




	新手群	专业群	研究群
二维码	 <p>D3.js 扫一扫二维码，加入该群。</p>		 <p>大数据可视化 扫一扫二维码，加入该群。</p>

下面是译文，欢迎一起翻译，探讨~

## D3: 数据驱动文档 ( Data-Driven Documents )



D3 (或D3.js) 是一个用来使用Web标准做数据可视化的JavaScript函数库。

D3帮助我们融合SVG, Canvas 和 HTML操作技术让数据变得生动有趣。

D3将强大的**可视化**，**动态交互**和**数据驱动的DOM操作方法**完美结合，让我们可以充分发挥现代浏览器的功能，自由地设计正确的可视化界面。

### 资料

- [API参考](#)
- [发行版](#)
- [展廊](#)
- [案例](#)
- [Wiki](#)

### 安装

最近的稳定版是 (4.4.0), 可以按照wiki里的 [安装介绍](#) 安装使用。如果你使用NPM, 可执行 `npm install d3` 命令。不然的话可以下载[最新版](#)。发布包支持AMD, CommonJS, 和 vanilla 环境。自定义编译可以使用 [Rollup](#) 或者其他打包工具。也可以直接从 [d3js.org](#)引用:

```
<script src="https://d3js.org/d3.v4.min.js"></script>
```

非压缩版移除上面的 `.min` 即可。

### API 总览

#### 选择元素

- [选择](#) ([选择](#), [修改](#), [数据](#), [事件](#), [控制](#), [命名空间](#))

#### 数据类型

- [数组](#) ([统计](#), [直方图](#), [查找](#), [转换](#))
- [集合](#) ([对象](#), [映射](#) ( `map` ), [集合](#) ( `set` ), [嵌套](#))
- [颜色](#)
- [DSV](#) ( [分隔符分割的值](#) )
- [随机值](#)
- [时间序列](#)

## 格式化

- [数字格式化](#)
- [时间格式化](#)

## 加载数据

- [队列](#)
- [请求](#)

## 数据映射

- [比例尺](#) (连续型, 颜色序数型, 数值型, 序数型, 分类颜色型)

## 图形几何

- [形状](#) (弧, 饼, 线, 面积, 曲线, 符号, 堆叠)
- [轴](#)
- [泰森多边形](#)
- [路径](#)
- [多边形](#)
- [四叉树](#)

## 布局

- [力布局](#)
- [层次布局](#)

## 动态交互

- [定时器](#)
- [过渡](#)
- [插值器](#)
- [缓动](#)
- [事件分派](#)
- [拖动](#)
- [缩放](#)

D3 使用 [语义命名](#)。可使用d3.version获取当前版本号。

## 数组

数组操作，排序，查找，汇总等。

### 统计

计算基本汇总统计的一些方法。

- [d3.min](#) - 计算数组中的最小值。
- [d3.max](#) - 计算数组中的最大值。
- [d3.extent](#) - 计算数组的范围。
- [d3.sum](#) - 数组中所有元素求和。
- [d3.mean](#) - 计算数组的算术平均值。
- [d3.median](#) - 计算数组的中位数。
- [d3.quantile](#) - 计算一个数字数组排序后的分位数。
- [d3.variance](#) - 数组中数字的方差。
- [d3.deviation](#) - 数组中数字的标准差。

### 直方图

将离散样本分成连续的无重叠的间隔。

- [d3.histogram](#) - 创建一个新的直方图生成器。
- [histogram](#) - 对给定的样本数组计算直方图。

- [histogram.value](#) - 为每个样本指定一个值访问器。
- [histogram.domain](#) - 指定可观测值的间隔。
- [histogram.thresholds](#) - 指定值划分成不同箱的方法。
- [d3.thresholdFreedmanDiaconis](#) - Freedman–Diaconis 装箱规则。
- [d3.thresholdScott](#) - Scott's normal reference 装箱规则。
- [d3.thresholdSturges](#) - Sturges' 装箱准则。

## 查找

检索数组中特定的值。

- [d3.scan](#) - 使用比较器线查找。
- [d3.bisect](#) - 二分查找排序数组中的值。
- [d3.bisectRight](#) - 二分查找排序数组中的值。
- [d3.bisectLeft](#) - 二分查找排序数组中的值。
- [d3.bisector](#) - 使用访问器和比较器二分查找。
- [bisector.left](#) - 使用给定的比较器的 bisectLeft。
- [bisector.right](#) - 使用给定的比较器的 bisectRight。
- [d3.ascending](#) - 升序排序。
- [d3.descending](#) - 降序排序。

## 转换

转换数组并返回一个新的数组。

- [d3.merge](#) - 将多个数组合并成一个。
- [d3.pairs](#) - 创建一个相邻对数组。
- [d3.permute](#) - 安装指定的索引数组重排数组。
- [d3.shuffle](#) - 数组随机排序。
- [d3.ticks](#) - 从一个数组间隔生成有代表的值。
- [d3.tickStep](#) - 从一个数组间隔生成有代表的值。
- [d3.range](#) - 生成一组数值。
- [d3.transpose](#) - 数组转置。
- [d3.zip](#) - 转置多个数组。

## 轴

人类可读的刻度轴。

- [d3.axisTop](#) - 创建一个上部轴生成器。
- [d3.axisRight](#) - 创建一个右部轴生成器。
- [d3.axisBottom](#) - 创建一个底部轴生成器。
- [d3.axisLeft](#) - 创建一个左部轴生成器。
- [axis](#) - 为给定的选择生成轴。
- [axis.scale](#) - 设置比例尺。
- [axis.ticks](#) - 自定义刻度的生成和格式化方式。
- [axis.tickArguments](#) - 自定义刻度的生成和格式化方式。
- [axis.tickValues](#) - 明确地指定刻度值。
- [axis.tickFormat](#) - 明确地指定刻度格式。
- [axis.tickSize](#) - 设置刻度的大小。
- [axis.tickSizeInner](#) - 设置内刻度的大小。
- [axis.tickSizeOuter](#) - 设置外刻度的大小。
- [axis.tickPadding](#) - 设置刻度和标签之间的间距。

## 集合

便捷的数据结构，元素的键是字符串类型。

## 对象

将对象转为数组的方法。

- `d3.keys` - 列举关联数组的键。
- `d3.values` - 列举关联数组的值。
- `d3.entries` - 列举关联数组的键值对实体。

## 映射

类似ES6 Map，但是键是字符串类型的，并且有点其他区别。

- `d3.map` - 创建一个空的map。
- `map.has` - 返回map中是否包含某个值。
- `map.get` - 获取值。
- `map.set` - 设置值。
- `map.remove` - 移除值。
- `map.clear` - 移除所有值。
- `map.keys` - 获取键数组。
- `map.values` - 获取值数组。
- `map.entries` - 获取键值对数组。
- `map.each` - 为每个元素调用一次指定的方法。
- `map.empty` - 返回map是否为空。
- `map.size` - 计算值的数量。

## 集合

类似ES6 Set，但是键是字符串类型的，并且有点其他区别。

- `d3.set` - 创建一个空的set。
- `set.has` - 返回set中是否包含某个值。
- `set.add` - 添加指定值。
- `set.remove` - 删除指定值。
- `set.clear` - 移除所有值。
- `set.values` - 获取值数组。
- `set.each` - 为每个元素调用一次指定的方法。
- `set.empty` - 返回set是否为空。
- `set.size` - 计算值的数量。

## 嵌套

将数据组织成任意层次。

- `d3.nest` - 创建一个嵌套生成器。
- `nest.key` - 在嵌套层级中加一级。
- `nest.sortKeys` - 当前层级按键排序。
- `nest.sortValues` - 叶子层级按值排序。
- `nest.rollup` - 为叶子层指定汇总函数。
- `nest.map` - 生成一个嵌套，返回一个map。
- `nest.object` - 生成一个嵌套，返回一个关联数组。
- `nest.entries` - 生成一个嵌套，返回一个键值对数组。

## 颜色

颜色操作和颜色空间转换。

- `d3.color` - 解析给定的CSS颜色名。
- `color.rgb` - 计算该颜色的RGB值。
- `color.brighter` - 该颜色的高亮副本。
- `color.darker` - 该颜色的较亮副本。
- `color.displayable` - 如果该颜色在标准硬件上可以显示则返回true。

- [color.toString](#) - 将该颜色格式化为一个十六进制 RGB值字符串。
- [d3.rgb](#) - 创建一个RGB颜色。
- [d3.hsl](#) - 创建一个HSL颜色。
- [d3.lab](#) - 创建一个Lab颜色。
- [d3.hcl](#) - 创建一个HCL颜色。
- [d3.cubehelix](#) - 创建一个Cubehelix颜色。

## 分隔符分隔的值

解析和格式分隔符分隔的值（特别是TSV和CSV）

- [d3.dsvFormat](#) - 为指定的分隔符指定一个解析器和格式化器。
- [dsv.parse](#) - 解析给定的字符串返回一组对象。
- [dsv.parseRows](#) - 解析给定的字符串返回行数组。
- [dsv.format](#) - 格式化一组对象。
- [dsv.formatRows](#) - 格式化行数组。
- [d3.csvParse](#) - 解析给定的CSV字符串，返回一组对象。
- [d3.csvParseRows](#) - 解析给定的CSV字符串，返回行数组。
- [d3.csvFormat](#) - 格式化给定的CSV对象。
- [d3.csvFormatRows](#) - 格式化给定的CSV行数组。
- [d3.tsvParse](#) - 解析给定的TSV字符串，返回一组对象。
- [d3.tsvParseRows](#) - 解析给定的TSV字符串，返回行数组。
- [d3.tsvFormat](#) - 格式化给定的TSV对象。
- [d3.tsvFormatRows](#) - 格式化给定的TSV行数组。

## 事件分发

命名回调函数。

- [d3.dispatch](#) - 创建一个定制的事件分发器。
- [dispatch.on](#) - 注册或者解除注册事件监听器。
- [dispatch.copy](#) - 创建分发器副本。
- [dispatch.call](#) - 给注册的监听器分发事件。
- [dispatch.apply](#) - 给注册的监听器分发事件。

## 拖曳

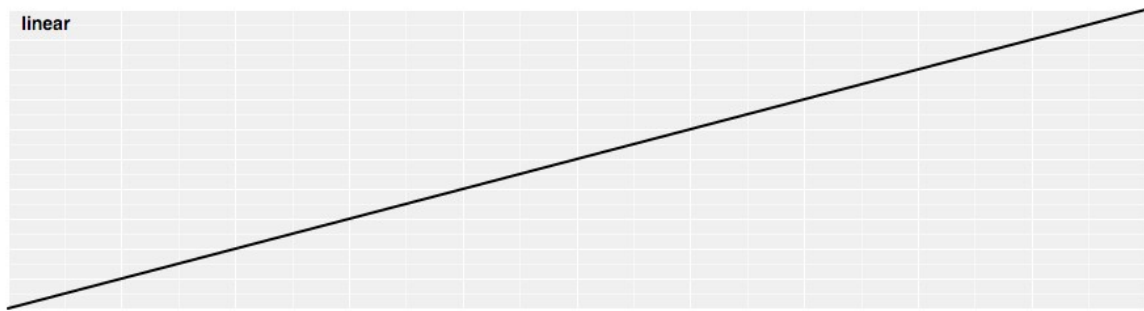
使用鼠标或触屏拖曳SVG，HTML 或 Canvas。

- [d3.drag](#) - 创建一个拖曳行为。
- [drag](#) - 对一个选择应用拖曳行为。
- [drag.container](#) - 设置坐标系统。
- [drag.filter](#) - 忽略一些初始的事件。
- [drag.subject](#) - 设置被拖曳对象。
- [drag.x](#) - 设置被拖曳对象的x-坐标。
- [drag.y](#) - 设置被拖曳对象的y-坐标。
- [drag.on](#) - 监听拖曳事件。
- [event.on](#) - 监听当前动作的拖曳事件。

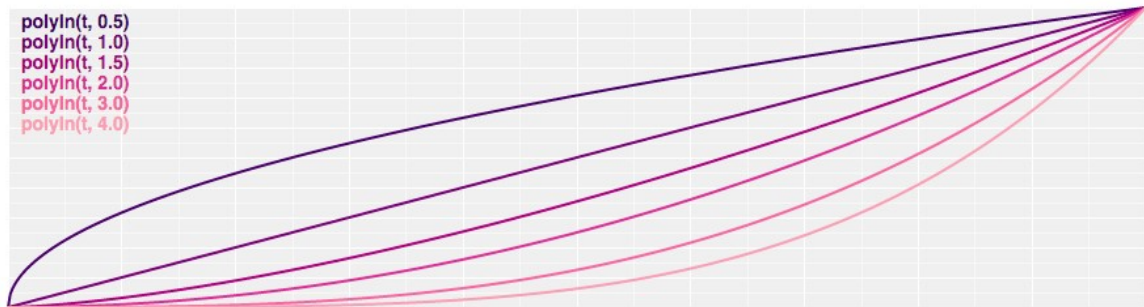
## 缓动

用来平滑过渡的缓动函数。

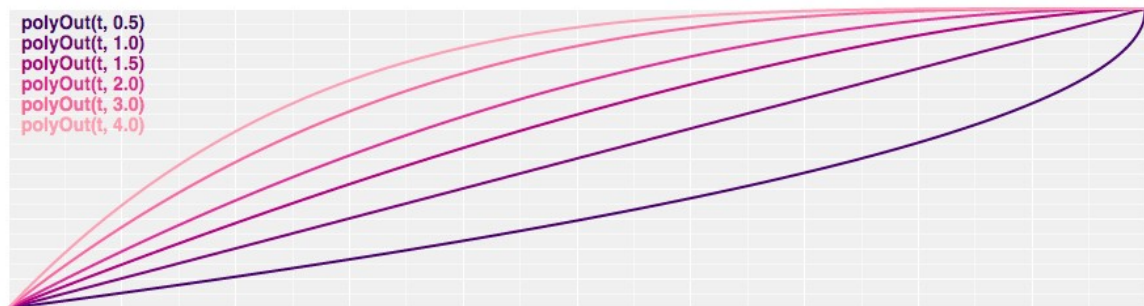
- [ease](#) - 缓动给定的标准化时间。
- [d3.easeLinear](#) - 线性缓动，就是个恒等函数。



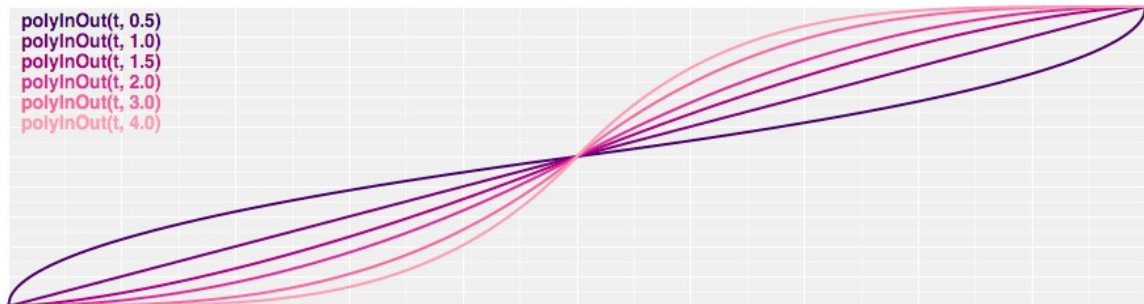
- `d3.easePolyIn` - 多项式缓动，加速到指定的速率。



- `d3.easePolyOut` - 逆多项式缓动，等价于  $1 - \text{polyIn}(1 - t)$ 。



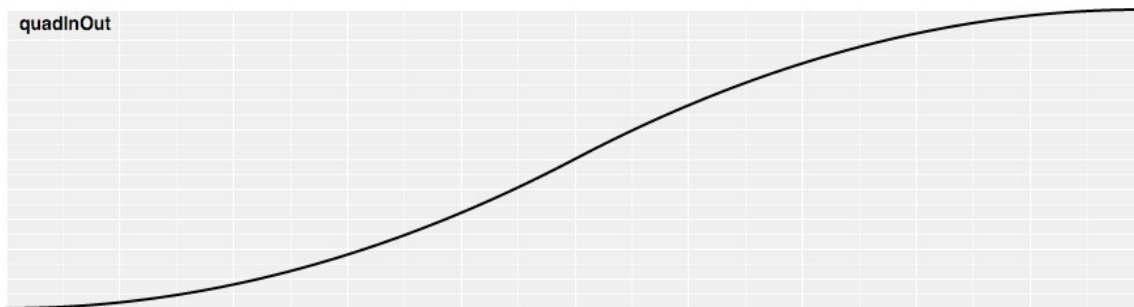
- `d3.easePolyInOut` - 对称多项式缓动。



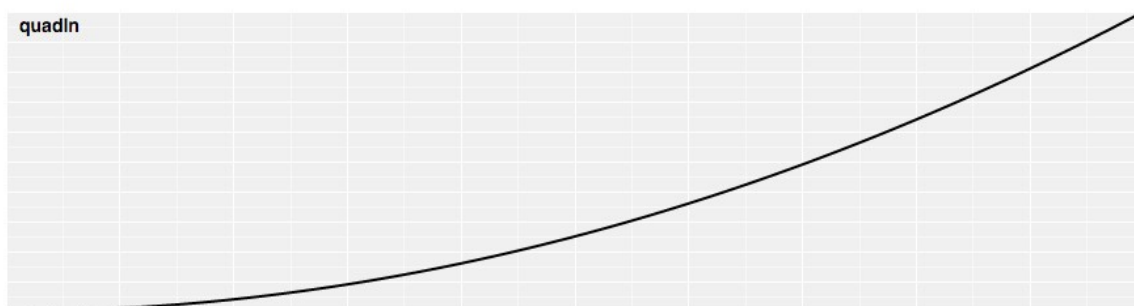
- `poly.exponent` - 指定缓动多项式的指数。

```
var linear = d3.easePoly.exponent(1),
    quad = d3.easePoly.exponent(2),
    cubic = d3.easePoly.exponent(3);
```

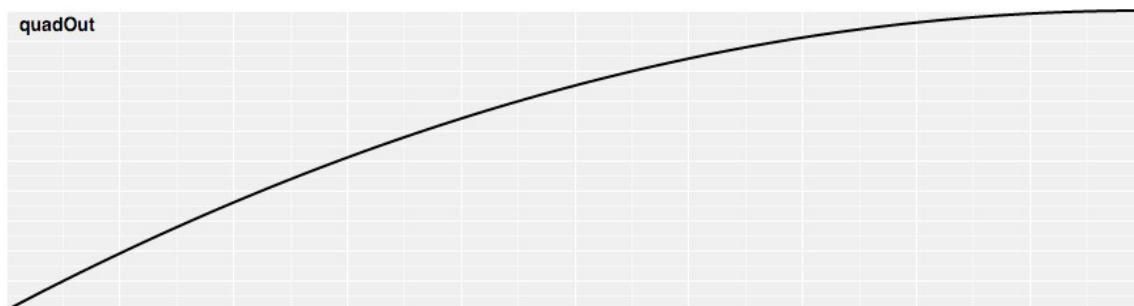
- `d3.easeQuad` - `easeQuadInOut`的别名。
- `d3.easeQuadInOut` - 对称平方缓动。



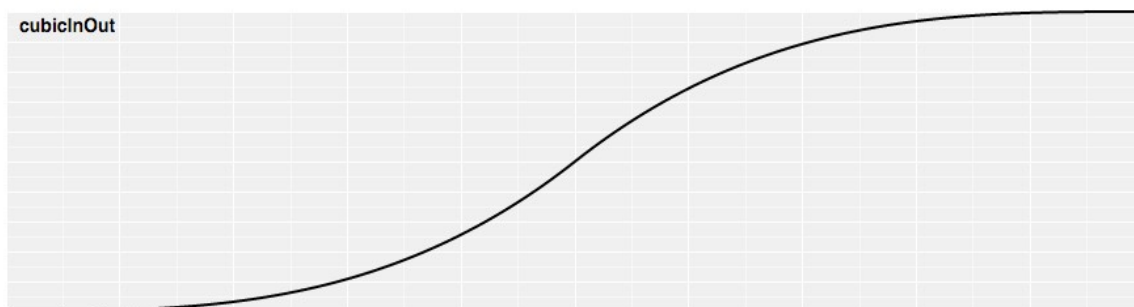
- [d3.easeQuadIn](#) - 平方缓动。



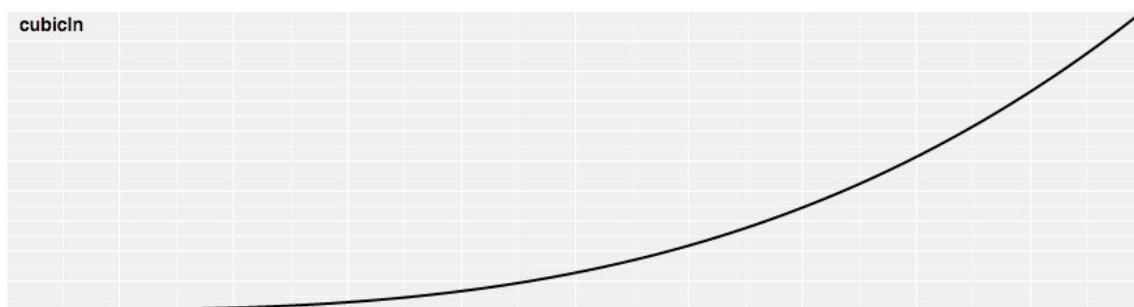
- [d3.easeQuadOut](#) - 逆平方缓动。



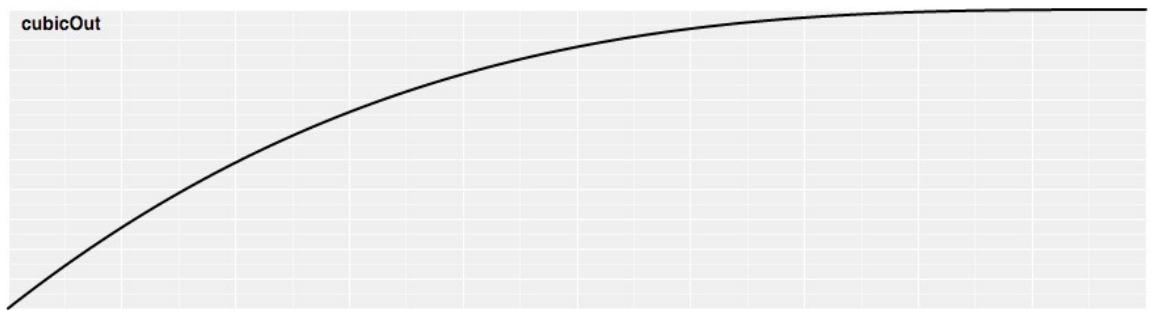
- [d3.easeCubic](#) - easeCubicInOut的别名。
- [d3.easeCubicInOut](#) - 对称立方缓动。



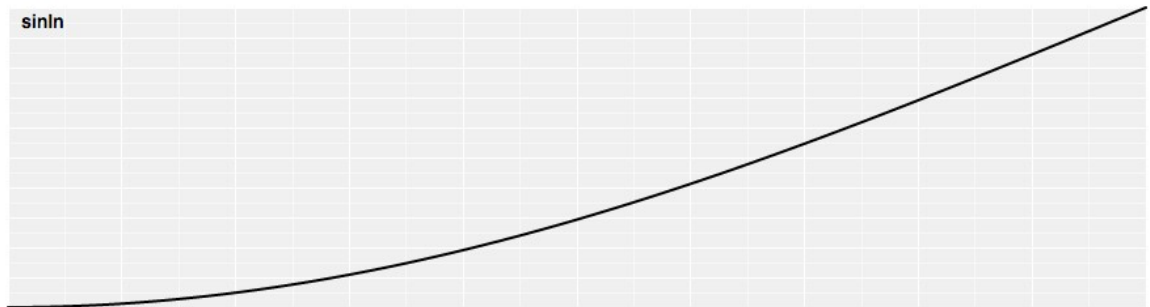
- [d3.easeCubicIn](#) - 立方缓动。



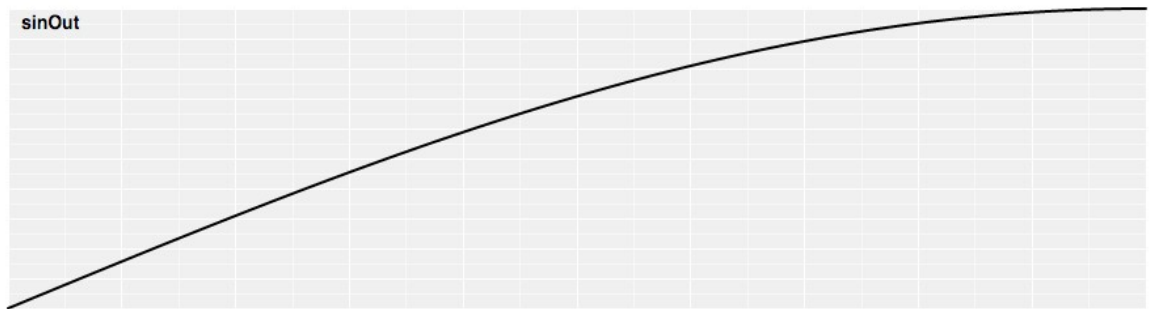
- [d3.easeCubicOut](#) - 逆立方缓动。



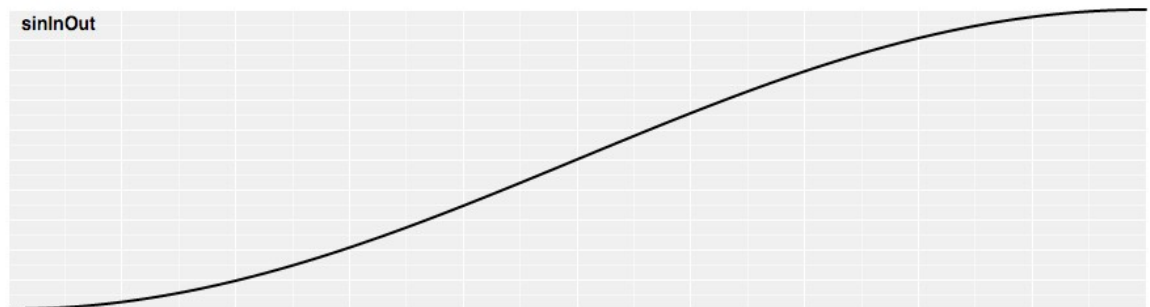
- [d3.easeSinIn](#) - 正弦缓动。



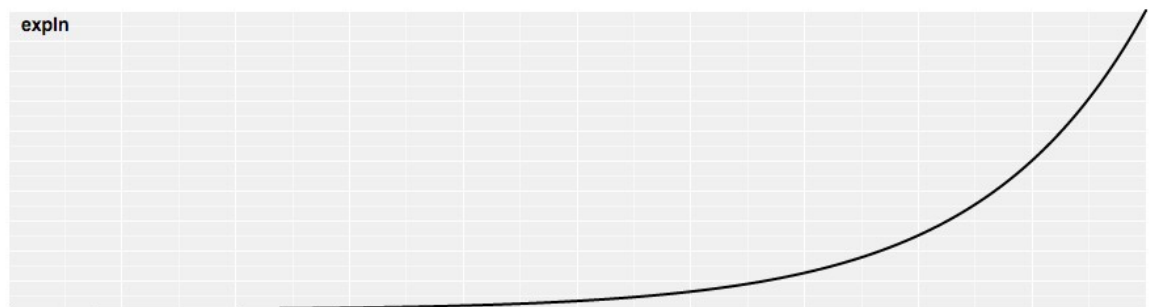
- [d3.easeSinOut](#) - 逆正弦缓动。



- [d3.easeSin](#) - easeSinInOut的别名。
- [d3.easeSinInOut](#) - 对称正弦缓动。

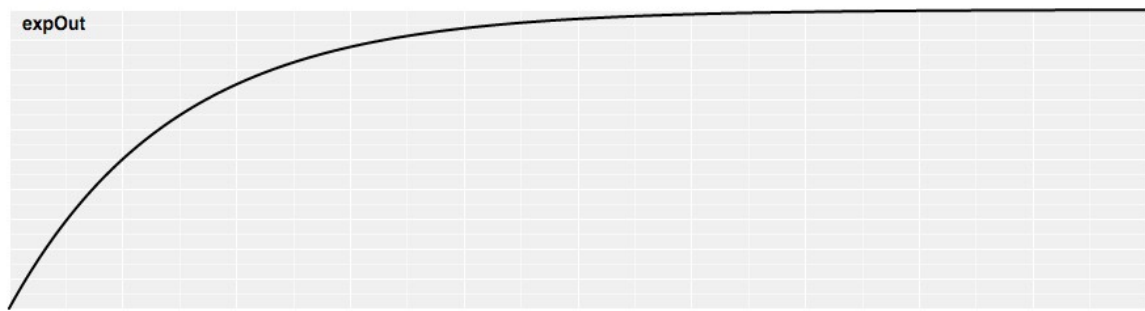


- [d3.easeExpIn](#) - 指数缓动。

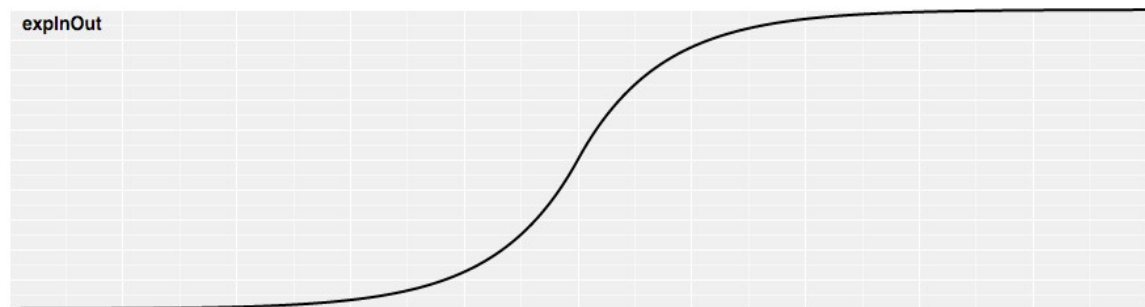


- [d3.easeExpOut](#) - 逆指数缓动。

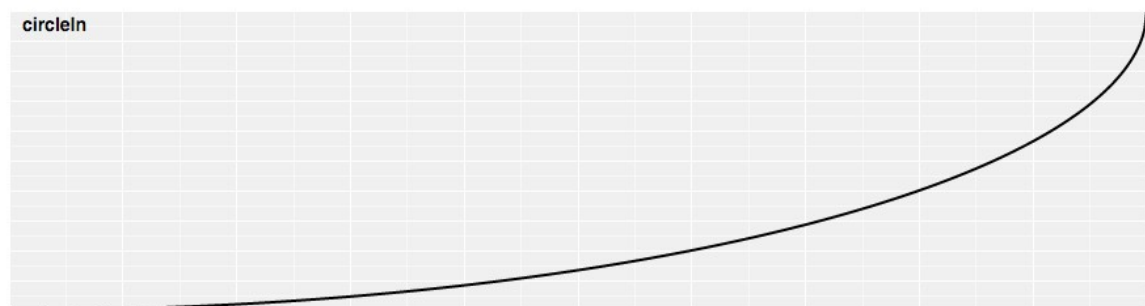




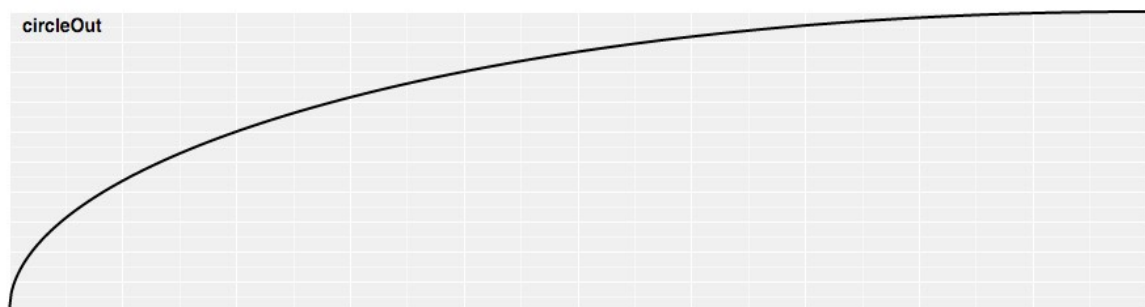
- [d3.easeExp](#) - easeExpInOut的别名。
- [d3.easeExpInOut](#) - 对称指数缓动。



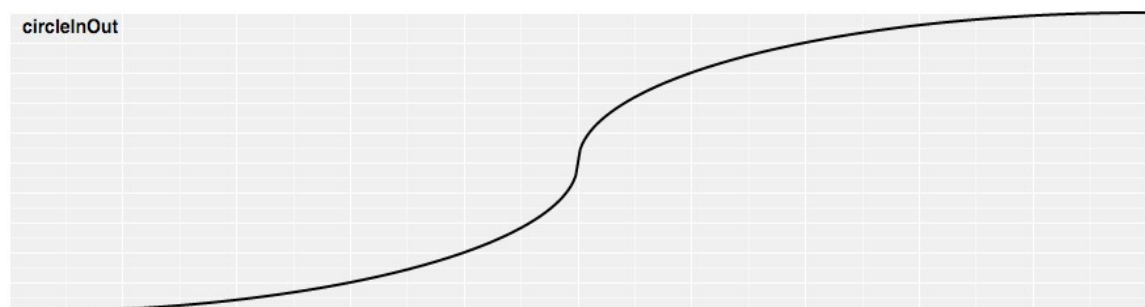
- [d3.easeCircleIn](#) - 圆形缓动。



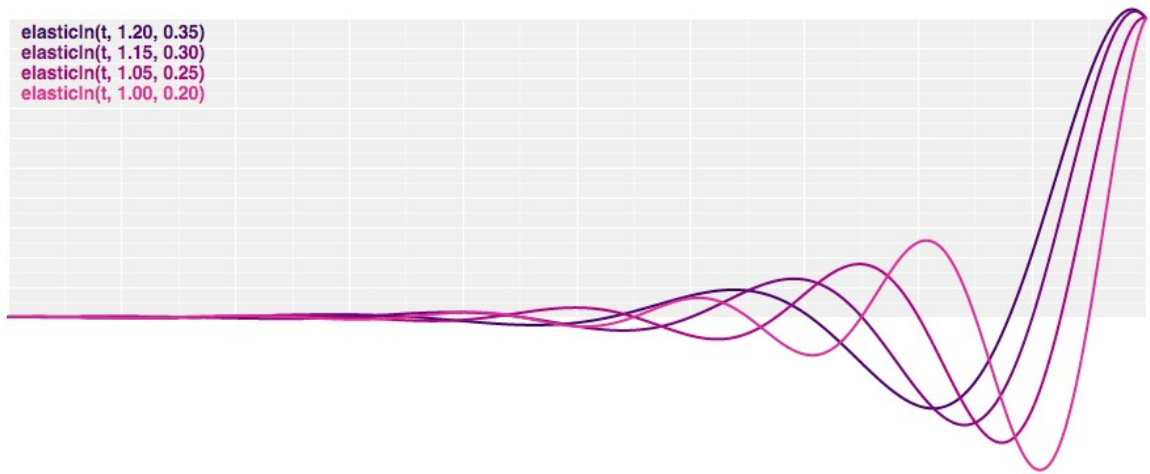
- [d3.easeCircleOut](#) - 逆圆形缓动。



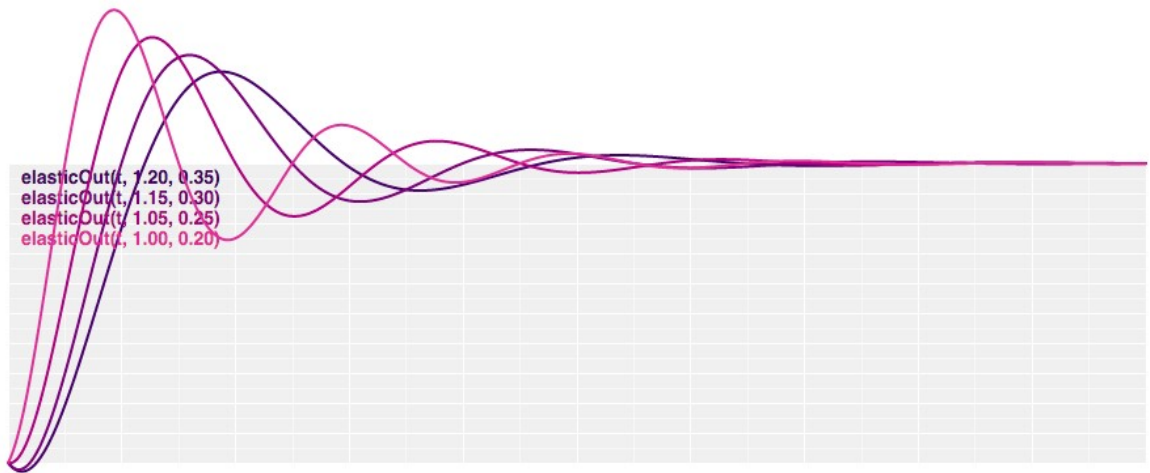
- [d3.easeCircle](#) - easeCircleInOut的别名。
- [d3.easeCircleInOut](#) - 对称圆形缓动。



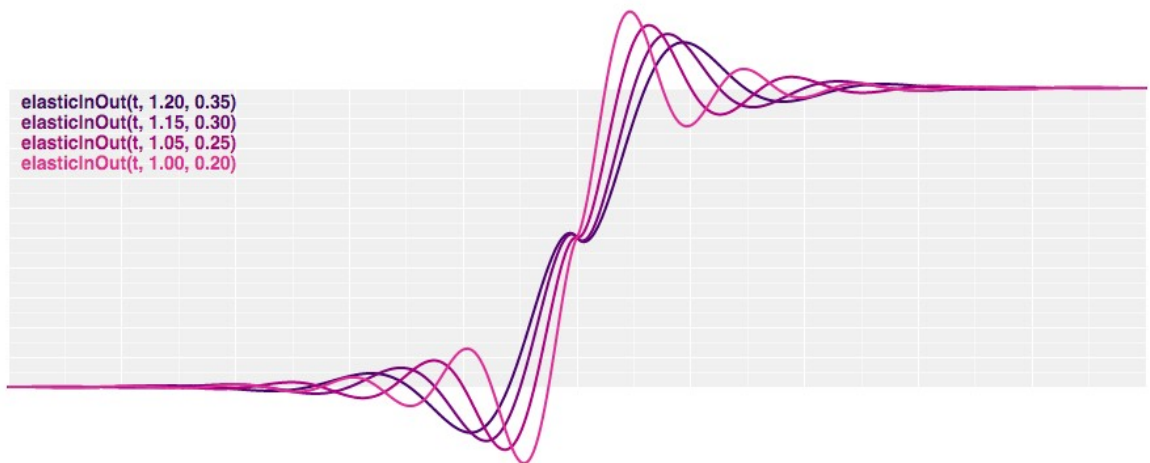
- [d3.easeElasticIn](#) - 弹性缓动，类似松紧带。



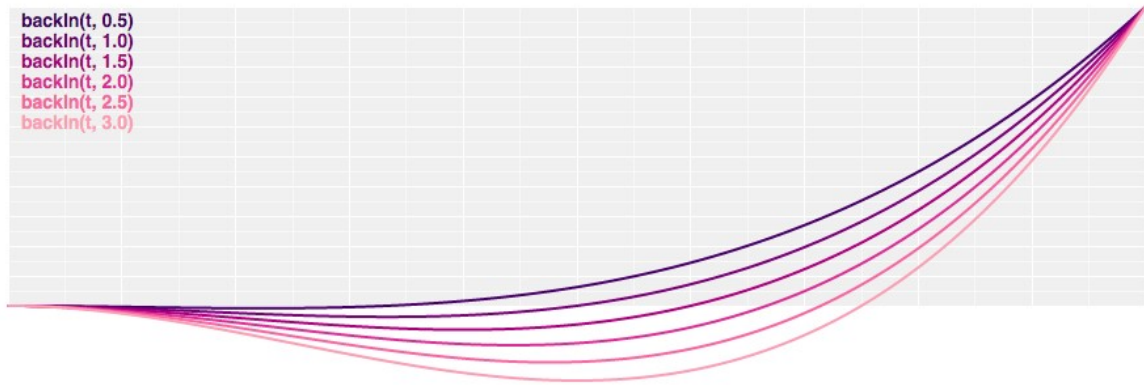
- `d3.easeElastic` - `easeElasticOut`的别名。
- `d3.easeElasticOut` - 逆弹性缓动。



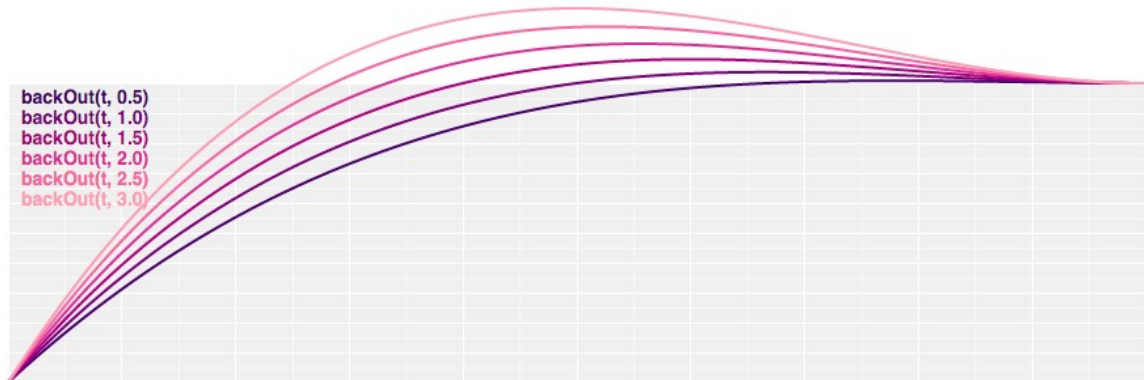
- `d3.easeElasticInOut` - 对称弹性缓动。



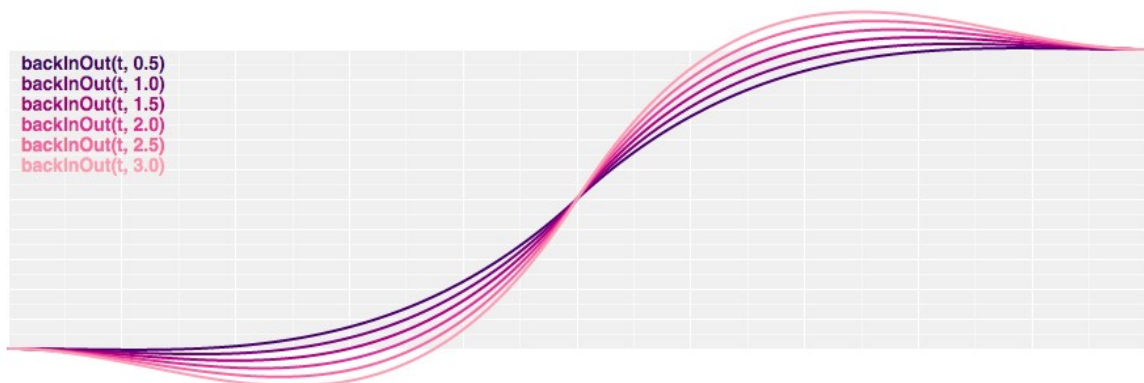
- `elastic.amplitude` - 指定弹性振幅。
- `elastic.period` - 指定弹性周期。
- `d3.easeBackIn` - 预期缓动。



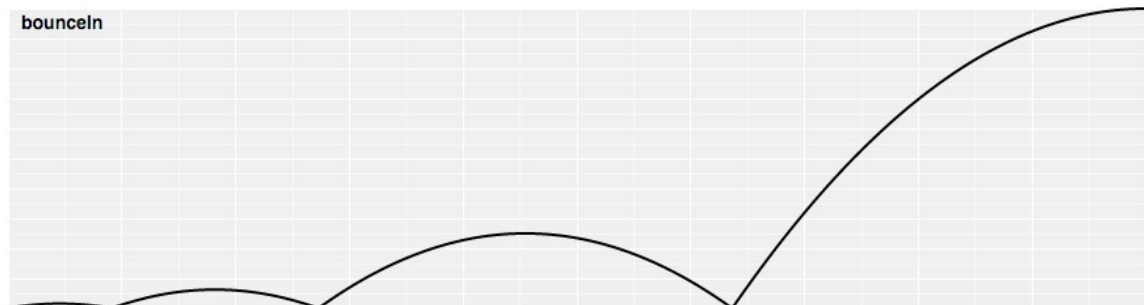
- `d3.easeBackOut` - 逆预期缓动。



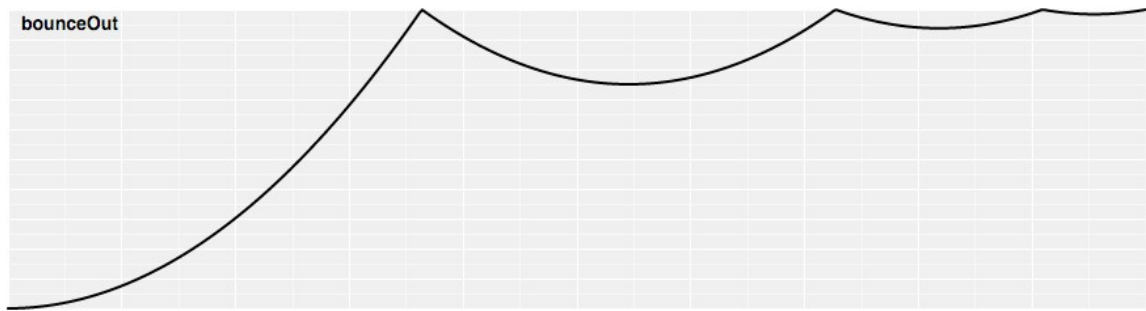
- `d3.easeBack` - `easeBackInOut`的别名。
- `d3.easeBackInOut` - 对称预期缓动。



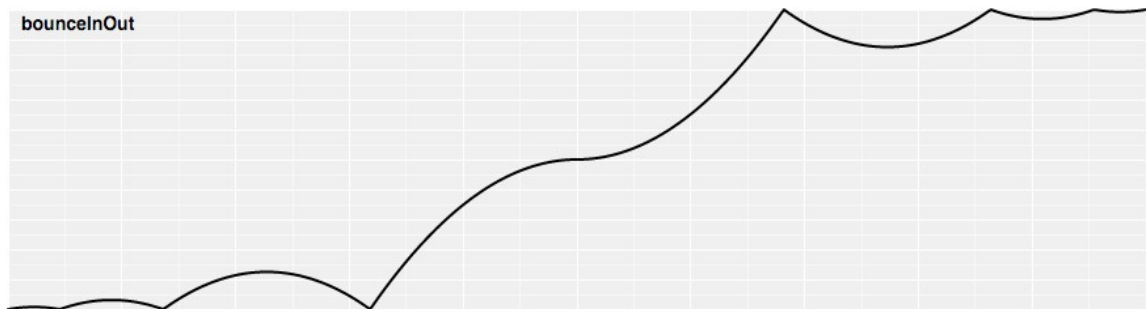
- `back.overshoot` - 指定超调量。
- `d3.easeBounceIn` - 弹跳缓动，类似弹跳的小球。



- `d3.easeBounce` - `easeBounceOut`的别名。
- `d3.easeBounceOut` - 逆弹跳缓动。



- [d3.easeBounceInOut](#) - 均匀弹跳缓动。



## 力导向图

力导向图使用velocity Verlet整合算法实现。

- [d3.forceSimulation](#) - 创建一个力模拟。
- [simulation.restart](#) - 重启力模拟。
- [simulation.stop](#) - 停止力模拟。
- [simulation.tick](#) - 将力模拟向前推进一步。
- [simulation.nodes](#) - 设置力模拟的节点。
- [simulation.alpha](#) - 设置当前的  $\alpha$  值。
- [simulation.alphaMin](#) - 设置  $\alpha$  最小阈值。
- [simulation.alphaDecay](#) - 设置  $\alpha$  指数衰减率。
- [simulation.alphaTarget](#) - 设置目标  $\alpha$ 。
- [simulation.drag](#) - 设置曳引系数。
- [simulation.force](#) - 添加或移除力。
- [simulation.fix](#) - 固定节点位置。
- [simulation.unfix](#) - 释放固定的节点。
- [simulation.find](#) - 查找给定位置最近的节点。
- [simulation.on](#) - 添加或移除事件监听器。
- [force](#) - 应用力模拟。
- [force.initialize](#) - 使用给定的节点初始化力布局。
- [d3.forceCenter](#) - 创建一个力中心。
- [center.x](#) - 设置中心的x-坐标。
- [center.y](#) - 设置中心的y-坐标。
- [d3.forceCollide](#) - 创建一个圆碰撞力。
- [collide.radius](#) - 设置圆的半径。
- [collide.strength](#) - 设置碰撞检测强度。
- [collide.iterations](#) - 设置迭代次数。
- [d3.forceLink](#) - 创建连接力。
- [link.links](#) - 设置连接数组。
- [link.id](#) - 连接数组。
- [link.distance](#) - 设置连接距离。
- [link.strength](#) - 设置连接强度。
- [link.iterations](#) - 设置迭代次数。
- [d3.forceManyBody](#) - 创建多体力。

- [manyBody.strength](#) - 设置力强度。
- [manyBody.theta](#) - 设置Barnes-Hut近似精度。
- [manyBody.distanceMin](#) - 当节点关闭限制力。
- [manyBody.distanceMax](#) - 当节点太远限制力。
- [d3.forceX](#) - 创建x-定位力。
- [x.strength](#) - 设置力强度。
- [x.x](#) - 设置目标x-坐标。
- [d3.forceY](#) - 创建y-定位力。
- [y.strength](#) - 设置力强度。
- [y.y](#) - 设置目标y-坐标。

## 层次布局

---

用来可视化层次型数据的布局算法。

- [d3.hierarchy](#) - 从层次型的数据构造一个根节点。
- [node.ancestors](#) - 生成祖先数组。
- [node.descendants](#) - 生成后代数组。
- [node.leaves](#) - 生成层级数组。
- [node.path](#) - 生成到达另一个节点的最短路径。
- [node.sum](#) - 求和。
- [node.sort](#) - 排序所有后代的兄弟节点。
- [node.each](#) - 广度优先遍历。
- [node.eachAfter](#) - 后序遍历。
- [node.eachBefore](#) - 先序遍历。
- [node.copy](#) - 拷贝层次布局。
- [d3.stratify](#) - 创建一个层操作符。
- [stratify](#) - 从表格式的数据构造一个根节点。
- [stratify.id](#) - 设置节点ID访问器。
- [stratify.parentId](#) - 设置父节点ID访问器。
- [d3.cluster](#) - 创建一个新的簇（系统树图）布局。
- [cluster](#) - 将给定的层次数据排列到簇中。
- [cluster.size](#) - 设置布局大小。
- [cluster.nodeSize](#) - 设置节点大小。
- [cluster.separation](#) - 设置层间距。
- [d3.tree](#) - 创建新的整齐树布局。
- [tree](#) - 将给定的层次数据排列到树中。
- [tree.size](#) - 设置布局大小。
- [tree.nodeSize](#) - 设置节点大小。
- [tree.separation](#) - 设置层间距。
- [d3.treemap](#) - 创建一个新的矩形填充树布局（简称矩形树布局）。
- [treemap](#) - 使用矩形填充树布局排列给定的层次数据。
- [treemap.tile](#) - 设置铺砌方法。
- [treemap.size](#) - 设置布局大小。
- [treemap.round](#) - 设置输出坐标是否是取整的。
- [treemap.padding](#) - 设置间距。
- [treemap.paddingInner](#) - 设置兄弟节点之间的间距。
- [treemap.paddingOuter](#) - 设置父子节点之间的间距。
- [treemap.paddingTop](#) - 设置父节点顶边缘和子节点的间距。
- [treemap.paddingRight](#) - 设置父节点右边缘和子节点的间距。
- [treemap.paddingBottom](#) - 设置父节点底边缘和子节点的间距。
- [treemap.paddingLeft](#) - 设置父节点左边缘和子节点的间距。
- [d3.treemapBinary](#) - 使用平衡二叉树铺砌。
- [d3.treemapDice](#) - 水平列铺砌。
- [d3.treemapSlice](#) - 垂直列铺砌。

- [d3.treemapSliceDice](#) - 水平和垂直交替铺砌。
- [d3.treemapSquarify](#) - 正方形铺砌。
- [squarify.ratio](#) - 设置期望的矩形长宽比。
- [d3.partition](#) - 创建新的分区布局（旭日图和冰柱图）。
- [partition](#) - 使用分区布局排列给定的层次数据。
- [partition.size](#) - 设置布局大小。
- [partition.round](#) - 设置输出坐标是否是取整的。
- [partition.padding](#) - 设置间距。
- [d3.pack](#) - 创建一个新的圆形填充布局（简称包布局）。
- [pack](#) - 使用包布局排列给定的层次数据。
- [pack.radius](#) - 设置半径访问器。
- [pack.size](#) - 设置布局大小。
- [pack.padding](#) - 设置间距。
- [d3.packSiblings](#) - 包装指定的圆数组。
- [d3.packEnclose](#) - 围住指定的圆数组。

## 插值器

插补数字，字符串，颜色，数组，对象等。

- [d3.interpolate](#) - 插补任意值。
- [d3.interpolateArray](#) - 插补任意值的数组。
- [d3.interpolateNumber](#) - 插补数组。
- [d3.interpolateObject](#) - 插补充任意对象
- [d3.interpolateRound](#) - 插补整数。
- [d3.interpolateString](#) - 插补嵌入数字的字符串。
- [d3.interpolateTransformCss](#) - 插补2D CSS变换。
- [d3.interpolateTransformSvg](#) - 插补2D SVG变换。
- [d3.interpolateZoom](#) - 两个视图之间平移和缩放。
- [d3.interpolateRgb](#) - 插补RGB颜色。
- [d3.interpolateHsl](#) - 插补HSL颜色。
- [d3.interpolateHslLong](#) - 插补HSL颜色（长整型）。
- [d3.interpolateLab](#) - 插补Lab颜色。
- [d3.interpolateHcl](#) - 插补HCL颜色。
- [d3.interpolateHclLong](#) - 插补HCL颜色（长整型）。
- [d3.interpolateCubehelix](#) - 插补Cubehelix颜色。
- [d3.interpolateCubehelixLong](#) - 插补Cubehelix颜色（长整型）。
- [interpolate.gamma](#) - 在插值过程中使用  $\gamma$  矫正。

## 数字格式化

将数字格式化为人类可读的形式。

- [d3.format](#) - [enUs.format](#)的别名。
- [d3.formatPrefix](#) - [enUs.formatPrefix](#)的别名。
- [d3.formatSpecifier](#) - 解析数字格式说明符。
- [d3.formatLocale](#) - 定义一个自定义的本地化。
- [locale.format](#) - 创建一个数字格式。
- [locale.formatPrefix](#) - 创建一个SI-前缀数字格式化。
- [d3.formatCaEs](#) - Catalan (西班牙) 本地化。
- [d3.formatCsCz](#) - Czech (捷克) 本地化。
- [d3.formatDeCh](#) - German (瑞士) 本地化。
- [d3.formatDeDe](#) - German (德国) 本地化。
- [d3.formatEnCa](#) - English (加拿大) 本地化。
- [d3.formatEnGb](#) - English (英国) 本地化。

- [d3.formatEnUs](#) - English (美国) 本地化。
- [d3.formatEsEs](#) - Spanish (西班牙) 本地化。
- [d3.formatFiFi](#) - Finnish (芬兰) 本地化。
- [d3.formatFrCa](#) - French (加拿大) 本地化。
- [d3.formatFrFr](#) - French (法国) 本地化。
- [d3.formatHeIl](#) - Hebrew (以色列) 本地化。
- [d3.formatHuHu](#) - Hungarian (匈牙利) 本地化。
- [d3.formatItIt](#) - Italian (意大利) 本地化。
- [d3.formatJaJp](#) - Japanese (日本) 本地化。
- [d3.formatKoKr](#) - Korean (韩国) 本地化。
- [d3.formatMkMk](#) - Macedonian (马其顿) 本地化。
- [d3.formatNlNl](#) - Dutch (荷兰) 本地化。
- [d3.formatPlPl](#) - Polish (波兰) 本地化。
- [d3.formatPtBr](#) - Portuguese (巴西) 本地化。
- [d3.formatRuRu](#) - Russian (俄罗斯) 本地化。
- [d3.formatSvSe](#) - Swedish (瑞典) 本地化。
- [d3.formatZhCn](#) - Chinese (中国) 本地化。
- [d3.precisionFixed](#) - 计算定点表示法的小数精度。
- [d3.precisionPrefix](#) - 计算SI-前缀记号法的小数精度。
- [d3.precisionRound](#) - 计算有效数字。

## 路径

序列化Canvas路径命令为SVG。

- [d3.path](#) - 创建一个新的路径序列化。
- [path.moveTo](#) - 移动到给定的点。
- [path.closePath](#) - 关闭当前的子路径。
- [path.lineTo](#) - 画一条直线段。
- [path.quadraticCurveTo](#) - 画一条二次贝塞尔曲线段。
- [path.bezierCurveTo](#) - 画一条三次贝塞尔曲线段。
- [path.arcTo](#) - 画一条三次贝塞尔曲线弧。
- [path.arc](#) - 画一条三次贝塞尔曲线弧。
- [path.rect](#) - 画一个矩形。
- [path.toString](#) - 序列化为SVG路径的data属性字符串。

## 多边形

二维多边形的几何操作。

- [d3.polygonArea](#) - 计算给定多边形的面积。
- [d3.polygonCentroid](#) - 计算给定多边形的中心。
- [d3.polygonHull](#) - 计算给定点集的凸包。
- [d3.polygonContains](#) - 测试点是否在多边形内。
- [d3.polygonLength](#) - 计算给定多边形的周长的长度。

## 四叉树

二维递归空间细分。

- [d3.quadtree](#) - 创建一个新的空的四叉树。
- [quadtree.x](#) - 设置x访问器。
- [quadtree.y](#) - 设置y访问器。
- [quadtree.add](#) - 添加数据到四叉树中。
- [quadtree.remove](#) - 删除四叉树中的数据。
- [quadtree.copy](#) - 创建一个四叉树的副本。



- [quadtrees.root](#) - 获取四叉树的根节点。
- [quadtrees.data](#) - 从四叉树检索所有数据。
- [quadtrees.size](#) - 计算在四叉树中数据的数量。
- [quadtrees.find](#) - 在四叉树快速找到最接近的数据。
- [quadtrees.visit](#) - 选择性地访问四叉树中的节点。
- [quadtrees.visitAfter](#) - 访问四叉树中的所有节点。
- [quadtrees.cover](#) - 扩充四叉树覆盖一个点。
- [quadtrees.extent](#) - 扩充四叉树覆盖一个范围。

## 队列

使用可配置的并发性评估异步任务。

- [d3.queue](#) - 管理异步任务的并发评估。
- [queue.defer](#) - 注册一个用来评估的任务。
- [queue.abort](#) - 中止任何活动任务,取消任何挂起任务。
- [queue.await](#) - 注册一个任务结束后的回调函数。
- [queue.awaitAll](#) - 注册一个所有任务结束后的回调函数。

## 随机数

生成不同分布的随机数。

- [d3.randomUniform](#) - 均匀分布。
- [d3.randomNormal](#) - 正态分布。
- [d3.randomLogNormal](#) - 对数正态分布。
- [d3.randomBates](#) - 贝茨分布。
- [d3.randomIrwinHall](#) - Irwin-Hall分布。
- [d3.randomExponential](#) - 指数分布

## 请求

XMLHttpRequest的简便封装。

- [d3.request](#) - 创建一个异步请求。
- [request.header](#) - 设置请求头。
- [request.user](#) - 设置用来认证身份的用户名。
- [request.password](#) - 设置用来认证身份的密码。
- [request.mimeType](#) - 设置MIME类型。
- [request.timeout](#) - 设置超时时长（以秒为单位）。
- [request.responseType](#) - 设置响应类型。
- [request.response](#) - 设置响应函数。
- [request.get](#) - 发送GET请求。
- [request.post](#) - 发送POST请求。
- [request.send](#) - 设置请求。
- [request.abort](#) - 中断请求。
- [request.on](#) - 监听请求事件。
- [d3.csv](#) - 获取CSV文件。
- [d3.html](#) - 获取HTML文件。
- [d3.json](#) - 获取JSON文件。
- [d3.text](#) - 获取文本文件。
- [d3.tsv](#) - 获取TSV文件。
- [d3.xml](#) - 获取XML文件。

## 比例尺

映射抽象数据为可视化表示所需要的形式。



## 连续型比例尺

将连续的，数量的定义域映射为连续的值域上。

- *continuous* - 计算对应于给定的定义域的值域。
- *continuous.invert* - 计算对应于给定的值域的定义域。
- *continuous.domain* - 设置输入的定义域。
- *continuous.range* - 设置输出的值域。
- *continuous.rangeRound* - 设置取整后的值域
- *continuous.clamp* - 启用闭合。
- *continuous.interpolate* - 设置输出插值器。
- *continuous.ticks* - 计算定义域中有代表性的刻度值。
- *continuous.tickFormat* - 格式化刻度值。
- *continuous.nice* - 优化定义域。
- *continuous.copy* - 创建比例尺的副本。
- *d3.scaleLinear* - 创建定量线性比例尺。
- *d3.scalePow* - 创建定量幂比例尺。
- *pow* - 计算对应于给定的定义域的值域。
- *pow.invert* - 计算对应于给定的值域的定义域。
- *pow.exponent* - 设置幂指数。
- *pow.domain* - 设置输入的定义域。
- *pow.range* - 设置输出的值域。
- *pow.rangeRound* - 设置取整后的值域
- *pow.clamp* - 启用闭合。
- *pow.interpolate* - 设置输出插值器。
- *pow.ticks* - 计算定义域中有代表性的刻度值。
- *pow.tickFormat* - 格式化刻度值。
- *pow.nice* - 优化定义域。
- *pow.copy* - 创建比例尺的副本。
- *d3.scaleSqrt* - 创建一个幂比例尺，指数是0.5。
- *d3.scaleLog* - 创建定量对数比例尺。
- *log* - 计算对应于给定的定义域的值域。
- *log.invert* - 计算对应于给定的值域的定义域。
- *log.base* - 设置对数基底。
- *log.domain* - 设置输入的定义域。
- *log.range* - 设置输出的值域。
- *log.rangeRound* - 设置取整后的值域
- *log.clamp* - 启用闭合。
- *log.interpolate* - 设置输出插值器。
- *log.ticks* - 计算定义域中有代表性的刻度值。
- *log.tickFormat* - 格式化刻度值。
- *log.nice* - 优化定义域。
- *log.copy* - 创建比例尺的副本。
- *d3.scaleIdentity* - 创建定量恒等比例尺。
- *d3.scaleTime* - 创建时间线性比例尺。
- *time* - 计算对应于给定的定义域的值域。
- *time.invert* - 计算对应于给定的值域的定义域。
- *time.domain* - 设置输入的定义域。
- *time.range* - 设置输出的值域。
- *time.rangeRound* - 设置取整后的值域
- *time.clamp* - 启用闭合。
- *time.interpolate* - 设置输出插值器。
- *time.ticks* - 计算定义域中有代表性的刻度值。
- *time.tickFormat* - 格式化刻度值。
- *time.nice* - 优化定义域。

- [time.copy](#) - 创建比例尺的副本。
- [d3.scaleUtc](#) - 创建UTC时间的线性比例尺。

## 连续颜色比例尺

将连续的，数量的定义域映射为连续的，固定的颜色插值器。

- [d3.scaleSequential](#) - 创建一个顺序比例尺。create a sequential scale.
- [sequential.interpolator](#) - 设置比例尺的输出插值器。

- [d3.interpolateViridis](#) - 暗到明的颜色组合。



- [d3.interpolateInferno](#) - 暗到明的颜色组合。



- [d3.interpolateMagma](#) - 暗到明的颜色组合。



- [d3.interpolatePlasma](#) - 暗到明的颜色组合。



- [d3.interpolateWarm](#) - 色相环颜色组合。



- [d3.interpolateCool](#) - 色相环颜色组合。



- [d3.interpolateRainbow](#) - 循环的色相环颜色组合。



- [d3.interpolateCubehelixDefault](#) - 暗到明的色相环颜色组合。



## 量化比例尺

将连续的数量的定义域映射为离散的值域。

- [d3.scaleQuantize](#) - 创建一个均匀的量化的线性比例尺。
- [quantize](#) - 计算对应于给定的定义域的值域。
- [quantize.invertExtent](#) - 计算对应于给定的值域的定义域。
- [quantize.domain](#) - 设置输入的定义域。
- [quantize.range](#) - 设置输出的值域。
- [quantize.nice](#) - 优化定义域。
- [quantize.ticks](#) - 计算定义域中有代表性的刻度值。
- [quantize.tickFormat](#) - 格式化刻度值。
- [quantize.copy](#) - 创建比例尺的副本。
- [d3.scaleQuantile](#) - 创建一个分位数的量化的线性比例尺。
- [quantile](#) - 计算对应于给定的定义域的值域。
- [quantile.invertExtent](#) - 计算对应于给定的值域的定义域。
- [quantile.domain](#) - 设置输入的定义域。
- [quantile.range](#) - 设置输出的值域。
- [quantile.quantiles](#) - 设置分位数的阈值。
- [quantile.copy](#) - 创建比例尺的副本。
- [d3.scaleThreshold](#) - 创建一个任意的量化的线性比例尺。
- [threshold](#) - 计算对应于给定的定义域的值域。

- [threshold.invertExtent](#) - 计算对应于给定的值域的定义域。
- [threshold.domain](#) - 设置输入的定义域。
- [threshold.range](#) - 设置输出的值域。
- [threshold.copy](#) - 创建比例尺的副本。

## 序数比例尺

定义域和值域都是离散的。

- [d3.scaleOrdinal](#) - 创建一个序数比例尺。
- [ordinal](#) - 计算对应于给定的定义域的值域。
- [ordinal.domain](#) - 设置输入的定义域。
- [ordinal.range](#) - 设置输出的值域。
- [ordinal.unknown](#) - 设置未知输入域的输出值。
- [ordinal.copy](#) - 创建比例尺的副本。
- [d3.scaleImplicit](#) - 隐域的一个特殊的未知值。
- [d3.scaleBand](#) - 创建序数段比例尺。
- [band](#) - 计算对应于给定的定义域的值域。
- [band.domain](#) - 设置输入的定义域。
- [band.range](#) - 设置输出的值域。
- [band.rangeRound](#) - 设置输出的值域并取整。
- [band.round](#) - 取整。
- [band.paddingInner](#) - 段间距。
- [band.paddingOuter](#) - 外边距。
- [band.padding](#) - 设置间距（段间距和外边距）。
- [band.align](#) - 设置段对齐。
- [band.bandwidth](#) - 获取每段宽度。
- [band.step](#) - 开始相邻段之间的距离。
- [band.copy](#) - 创建比例尺的副本。
- [d3.scalePoint](#) - 创建序数点比例尺。
- [point](#) - 计算对应于给定的定义域的值域。
- [point.domain](#) - 设置输入的定义域。
- [point.range](#) - 设置输出的值域。
- [point.rangeRound](#) - 设置输出的值域并取整。
- [point.round](#) - 取整。
- [point.padding](#) - 外边距。
- [point.align](#) - 设置点对齐。
- [point.bandwidth](#) - 返回0。
- [point.step](#) - 开始相邻点之间的距离。
- [point.copy](#) - 创建比例尺的副本。

- [d3.schemeCategory10](#) - 10种分类颜色。



- [d3.schemeCategory20](#) - 20种分类颜色。



- [d3.schemeCategory20b](#) - 20种分类颜色。



- [d3.schemeCategory20c](#) - 20种分类颜色。



## 选择器

通过选择元素和加入数据来转换DOM。

## 选择元素

- `d3.selection` - 选择根文档元素。
- `d3.select` - 从文档中选择一个元素。
- `d3.selectAll` - 从文档中选择多个元素。
- `selection.select` - 选择每个选中元素的一个后代元素。
- `selection.selectAll` - 选择每个选中元素的多个后代元素。
- `selection.filter` - 基于数据过滤元素。
- `selection.merge` - 合并两个选择。
- `d3.matcher` - 测试一个元素是否匹配选择器。
- `d3.selector` - 选择一个元素。
- `d3.selectorAll` - 选择多个元素。
- `d3.window` - 得到节点的所有者窗口。

## 修改元素

- `selection.attr` - 设置或获取特性。
- `selection.classed` - 获取，添加或移除CSS类。
- `selection.style` - 设置或获取样式。
- `selection.property` - 设置或获取行内属性。
- `selection.text` - 设置或获取文本内容。
- `selection.html` - 设置或获取inner HTML。
- `selection.append` - 创建，添加或选择新的元素。
- `selection.remove` - 移除文档中的元素。
- `selection.sort` - 基于数据给文档中的元素排序。
- `selection.order` - 重排列文档中的元素以匹配选择中的顺序。
- `selection.raise` - 重新排列每个元素为父元素的最后一个子节点。
- `selection.lower` - 重新排列每个元素为父元素的第一个子节点。
- `d3.creator` - 通过名称创建元素。

## 数据绑定

- `selection.data` - 元素和数据绑定。
- `selection.enter` - 获得进入（enter）选择器（数据无元素）。
- `selection.exit` - 获得退出（exit）选择器（元素无数据）。
- `selection.datum` - 获取或设置元素的数据（不绑定）。

## 事件处理

- `selection.on` - 添加或移除事件监听器。
- `selection.dispatch` - 分发自定义事件。
- `d3.event` - 交互中的当前用户事件。
- `d3.customEvent` - 暂时定义一个自定义事件。
- `d3.mouse` - 获取相对给定容器的鼠标位置。
- `d3.touch` - 获取相对给定容器的单点触控位置。
- `d3.touches` - 获取相对给定容器的多点触控位置。

## 控制语句

- `selection.each` - 为每个元素调用一次指定的方法。
- `selection.call` - 选择器调用指定的方法。
- `selection.empty` - 返回是否是空选择。
- `selection.nodes` - 返回所有选中元素的数组。
- `selection.node` - 返回第一个非空元素。
- `selection.size` - 返回元素数量。

## 命名空间

- `d3.namespace` - 限定XML命名空间，如“`xlink:href`”。
- `d3.namespaces` - 内置的XML命名空间。

## 形状

可视化的图形原语。

### 弧

圆形或环形扇区，如饼图或甜甜圈图。

- `d3.arc` - 创建一个新的弧生成器。
- `arc` - 创建给定数据的弧。
- `arc.centroid` - 弧中心。
- `arc.innerRadius` - 设置内径。
- `arc.outerRadius` - 设置外径。
- `arc.cornerRadius` - 设置圆角半径。
- `arc.startAngle` - 设置起始角度。
- `arc.endAngle` - 设置结束角度。
- `arc.padAngle` - 设置相邻弧之间的夹角。
- `arc.padRadius` - 设置线性填充半径。
- `arc.context` - 设置渲染上下文。

### 饼

计算用于展示饼图或甜环形图的必要的角度值。

- `d3.pie` - 创建一个饼生成器。
- `pie` - 计算给定数据集的角度值。
- `pie.value` - 设置值访问器。
- `pie.sort` - 设置排序比较器。
- `pie.sortValues` - 设置排序比较器。
- `pie.startAngle` - 设置整体的起始角度。
- `pie.endAngle` - 设置整体的结束角度。
- `pie.padAngle` - 设置相邻弧间隔角度。

### 线

用于绘制线图的样条曲线或者折线。

- `d3.line` - 创建一个新的线生成器。
- `line` - 生成给定数据的线。
- `line.x` - 设置x访问器。
- `line.y` - 设置y访问器。
- `line.defined` - 设置定义访问器。
- `line.curve` - 设置曲线插值器。
- `line.context` - 设置渲染上下文。
- `d3.radialLine` - 创建一个新的径向线生成器。
- `radialLine` - 生成给定数据的线。
- `radialLine.angle` - 设置角度访问器。
- `radialLine.radius` - 设置半径访问器。
- `radialLine.defined` - 设置定义访问器。
- `radialLine.curve` - 设置曲线插值器。
- `radialLine.context` - 设置渲染上下文。

## 面积

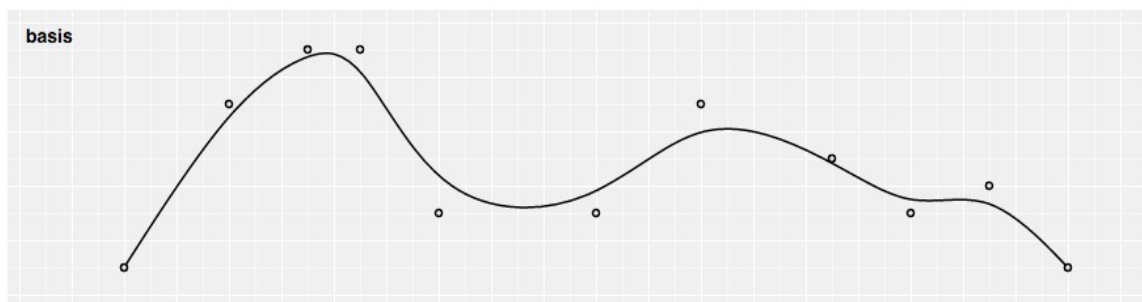
由顶线基线构成，用于面积图。

- [d3.area](#) - 创建一个新的面积生成器。
- [area](#) - 为给定数据集生成面积。
- [area.x](#) - 设置  $x0$  和  $x1$  访问器。
- [area.x0](#) - 设置基线的  $x$  访问器。
- [area.x1](#) - 设置顶线的  $x$  访问器。
- [area.y](#) - 设置  $y0$  和  $y1$  访问器。
- [area.y0](#) - 设置基线的  $y$  访问器。
- [area.y1](#) - 设置顶线的  $y$  访问器。
- [area.defined](#) - 设置定义点访问器。
- [area.curve](#) - 设置曲线插值器。
- [area.context](#) - 设置渲染上下文。
- [area.lineX0](#) - 为区域左边缘得到一条线。
- [area.lineX1](#) - 为区域右边缘得到一条线。
- [area.lineY0](#) - 为区域上边缘得到一条线。
- [area.lineY1](#) - 为区域下边缘得到一条线。
- [d3.radialArea](#) - 创建一个新的径向面积生成器。
- [radialArea](#) - 为给定数据集生成面积。
- [radialArea.angle](#) - 设置起始角度/结束角度访问器。
- [radialArea.startAngle](#) - 设置起始角度访问器。
- [radialArea.endAngle](#) - 设置结束角度访问器。
- [radialArea.radius](#) - 设置内半径/外半径访问器。
- [radialArea.innerRadius](#) - 设置内半径访问器。
- [radialArea.outerRadius](#) - 设置外半径访问器。
- [radialArea.defined](#) - 设置定义点访问器。
- [radialArea.curve](#) - 设置曲线插值器。
- [radialArea.context](#) - 设置渲染上下文。
- [radialArea.lineStartAngle](#) - 为区域起始边缘得到一条线。
- [radialArea.lineEndAngle](#) - 为区域结束边缘得到一条线。
- [radialArea.lineInnerRadius](#) - 为区域内边缘得到一条线。
- [radialArea.lineOuterRadius](#) - 为区域外边缘得到一条线。

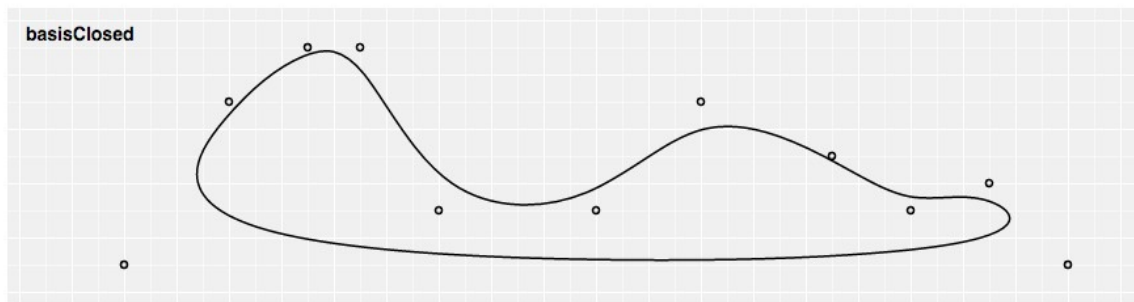
## 曲线

通过在点间插值生成一条曲线。

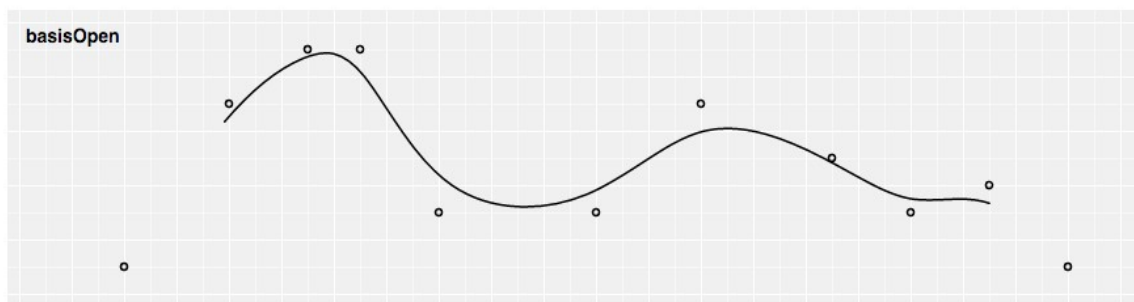
- [d3.curveBasis](#) - 立方基本样条，终点循环。



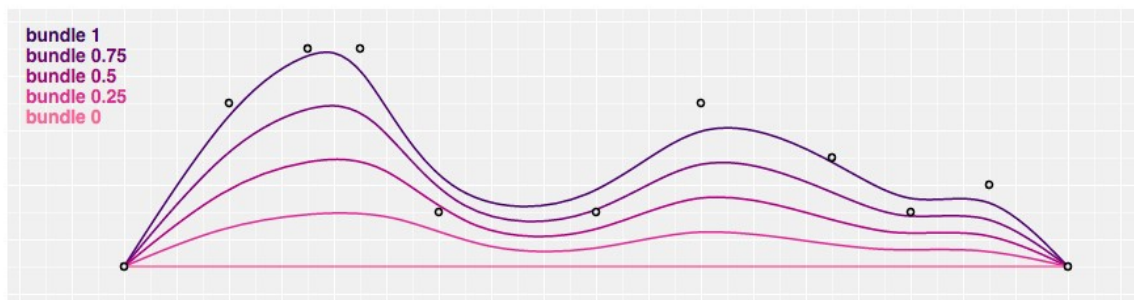
- [d3.curveBasisClosed](#) - 闭合立方基本样条。



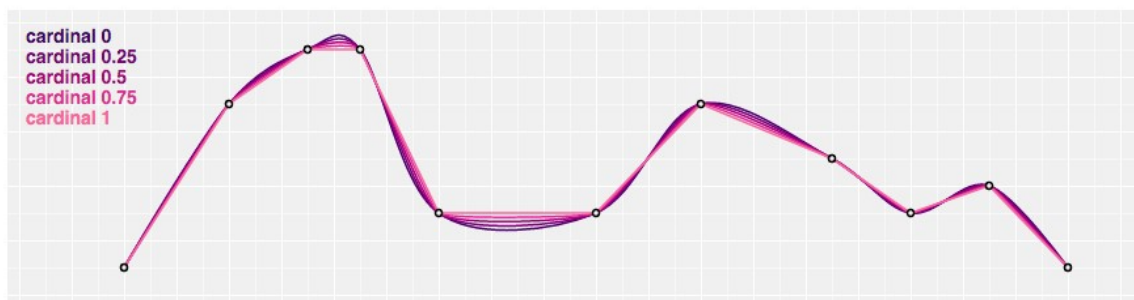
- [d3.curveBasisOpen](#) - 开放立方基本样条。



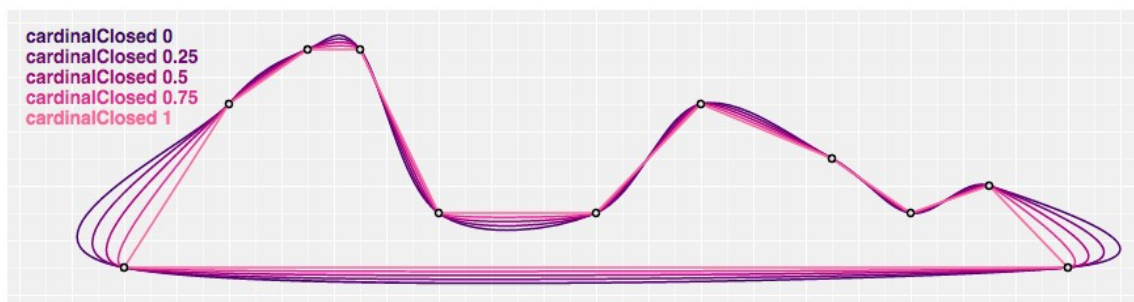
- [d3.curveBundle](#) - 直立方基本样条。



- [d3.curveCardinal](#) - 三次C样条。

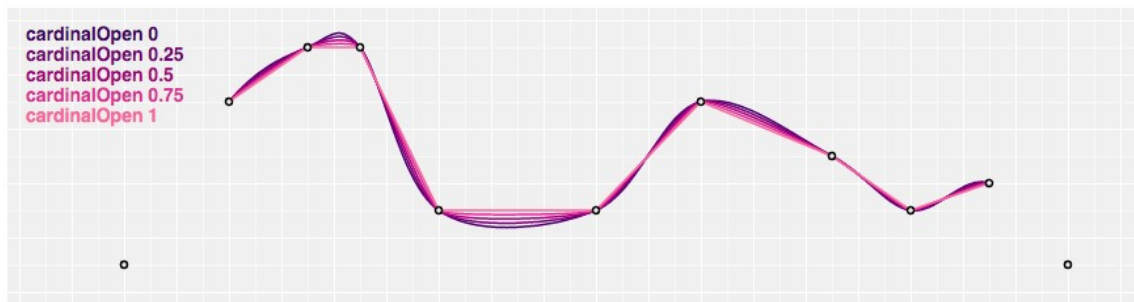


- [d3.curveCardinalClosed](#) - 闭合三次C样条。

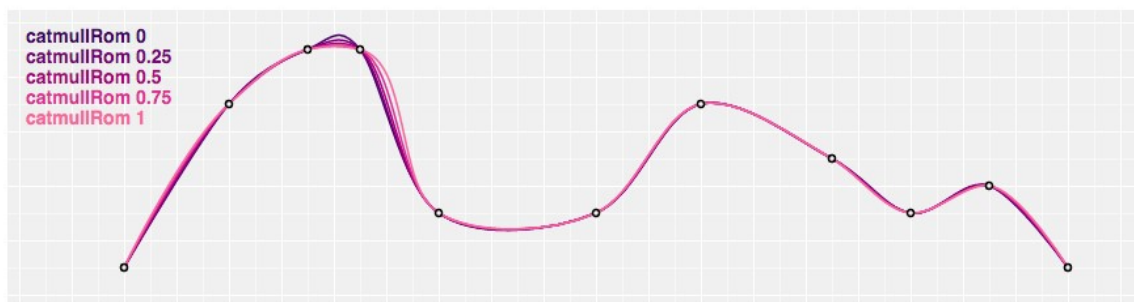


- [d3.curveCardinalOpen](#) - 开放三次C样条。

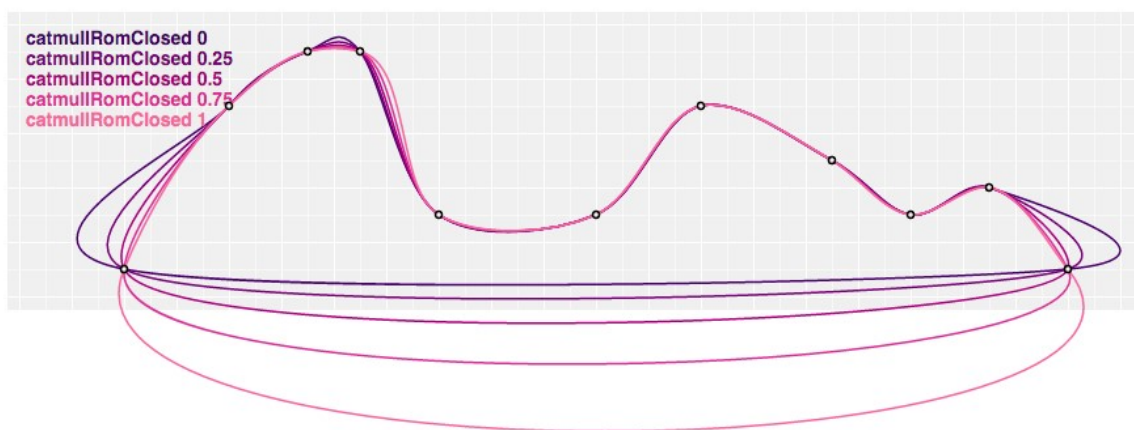




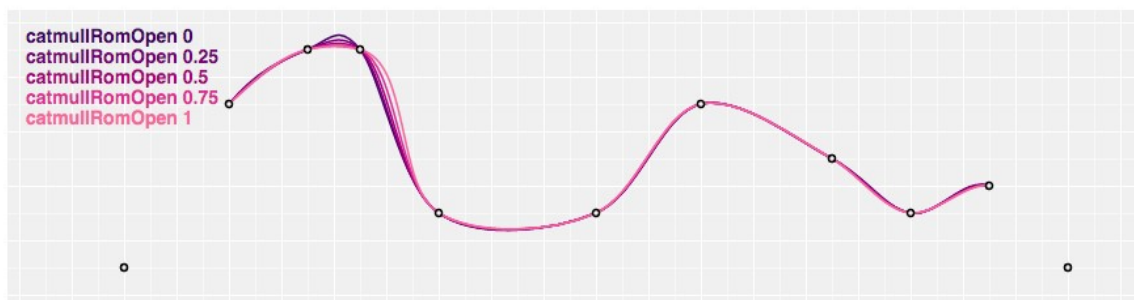
- `cardinal.tension` - 设置基数样条曲线的张力。
- `d3.curveCatmullRom` - 立方Catmull-Rom样条。



- `d3.curveCatmullRomClosed` - 闭合立方Catmull-Rom样条。

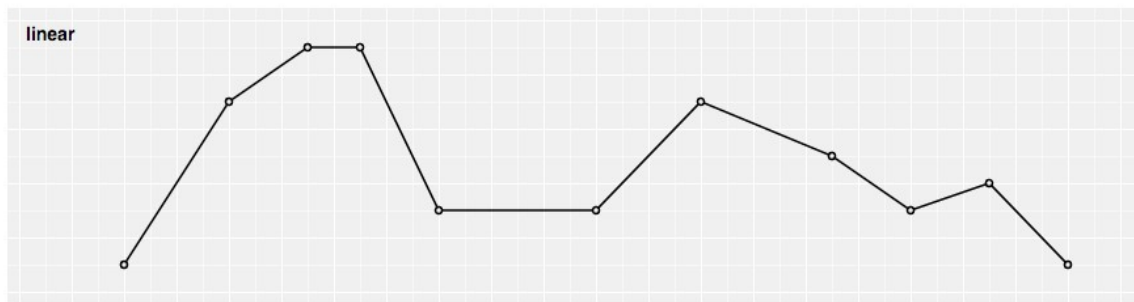


- `d3.curveCatmullRomOpen` - 开放立方Catmull-Rom样条。

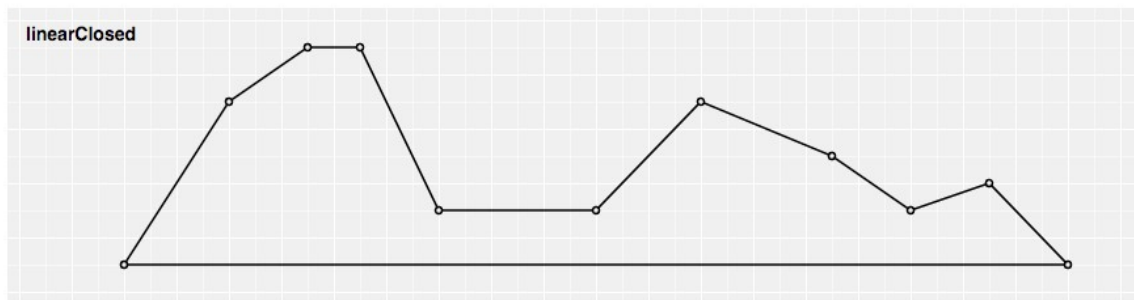


- `catmullRom.alpha` - 设置Catmull-Rom的 $\alpha$ 参数。
- `d3.curveLinear` - 折线。

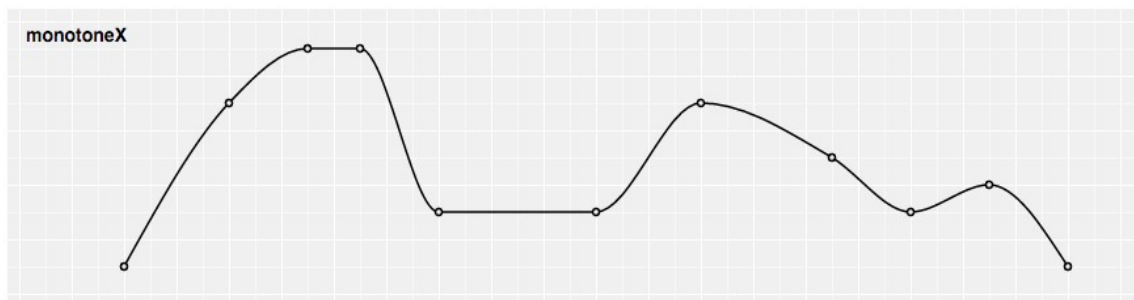




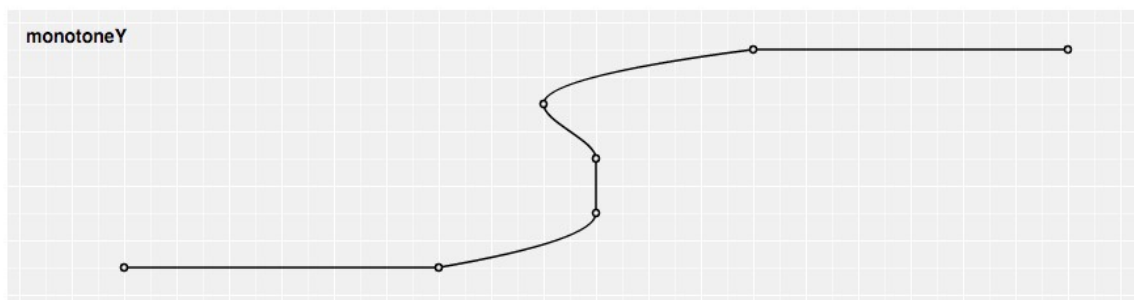
- [d3.curveLinearClosed](#) - 闭合折线。



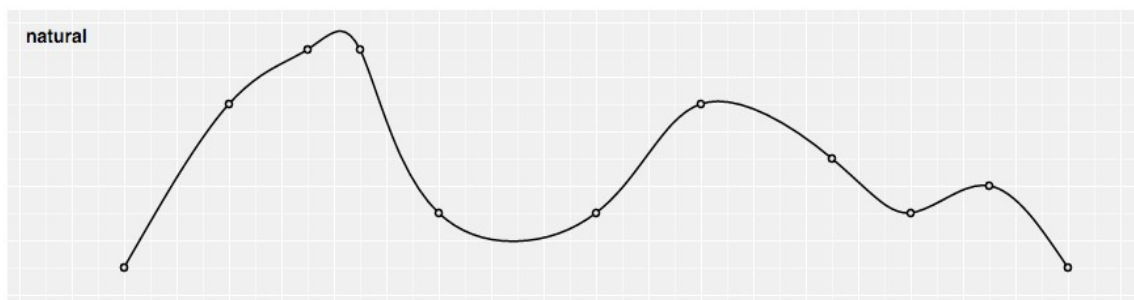
- [d3.curveMonotoneX](#) - 立方样条。假设y是单调的，保持x的单调性。



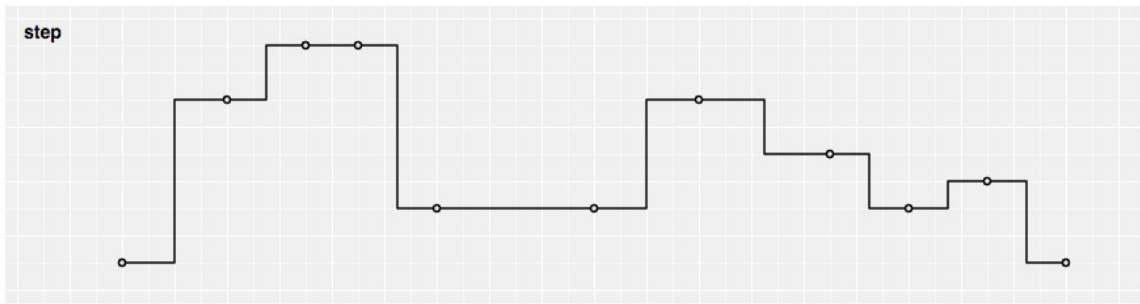
- [d3.curveMonotoneY](#) - 立方样条。假设x是单调的，保持y的单调性。



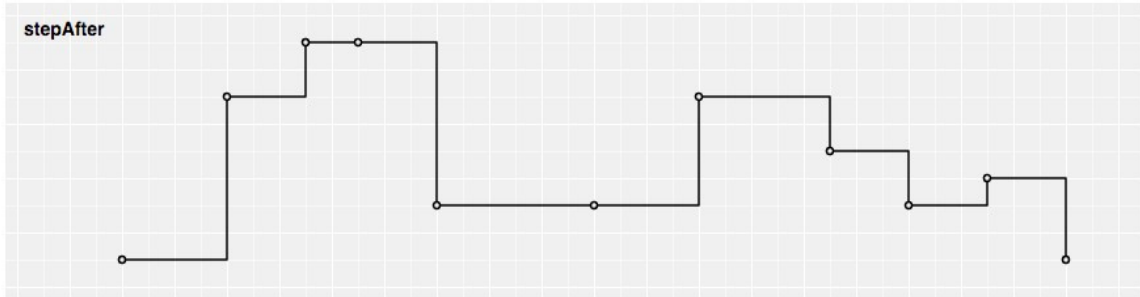
- [d3.curveNatural](#) - 自然三次样条。



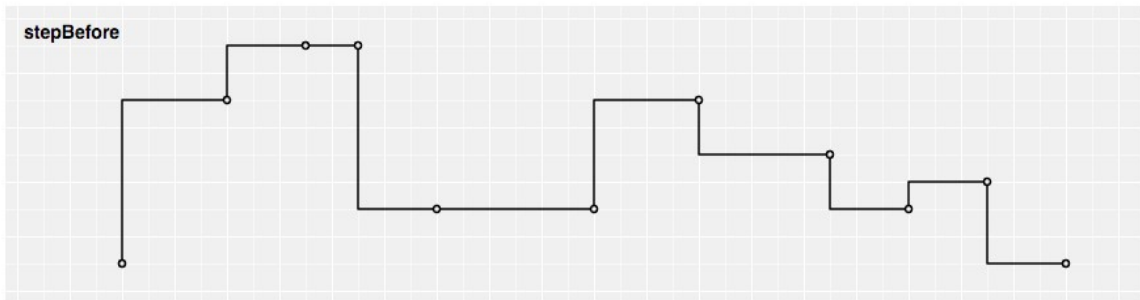
- [d3.curveStep](#) - 分段常值函数。



- [d3.curveStepAfter](#) - 分段常值函数。



- [d3.curveStepBefore](#) - 分段常值函数。



- [curve.areaStart](#) - 开始一个面积片段。
- [curve.areaEnd](#) - 结束一个面积片段。
- [curve.lineStart](#) - 开始一条新的线段。
- [curve.lineEnd](#) - 结束一条新的线段。
- [curve.point](#) - 给当前线段加一个点。

## 符号

一些内置形状，用于散点图。

- [d3.symbol](#) - 创建一个新的形状生成器。
- [symbol](#) - 为给定数据生成符号。
- [symbol.type](#) - 设置符号类型。
- [symbol.size](#) - 设置符号尺寸。
- [symbol.context](#) - 设置渲染上下文。
- [d3.symbols](#) - 符号类型数组。
- [d3.symbolCircle](#) - 圆形。
- [d3.symbolCross](#) - 十字。
- [d3.symbolDiamond](#) - 菱形。
- [d3.symbolSquare](#) - 方形。
- [d3.symbolStar](#) - 五角星。
- [d3.symbolTriangle](#) - 上三角。
- [d3.symbolWye](#) - Y形。
- [symbolType.draw](#) - 在给定的容器中绘制符号。

## 堆叠

将形状堆叠起来，就像分段条形图那样。

- [d3.stack](#) - 创建一个新的堆叠生成器。
- [stack](#) - 为给定数据生成堆叠数据。
- [stack.keys](#) - 设置键访问器。
- [stack.value](#) - 设置值访问器。
- [stack.order](#) - 设置排序访问器。
- [stack.offset](#) - 设置偏移访问器。
- [d3.stackOrderAscending](#) - 将最小值放在底部。
- [d3.stackOrderDescending](#) - 将最大值放在底部。
- [d3.stackOrderInsideOut](#) - 将最大值放在中部。
- [d3.stackOrderNone](#) - 使用给定的系列顺序。
- [d3.stackOrderReverse](#) - 系列给定的系列相反的顺序。
- [d3.stackOffsetExpand](#) - 标准化为0=1之间。
- [d3.stackOffsetNone](#) - 应用零基准。
- [d3.stackOffsetSilhouette](#) - 将流图居中在0附近。
- [d3.stackOffsetWiggle](#) - 流图最小摆动。

## 时间格式化

解析和格式化时间。

- [d3.timeFormat](#) - [enUs.format](#)别名。
- [d3.timeParse](#) - [enUs.parse](#)别名。
- [d3.utcFormat](#) - [enUs.utcFormat](#)别名。
- [d3.utcFormat](#) - [enUs.utcParse](#)别名。
- [d3.isoFormat](#) - ISO 8601 UTC格式化。
- [d3.isoParse](#) - ISO 8601 UTC解析器。
- [d3.timeFormatLocale](#) - 自定义本地化。
- [locale.format](#) - 创建一个时间格式化器。
- [locale.parse](#) - 创建一个时间解析器。
- [locale.utcFormat](#) - 创建一个UTC格式化器。
- [locale.utcParse](#) - 创建一个UTC解析器。

## 时间间隔

时间计算。

- [d3.timeInterval](#) - 实现一个新的自定义时间间隔。
- [interval](#) - [interval.floor](#)的别名。
- [interval.floor](#) - 下取整到最近的边界。
- [interval.round](#) - 四舍五入到最近的边界。
- [interval.ceil](#) - 上取整到最近的边界。
- [interval.offset](#) - 抵消一些日期的间隔。
- [interval.range](#) - 生成的日期范围区间边界。
- [interval.filter](#) - 创建一个这个区间的过滤子集。
- [interval.every](#) - 创建一个这个区间的过滤子集。
- [interval.count](#) - 计算两个时间之间的间隔。
- [d3.timeMillisecond](#), [d3.utcMillisecond](#) - 毫秒间隔。
- [d3.timeMilliseconds](#), [d3.utcMilliseconds](#) - [millisecond.range](#)的别名。
- [d3.timeSecond](#), [d3.utcSecond](#) - 秒间隔。
- [d3.timeSeconds](#), [d3.utcSeconds](#) - [second.range](#)的别名。
- [d3.timeMinute](#), [d3.utcMinute](#) - 分钟间隔。
- [d3.timeMinutes](#), [d3.utcMinutes](#) - [minute.range](#)的别名。
- [d3.timeHour](#), [d3.utcHour](#) - 小时间隔。

- [d3.timeHours](#), [d3.utcHours](#) - [hour.range](#)的别名。
- [d3.timeDay](#), [d3.utcDay](#) - 天间隔。
- [d3.timeDays](#), [d3.utcDays](#) - [day.range](#)的别名。
- [d3.timeWeek](#), [d3.utcWeek](#) - 星期天的别名。
- [d3.timeWeeks](#), [d3.utcWeeks](#) - [week.range](#)的别名。
- [d3.timeSunday](#), [d3.utcSunday](#) - 星期间隔（周日开始）。
- [d3.timeSundays](#), [d3.utcSundays](#) - [sunday.range](#)的别名。
- [d3.timeMonday](#), [d3.utcMonday](#) - 星期间隔（周一开始）。
- [d3.timeMondays](#), [d3.utcMondays](#) - [monday.range](#)的别名。
- [d3.timeTuesday](#), [d3.utcTuesday](#) - 星期间隔（周二开始）。
- [d3.timeTuesdays](#), [d3.utcTuesdays](#) - [tuesday.range](#)的别名。
- [d3.timeWednesday](#), [d3.utcWednesday](#) - 星期间隔（周三开始）。
- [d3.timeWednesdays](#), [d3.utcWednesdays](#) - [wednesday.range](#)的别名。
- [d3.timeThursday](#), [d3.utcThursday](#) - 星期间隔（周四开始）。
- [d3.timeThursdays](#), [d3.utcThursdays](#) - [thursday.range](#)的别名。
- [d3.timeFriday](#), [d3.utcFriday](#) - 星期间隔（周五开始）。
- [d3.timeFridays](#), [d3.utcFridays](#) - [friday.range](#)的别名。
- [d3.timeSaturday](#), [d3.utcSaturday](#) - 星期间隔（周六开始）。
- [d3.timeSaturdays](#), [d3.utcSaturdays](#) - [saturday.range](#)的别名。
- [d3.timeMonth](#), [d3.utcMonth](#) - 月间隔。
- [d3.timeMonths](#), [d3.utcMonths](#) - [month.range](#)的别名。
- [d3.timeYear](#), [d3.utcYear](#) - 年间隔。
- [d3.timeYears](#), [d3.utcYears](#) - [year.range](#)的别名。

## 定时器

管理成千上万的并发动画的队列。

- [d3.now](#) - 获取当前高分辨率时间。
- [d3.timer](#) - 安排一个新的定时器。
- [timer.restart](#) - 重置定时器的开始时间和回调。
- [timer.stop](#) - 停止定时器。
- [d3.timerFlush](#) - 立即执行任何合格的定时器。
- [d3.timeout](#) - 安排一个定时器在首次回调时停止。
- [d3.interval](#) - 安排一个可配置时期的定时器。

## 过渡

[selections](#)的动画过渡。

- [selection.transition](#) - 为选定的元素安排一个过渡。
- [selection.interrupt](#) - 中断和取消选定元素的过渡。
- [d3.transition](#) - 为文档根元素安排一个过渡。
- [transition.select](#) - 为选定的元素安排一个过渡。
- [transition.selectAll](#) - 为选定的所有元素安排一个过渡。
- [transition.filter](#) - 基于数据过滤元素。
- [transition.merge](#) - 两个过渡合并。
- [transition.selection](#) - 返回一个过渡选择。
- [transition.transition](#) - 链式过渡。
- [transition.call](#) - 过渡调用一个函数。
- [transition.nodes](#) - 返回选定元素数组。
- [transition.node](#) - 返回第一个非空元素。
- [transition.size](#) - 返回元素数。
- [transition.empty](#) - 如果过渡为空返回true。
- [transition.each](#) - 为每个元素调用一个函数。

- [transition.on](#) - 添加或删除过渡的事件侦听器。
- [transition.attr](#) - 使用默认插值器过渡属性。
- [transition.attrTween](#) - 使用自定义插值器过渡属性。
- [transition.style](#) - 使用默认插值器过渡样式。
- [transition.styleTween](#) - 使用自定义插值器过渡样式。
- [transition.text](#) - 在过渡开始时设置文本内容。
- [transition.remove](#) - 过渡结束后移除元素。
- [transition.tween](#) - 在过渡期间运行自定义代码。
- [transition.delay](#) - 指定每个元素的延迟时间（以毫秒为单位）。
- [transition.duration](#) - 指定每个元素的持续时间（以毫秒为单位）。
- [transition.ease](#) - 指定缓动函数。
- [d3.active](#) - 对于一个给定的节点选择活跃中的过渡。
- [d3.interrupt](#) - 中断过渡？。

## 冯罗诺

为给定的点集计算冯罗诺图。

- [d3.voronoi](#) - 创建一个新的冯罗诺生成器。
- [voronoi](#) - 为给定的点集计算泰森多边形。
- [voronoi.polygons](#) - 为给定的点集计算冯罗诺多边形。
- [voronoi.triangles](#) - 为给定的点集计算Delaunay三角形。
- [voronoi.links](#) - 为给定的点集计算Delaunay连接。
- [voronoi.x](#) - 设置x访问器。
- [voronoi.y](#) - 设置y访问器。
- [voronoi.extent](#) - 设置点的观察范围。
- [voronoi.size](#) - 设置点的观察范围。

## 缩放

使用鼠标或触摸平移和缩放SVG, HTML 或 Canvas。

- [d3.zoom](#) - 创建缩放行为。
- [zoom](#) - 将缩放行为应用到选中的元素上。
- [zoom.transform](#) - 改变选中元素的仿射变换。
- [zoom.translateBy](#) - 平移。
- [zoom.scaleBy](#) - 缩放。
- [zoom.scaleTo](#) - 缩放到。
- [zoom.filter](#) - 控制那个输入事件初始化缩放。
- [zoom.extent](#) - 设置viewport的尺寸。
- [zoom.scaleExtent](#) - 设置缩放范围。
- [zoom.translateExtent](#) - 设置平移范围。
- [zoom.duration](#) - 设置缩放过渡的延时时间。
- [zoom.on](#) - 监听缩放事件。
- [d3.zoomTransform](#) - 获取给定元素的仿射变换。
- [transform.scale](#) - 缩放指定大小。
- [transform.translate](#) - 平移指定大小。
- [transform.apply](#) - 将变换应用到给定的点。
- [transform.applyX](#) - 将变换应用到给定的x坐标。
- [transform.applyY](#) - 将变换应用到给定的y坐标。
- [transform.invert](#) - 解除给定点的变化。
- [transform.invertX](#) - 解除给定x坐标的变化。
- [transform.invertY](#) - 解除给定y坐标的变化。
- [transform.rescaleX](#) - 应用平移到指定的x比例尺的定义域。
- [transform.rescaleY](#) - 应用平移到指定的y比例尺的定义域。

- [transform.toString](#) - 将变换转换成SVG变换字符串。
- [d3.zoomIdentity](#) - 恒等变换。

