Vega.js

Visualização 2017.2

Leonardo Alves (las3), Marcos Barreto (msb5) e Thiago Bastos (tmb2)

O que é?

Vega é uma gramática de visualização, uma linguagem declarativa para criar, salvar e compartilhar projetos de visualização interativa.

O que é?

Com Vega, as visualizações são descritas em JSON e geradas visualizações interativas usando HTML5 Canvas e SVG.

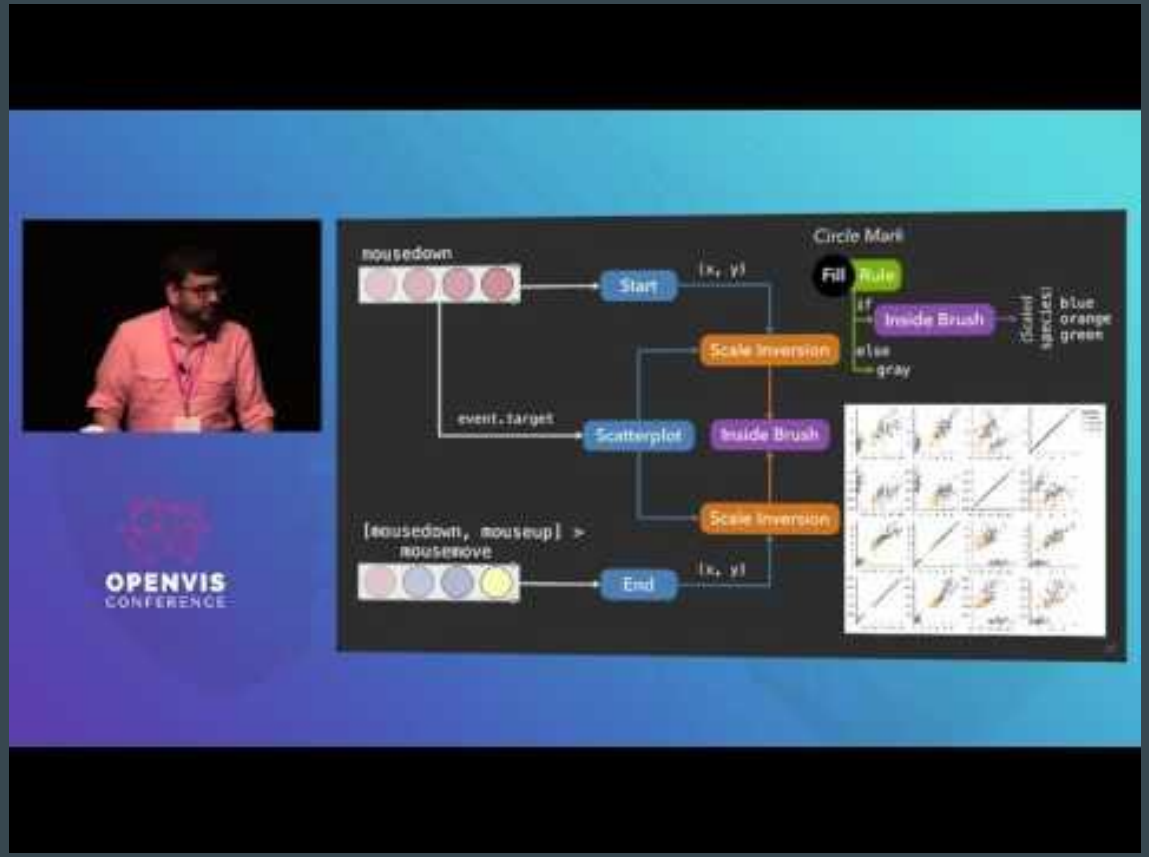
JSON

- JavaScript Object Notation.
- Formatação leve de troca de dados.
- Baseado em Strings.
- Chaves e valores.
- Fácil entendimento humano e interpretação da máquina.

Vega e D3

- Vega é construído como uma camada de abstração do D3.
- D3 ainda continua sendo melhor para visualizações complexas, mas Vega facilita o desenvolvimento de visualizações comuns.

Reactive Building Blocks Interactive Visualizations with Vega



Principais propriedades

```
{
    "$schema": "https://vega.github.io/schema/vega/v3.0.json",
    "description": "A specification outline example.",
    "width": 500,
    "height": 200,
    "padding": 5,
    "autosize": "pad",

    "signals": [],
    "data": [],
    "scales": [],
    "projections": [],
    "axes": [],
    "marks": []
}
```

Autosize

- O tamanho total de uma visualização Vega pode ser determinado por múltiplos fatores: ***width***, ***height*** e ***padding***, bem como conteúdo como ***axes***, ***legends*** e ***titles***.
- A Vega fornece três tipos de *autosize* diferentes:
  - `"none"`: Nenhum dimensionamento automático é realizado. O tamanho da visualização total é determinado unicamente por: width, height e padding. Qualquer conteúdo que se encontre fora desta região será cortado.
  - `"pad"`: Esta é a configuração padrão. Aumenta automaticamente o tamanho da *view* de modo que todo o conteúdo da visualização seja visível.
  - `"fit"`: Ajuste automaticamente o layout em uma tentativa de forçar o tamanho de visualização total para caber dentro da largura, altura e valores de preenchimento dados.

Signals

- São variáveis dinâmicas que parametrizam uma visualização e podem gerar comportamentos interativos.
- Podem ser usados em toda a especificação do Vega, por exemplo, para definir uma propriedade de Mark ou um parâmetro em Transform nos dados.
- Os valores do sinal são reativos: eles podem atualizar em resposta aos fluxos de eventos de entrada, como ***mousedown*** ou ***touchmove***.

```
[{
    "name": "tooltip",
    "value": {},
    "on": [
        {"events": "rect:mouseover", "update": "datum"},
        {"events": "rect:mouseout",  "update": "{}"}
    ]
}]
```

Data

- Definições e transformações de conjuntos de dados definem os dados a serem carregados e como processá-lo.
- O modelo básico de dados utilizado pela Vega é um dado tabular, semelhante a uma planilha Excel ou uma base de dados.
- A fonte dos dados deve ser no máximo uma: ***source***, ***values*** e ***url***.

```
"data": [{
    "name": "table",
    "values": [12, 23, 47, 6, 52, 19]
 },
 {
    "name": "points",
    "url": "data/points.js"
 },
 {
    "name": "stats",
    "source": "table"
 }]
```

Transforms

- Processar os dados para filtrar ou calcular novos campos.
- Especificado dentro do array de **transform** na definição dos dados.
- Tem vários tipos, como por exemplo: **aggregate**, **filter** e **geopoint**.

```
"data": [
    {
        "name": "table",
        "transform": [
            { "type": "filter", "expr": "datum.value > 5"}
        ]
    }
]
```

Scale

- Scale mapeia valores em Data (números, datas, categorias, etc.) para valores visuais (pixels, cores e tamanhos) semelhante ao D3.
- Vega inclui uma gama de escalas para dados de entrada contínua e discreta e suporta mapeamentos para a posição, a forma, o tamanho e as codificações de cores.
- Scale varia em três tipos: ***quantitativo***, ***discreto*** e ***discretizado***.

```
"scales":[
    {
        "name": "color",
        "type": "quantize",
        "domain": [0, 0.15],
        "range": {"scheme": "blues-9"}
    }
]
```

Projections

- Mapeia pares (latitude, longitude) em pares (x, y).
- Usa-se ***projections*** tanto para pontos (como localizações em um mapa) quanto para regiões (como países e estados), a partir de dados retirados de GeoJSON's.
- É possível utilizar vários tipos de projeções, como, por exemplo: ***albers***, ***albersUSA*** e ***mercator***.

```
"projections": [
    {
        "name": "proj",
        "type": "mercator"
    }
 ]
```

Axes

- Axes visualizam mapeamentos de escala espacial usando ***ticks***, ***lines*** e ***labels***.
- Vega atualmente suporta axes para coordenadas cartesianas (retangulares).
- Também semelhante ao D3.

```
"axes":[
    {
        "orient": "bottom",
        "scale": "xscale"
    },
    {
        "orient": "left",
        "scale": "yscale"
    }
]
```

Marks

- *Marks* codificam de forma visual os dados usando formas geométricas como ***arc***, ***path*** e ***rect***. As definições de propriedade da marca podem ser constantes simples ou campos de dados, ou escalas podem ser usadas para mapear valores de dados para valores visuais.

```
{
    "type": "rect",
    "from": {"data": "table"},
    "encode": {
        "enter": {
            "x": {"scale": "xscale", "field": "value"},
            "y": {"scale": "yscale", "value": 0},
        }
    }
}
```

Marks - Encode

- Cada *mark* suporta um conjunto de propriedades de codificação visual (encode) que determinam sua posição e a aparência.
- Semelhante ao D3, existem três conjuntos de propriedades principais: ***enter***, ***update*** e ***exit***.
- As propriedades de *mark* são especificadas como pares nome-valor.
- O nome é simplesmente o nome da propriedade visual e o valor deve ser:
  - Value Reference: uma constante, atributo de um objeto ou o valor de um *Scale*.

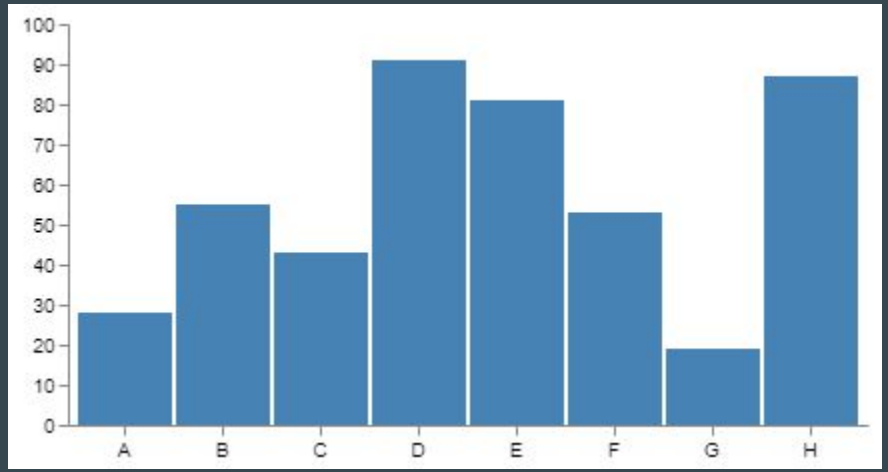
```
{"value": "left"}, {"field": "count"}, {"scale": "x", "field": "count"}
```

  - Production Rules: podem ser definidas através da avaliação de um ***if-then-else***.

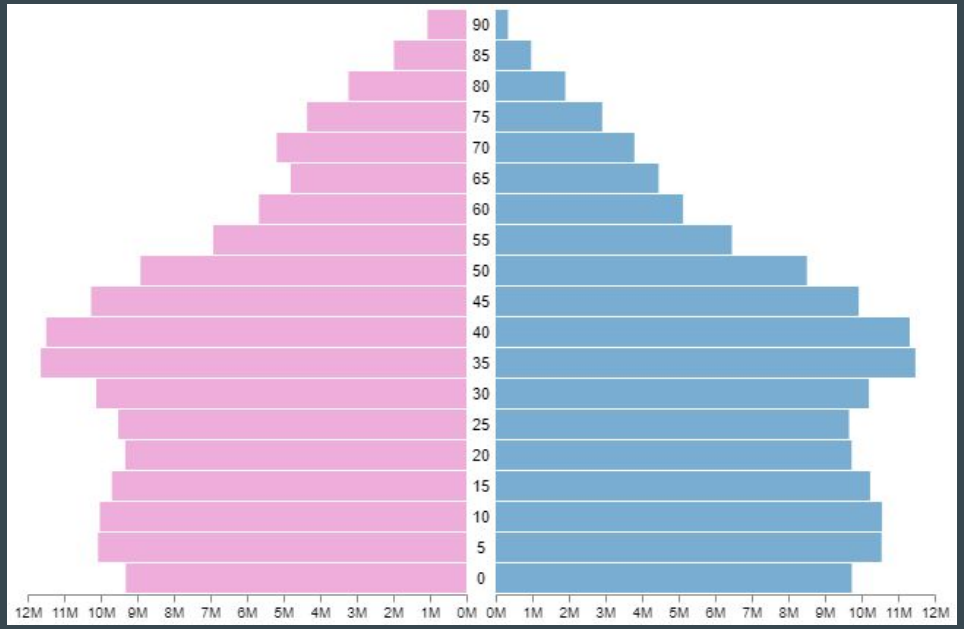
```
[{"test": "datum.count > 5", "value": "blue"}, {"value": "red"}]
```

Quais visualizações possíveis com Vega.js?

Bar Charts

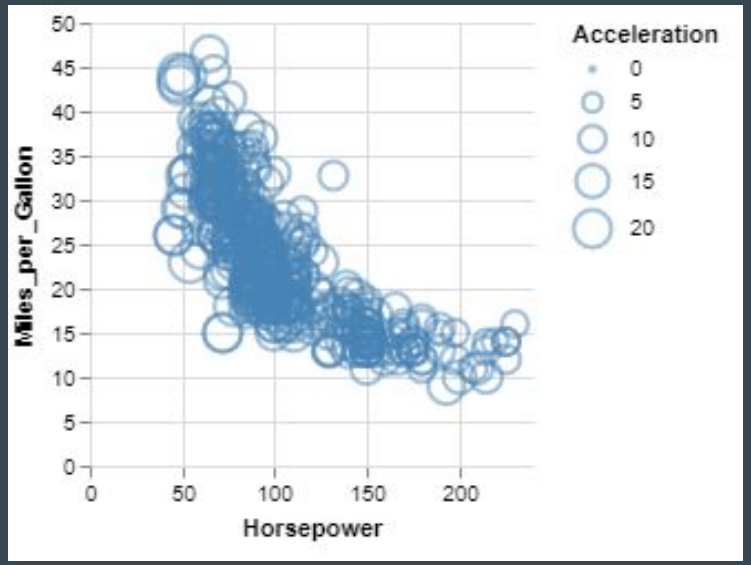


Bar Chart

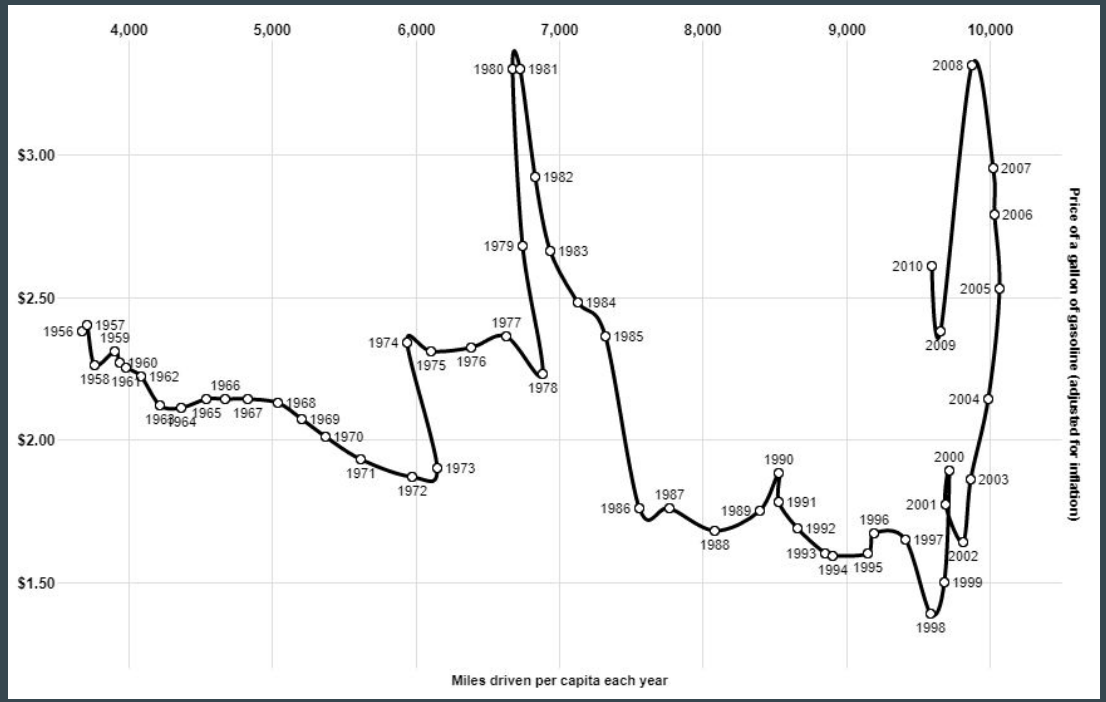


Population Pyramid

Dot & Scatter Plots

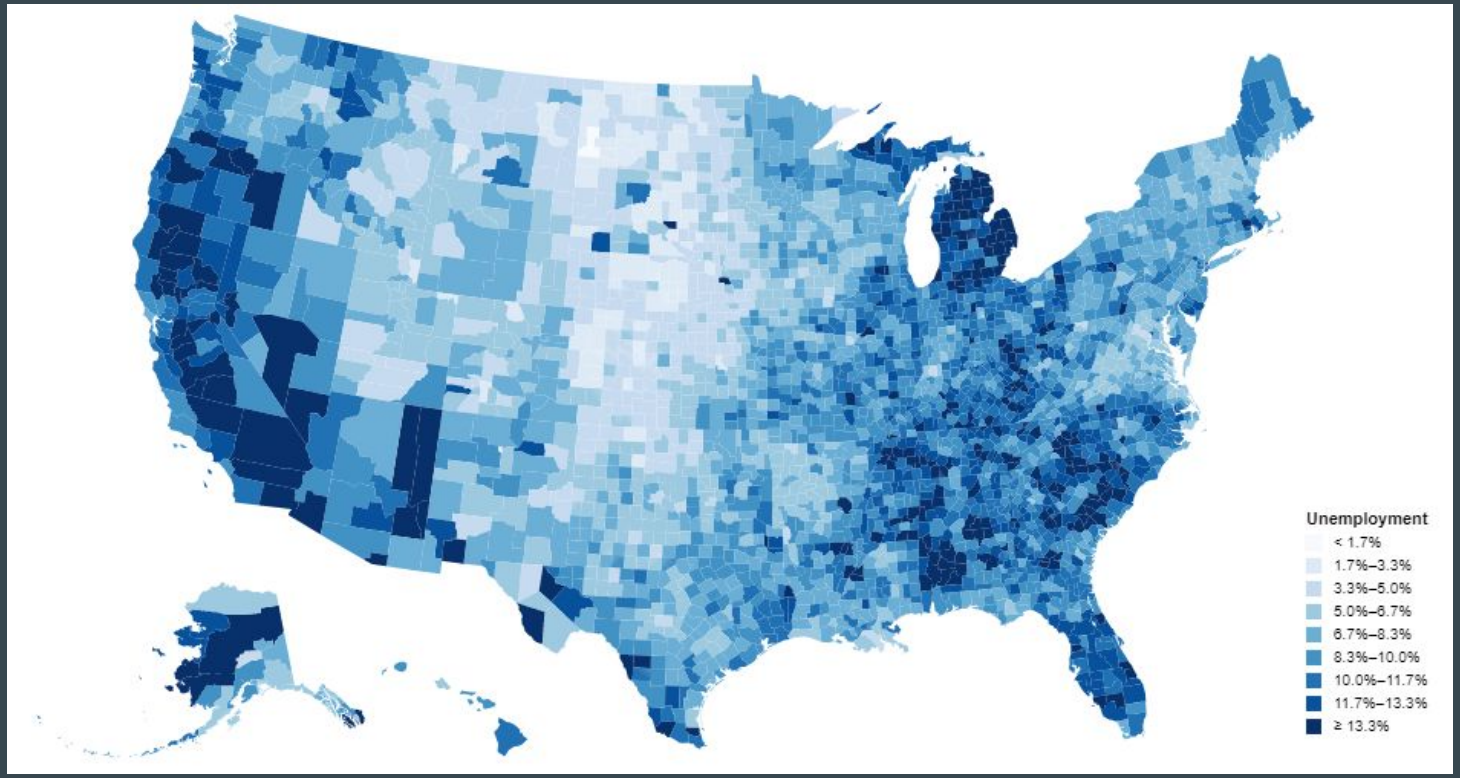


Scatterplot



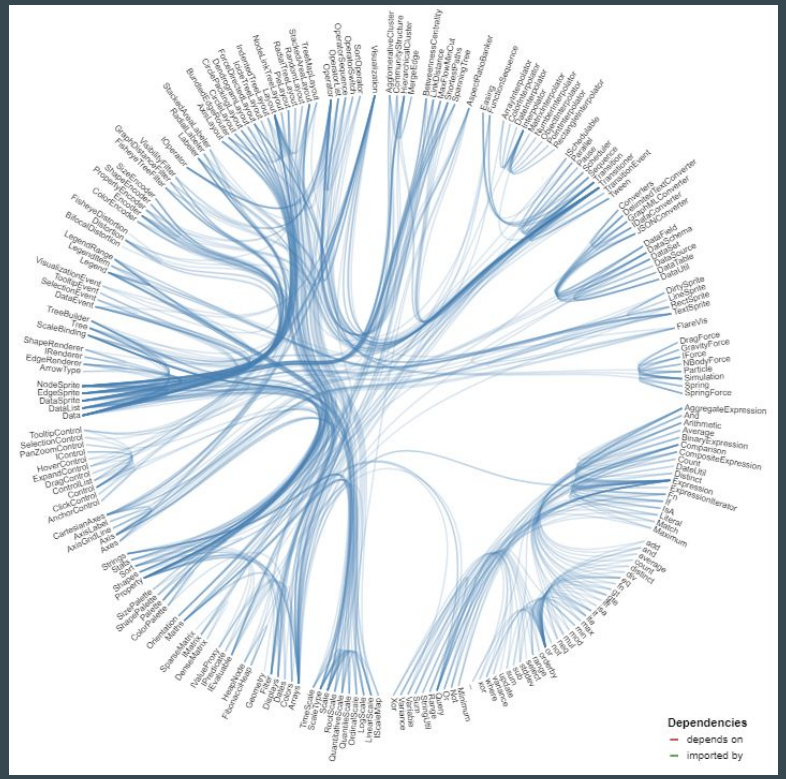
Connected Scatterplot

Geographic Maps



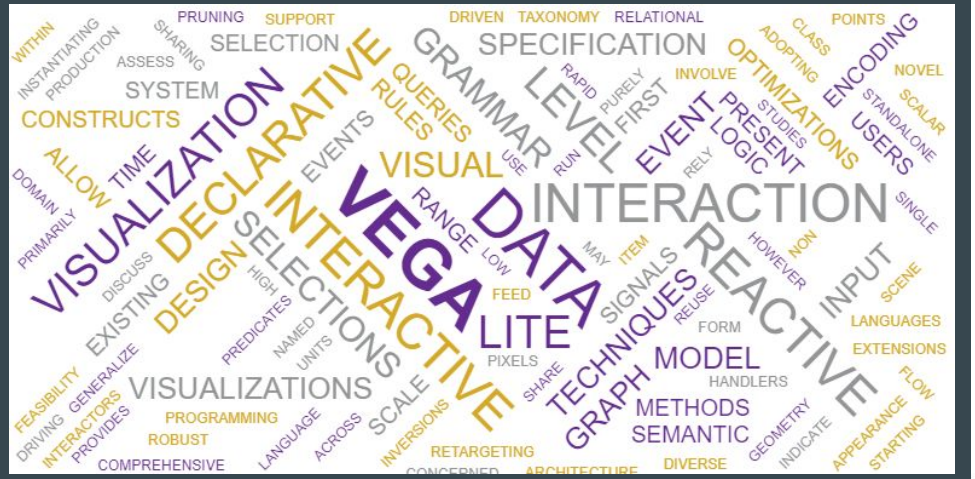
Maps

Network Diagrams

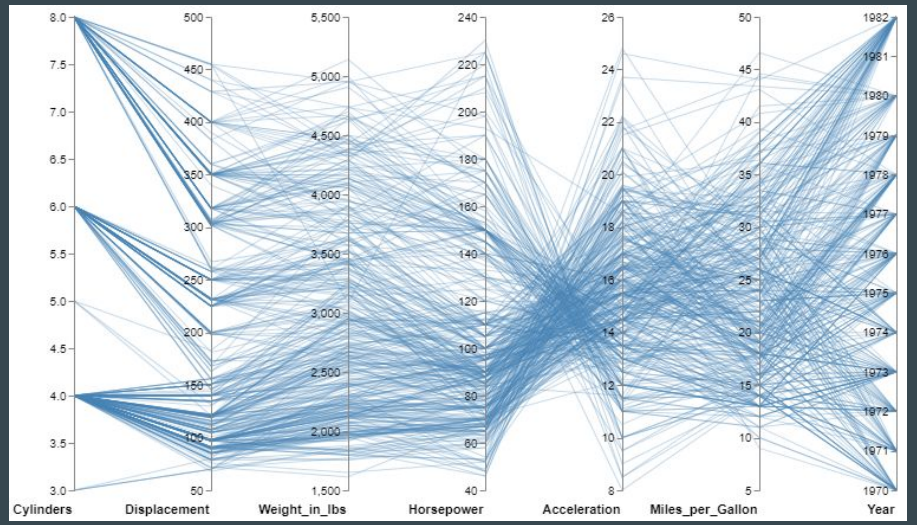


Edge Bundling

Chart Types



Word Cloud



Parallel Coordinates

Importando

```
<head>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/vega/3.0.7/vega.js"></script>
    <script
    src="https://cdnjs.cloudflare.com/ajax/libs/vega-embed/3.0.0-rc7/vega-embed.js">
    </script>
</head>
<body>
        <div id="vis"></div>
        <script>
        const spec = "exemplo.json";
        vegaEmbed('#vis', spec).catch(console.warn);
    </script>
</body>
```

vegaEmbed

- Vega-Embed fornece suporte avançado para incorporar visualizações interativas Vega em páginas da web.
- Tem como parâmetros: o ***elemento HTML***, o ***arquivo Vega*** e (opcional) um objeto js com ***opções de embed***.
- Algumas opções de embed:
  - `renderer`: tipo do elemento HTML para a *view*, podendo ser `"canvas"` (padrão) ou `"svg"`.
  - `actions`: ativar ou desativar ações de links, `true` (padrão) ou `false`.

Exercícios!

Editor Vega.js