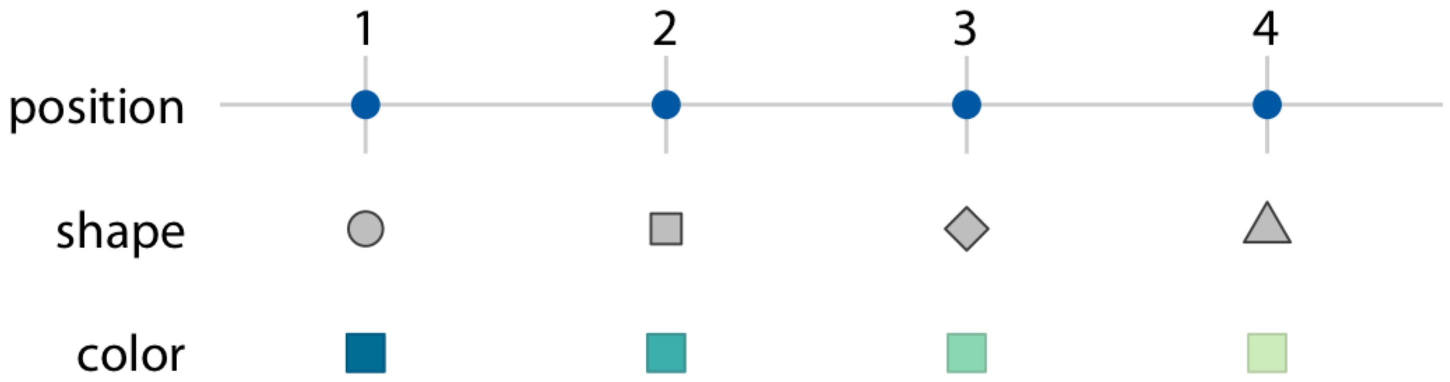
S. Rinzivillo – rinzivillo@isti.cnr.it

DATA VISUALIZATION AND VISUAL ANALYTICS

SCALES FUNCTION

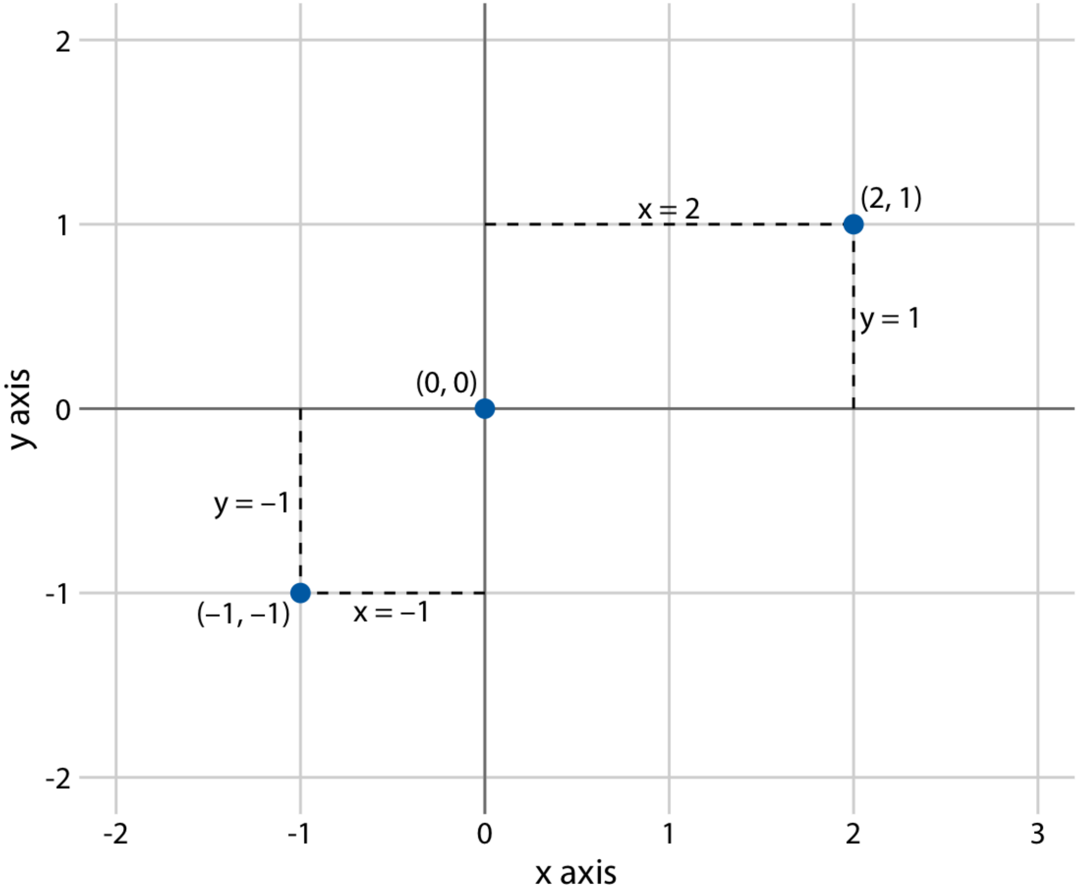
Map data to VV

- 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

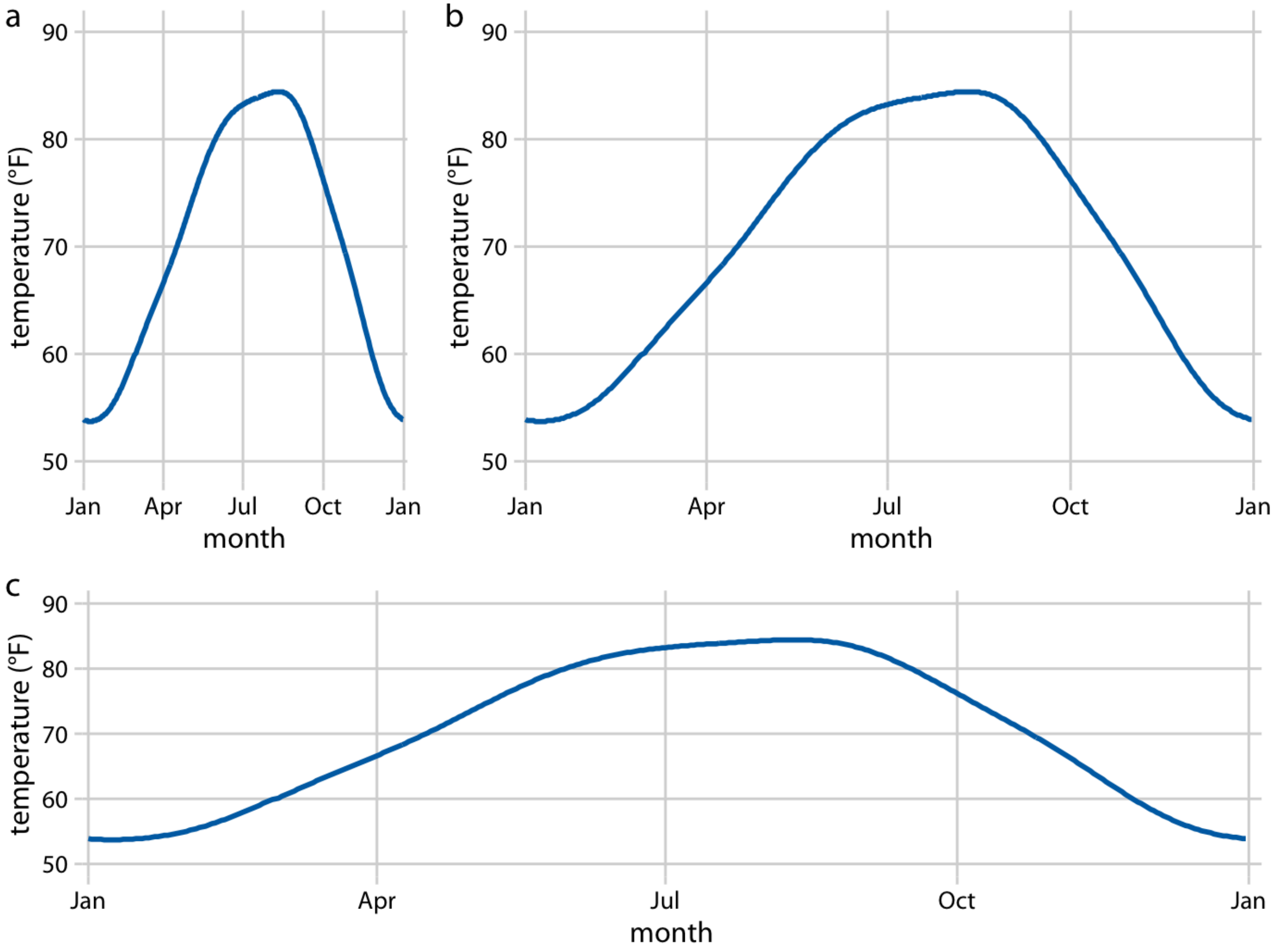


Positional scales: axis

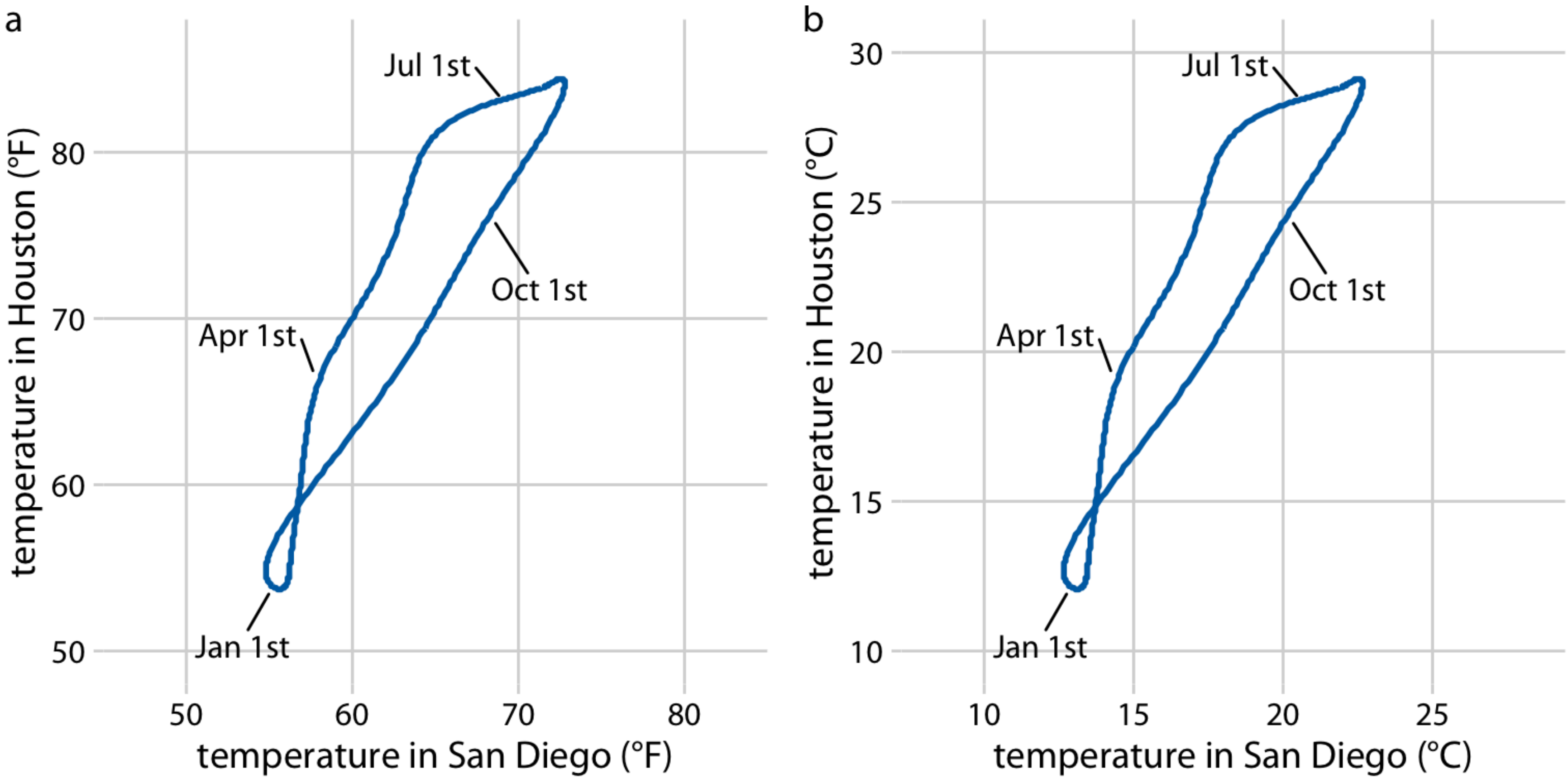
- Axis are at the base of many scientific plots
- Cartesian coordinate systems are composed of two orthogonal axis
- Values are positioned proportionally on the axes



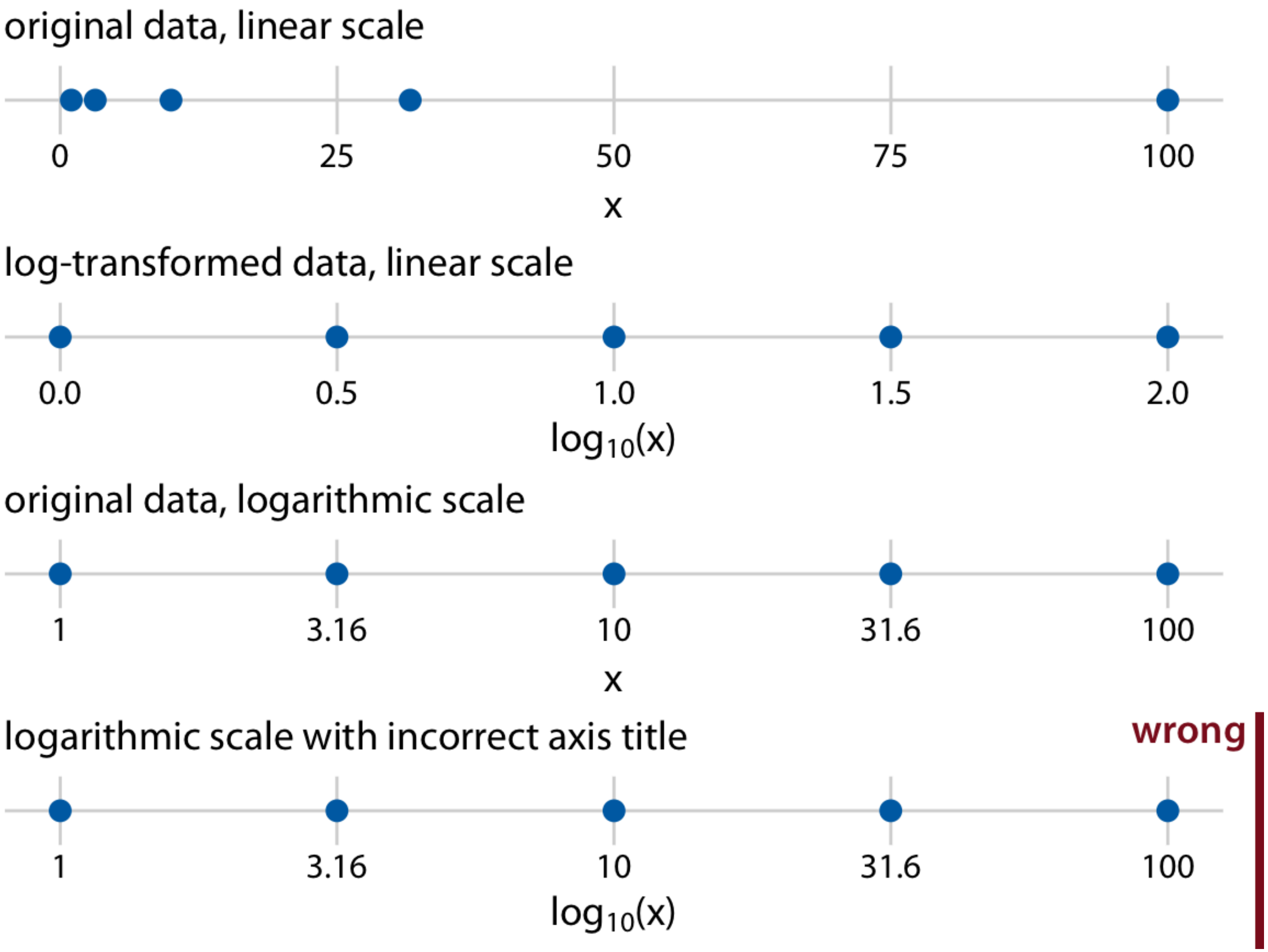
Cartesian diagram with different scales



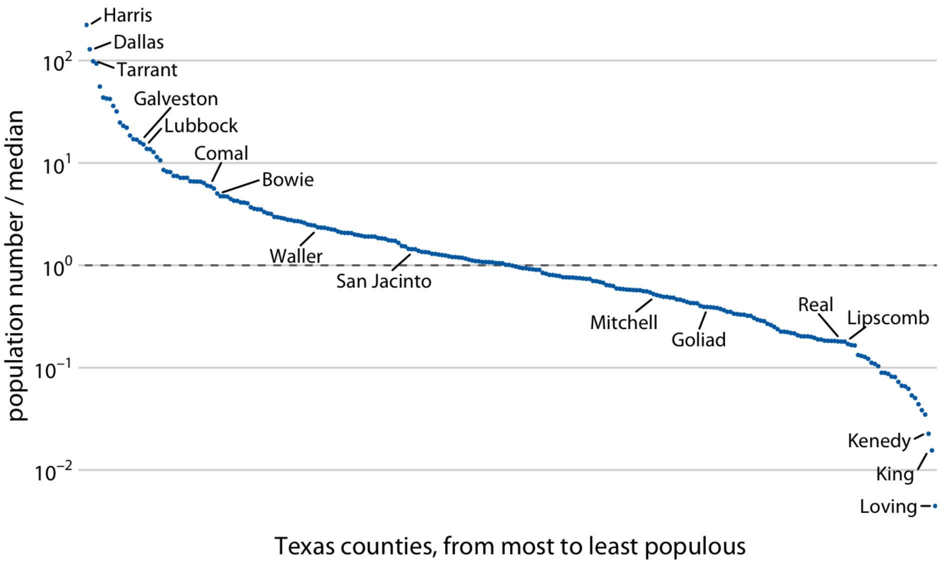
Cartesian axes with same scale

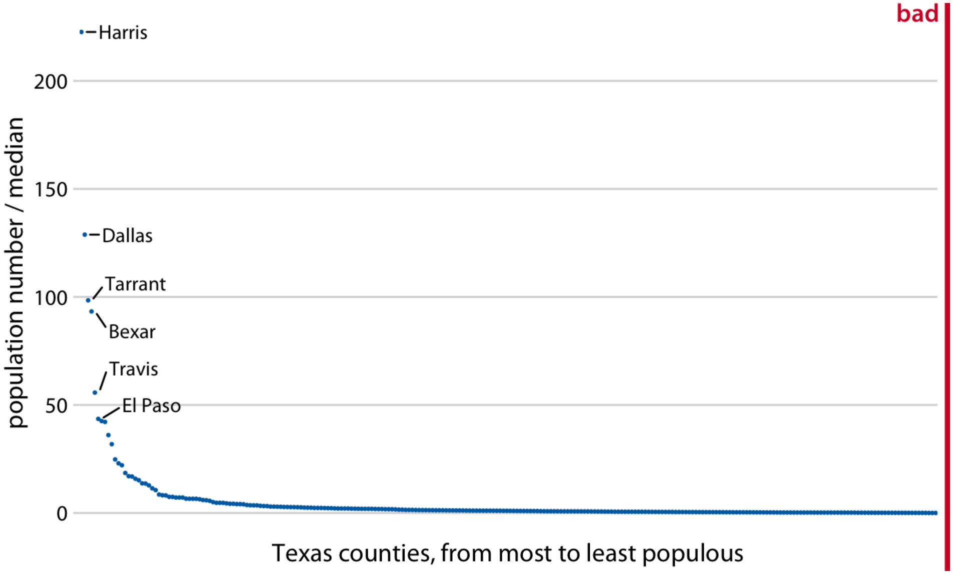


Non linear axes

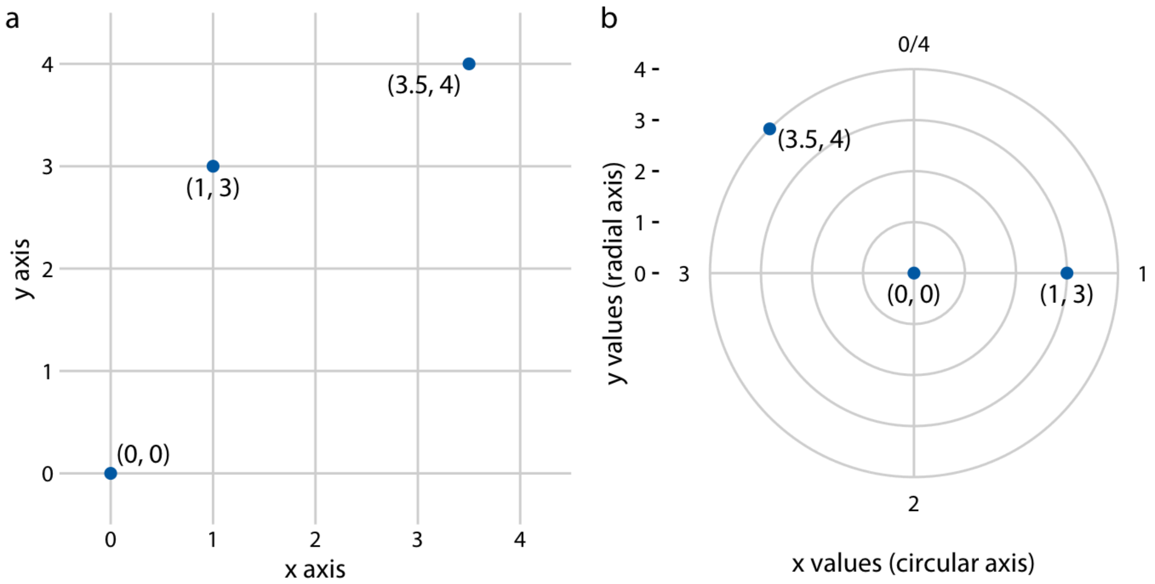


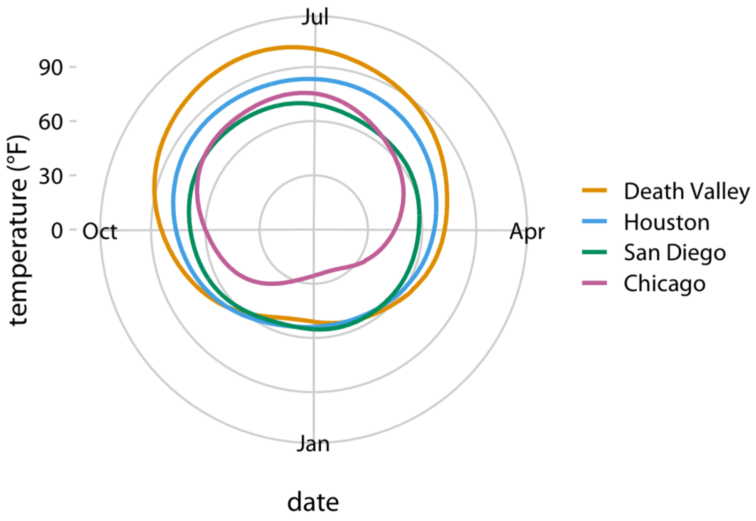
Non linera axes





Curved axes


a
4
3
2
1
0
y axis
0
1
2
3
4
x axis
(3.5, 4)
(1, 3)
(0, 0)
b
0/4
4 -
3 -
2 -
1 -
0 -
y values (radial axis)
3
1
2
(3.5, 4)
(0, 0)
(1, 3)
x values (circular axis)


Jul
90
60
30
0
Oct
Apr
Jan
temperature (°F)
date
Death Valley
Houston
San Diego
Chicago

Example

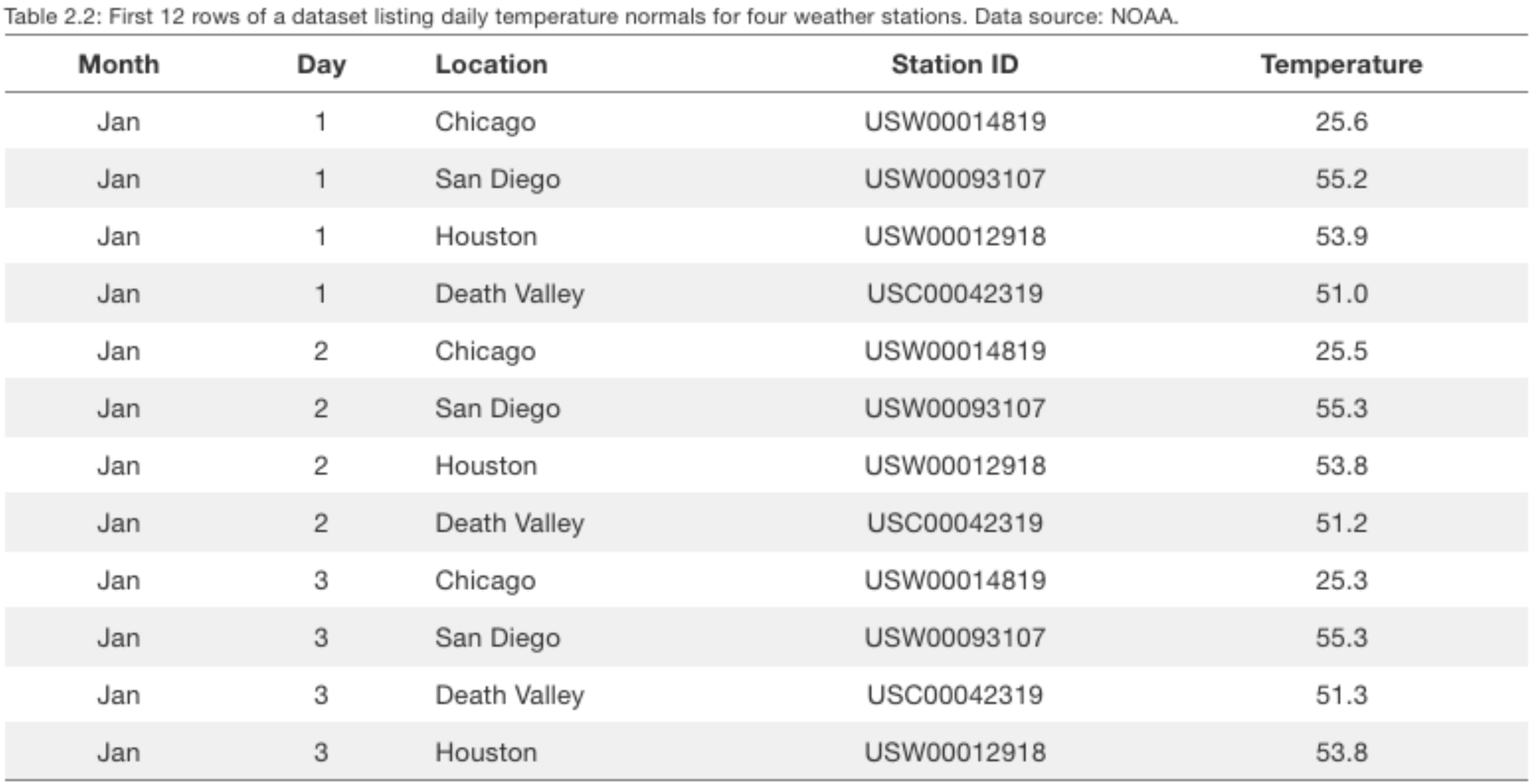


Table 2.2: First 12 rows of a dataset listing daily temperature normals for four weather stations. Data source: NOAA.

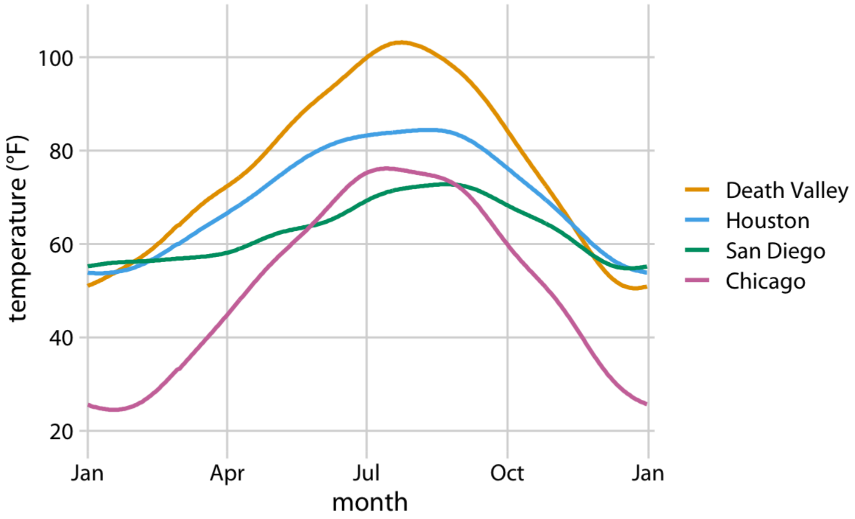
Month	Day	Location	Station ID	Temperature
Jan	1	Chicago	USW00014819	25.6
Jan	1	San Diego	USW00093107	55.2
Jan	1	Houston	USW00012918	53.9
Jan	1	Death Valley	USC00042319	51.0
Jan	2	Chicago	USW00014819	25.5
Jan	2	San Diego	USW00093107	55.3
Jan	2	Houston	USW00012918	53.8
Jan	2	Death Valley	USC00042319	51.2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

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

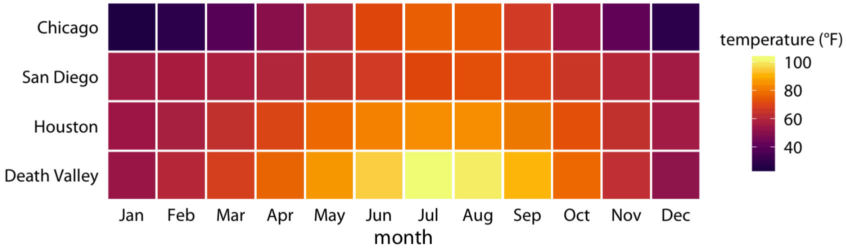
Example

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



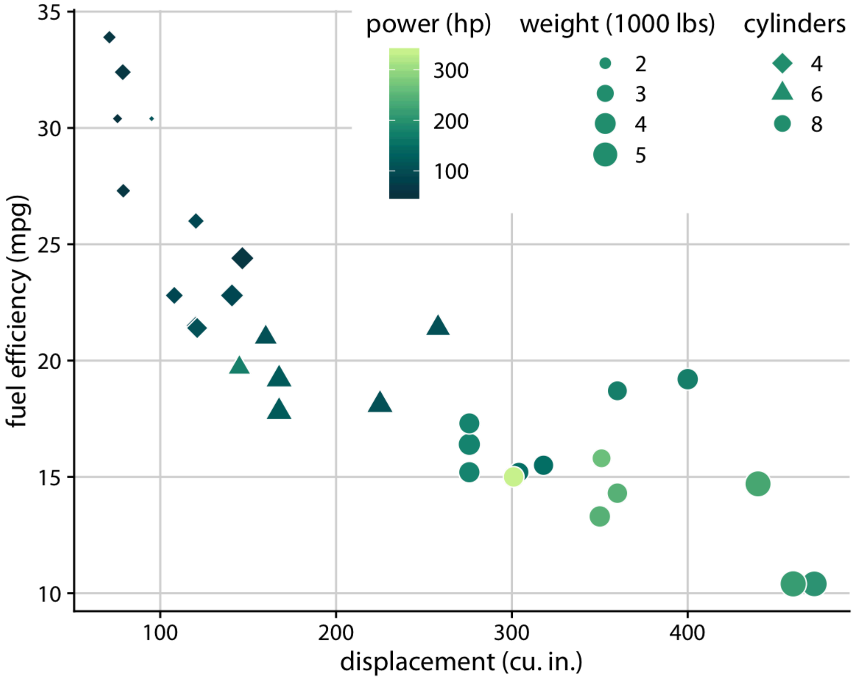
Example

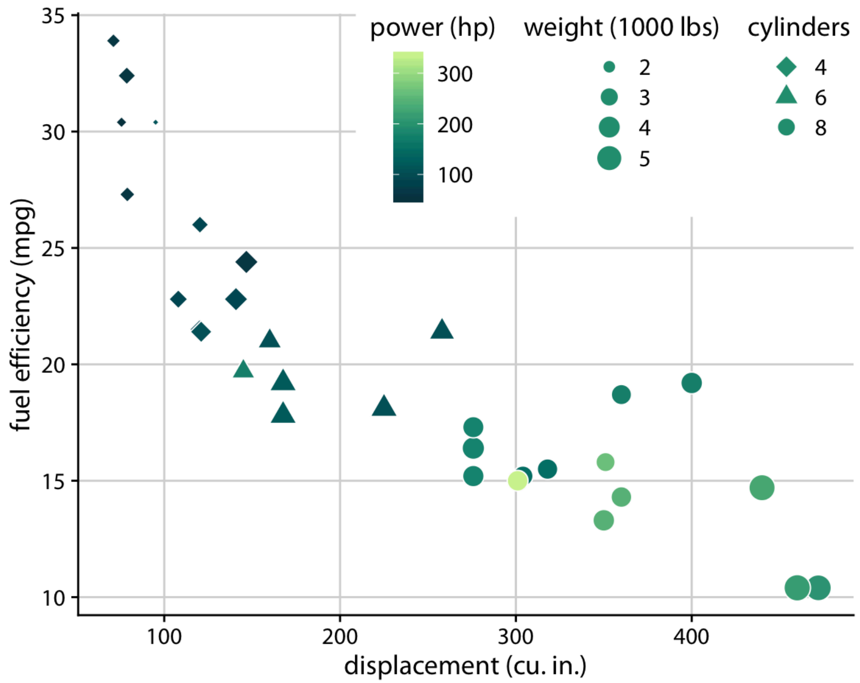
- Month (ordinal) on a ordinal axis (x)
- City (nominal) on a ordinal axis (y) (order determined on sum of temperatures on the line
- Temperature (quantitative) on a color scale

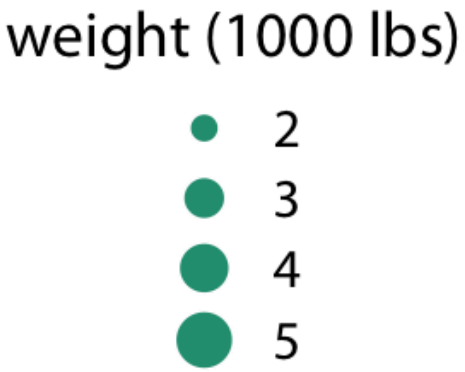


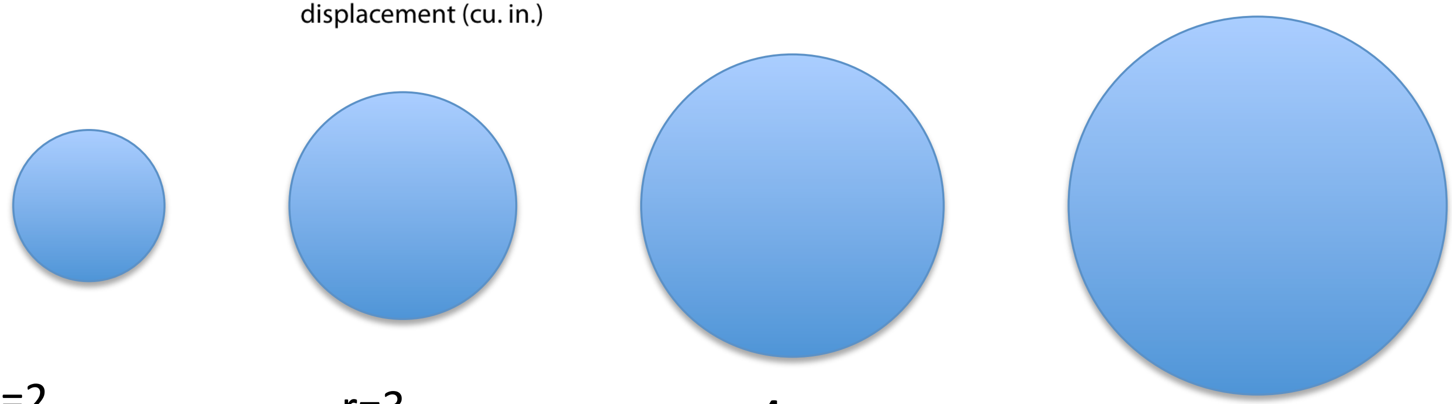
Example

- 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









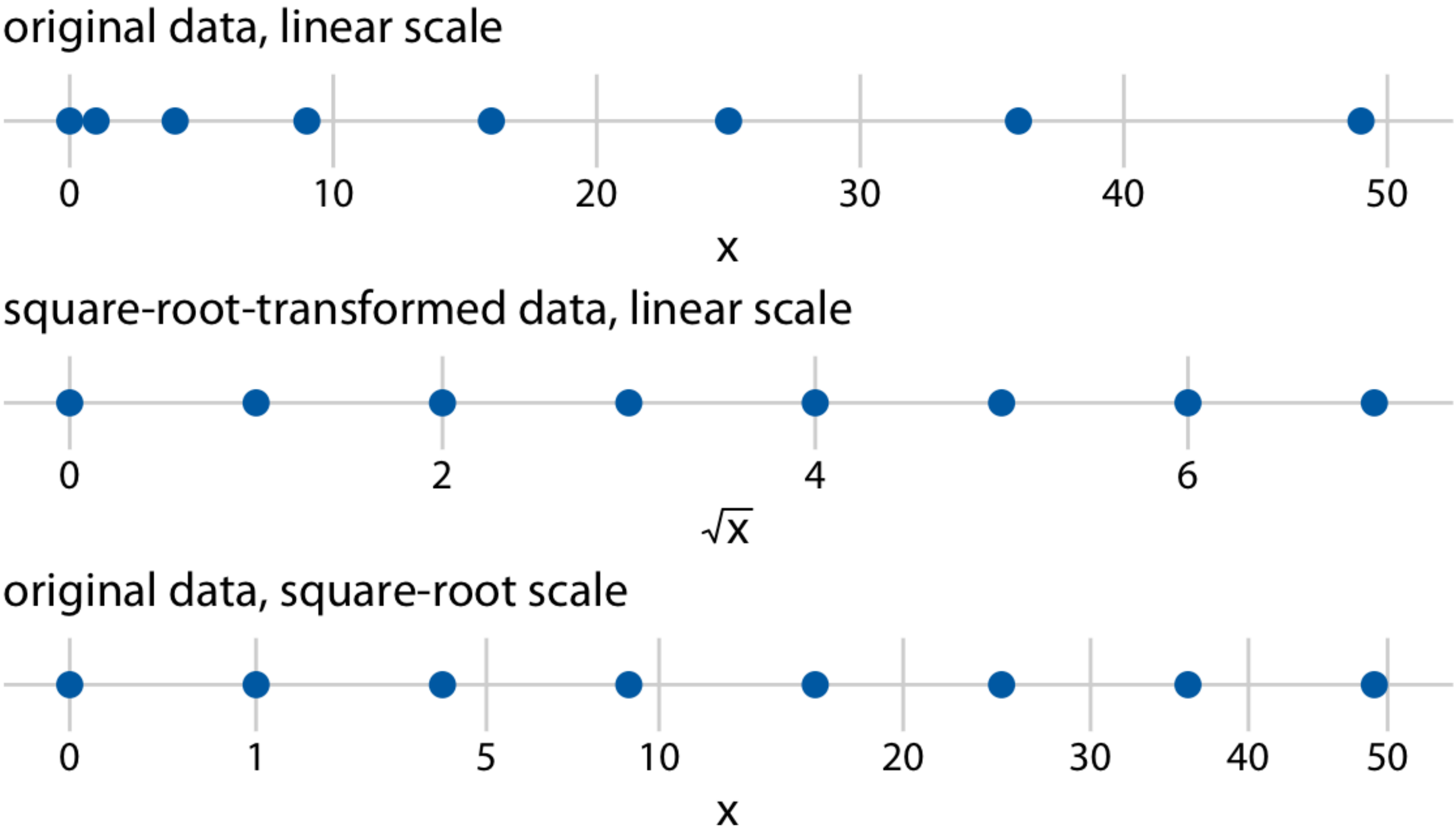
r=2
A = 4*PI

r=3
A = 9*PI

r=4
A = 16*PI

r=5
A = 25*PI

Non linear axes



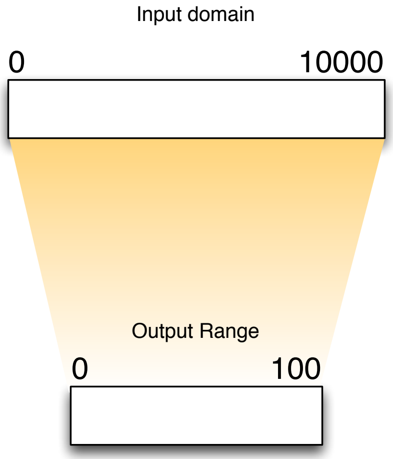
original data, linear scale

square-root-transformed data, linear scale

original data, square-root scale

Manual Mapping

1. For input domain
   1. Select the largest number in original interval (10000)
   2. Select the smallest number in original interval (0)
   3. Select the difference of the two values (10000)
2. For output range
   1. Select the largest number in the new interval (100)
   2. Select the minimum number in the new interval (0)
   3. Select the difference of the two values (100)
3. Compute the ratio of the two intervals’range (10000/100 = 100)



- This is an example of a linear scaling
  - y = mx +b, where b=0 and m=1/100
  - 100 units in the original interval correspond to 1 unit in the destination interval

An example – an alternative solution

```
join.enter()
    .append("rect")
    .attr("x", function(d,i){
        return i*barw;
    })
    .attr("y",function(d){
        return height - d*4;
    })
    .attr("width", barw)
    .attr("height",function(d){
        return d*4;
    });
```

```
join.enter()
    .append("rect")
    .attr("x", function(d,i){
        return x(i);
    })
    .attr("y",function(d){
        return y(d);
    })
    .attr("width", barw)
    .attr("height",function(d){
        return h(d);
    });
function x(d){return m*d + b};
function y(d){return m’*d + b’};
function h(d){return m’’*d + b’’};
```

D3.js Scales generator

- D3 provides several scale types
  - Quantitative
    - Continuous
      - Identity
      - Linear (y=mx+b)
      - Power (y=mx^k+b)
      - Log (y=m log(x) + b)
    - Discrete
      - Quantize
      - Quantile
      - Threshold
  - Ordinal
  - Time

Creating a scale

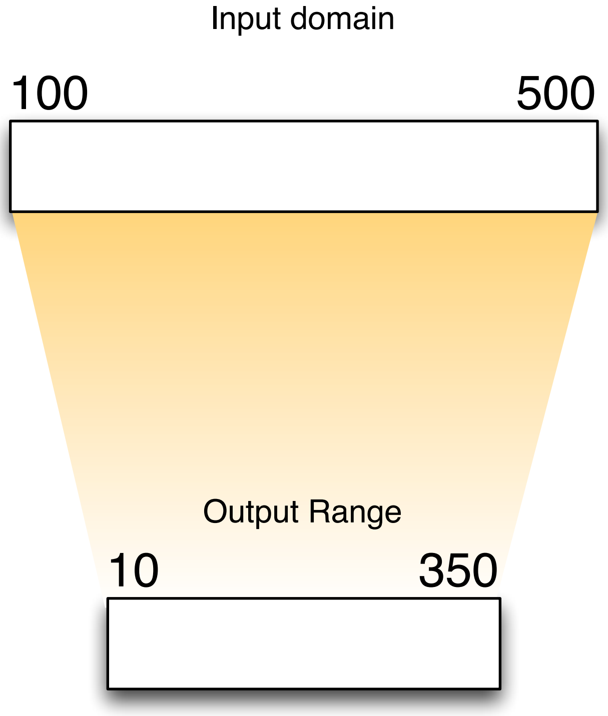
```
var scale = d3.scaleLinear();
```

- Default scale uses
  - Domain is [0,1]
  - Range is [0,1]
  - Function is Identity
  - `scale(2.5); //returns 2.5`

Creating a scale – setting domain and range

```
var scale = d3.scaleLinear()
    .domain([100,500])
    .range([10,350]);
```

- Default scale uses
  - `scale(100); //returns 10`
  - `scale(300); //returns 180`
  - `scale(500); //returns 350`



Quantitative power scale – circle radius

Previous example

```
g.append("circle")
    .attr("fill","pink")
    .attr("stroke","red")
    .attr("r",function(d){
        return Math.sqrt(d*100);
    })
```

Refined solution

```
var r = d3.scaleSqrt()
    .domain([0,20])
    .range([0,30];

g.append("circle")
    .attr("fill","pink")
    .attr("stroke","red")
    .attr("r",function(d){
        return r(d);
    })
```

Domains & Ranges

- Typically, domains are derived from data while ranges are constant.

```
var x = d3.scale.linear()
    .domain([0, d3.max(numbers)])
    .range([0, 720]);
```

```
var x = d3.scale.log()
    .domain(d3.extent(numbers))
    .range([0, 720]);
```

```
function value(d) { return d.value; }

var x = d3.scale.log()
    .domain(d3.extent(objects, value))
    .range([0, 720]);
```

Utility functions: d3.min, d3.max, d3.extent

- To determine the domain and range interval we should know min and max of the two intervals
- D3.js provides utility functions to access such values
  - `d3.min(array[,accessor])`
  - `d3.max(array[,accessor])`
  - `d3.extent(array[,accessor])`

Utility Functions: examples

```
d3.min([10,30,40,70,100]) //returns 10
d3.max([10,30,40,70,100]) //returns 100
d3.extent([10,30,40,70,100]) //returns
[10,100]
```

Interpolators

```
var x = d3.scaleLinear()
    .domain([12, 24])
    .range(["steelblue", "brown"]);

x(16); // #666586
```

```
var x = d3.scaleLinear()
    .domain([12, 24])
    .range(["0px", "720px"]);

x(16); // 240px
```

```
var x = d3.scaleLinear()
    .domain([12, 24])
    .range(["steelblue", "brown"])
    .interpolate(d3.interpolateHsl);

x(16); // #3cb05f
```

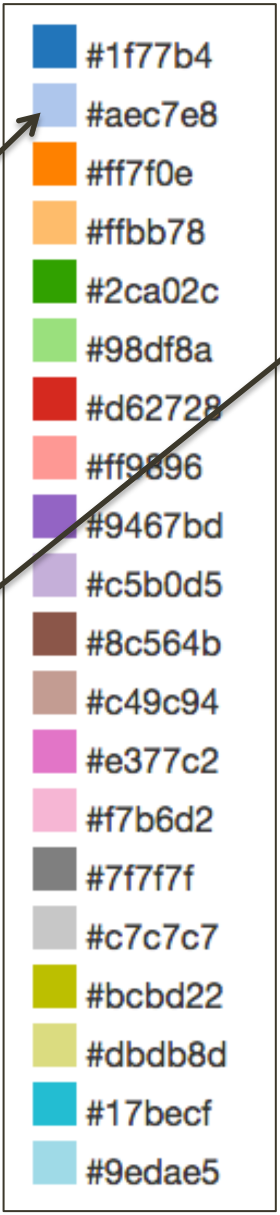
Ordinal Scales

```
var x = d3.scaleOrdinal()
    .domain(["A", "B", "C", "D"])
    .range([0, 10, 20, 30]);

x("B"); // 10
```

```
var x = d3.scaleCategory20()
    .domain(["A", "B", "C", "D"]);

x("B"); // #aec7e8
```

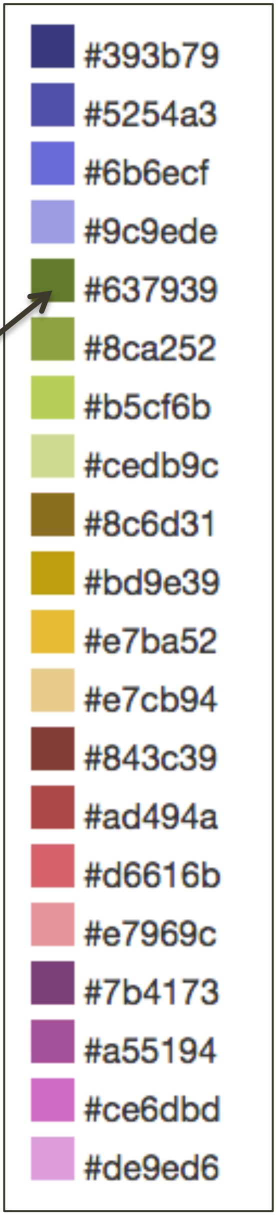
```
var x = d3.scale.category20b()
    .domain(["A", "B", "C", "D"]);

x("E"); // #637939
x.domain(); // A, B, C, D, E
```

category20



#1f77b4
#aec7e8
#ff7f0e
#ffbb78
#2ca02c
#98df8a
#d62728
#ff9896
#9467bd
#c5b0d5
#8c564b
#c49c94
#e377c2
#f7b6d2
#7f7f7f
#c7c7c7
#bcbd22
#dbdb8d
#17becf
#9edae5

category20b



#393b79
#5254a3
#6b6ecf
#9c9ede
#637939
#8ca252
#b5cf6b
#cedb9c
#8c6d31
#bd9e39
#e7ba52
#e7cb94
#843c39
#ad494a
#d6616b
#e7969c
#7b4173
#a55194
#ce6dbd
#de9ed6