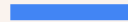




DATAACTIVIST



Comment réaliser des datavisualisations avec R ?

Première approche - jour 1

PROGRAMME JOUR 1

1. ***Ice breaker***
2. **Première partie : un peu de théorie**, *maîtriser les enjeux de la visualisation de données ;*
3. **Deuxième partie : la programmation, R et RStudio**, *comprendre les principes de la programmation, et les spécialités du langage R et de l'interface RStudio ;*
4. **Troisième partie : le tidyverse**, *connaître les principales fonctions pour nettoyer et manipuler les données, avec des exercices de mise en pratique ;*
5. **Conclusion : vos retours**, *que l'on espère positifs.*

PROGRAMME JOUR 2

1. **Première partie : dérouillage**, *réviser les fonctions présentées et assimilées lors de la première demi-journée, permettant la manipulation de données ;*
2. **Deuxième partie : ggplot ou comment faire de la visualisation sous R**, *appréhender la librairie de visualisation de données du tidyverse, avec des exercices de mise en pratique ;*
3. **Troisième partie : exercice final**, *reproduire une datavisualisation impliquant le traitement et la visualisation de données ;*
4. **Conclusion : vos retours, le mot de fin.**

JOUR 1

Avant de commencer...

Savez-vous réaliser une dataviz ?

Avez-vous déjà utilisé R ?

Créons les binômes

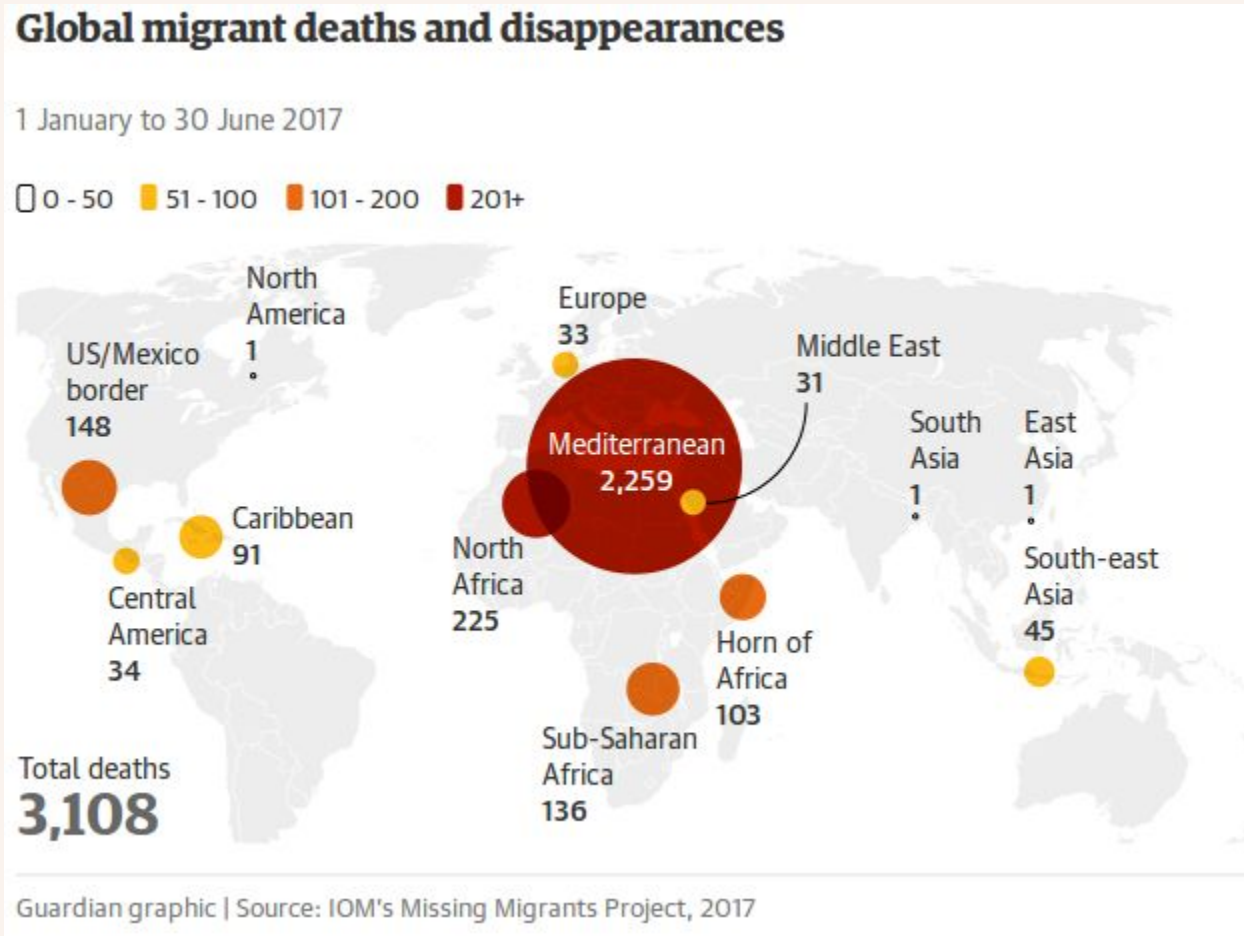
Un peu de théorie

Commençons par un exercice de **visualisation**

Selon l'Organisation internationale des migrations, 3108 migrant·e·s ont trouvé la mort ou sont disparu·e·s durant le premier semestre 2017.

- 2259 en Méditerranée
- 225 en Afrique du Nord
- 148 à la frontière américano-mexicaine
- 136 en Afrique subsaharienne
- 103 dans la Corne de l'Afrique
- 91 dans les Caraïbes
- 45 en Asie du Sud-Est
- 34 en Amérique centrale
- 33 en Europe
- 31 au Moyen-Orient
- un·e en Amérique du Nord
- un·e en Asie du Sud
- un·e en Asie de l'Est.

Une autre représentation



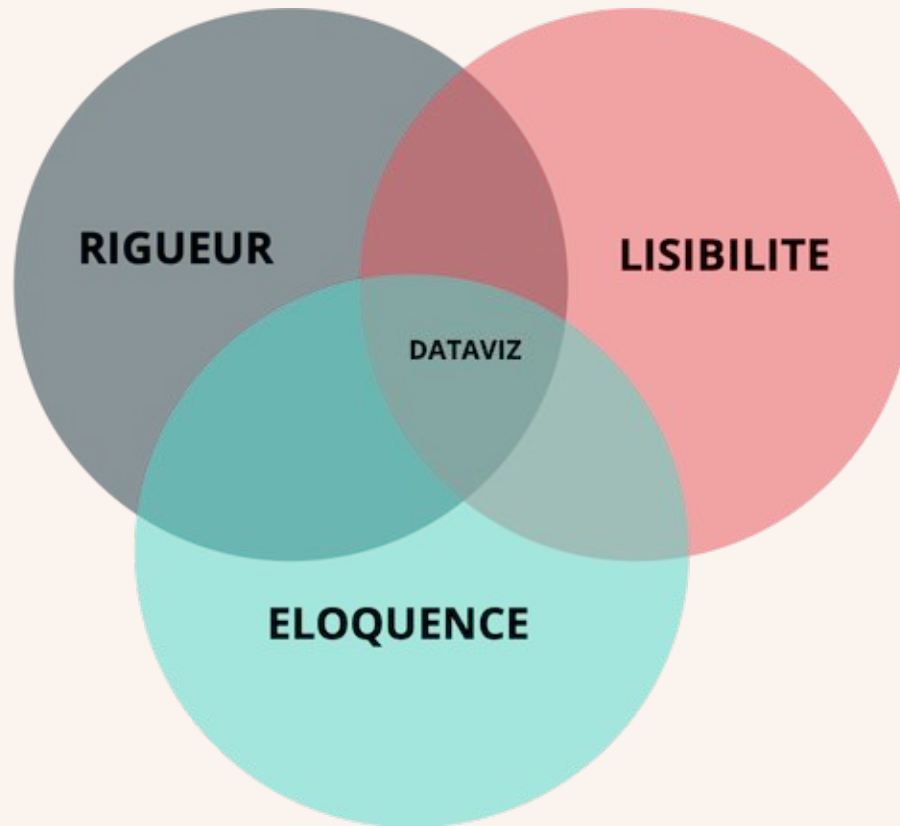
The Guardian, "Migrant sea route to Italy is world's most lethal",
11/09/2017

Démarche d'encodage / décodage

- La visualisation de la donnée consiste en un **encodage**, c'est-à-dire une "conversion" d'un format à un autre.
- En l'occurrence, le passage d'une forme brute non interprétée (les *data*), à une forme raffinée interprétée (la *dataviz*).
- La mise en forme implique également un **angle**, un *message* que doit transmettre l'image.
- Nous avons donc trois ingrédients indispensables à la démarche :
 1. des données (qu'il faudra peut-être retravailler) ;
 2. une mise en forme ;
 3. un message.

**Qu'est-ce qu'une
dataviz?**

Qu'est-ce qu'une **bonne dataviz** ?



**Premier objectif :
la rigueur**

Comprendre la nature et le sens des données

- **La dataviz étant un encodage**, elle substitue aux données des équivalences visuelles.
- L'encodage doit permettre :
 1. de comprendre **la nature** des données ;
 2. d'apprécier **le rapport** qu'elles entretiennent entre elles ;
 3. de saisir les points saillants et **phénomènes** clefs ;
 4. le tout sans déperdition de sens.
- Chacun de ces aspects repose sur des **caractéristiques visuelles liées à des représentations**.

Expliciter la nature de la donnée

Extensive Data Shows Punishing Reach of Racism for Black Boys

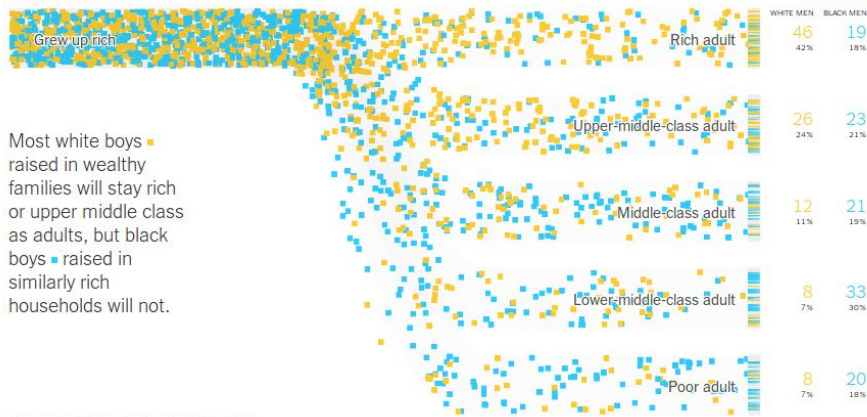
By EMILY BADGER, CLAIRE CAIN MILLER, ADAM PEARCE and KEVIN QUEALY MARCH 19, 2018

Black boys raised in America, even in the wealthiest families and living in some of the most well-to-do neighborhoods, still earn less in adulthood than white boys with similar backgrounds, according to a sweeping new study that traced the lives of millions of children.

White boys who grow up rich are likely to remain that way. Black boys raised at the top, however, are more likely to become poor than to stay wealthy in their own adult households.

Follow the lives of 2,433 boys who grew up in rich families ...

...and see where they end up as adults:



L'encodage doit permettre de saisir immédiatement le sens de chaque élément :

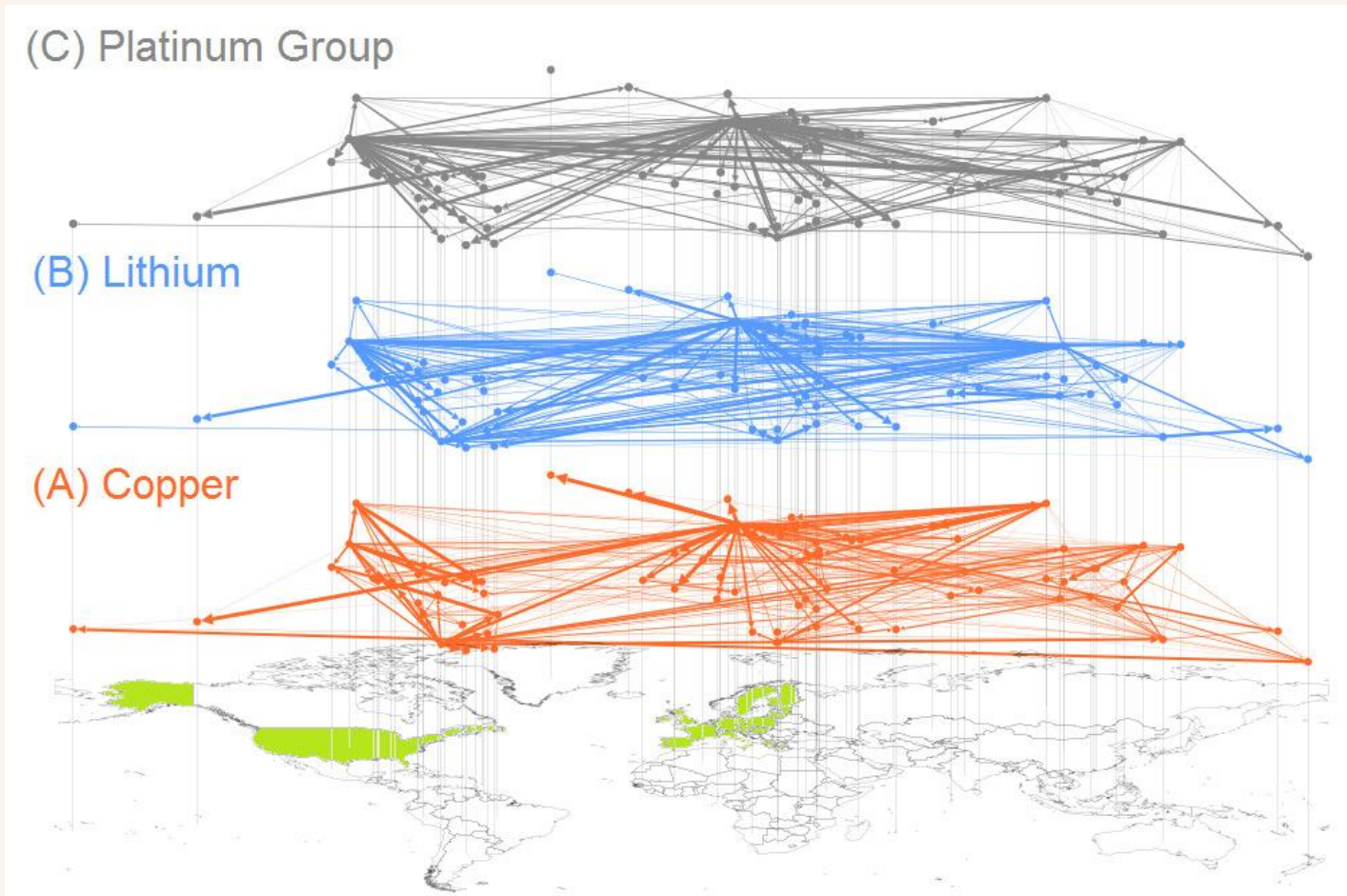
- Où sont les points de données ?
- Que représentent-ils individuellement ? Collectivement ?

La dataviz explicite ici le phénomène lui-même. Les causes, elles, sont explorées et détaillées dans le texte.

([source](#))

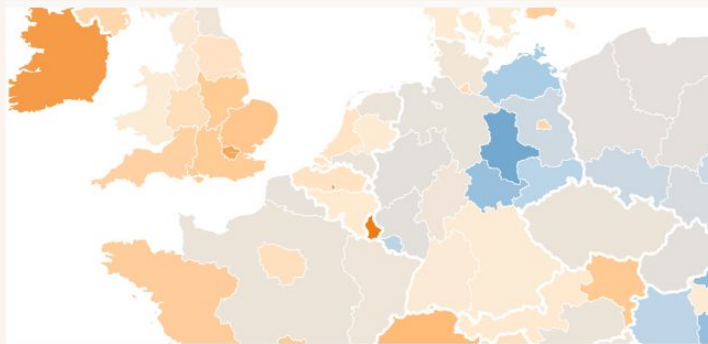
Choisir la **représentation adaptée au phénomène**

Un mauvais choix graphique peut fausser la représentation de la donnée en ne donnant pas à constater le phénomène qu'elles permettent de décrire.

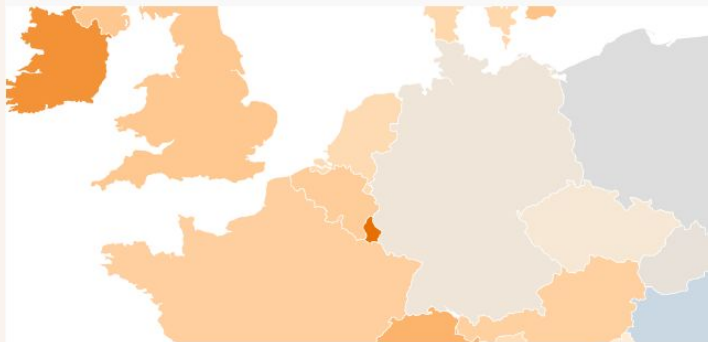


Choisir le bon niveau de granularité

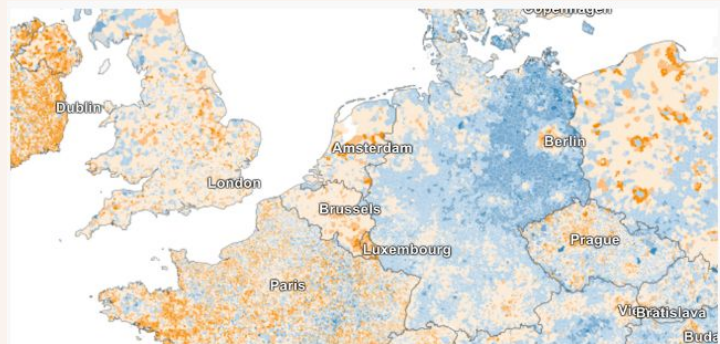
Le niveau de précision minimale d'une donnée (aussi appelé **granularité**) peut modifier de façon radicale la lecture d'un phénomène. Dans le cas d'une carte, le choix de l'échelle peut mener à la généralisation de phénomènes extrêmement circonscrits (ou inversement) comme le montre cet [exemple sur le blog de Datawrappier](#)



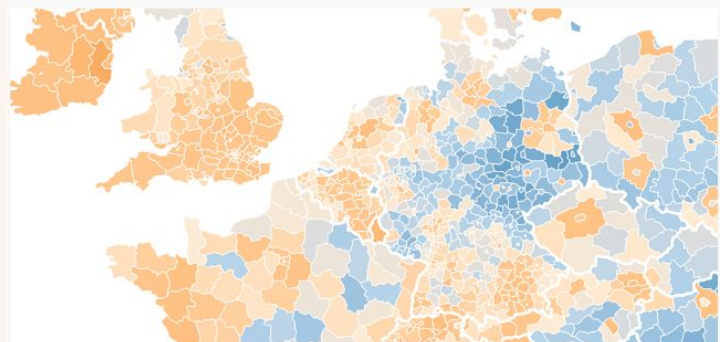
NUTS-1: Almost all regions get more washed out. Only regions in which sub-areas have the same kind of extreme increase or decrease in population stand out, e.g. Ireland, Luxembourg or center Germany.



Countries: As and all the nuance is gone. Only Ireland and Luxembourg stand out. However, it's the only map that shows us if a country population has increased or decreased in the last two decades.



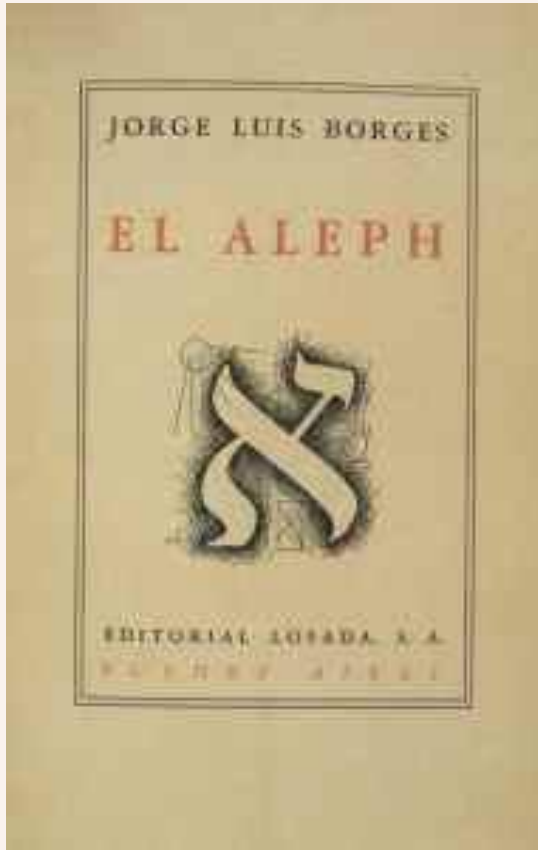
Municipalities: In Germany and Poland, cities are the lone orange wolfs. France and UK seem like a mixed bag.



NUTS-3: The Berlin area that grows in population seems far bigger now. In Poland, we can still see metropolitan areas thanks to nicely drawn region borders. France is still a mixed bag, but the UK shows a clear growth now.

Deuxième objectif : la lisibilité

Un arbitrage nécessaire



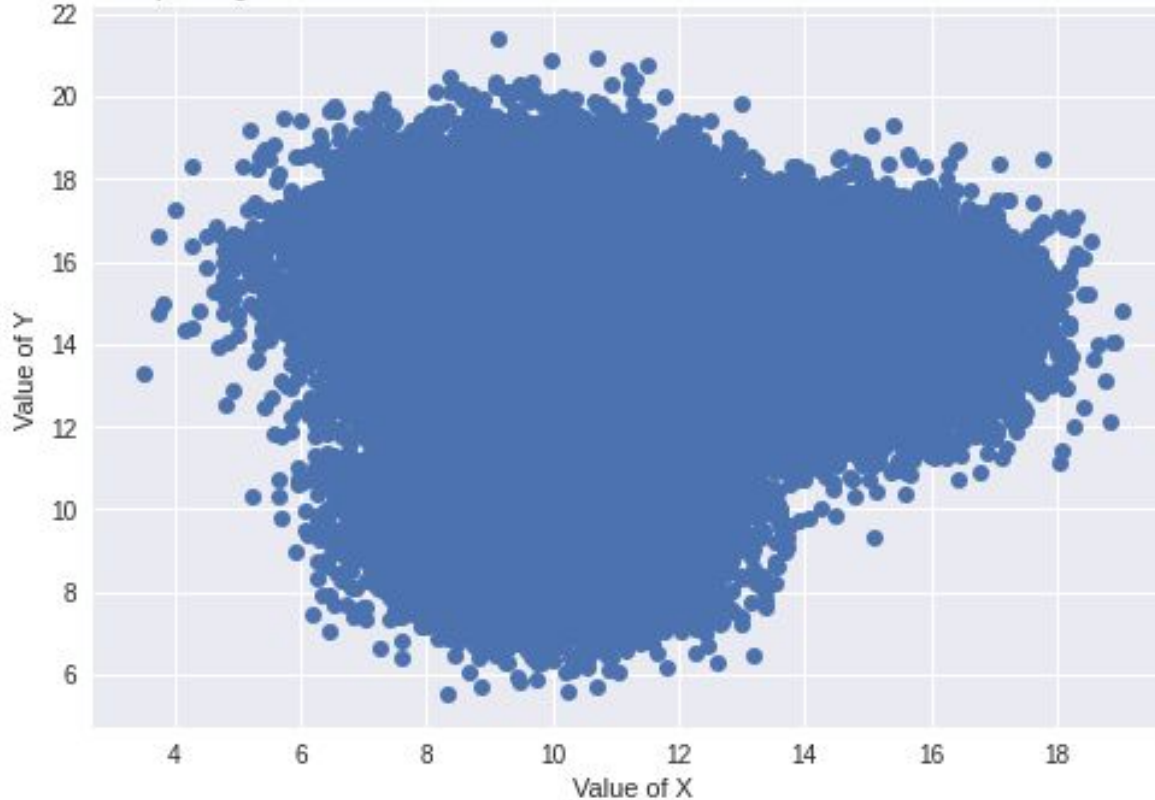
Jose Luis Borges, "De la rigueur de la science", 1946

L'allégorie de la carte "à l'échelle 1:1" de Borges donne une bonne idée du problème :

En cet empire, l'Art de la Cartographie fut poussé à une telle Perfection que la Carte d'une seule Province occupait toute une ville et la Carte de l'Empire toute une Province. Avec le temps, ces Cartes Démesurées cessèrent de donner satisfaction et les Collèges de Cartographes levèrent une Carte de l'Empire, qui avait le Format de l'Empire et qui coïncidait avec lui, point par point.

1^{er} risque : "l'overplotting"

Overplotting looks like that:



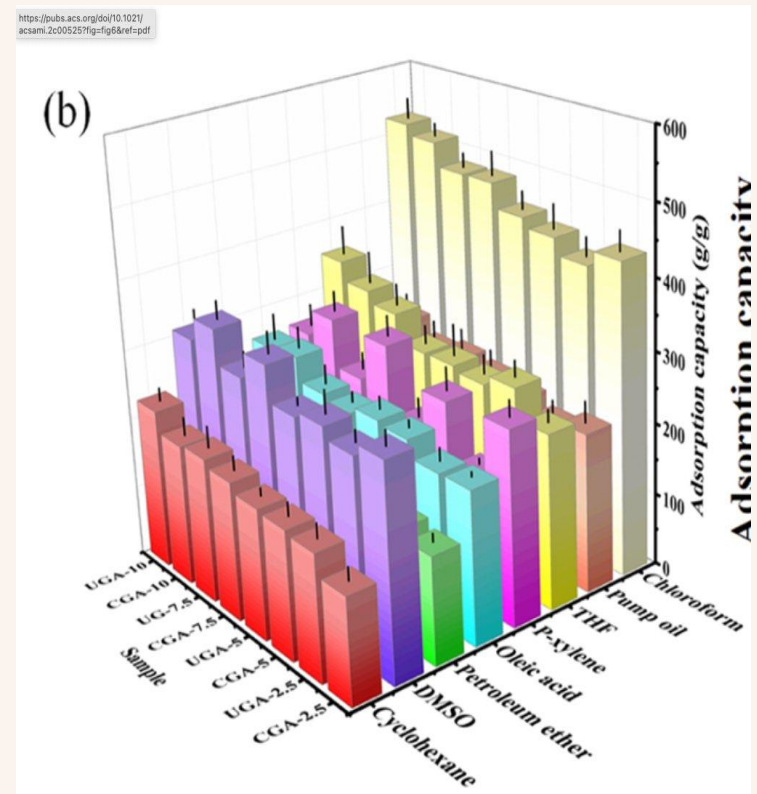
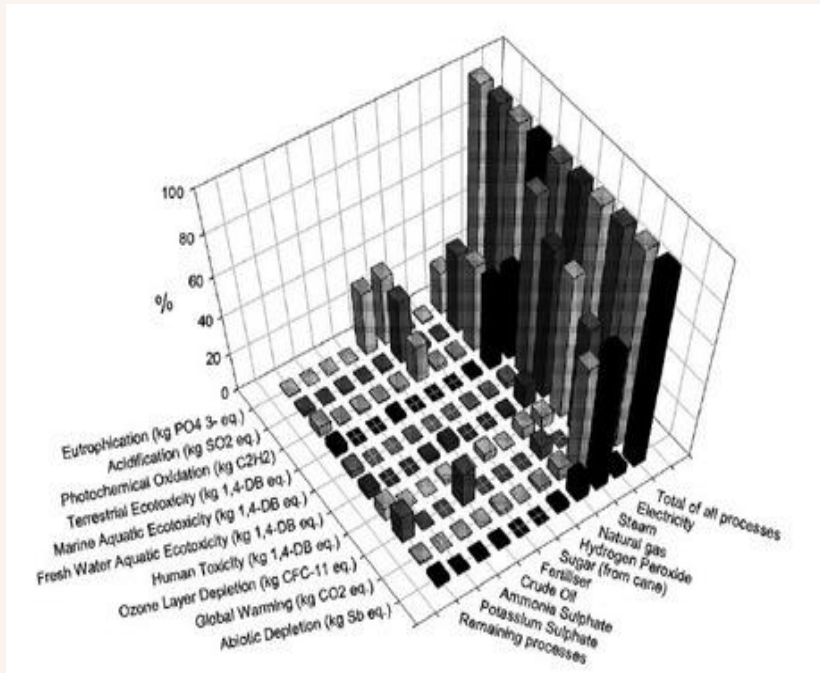
En statistique, le terme "overplotting" se réfère à l'effet produit par l'intégration d'un trop grand nombre de points de données dans un graphe, le rendant illisible.

Note : c'est un vrai terme de stat, vous pouvez l'utiliser pour vous la jouer.

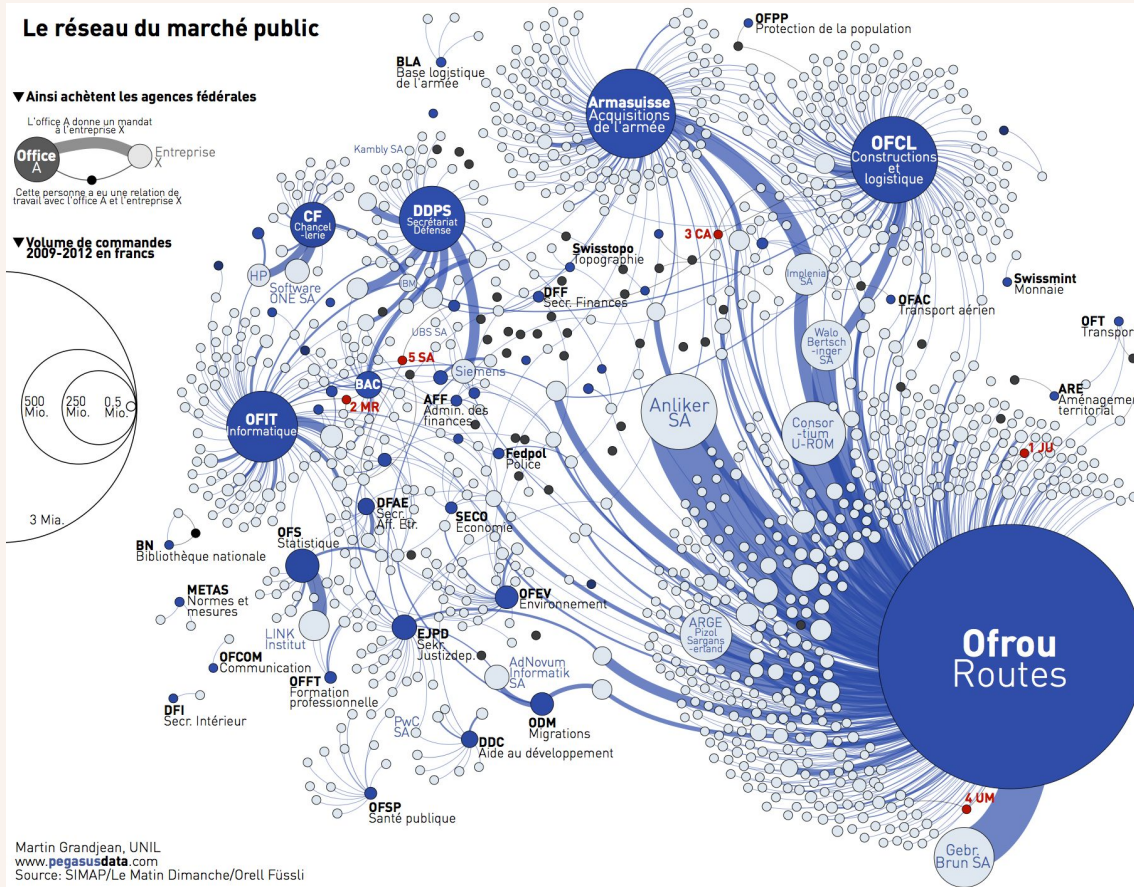
2^e risque : "l'overcomplicated"

L'autre risque courant est de multiplier les dimensions et axes de lecture jusqu'à rendre le graphique incompréhensible.

Note : ce terme n'a rien d'officiel, ne l'utilisez pas pour vous la jouer.



3^e risque : l'excès d'esthétisme

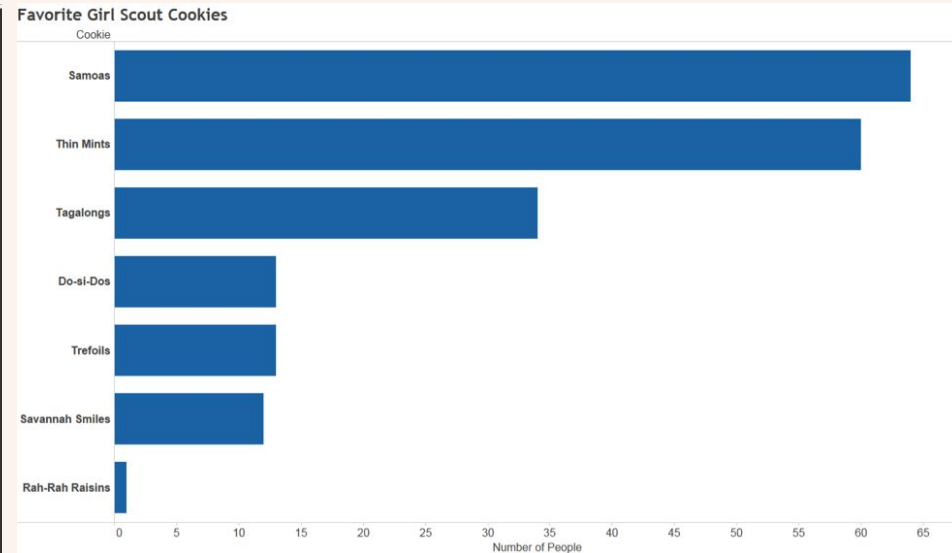
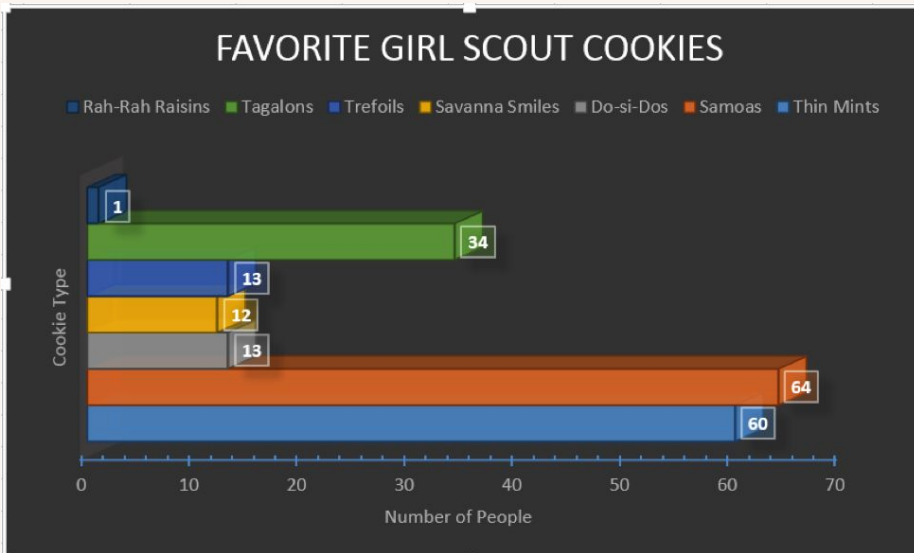


La tentation de mêler précision et esthétique peut produire de très beaux graphes... trop compliqués pour être compris du premier coup.

Martin Grandjean, *Pegasus Data*

Laissez parler les données

Les fioritures peuvent constituer des distractions : quand les données sont claires, autant leur laisser le champ libre.



Nazirah Jetha, "[5 data-viz tips to let your data speak for itself](#)", Mai 2016

Sans message, l'image devient vaine

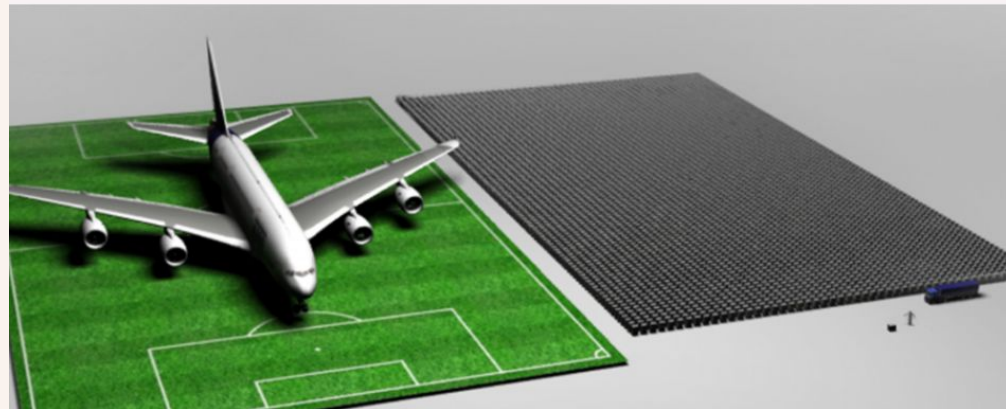
Il peut arriver que la dataviz n'ait d'intention que décorative ou spectaculaire. Précise et lisible, elle devient un panneau vide de sens et de propos qui n'explicite rien du monde faute de choisir comment parler de son sujet.

Ex. : une "visualisation concrète de la dette française" qui n'a aucun sens

Dette publique: 2.000 milliards d'euros, ça représente deux Stade de France en billets de 100 euros

Fred Hasselot et Jean-Marie Pottier — 30 septembre 2014 à 13h50 — mis à jour le 30 septembre 2014 à 13h54

Une visualisation concrète de la dette de la France.

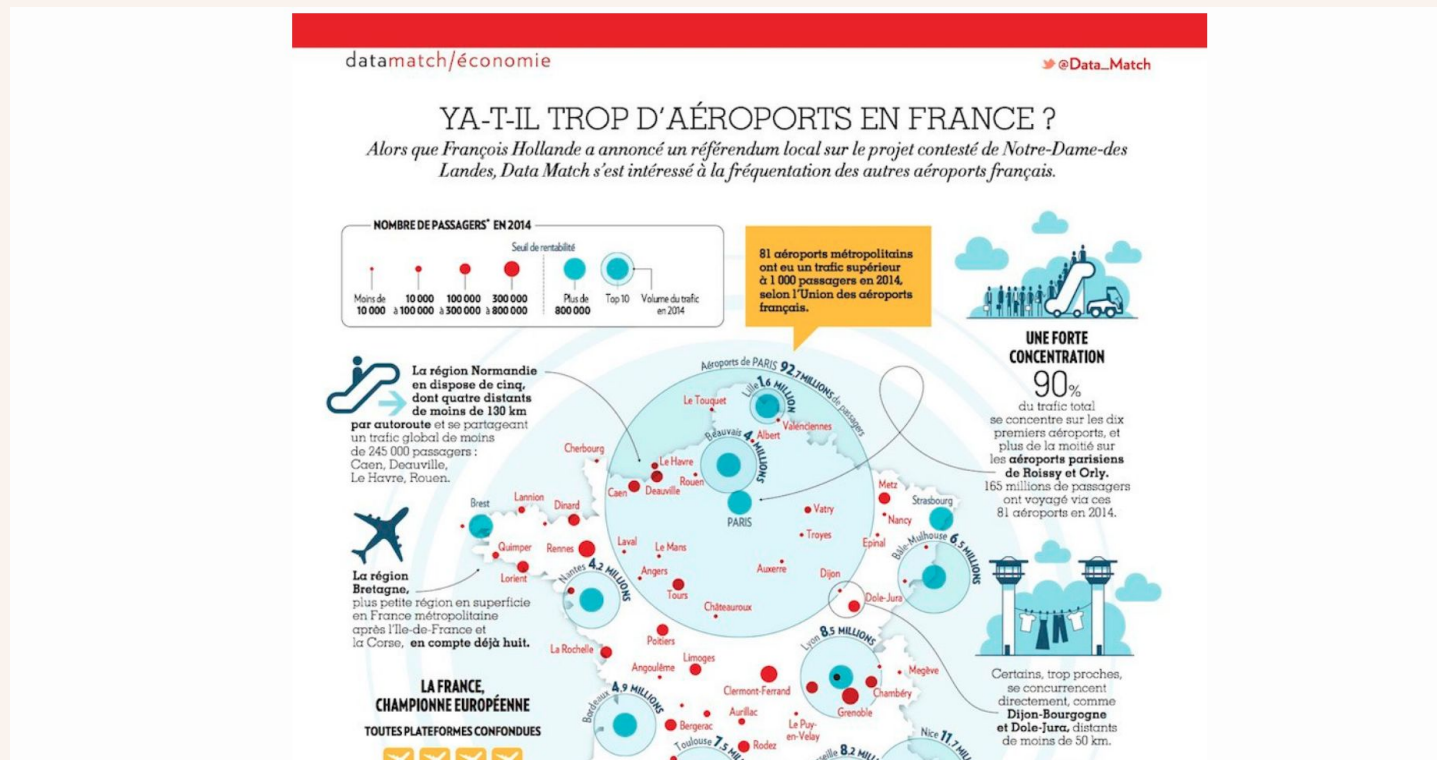


Troisième objectif : l'éloquence

Donner des points de repère

La façon la plus simple de mettre en avant un message reste encore de l'expliciter : souligner des points saillants ou relever les infos clefs permet de guider dans la lecture.

Ex. : rubrique DataMatch de Paris Match



La symbolique **comme message**

Le choix d'une représentation figurative allégorique plutôt que littérale peut constituer en soi l'angle d'une dataviz. Claire et bien choisie, la référence agit alors comme un sous-texte à l'image.

Pour illustrer la censure en Iran, l'équipe de *Journalism is not a crime* a adopté l'esthétique des mosaïques d'Ispahan, faisant écho à l'imaginaire visuel du pays et à sa grandeur. Contrepied de la politique de censure menée par le régime ainsi dénoncé.



Quel message veut-on faire passer ?

- Une dataviz porte avant tout une **intention**, la volonté de transmettre une information.
- De cette intention découlera le **type de dataviz** à utiliser.

Qu'est-ce que nous voulons faire ?	Choix de présentation
Comparer des valeurs de différentes catégories	Diagramme à barres
Suivre la valeur au fil du temps (séries chronologiques)	Graphique en courbes
Afficher l'interaction entre deux valeurs	Nuage de points
Afficher les données relatives à la géographie	Carte

Des outils pour vous aider après la formation

Pour choisir le bon format de visualisation de données, comme on l'a vu, il existe plusieurs outils qui tendent à se multiplier. **Nous en retiendrons trois :**

Le carnet de formation "Visualisation des données ouvertes"

La qualification des données

Les données catégorielles (ou qualitatives)

Une variable catégorielle est une variable qui désigne des catégories ou des niveaux.
Parfois, une variable catégorielle est représentée par des nombres, mais les additionner (ou soustraire, multiplier, etc...) n'a pas de sens. C'est le cas des numéros de départements.

Exemples :
Mentions au bac, couleur des yeux, genre ou statut.

Statut

En travaux
Fermé
Ouvert

Les données numériques (ou quantitatives)

Une variable numérique est une variable ou grandeur qui est représentée par des nombres que l'on peut additionner, soustraire, multiplier, etc...

Exemples :
Une taille, un montant ou une fréquence (le nombre de signalements).

Montant

300
300
153

Les données géographiques

Des données sont géographiques si un champ du jeu de données permet de localiser une information sur une carte.

Exemples :
Coordonnées géographiques, adresses, codes INSEE ou codes postaux.

Longitude	Latitude
2.169...	48.924...
6.168...	48.613...

Les données historisées

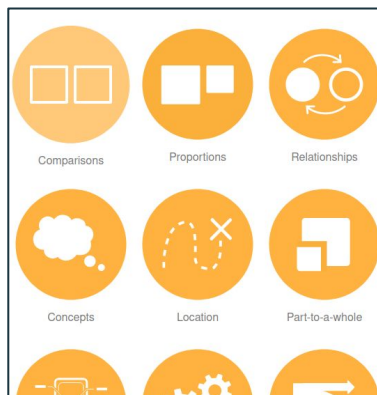
Des données historisées comprennent un ou plusieurs champs qui de donnée temporelle permettant de situer une information dans le temps.

Exemples :
Date, date et heure, minutes et secondes.

Date

2020
2021
2022

Trois sites en ligne de choix de dataviz



From data to viz :

<https://www.data-to-viz.com/>

The DatavizProject :

<https://datavizproject.com/>

Dataviz Catalogue :

<https://datavizcatalogue.com/>

"Culture D" d'OpenDataFrance

10 ALTERNATIVES AU CAMEMBERT

Le catalogue des datavisualisations s'est considérablement étoffé ces dernières années. Voici 10 pistes de graphiques pour représenter une répartition ou une évolution. Avec aucune prétention d'exhaustivité.

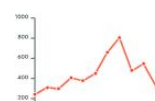
Anneau

Déclinaison du camembert, le graphique en anneau ou "donut" révèle la part relative de catégories dans un tout. Il est d'autant plus parlant que les chiffres (et donc les angles) ont des écarts importants. Il est à déconseiller quand il y a plus de cinq catégories. Le centre de l'anneau peut être utilisé pour valoriser un titre ou un chiffre.



Courbe

Une courbe sert à décrire une évolution, la "pente" entre deux valeurs permettant de la visualiser. Le choix de l'échelle des ordonnées, le format (portrait ou paysage), et la taille même du graphique jouent un grand rôle dans l'interprétation des



Le guide (avec un lien vers les fiches):
<https://opendatafrance.gitbook.io/editorialisation-des-donnees-publiques/>



Data visualization with ggplot2 :: CHEAT SHEET

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and **geoms**—visual marks that represent data points.



Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

```
a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))
```

a + geom_blank() and **a + expand_limits()**
Ensure limits include values across all plots.

b + geom_curve()(aes(yend = lat + 1, xend = long + 1), curvature = 1) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

TWO VARIABLES

both continuous

```
e <- ggplot(mpg, aes(cty, hwy))
```

e + geom_label()(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_point()
x, y, alpha, color, fill, shape, size, stroke

continuous bivariate distribution

```
h <- ggplot(diamonds, aes(carat, price))
```

h + geom_bin2d()(binwidth = c(0.25, 500))
x, y, alpha, color, fill, linetype, size, weight

h + geom_density_2d()
x, y, alpha, color, group, linetype, size

h + geom_hew()

[Cheat sheets data-visualisation avec ggplot2](#)

La programmation, R et RStudio

```
    }).done(function(response) {  
      for (var i = 0; i < response.length; i++) {  
        var layer = L.marker(  
          [response[i].latitude, response[i].longitude]  
          // , {icon: myIcon}  
        );  
        layer.addTo(group);  
  
        layer.bindPopup(  
          "<p>" + "Species: " + response[i].species + "<br>" +  
          "<p>" + "Description: " + response[i].description + "<br>" +  
          "<p>" + "Seen at: " + response[i].latitude + " " + response[i].longitude + "<br>" +  
          "<p>" + "On: " + response[i].sighted_at + "</p>"  
        );  
      }  
  
      $('select').change(function() {  
        species = this.value;  
      });  
    });  
  }  
  $.ajax({  
    url: queryURL,  
    method: "GET"  
  }).done(function(response) {  
    for (var i = 0; i < response.length; i++) {  
      var layer = L.marker(  
        [response[i].latitude, response[i].longitude]  
        // , {icon: myIcon}  
      );  
      layer.addTo(group);  
    }  
  });  
}
```

Avant de commencer, un support complet pour approfondir les notions :

Introduction au langage de programmation R

Cette formation a pour but de rendre **accessible à tous** la programmation en R. Pour cela, nous introduirons la notion de programmation dans un contexte général, puis nous nous pencherons sur le langage R en partant d'une vue globale pour entrer petit à petit dans les manipulations techniques.

https://dianethy.github.io/cours_R/Introduction_R.html

La programmation, tout un concept :

- **Définition** : ensemble des activités qui permettent l'écriture de programmes informatiques.
- **Synonymes** : codage, développement.
- **Contexte** : révolution informatique, 2.5 trillions d'octets de données créés chaque jour.
- **Utilité** : obtenir des informations et connaissances à partir de données.
- **Usage** : de nombreux langages existent (HTML, Javascript, Java, C, Python, R...)

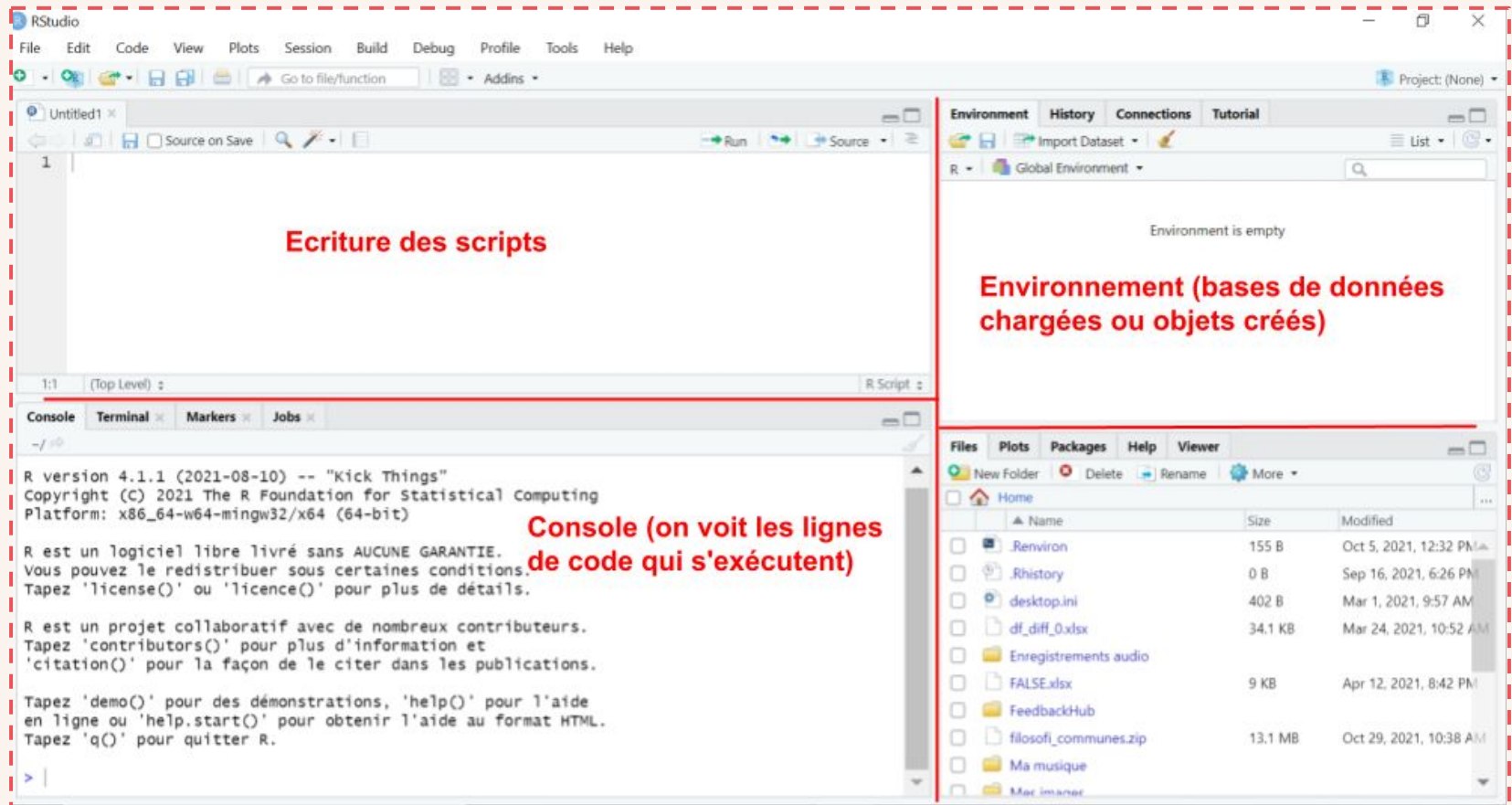
R, un langage de programmation **destiné aux statistiques**

If statistics programs/languages were cars...



Gratuit, puissant, design, avec une communauté importante ([Stack Overflow](#), [R-bloggers](#) etc.)

RStudio, une interface permettant d'interagir avec R



Les concepts clés :


漢

- run
- package
- library
- script

Abc

- **exécuter** (une ligne de code)
- **ensemble de fonctions**
- **endroit où sont gardés les packages**
- **document où l'on code** (extension *.R*)

Assez parlé, ouvrons RStudio !

- **Créer un nouveau script** via *File > New File > R script*
- **Commencer à interagir par de simples opérations arithmétiques:** $2+3$
- **Assigner des valeurs à des objets:**
 - $x \leftarrow 2$ (ou $x = 2$ mais la bonne pratique est d'utiliser ' \leftarrow ')
 - $y \leftarrow 7$
 - $resultat \leftarrow x * y$
 - $resultat$
- **Exécuter les lignes de code:** *Ctrl/Enter* ou bouton 
- **Commenter le code** via le `#`

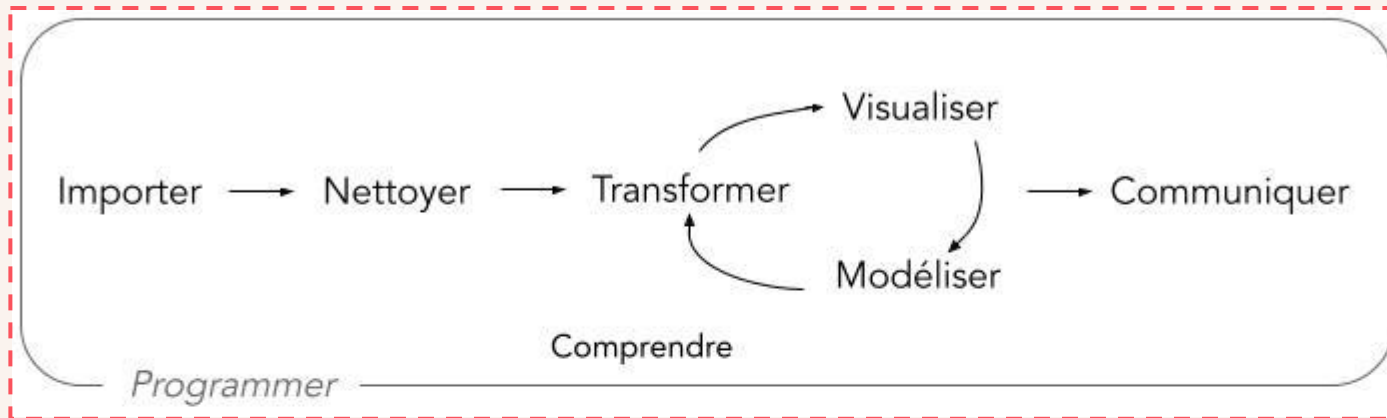
Le tidyverse



Installation des packages

- **Stockage des packages R sur le CRAN**
 - `install.packages("readxl")`
 - `install.packages("tidyverse")`

Schéma d'exploitation des données



source

Deux manières de coder en R

- **R-base**

- exemple 1 : `exemple[,2]`
- exemple 2 : `f(g(h(exemple)))`

- **Tidyverse**

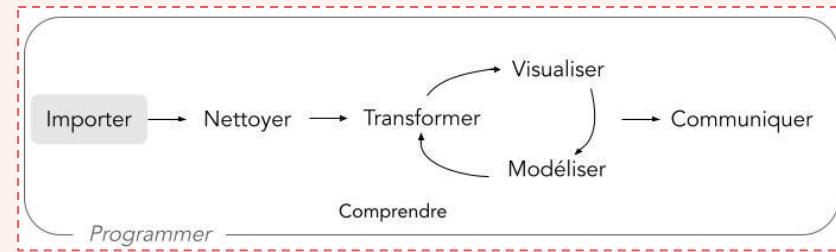
- exemple 1 : `exemple %>% select(2)`
- exemple 2 : `exemple %>% h() %>% g() %>% f()`



Le tidyverse fonctionne grâce au pipe (`%>%`), qui permet d'appliquer des fonctions à une base de données. L'opération à droite du pipe est appliquée à la valeur située à gauche du pipe, il devient alors possible d'enchaîner les traitements.

Codons ensemble avec les principales fonctions du tidyverse

Importer des données



- **Excel**
 - `library(readxl)`
 - `exemple <- read_excel(path = "fichier.xlsx", sheet = "nom_feuille")`
- **CSV : valeurs séparées par des virgules**
 - `library(readr)`
 - `exemple <- read_csv("fichier.csv")`
- **SCSV : valeurs séparées par des points-virgules (“faux” CSV)**
 - `library(readr)`
 - `exemple <- read_delim("fichier.csv", delim = ";")`
- **À vous !** Importer le jeu de données **‘Médiathèques - Ouvrages acquis par les médiathèques de Lille, Lomme et Hellemmes’** disponible sur le portail open data de la MEL :
 - au format CSV en utilisant le lien
 - si vous avez fini, au format tableur Excel en exportant le fichier

Importer des données

Solution

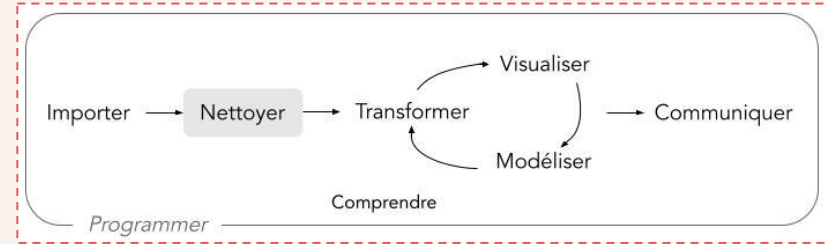
- Importer le jeu de données '**Médiathèques - Ouvrages acquis par les médiathèques de Lille, Lomme et Hellemmes**' disponible sur le portail open data de la MEL :
 - en CSV en utilisant le lien

```
library(tidyverse)
mediatheques <-
read_delim("https://opendata.lillemetropole.fr/api/explore/v2.1/catalog/datasets/ouvrages-acquis-par-les-mediatheques-/exports/csv?lang=fr&timezone=Europe%2FBerlin&use_labels=true&delimiter=%3B", delim = ";")
```

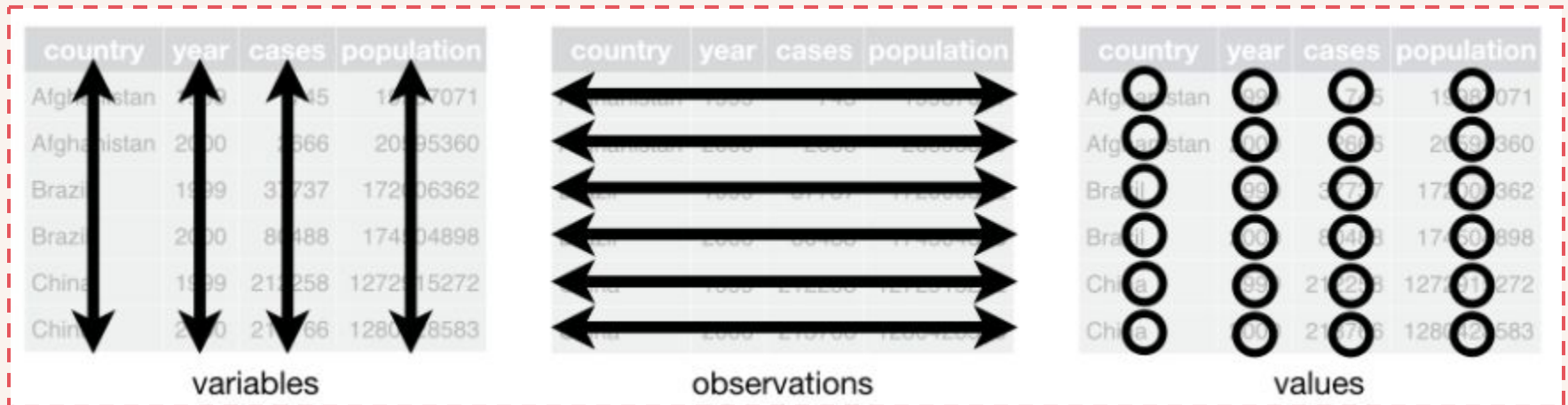
- en excel en exportant le fichier

```
library(readxl)
mediatheques <-
read_excel("~/Downloads/ouvrages-acquis-par-les-mediatheques-.xlsx")
```

Nettoyer des données



- **Structure de données propres**



- chaque ligne est une observation (entrée, enregistrement)
- chaque colonne est une variable
- une seule valeur dans chaque cellule

Nettoyer des données

- À vous !

- Quel jeu de données ci-dessous est proprement structuré ?
- Qu'est-ce qui ne va pas dans les 2 autres jeux ?

country	year	rate
Afghanistan	1999	745/19987071
Afghanistan	2000	2666/20595360
Brazil	1999	37737/172006362
Brazil	2000	80488/174504898
China	1999	212258/1272915272
China	2000	213766/1280428583

A.

country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

B.

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

C.

Nettoyer des données

Solution

- Quel jeu de données ci-dessous est proprement structuré ?
 - Le C.
- Qu'est-ce qui ne va pas dans les 2 autres jeux ?
 - Dans le A. on trouve 2 valeurs dans 1 case
 - Dans le B. les valeurs '*cases*' et '*population*' ne sont pas dans 2 colonnes séparées

Nettoyer des données

Avant de commencer toute analyse, il est nécessaire d'observer les données pour détecter les anomalies

- **Types de données**

Famille de données	Structure	Nom retourné sous R	Exemple
quantitatif	entier	int = integer	3
	décimal inexact	dbl = double	3.4
	décimal exact	num = numeric	3.400001
qualitatif	chaîne de caractères	chr = character	Ville de Paris
	facteur à n niveaux	Factor	petit / moyen / grand
	facteur à 2 niveaux dit "booléen"	bool = boolean	0 / 1, True / False
autre	date	POSIX	13-12-1998

- **Les fonctions pour observer les données**

- ***glimpse(data)*** : structure des données
- ***names(data)*** : nom des colonnes
- ***summary(data\$column)*** : résumé des valeurs
- ***table(data\$column)*** : occurrences des valeurs

- **À vous !** Répondre aux questions suivantes :

- le typage de la variable 'copy_number' est-il correct ?
- quel est le nombre de prêts maximal ?
- combien de vidéos ont été acquises par les médiathèques ?

Nettoyer des données

Solution

- le typage de la variable '*copy_number*' est-il correct ?

Le numéro de copie correspond à un identifiant et non à un nombre à proprement parler (somme, taux, note etc.), donc doit être considéré comme une chaîne de caractères, qui correspond au format '*character*' sous R. Or, avec la fonction **glimpse(data)** on voit qu'il est considéré comme un nombre ('*dbl*'), donc son typage est **incorrect**.

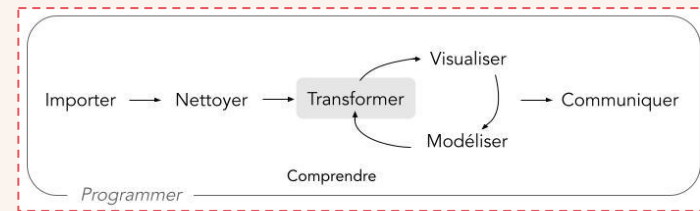
- quel est le nombre de prêts maximal ?

La fonction **summary(mediatheques\$nb_prets)** nous informe que le maximum est de **62**.

- combien de vidéos ont été acquises par les médiathèques ?

La fonction **table(mediatheques\$libelle_support)** nous informe que **6579** vidéos ont été acquises.

Transformer des données



Les 6 principales fonctions du **dplyr**, package de transformation de données contenu dans le **tidyverse**, sont les suivantes :

- **rename()** : pour renommer des colonnes
- **select()** : pour sélectionner ou supprimer certaines colonnes, et/ou changer leur ordre
- **filter()** : pour sélectionner certaines lignes selon une condition
- **mutate()** : pour créer de nouvelles variables
- **summarise()** : pour résumer plusieurs valeurs en 1 valeur
- **group_by()** : pour grouper les observations avant d'appliquer une fonction

Transformer des données : `rename()`

Cette fonction permet de renommer une ou plusieurs colonnes d'une base de données

- **Paramètres :**

- `data <- data %>% rename(nouveau_nom = `ancien nom`,
nv_nom = ancien_nom)`

- **Exemple :**

- `exemple <- exemple %>% rename(tx_chomage = `Taux de chômage`,
regions = régions)`

- **À vous !**

- renommer la colonne 'Année de publication' par 'annee_publication'
 - renommer la colonne 'date de reception' par 'date_reception'
 - renommer la colonne 'Bibliothèque' par 'bibliotheque'
 - renommer la colonne 'Type de document' par 'type_document'

Transformer des données : `rename()`

Solution

- renommer la colonne 'Année de publication' par 'annee_publication'
- renommer la colonne 'date de reception' par 'date_reception'
- renommer la colonne 'Bibliothèque' par 'bibliotheque'
- renommer la colonne 'Type de document' par 'type_document'

```
mediatheques <- mediatheques %>%  
  rename(annee_publication = `Année de publication`,  
         date_reception = `date de reception`,  
         bibliotheque = Bibliothèque,  
         type_document = `Type de document`)
```

Transformer des données : `select()`

Cette fonction permet de sélectionner certaines colonnes, par leur nom ou leur position dans la base de données

- **Paramètres :**
 - `data <- data %>% select(col1, col3, col2, col4)`
- **Exemples :**
 - `exemple <- exemple %>% select(-2)`
 - `exemple <- exemple %>% select(1:3, 7, 4:6)`
- **À vous !**
 - **créer un nouvel objet** sans les variables *'auteur'* et *'type_document'*
 - **créer un nouvel objet** avec toutes les variables de *'editeur'* à *'nb_prets'*
 - **créer un nouvel objet** où *'titre'* est la première variable et *'type_document'* est la 5è

Transformer des données : `select()`

Solution

- **créer un nouvel objet** sans les variables *'auteur'* et *'type_document'*

```
med_select1 <- mediatheques %>% select(-auteur, -type_document)
```

```
med_select1 <- mediatheques %>% select(-c(auteur, type_document))
```

- **créer un nouvel objet** avec toutes les variables de *'editeur'* à *'nb_prets'*

```
med_select2 <- mediatheques %>% select(editeur:nb_prets)
```

- **créer un nouvel objet** où *'titre'* est la première variable et *'type_document'* est la 5^è

```
med_select3 <- mediatheques %>% select(titre, 1, 2, 4,  
type_document, 5:9, 11)
```

Transformer des données : `filter()`

Cette fonction sert à sélectionner des lignes dans une base de données, répondant à certains critères ou conditions logiques

- **Paramètres :**

- `data <- data %>% filter(col1 == "valeur")`
- opérations possibles : `==`, `!=`, `>`, `<`, `>=`, `<=`
- combinaisons d'opérations : `&` pour "et", `|` pour "ou"
- utiliser `nrow()` pour compter le nombre de ligne filtrées

- **Exemples :**

- `exemple <- exemple %>% filter(col1 != 10 & col3 == "red")`
- `exemple %>% filter(col > 0 | col1 <= 100) %>% nrow()`

- **À vous !**

- **dans un nouvel objet**, filtrer les acquisitions d'affiches
- **dans un nouvel objet**, filtrer les acquisitions publiées entre 1950 et 1960
- filtrer les musiques nouvelles ayant été prêtées au moins une fois et compter le nombre de lignes

Transformer des données : `filter()`

Solution

- dans un nouvel objet, filtrer les acquisitions d'affiches

```
med_filter1 <- mediatheques %>% filter(libelle_support == "Affiche")
```

- dans un nouvel objet, filtrer les acquisitions publiées entre 1950 et 1960

```
med_filter2 <- mediatheques %>% filter(annee_publication >= "1950" &  
                                     annee_publication <= "1960")
```

- filtrer les musiques nouvelles ayant été prêtées au moins une fois et
compter le nombre de lignes

```
data %>% filter(type_document == "Musiques nouvelles" & nb_prets >= 1)  
%>% nrow()
```

Transformer des données : `mutate()`

Cette fonction permet de créer de nouvelles variables à partir des variables existantes, elle est très utile pour l'analyse de données

- **Paramètres :**

- `data <- data %>% mutate(new_col = col1 * col2)`

- **Exemples :**

- `exemple <- exemple %>% mutate(new_col = col1 / col2 * 100)`

- `exemple <- exemple %>% mutate(new_col2 = col3 - 52)`

- **À vous !**

- créer une **nouvelle variable** du nombre de prêts + 10

- créer une **nouvelle variable** de l'année extraite de la date de réception

Transformer des données : `mutate()`

Solution

- créer une **nouvelle variable** du nombre de prêts + 10

```
mediatheques <- mediatheques %>% mutate(nb_prets10 =  
nb_prets + 10)
```

- créer une **nouvelle variable** de l'année extraite de la date de réception

```
mediatheques <- mediatheques %>% mutate(annee_reception =  
format(as.Date(date_reception, format="%Y-%m-%d"), "%Y"))
```

ou alors

```
mediatheques <- mediatheques %>% mutate(annee_reception =  
substr(date_reception, 1, 4))
```

Transformer des données : `summarise()`

Cette fonction permet de résumer plusieurs valeurs en une seule, ce qui est très utile pour avoir quelques statistiques sur les bases de données

- **Paramètres :**

- `data %>% summarise(moyenne = mean(col1))`
- statistiques possibles : `mean`, `median`, `min`, `max`, `range`, `sum`, `n`
- si au moins une valeur manque (`NA`), ajouter l'argument `na.rm = TRUE`

- **Exemple :**

- `exemple %>% summarise(min = min(col1, na.rm = TRUE),
max = max(col1, na.rm = TRUE))`

- **À vous !**

- calculer le nombre de prêts moyen et médian
- calculer l'étendue du nombre de prêts

Transformer des données : summarise()

Solution

- calculer le nombre de prêts moyen et médian

```
mediatheques %>% summarise(moy = mean(nb_prets),  
                             med = median(nb_prets))
```

- calculer l'étendue du nombre de prêts

```
mediatheques %>% summarise(etendue = range(nb_prets))
```

Transformer des données : `group_by()`

Cette fonction permet de grouper les variables, elle est spécialement utile en analyse de données pour calculer des statistiques par groupe

- **Paramètres :**

- `data %>% group_by(group) %>% summarise(min = min(col))`
- combiner avec les opérations déjà présentées : `mean`, `median`, `min`, `max`, `range`, `sum`, `n`
- penser à dégrouper les données pour éviter tout conflit avec les créations de variables `mutate`, via `ungroup()`

- **Exemple :**

- `exemple %>% group_by(group) %>% mutate(n = n()) %>% ungroup()`

- **À vous !**

- calculer le nombre d'acquisitions par support
- calculer le nombre de prêts par année de publication de l'acquisition
- calculer **dans une nouvelle variable de la base**, le nombre d'acquisitions par titre

Transformer des données : group_by()

Solution

- calculer le nombre d'acquisitions par support

```
mediatheques %>% group_by(libelle_support) %>% summarise(n =  
n())
```

- calculer le nombre de prêts par année de publication de l'acquisition

```
mediatheques %>% group_by(annee_publication) %>%  
summarise(nb_prets_total = sum(nb_prets))
```

- calculer **dans une nouvelle variable de la base**, le nombre d'acquisitions par titre

```
mediatheques <- mediatheques %>% group_by(titre) %>%  
mutate(nb_acquis = n()) %>% ungroup()
```

Des questions ?

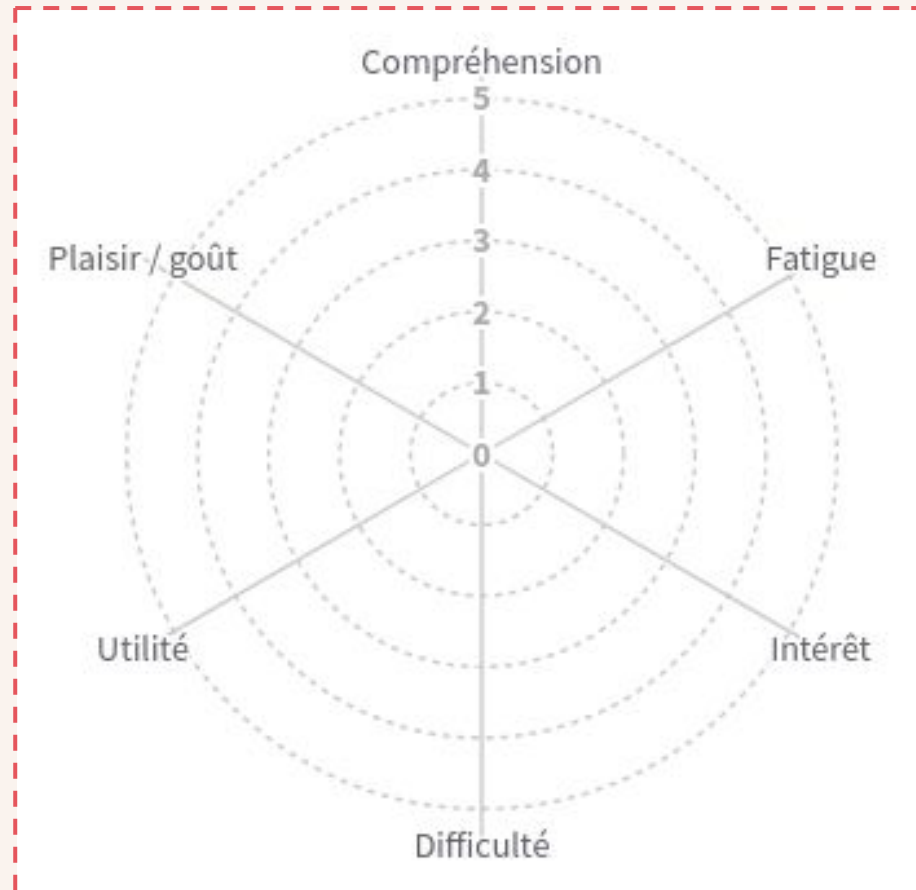
Conclusion



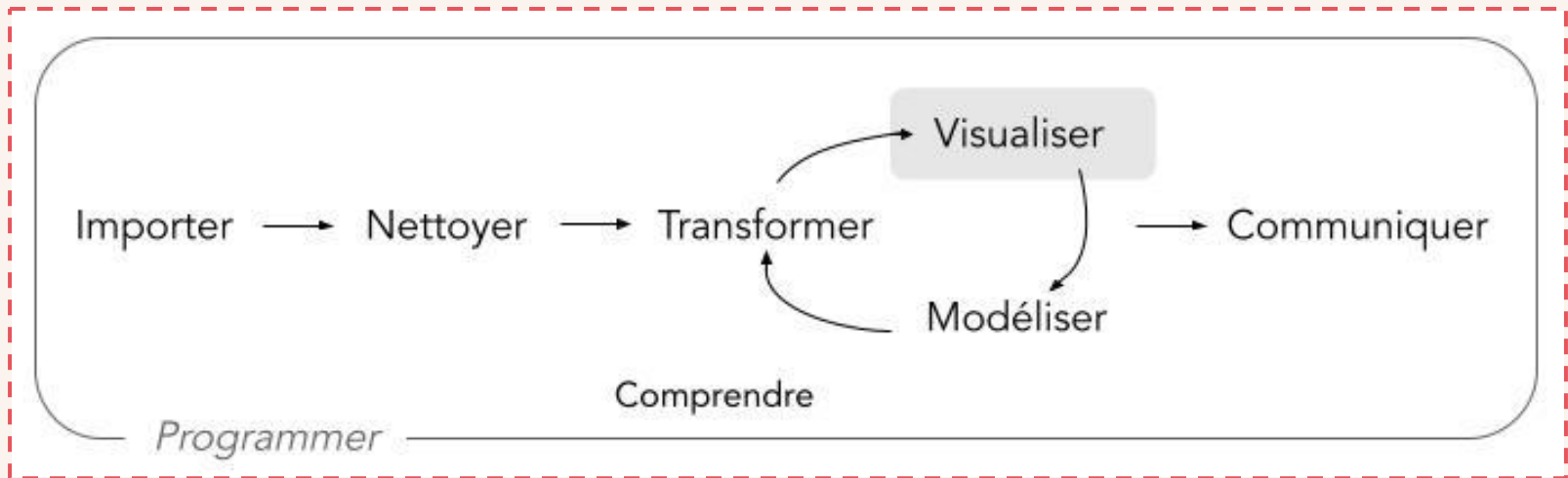
Résumons cette première demi-journée

- **Théorie** : vous connaissez désormais les rouages d'une bonne datavisualisation ;
- **Pratique** :
 - vous comprenez à quoi sert la programmation et pourquoi utiliser R via l'interface RStudio
 - vous avez fait vos premiers pas en code R :
 - import de données
 - nettoyage de données
 - transformation de données (`rename()`, `select()`, `filter()`, `mutate()`, `summarise()`, `group_by()`)

- Vous pouvez faire le bilan de cette journée en évaluant chaque pôle de 0 à 5, où 0 correspond à “faible/bas” et 5 “élevé/fort”



- Au programme de demain :



À demain !