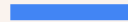




DATAACTIVIST



Comment réaliser des datavisualisations avec R ?

Première approche - jour 2

PROGRAMME JOUR 2

1. **Introduction et discussion autour des IA**
2. **Première partie : dérouillage**, *réviser les fonctions présentées et assimilées lors de la première demi-journée, permettant la manipulation de données ;*
3. **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 ;*
4. **Troisième partie : exercice final**, *reproduire une datavisualisation impliquant le traitement et la visualisation de données ;*
5. **Conclusion : vos retours, le mot de fin.**

Question sociétale

Connaissez-vous ChatGPT-3 ?

Capacité à être critique

- garder le contrôle sur le code, pouvoir vérifier le code
- Chat GPT peut halluciner du code, ne vérifie pas la sémantique de ta demande.
- Avoir le recul nécessaire pour évaluer le code sorti ==> insertion de code caché, effets de bord sur les bases de données...
- Un outil pour accélérer mais il faut quand même savoir conduire

Capacité d'introspection

Pour ChatGPT nous ne sommes pas capable aujourd'hui de certifier les propriétés de comportement du modèle. La question est : **Le chat a-t'il une morale ?**

Car finalement nous ne connaissons pas les intentions de l'IA ou plutôt de ceux qui l'ont créée

Cela rejoint le sujet de la transparence algorithmique

Attention à l'aide à la décision !

Dérouillage du matin

Exercice de mise en pratique

Vous avez 15 minutes pour effectuer ces quelques manipulations sur un jeu de données

- Depuis la base de données “**Carrefour à feu avec priorité bus**” disponible sur le portail open data de la MEL :
 - quelles sont les dimensions du jeu de données ?
 - combien y a-t-il de carrefours en poste de commandement ?

Exercice de mise en pratique

Solution

- Depuis la base de données “**Carrefour à feu avec priorité bus**” disponible sur le portail open data de la MEL :

- quelles sont les dimensions du jeu de données ?

```
carrefour_bus <-  
read_delim("https://opendata.lillemetropole.fr/api/explore/v2.1/catalog/datasets/carrefour-feu-avec-priorite-bus/exports/csv?lang=fr&timezone=Euro  
pe%2FBerlin&use_labels=true&delimiter=%3B", ";")  
dim(carrefour_bus)
```

- combien y a-t-il de carrefours en poste de commandement ?

```
table(carrefour_bus$Gestion)
```

OU

```
carrefour_bus %>% filter(Gestion == "en_PC") %>% nrow()
```

Rappel de la journée d'hier

https://docs.google.com/presentation/d/1e1vUAYpmQHvn45uLa-Xr9S_T-jbTsjsIOPEqaWd97rU/edit#slide=id.g1dc91d7b2a60_0

Exemple de dataviz

https://archivistes75.github.io/registre_entrees/Dashboard_viz/index.html

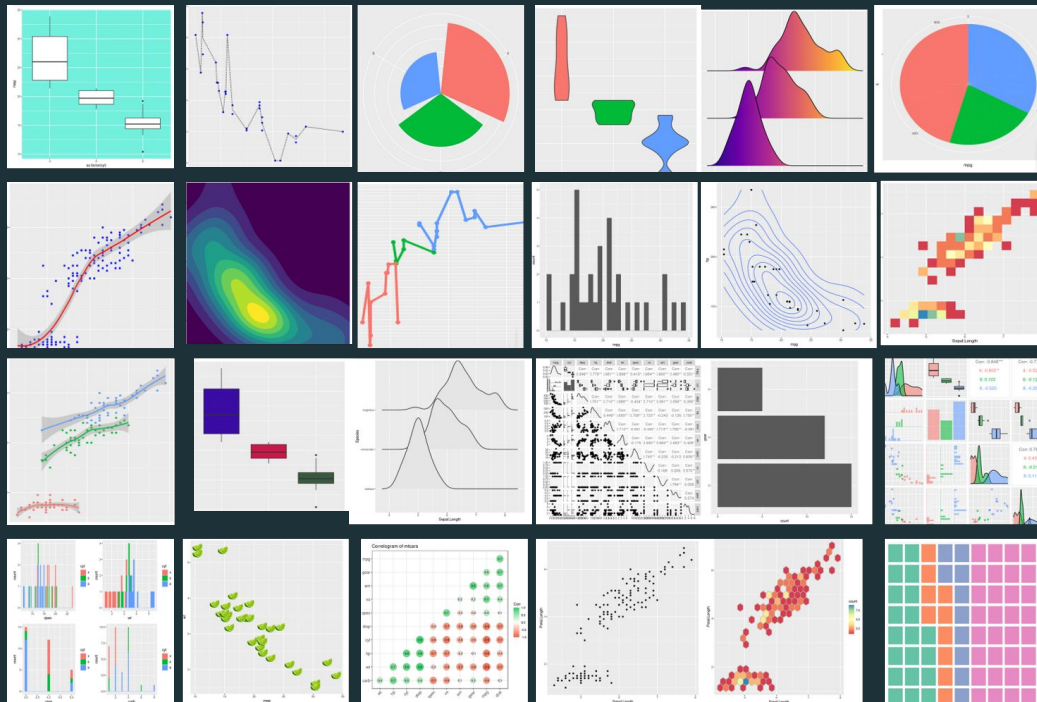
Exercice de mise en pratique (exercice en plus)

Solution

- créer un **nouvel objet** avec les carrefours de Lambersart ayant reçu plus de 50 appels un dimanche, et sommer le nombre d'appels au nombre d'actions.

```
subdf <- carrefour_bus %>% filter(Commune == "LAMBERSART",  
                                `Nombre appels` > 50,  
                                `Jour de la semaine` == "DIMANCHE") %>%  
mutate(somme_appels_actions = `Nombre appels` + `Nombre d'actions`)
```

Visualisation de données sous R

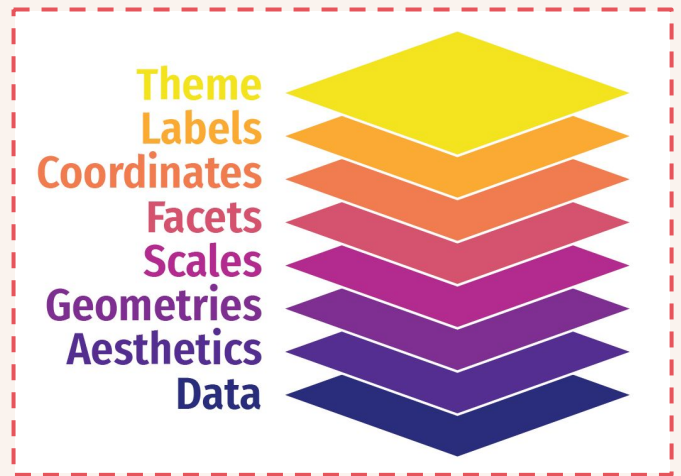


ggplot2, LA librairie de datavisualisation



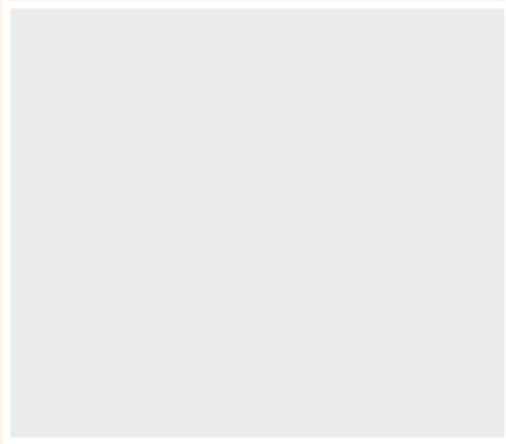
La librairie **ggplot2** contenue dans le tidyverse permet de faire des graphiques sous R. Initialement développée par Hadley Wickham, cette librairie est l'une des plus connues du tidyverse.

- **Sémantique des graphiques **ggplot2** :**
 1. **data** : les données à visualiser
 2. **aes** : les propriétés esthétiques
 3. **geom** : la géométrie/type de graphique
 4. **scale** : la gamme des valeurs
 5. **coord** : les coordonnées pour organiser la disposition
 6. **labels** : les étiquettes (noms)
 7. **facet** : les facettes pour sous-graphes
 8. **theme** : le thème appliqué aux éléments

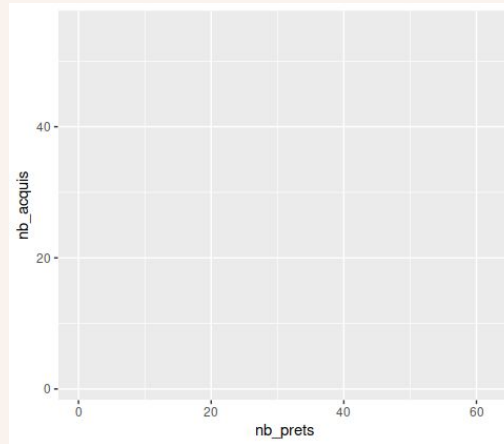


Le déroulé

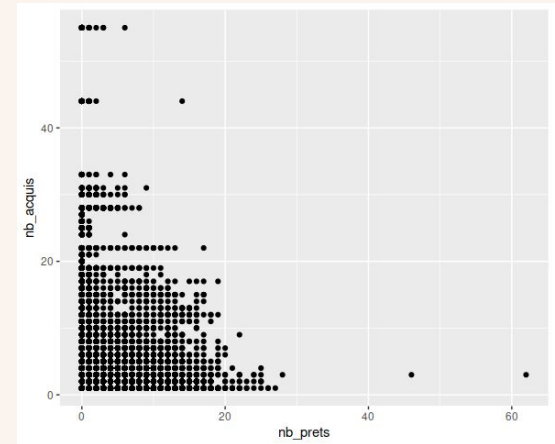
data : les données à visualiser



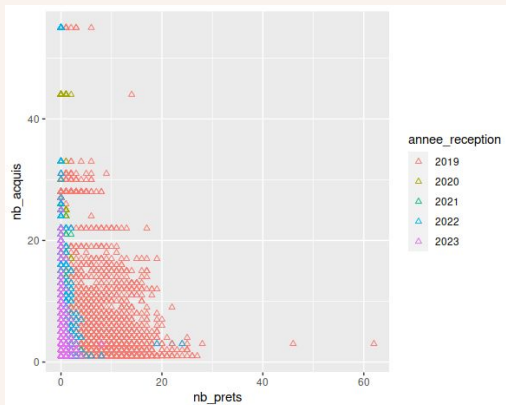
aes : les propriétés esthétiques



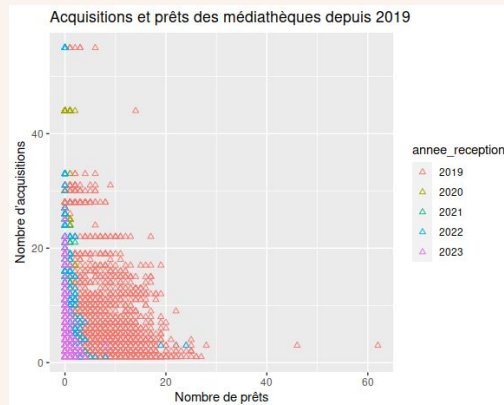
geom : la géométrie/type de graphique



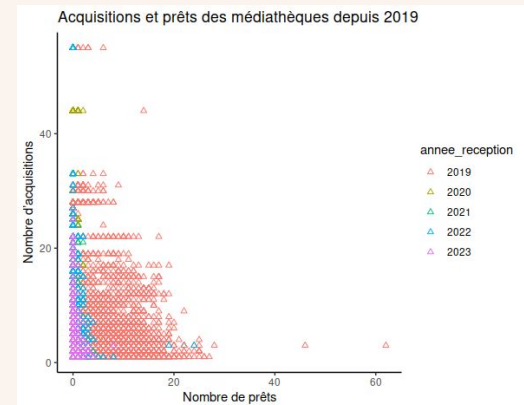
col, **shape** : les couleurs et les formes



labs : les noms et titres



theme : l'apparence



C'est parti !

Structuration de ggplot2 : data

Cet argument sert à appeler une base dont les données seront visualisées

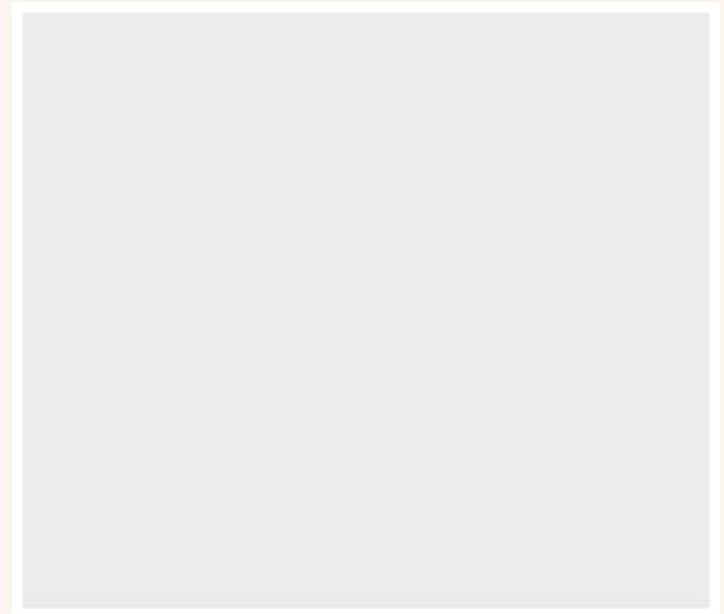
- **Paramètres :**
 - `ggplot(data = data)`
 - les prochains paramètres seront ajoutés au graphique par un `+` et non par le pipe `%>%`
- **Exemple :**
 - `ggplot(exemple)` ou `exemple %>% ggplot()`
- **À vous !**
 - exécuter le code d'hier d'import et de manipulations des données de médiathèques issues du portail open data de la MEL
 - commencer un graphique ggplot2 sur la base de données chargée dans votre session R

Structuration de ggplot2 : data

Solution

- exécuter le code d'hier d'import et de manipulations des données de médiathèques issues du portail open data de la MEL
- commencer un graphique ggplot sur la base de données chargée dans votre session R

```
ggplot(mediatheques)
```



Structuration de ggplot2 : aes()

Cet argument sert à indiquer les éléments et paramètres visuels qui dépendent de variables (on parle de *mapping*)

- **Paramètres :**

- `ggplot(data = data,
 mapping = aes(x = x, y = y))`
- **x** : abscisses, axe horizontal
- **y** : ordonnées, axe vertical

- **Exemple :**

- `ggplot(exemple, aes(x = col1, y = col2))`

- **À vous !**

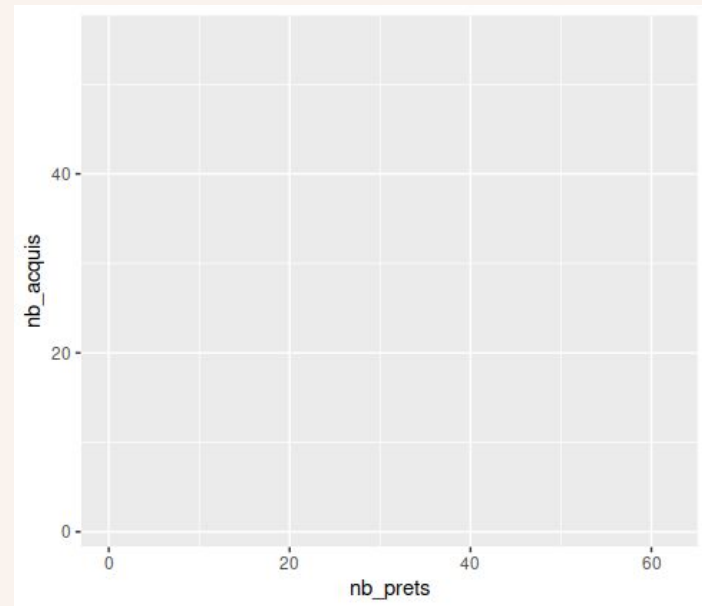
- continuer le graphique avec le nombre de prêts en abscisses et le nombre d'acquisitions (variable créée via `mutate()`) en ordonnées

Structuration de ggplot2 : aes()

Solution

- continuer le graphique avec le nombre de prêts en abscisses et le nombre d'acquisitions (variable créée via `mutate()`) en ordonnées

```
ggplot(mediatheques,  
      aes(x = nb_prets,  
          y = nb_acquis))
```



Structuration de ggplot2 : `geom_*()`

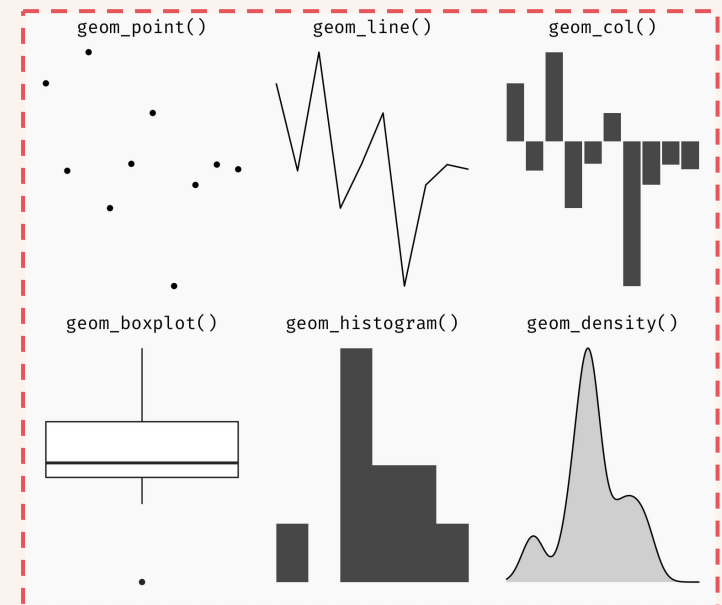
Cet argument sert à indiquer la géométrie utilisée, c'est-à-dire le type de datavisualisation

- **Paramètres :** `geom_*()`

- `geom_point()` : nuage de points
- `geom_line()` : ligne
- `geom_boxplot()` : boîte à moustache
- `geom_violin()` : violon
- `geom_density()` : densité
- `geom_histogram()` : histogramme
- `geom_bar()` : barres

- **Exemple :**

- `ggplot(exemple,`
 `aes(x = col1, y = col2)) +`
 `geom_histogram()`



- **À vous !**

- continuer le graphique en ajoutant la géométrie nuage de points

Une différence subtile entre `geom_bar()` et `geom_col()`

`geom_col()` est un cas particulier de `geom_bar()` où `stat = "identity"`

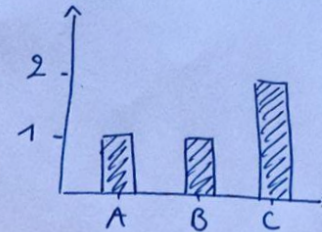
- Une illustration sur des données simples

X	Y
A	2
B	3
C	1
C	5

- Quelques explications :

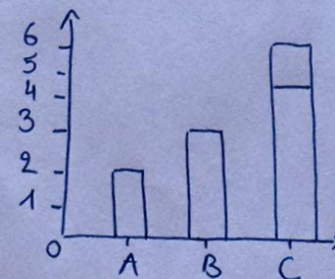
- par défaut `geom_bar()` visualise le **nombre de lignes** par catégorie de X
- en précisant `stat = "identity"` ou en utilisant `geom_col()`, ce sont **les valeurs d'une variable Y** qui sont représentées par la hauteur des barres

• `ggplot() + aes(x = X) +
geom_bar()`



• `ggplot() + aes(x = X, y = Y) +
geom_bar(stat = "identity")`

ou `ggplot() + aes(x = X, y = Y) +
geom_col()`



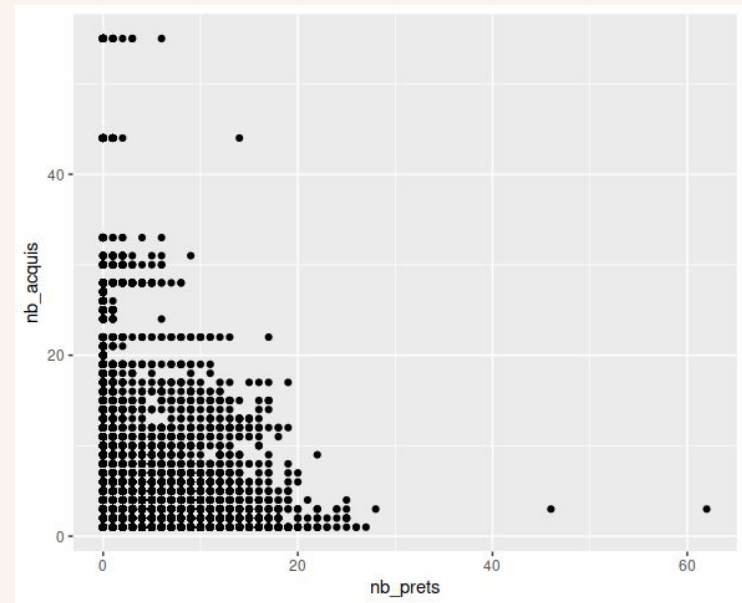
← va mettre toutes
les valeurs donc ici
1 et 5. Il faut passer
par une nouvelle variable
dans ce cas. `mutate()`

Structuration de ggplot2 : `geom_*()`

Solution

- continuer le graphique en ajoutant la géométrie nuage de points

```
ggplot(mediatheques,  
      aes(x = nb_prets,  
          y = nb_acquis)) +  
geom_point()
```



Structuration de ggplot2 : data, aes, geom

L'argument `aes()` peut être spécifié à différents endroits du code ; l'écriture est différente mais le résultat est identique

- **Différentes écritures :**

- `ggplot(exemple, aes(x = col1, y = col2)) +
 geom_histogram()`
- `ggplot(exemple) +
 aes(x = col1, y = col2) +
 geom_histogram()`
- `ggplot(exemple) +
 geom_histogram(aes(x = col1, y = col2))`

Structuration de ggplot2 : data, aes, geom

Avec ces 3 couches essentielles, on peut “jouer” avec les paramètres pour améliorer les graphiques

- **Paramètres modifiables :**

- color, fill, alpha, size, shape, width...
- valeurs fixes ou dépendant de variables de la base de données

- **Exemple :**

- `ggplot(exemple,
aes(x = col1, y = col2, size = col2),
col = "black", fill = "red") +
geom_histogram()`

- **À vous !**

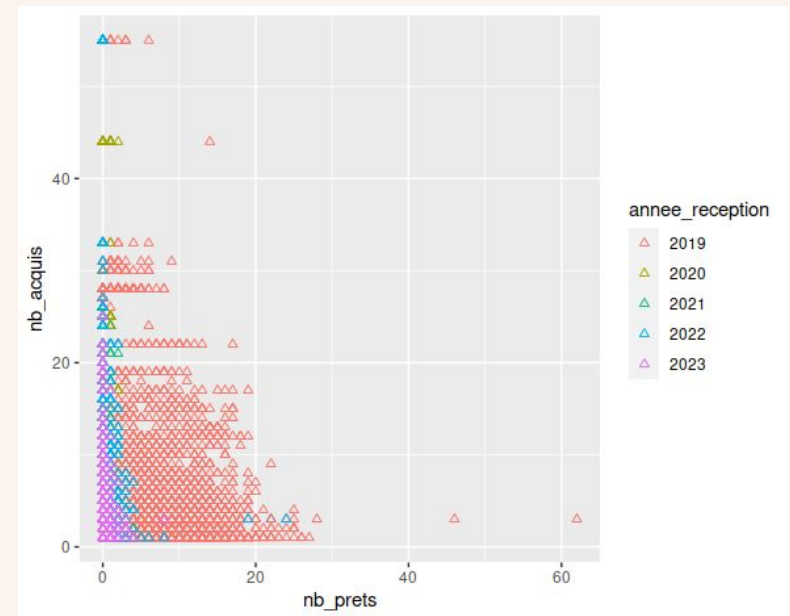
- continuer le graphique en colorant les points selon l'année de réception, avec une forme de points (shape) égale à 2

Structuration de ggplot2 : data, aes, geom

Solution

- continuer le graphique en colorant les points selon l'année de réception, avec une forme de points (**shape**) égale à 2

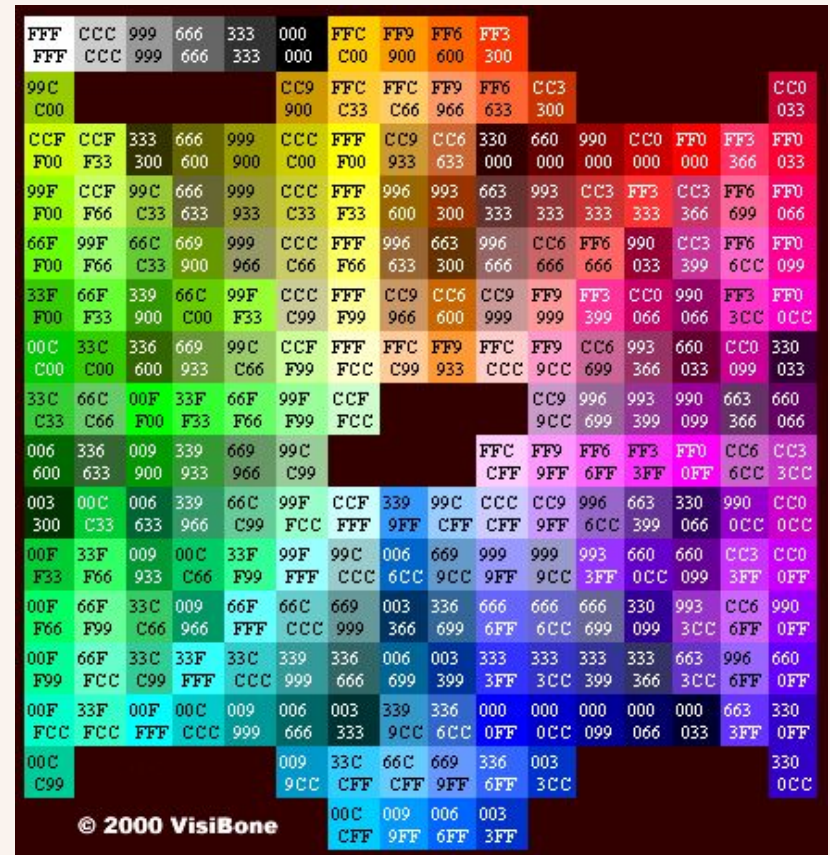
```
ggplot(mediatheques,  
  aes(x = nb_prets,  
    y = nb_acquis,  
    col = annee_reception)) +  
  geom_point(shape = 2)
```



Structuration de ggplot2 : colors

Les couleurs des graphiques sont importantes pour l'esthétique, la lisibilité, la compréhension, et le message à faire passer

- **2 manières de renseigner les couleurs :**
 - par son nom, ex: "red"
 - par son code HEX, ex: "#FFCC00"
- **Créer sa propre couleur :**
 - <https://www.color-hex.com/>



Structuration de ggplot2 : labs()

Cet argument sert à modifier les noms qui apparaissent sur le graphique

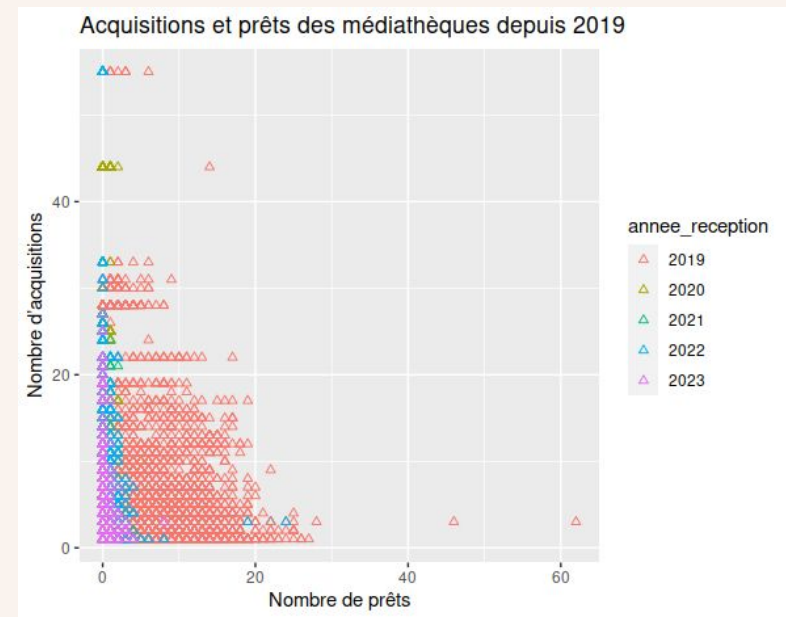
- **Paramètres :**
 - `labs(title, subtitle, caption, x, y)`
- **Exemple :**
 - ```
ggplot(exemple,
 aes(x = col1, y = col2)) +
 geom_histogram() +
 labs(title = "mon graphique", x = "Aa", y = "Bb")
```
- **À vous !**
  - continuer le graphique en nommant les axes X et Y, et en donnant un titre

# Structuration de ggplot2 : labs()

## Solution

- continuer le graphique en nommant les axes X et Y, et en donnant un titre

```
ggplot(mediatheques,
 aes(x = nb_prets,
 y = nb_acquis,
 col = annee_reception)) +
geom_point(shape = 2) +
labs(x = "Nombre de prêts",
 y = "Nombre d'acquisitions",
 title = "Acquisitions et prêts des médiathèques
depuis 2019")
```



# Structuration de ggplot2 : theme\_\*

Cet argument sert à modifier l'apparence du graphique (taille, polices, couleur des titres, arrière-plan, quadrillages etc.)

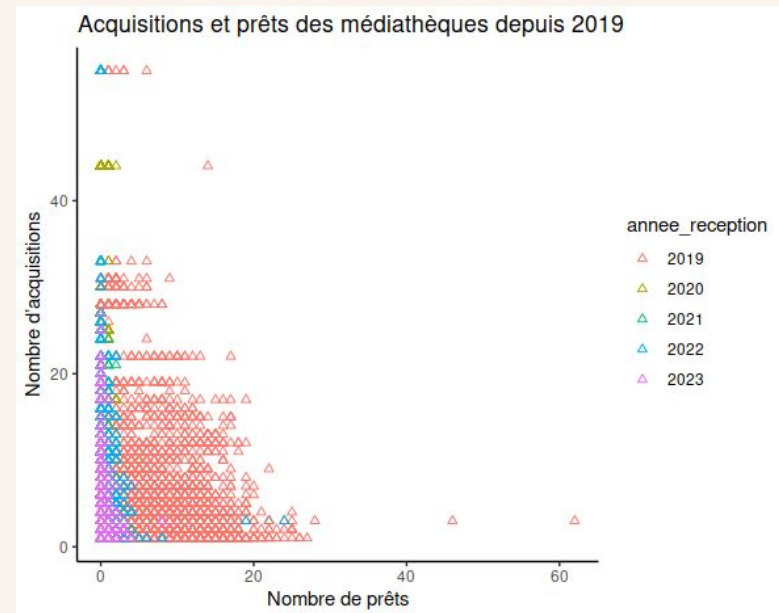
- **Paramètres :**
  - `theme_*`
  - tous les thèmes prédéfinis dans ggplot2, consultables [ici](#)
- **Exemple :**
  - ```
ggplot(exemple, aes(x = col1, y = col2)) +  
  geom_histogram() +  
  theme_bw()
```
- **À vous !**
 - continuer le graphique en ajoutant le thème 'classic'

Structuration de ggplot2 : theme_*()

Solution

- continuer le graphique en ajoutant le thème 'classic'

```
ggplot(mediatheques,  
      aes(x = nb_prets,  
          y = nb_acquis,  
          col = annee_reception)) +  
geom_point(shape = 2) +  
labs(x = "Nombre de prêts",  
     y = "Nombre d'acquisitions",  
     title = "Acquisitions et prêts des médiathèques  
depuis 2019") +  
theme_classic()
```



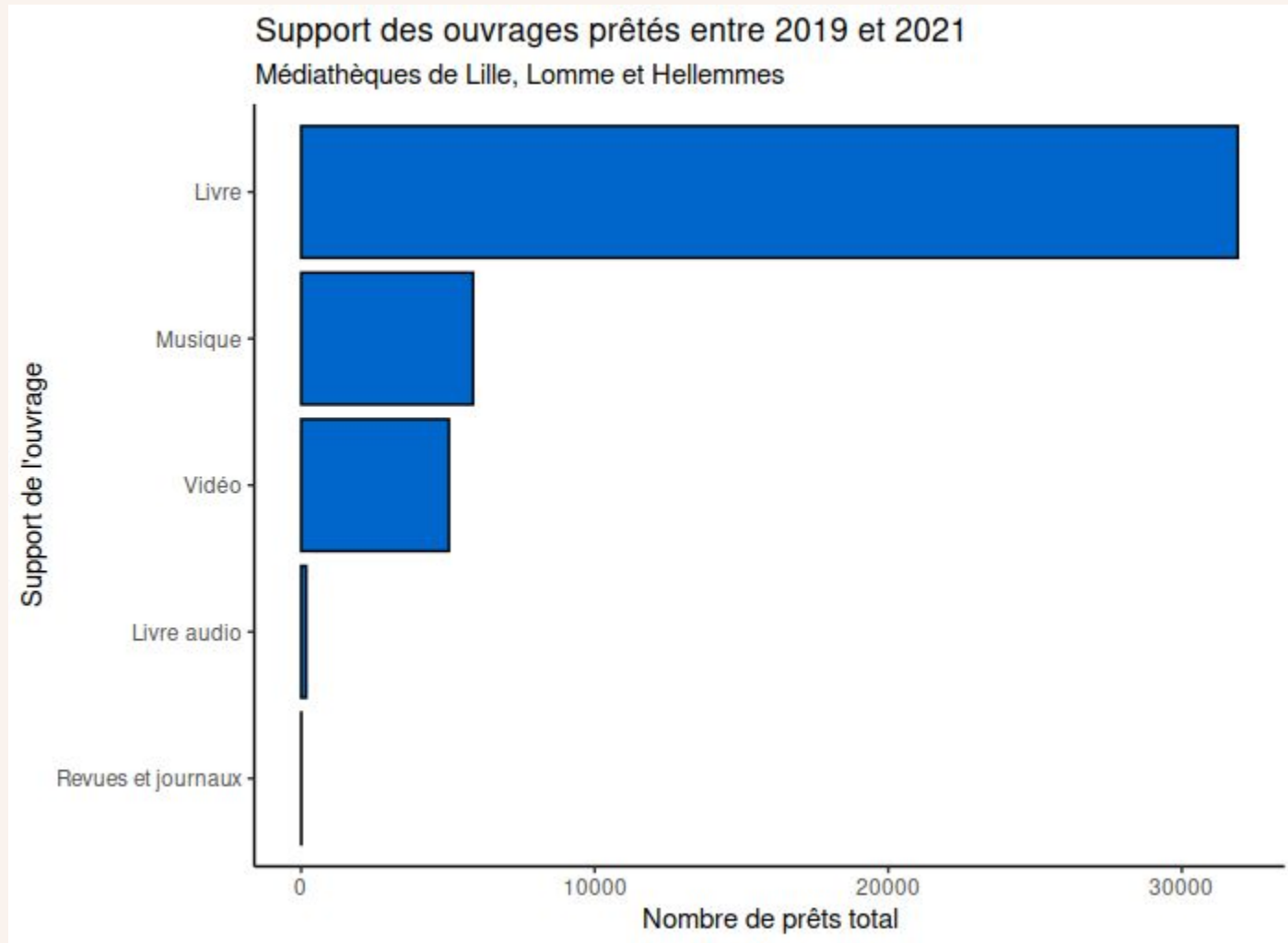
Des questions ?

Exercise final

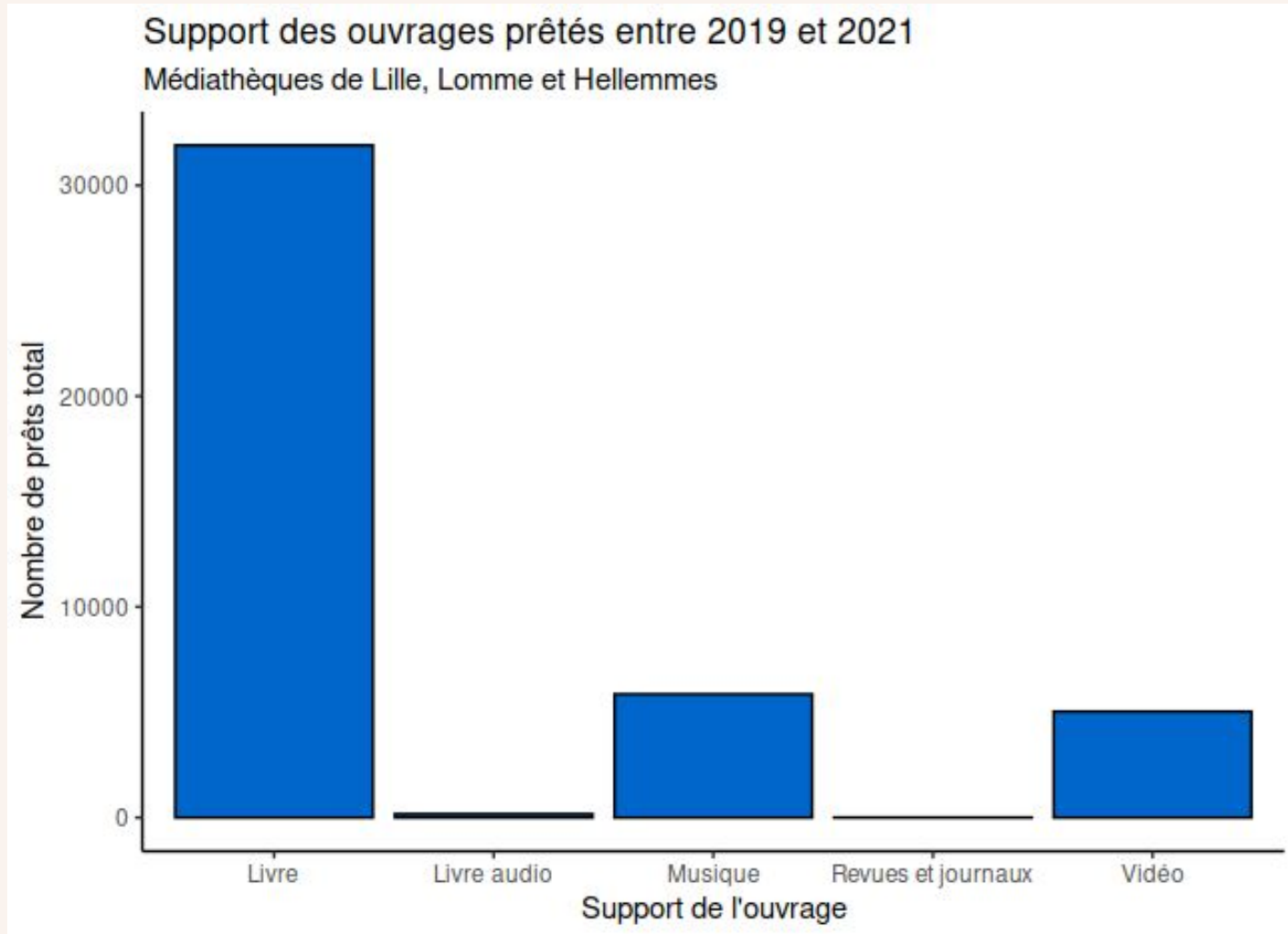


Datavisualisation à reproduire

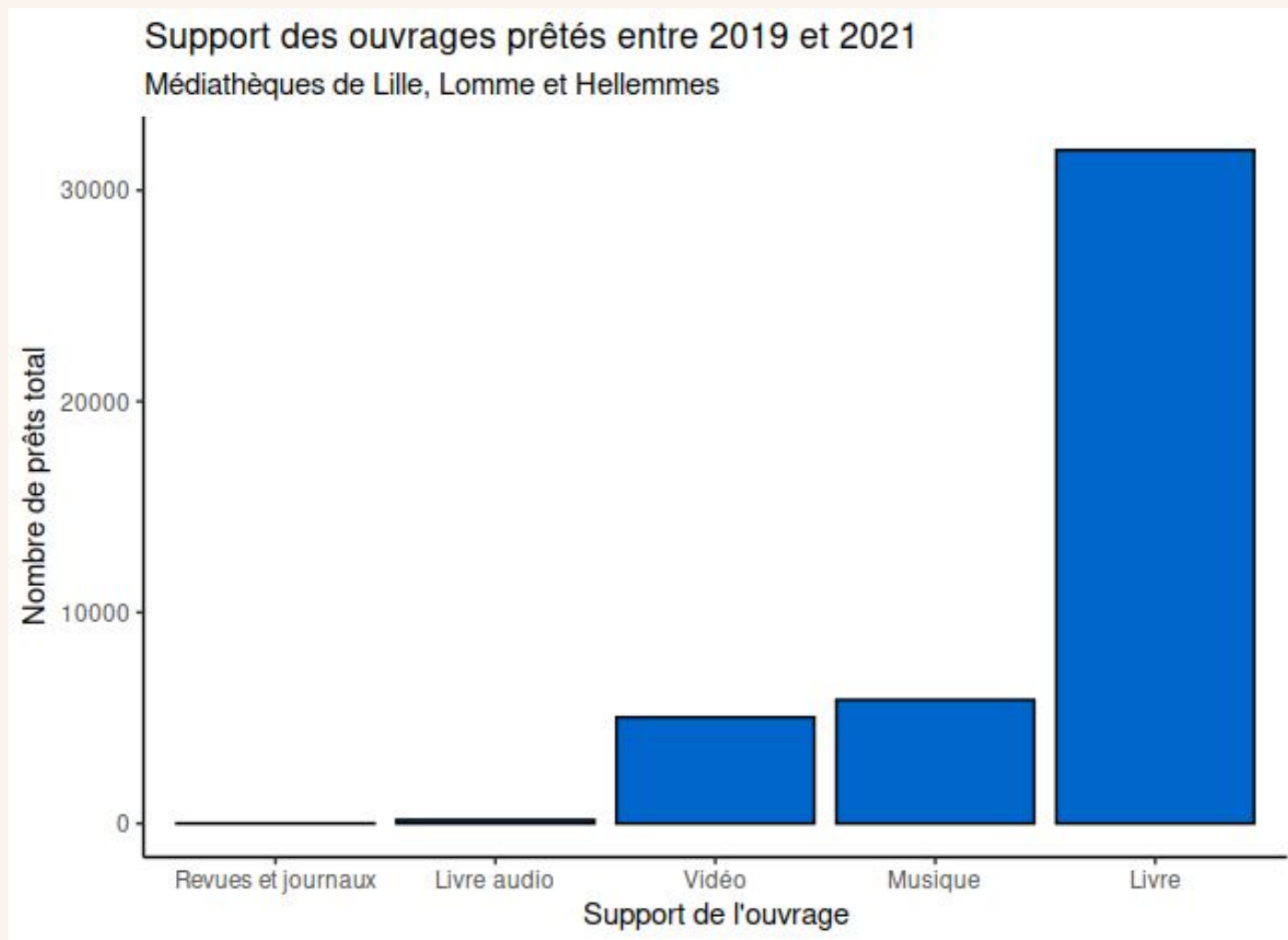
Vous avez une heure pour reproduire la datavisualisation suivante, en transformant et visualisant les données des médiathèques



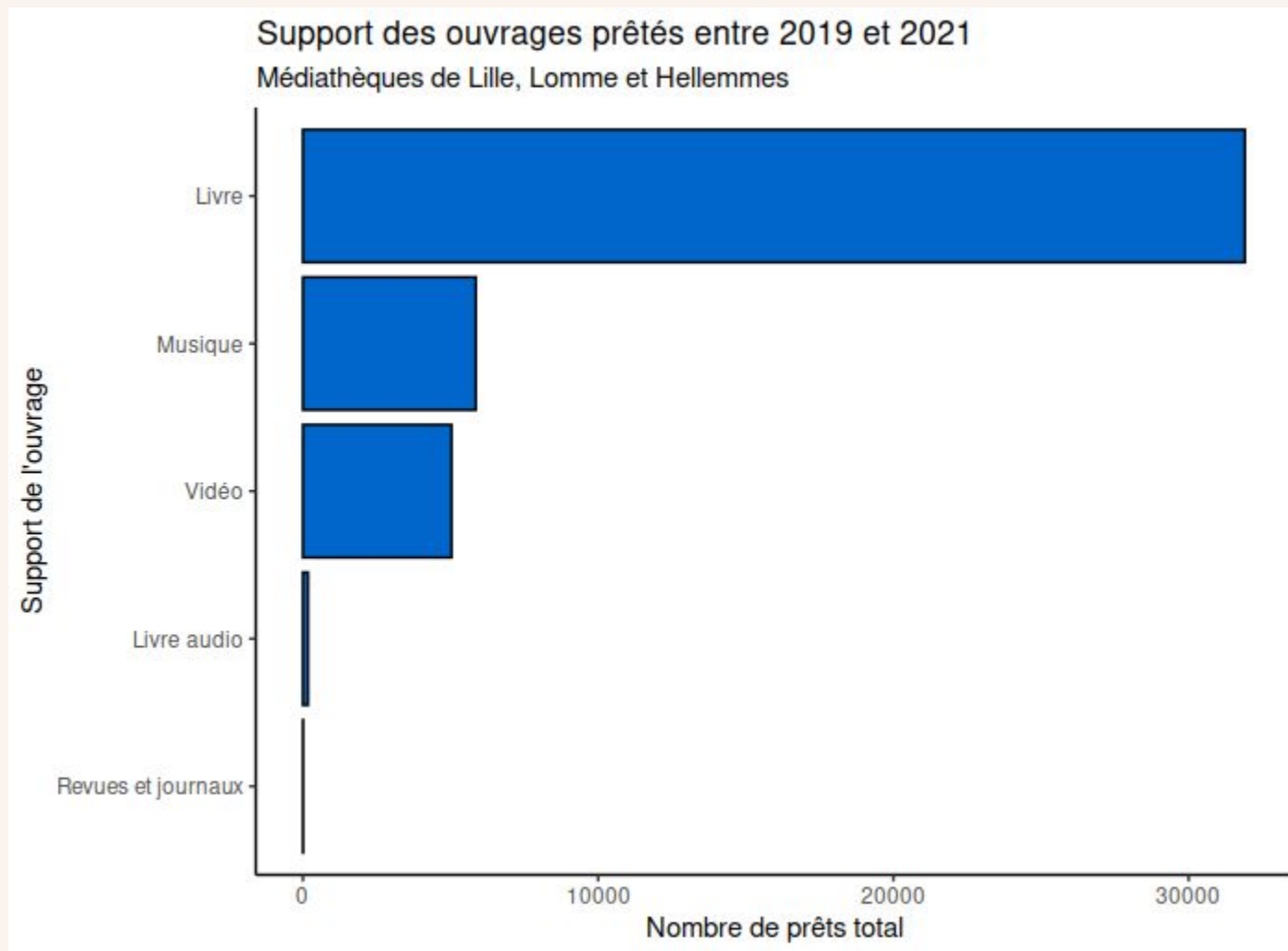
- Avec les fonctions et paramètres présentés tout au long de cet atelier



- Avec les barres triées par nombre de prêts



- Avec les barres triées et à l'horizontal



Datavisualisation à reproduire

Solution

Tableau

```
table_stat <- mediatheques %>% filter(annee_publication == "2019" | annee_publication == "2020" |  
annee_publication == "2021") %>%  
  group_by(libelle_support) %>%  
  summarise(nb_prets_total = sum(nb_prets)) %>%  
  filter(nb_prets_total != 0)
```

Graphique

```
table_stat %>% mutate(libelle_support = reorder(libelle_support, nb_prets_total, desc = T)) %>%  
  ggplot(aes(x = libelle_support, y = nb_prets_total)) +  
  geom_col(fill = "#0066CC", col = "black") +  
  labs(title = "Support des ouvrages prêtés entre 2019 et 2021",  
        subtitle = "Médiathèques de Lille, Lomme et Hellemmes",  
        x = "Support de l'ouvrage",  
        y = "Nombre de prêts") +  
  coord_flip() +  
  theme_classic()
```

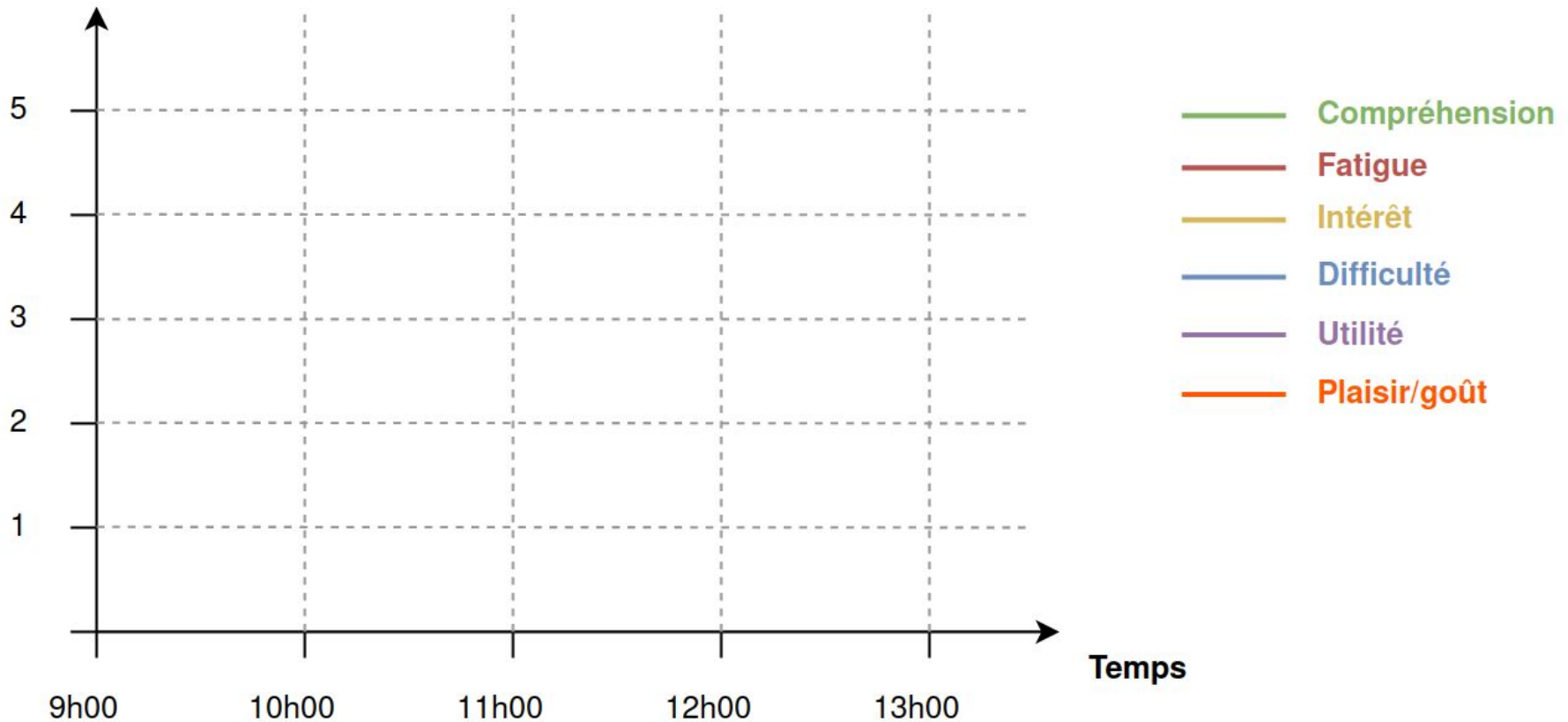
Conclusion

Résumons cette deuxième demi-journée

- **Mains dans le cambouis**, vous avez prouvé tôt le matin que vous saviez manipuler des données avec l'exercice de dérouillage ;
- **ggplot2**, vous avez appris à construire un graphique pas à pas sur R, avec la librairie de datavisualisation
 - les incontournables : `data`, `aes()`, `geom_*()`
 - les plus : `colors`, `labs()`, `theme_*()`...
- **Reproduire une dataviz**, vous avez transformé puis visualisé les données de médiathèques, dans le but de s'approcher le plus possible du modèle de dataviz proposé.

Et aujourd'hui ?

Note /5



Bibliographie

Des documentations pour approfondir :

- [Cheat sheets](#), “antisèches” rendant facile l’apprentissage et l’utilisation de packages R (*recharger les pages si les PDF ne s’affichent pas*) :
 - [import de données](#)
 - [nettoyage de données](#)
 - [transformation de données](#)
 - [visualisation de données avec ggplot2](#)
- [R for Data Science](#), livre incontournable pour apprendre la science de données sous R, écrit par *Hadley Wickham* et *Garrett Grolemund* ;
- [R Graph Gallery](#), aide et inspiration pour réaliser des graphiques sous R ;
- [What would you like to show](#), schéma récapitulatif de la visualisation à utiliser en fonction du type de données représenté.

MERCI !

diane@dataactivist.coop

magalie@dataactivist.coop