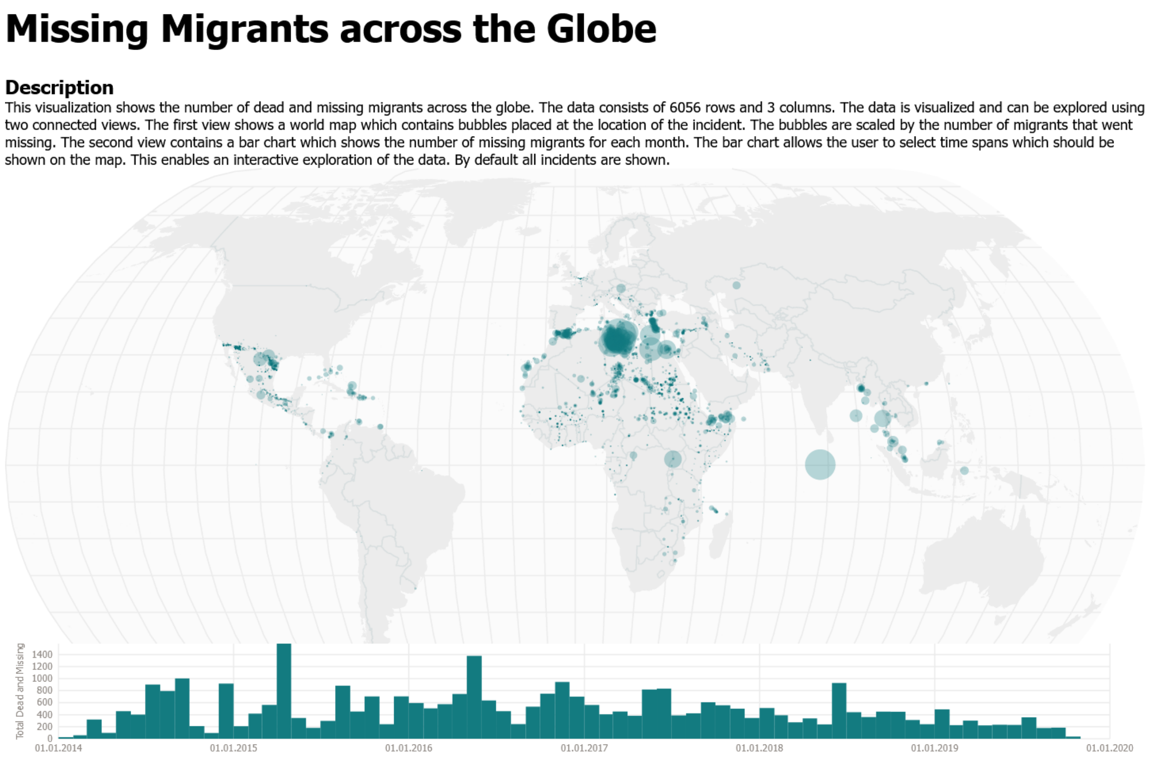
Once the histogram object is initialized, we call it as a function and pass the `data` into it, which performs the binning. This is a good moment to pause and think where `data` is actually coming from. It should be passed into the `Histogram` component (along with width and height). Don't forget to include the *Histogram* component into *app.js.* The resulting object contains the bins. We call a map function on that resulting object to generate a new data object that stores three values for each bin. We will set `x0: array.x0`, `x1: array.x1`, and `y: d3.sum(array, yValue)`. The values `x0` and `x1` store the beginning and end value of the bin on the x-axis, and we define `y` which will simply sums up all the elements in the bin. The resulting list of triples is stored into a `const binnedData` variable.

Task 2: Bar Chart (4 Pts)

In this task we will draw the complete bar chart which shows our aggregated data. The first thing we will do is to remove the placeholder rectangle. Since we don't want to have the world map as the background to our chart we will return a filled white rectangle instead, inside a React fragment. All the other components will be placed inside a separate group component that transforms them according to the margins. Inside that we will place the following components:

- *AxisBottom*,
- *AxisLeft*,
- *Bars*,
- and a label for axis left.

For all of those components you will find detailed ToDos at the top of the file. We recommend to start with the *Bars* component, proceed to the two axis and at the labels last. Whenever you have finished one of the components, you can add them to the group component. As always, feel free to style the Bar Chart anyway you like but make sure to stick to guidelines you've learned in the lecture.



Missing Migrants across the Globe

Description

This visualization shows the number of dead and missing migrants across the globe. The data consists of 6056 rows and 3 columns. The data is visualized and can be explored using two connected views. The first view shows a world map which contains bubbles placed at the location of the incident. The bubbles are scaled by the number of migrants that went missing. The second view contains a bar chart which shows the number of missing migrants for each month. The bar chart allows the user to select time spans which should be shown on the map. This enables an interactive exploration of the data. By default all incidents are shown.

Fig. 1: Solution to this exercise: Display of a histogram of missing migrants over time.