Faculty of Science, Engineering and Technology

COS30045 Data Visualisation

Task 8.1 D3 Pie Chart

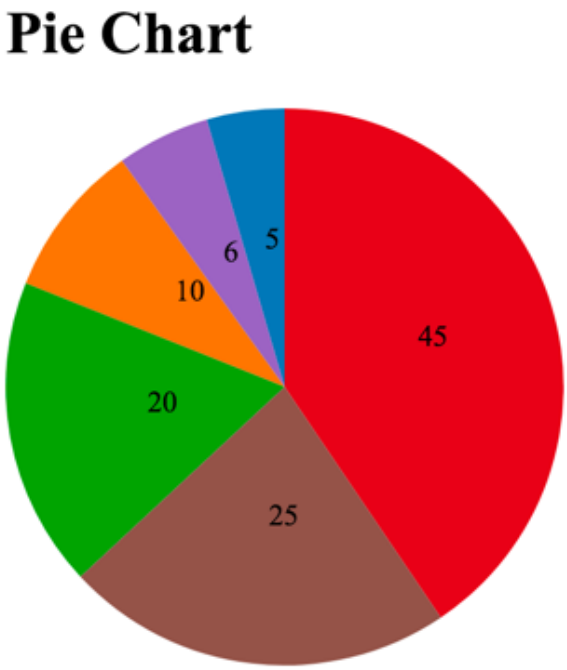
ILO	Create web-based interactive visualisations using real-world data sets.
Aim:	Generate a Pie Chart using D3
Resources:	*Textbook:* Murray Ch 13 Murray on ProQuest Murray on Safari (Make sure you use v2 of Murray as per links above)
To be marked as Complete your submission must:	Submit working code 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 • code that demonstrates a pie chart 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.

Code in this Task based on Murray Ch 13

Overview

In this exercise we will use D3 to draw a pie chart.



Step 1: Set up the data

First we need some data to display. So start by adding a data set with between 5 and 10 numbers. We will also be displaying this chart on a SVG with a width and height of 300.

Step 2: Set up the pie chart parameters

To generate the paths for our pie chart we can use `d3.pie()`. If you add our data into the pie function it will generate angles which can be used to draw the segments of our pie chart. However, it does not draw the chart. For the we need `d3.arc()`. To customise the size of our pie chart we can specify the outer and inner radius of the pie. The outer radius will be specified according to the size of the SVG (i.e., using `w`).

```
var outerRadius = w /2;
var innerRadius = 0;

var arc = d3.arc()
            .outerRadius(outerRadius)
            .innerRadius(innerRadius);

var pie = d3.pie();
```

Later you can change the value of the inner radius to generate donut charts.

Step 3: Set up the SVG and set up the arcs

As per usual set up the SVG canvas.

```
var svg = d3.select("body")
            .append("svg")
            .attr("width", w)
            .attr("height", h);
```

After this we can set up our arcs.

```
var arcs = svg.selectAll("g.arc")
              .data(pie(dataset1))
              .enter()
              .append("g")
              .attr("class", "arc")
              .attr("transform", "translate(" + outerRadius + "," + outerRadius + ")");
```

Note that the data being read into the SVG is that form the `pie()` function (i.e., the one that generates the angles we need to draw the segments. Also note that the arcs are being added as a group set.

Finally, the default position for the chart is with the centre of the chart to be positioned at 0, 0 (i.e., the top left hand corner of the SVG). Therefore we need to move (i.e., transform) the centre of the chart to the centre of the SVG.

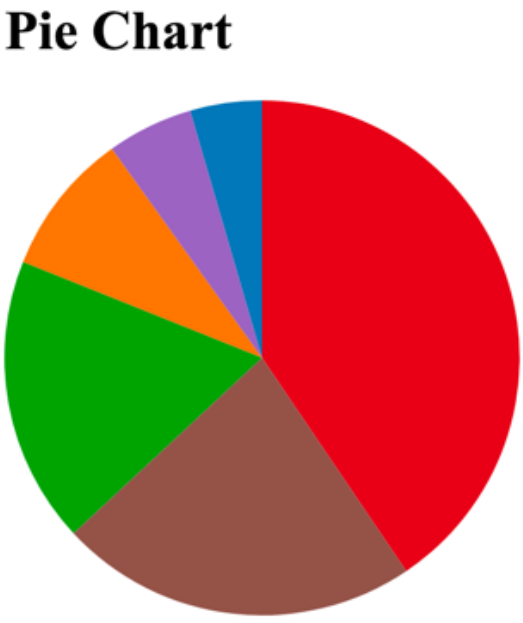
Step 4: Draw the arcs

The arc generator (`d3.arc()`)can be used to generate the paths for the data bound to the `arcs` group. We can also attach the colour fill using a colour scale. The one used here uses a basic d3 native scheme (no. 10). You can update this to something from colorbrewer if you wish.

```
arcs.append("path")
    .attr("fill", function(d, i) {
        return color(i);
    })
    .attr("d", function(d, i) {
        return arc(d, i);
    });
```

```
var color = d3.scaleOrdinal(d3.schemeCategory10);
```

You should now be able to generate a pie chart!



Step 5: Add text labels

We can add some text labels to the chart:

```
arcs.append("text")
    .text(function(d) {
        return d.value;
    });
```

Note that because the pie generator generates an array containing all the path data as well as the data value, to access the value to display we need to use `d.value`. If you run this you will notice that all the values are positioned at the centre of the chart. To push them out to their respective segments we can use the `transform` and `arc.centroid()`. `arc.centroid()` finds the middle of an irregular shape.

```
.attr("transform", function(d) {
    return "translate(" + arc.centroid(d) + ")";
})
```