D3.js – Stack

张松海、张少魁、周文洋、蔡韵

数据可视化 – D3.js

清华大学 可视媒体研究中心

Stack?

```
const naiveData = [
  {month: new Date(2015, 0, 1), apples: 3840, bananas: 1920, cherries: 960, dates: 400},
  {month: new Date(2015, 1, 1), apples: 1600, bananas: 1440, cherries: 960, dates: 400},
  {month: new Date(2015, 2, 1), apples:  640, bananas:  960, cherries: 640, dates: 400},
  {month: new Date(2015, 3, 1), apples:  320, bananas:  480, cherries: 640, dates: 400}
];
```

- 不是数据结构的“Stack”（栈😀） Stack -> ‘堆叠’
- 本章节的主角：**d3.stack()**
- **本质上是D3.js提供的用于数据预处理的接口**（或功能、模块，接口可能略微狭义）；

```
▼0: Array(4)
  ▶0: (2) [0, 3840, data: {…}]
  ▶1: (2) [0, 1600, data: {…}]
  ▶2: (2) [0, 640, data: {…}]
  ▶3: (2) [0, 320, data: {…}]
    key: "apples"
    index: 0
    length: 4
  ▶__proto__: Array(0)
```

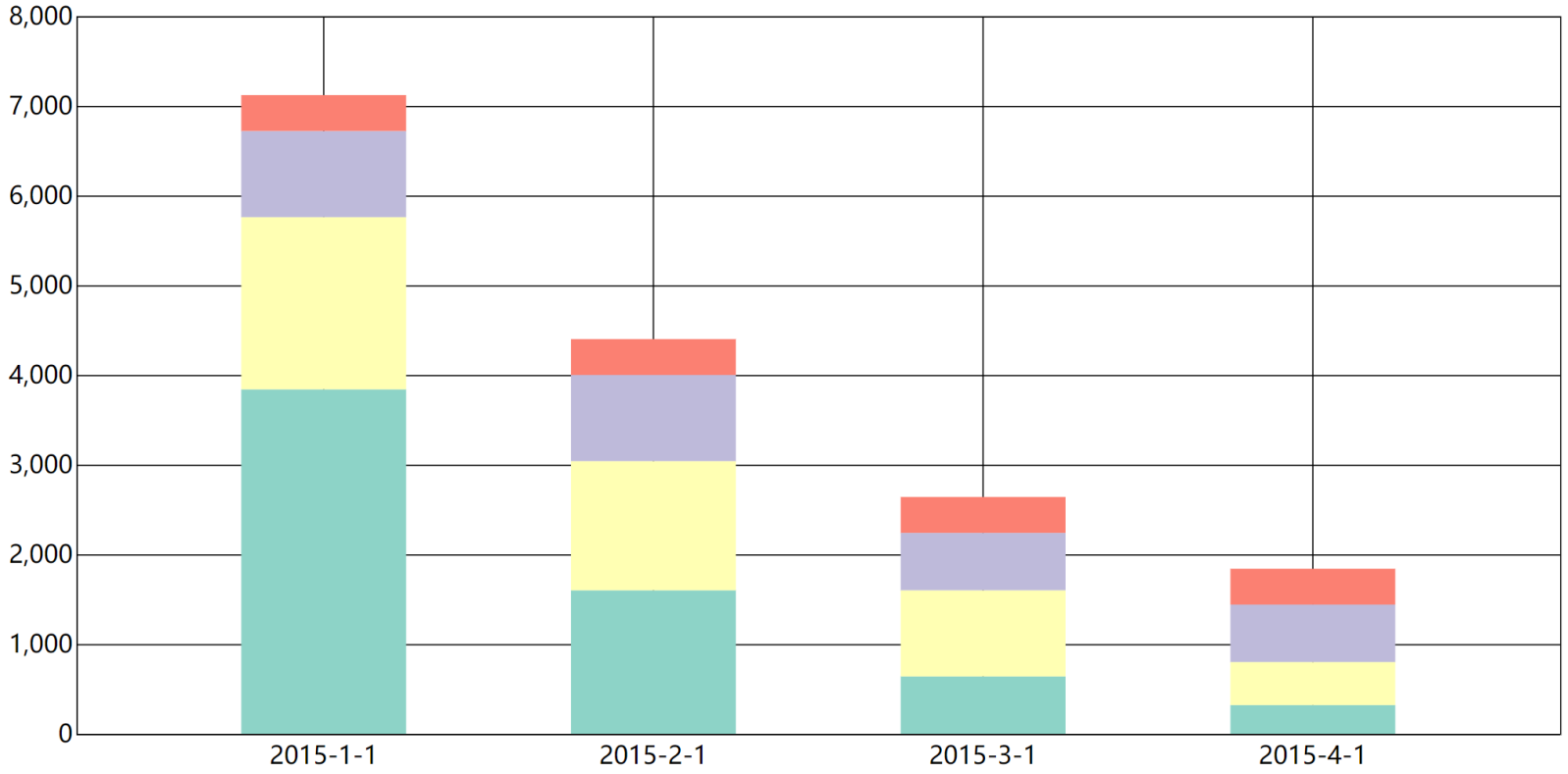
```
▼1: Array(4)
  ▶0: (2) [3840, 5760, data: {…}]
  ▶1: (2) [1600, 3040, data: {…}]
  ▶2: (2) [640, 1600, data: {…}]
  ▶3: (2) [320, 800, data: {…}]
    key: "bananas"
    index: 1
    length: 4
  ▶__proto__: Array(0)
2: (4) [Array(2), Array(2), Array(2), Array(2), key: "cherries", index: 2]
3: (4) [Array(2), Array(2), Array(2), Array(2), key: "dates", index: 3]
```

Month	Apples	Bananas	Cherries	Dates
1/2015	3840	1920	960	400
2/2015	1600	1440	960	400
3/2015	640	960	640	400
4/2015	320	480	640	400

https://github.com/d3/d3-shape/blob/v1.3.7/README.md#stack

堆叠柱状图

- code: https://github.com/Shao-Kui/D3.js-Demos/blob/master/static/d3-tutorial/stackbarchart-simple.html



将CSV数据‘堆叠’

- 本质上还是定义并‘配置’一个函数;
- .keys: 设置要堆叠的属性有哪些;
- .order: 这些属性要按照什么顺序;

```
var naiveStack = d3.stack()
.keys(naiveKeys)
.order(d3.stackOrderNone)(naiveData);
```

比例尺：离散到离散 & D3内嵌的配色方案

- 定义一个离散数据到离散数据的映射
  - 如：每个水果对应到某个颜色

```
const color = d3.scaleOrdinal()
.domain(naiveKeys)
.range(d3.schemeSet3)
```

- D3.js的内嵌（自带）配色方案？
  - https://github.com/d3/d3-scale-chromatic

d3.**schemePastel1** <>

d3.**schemePastel2** <>

d3.**schemeSet1** <>

An array of nine categorical colors represented as RGB hexadecimal strin

d3.**schemeSet2** <>

An array of eight categorical colors represented as RGB hexadecimal strir

d3.**schemeSet3** <>

堆叠数据的Data-Join

- 每条数据绑定的是长度为 **2** 的数组;
  - 两个数字表示在堆叠数据中的‘区间’
  - 数组的data属性可以映射到原本数据
- 用堆叠后的数据设置比例尺、位置、高度
- 注意：请尽可能先调用比例尺映射数据，再做运算！ （**绿色**框）

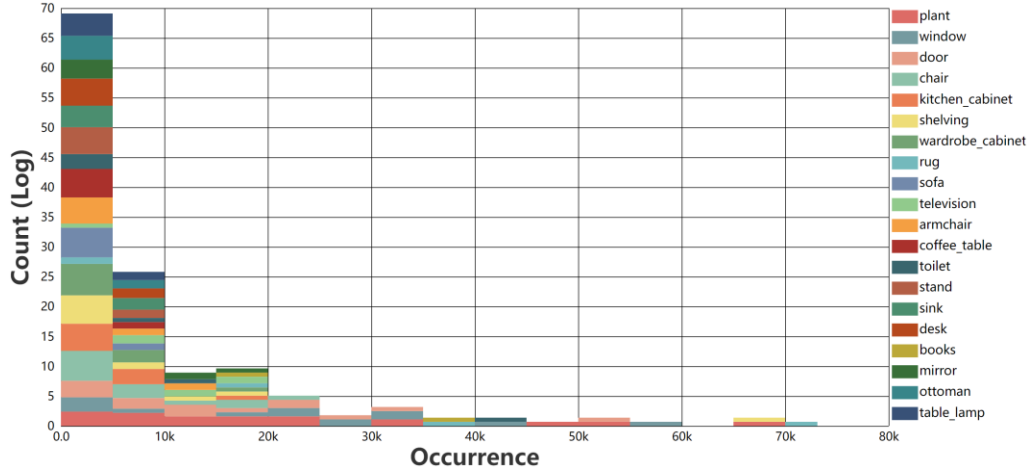
```
const yScale = d3.scaleLinear()
.domain([0, d3.max(naiveStack,
d => d3.max(d, subd => subd[1]))])
.range([innerHeight, 0])
.nice();
```

```
g.selectAll('.datagroup').data(naiveStack).join('g')
.attr('class', 'datagroup')
.attr('fill', d => color(d.key))
.selectAll('.datarect').data(d => d).join('rect')
.attr('class', 'datarect')
.attr('y', d => yScale(d[1]))
.attr('x', d => xScale(xValue(d.data)))
.attr('height', d => yScale(d[0]) - yScale(d[1]))
.attr('width', xScale.bandwidth());
```

Moment.js

- https://momentjs.com/
- 用于处理格式化输出、读取、操作日期等
- （非常好用，但仅个人推荐）
- 将日期转换成期望的格式
  - const xValue =
  - d => **moment**(d.month.toISOString()).**format**('**YYYY-M-D**');
- 不要忘记读取：<script src="../js/library/moment.min.js"></script>

Beyond ‘Stack’…



- D3.js也含有其他数据预处理的方法
- d3.histogram: 用于将数据按照某一属性分布在不同的区域
  - 常用于绘制**直方图**
- d3.pie: 用于将数据映射到圆周的各个弧度
  - 常用于绘制**饼图**
- Stack与Histogram的结合?
  - code: https://github.com/Shao-Kui/D3.js-Demos/blob/master/static/stack_histogram.html
  - Url: http://127.0.0.1:11666/stack_histogram