Shape Generators

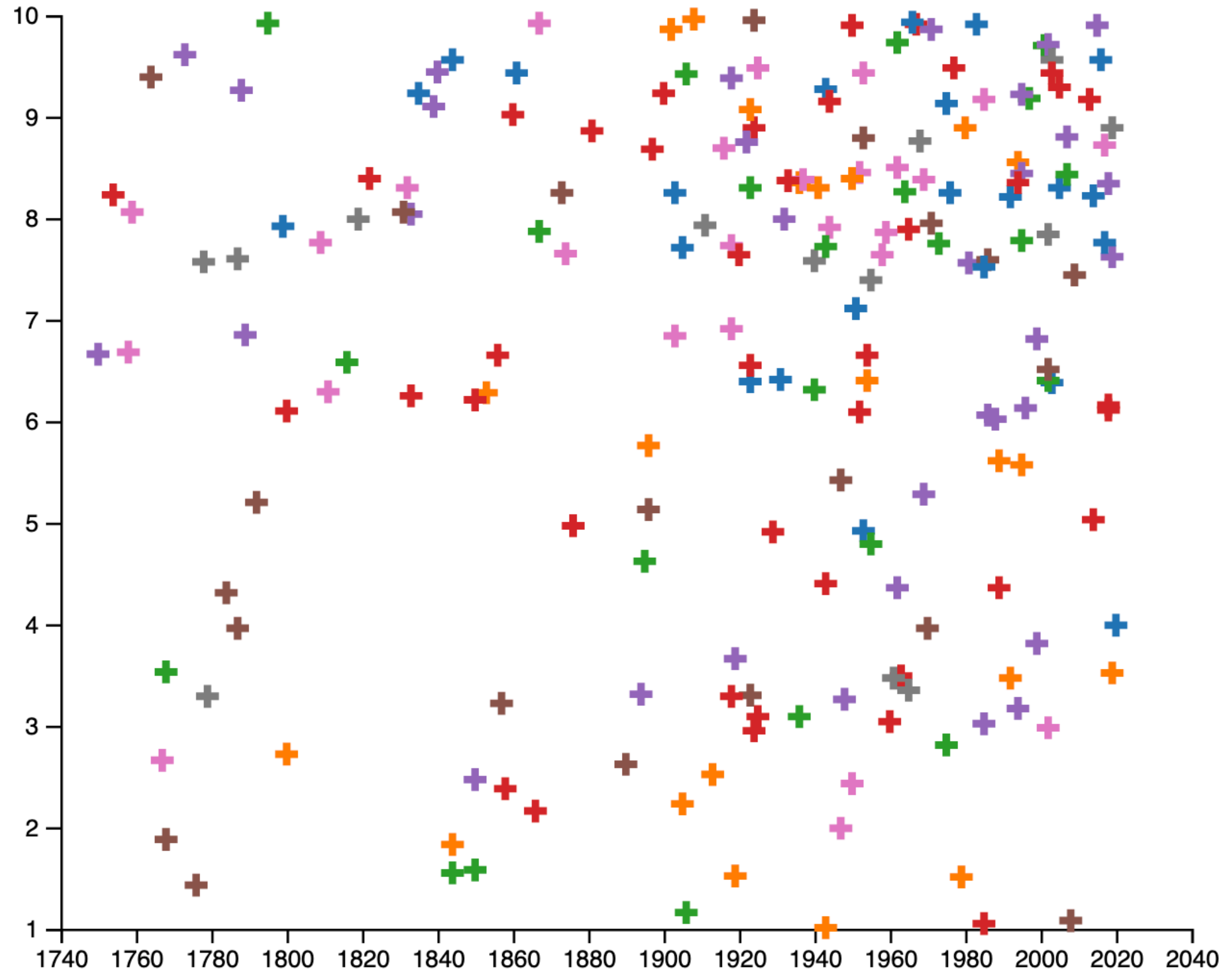
Shape Generators

D3 helper functions for building common SVG paths

Shape Generators

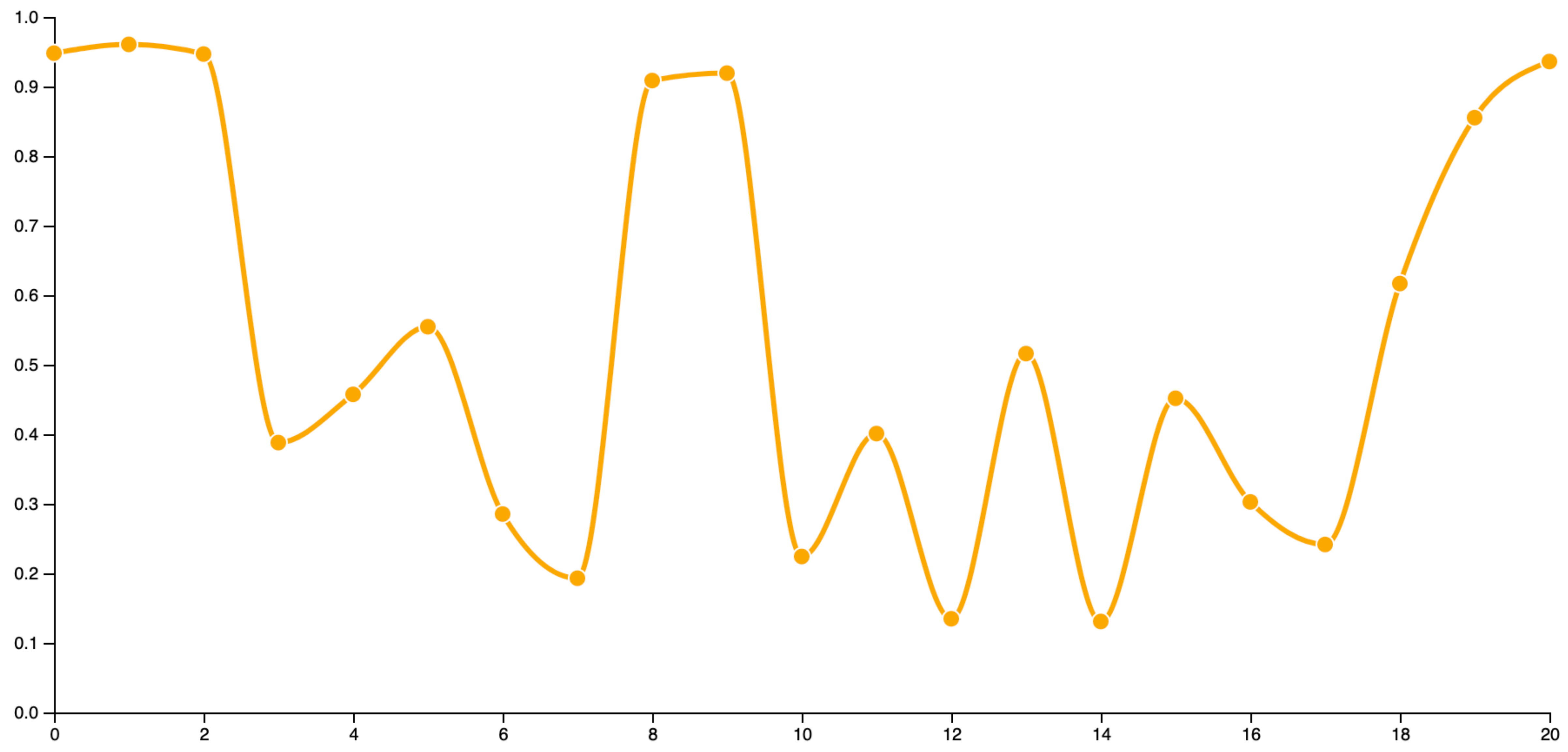
d3.symbols()	d3.line()	d3.curve()
d3.area()	d3.stack()	d3.pie()

d3.symbols()



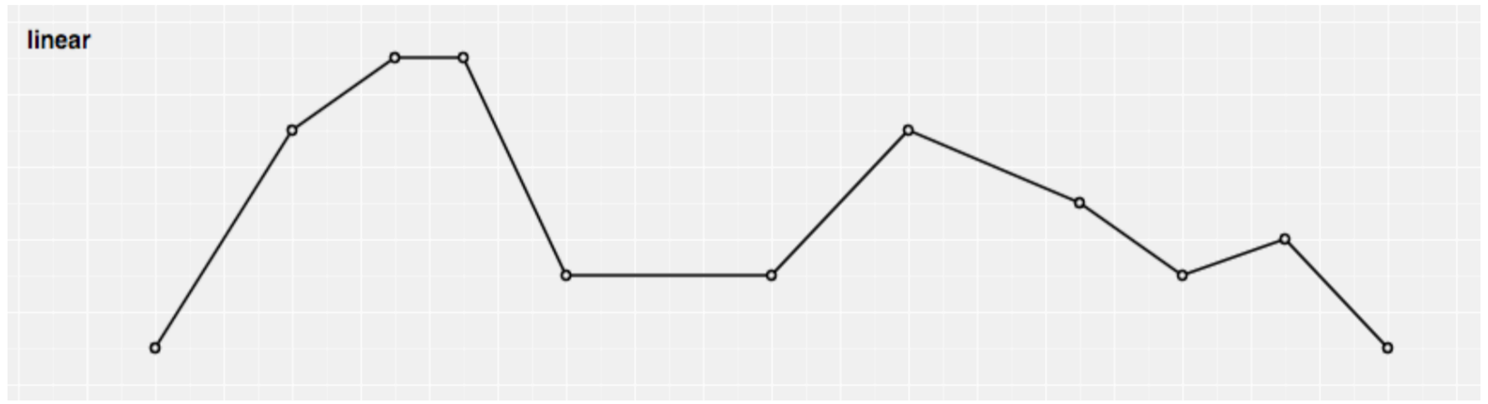
https://observablehq.com/@d3/fitted-symbols
http://using-d3js.com/05_10_symbols.html

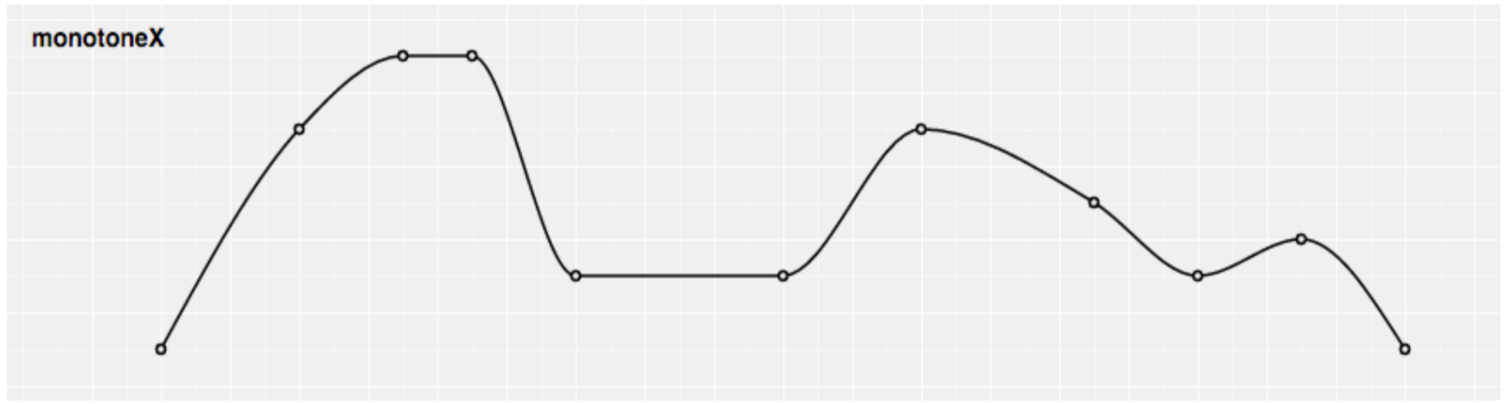
d3.line()

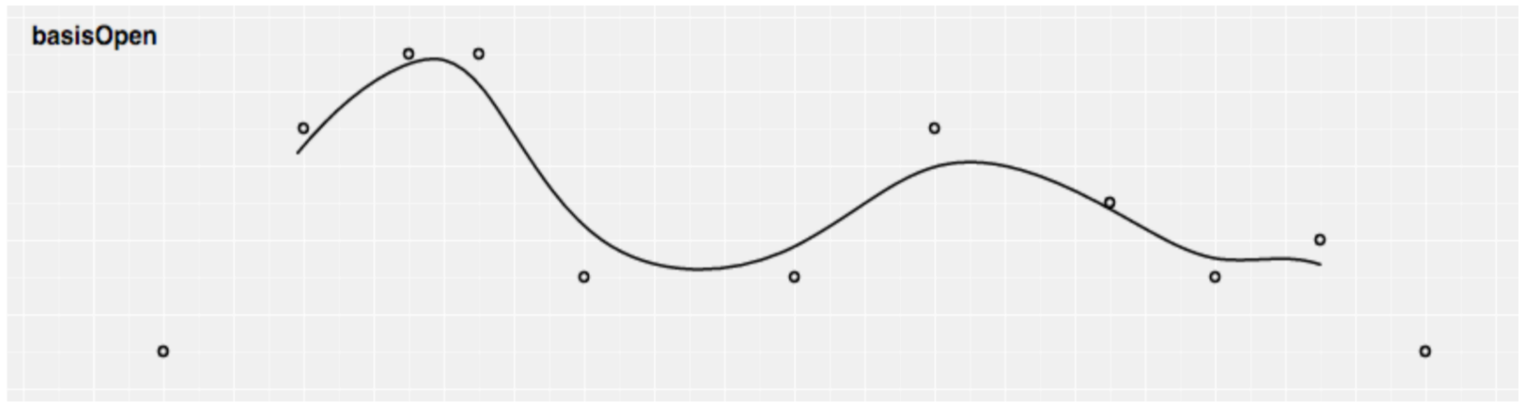


https://bl.ocks.org/gordlea/27370d1eea8464b04538e6d8ced39e89

d3.curve...()

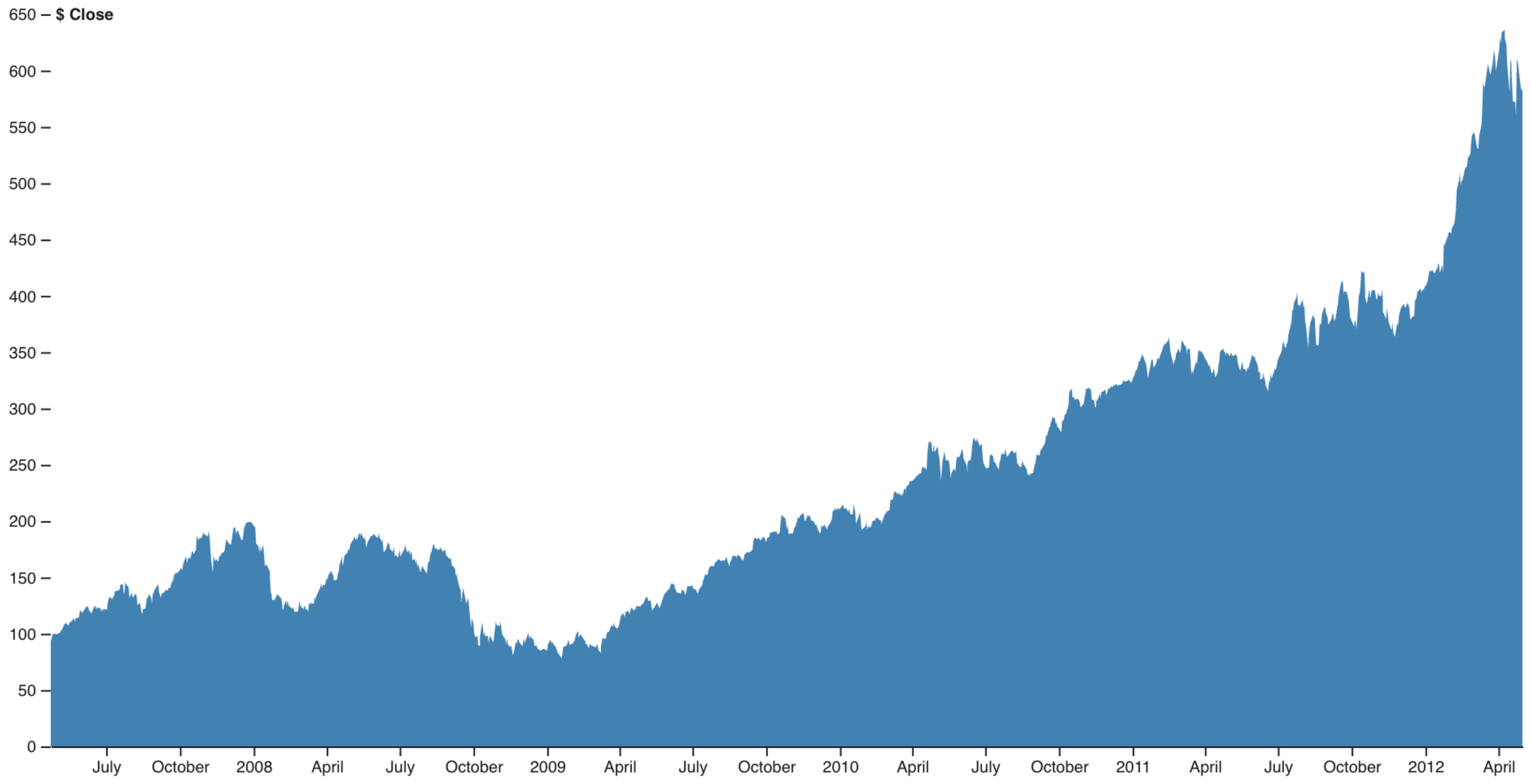






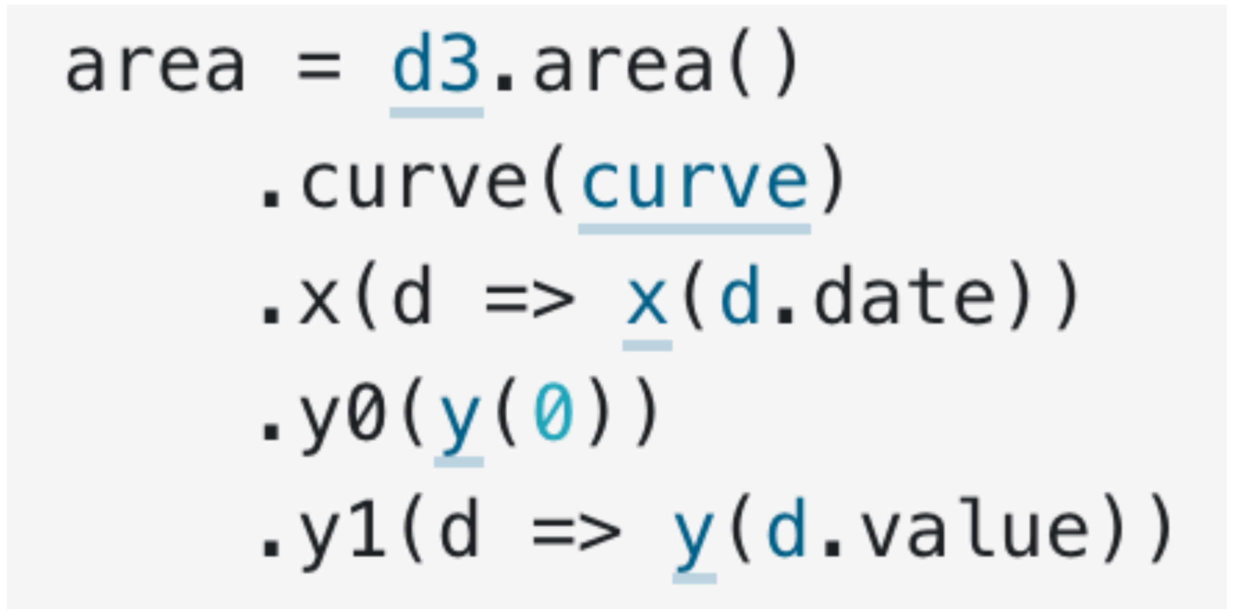
https://github.com/d3/d3-shape#curves

d3.area()



https://observablehq.com/@d3/area-chart

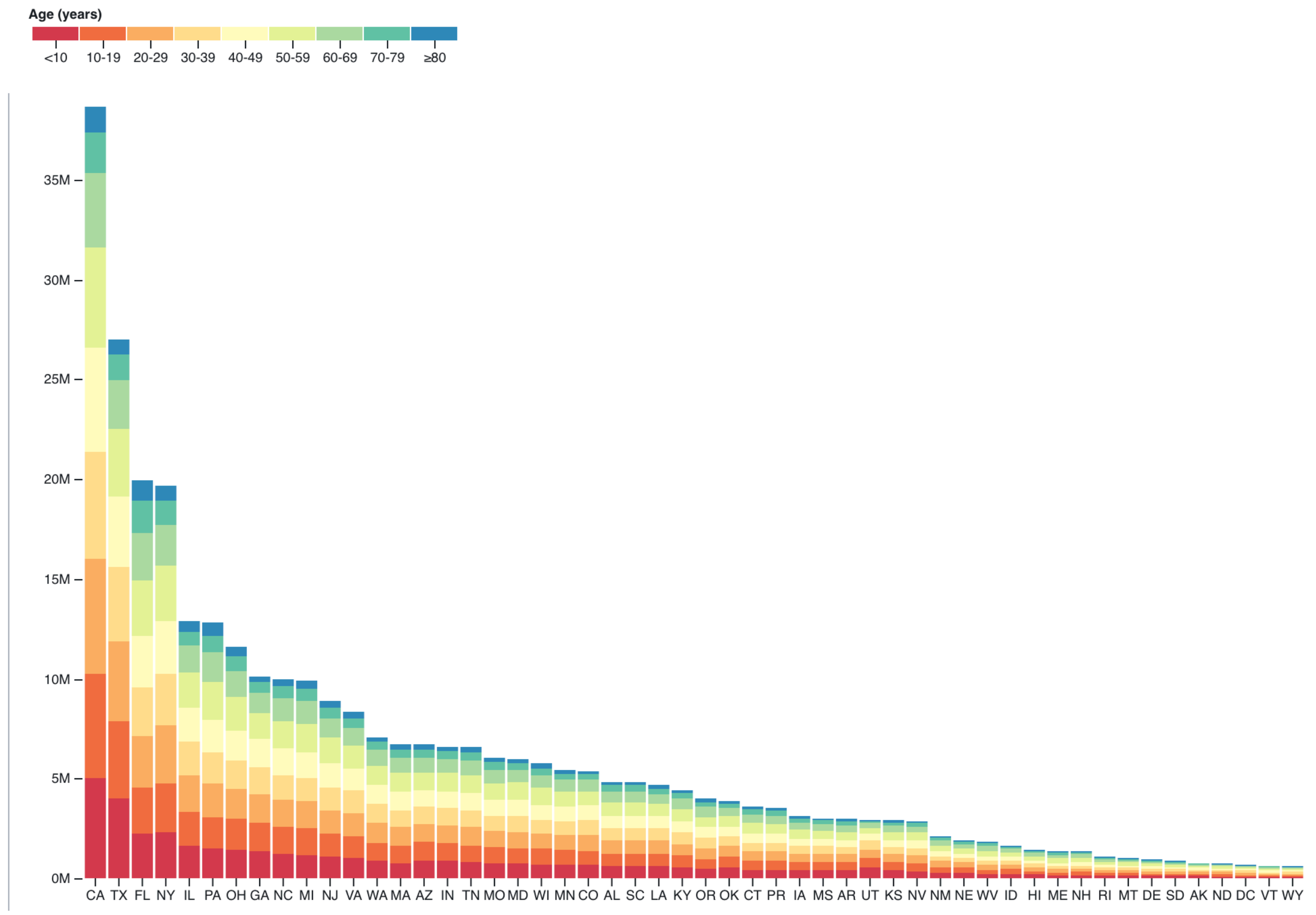
d3.area()

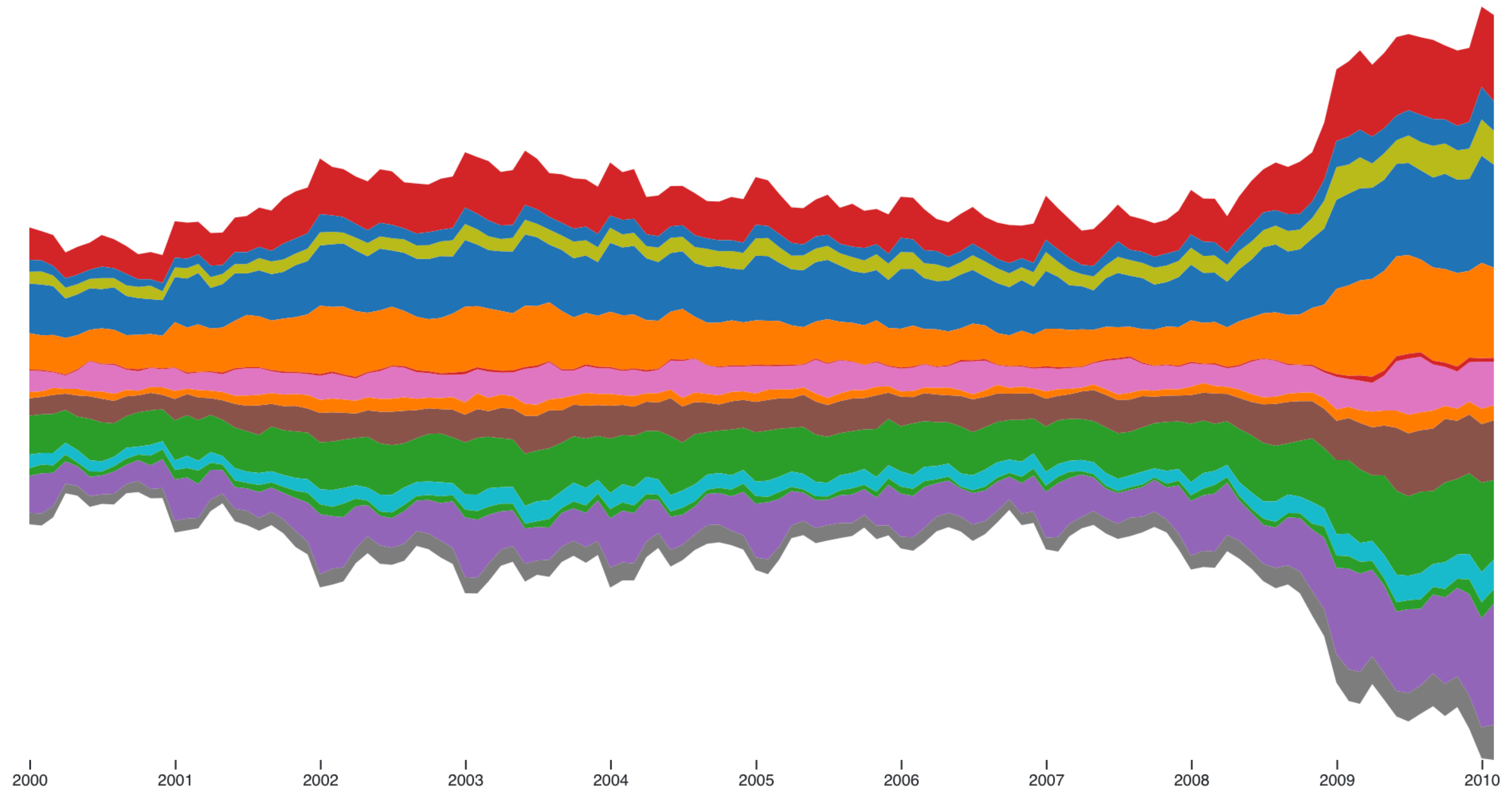


```
area = d3.area()
    .curve(curve)
    .x(d => x(d.date))
    .y0(y(0))
    .y1(d => y(d.value))
```

https://observablehq.com/@d3/area-chart

d3.stack()



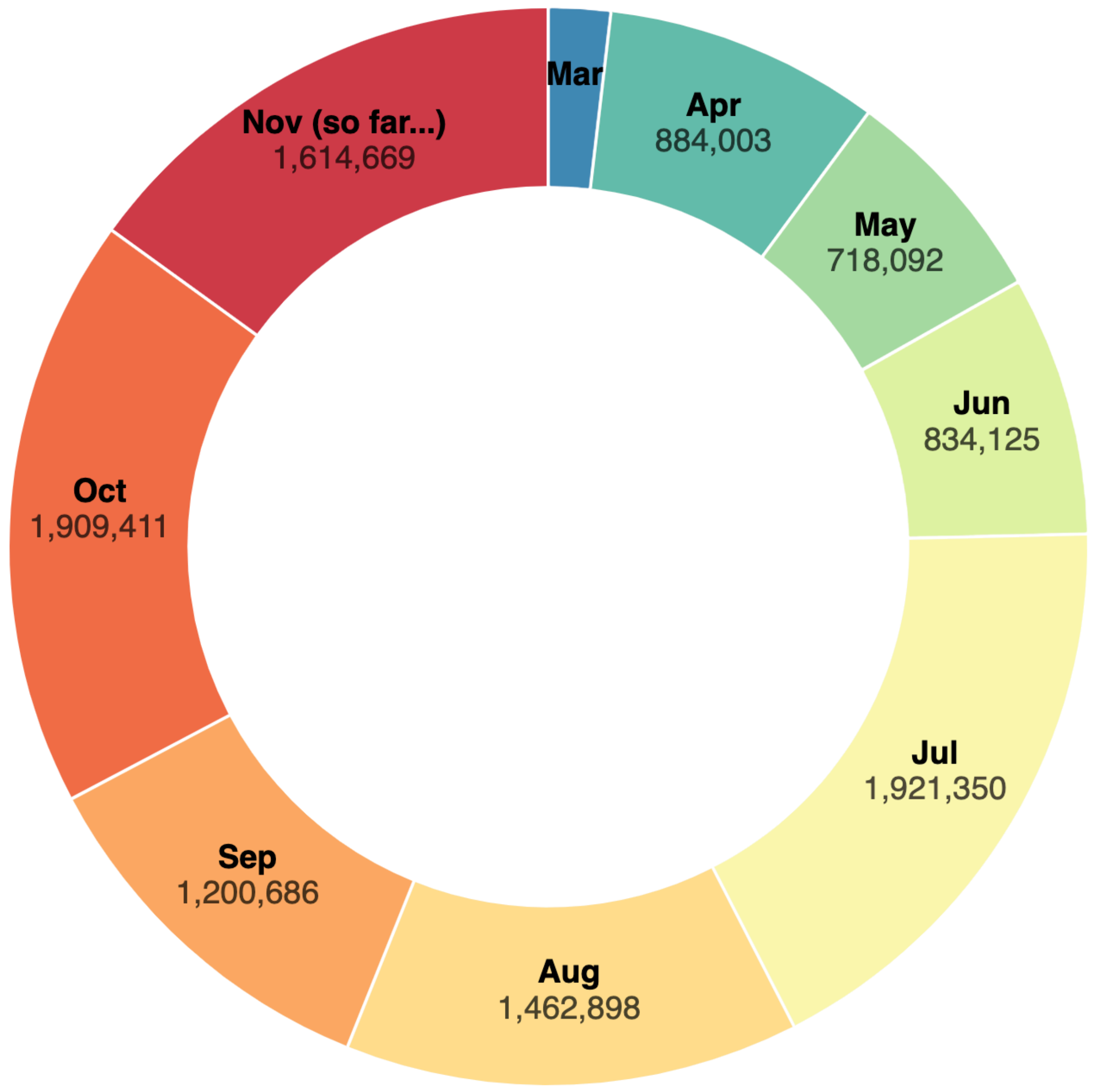


https://observablehq.com/@d3/stacked-bar-chart

https://observablehq.com/@d3/streamgraph-transitions

https://observablehq.com/@d3/streamgraph

d3.pie()



https://github.com/d3/d3-shape#pie

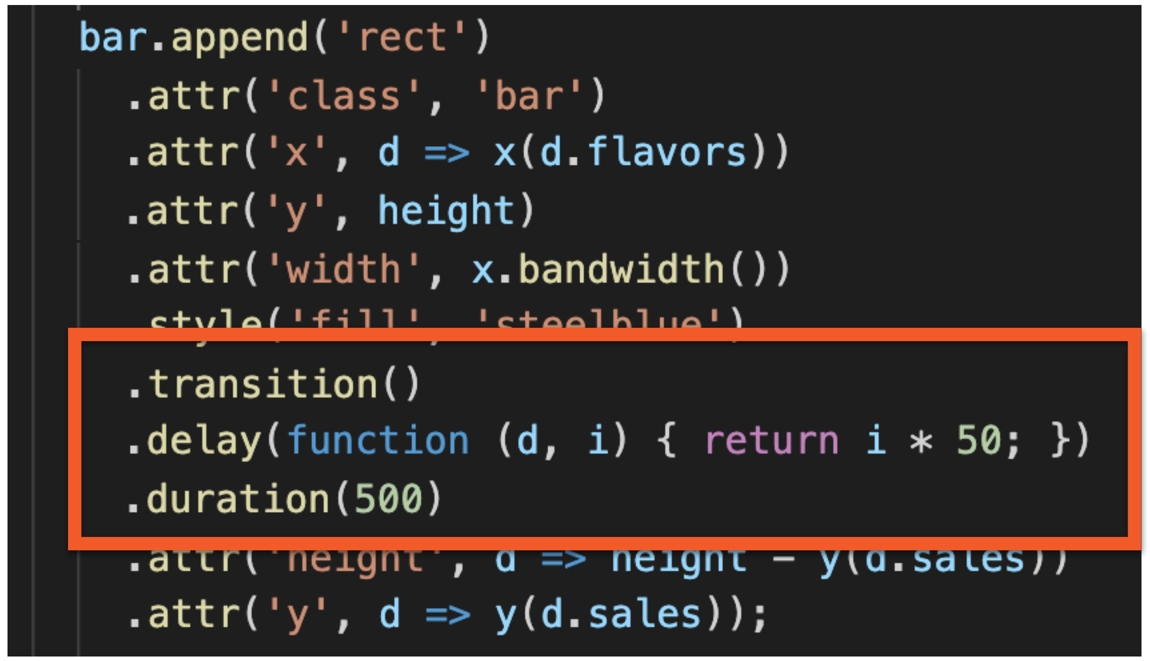
Refactoring Code

Scope

Scope determines the accessibility of variables.

Animation

transition()



```
bar.append('rect')
  .attr('class', 'bar')
  .attr('x', d => x(d.flavors))
  .attr('y', height)
  .attr('width', x.bandwidth())
  .style('fill', 'steelblue')
  .transition()
  .delay(function (d, i) { return i * 50; })
  .duration(500)
  .attr('height', d => height - y(d.sales))
  .attr('y', d => y(d.sales));
```

Modifying Bar Chart

(1)

Refactor code

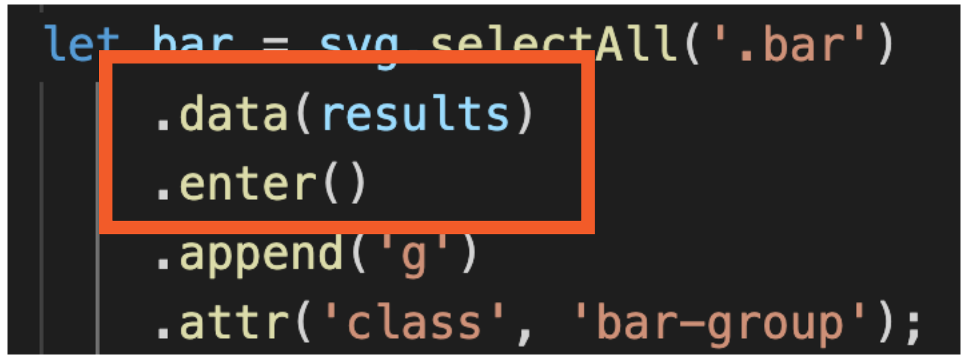
(2)

Implement join()

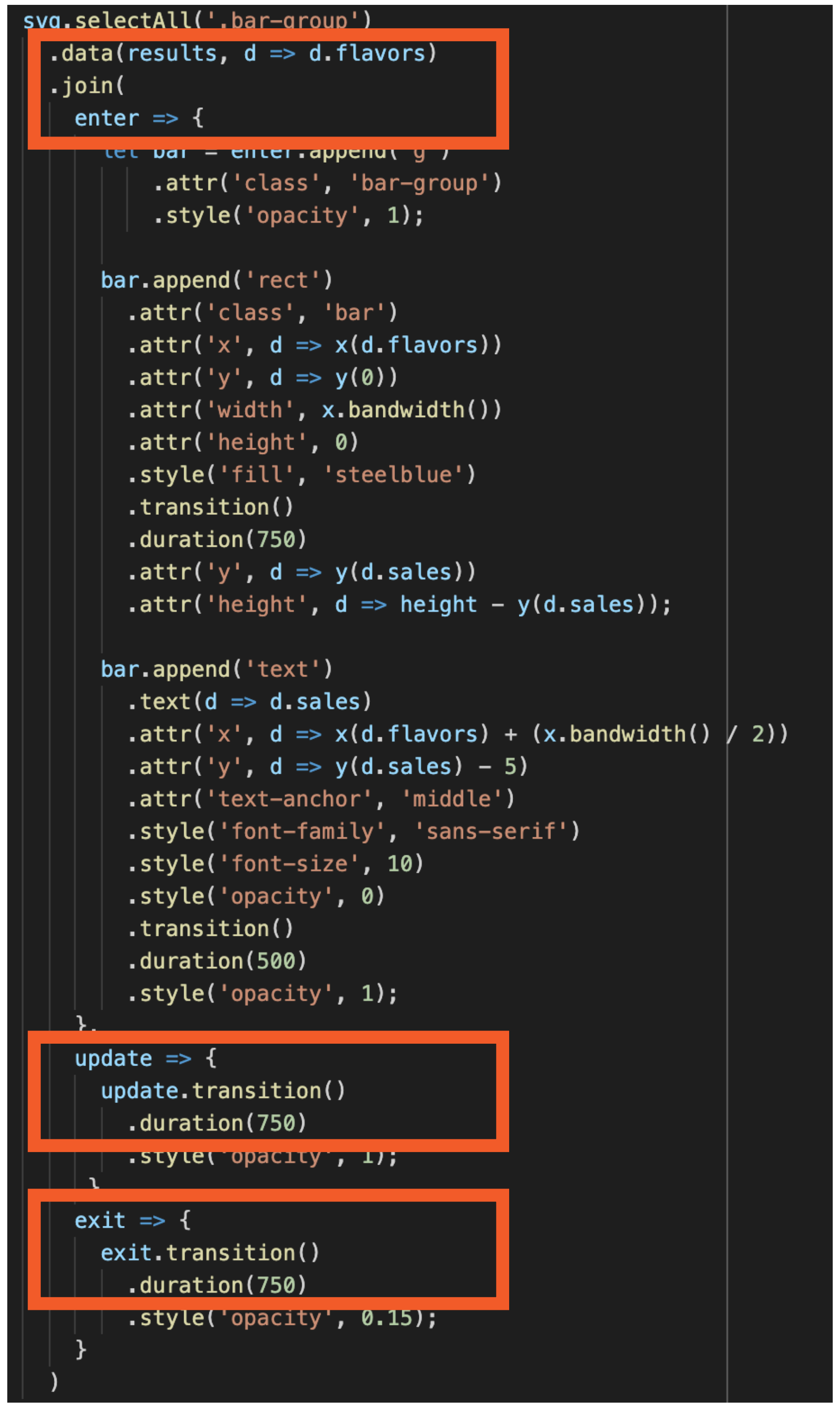
(3)

Hook up buttons

Data join versus selection join



```
let bar = svg.selectAll('.bar')
    .data(results)
    .enter()
    .append('g')
    .attr('class', 'bar-group');
```



```
svg.selectAll('.bar-group')
  .data(results, d => d.flavors)
  .join(
    enter => {
      let bar = enter.append('g')
          .attr('class', 'bar-group')
          .style('opacity', 1);

      bar.append('rect')
        .attr('class', 'bar')
        .attr('x', d => x(d.flavors))
        .attr('y', d => y(0))
        .attr('width', x.bandwidth())
        .attr('height', 0)
        .style('fill', 'steelblue')
        .transition()
        .duration(750)
        .attr('y', d => y(d.sales))
        .attr('height', d => height - y(d.sales));

      bar.append('text')
        .text(d => d.sales)
        .attr('x', d => x(d.flavors) + (x.bandwidth() / 2))
        .attr('y', d => y(d.sales) - 5)
        .attr('text-anchor', 'middle')
        .style('font-family', 'sans-serif')
        .style('font-size', 10)
        .style('opacity', 0)
        .transition()
        .duration(500)
        .style('opacity', 1);
    },
    update => {
      update.transition()
        .duration(750)
        .style('opacity', 1);
    },
    exit => {
      exit.transition()
        .duration(750)
        .style('opacity', 0.15);
    }
  )
```