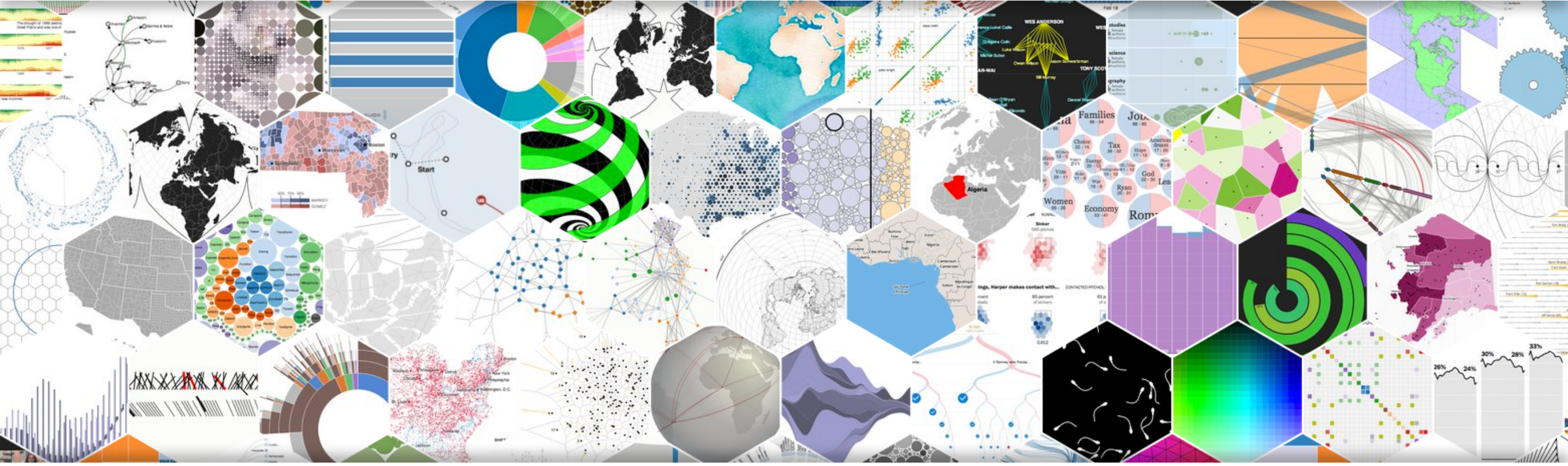
Data Visualization

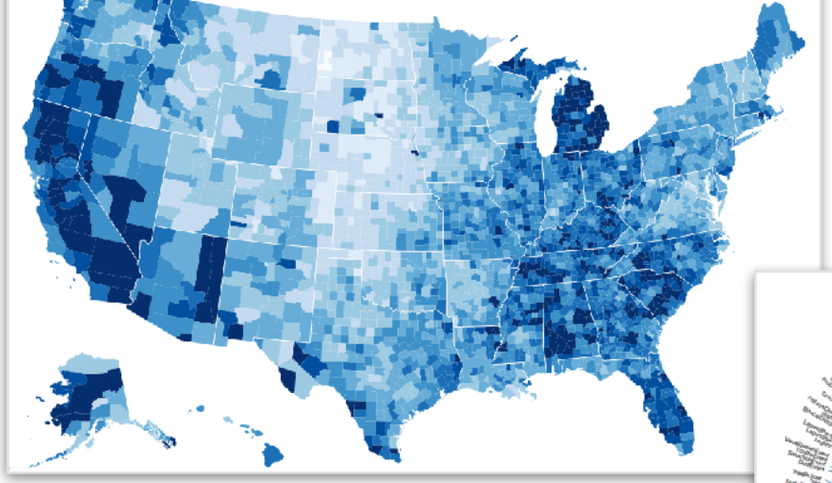
M2ID - Session 02 - exercices
Introduction to D3

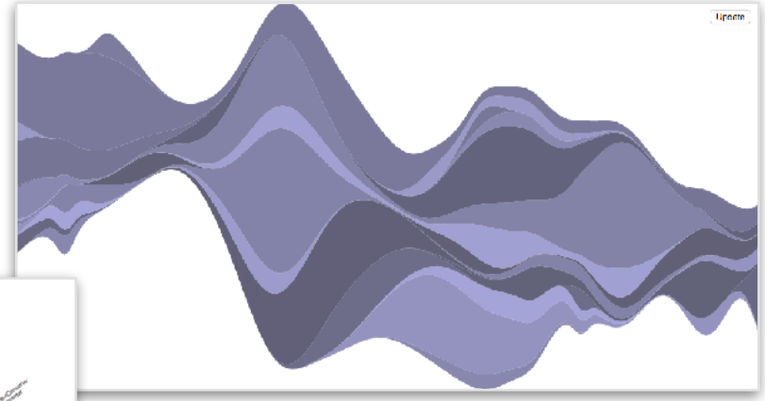
Dauphine | PSL UNIVERSITÉ PARIS

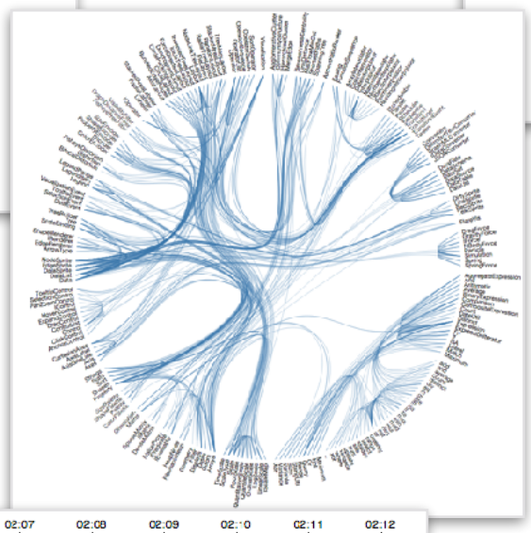
Inria

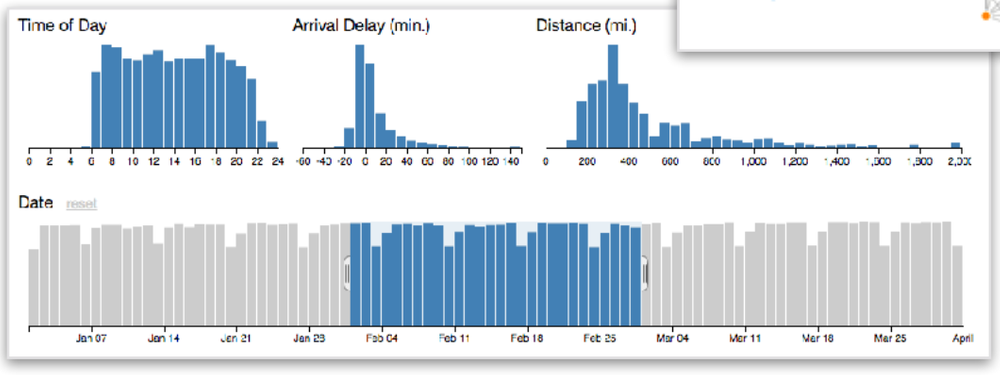
emmanuel.pietriga@inria.fr

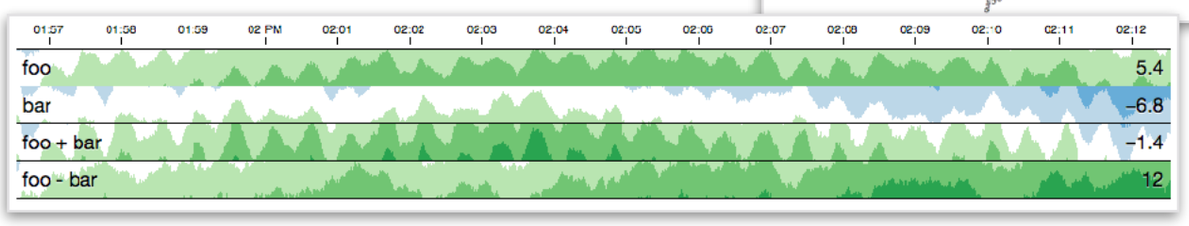












http://d3js.org

```
var circle = document.createElementNS(ctx.SVG_NS, "circle");
circle.setAttribute("cx", pos[0]);
circle.setAttribute("cy", pos[1]);
circle.setAttribute("r", ctx.GLYPH_SIZE/2.0);
circle.setAttribute("fill", color);
```

```
var circleGenerator = d3.symbol().type(d3.symbolCircle)
                        .size(6);
d3.selectAll("path")
  .data(someData)
  .enter()
  .append("path")
  .attr("d", circleGenerator());
```

```
"data": {
    "url": "exoplanet.eu_catalog.csv",
},
"mark": "point",
"encoding": {
  "x": {"field": "star_mass", "type": "quantitative"}},
  "y": {"field": "mass", "type": "quantitative"}}
}
```

DOM+JS

D3

Vega-lite

Level of abstraction

What D3 does

- Loading data into the browser's memory
- Binding data to elements within the document, creating new elements as needed
- Transforming those elements by interpreting each element's bound datum and setting its visual properties accordingly
- Transitioning elements between states in response to user input
- API reference: `https://github.com/d3/d3/blob/master/API.md`

- D3 uses CSS-style selectors to identify elements on which to operate:

```
// select all <circle> elements in the document tree
d3.selectAll("circle")

// select all elements with class 'bar' regardless of their tag name
d3.selectAll(".bar")

// select the unique element whose ID is 'foo'
d3.select("#foo")

// select all lines that are children of the <g> element whose ID is 'foo'
d3.select("g#foo").selectAll("line")
```

D3 - Chain Syntax

```
var body = d3.select("body");
var aDiv = body.append("div");
aDiv.text("Some text.");

// can be written more concisely as:
d3.select("body").append("div").text("Some text.");
```

D3 - Loading Data

The old way:

```
d3.json(dataURL, function(error, jsonData){
        if (error){
            console.log(error);
        }
        else {
            doSomethingWith(jsonData);
        }
    }
);
```

Calls to these methods are asynchronous

The new way, using Promises:

```
d3.json(dataURL).then(
    function(jsonData){doSomethingWith(jsonData);}
)
.catch(
    function(error){console.log(error);}
);
```

Use **`d3.csv()`** or **`d3.tsv()`** for various types of CSV files

D3 - Binding data to DOM elements

```
var dataset = [43, 9, 100, 99, 56];

d3.select("body").selectAll("div")
  .data(dataset)
  .enter()
  .append("div")
  .text(function(d){return d;});
```

```
<body>
    <div>43</div>
    <div>9</div>
    <div>100</div>
    <div>99</div>
    <div>56</div>
</body>
```

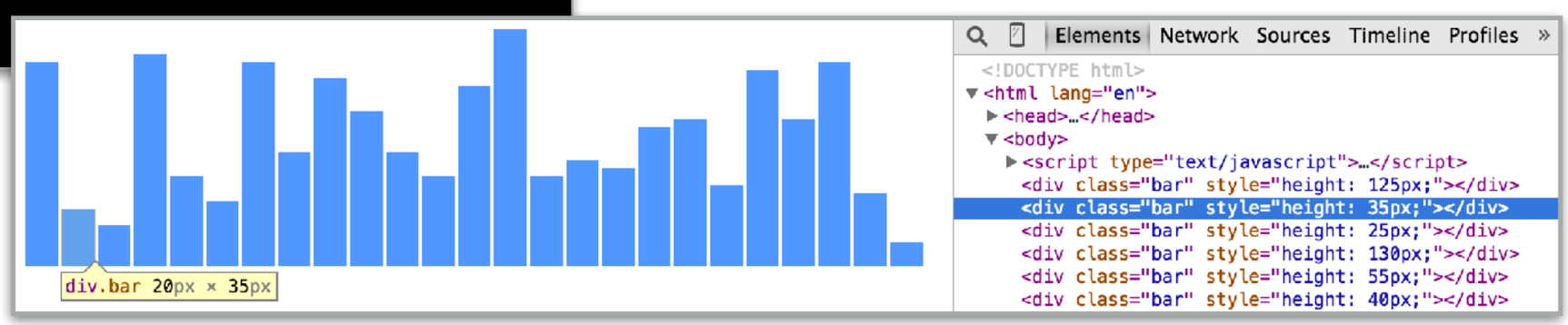
D3 - `attr()` and `style()`

- `attr()` sets an HTML attribute on the current DOM selection
- `style()` sets a CSS property on the current DOM selection

```
        <style type="text/css" media="all">
div.bar {
    display: inline-block;
    width:20px;
    height:75px;
    background-color: #559aff;
    margin: 0 1px;
}
        </style>
    </head>

    <body>
        <script type="text/javascript">
            var dataset = [25, 7, 5, 26, 11, 8, 25, 14, 23, 19,
                14, 11, 22, 29, 11, 13, 12, 17, 18, 10,
                24, 18, 25, 9, 3];

            d3.select("body").selectAll("div")
              .data(dataset)
              .enter()
              .append("div")
              .attr("class", "bar")
              .style("height", function(d){
                  return (5*d) + "px";
              });
        </script>
```



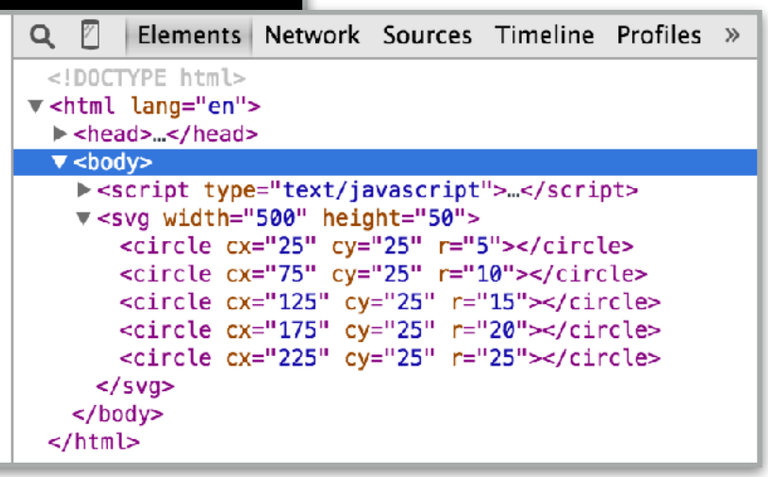
D3 - Drawing with SVG

```
var dataset = [5, 10, 15, 20, 25];
var w = 500;
var h = 50;

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

var circles = svg.selectAll("circle")
                 .data(dataset)
                 .enter()
                 .append("circle");

circles.attr("cx", function(d, i){return i*50 + 25;})
       .attr("cy", h/2)
       .attr("r", function(d){return d;});
```



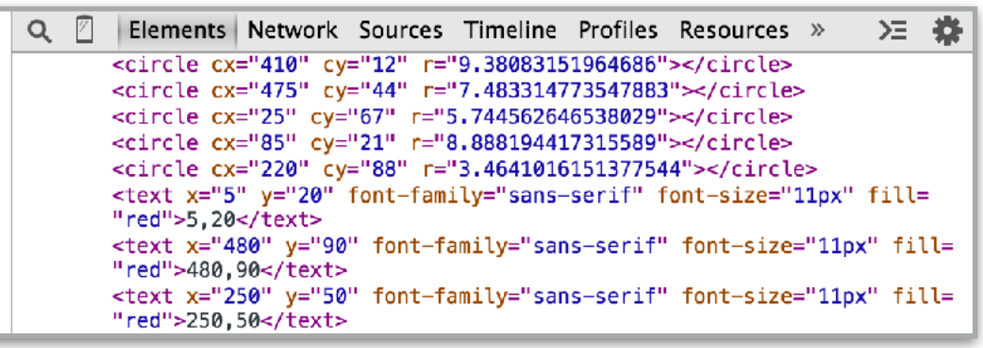
D3 - Drawing a slightly more complex dataset

```
var w = 500;
var h = 100;
var dataset = [[5, 20], [480, 90], [250, 50], [100, 33], [330, 95]
               [410, 12], [475, 44], [25, 67], [85, 21], [220, 88]];

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

svg.selectAll("circle")
   .data(dataset)
   .enter()
   .append("circle")
   .attr("cx", function(d){return d[0];})
   .attr("cy", function(d){return d[1];})
   .attr("r", function(d){return Math.sqrt(h - d[1]);});

svg.selectAll("text")
   .data(dataset)
   .enter()
   .append("text")
   .text(function(d){return d[0] + "," + d[1];})
   .attr("x", function(d){return d[0];})
   .attr("y", function(d){return d[1];})
   .attr("font-family", "sans-serif")
   .attr("font-size", "11px")
   .attr("fill", "red");
```



D3 - Defining a symbol and using it

- Instead of tediously drawing shapes with SVG elements,
  use D3 symbols for circles, squares, triangles, crosses, *etc*.

```
var circleGenerator = d3.symbol().type(d3.symbolCircle)
                        .size(6);

d3.selectAll("path")
  .data(someData)
  .enter()
  .append("path")
  .attr("d", circleGenerator());
```

- The above symbols would be positioned in `(0,0)` by default.
  Use affine transforms to move them to the write place.

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

https://github.com/d3/d3-shape/blob/master/README.md#symbol

- Syntactic sugar for strings:

```
(d) => (`translate(${d.x}, ${d.x})`)
```

D3 - Scales

- Scales are extremely useful, and can be applied to many cases:

```
var x = d3.scaleLinear()
          .domain([10, 130])
          .range([0, 960]);

x(20); // 80
x(50); // 320
```

```
var color = d3.scaleLinear()
              .domain([10, 100])
              .range(["brown", "steelblue"]);

color(20); // "#9a3439"
color(50); // "#7b5167"
```

https://github.com/d3/d3-scale/blob/master/README.md

★ In particular, pay attention to the interpolation method when creating a color mapping.

https://github.com/d3/d3-scale/blob/master/README.md#continuous_interpolate

- Useful functions to set a scale's domain: `d3.min(), d3.max(), d3.extent(), …`

https://github.com/d3/d3-array/blob/master/README.md