



Sahil Chinoy • Oct 11, 2018
gfx



25

Listed in Visualization and NYT stories

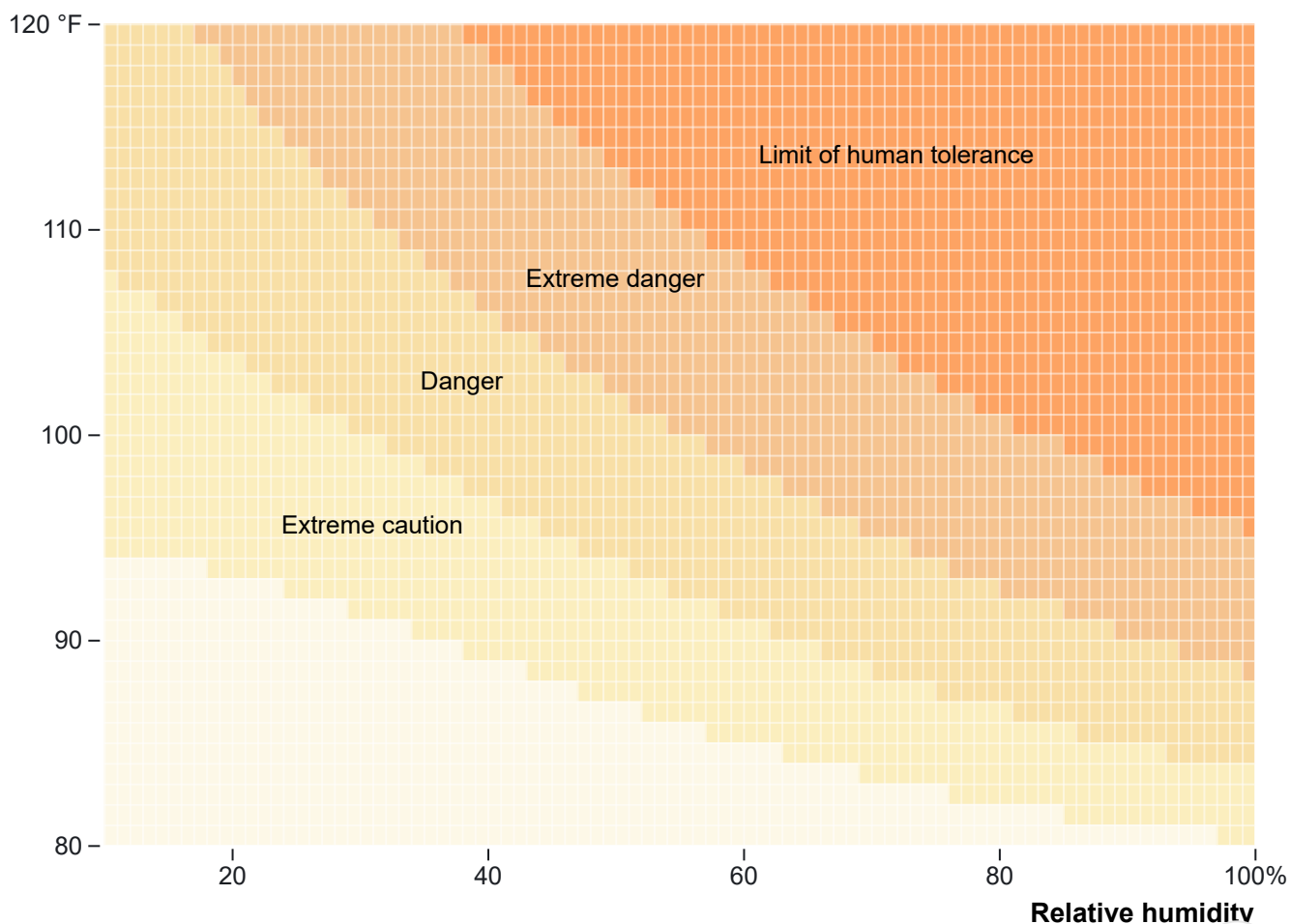
3 forks

Heat index

md`# Heat index`

For a New York Times story about rising heat and humidity creating conditions beyond the limit of human tolerance.

md`For a [New York Times story]
(<https://www.nytimes.com/interactive/2018/10/11/opinion/heat-humidity-killer-combination.html>) about rising heat and humidity creating conditions beyond the limit of human tolerance.`



You have 21 unsaved changes. Fork this notebook to save.

```

{
  const svg = d3.select(DOM.svg(width, height));

  const margin = {
    left: 85,
    top: 30,
    right: 20,
    bottom: 45
  };

  const chartWidth = width - margin.left - margin.right;
  const chartHeight = height - margin.top - margin.bottom;

  const chart = svg.append('g')
    .style('font-family', 'sans-serif')
    .attr('transform', `translate(${margin.left}, ${margin.top})`);

  const x = d3.scaleLinear()
    .domain(d3.extent(humidityTicks))
    .range([0, chartWidth]);

  const y = d3.scaleLinear()
    .domain(d3.extent(temperatureTicks))
    .range([chartHeight, 0]);

  chart.append("g")
    .attr('transform', `translate(0, ${chartHeight + 2})`)
    .style('font-size', '13px')
    .style('font-weight', 300)
    .call(d3.axisBottom(x).ticks(5))
    .call(g => g.select('.tick:last-of-type text').text('100%'))
    .selectAll('.domain').remove();

  chart.append('text')
    .attr('x', chartWidth)
    .attr('y', chartHeight + 40)
    .text('Relative humidity')
    .style('text-anchor', 'end')
    .style('font-weight', 600)
    .style('font-size', '14px');

  chart.append("g")
    .attr('transform', 'translate(-2, 0)')
    .style('font-size', '13px')
    .style('font-weight', 300)

```

```
.call(g => g.select('.tick:last-of-type text').text('120 °F'))
.selectAll('.domain').remove();

chart.append('text')
.attr('x', 0)
.attr('y', -15)
.text('Temperature')
.style('text-anchor', 'end')
.style('font-weight', 600)
.style('font-size', '14px');

const area = d3.area()
.x(d => x(d.rh))
.y0(d => y(d.temp))
.y1(d => y(d.prevTemp))
.curve(d3.curveStepAfter);

const paths = chart.append('g')
.selectAll('path')
.data(chartData)
.enter().append("path")
.attr('d', d => area(d.values))
.style('fill', d => colors[+d.key])
.style('opacity', .8);

chart.append('g')
.attr('transform', `translate(0, ${chartHeight})`)
.call(d3.axisBottom(x)
  .ticks(humidityTicks.length)
  .tickSize(-chartHeight)
  .tickFormat('')
).selectAll('line')
.style('stroke', 'white')
.style('stroke-width', 1)
.style('stroke-opacity', .5);

chart.append('g')
.call(d3.axisLeft(y)
  .ticks(temperatureTicks.length)
  .tickSize(-chartWidth)
  .tickFormat('')
).selectAll('line')
.style('stroke', 'white')
.style('stroke-width', 1)
.style('stroke-opacity', .5);
```

```

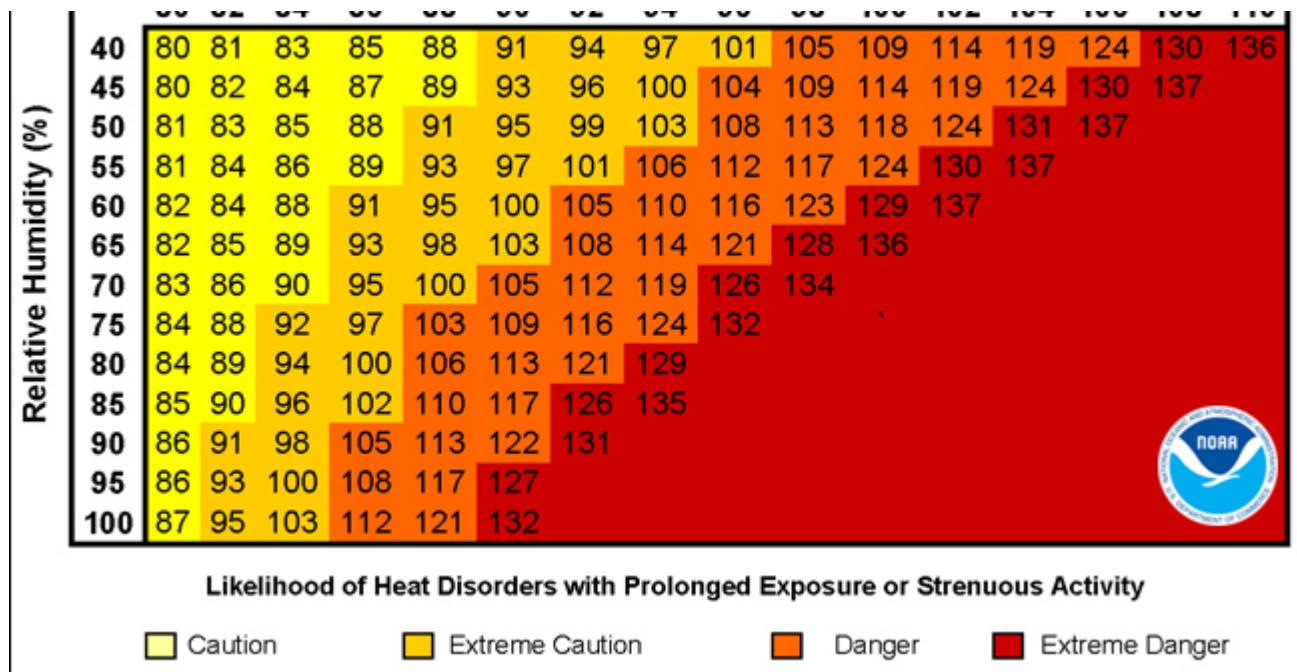
chart.append('g')
.selectAll('text')
.data(thresholds)
.enter().append('text')
.text(d => d.text)
.attr('transform', d => `translate(${x(d.H)}, ${y(d.T) - 2})`)
.attr('x', 0)
.attr('y', 0)
.style('text-anchor', 'middle')
.style('font-size', '13px')
.style('font-weight', 300);

return svg.node();
}

```

Weather Service.

We wanted to recreate this classic, gridded heat index chart from the National Weather Service.



The heat index, which measures how hot it actually feels when considering both

```

html`

```

md The heat index, which measures how hot it actually feels when considering both temperature and humidity, is defined by a pretty gnarly [nonlinear function] (https://www.wpc.ncep.noaa.gov/html/heatindex_equation.shtml):`

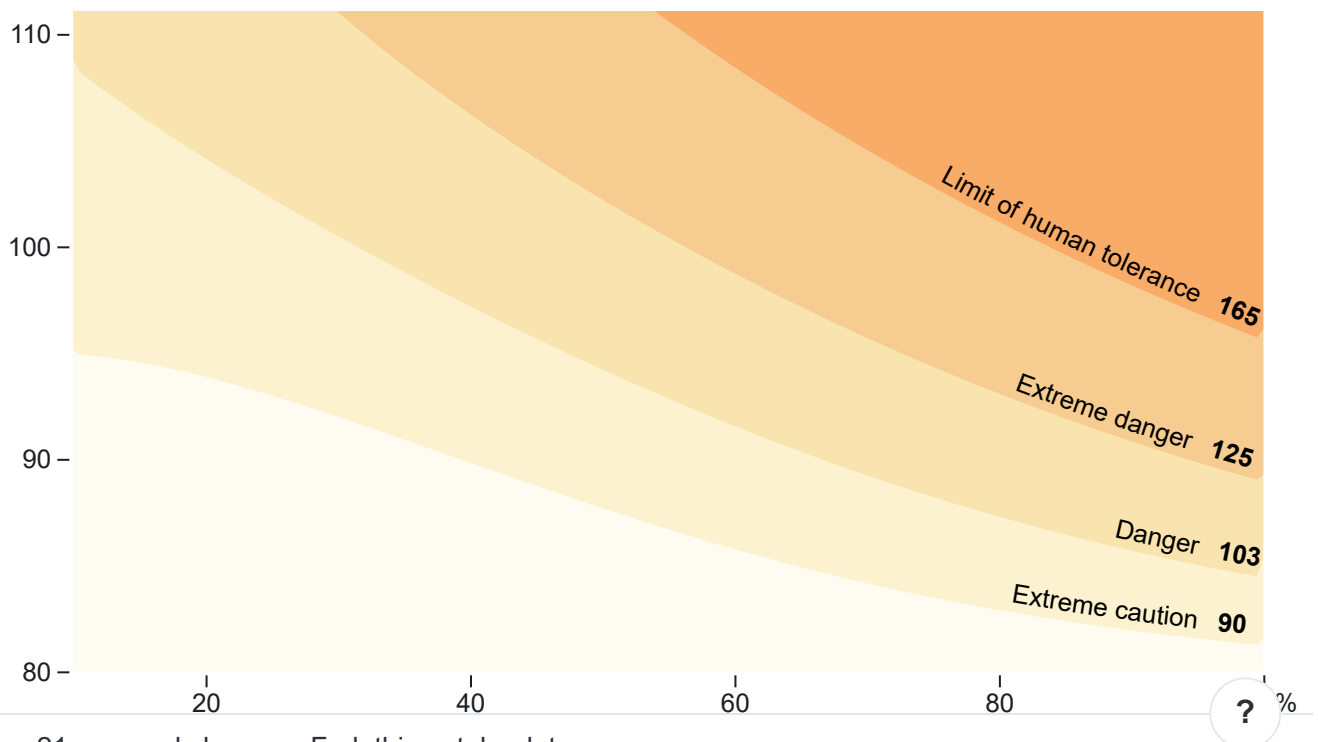
```
tex`HI = -42.379 + 2.04901523 \times T + 10.14333127 \times RH - .22475541 \times T \times RH - .00683783 \times T^2 - .05481717 \times RH^2 + .00122874 \times T^2 \times RH + .00085282 \times T \times RH^2 - .00000199 \times T^2 \times RH^2`
```

individual cells. (This approach, h/t Adam Pearce, is mostly for performance reasons — in the story, we wanted to show and hide a portion of the chart.)

I also played with drawing contours instead of individual cells, but we decided the result looked *too* clean.

md`Here, we calculate the heat index at every combination of temperature and humidity. But instead of drawing each cell as a `\`rect\``, we do some tricks with `\`d3.nest\`` and then use `\`d3.area\`` to plot paths, superimposing white gridlines to give the appearance of individual cells. (This approach, h/t Adam Pearce, is mostly for performance reasons — in the story, we wanted to show and hide a portion of the chart.)

I also played with drawing contours instead of individual cells, but we decided the result looked **too** clean.`



You have 21 unsaved changes. Fork this notebook to save.

```
{
  const svg = d3.select(DOM.svg(width, height))

  const margin = {
    left: 85,
    top: 30,
    right: 20,
    bottom: 45
  };

  const chartWidth = width - margin.left - margin.right;
  const chartHeight = height - margin.top - margin.bottom;

  const chart = svg.append('g')
    .style('font-family', 'sans-serif')
    .attr('transform', `translate(${margin.left}, ${margin.top})`);

  const x = d3.scaleLinear()
    .domain(d3.extent(humidityTicks))
    .range([0, chartWidth]);

  const y = d3.scaleLinear()
    .domain(d3.extent(temperatureTicks))
    .range([chartHeight, 0]);

  chart.append('g')
    .attr('transform', `translate(0, ${chartHeight + 2})`)
    .style('font-size', '13px')
    .style('font-weight', 300)
    .call(d3.axisBottom(x).ticks(5))
    .call(g => g.select('.tick:last-of-type text').text('100%'))
    .selectAll('.domain').remove();

  chart.append('text')
    .attr('x', chartWidth)
    .attr('y', chartHeight + 40)
    .text('Relative humidity')
    .style('text-anchor', 'end')
```

```
chart.append("g")
.attr('transform', 'translate(-2, 0)')
.style('font-size', '13px')
.style('font-weight', 300)
.call(d3.axisLeft(y).ticks(5))
.call(g => g.select('.tick:last-of-type text').text('120 °F'))
.selectAll('.domain').remove();

chart.append('text')
.attr('x', 0)
.attr('y', -15)
.text('Temperature')
.style('text-anchor', 'end')
.style('font-weight', 600)
.style('font-size', '14px');

//const values = cartesianProduct(temperatureTicks, humidityTicks).map(d => HI(...d));

const contours = d3.contours()
.size([humidityTicks.length, temperatureTicks.length])
.thresholds(thresholds.map(d => d.value))(values.map(d => d.HI));

// background color
chart.append('rect')
.attr('x', 0)
.attr('y', 0)
.attr('width', chartWidth)
.attr('height', chartHeight)
.style('fill', colors[0])
.style('opacity', .5);

const projection = d3.geoTransform({
  point: function(x, y) {
    const newX = x * chartWidth / humidityTicks.length;
    const newY = y * chartHeight / temperatureTicks.length;
    this.stream.point(newX, newY);
  }
});

chart.append('g')
.selectAll('path')
.data(contours)
.enter().append('path')
.attr('id', d => 'g-' + d.value)
```

```

.style('fill', (d, i) => colors[i + 1])
.style('opacity', .5);

chart.append('g')
.selectAll('text')
.data(thresholds)
.enter().append('text')
.attr('transform', 'translate(0, -5)')
.append('textPath')
.attr('xlink:href', d => `#g-${d.value}`)
.attr('startOffset', d => '99.5%')
.style('text-anchor', 'end')
.style('font-size', '13px')
.append('tspan')
.style('font-weight', 300)
.text(d => d.text)
.append('tspan')
.attr('dx', 10)
.text(d => d.value)
.style('font-weight', 600)

return svg.node();
}

```

```

temperatureTicks = ▶ Array(41) [120, 119, 118, 117, 116, 115, 114, 113, 112, 111, 110, 109,

```

```

humidityTicks = d3.range(10, 101, 1)

```

```

temperatureTicks = d3.range(80, 121, 1).reverse()

```

```

values = cartesianProduct(temperatureTicks, humidityTicks).map(d => {
  return {
    temp: d[0],
    rh: d[1],
    HI: HI(...d),
    threshold: thresholdScale(HI(...d))
  }
})

```



```

chartData = d3.nest()
  .key(d => d.threshold)
  .sortKeys(d3.ascending)
  .key(d => d.rh)
  .rollup(vals => d3.max(vals))
  .entries(values)
  .map(d => {
    return {
      key: d.key,
      values: humidityTicks.map(h => {
        const v = d.values.find(f => +f.key == h);
        return {
          rh: h,
          temp: v ? v.value.temp : 120
        }
      })
    }
  })
  .map((d, i, arr) => {
    d.values = d.values.map((f, j) => {
      f.prevTemp = i > 0 ? arr[i - 1].values[j].temp : 80;
      return f;
    })
    return d;
  })

```

```

thresholds = [
  {
    'value': 90,
    'text': 'Extreme caution',
    'T': 95,
    'H': 31
  },
  {
    'value': 103,
    'text': 'Danger',
    'T': 102,
    'H': 38
  }
]

```

```

    'value': 125,
    'text': 'Extreme danger',
    'T': 107,
    'H': 50
  },
  {
    'value': 165,
    'text': 'Limit of human tolerance',
    'T': 113,
    'H': 72
  }
]

+ colors = ▶ Array(8) ["#fdf7e1", "#faeaae", "#f7d790", "#f3b473", "#fd8d3c", "#fc4e2a", "#e31a1c", "#b10026"]

thresholdScale = d3.scaleThreshold()
  .domain(thresholds.map(d => d.value))
  .range([0,1,2,3,4])

+

colors =
  ['#fdf7e1', '#faeaae', '#f7d790', '#f3b473', '#fd8d3c', '#fc4e2a', '#e31a1c', '#b10026']

+

cartesianProduct = f(...)

// compute the heat index given the temperature, T, and relative humidity, RH
HI = {
  return (T, RH) => -42.379 + 2.04901523*T + 10.14333127*RH - .22475541*T*RH -
    .00683783*T*T - .05481717*RH*RH + .00122874*T*T*RH + .00085282*T*RH*RH -
    .00000199*T*T*RH*RH;
}

+

width = 700

cartesianProduct = {
  // product-of-multiple-arrays-in-javascript
  const f = (a, b) => [].concat(...a.map(d => b.map(e => [].concat(d, e))));
  const cartesian = (a, b, ...c) => (b ? cartesian(f(a, b), ...c) : a);
  return cartesian;
}

+ d3 = ▶ Object {event: null, format: f(t), formatPrefix: f(t, n), timeFormat: f(t), timePar:

width = 700

```

```
⋮  
|  
height = 500  
+  
⋮  
|  
d3 = require("d3@5")  
+
```

Continue reading NYT stories

NEXT

The Cube Root Law

© 2019 Observable, Inc.

[Terms](#) [Open Source](#)