



EECS E6893 Big Data Analytics

HW3: Data visualization -- Part II

Gudmundur Jonasson, gmj2122@columbia.edu

Introduction to D3.js

- What is D3.js?
- Why use D3.js?



Data-Driven
Documents

What is D3.js

- D3 stands for “Data-Driven Documents”
- D3.js is a free, open-source JavaScript library for producing dynamic, interactive data visualizations in web browsers
- Its low-level approach built on web standards offers unparalleled flexibility in authoring dynamic, data-driven graphics
- Visit <https://d3js.org/> for more tutorials!



Why use D3.js

- Dynamic and Interactive
- Directly binds data to DOM (Document Object Model)
- Extensive flexibility and control over visualization
- Large community and extensive documentation
- D3 Links:
 - Homepage: <https://d3js.org/>
 - Github: <https://github.com/d3/d3>

D3 Live Examples

- D3 Examples page: <https://observablehq.com/@d3/gallery>

Why Observable ▾ Product ▾ Enterprise Resources ▾ Pricing Search Sign in Sign up

D3
Bring your data to life.


Public 2 collections By Mike Bostock Edited Aug 23 Paused ISC 101 forks Importers 847 Likes

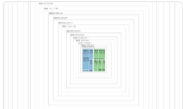
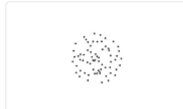
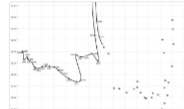
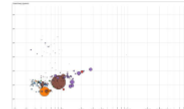


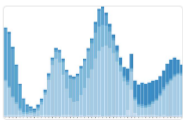
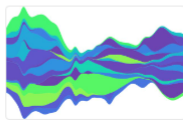


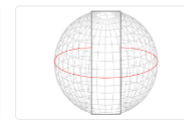

D3 gallery

Looking for a good D3 example? Here's a few (okay, 169...) to peruse.

Animation

D3's [data join](#), [interpolators](#), and [easings](#) enable flexible [animated transitions](#) between views while preserving [object constancy](#).




 Animated treemap	 Temporal force-directed gra...	 Connected scatterplot	 The wealth & health of natio...	 Scatterplot tour	 Bar chart race 2000
 Stacked-to-grouped bars	 Streamgraph transitions	 Smooth zooming	 Zoom to bounding box	 Orthographic to equirectang...	 World tour

Running D3

- What does D3 require to run?
 - A web browser
 - D3 source code
 - Valid HTML document
 - Server
- Most people probably have a web browser
- D3 source code can be added to any HTML file by including:
 - `<script src="https://d3js.org/d3.v6.min.js"></script>`
- Server:
 - Can use a remote setup
 - Host a local server

Starting D3

- “src” in script tag
- Sample code to get started. Save file as d3basic.html

```
<> d3basic.html >  html
1  <!DOCTYPE html>
2  <html>
3
4      <head>
5          |   <script src="https://d3js.org/d3.v4.js"></script>
6      </head>
7
8      <body>
9          |   <p>Hello</p>
10     </body>
11 </html>|
```

d3basic.html

- If we want our server to display that page in our browser we can go to: localhost:8000/d3basic.html
- We could also change d3basic.html to be called index.html, and our localhost:8000 will default to that page
- Key components of this file:
 - Valid HTML
 - Included script tag for D3 source

← → ↻ (i) File | / [redacted] /d3basic.html

Hello

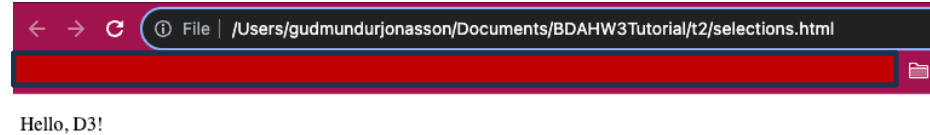
Core Concepts

- Selections
- Data Bindings
- Dynamic Properties
- Transitions

Selections

Selecting elements from the DOM

```
t2 > <> selections.html > html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>D3.js Selection Example</title>
6   <!-- Include D3.js library -->
7   <script src="https://d3js.org/d3.v6.min.js"></script>
8 </head>
9 <body>
10  <!-- D3.js script -->
11  <script>
12    // Use D3.js to select the body and append a paragraph with text
13    d3.select("body")
14      .append("p")
15      .text("Hello, D3!");
16  </script>
17 </body>
18 </html>
19
```



Data Binding

D3 allows binding of data to DOM elements

This is number 4

This is number 8

This is number 15

This is number 16

This is number 23

This is number 42

```
t2 > <> databinding.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>D3.js Data Binding Example</title>
6      <!-- Include D3.js library -->
7      <script src="https://d3js.org/d3.v6.min.js"></script>
8  </head>
9  <body>
10     <!-- D3.js script -->
11     <script>
12         d3.select("body")
13         .selectAll("p")
14         .data([4, 8, 15, 16, 23, 42])
15         .enter().append("p")
16         .text(d => "This is number " + d);
17     </script>
18 </body>
19 </html>
```

Dynamic Properties

Set properties of DOM elements based on data

```
t2 > <> dynamicproperties.html > html
~/Documents/BDAH3Tutorial/t2/
selections.html
3 </head>
4 <meta charset="UTF-8">
5 <title>D3.js Dynamic Properties Example</title>
6 <!-- Include D3.js library -->
7 <script src="https://d3js.org/d3.v6.min.js"></script>
8 </head>
9 <body>
10 <!-- D3.js script -->
11 <script>
12     d3.select("body")
13       .selectAll("p")
14       .data([4, 8, 15, 16, 23, 42])
15       .enter().append("p")
16       .text(d => "This is number " + d);
17
18     d3.select("body").selectAll("p")
19       .data([4, 8, 15, 16, 23, 42])
20       .style("font-size", d => d + "px");
21 </script>
22 </body>
23 </html>
```

This is number 8

This is number 15

This is number 16

This is number 23

This is number 42

Transitions

Add animations to visualizations

```
t2 > transitions.html > html > head > title
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>D3.js Transitions Example</title>
6   <!-- Include D3.js library -->
7   <script src="https://d3js.org/d3.v6.min.js"></script>
8 </head>
9 <body>
10 <!-- D3.js script -->
11 <script>
12   d3.select("body")
13     .selectAll("p")
14     .data([4, 8, 15, 16, 23, 42])
15     .enter().append("p")
16     .text(d => "This is number " + d);
17
18   d3.select("body").selectAll("p")
19     .data([4, 8, 15, 16, 23, 42])
20     .style("font-size", d => d + "px");
21
22   d3.select("body").selectAll("p")
23     .transition()
24     .duration(1750)
25     .style("color", "red");
26
27 </script>
28 </body>
29 </html>
```

This is number 8

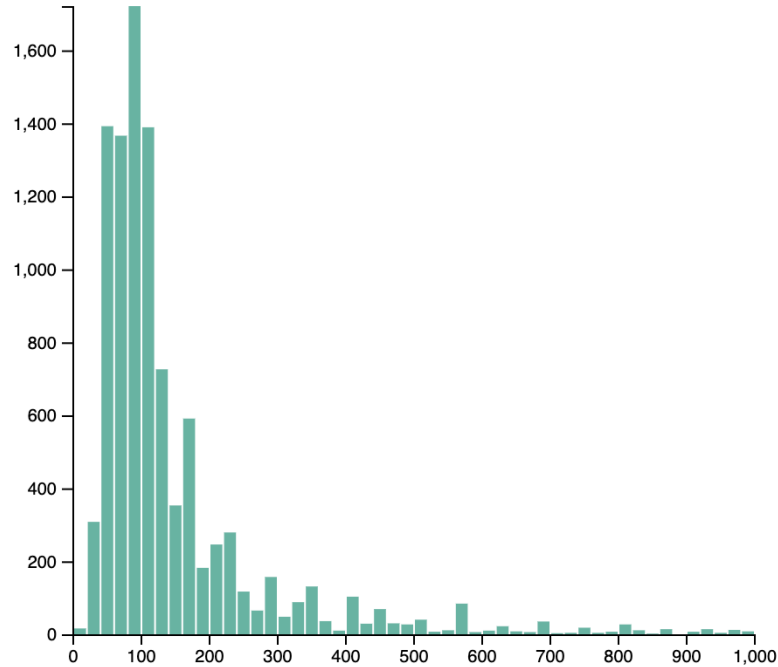
This is number 15

This is number 16

This is number 23

This is number 42

Histogram with interactive component example



bins

```
t2 > <> histogrambins.html > ...
1 <!DOCTYPE html>
2 <meta charset="utf-8">
3
4 <!-- Load d3.js -->
5 <script src="https://d3js.org/d3.v4.js"></script>
6
7 <!-- Create a div where the graph will take place -->
8 <div id="my_dataviz"></div>
9
10 <p>
11   <label># bins</label>
12   <input type="number" min="1" max="100" step="30" value="20" id="nBin">
13 </p>
14
15
16 <script>
17
18 // set the dimensions and margins of the graph
19 var margin = {top: 10, right: 30, bottom: 30, left: 40},
20   width = 460 - margin.left - margin.right,
21   height = 400 - margin.top - margin.bottom;
22
23 // append the svg object to the body of the page
24 var svg = d3.select("#my_dataviz")
25   .append("svg")
26   .attr("width", width + margin.left + margin.right)
27   .attr("height", height + margin.top + margin.bottom)
28   .append("g")
29   .attr("transform",
30     "translate(" + margin.left + "," + margin.top + ")");
31
32 // get the data
33 d3.csv("https://raw.githubusercontent.com/holtzy/data_to_viz/master/Example_dataset/1_OneNum.csv", function(data) {
34
35   // X axis: scale and draw:
36   var x = d3.scaleLinear()
37     .domain([0, 1000]) // can use this instead of 1000 to have the max of data: d3.max(data, function(d) { return +d.price })
38     .range([0, width]);
39   svg.append("g")
40     .attr("transform", "translate(0," + height + ")")
41     .call(d3.axisBottom(x));
42
43   // Y axis: initialization
44   var y = d3.scaleLinear()
45     .range([height, 0]);
46   var yAxis = svg.append("g")
47
```

```

48 // A function that builds the graph for a specific value of bin
49 function update(nBin) {
50     // set the parameters for the histogram
51     var histogram = d3.histogram()
52     .value(function(d) { return d.price; }) // I need to give the vector of value
53     .domain(x.domain()) // then the domain of the graphic
54     .thresholds(x.ticks(nBin)); // then the numbers of bins
55
56     // And apply this function to data to get the bins
57     var bins = histogram(data);
58
59     // Y axis: update now that we know the domain
60     y.domain([0, d3.max(bins, function(d) { return d.length; })]); // d3.hist has to be called before the Y axis obviously
61     yAxis
62     .transition()
63     .duration(1000)
64     .call(d3.axisLeft(y));
65     // Join the rect with the bins data
66     var u = svg.selectAll("rect")
67     .data(bins)
68     // Manage the existing bars and eventually the new ones:
69     u
70     .enter()
71     .append("rect") // Add a new rect for each new elements
72     .merge(u) // get the already existing elements as well
73     .transition() // and apply changes to all of them
74     .duration(1000)
75     .attr("x", 1)
76     .attr("transform", function(d) { return "translate(" + x(d.x0) + "," + y(d.length) + ")"; })
77     .attr("width", function(d) { return x(d.x1) - x(d.x0) - 1; })
78     .attr("height", function(d) { return height - y(d.length); })
79     .style("fill", "#69b3a2")
80     // If less bar in the new histogram, I delete the ones not in use anymore
81     u
82     .exit()
83     .remove()
84 }
85 // Initialize with 20 bins
86 update(20)
87
88 // Listen to the button -> update if user change it
89 d3.select("#nBin").on("input", function() {
90     update(+this.value);
91 });
92
93 });
94 </script>

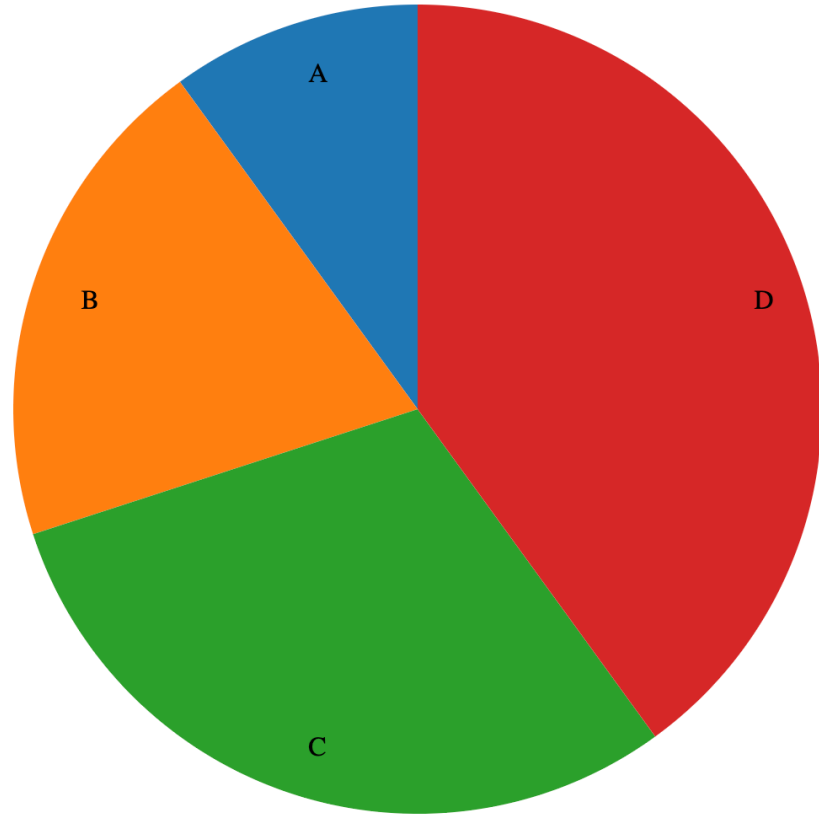
```


Table example with CSS style

Name	Age	Job
Alice	25	Engineer
Bob	30	Designer
Charlie	28	Teacher

```
t2 > <> table.html > <html > <head > <meta
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <script src="https://d3js.org/d3.v5.min.js"></script>
6   <style>
7     /* Table styles */
8     table {
9       border-collapse: collapse;
10      width: 100%;
11    }
12    th, td {
13      border: 1px solid black;
14      padding: 8px 12px;
15      text-align: left;
16    }
17    th {
18      background-color: #f2f2f2;
19    }
20  </style>
21 </head>
22 <body>
23   <table></table>
24   <script>
25     // Sample data
26     const data = [
27       { Name: "Alice", Age: 25, Job: "Engineer" },
28       { Name: "Bob", Age: 30, Job: "Designer" },
29       { Name: "Charlie", Age: 28, Job: "Teacher" }
30     ];
31     // Select the table
32     const table = d3.select("table");
33     // Append table headers
34     const thead = table.append("thead");
35     thead.append("tr")
36       .selectAll("th")
37       .data(Object.keys(data[0]))
38       .enter().append("th")
39       .text(d => d);
40     // Append table rows and cells
41     const tbody = table.append("tbody");
42     tbody.selectAll("tr")
43       .data(data)
44       .enter().append("tr")
45       .selectAll("td")
46       .data(d => Object.values(d))
47       .enter().append("td")
48       .text(d => d);
49   </script>
50 </body>
51 </html>
```

Pie Chart example



```
t2 > <> pie.html > <html > <head > <script
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <script src="https://d3js.org/d3.v5.min.js"></script>
6 </head>
7 <body>
8   <svg width="500" height="500"></svg>
9   <script>
10     // Sample data
11     const data = [
12       { label: "A", value: 10 },
13       { label: "B", value: 20 },
14       { label: "C", value: 30 },
15       { label: "D", value: 40 }
16     ];
17
18     // SVG setup
19     const width = 500;
20     const height = 500;
21     const radius = Math.min(width, height) / 2;
22     const svg = d3.select("svg")
23       .append("g")
24       .attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");
25
26     // Color scale
27     const color = d3.scaleOrdinal(d3.schemeCategory10);
28
29     // Pie layout
30     const pie = d3.pie().value(d => d.value);
31     const path = d3.arc().outerRadius(radius - 10).innerRadius(0);
32     const labelArc = d3.arc().outerRadius(radius - 40).innerRadius(radius - 40);
33
34     // Bind data, create pie chart slices
35     const g = svg.selectAll("arc")
36       .data(pie(data))
37       .enter().append("g")
38       .attr("class", "arc");
39
40     g.append("path")
41       .attr("d", path)
42       .style("fill", d => color(d.data.label));
43
44     g.append("text")
45       .attr("transform", d => "translate(" + labelArc.centroid(d) + ")")
46       .attr("dy", ".35em")
47       .text(d => d.data.label);
48
49   </script>
50 </body>
51 </html>
```

Graphical Analysis using D3

- Network analysis and network visualization are more common now with the growth of online social networks like Twitter and Facebook, as well as social media and linked data, all of which are commonly represented with network structures.
- In general, when dealing with networks you refer to the things being connected (like people) as nodes and the connections between them (such as being a friend on Facebook) as edges or links.
- Networks may also be referred to as graphs, because that's what they're called in mathematics.



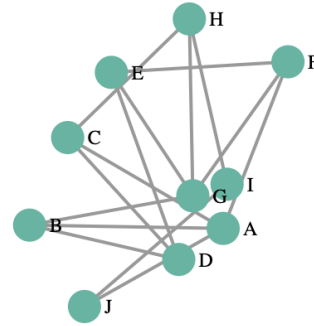
Graphical Analysis using D3

- D3 is particularly adept at creating Graphical visualizations due to its data-binding capability and built-in force simulations
- Real-world applications:
 - Social network analysis – friendships, interactions
 - Biological networks – protein interaction, genetic pathways
 - Infrastructure – internet routing, power grids

Force-directed Network diagrams and simulations in D3

- Automatically positions nodes
- Nodes repel each other, links act as springs
- Custom forces can be added
 - Centering, collision detection, etc.
- The force layout dynamically updates the positions of its elements to find the best fit, it does it continuously in real-time rather than as a preprocessing step before rendering

Force- directed diagram example




```

t2 > <> force.html > html > body > script > simulation.on("tick") callback
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Force-Directed Graph with D3.js</title>
6   <script src="https://d3js.org/d3.v5.min.js"></script>
7 </head>
8 <body>
9   <svg width="800" height="600"></svg>
10  <script>
11    // Sample data
12  const nodes = [
13    { id: "A" },
14    { id: "B" },
15    { id: "C" },
16    { id: "D" },
17    { id: "E" },
18    { id: "F" },
19    { id: "G" },
20    { id: "H" },
21    { id: "I" },
22    { id: "J" }
23  ];
24
25  const links = [
26    { source: "A", target: "B" },
27    { source: "A", target: "C" },
28    { source: "B", target: "D" },
29    { source: "C", target: "D" },
30    { source: "D", target: "E" },
31    { source: "E", target: "F" },
32    { source: "F", target: "G" },
33    { source: "G", target: "H" },
34    { source: "H", target: "I" },
35    { source: "I", target: "J" },
36    { source: "J", target: "A" },
37    { source: "E", target: "G" },
38    { source: "C", target: "H" },
39    { source: "A", target: "F" },
40    { source: "B", target: "G" }
41  ];
42
43  const svg = d3.select("svg");
44  const width = +svg.attr("width");
45  const height = +svg.attr("height");
46

```

```

47  const simulation = d3.forceSimulation(nodes)
48    .force("link", d3.forceLink(links).id(d => d.id).distance(100))
49    .force("charge", d3.forceManyBody())
50    .force("center", d3.forceCenter(width / 2, height / 2));
51
52  const link = svg.append("g")
53    .selectAll("line")
54    .data(links)
55    .enter().append("line")
56    .attr("stroke", "#999")
57    .attr("stroke-width", 2);
58
59  const node = svg.append("g")
60    .selectAll("circle")
61    .data(nodes)
62    .enter().append("circle")
63    .attr("r", 10)
64    .attr("fill", "#69b3a2")
65    .call(d3.drag()
66      .on("start", dragstarted)
67      .on("drag", dragged)
68      .on("end", dragended));
69
70  const label = svg.append("g")
71    .selectAll("text")
72    .data(nodes)
73    .enter().append("text")
74    .attr("font-size", 12)
75    .attr("dx", 12)
76    .attr("dy", ".35em")
77    .text(d => d.id);
78
79  simulation.on("tick", () => {
80    link.attr("x1", d => d.source.x)
81      .attr("y1", d => d.source.y)
82      .attr("x2", d => d.target.x)
83      .attr("y2", d => d.target.y);
84
85    node.attr("cx", d => d.x)
86      .attr("cy", d => d.y);
87
88    label.attr("x", d => d.x)
89      .attr("y", d => d.y);
90  });

```

```
91
92 function dragstarted(d) {
93     if (!d3.event.active) simulation.alphaTarget(0.3).restart();
94     d.fx = d.x;
95     d.fy = d.y;
96 }
97
98 function dragged(d) {
99     d.fx = d3.event.x;
100    d.fy = d3.event.y;
101 }
102
103 function dragended(d) {
104     if (!d3.event.active) simulation.alphaTarget(0);
105     d.fx = null;
106     d.fy = null;
107 }
108
109
110     </script>
111 </body>
112 </html>
```

Explore more examples

- <https://d3js.org/>
- <https://observablehq.com/@d3/gallery>
- <https://github.com/d3/d3>
- https://www.tutorialspoint.com/d3js/d3js_working_example.htm

THANK YOU