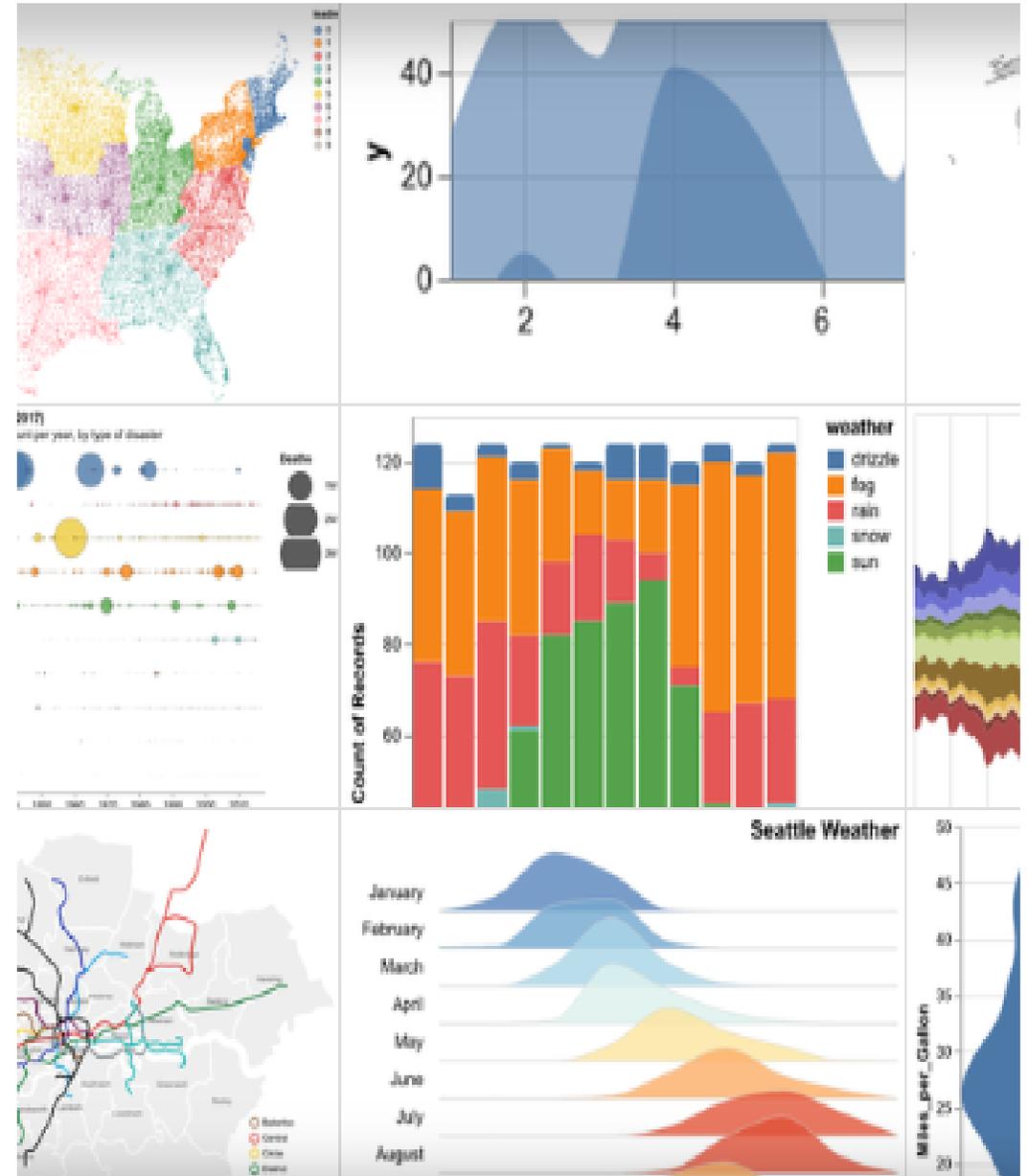


5 • SCALES FUNCTIONS

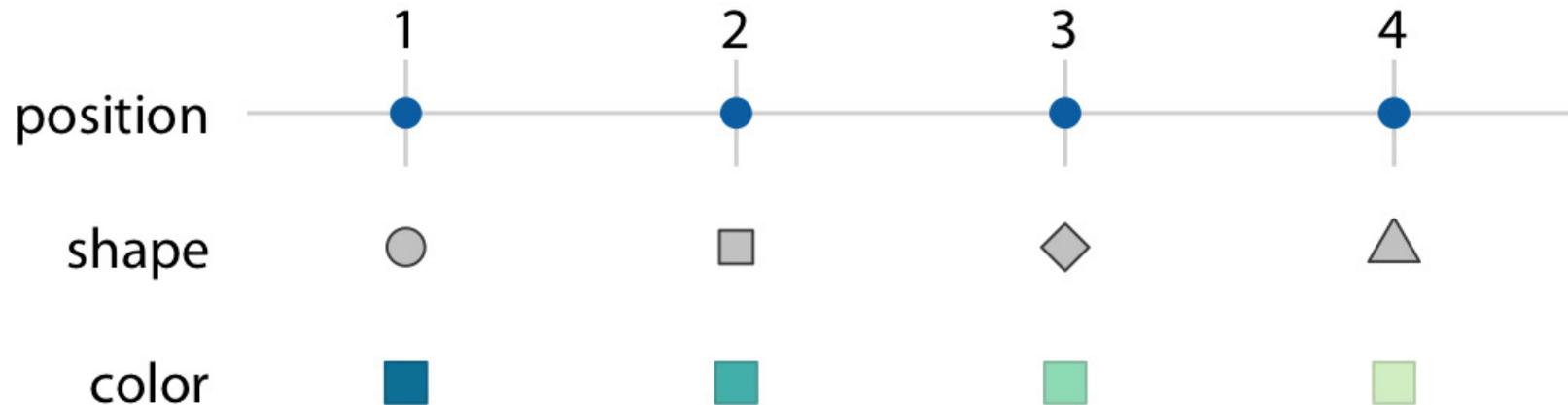
TEACHER
Salvatore Rinzivillo

University of Pisa
Department of Computer Science
Course: Visual Analytics (602AA)
Academic Year: 2024/2025

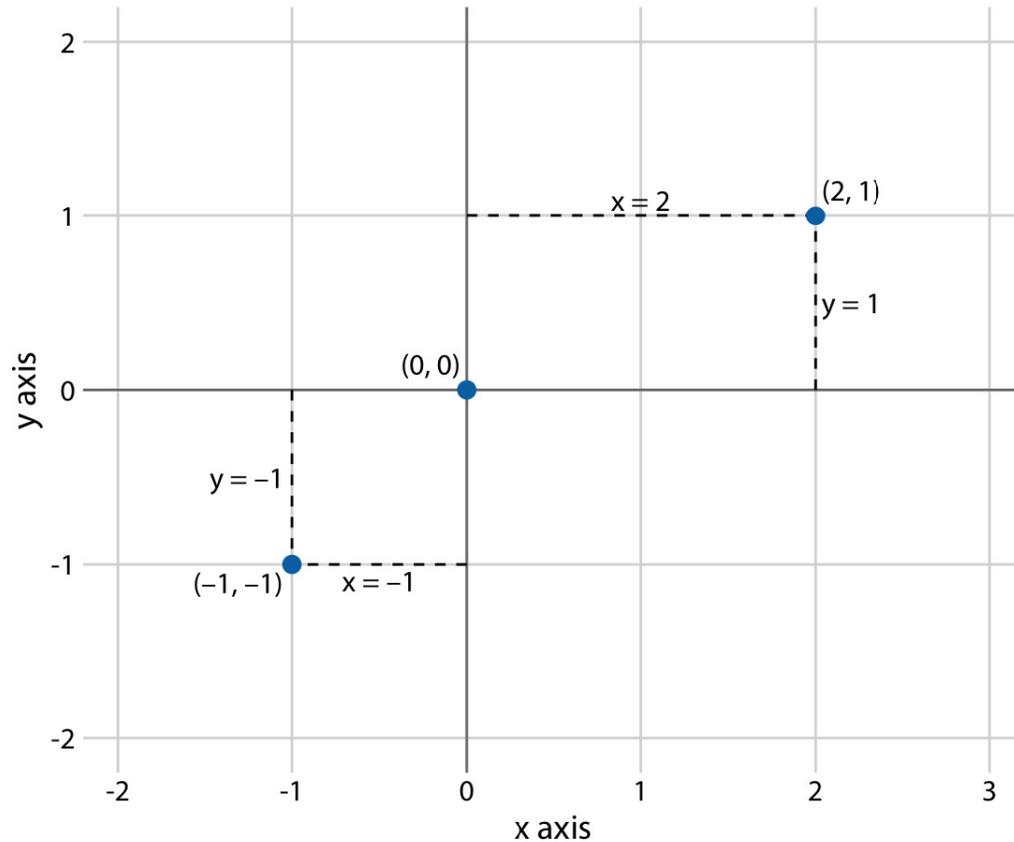


Map Data to Visual Variables

- We specify a scaling function to map data values to the visual representation
- A **scale** is a unique mapping between data and visual representation
- Scales are **functions** that map from an *input* domain to an *output* range

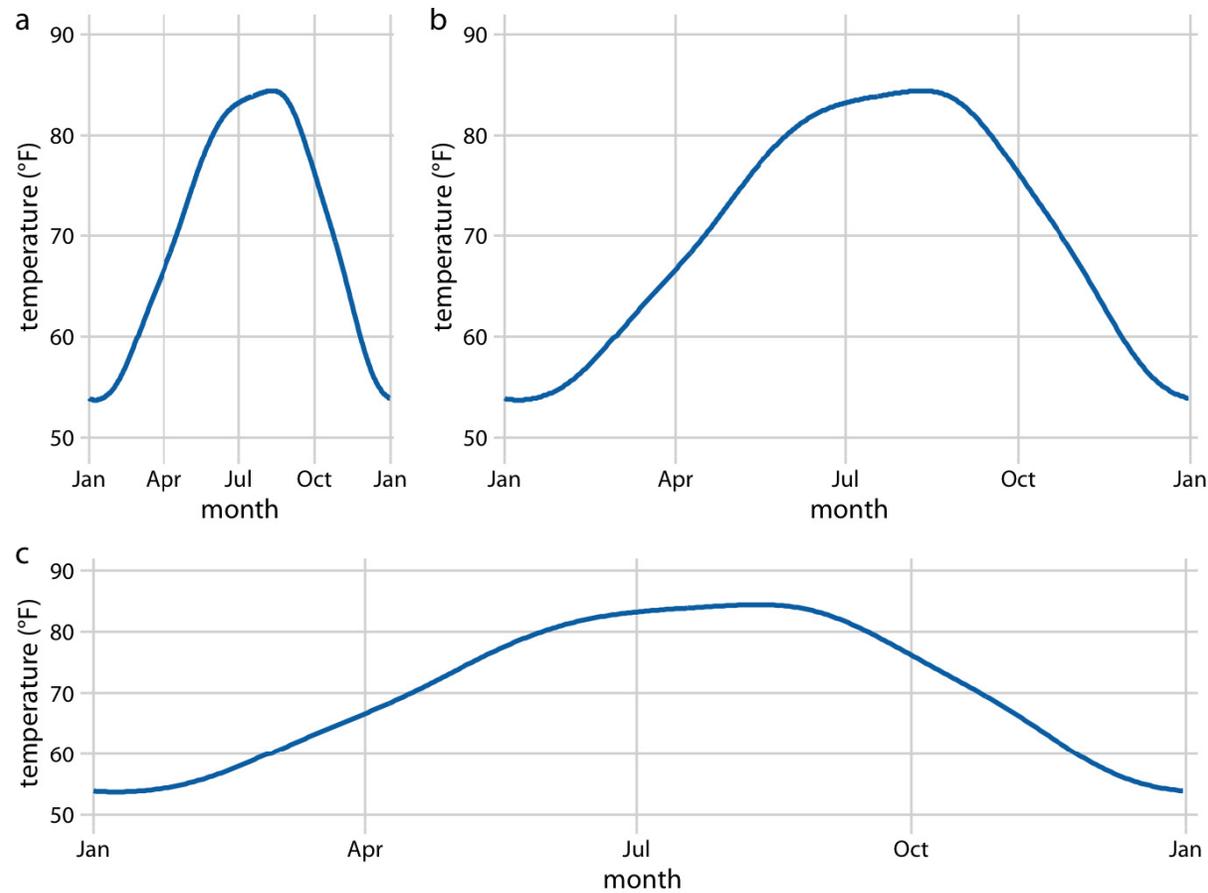


Positional Scales: Axis



- Axis are at the base of many scientific plots
- Cartesian coordinate systems are composed of two orthogonal axis
- Values are positioned proportionally on the axes

Cartesian Diagram with Different Scales



Cartesian Diagram with Different Scales

CHART THAT TRUMP SIGNED AND SENT TO HIS FOLLOWERS

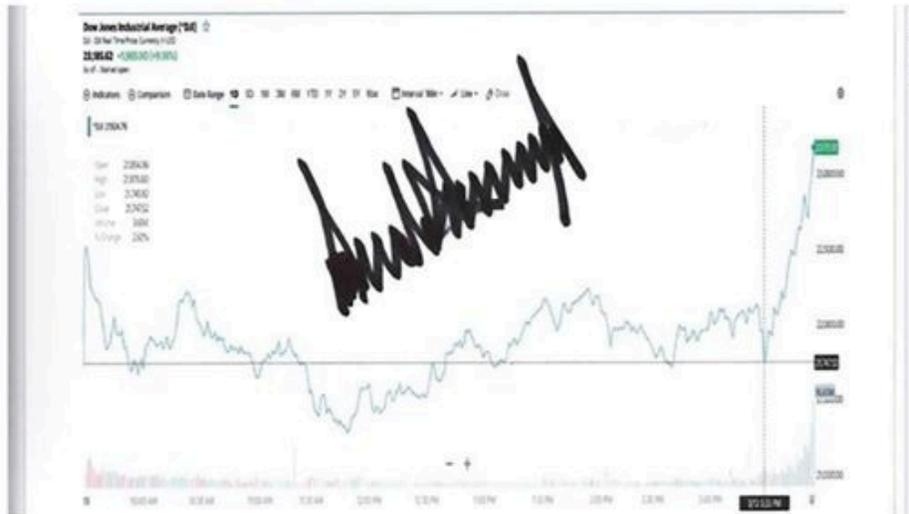
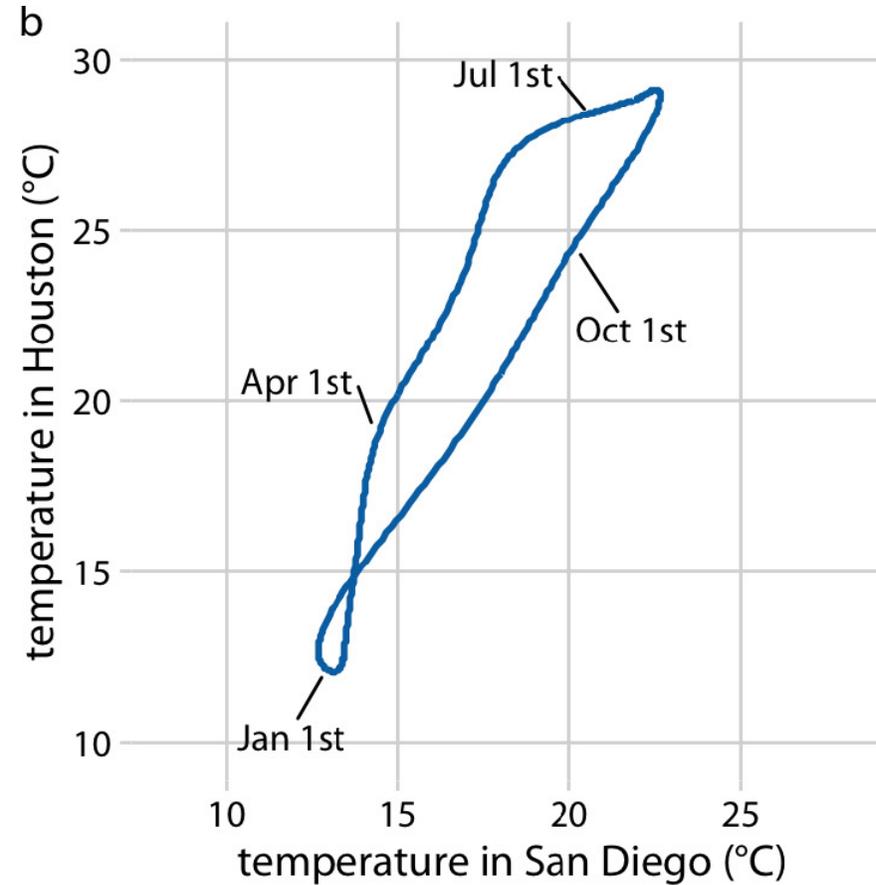
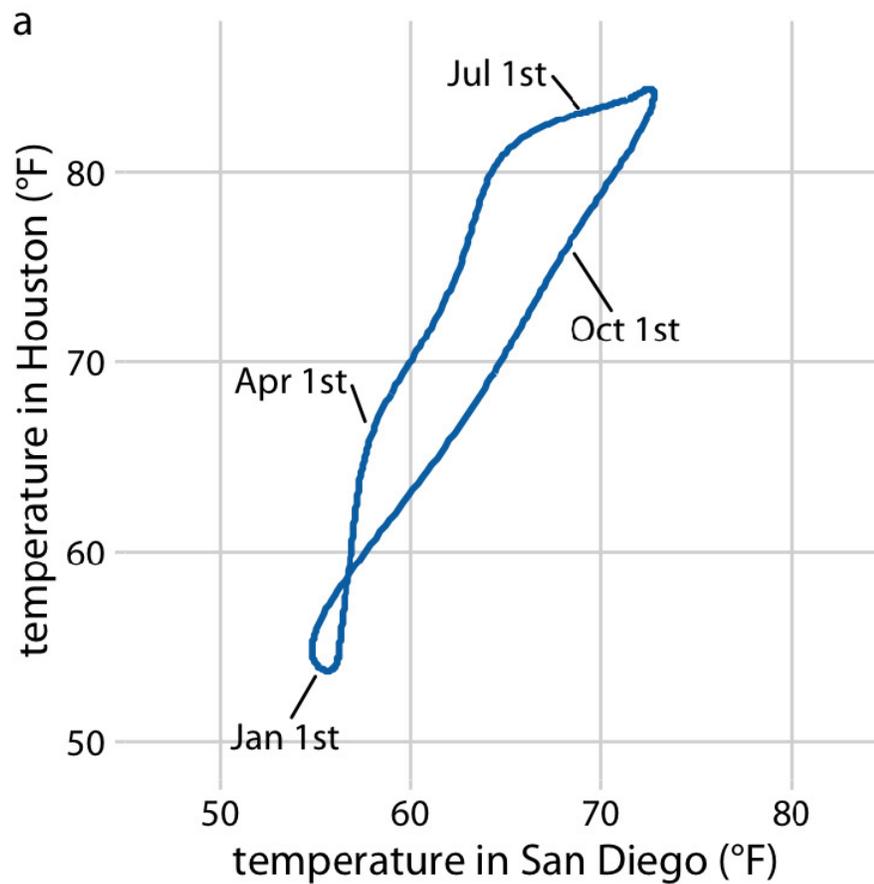


CHART THAT I SIGNED AND SENT TO MY FOLLOWERS

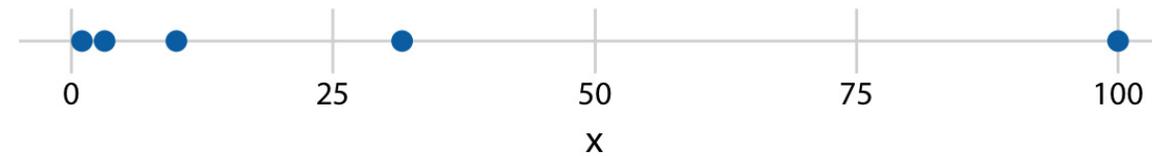


Cartesian Axes with Same Scale

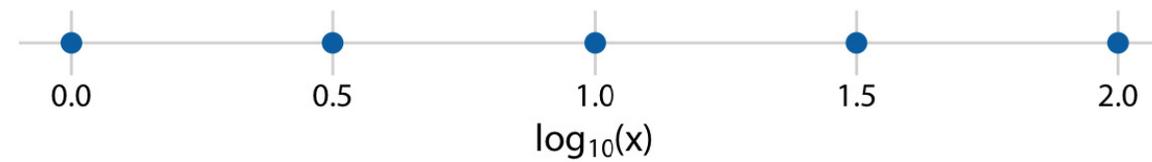


Non Linear Axes

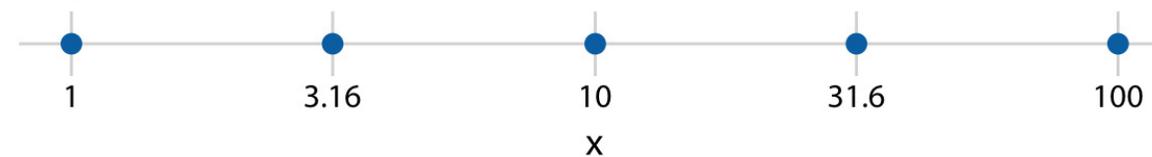
original data, linear scale



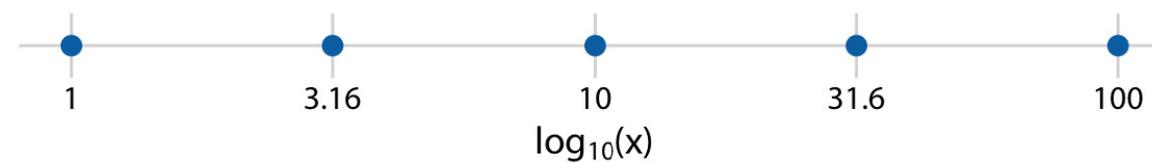
log-transformed data, linear scale



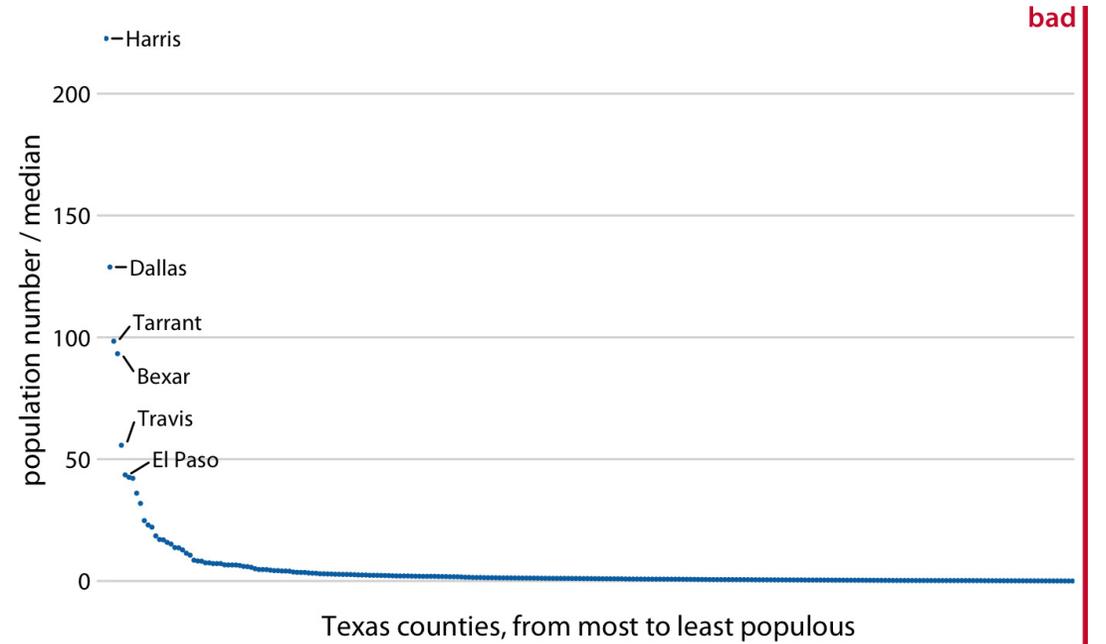
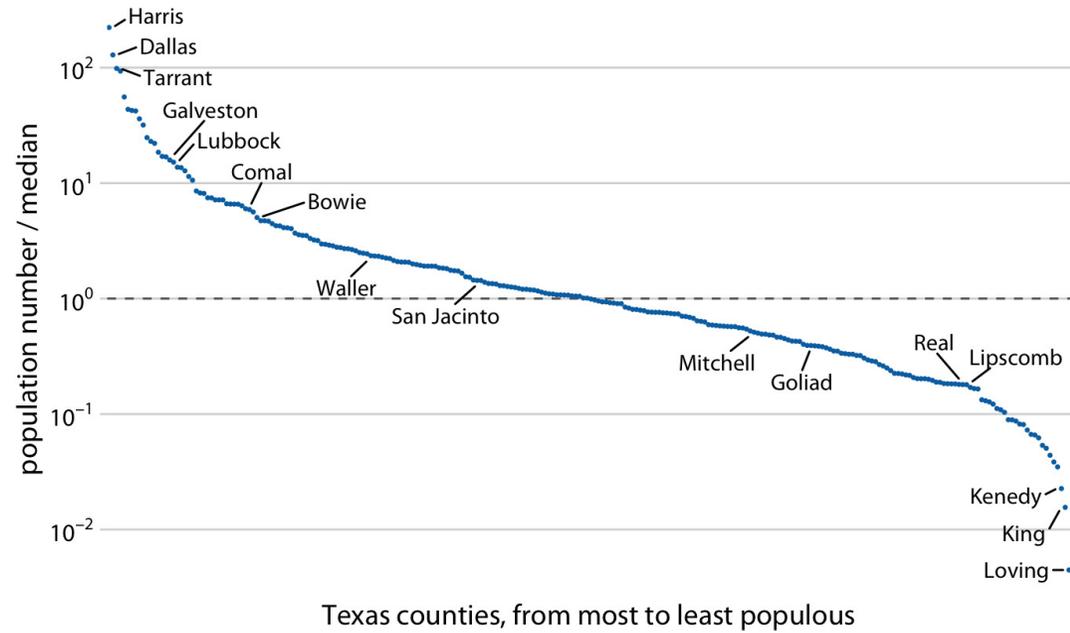
original data, logarithmic scale



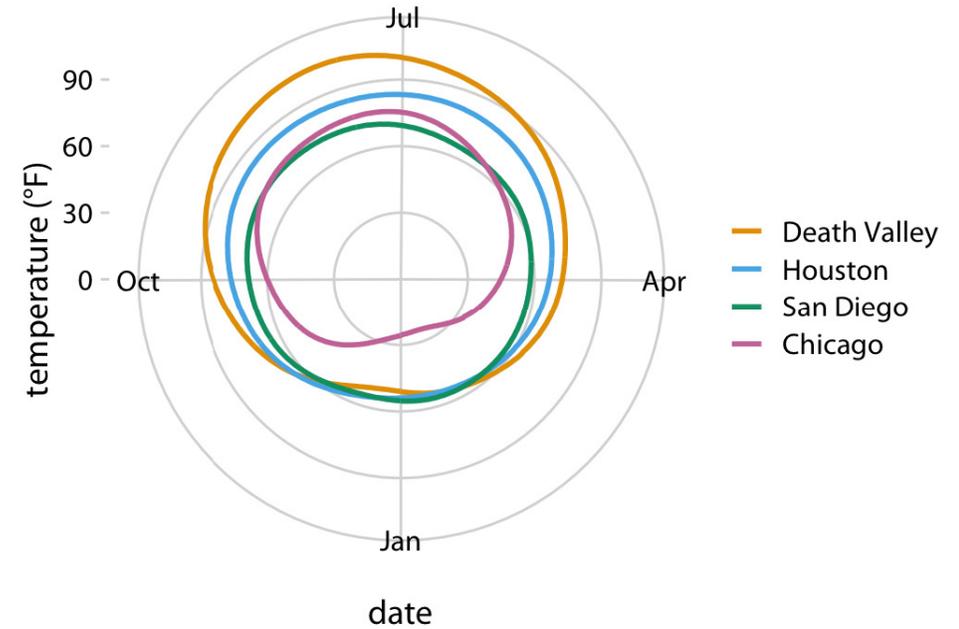
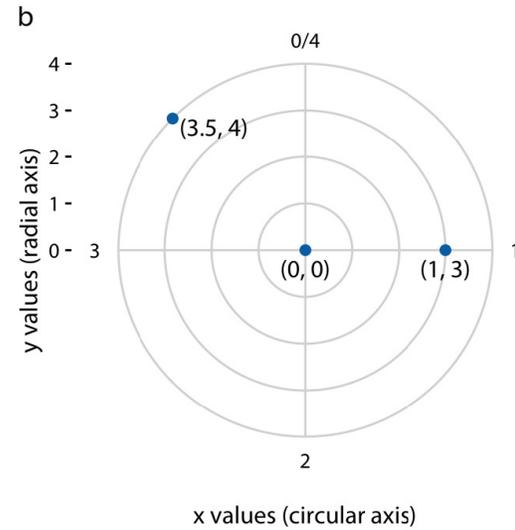
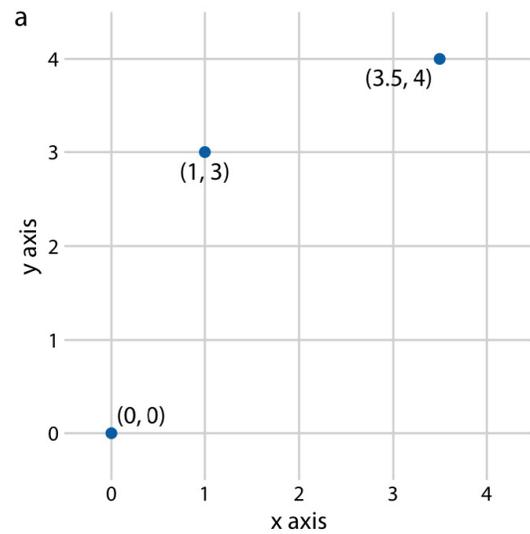
logarithmic scale with incorrect axis title



Non Linear Axes



Curved Axes



Example

Adapted from: Fundamental of Data Visualization, Claus O. Wilke, O'Reilly editor

Table 2.2: First 12 rows of a dataset listing daily temperature normals for four weather stations. Data source: NOAA.

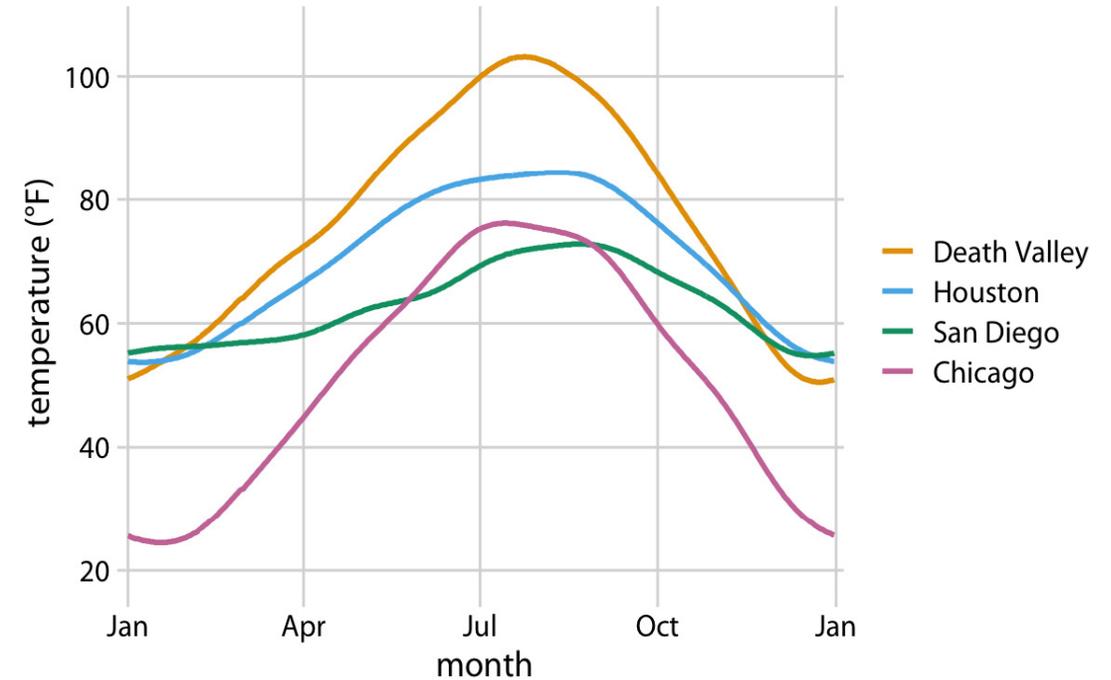
Month	Day	Location	Station ID	Temperature
Jan	1	Chicago	USW00014819	25.6
Jan	1	San Diego	USW00093107	55.2
Jan	1	Houston	USW00012918	53.9
Jan	1	Death Valley	USC00042319	51.0
Jan	2	Chicago	USW00014819	25.5
Jan	2	San Diego	USW00093107	55.3
Jan	2	Houston	USW00012918	53.8
Jan	2	Death Valley	USC00042319	51.2
Jan	3	Chicago	USW00014819	25.3
Jan	3	San Diego	USW00093107	55.3
Jan	3	Death Valley	USC00042319	51.3
Jan	3	Houston	USW00012918	53.8

Ordinal Ordinal Nominal Nominal Quantitative

Scales Functions

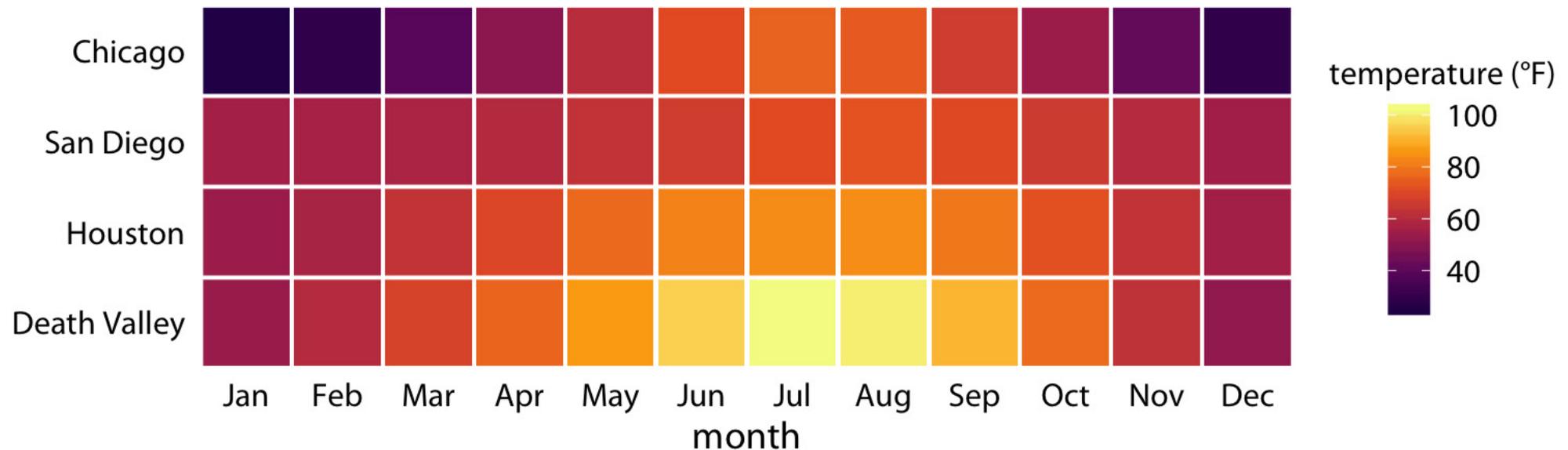
Example

- Temperature (quantitative) on a linear axis (y)
- Month and day (ordinal) on a linear axis (x)
- City (nominal) on a color hue scale



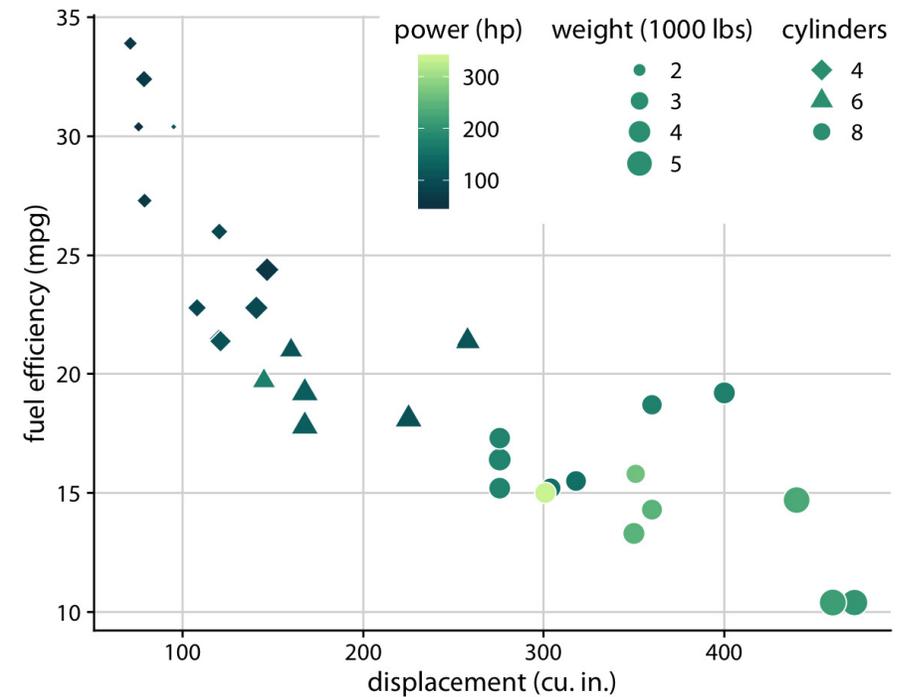
Example

- Month (ordinal) on a ordinal axis (x)
- City (nominal) on a ordinal axis (y) (order determined on sum of temperatures on the line)
- Temperature (quantitative) on a color scale

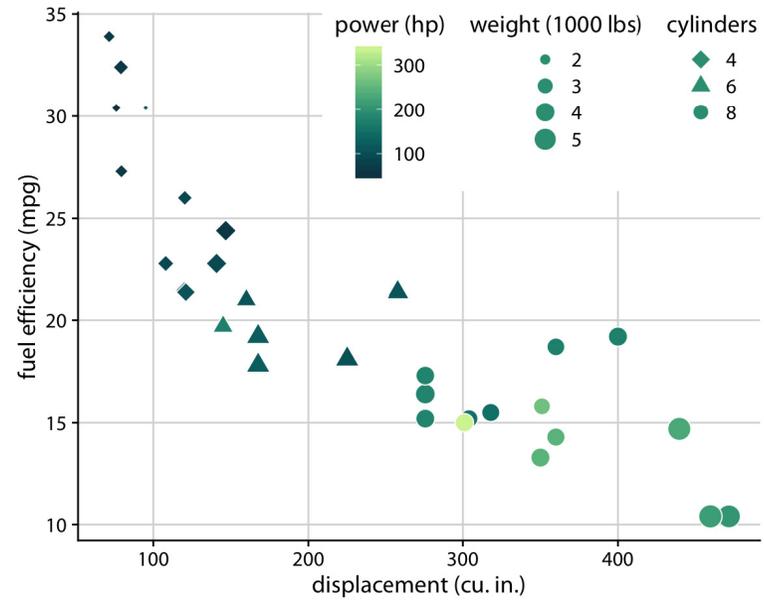


Example

- Displacement (quantitative) on linear axis (x)
- Fuel efficiency (quantitative) on linear axis (y)
- Power (quantitative) on lineal color scale
- Weight (quantitative -> ordinal) on linear squared size scale
- Cylinders (ordinal -> nominal) on shape scale



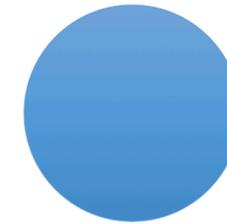
Size Scale Issues



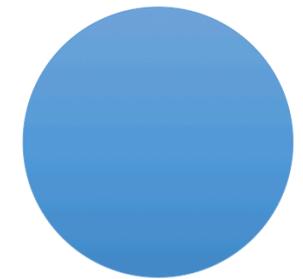
$r=2$
 $A = 4 \cdot \pi$



$r=3$
 $A = 9 \cdot \pi$



$r=4$
 $A = 16 \cdot \pi$



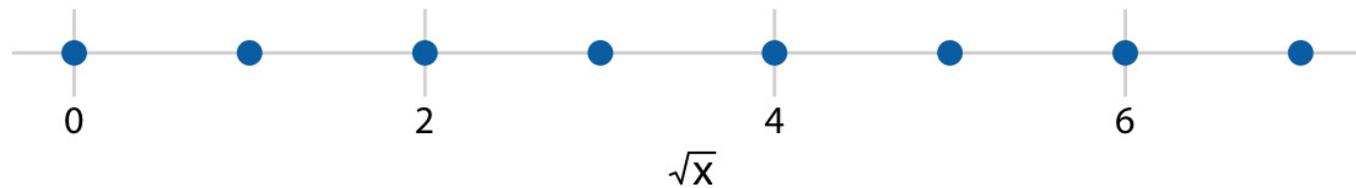
$r=5$
 $A = 25 \cdot \pi$

Non Linear Axes

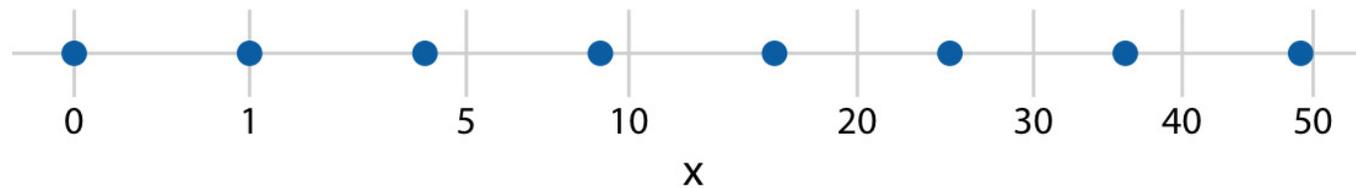
original data, linear scale



square-root-transformed data, linear scale

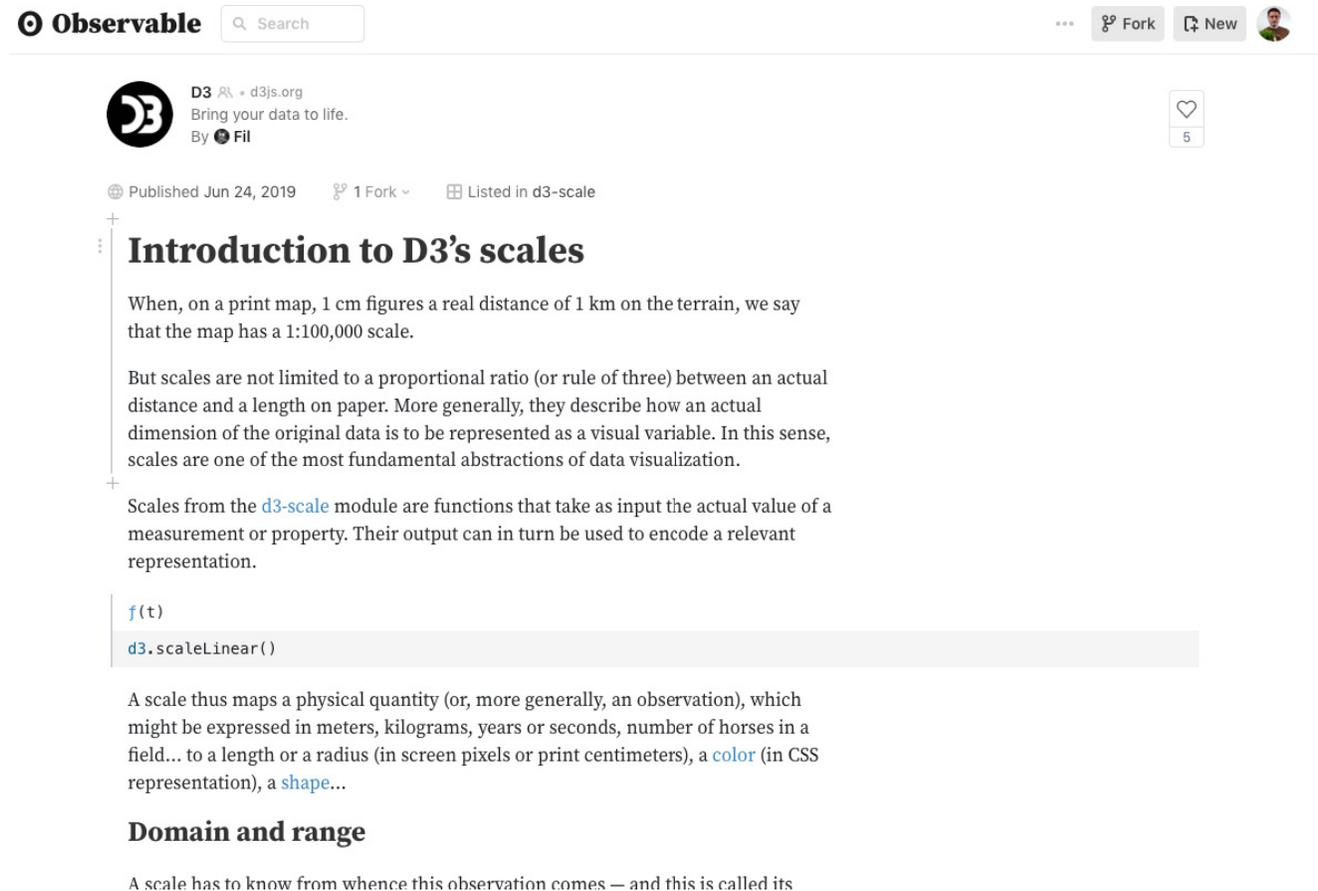


original data, square-root scale



ObservableHQ – Introduction to D3.js Scales

<https://observablehq.com/@d3/introduction-to-d3s-scales>



Observable

... Fork New 

 **D3** • d3js.org
Bring your data to life.
By  **Fil**

Published Jun 24, 2019 1 Fork Listed in d3-scale  5

Introduction to D3's scales

When, on a print map, 1 cm figures a real distance of 1 km on the terrain, we say that the map has a 1:100,000 scale.

But scales are not limited to a proportional ratio (or rule of three) between an actual distance and a length on paper. More generally, they describe how an actual dimension of the original data is to be represented as a visual variable. In this sense, scales are one of the most fundamental abstractions of data visualization.

Scales from the [d3-scale](#) module are functions that take as input the actual value of a measurement or property. Their output can in turn be used to encode a relevant representation.

```
f(t)  
d3.scaleLinear()
```

A scale thus maps a physical quantity (or, more generally, an observation), which might be expressed in meters, kilograms, years or seconds, number of horses in a field... to a length or a radius (in screen pixels or print centimeters), a [color](#) (in CSS representation), a [shape](#)...

Domain and range

A scale has to know from whence this observation comes — and this is called its

Scales Functions

ObservableHQ – Discrete Scales

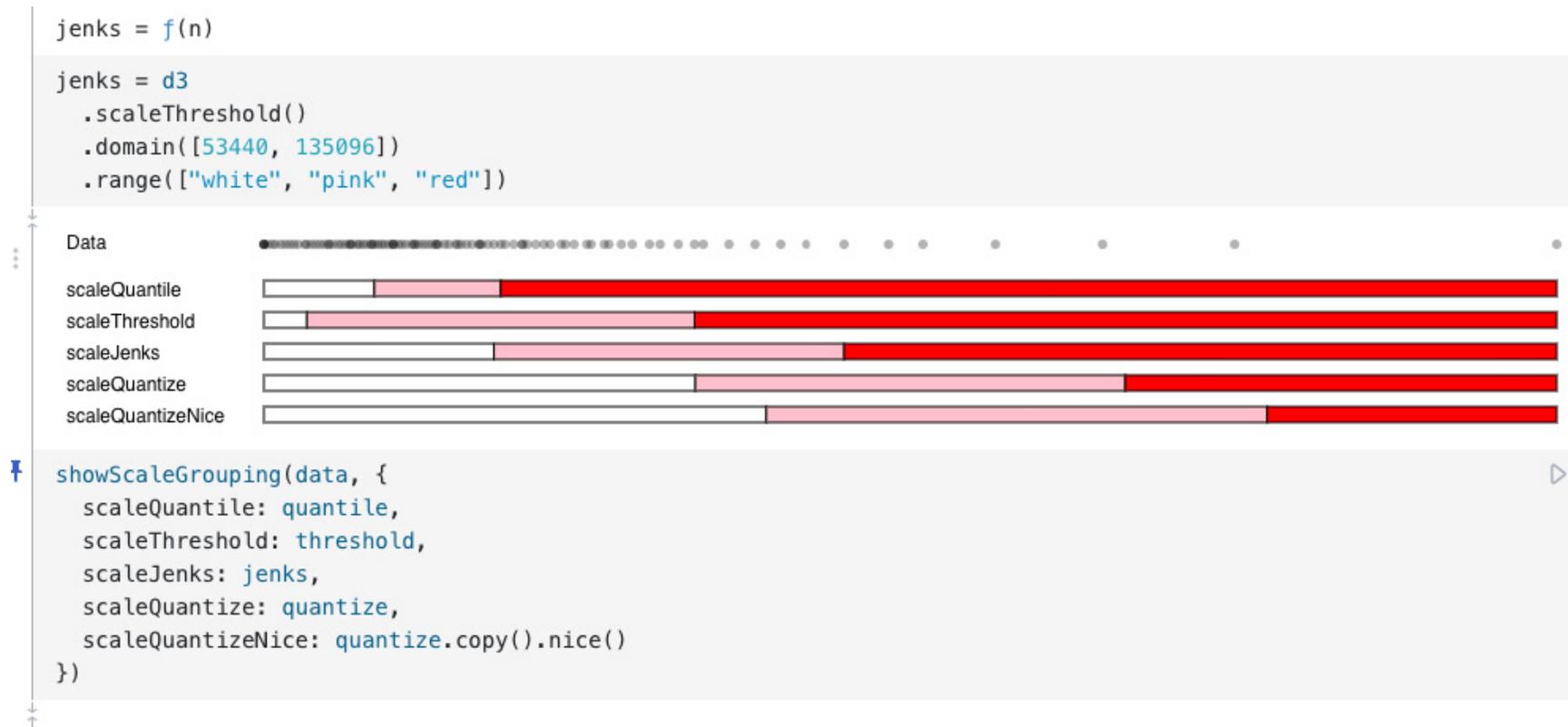
<https://observablehq.com/@d3/quantile-quantize-and-threshold-scales>

```
jenks = f(n)

jenks = d3
  .scaleThreshold()
  .domain([53440, 135096])
  .range(["white", "pink", "red"])

Data
scaleQuantile
scaleThreshold
scaleJenks
scaleQuantize
scaleQuantizeNice

showScaleGrouping(data, {
  scaleQuantile: quantile,
  scaleThreshold: threshold,
  scaleJenks: jenks,
  scaleQuantize: quantize,
  scaleQuantizeNice: quantize.copy().nice()
})
```



ObservableHQ – Sequential Scales

<https://observablehq.com/@d3/sequential-scales>

```

import { rainbow } from 'd3-scale-chromatic';
const angryRainbow = rainbow;

viewof t0 = ramp(angryRainbow)

```



Note. Besides demonstration purposes, this color scheme is not recommended. It is not perceptually uniform (most people will see spikes around the yellow, light blue and pink colors — hence the “angry rainbow” nickname); [d3-scale-chromatic](#) provides better alternatives for [cyclical color scales](#), such as `d3.interpolateSinebow`:

```

"rgb(255, 64, 64)"
d3.interpolateSinebow(t1)

```



```

viewof t1 = ramp(d3.scaleSequential(d3.interpolateSinebow))

```

(All manners of sequential color scales are available in the `d3-scale-chromatic` module: [diverging](#), [single-hue](#), [multi-hue](#) and [cyclical](#).)

`sequential.interpolator` allows to read or modify the interpolator function f in an existing sequential scale. It can be useful to construct a scale iteratively... see [Curran Kelleher's block](#) for an example. We use it below to read a scale's interpolator and create a mirror image (by applying it to $1-t$ instead of t):



```

{

```

ObservableHQ – Diverging Scales

<https://observablehq.com/@d3/diverging-scales>



```
chart(scaleAnomalyPuOr) // chart is defined in the Annex, below
```

While the “PuOr” (purple-orange) interpolator looks good, we’ll prefer in this case a blue (for negative) to red (for positive) color interpolator, passing through white (for neutral). The `interpolator` is a function that takes its inputs in $[0,1]$, and we’re free to create our own.

As D3 offers a standard “RdBu” diverging color interpolator, that goes from red to white to blue. Almost what we needed: we’ll just reverse it to blue-white-red, by applying it to $(1-t)$ instead of t .

The interpolator can be given, in a shorthand notation, as an argument to `d3.scaleDiverging`, so our final code is:

```
scaleAnomaly = f(n)
scaleAnomaly = d3.scaleDiverging(t => d3.interpolateRdBu(1 - t))
  .domain([extent[0], 0, extent[1]])
```



```
chart(scaleAnomaly)
```

To be complete, the shorthand notation also accepts the domain as an optional first argument:



```
chart(d3.scaleDiverging([extent[0], 0, extent[1]], t => d3.interpolateRdBu(1 - t)))
```

Variations are another typical use case to visualize a value change on a map (in that

ObservableHQ – Qualitative Scales

<https://observablehq.com/@d3/d3-scaleordinal>

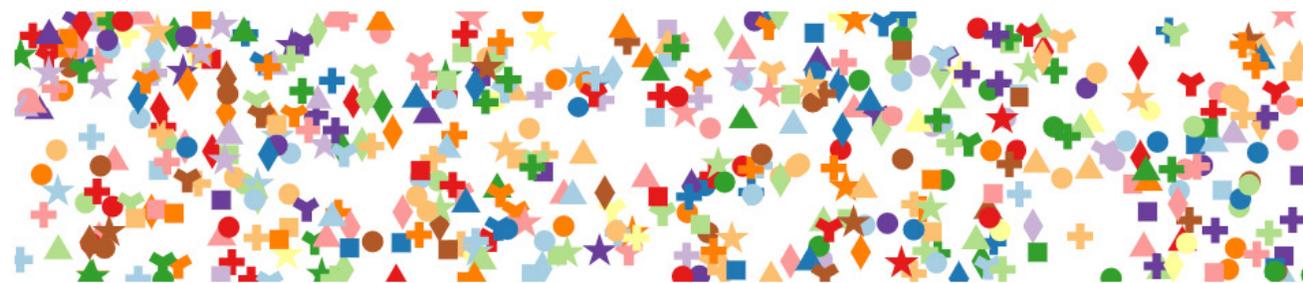
†

Colors & Symbols

Color palettes are a quite common use case for ordinal ranges. You are encouraged to create your own, by hand or using [one of the many tools available](#), but you can also use the list of color schemes provided by [d3-scale-chromatic](#).

A list of twelve words to explore d3 schemePaired color scheme |

A useful range for an ordinal scale can be a set of symbols that will be used to draw shapes, like for instance [d3.symbols](#).



```
{
  const symbols = d3.scaleOrdinal().range(d3.symbols),
    color = d3.scaleOrdinal(d3.schemePaired),
    height = 200,
    symbol = d3.symbol().size(200),
    data = d3.range(500).map(i => ({
      x: width * Math.random(),
      y: height * Math.random(),
      s: Math.floor(9 * Math.random()),
    }));
```