Faculty of Science, Engineering and Technology

COS30045 Data Visualisation

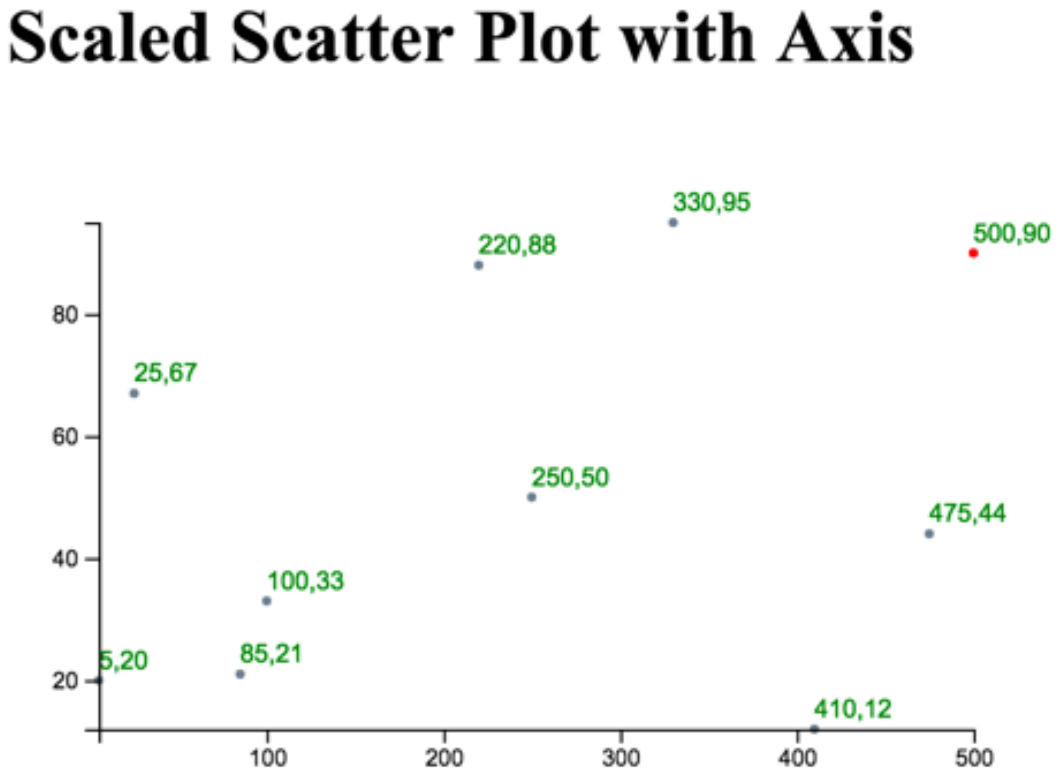
Task 3.2 D3 Adding axis to your charts

ILO	Create web-based interactive visualisations using real-world data sets.
Aim:	Use D3 adding axis to your charts.
Resources:	*Textbook:* Murray Ch 8 Murray on ProQuest Murray on Safari (Make sure you use v2 of Murray as per links above)
To be marked as Complete your submission must:	Submit working code that meets the requirements specified in document below. Demonstrate appropriate use of HTML, CSS and D3. Properly formatted code Well commented code with references to code sourced from web, stack overflow etc. where appropriate. Demonstrate and explain code to tutor in class.
Submission	Submit to Doubtfire • screenshot of final webpage demonstrating the use of axis • code Bring code to class to demonstrate to tutor

Note: The functions handling scale have changed between D3 v3 and D3 v4. This is something to be aware of if you are doing your on research into this topic. Make sure you use Murray Ed 2. Code examples from Ed 1 will not work.

Overview

In this tutorial we will start with your code from Task 3.1 (the scaled scatter plot). At the end of this Task you should end up with a scatter plot drawn using D3 generated SVGs that scales and has axis. It will look something like this.



Step 1: Start with the code from Task 3.1

We will start with the code from the previous scatter plot task which should be scalable (i.e., you can change the input data domain and the size of the SVG and the data will display properly).

Note: Make sure your D3 JavaScript code is in a separate .js file and call it from the header.

Step 3: Add the X-axis

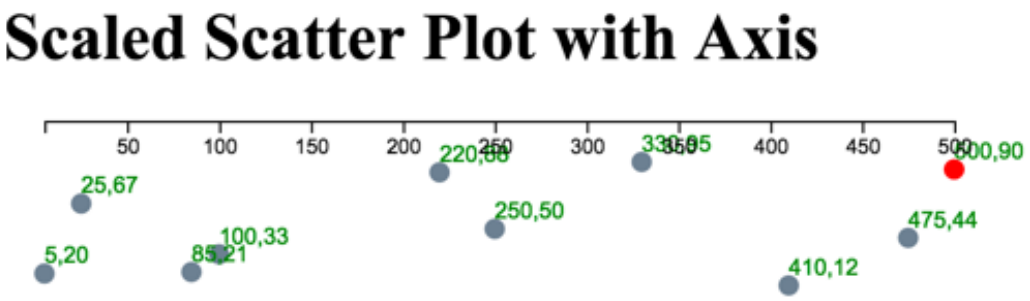
D3 has four functions to construct axis to place the ticks on the axis at either the top, bottom, left or right of the axis line. We will start by adding an axis at the bottom of our scatter plot. For this we want the ticks on our axis to point down so we will use the `d3.axisBottom()` function.

```
var xAxis = d3.axisBottom()
              .scale(xScale);
```

We want our axis to scale along with our plot, so we will pass the xScale to our axis as well. Although this generates our axis, it doesn't display it. To display it we need to append it to the page. Because SVGs are displayed in the order they appear in the code, we will need to put our axis as the bottom of our chart code so it is displayed (otherwise the chart SVG will just be put over the top). To draw the axis we use a `g` element. `g` elements are a grouping element that allows us to group elements together. Our axis is make up of a combination of lines and text and if we want to manipulate it, it's good to be able to do it to all elements at once (thus `g` element is made to contain them). The D3 `call()` function gets the selection (i.e., the `g` element) and gives it our function (i.e., `xAxis`). Thus all the bits generated by our axis function end up in the `g` element.

```
svg.append("g")
    .call(xAxis);
```

If you run this code you should get something like this:



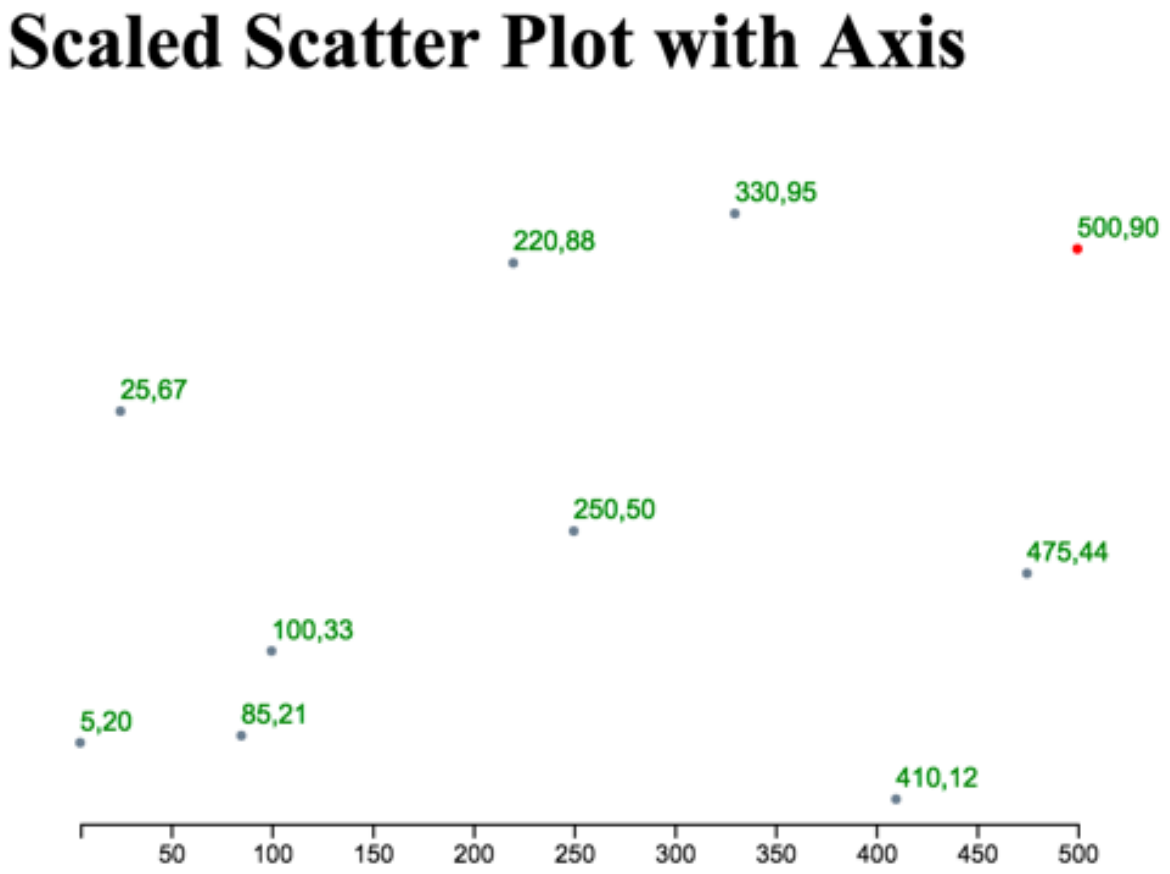
If you check the DOM you can see all the axis elements sitting in the `g` element.

```
▼<g fill="none" font-size="10" font-family="sans-serif" text-anchor="middle">
  <path class="domain" stroke="#000" d="M20.5,6V0.5H460.5V6"></path>
  ▼<g class="tick" opacity="1" transform="translate(60.5,0)">
    <line stroke="#000" y2="6"></line>
    <text fill="#000" y="9" dy="0.71em">50</text>
  </g>
  ▶<g class="tick" opacity="1" transform="translate(104.94444444444444,0)">…</g>
  ▶<g class="tick" opacity="1" transform="translate(149.38888888888889,0)">…</g>
  ▶<g class="tick" opacity="1" transform="translate(193.83333333333331,0)">…</g>
```

Unfortunately, the position of our axis is not great. Again, D3 has positioned it at (0,0). So we need to move it down to the bottom of the plot. We can do this using `transform`. There are a number of different transforms. The one we will use to move our axis is `translate`. Translate pushes the location of the SVG across or down some amount.

```
svg.append("g")
    .attr("transform", "translate(0, "+ (h - padding) +")")
    .call(xAxis);
```

In this case the the SVG is moved `0` pixels in the horizontal direction and `h-padding+10` pixels in the vertical direction (i.e., down to the bottom of the SVG canvas).



Note: the `h` of the SVG in this screenshot has been increased from 100 to 300 so the data is not so squashed up.

Step 4: Controlling the Ticks

The number of ticks can be specified using the tick function. However, D3 takes any number you put here as a suggestion only and may use a different number of ticks if it thinks your specified number won't work elegantly.

```
//from Murray
    var xAxis = d3.axisBottom()
                  .ticks(5)
                  .scale(xScale);
```

See Murray (Ch 8) for what to do if you are determined to specify particular tick marks.

Step 4: Adding the Y-axis

Adding the y-axis is pretty much the same as adding the x-axis, just using the yScale instead. The thing to watch out for is the values of the translate function.

Finally…

Take a look at Murray Ch 8 for some more options on formatting axis' and ticks. He also includes some code for randomly generating data sets to demonstrate the the flexibility of the code.