INF554 lab 7

Animated transitions

Dr. Luciano Nocera
Associate Director, Integrated Media Systems Center (IMSC)

USC Viterbi
School of Engineering
Integrated Media Systems Center

Catalina (new MacOS)

Reinstall xcode tools for git to work in VSCode by running in the terminal:

```
xcode-select --install
```

In your A7 repository create an ex1.html.

- Use data from your A1 for 10 countries and 1 year and load as JSON with d3:

```
data = [{key: "United States", value: 0.4},  //example data object after load
 {key: "Turkey", value: 0.6},
 {key: "France", value: 0.2},
 {key: "Italy", value: 0.1},
 {key: "Spain", value: 0.3},
 {key: "Algeria", value: 0.9},
 {key: "Mexico", value: 0.4},
 {key: "Brazil", value: 0.3},
 {key: "China", value: 0.1},
 {key: "Japan", value: 0.8}]
```

- Use d3 and `JSON.stringify` to print in the page:
  1. A copy of the array (use `array.slice`)
  2. The min key (use `d3.min`)
  3. The max value (use `d3.max`)
  4. Array sorted in ascending order by key (use `array.sort`, `d3.ascending`)
  5. Array sorted in descending order by value (use `array.sort`, `d3.descending`)
  6. Array containing the top 5 items sorted by value (use `array.slice`)
  7. Array with a label key as shown (use `array.map`)

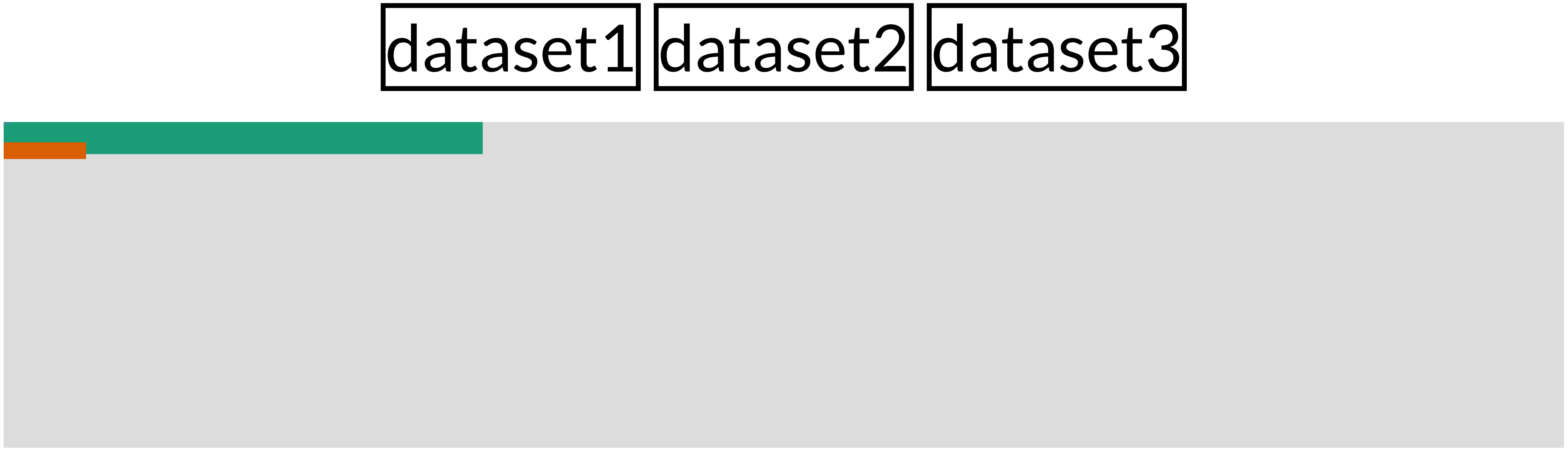
```
data = [{key: "United States", value: 0.4, label: "United States (0.4)"},
 {key: "Turkey", value: 0.6, label: "Turkey (0.6)"},
 {key: "France", value: 0.2, label: "France (0.2)"},
 {key: "Italy", value: 0.1, label: "Italy (0.1)"},
 {key: "Spain", value: 0.3, label: "Spain (0.3)"},
 {key: "Algeria", value: 0.9, label: "Algeria (0.9)"},
 {key: "Mexico", value: 0.4, label: "Mexico (0.4)"},
 {key: "Brazil", value: 0.3, label: "Brazil (0.3)"},
 {key: "China", value: 0.1, label: "China (0.1)"},
 {key: "Japan", value: 0.8, label: "Japan (0.8)"}]
```

```
// ////////////////////////////////////////////////// EXAMPLES WITH ARRAY OF NUMBERS
var array0 = [4, 1, 16, 9];
JSON.stringify(array0)  //prints "[4,1,16,9]"

//slice(start, end): slices array from start index to end index excluded
var array1 = array0.slice(0);  //make a copy: array1 = [4, 1, 16, 9]
array1 = array0.slice();  //make a copy: array1 = [4, 1, 16, 9]

var min = d3.min(array0);  //min = 1
var max = d3.max(array0);  //max = 16

array2 = array0;  //make a copy: array2 = [4, 1, 16, 9]

//array.sort([compareFunction]) sorts in place!
array2.sort();  //default sorts according to Unicode code! array2 = [1, 16, 4, 9]

array2.sort(d3.ascending);  //sorts in place! array2 = [1, 4, 9, 16]
array2.sort(d3.descending);  //sorts in place! array2 = [16, 9, 4, 1]

//array.splice(start, deleteCount [,items...]) Splices in place! Returns array of deleted!
array2 = array0;  //array2 = [16, 9, 4, 1]
array2.splice(1, 0, 5);  //insert 5 at index 1, other = [16, 5, 9, 4, 1]
array2.splice(1, 2);  //delete 2 elements starting at index 1, array2 = [16, 4, 1]

//array.map(function callback(currentValue[, index[, array]]) {...}) maps function to each value in array
var array3 = array0.map(function(d) { return d * 3; });  //array3 = [48, 12, 3]
array3 = array0.map(d => d * 3);  //same with ES6 syntax

// ////////////////////////////////////////////////// EXAMPLES WITH ARRAY OF OBJECTS
var data = [{key: 'first', val: 1}, {key: 'second', val: 2}];

var min = d3.min(data, function (d) { return d.val; });  //min = 4
var max = d3.max(data, function (d) { return d.key; });  //max = 'second'

var sorted = data.sort(function(a, b) {
    return d3.descending(a.key, b.key);  //sort alphabetically Z->A by key
}); //sorted = [{key: 'second', val: 2}, {key: 'first', val: 1}]7

var other = data.map(function(d) { return { key: d.key, val: d.val , date: Date.now() }; }); //return object
other = data.map(function(d) { d.date = Date.now(); return d; }); //same adding key first and returning object
other = data.map(d => { d.date = Date.now(); return d; }); //same as above with ES6 notation
JSON.stringify(other); //"[{"key":"second","val":2,"date":1570724664915},{"key":"first","val":1,"date":1570724664915}]"

other.splice(2, 0, {key: 'third', val: '3', date: Date.now()});  //insert object at index 2, other = [{...}, {...}, {...}
other.splice(1, 2);  //delete 2 elements starting at index 1, other = [{"key":"second","val":2,"date":1570724838178}]
```

Sample JavaScript and D3 functions to updated data arrays

In ex2.html:

- Implement the code shown in the next page
- Adapt the code to use the data you used in ex1.html
- Update the spans background color to indicate which one is selected

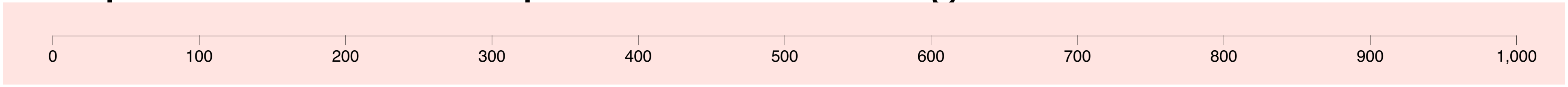


Rendering of starter code

```
<p>
  <span style="border: solid black;" id="dataset1">dataset1</span>
  <span style="border: solid black;" id="dataset2">dataset2</span>
  <span style="border: solid black;" id="dataset3">dataset3</span>
</p>
<svg width="100%" height="200" style="background-color: gainsboro" id="svg_ex2"></svg>
<script>
dataset1 = [{k: "A", v: 3}, {k: "B", v: 1}, {k: "C", v: 2}];
dataset2 = [{k: "A", v: 4}, {k: "B", v: 1}, {k: "X", v: 4}];
dataset3 = [{k: "A", v: 3}, {k: "C", v: 4}, {k: "D", v: 1}, {k: "E", v: 2.5}];

update(dataset1);

d3.select("#dataset1")
  .on("click", function () {
    update(dataset1);
});

d3.select("#dataset2")
  .on("click", function () {
    update(dataset2);
});

d3.select("#dataset3")
  .on("click", function () {
    update(dataset3);
});

d3.select("#svg_ex2")
  .on("click", function() {
    update(dataset1);  //reset
});

function update(data) {
  var svg = d3.select("#svg_ex2");
  var rects = svg.selectAll("rect")
    .data(data, function (d) { return d.k; });

  rects.exit()  //EXIT SELECTION -- here we decide to exit first
    .transition()
    .delay(1000)
    .duration(1000)
    .style("opacity", 0)
    .remove();

  var enter = rects.enter()  //ENTER
    .append("rect")  //add new rects
    .merge(rects)  // UPDATE + ENTER
    .transition()
    .duration(3000)
    .delay(function(d, i) { return i * 1000; })
    .attr("x", 0)
    .attr("y", function(d, i) { return i * 25; })
    .attr("width", function(d) { return d.v * 100; })
    .attr("height", 20)
    .attr("fill", function(d, i) { return d3.schemeDark2[i]; });
}
</script>
```

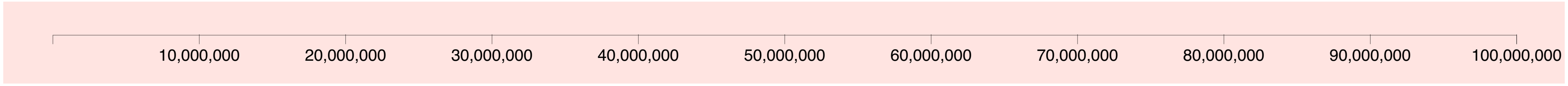
ex2.html starter code: data join with general update pattern

Complete the code to implement an axis range transition in ex3.html:



```
<svg id="svg30" width="960" height="50" style="background-color: mistyrose"></svg>
<script>
var x0 = d3.scaleLinear()
  .domain([..., ...])
  .range([0, 900]);

var axis0 = d3.axisBottom()
  .scale(x0);

var redo = false;
var svg = d3.select("#svg30");

svg.append("g")
  .attr("class", "axis0")
  .attr("transform", "translate(30,20)")
  .call(axis0);

svg.on("click", function () {
  x0.domain([0, redo ? 1000 : 500]);
  redo = !redo;
  d3.select(".axis0")
    .transition()
    .duration(1000)
    .call(axis0);
});
</script>
```

Start with the scaleLinear of ex3.html and complete the code to implement an axis scale type transition in ex4.html:

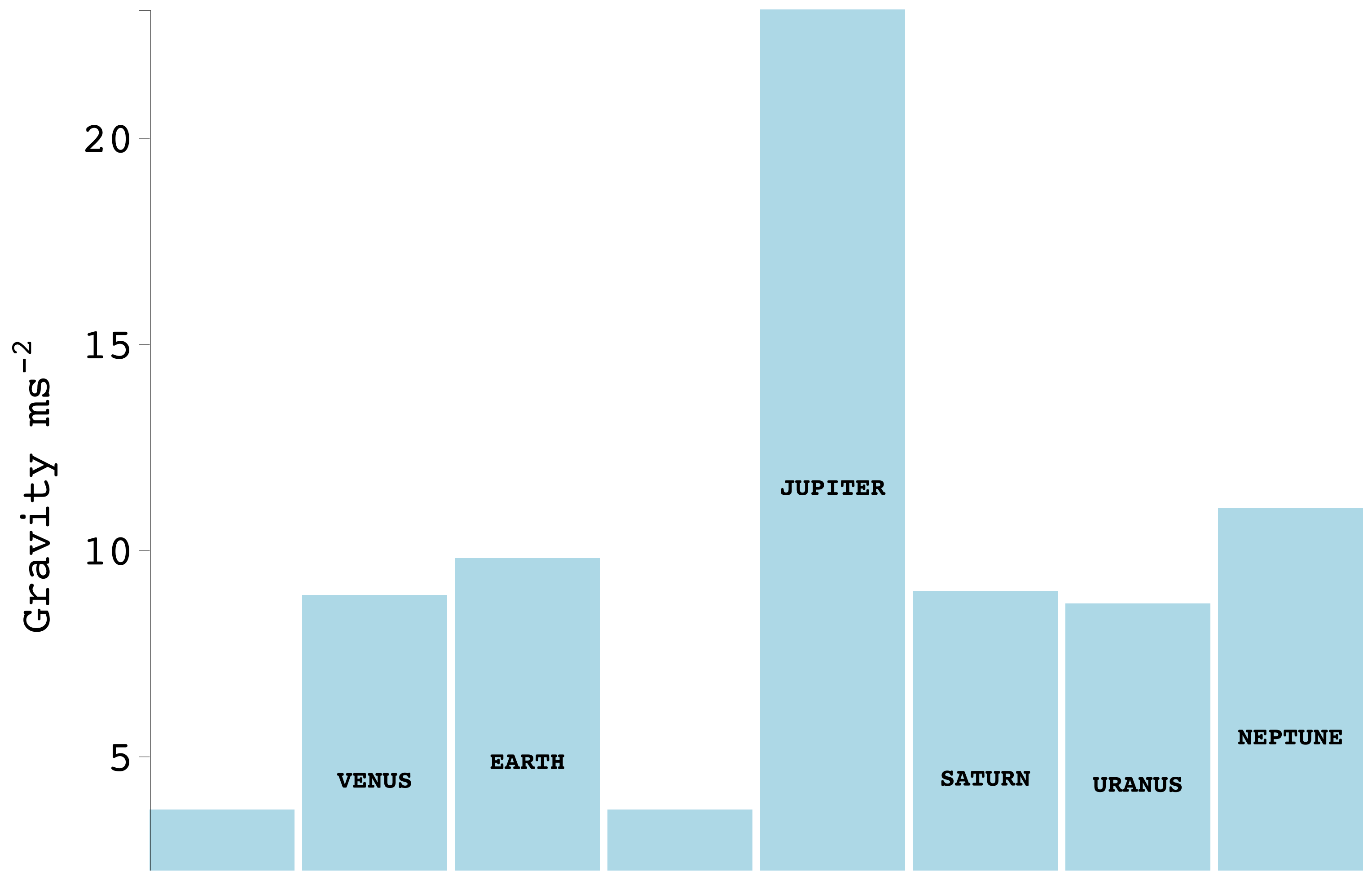


```
<svg id="svg31" width="960" height="50" style="background-color: mistyrose"></svg>
<script>
var x1 = d3.scaleLinear()
  .domain([..., ...])
  .range([0, 900]);

var x2 = d3.scaleLog()
  .domain([10, ...])  //why 10?
  .range([0, 900]);

var axis1 = d3.axisBottom()
  .scale(x1);

var redo = false;
var svg = d3.select("#svg31");

svg.append("g")
  .attr("class", "axis1")
  .attr("transform", "translate(30,20)")
  .call(axis1);

svg.on("click", function() {
  d3.select(".axis1")
    .transition()
    .duration(1000)
    .call(redo ? axis1.scale(x1) : axis1.scale(x2));
  redo = !redo;
});
</script>
```

In ex5.html:

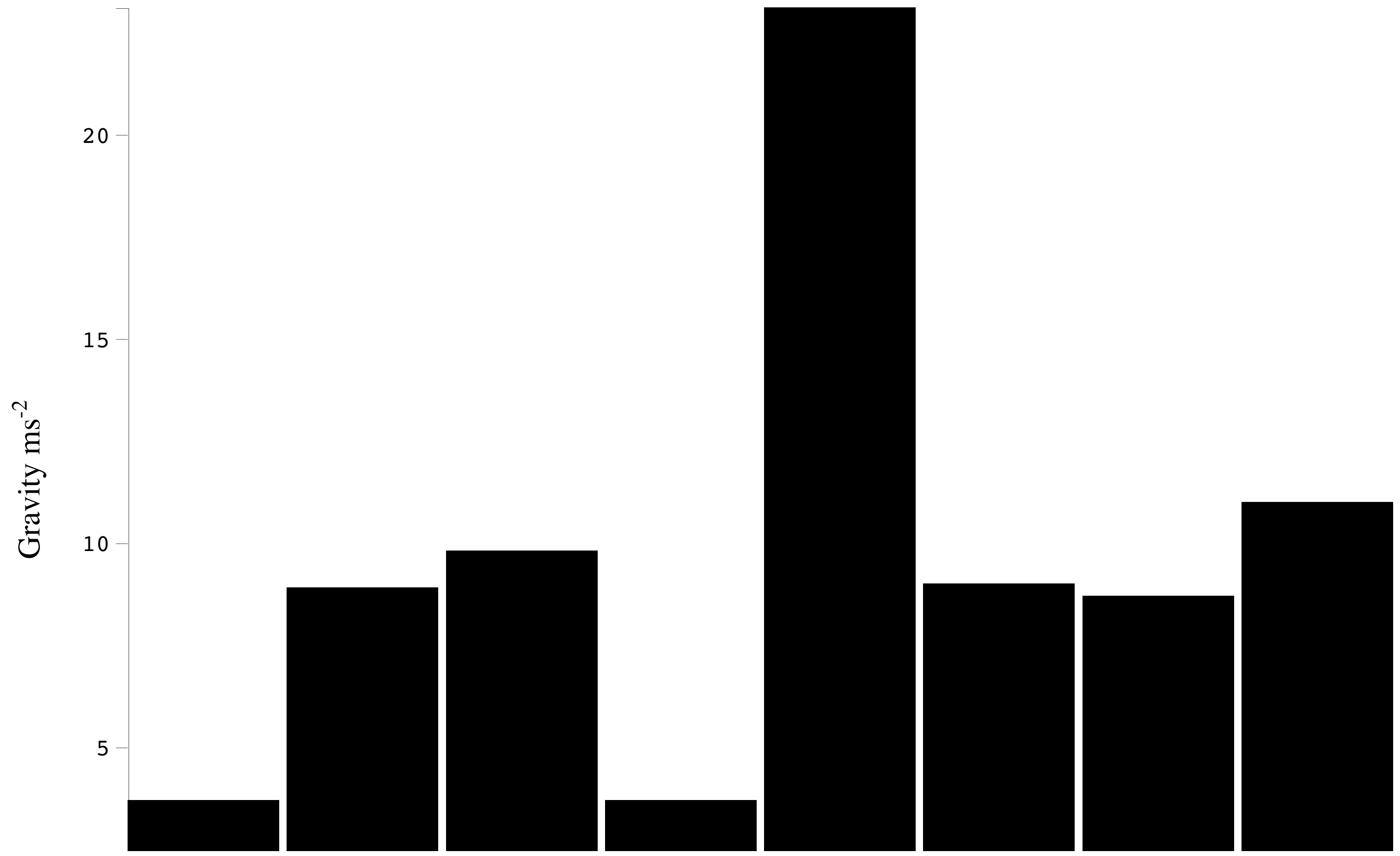
- Use the starter code provided in `planets_transition_1.html`
- Adapt the code to use d3 from node
- Separate the code of `ex5.html` into `ex6.html`, `ex6.css` and `ex6.js`

Order by: distance to sun, temperature, gravity ✓ Show Earth



Rendering of `planets_transition_1.html` starter code

Order by: distance to sun temperature gravity



`planets_transition_0.html` is another example you can use for the assignment with axis and bar transitions

Read the chapters from Murray before starting on the homework!

Assignments from '18 you can also look at... ex1 ex2 ex3