

TP 3 – Visualisations Standards en D3.JS

Visualisation Interactive de Données

Romain Vuillemot – romain.vuillemot@ec-lyon.fr

Objectifs du TP

- Créer 3 graphiques standards avec D3.JS : scatterplot, line chart et histogramme
- Rajouter une interaction minimale avec des widgets externes
- Rendre ces graphiques ré-utilisables sous forme de fonctions

1 Scatterplot

Nous allons tout d’abord réaliser un scatterplot. Pour rappel, ce type de graphique encode des items sur 2 axes quantitatifs. Le plus souvent avec une marque de type cercle dont la taille et la couleur peuvent encoder respectivement deux quantité et catégorie supplémentaires. Nous appliquerons le scatterplot au jeu de données Iris¹.

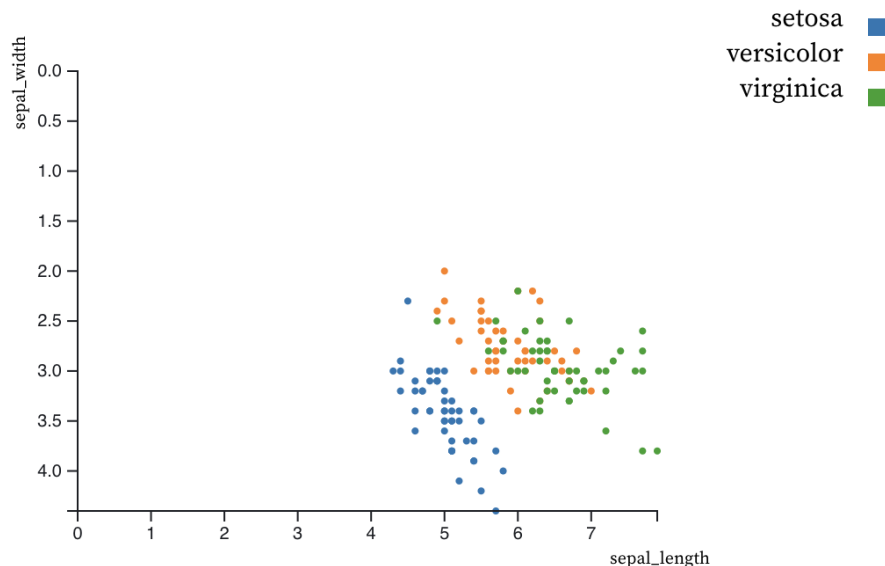


FIGURE 1 – Scatterplot à réaliser à partir des données Iris.

Les étapes de réalisation sont les suivantes :

- Réaliser une phase (rapide) de découverte de données (regarder les données brutes, types, etc.) et réaliser éventuellement des graphiques exploratoires avec Observable Plot.
- Réaliser le graphique de la figure 1 en D3.JS en partant de l’exemple du bar chart vu en cours.
- Pensez à bien utiliser les conventions de marges telles que :
<https://observablehq.com/@d3/margin-convention?collection=@d3/d3-axis>

1. <https://gitlab.ec-lyon.fr/rvuillem/visu/-/blob/master/dataset/iris.csv>

Une légende (en haut à droite sur la figure 1) permet à l'utilisateur de comprendre l'encoding des données.

- Rajouter une légende indiquant afin de comprendre le rôle de la couleur (que vous dessinerez avec un `<rect>`). Cette légende doit être générée à partir du jeu de données Iris automatiquement.
- Ecrire une fonction permettant de prendre en entrée le jeu de données et les paramètres du graphique. Voici un exemple d'une telle fonction (vous pouvez réaliser une fonction avec moins de paramètres bien sûr) :
<https://observablehq.com/@d3/bar-chart>
- Rajouter une liste de selection afin de choisir les paramètres de la visualisations (et donc de la fonction), voici un exemple de selection avec une liste de catégories :

```
viewof d = html`<select name="cars" id="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="mercedes">Mercedes</option>
  <option value="audi">Audi</option>
</select>`
```

BONUS : utilisez des symboles différents pour encoder la catégories des fleurs, au lieu d'utiliser un cercle. Conseil : utilisez le générateur de symboles de D3² tel que décrit ci-dessous combiné à un élément `<path>` et son attribut `d`.

```
shape = d3.scaleOrdinal()
  .domain(iris.map(d => d.species))
  .range(d3.symbols.map(s => d3.symbol().type(s)))
```

2. http://using-d3js.com/05_10_symbols.html

2 Line chart

Nous allons désormais réaliser un graphique en ligne (line chart). Pour rappel ce graphique permet de représenter des données quantitatives selon un axe temporel. Ce graphique va demander un peu plus de travail sur les données que le scatterplot du fait que chaque ligne représente tous les items d'une même catégorie à la fois.

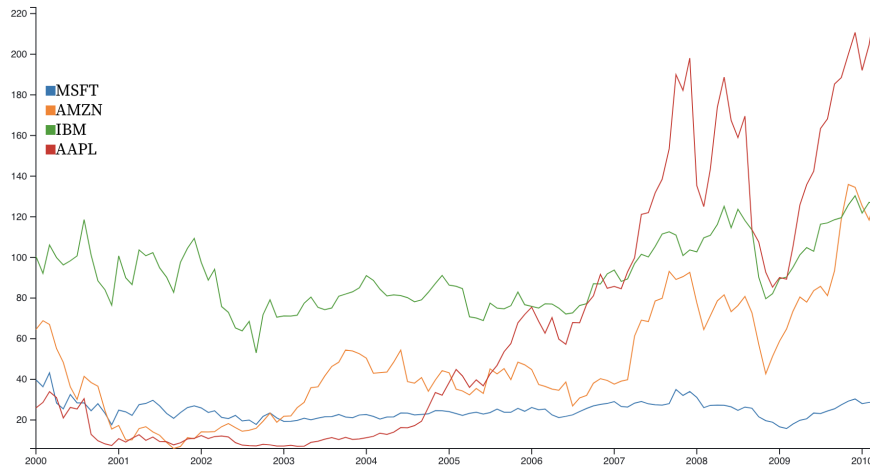


FIGURE 2 – Line chart à réaliser à partir des données de cours de bourse.

Chargez les données de cours bourse d'entreprises³ issues de l'API de Yahoo! Finance pendant une période donnée. Le prix de l'action est en dollars. Voici un échantillon de données :

```
symbol,date,price
MSFT,Jan 2000,39.81
MSFT,Feb 2000,36.35
MSFT,Mar 2000,43.22
MSFT,Apr 2000,28.37
```

- Réaliser des statistiques descriptifs sur les données en général (nombre d'entreprises, nombre de jours, valeurs min/max global pour le moment) ils pourront être utiles pour la conception du graphique. Eventuellement un ou plusieurs Observable Plot.
- Réaliser les étapes de parsing (typage de données) en utilisant la méthode `d3.timeParse`⁴ et le bon parsing de dates.

Nous allons désormais réaliser le line chart par entreprise, il est nécessaire de transformer les données en une collection de données pour chaque entreprise.

- Proposez une méthode de groupement (utilisation de `d3.group`⁵) et complétez les statistiques de la question précédente.

3. <https://gitlab.ec-lyon.fr/rvuille/visu/-/blob/master/dataset/stocks.csv>

4. <https://github.com/d3/d3-time-format>

5. <https://observablehq.com/@d3/d3-group>

Vous devrez obtenir des données similaires à celles-ci (pas forcément en JSON/Dictionnaire, cela peut être un `Set` ou une `Map` à itérer avec `Array.from()`.)

```
var stocks = [
  {symbol: "MSFT", values: [
    {date: "Jan 2000", price: 39.81},
    {date: "Feb 2000", price: 36.35},
    {date: "Mar 2000", price: 43.22},
    {date: "Apr 2000", price: 28.37}
  ]},
  {symbol: "AAPL", values: [
    {date: "Jan 2000", price: 123.45},
    {date: "Feb 2000", price: 98.76},
    {date: "Mar 2000", price: 56.54},
    {date: "Apr 2000", price: 12.34}
  ]},
  {symbol: "GOOG", values: [
    {date: "Jan 2000", price: 567.89},
    {date: "Feb 2000", price: 456.78},
    {date: "Mar 2000", price: 789.65},
    {date: "Apr 2000", price: 654.32}
  ]}
];
```

- Implémentez le line chart de la figure 2 en D3.JS en utilisant le générateur de ligne `d3-line` (<https://observablehq.com/@d3/d3-line>) qui permet de créer une marque de type `path SVG`.
- Pensez à bien démarrer d'un exemple de graphique avec conventions de marges mais en utilisant un `d3.scaleTime()` comme axe horizontal :
<https://observablehq.com/@d3/margin-convention?collection=@d3/d3-axis>
- Rajouter axes, couleurs, titres. Pensez bien à formater les années en utilisant la fonction de formatage de D3⁶ (similaire au parsing de la date).
- Rajouter une légende sous forme de rectangles par catégorie.

A noter que la figure 2 servira seulement d'exemple pour les choix esthétiques mais vous êtes libre de choisir d'autres paramètres visuels.

BONUS : dessiner la valeur cumulée du cours de bourse en calculant un champs supplémentaire et en mettant un widget adapté permettant de choisir cette valeur cumulée (ou non). Le module <https://observablehq.com/@d3/d3-cumsum> permet le calcul de telles valeurs.

6. <https://github.com/d3/d3-format>

3 Histogramme

A partir du bar chart vu en cours créez désormais un histogramme pour les valeurs quantitatives.

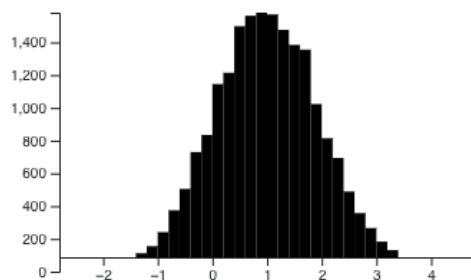


FIGURE 3 – Histogramme avec une dimension quantitative.

- Lire la documentation de la fonction `d3.bin()` ⁷
- Choisir un attribut quantitatif du jeu de données Iris et appliquer le binning dessus. Regarder les données générées. Vous pouvez aussi générer un tableau de valeur aléatoires pour tester avec une distribution intéressante, ex : normale. `Array.from(length: 20000, d3.randomNormal(1, 1))`.
- Appliquer le binning au bar chart, vous aurez un histogramme.

BONUS : réaliser une matrice de scatterplot du jeu de données Iris avec histogrammes pour chaque quantité en diagonale. Voici un exemple de matrice de scatterplot

<https://observablehq.com/@pstuffa/scatterplot-matrix-as-a-function>

7. <https://observablehq.com/@d3/d3-bin>