Cours sur la visualisation des données

Anne Imouza

12/3/2021

Objectifs du cours :

- Importance de visualiser les données avant de tirer des conclusions.
- Faciliter l'interprétation des données.
- Reconnaître les observations aberrantes.
- Connaître nos données.

Pourquoi utiliser `ggplot` ?

- Plus polyvalent.
- Incorporer dans le paquet tidyverse.
- Facile d'utilisation en suivant bien les règles.

Quel type de graphique pour quel type de variables ?

- Cela dépend en général du type de variables (qualitative ou quantitative) et du nombre de variables.
  - Types de graphiques avec une seule variable :

Type de variables	Une seule variable
Qualitative	Diagramme en bâton Diagramme circulaire Carte (map)
Quantitative	Histogramme (geom_hist) Boîte à moustaches

(Vissého Adjiwannou 2020, SICSS)

- Types de graphiques avec plusieurs variables :

*	*	Variable dépendante	Variable dépendante
*	Type de variables	Qualitative	Quantitative
Variable indpt	Qualitative	Diagramme en bâtons geom_bar	Boîte à moustaches geom_boxplot
Variable indpt	Quantitative	Transformer la variable en qualitative	Nuage de points geom_point

(Vissého Adjiwannou 2020, SICSS)

- Le choix des graphiques va aussi dépendre :
  - De l'audience (experts, public large, etc),
  - Ce que nous souhaitons raconter-présenter.

Quelques commandes utiles avant l'apprentissage de `ggplot 2`

- Voici plusieurs commandes que nous utiliserons avant de présenter les graphiques :

```
#Importer les librairies
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(haven) #importation de données sous forme dta

# importer la base de données
covid <- read_dta("Covid_2.dta")
```

- *%>%* : pipeline / pipe operator / tuyau. Il indique juste une succession d'opérations. ==> son raccourci ; Ctrl+Shift+M (Windows) ; Cmd+Shift+M (Mac)

```
covid1 <- covid %>%
  select(femme, educ, conf_manu, age) %>%
  group_by(femme) %>%
  summarise(age_moyen = mean(age),
            age_median = median(age))
```

- *select()* : sélectionner des variables.

```
covid1 <- covid %>%
  select(femme, educ, conf_manu)
```

- *filter()* : sélectionner des observations.

```
covid1 <- covid %>%
  filter(educ == 3)
```

- *mutate()* : recoder-transformer et créer de nouvelles variables.

```
covid1 <-
  covid %>%
  mutate(age_cat = case_when( #indique plusieurs conditions
    age < 20 ~ "adolescent",
    age >= 20 & age <= 34 ~ "jeune",
    age >= 35 & age <=59 ~ "adulte",
    age >=60 ~ "ainé"
  ))
```

- *class()* : connaître le type d'une variable.

```
class(covid$symptomes)
```

```
## [1] "numeric"
```

- *as.numeric / as.factor* : changer le type d'une variable. *as.factor* : La fonction factor permet de créer une variable **qualitative** ou **categorielle** ou **factorielle**, à partir d'une autre variable.
- *if_else* : recoder des variables avec des conditions.

```
covid1 <- covid %>%
  mutate(femme = if_else(femme == 0, "Homme", "Femme"))
```

- *group_by* : regrouper.

```
covid1 <-  covid %>%
  group_by(femme) %>%
  summarise(age_moyen = mean(age),
            age_median = median(age))
```

- En général pour visualiser des données, on doit nettoyer la base de données en fonction de ce que nous souhaitons raconter et présénter.
- Cela vaut autant pour les données numériques que textuelles (pre-processing)!

Utilisation de `ggplot2`

- Vous pouvez faire un point d'interrogation pour connaître les paramètres à utiliser avec la commande `ggplot()` :

```
?ggplot2
```

- *définition générale* : `ggplot()` permet de lier des données à une représentation graphique.

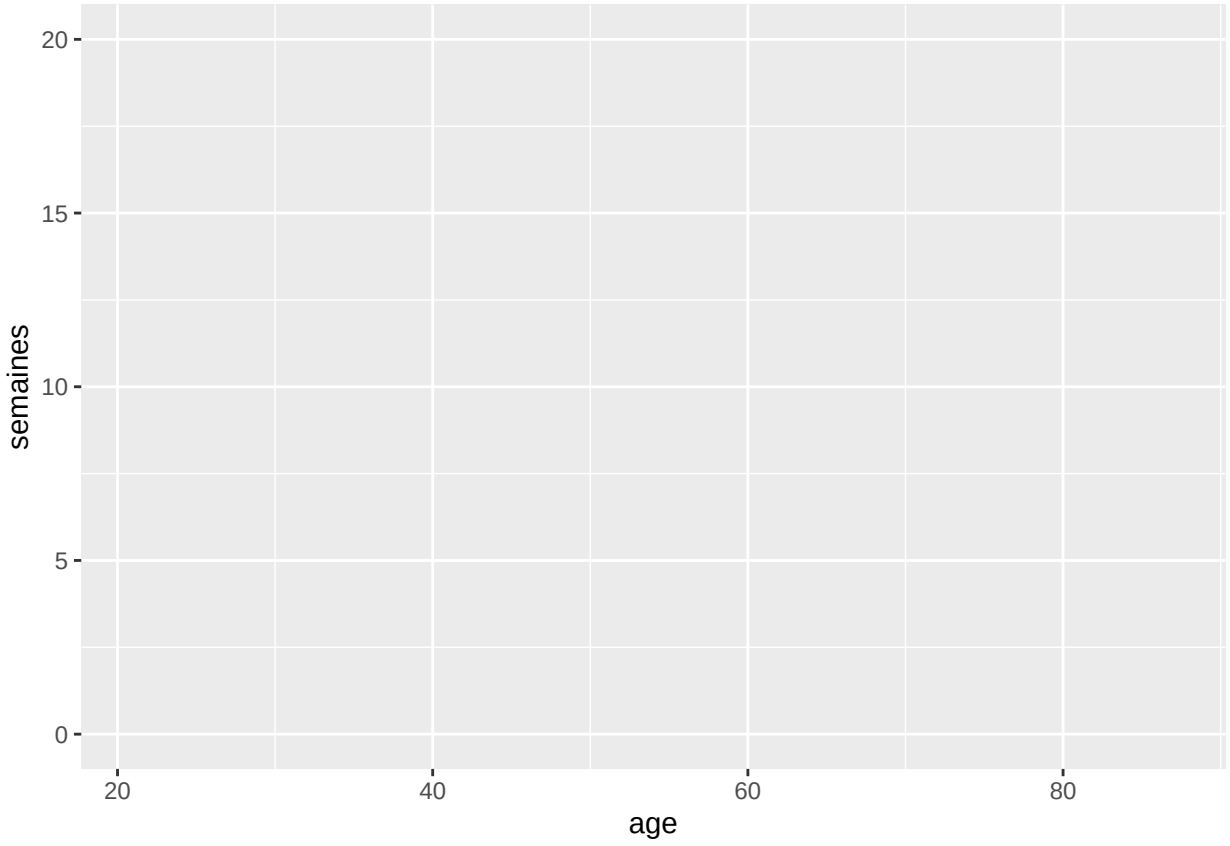
Présentation des fonctions de base avec la commande ggplot :

- Vous pouvez construire des graphiques à partir de mêmes composants :
- ÉTAPE 1 : un ensemble de données :

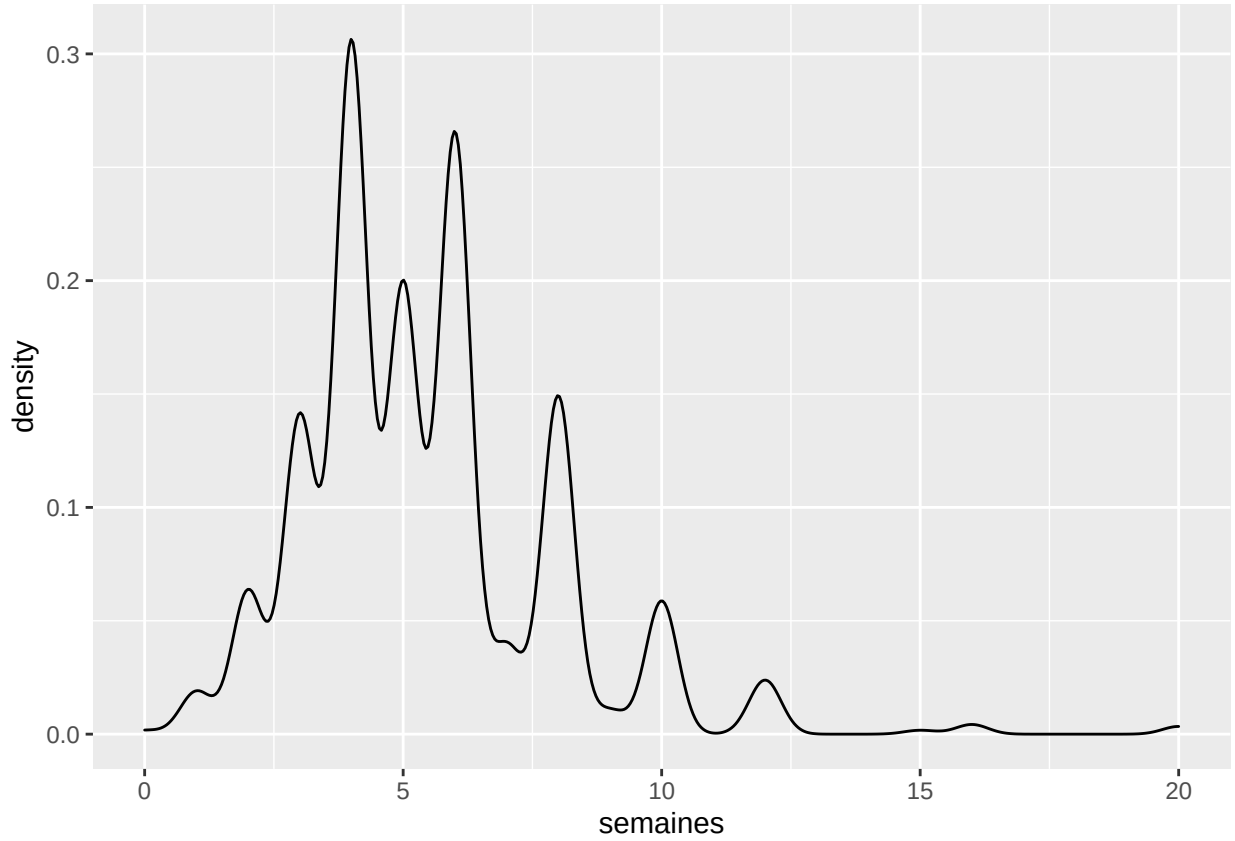
```
ggplot(data = covid)
```


- ÉTAPE 2 : un système de coordonnées (éléments visuels)

```
ggplot(data = covid,
       mapping = aes(x = age , y = semaines))
```



- Pour ajouter plus d'esthétisme et les valeurs, on ajoute des propriétés visuelles avec la commande *aes()* (aesthetic) pour :
  - la taille (*size*),
  - la position (*position*),
  - la transparence (*alpha*),
  - le remplissage (*fill*),
  - la forme (*shape*),
  - la couleur (*color*) et
  - les emplacements de vos variables dépendante et indépendante.
- ÉTAPE 3 : des geométries (types de graphique)



==========

Atelier pratique

Ouvrir un projet R :

- "File -> New project -> Existing Directory". Nommez votre nouveau projet et script.

Supprimer les fichiers dans votre environnement :

```
rm(list = ls())
```

Installer les paquets:

```
#install.packages("tidyverse")
```

Lire les paquets qui nous intéresse :

```
#library(haven) #importation de données sous forme dta
#library(tidyverse)
```

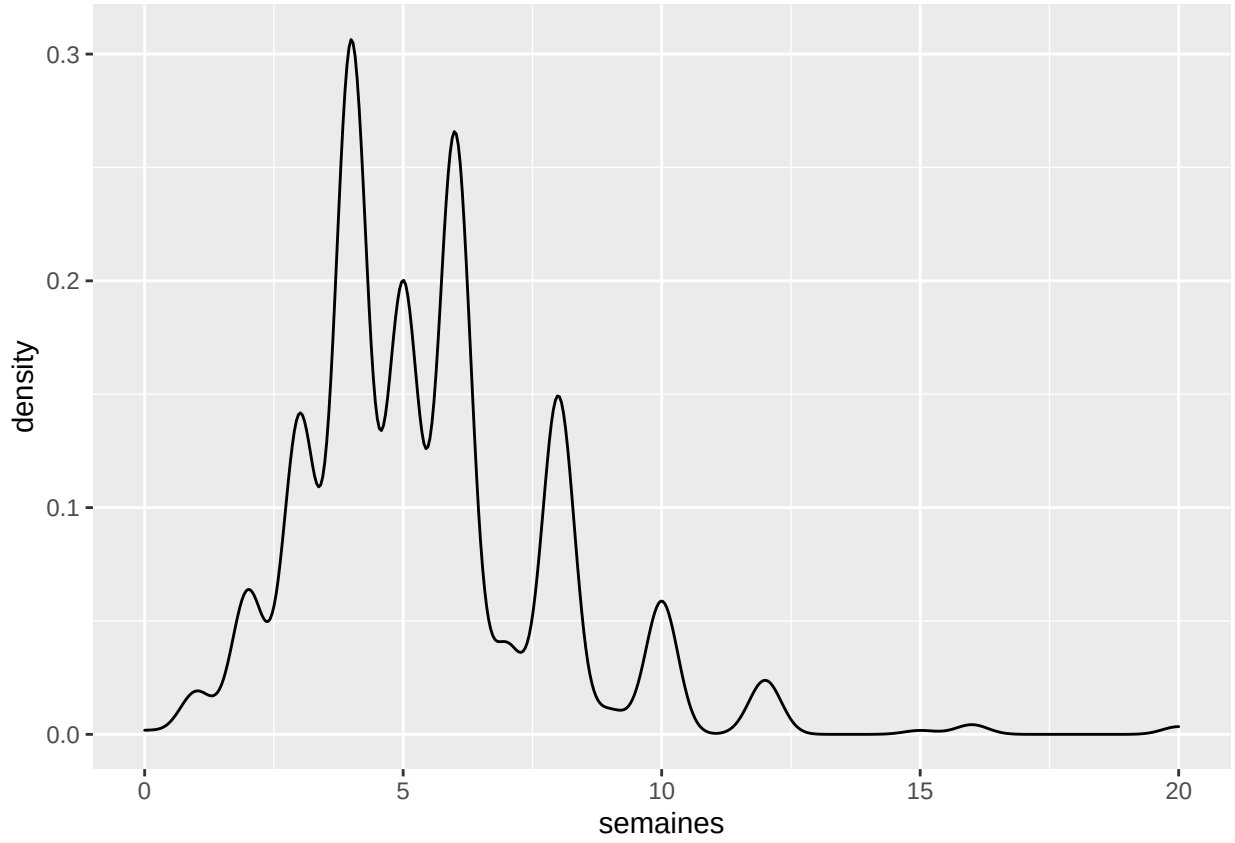
Importer nos bases de données :

```
covid <- read_dta("Covid_2.dta")
quality_governance <- read.csv("Quality_Governance.csv")
```

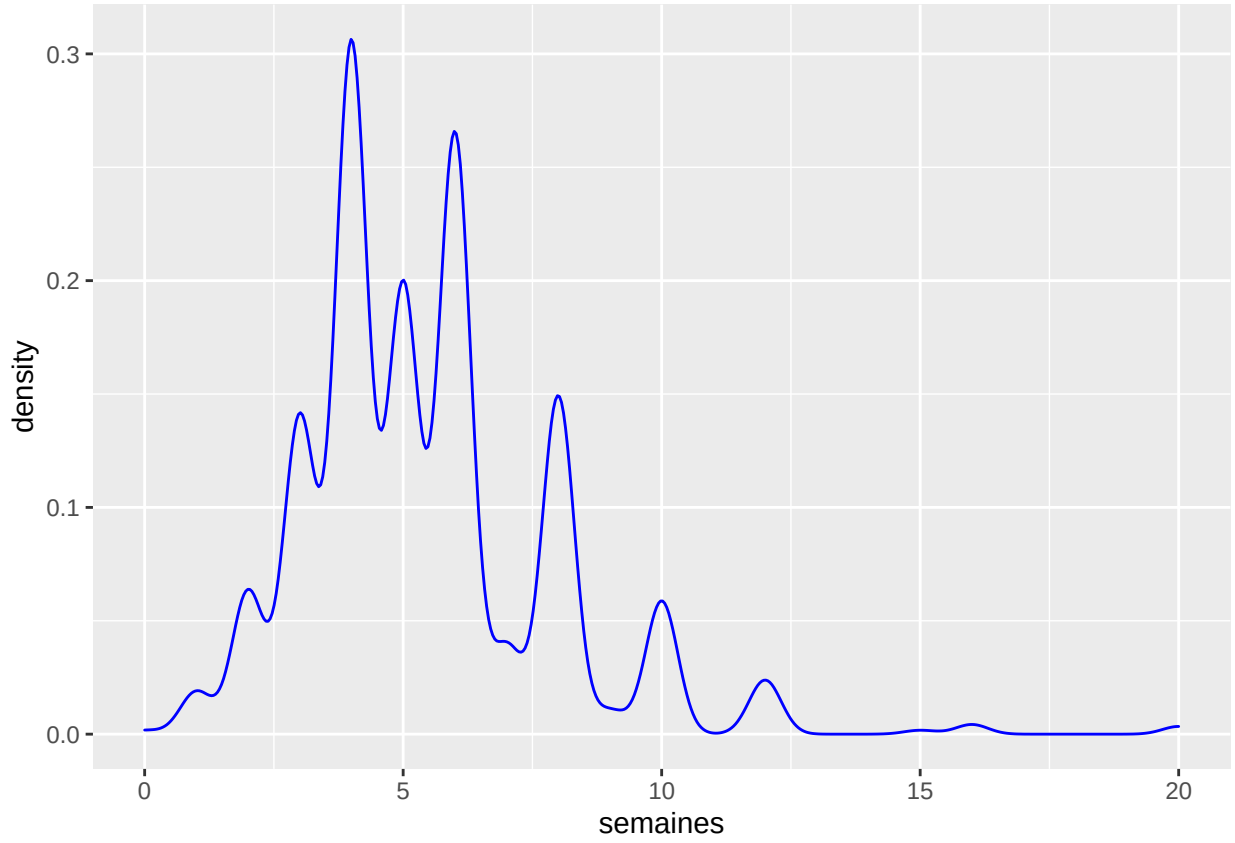
Faire un graphique univarié avec ggplot

Exemple utilisation ggplot avec la base de données *COVID* pour un graphique univarié :

```
a <- ggplot(data = covid, mapping = aes(x = semaines)) +
  geom_density()

a
```

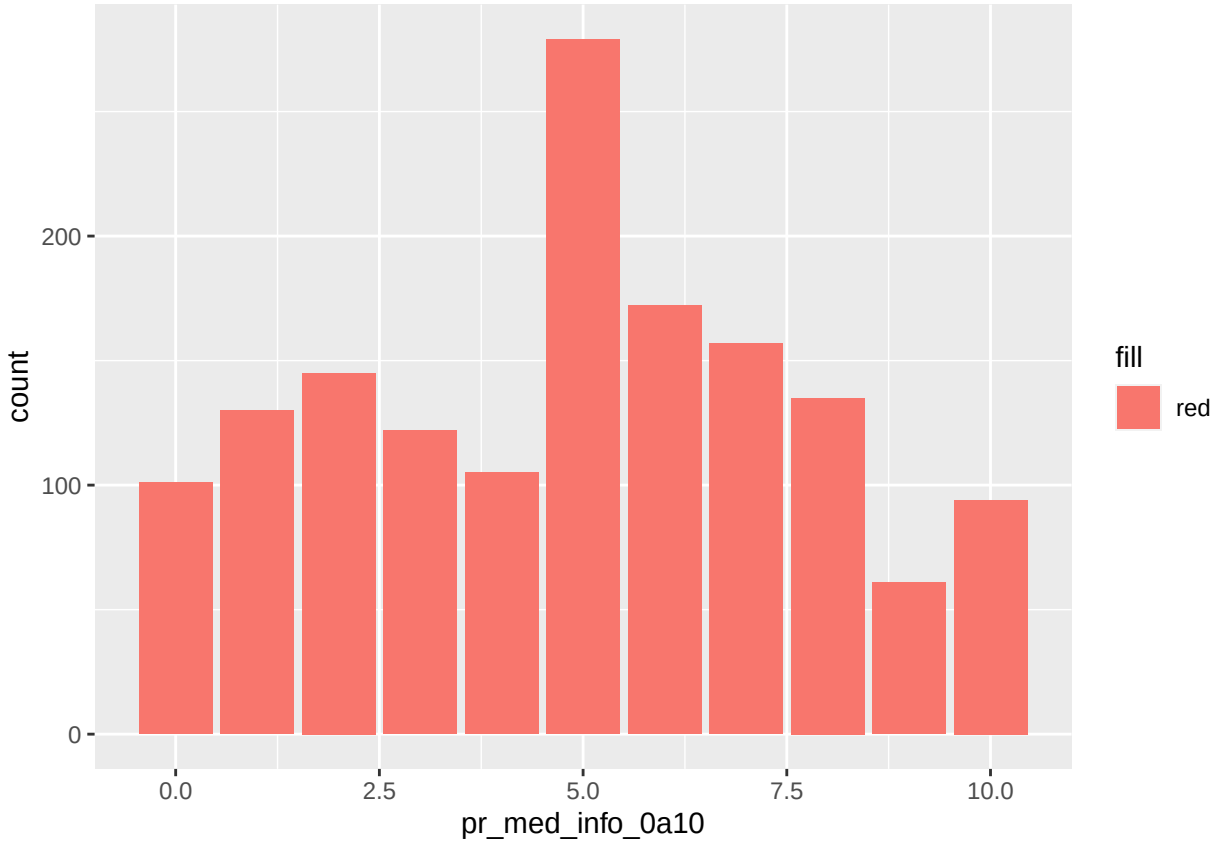



```
# Changer la couleur de la distribution  :
b <- ggplot(data = covid, mapping = aes(x = semaines)) +
  geom_density(color = "blue")
b
```



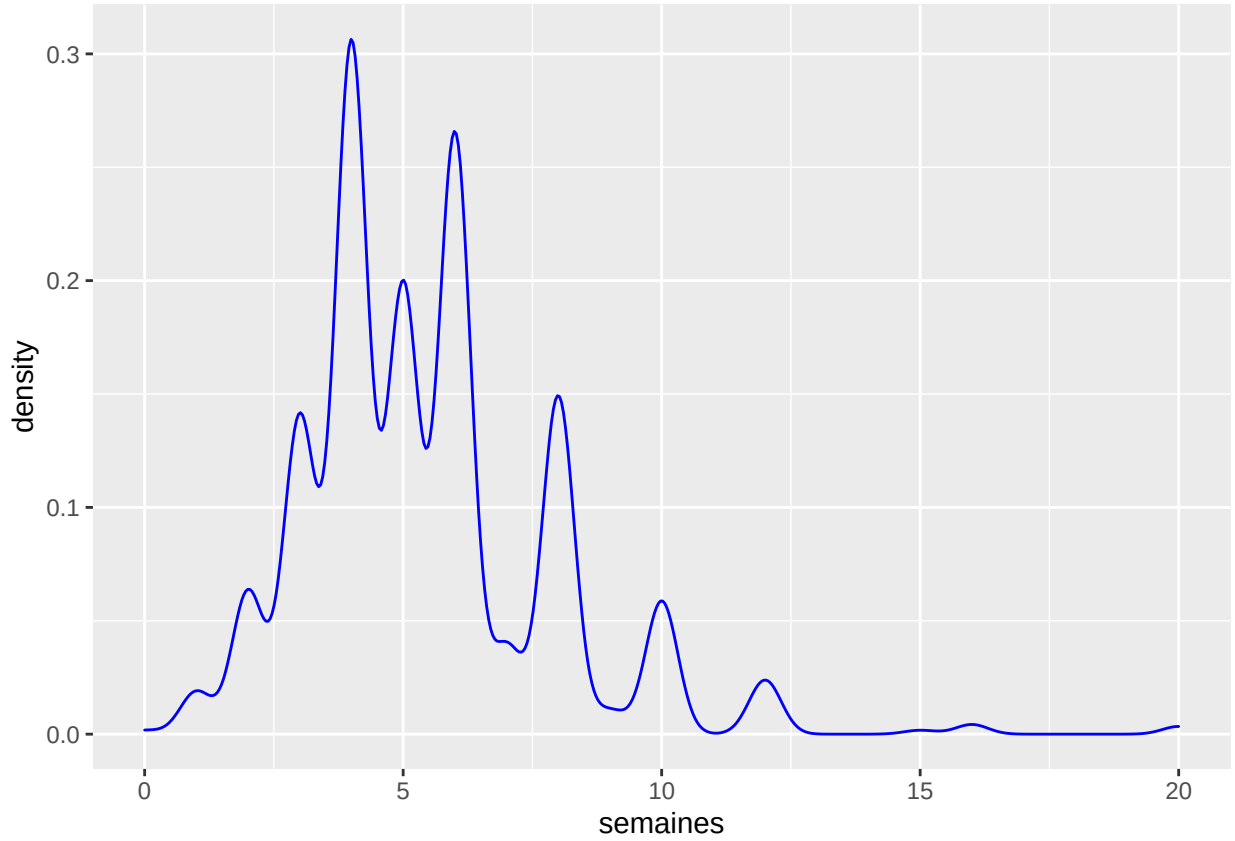

```
# Changer le type de graphique pour un histogramme :

b1 <- ggplot(data = covid, mapping = aes(x = pr_med_info_0a10)) +
  geom_bar(aes(fill = "red"))

b1
```

```
## Don't know how to automatically pick scale for object of type haven_labelled/vctrs_vctr/double. Defau
```

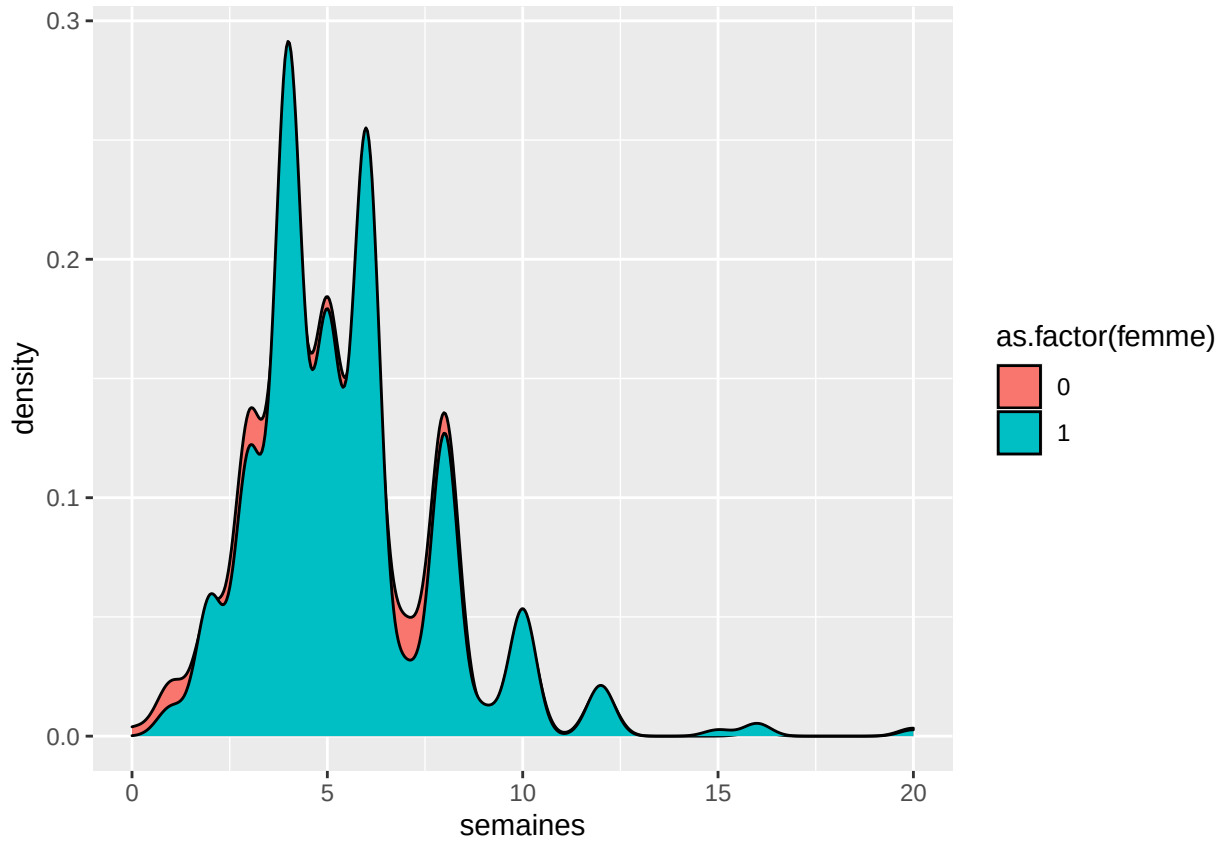


Les facettes : permettent d'effectuer plusieurs fois le même graphique selon les valeurs d'une ou plusieurs variables qualitatives (catégorielles) (notre *group_by*).

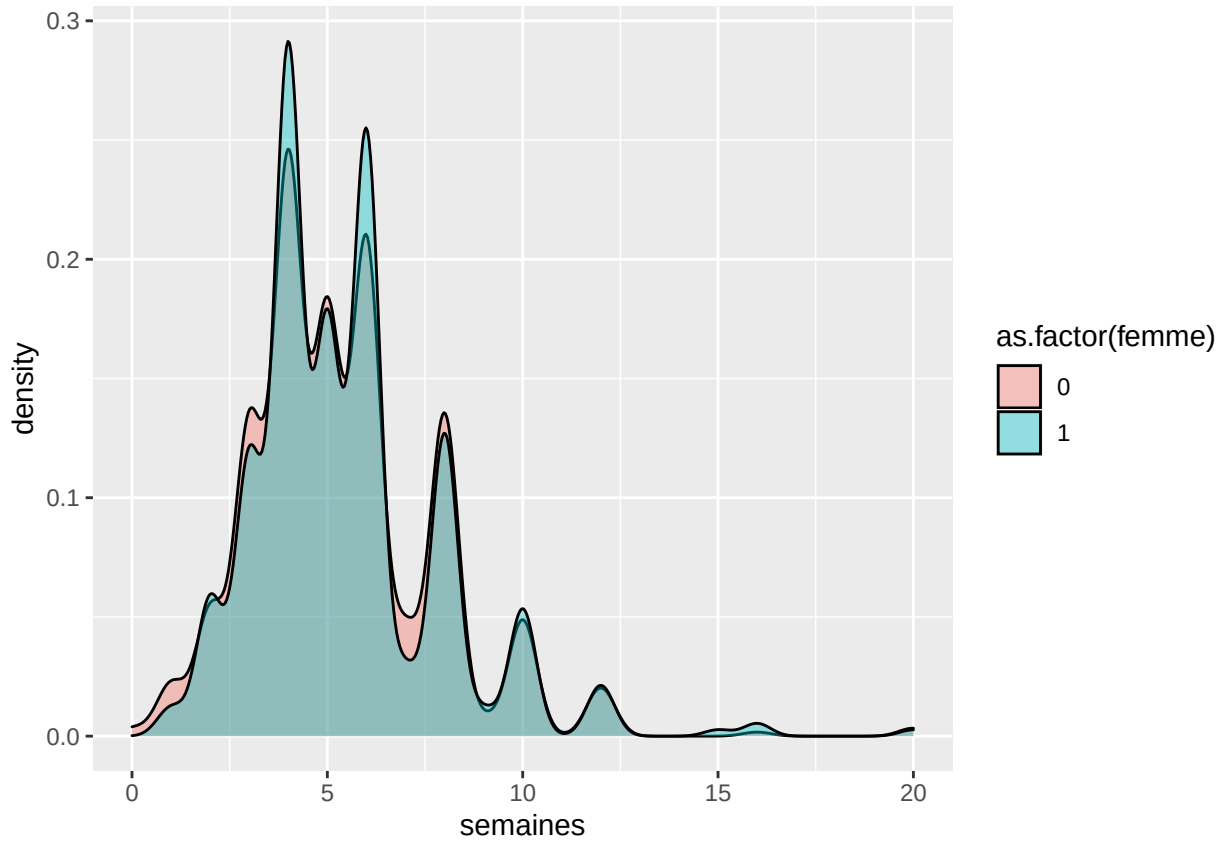
```
a <- ggplot(data = covid, mapping = aes(x = semaines)) +
  geom_density(color = "blue")
a
```



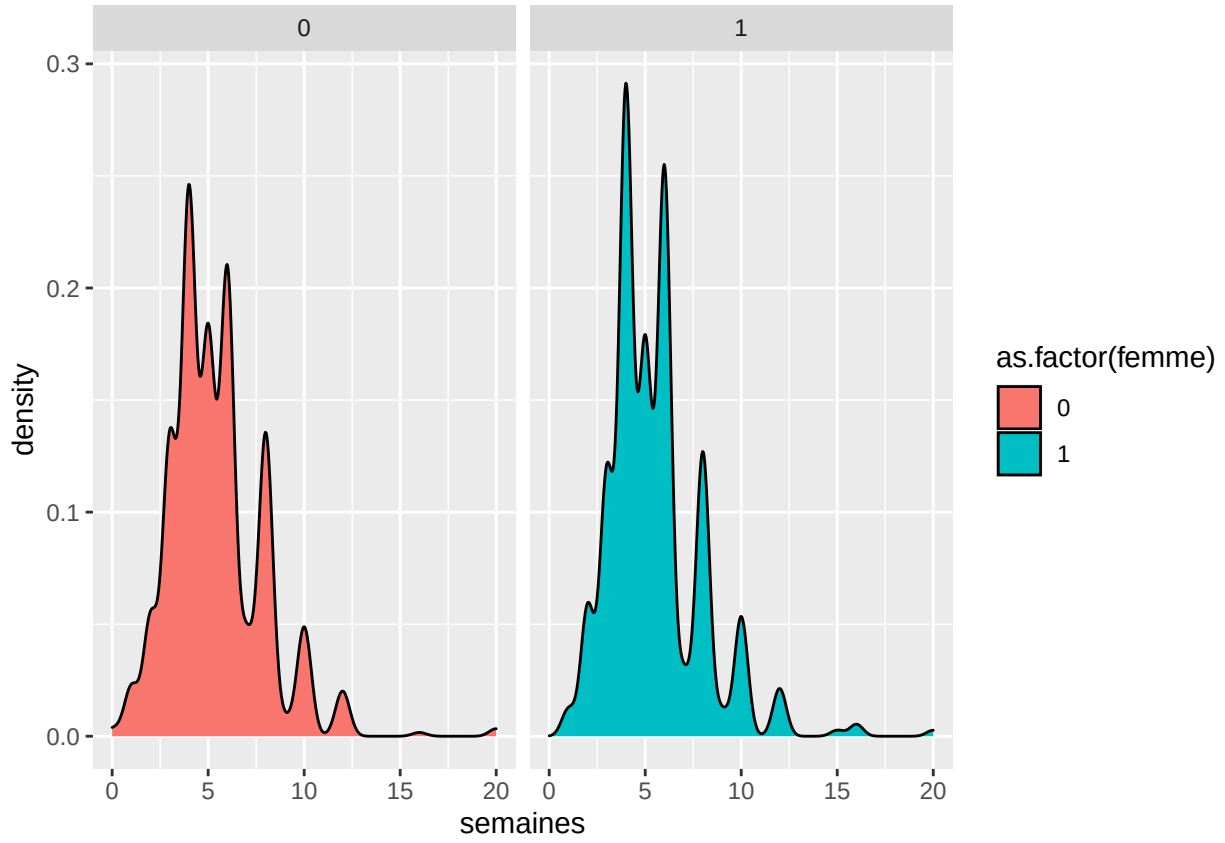

```
# On peut représenter la distribution pour les femmes et les hommes  :

a1 <- ggplot(data = covid, mapping = aes(x = semaines, fill = as.factor(femme))) +
  geom_density()
a1
```



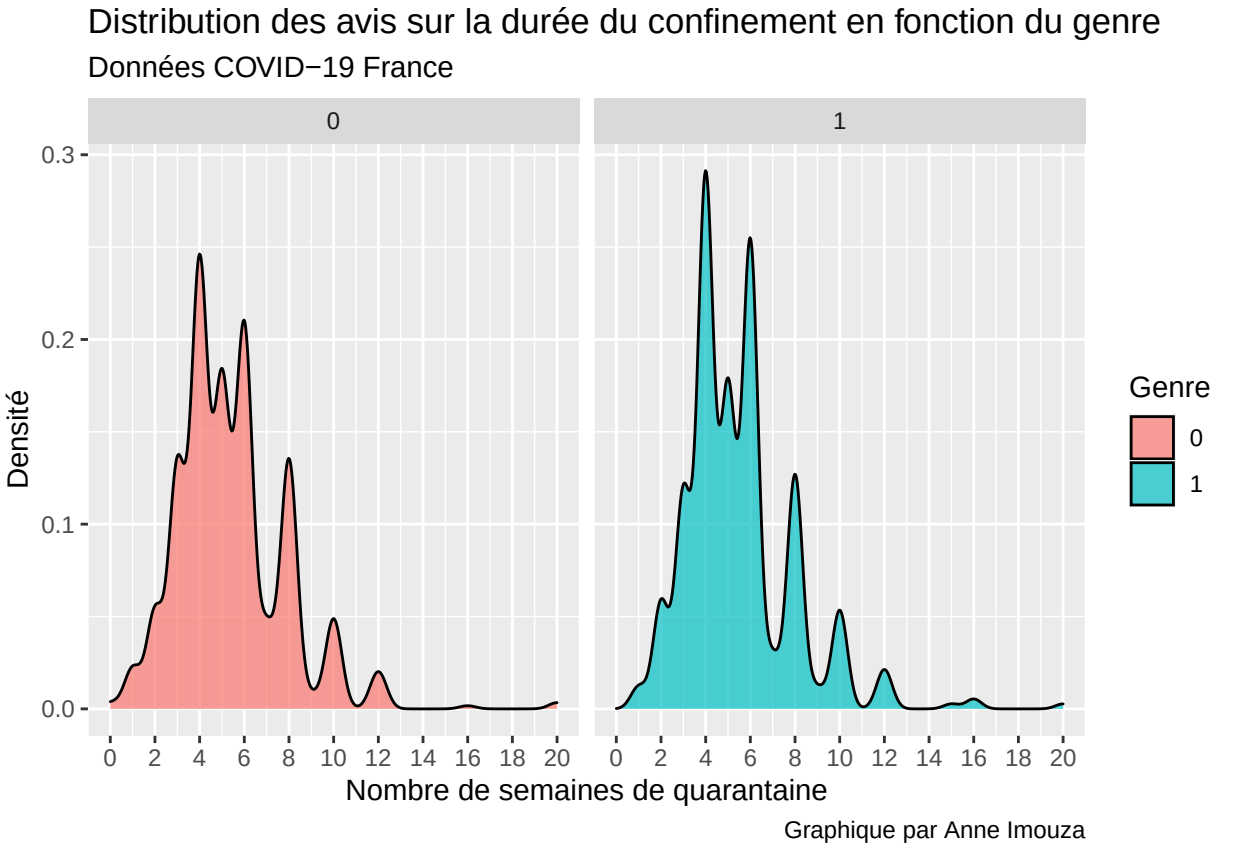

```
# On peut aussi modifier l'ombrage des distributions :
a2 <- ggplot(data = covid, mapping = aes(x = semaines, fill = as.factor(femme))) +
  geom_density(alpha = 0.4)

a2
```




```
# On peut créer deux graphiques distincts qui présentent la distribution
#des avis sur le nombre de semaines de quarantaine pour les femmes et
#pour les hommes genre :

a3 <- ggplot(data = covid, mapping = aes(x = semaines, fill = as.factor(femme))) +
  geom_density() +
  facet_wrap(~as.factor(femme))
a3
```

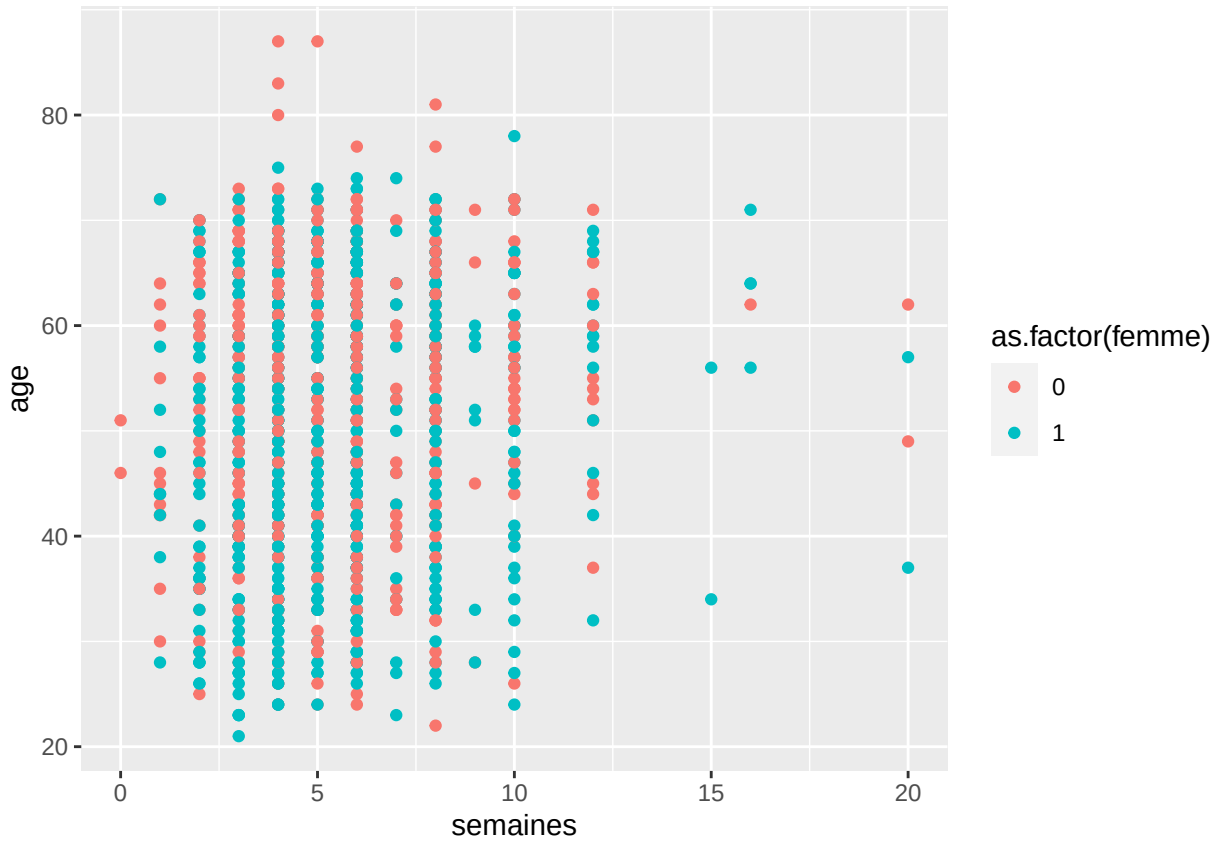



```
# Finalement, on peut ajouter les étiquettes :

a4 <- ggplot(data = covid, mapping = aes(x = semaines, fill = as.factor(femme))) +
  geom_density(alpha = 0.7) +
  facet_wrap(~as.factor(femme)) +
  scale_fill_discrete(name = "Genre") +
  scale_x_continuous(name = "Nombre de semaines de quarantaine",
                     breaks = seq(0,20, by = 2)) +
  labs(title = "Distribution des avis sur la durée du confinement en fonction du genre",
       subtitle = "Données COVID-19 France",
       x = "Nombres de semaines de confinement",
       y = "Densité",
       caption = "Graphique par Anne Imouza")
a4
```



Pour un graphique multivarié :

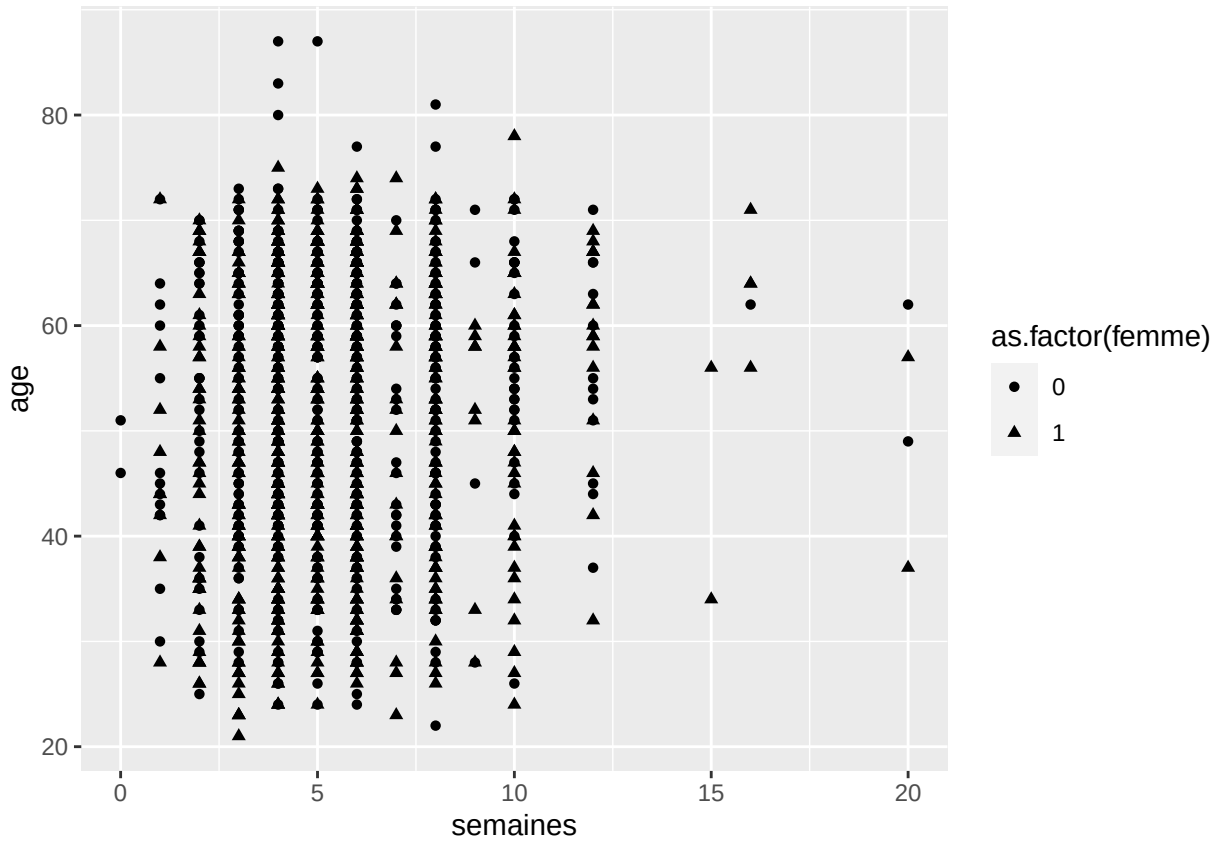
```
c <- ggplot(data = covid, mapping = aes(x = semaines, y = age, color = as.factor(femme))) +
  geom_point()
c
```




```
#on modifie la variable sexe (femme)  avec as.factor pour
#qu'elle soit une variable catégorielle, et non continue.
```

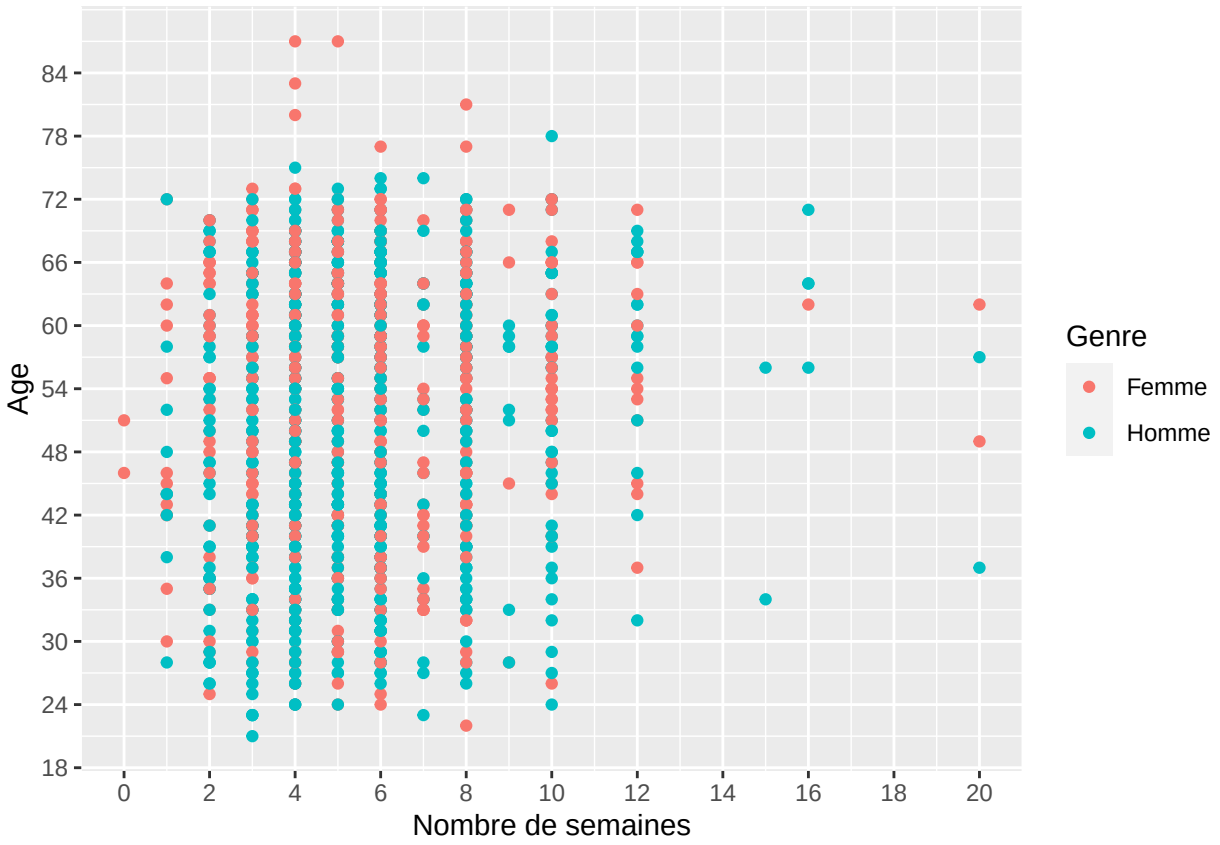
Utilisation de l'option shape :

```
d <- ggplot(data = covid, mapping = aes(x = semaines, y = age, shape = as.factor(femme))) +
  geom_point()
d
```



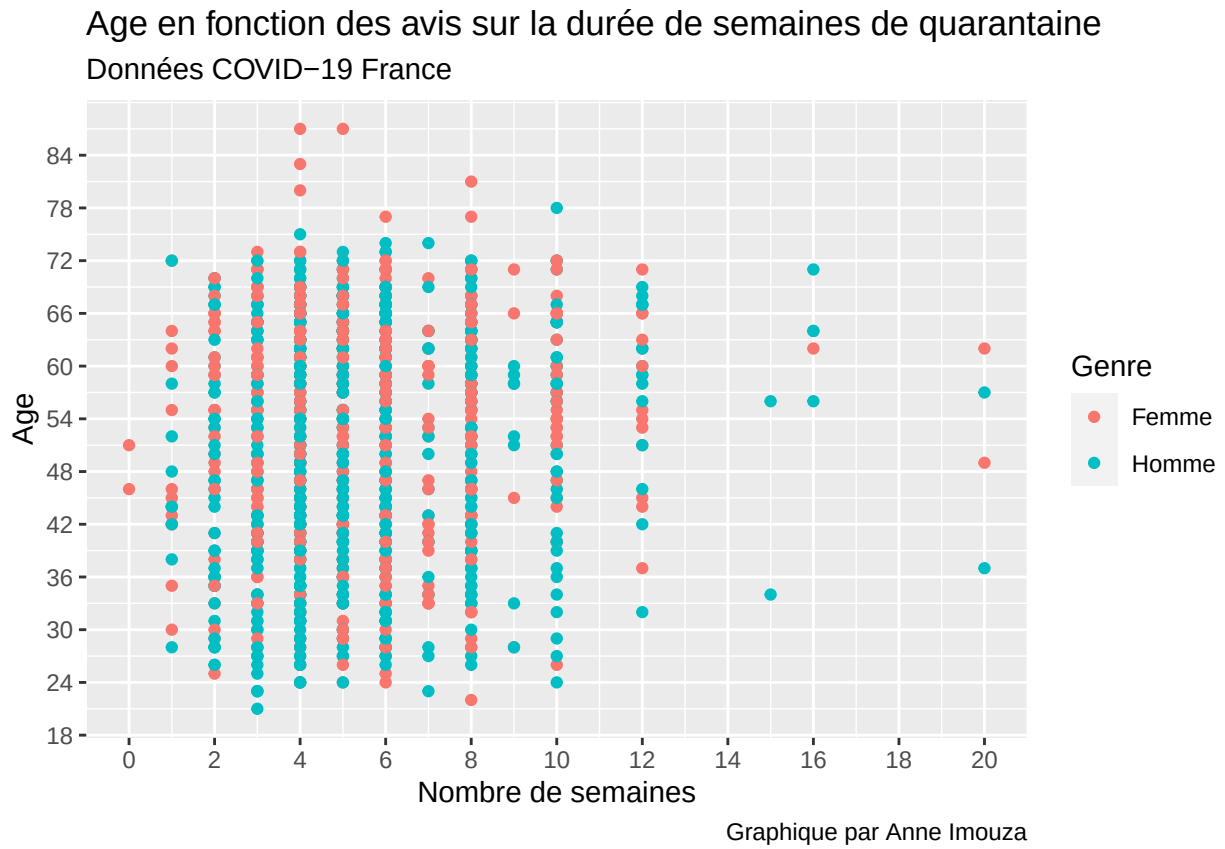
Les scales : Ils permettent de modifier la manière dont un attribut graphique va être relié aux valeurs d'une variable, et dont la légende correspondante va être affichée.

```
# scale
e <- c +
    scale_color_discrete(name = "Genre",
                         labels = c("Femme", "Homme")) +
    scale_x_continuous(name = "Nombre de semaines",
                       breaks = seq(0,20, by = 2)) +
    scale_y_continuous(name = "Age",
                       breaks = seq(0, 87, by = 6))

e
```



Les étiquettes : labs

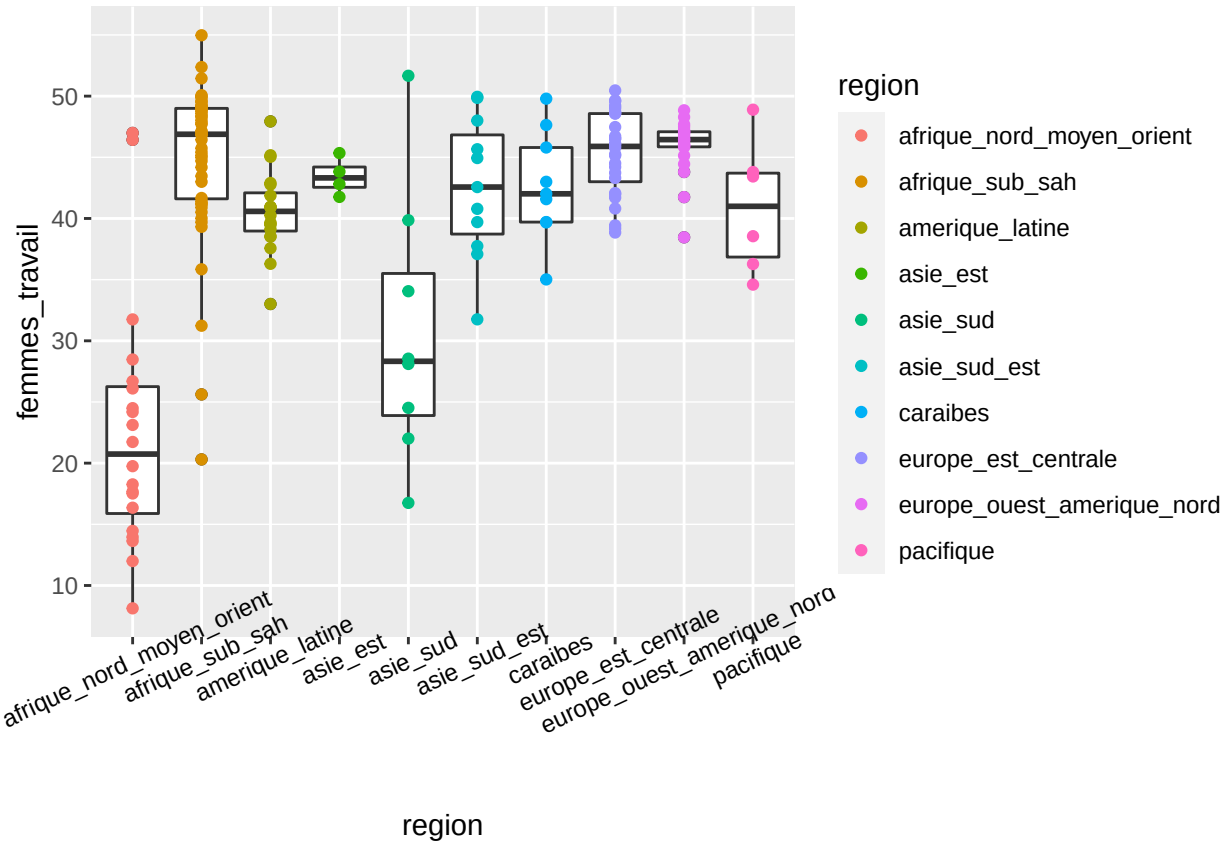
```
# labs
f <- e +
  labs(title = "Age en fonction des avis sur la durée de semaines de quarantaine",
       subtitle = "Données COVID-19 France",
       x = "Semaines de quarantaine",
       y = "Age",
       caption = "Graphique par Anne Imouza")

f
```



- Représentation de plusieurs graphiques sur une même figure

```
#Importation de la base de données Quality of governance :

quality_governance <- read.csv("Quality_Governance.csv")

# Un graphique avec plusieurs graphiques :

g <- ggplot(quality_governance) +
  geom_boxplot(aes(x = region, y = femmes_travail)) +
  geom_point(aes(x = region, y = femmes_travail, color = region)) +
  theme(axis.text.x = element_text(colour = "black", angle = 25)) #modification de l'angle du texte pou
g
```

```
## Warning: Removed 17 rows containing non-finite values (stat_boxplot).

## Warning: Removed 17 rows containing missing values (geom_point).
```

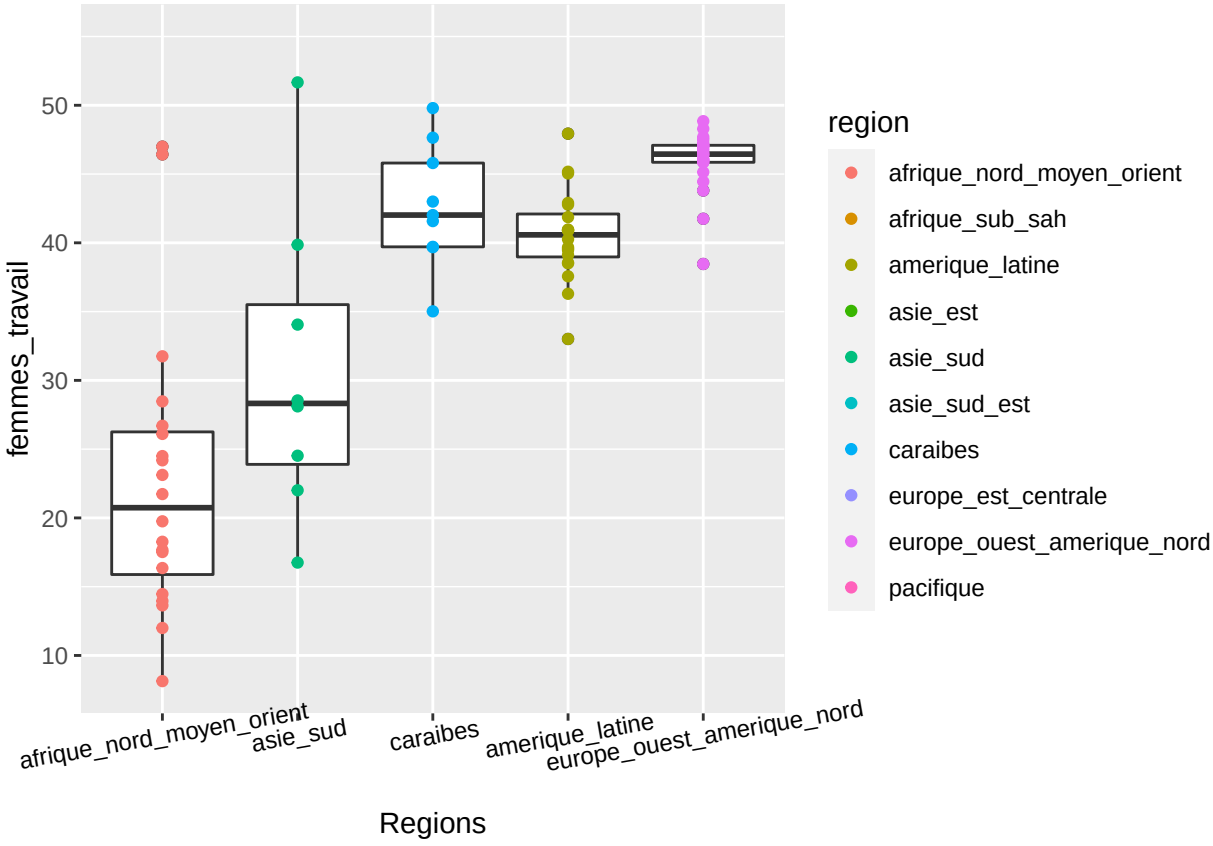



```
# On veut seulement voir pour les régions suivantes : afrique_nord_moyen_orient, asie_sud, caraibes, am

h <- ggplot(quality_governance) +
  geom_boxplot(aes(x = region, y = femmes_travail)) +
  geom_point(aes(x = region, y = femmes_travail, color = region)) +
  theme(axis.text.x = element_text(colour = "black", angle = 10)) +
  scale_x_discrete("Regions", limits = c("afrique_nord_moyen_orient", "asie_sud","caraibes", "amerique_
h
```

```
## Warning: Removed 106 rows containing missing values (stat_boxplot).
```

```
## Warning: Removed 8 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 114 rows containing missing values (geom_point).
```



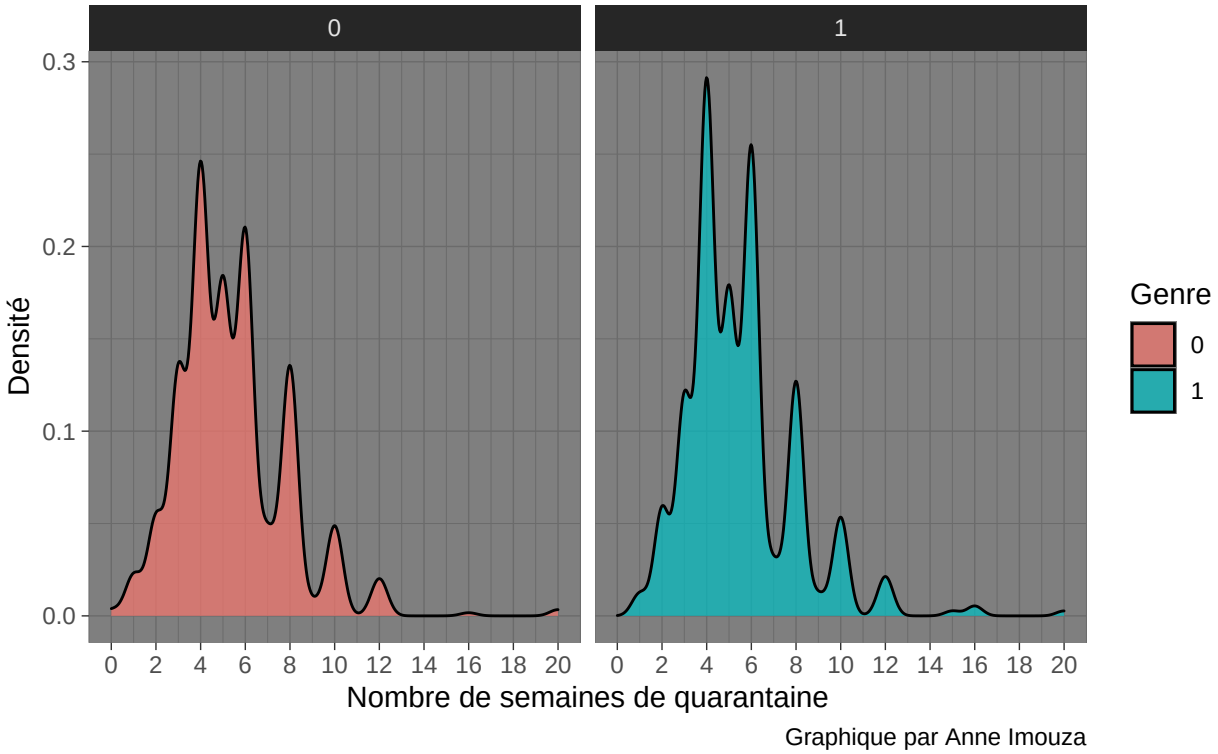
Les thèmes :

- Ils permettent de contrôler l'affichage de tous les éléments du graphique qui ne sont pas reliés aux données : **titres**, **grilles**, **fonds**, etc.

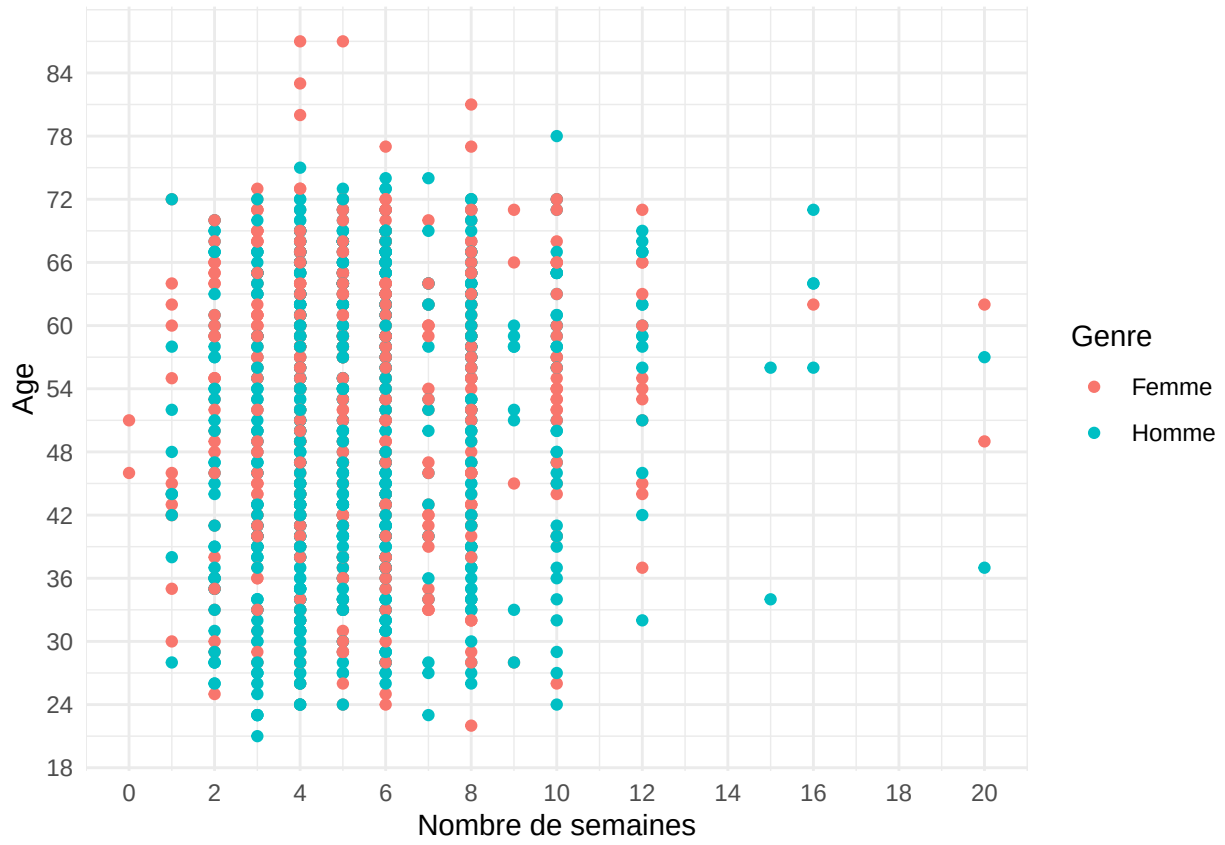
*On peut utiliser plusieurs thèmes pour présenter nos résultats : + r + theme_bw() : fond blanc avec des lignes de quadrillage + r + theme_gray() : Fond gris + r + theme_dark() : Noir pour les contrastes + r + theme_classic() + r + theme_light() + r + theme_linedraw() + r + theme_minimal(): thème minimal + r + theme_void() : Thème vide

Pour plus d'information : https://ggplot2.tidyverse.org/reference/theme.html

```
#Notre graphique de densité univarié :

i <- a4 +
  theme_dark()
i
```



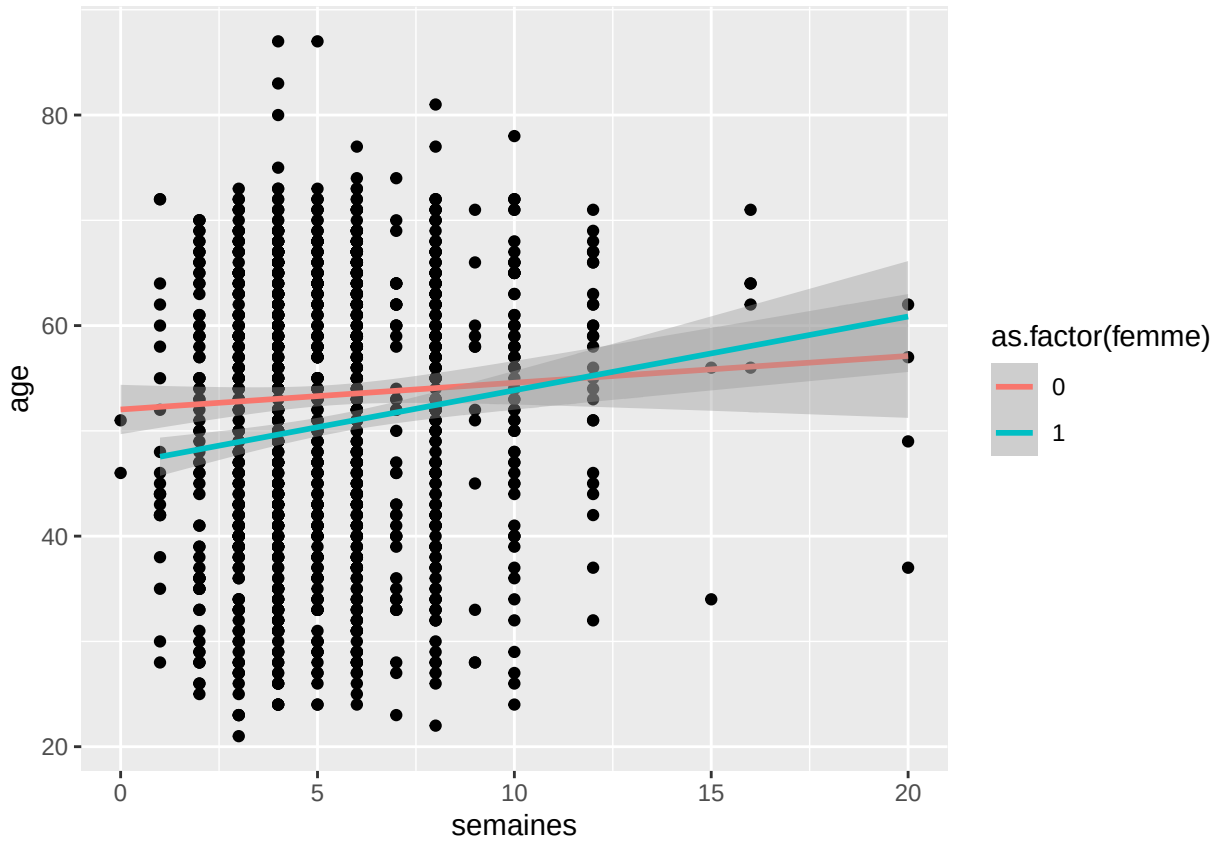

```
#Notre grahique multivarié
j <- e + theme_minimal()
j
```



Régression linéaire :

```
k <- ggplot(data = covid, mapping = aes(x = semaines, y = age)) +
  geom_point() +
  geom_smooth(aes(color = as.factor(femme)), method = "lm")
k
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Enregistrer un graphique :

```
ggsave(filename = "graphique_regression_lineaire.png", plot = k, width = 9, height = 6)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

==========

Exercices (avec la correction) :

1. Importez la base de données : Quality of Governance :

```
quality_governance <- read.csv("Quality_Governance.csv")
```

2. Résumez la variable `femmes_travail` qui se trouve dans la base de données `quality_governance`. Utilisez la fonction *summary* :

```
summary(quality_governance$femmes_travail)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   8.133  39.319  43.828  41.036  47.108  54.976      17
```

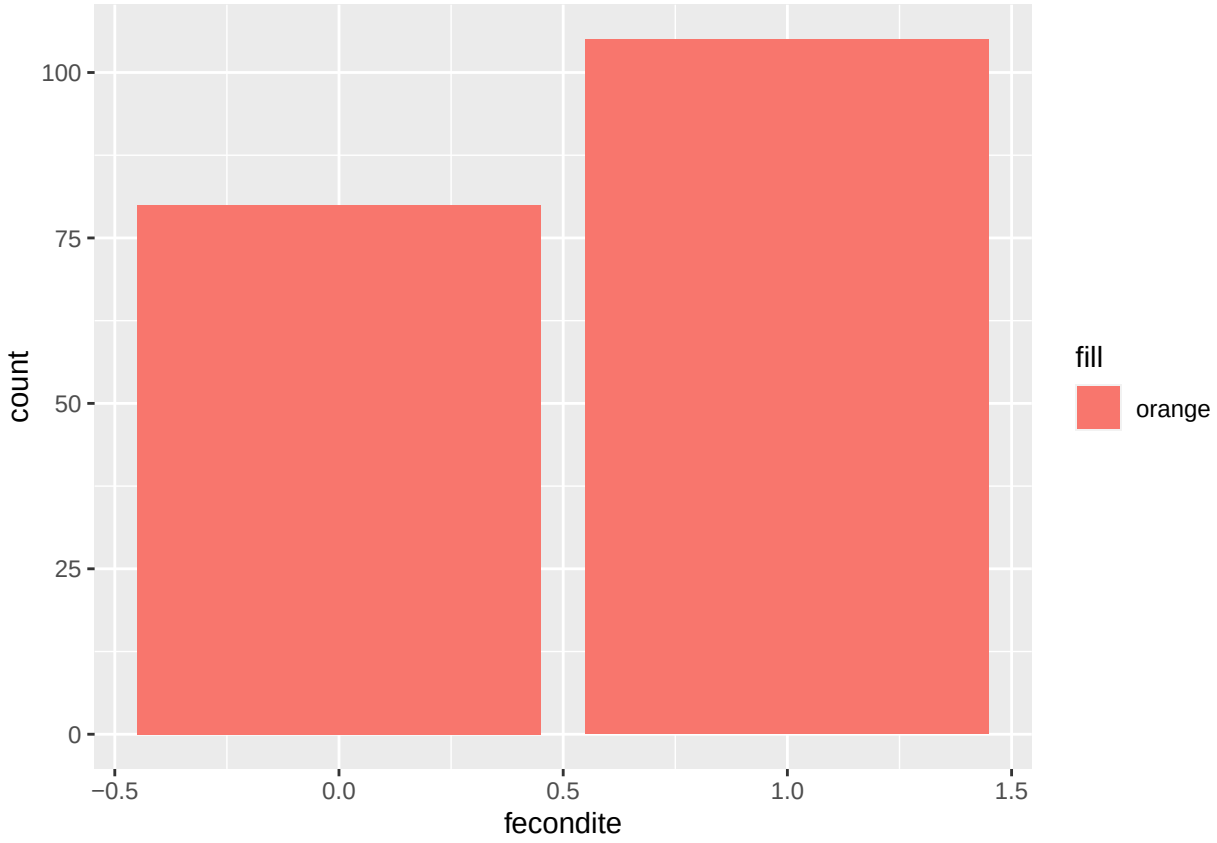
3. Créez une nouvelle base de données qui se nomme `quality_gov_a`qui ne contient pas les variables suivantes : `taux_dep`, `renouvelable` et `terre_arable`. utilisez la fonction *%>%* et *filter*.

```
quality_gov_a <- quality_governance %>%
  select(-taux_dep, -renouvelable, -terre_arable)
```

4. Créez un graphique univarié avec la variable `fecondite`.

```
graph_fec <- ggplot(quality_gov_a, aes(x=fecondite)) +
  geom_bar(aes(fill = "orange"))

graph_fec
```

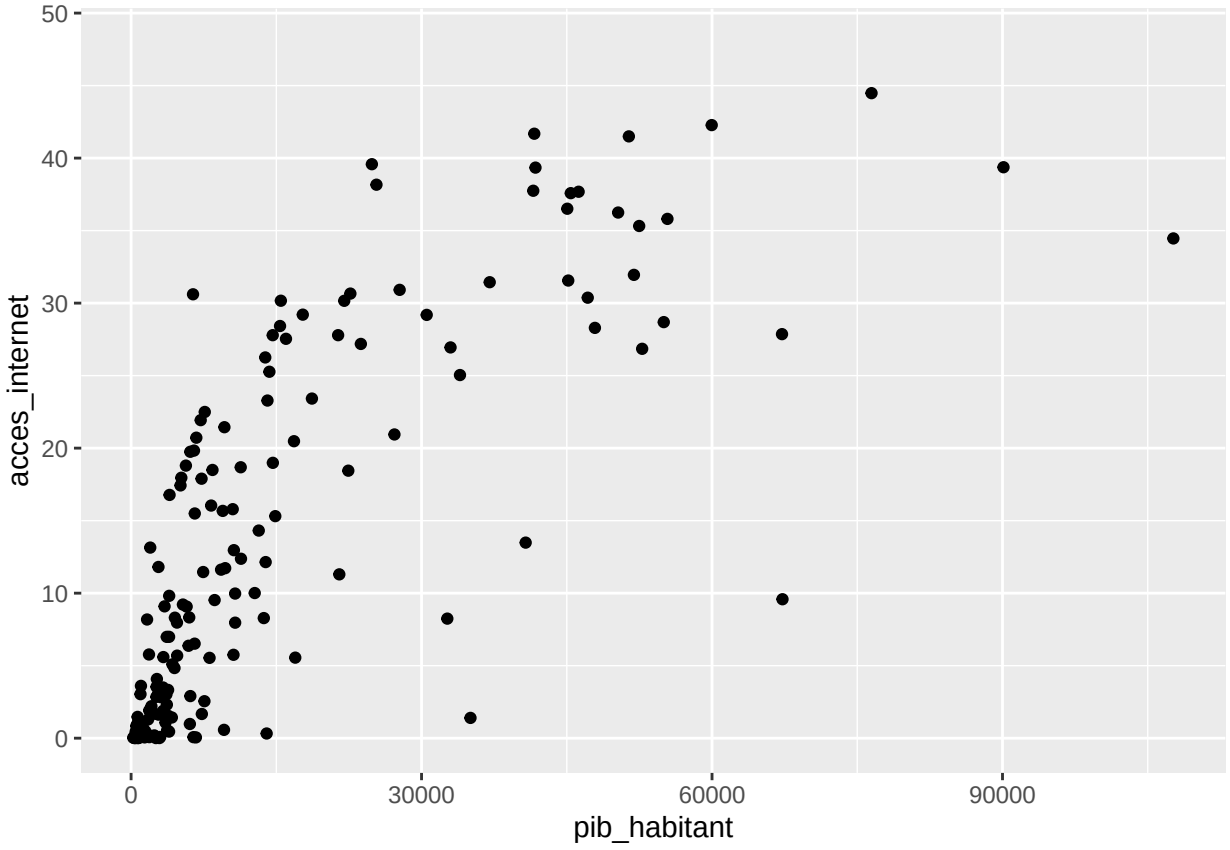
```
## Warning: Removed 9 rows containing non-finite values (stat_count).
```



5. Créez un graphique bivarié avec les variables `pib_habitant` (x) et `acces_internet` (y).

```
graph_bi <- ggplot(quality_gov_a, aes(x= pib_habitant, y = acces_internet)) +
  geom_point()
graph_bi
```

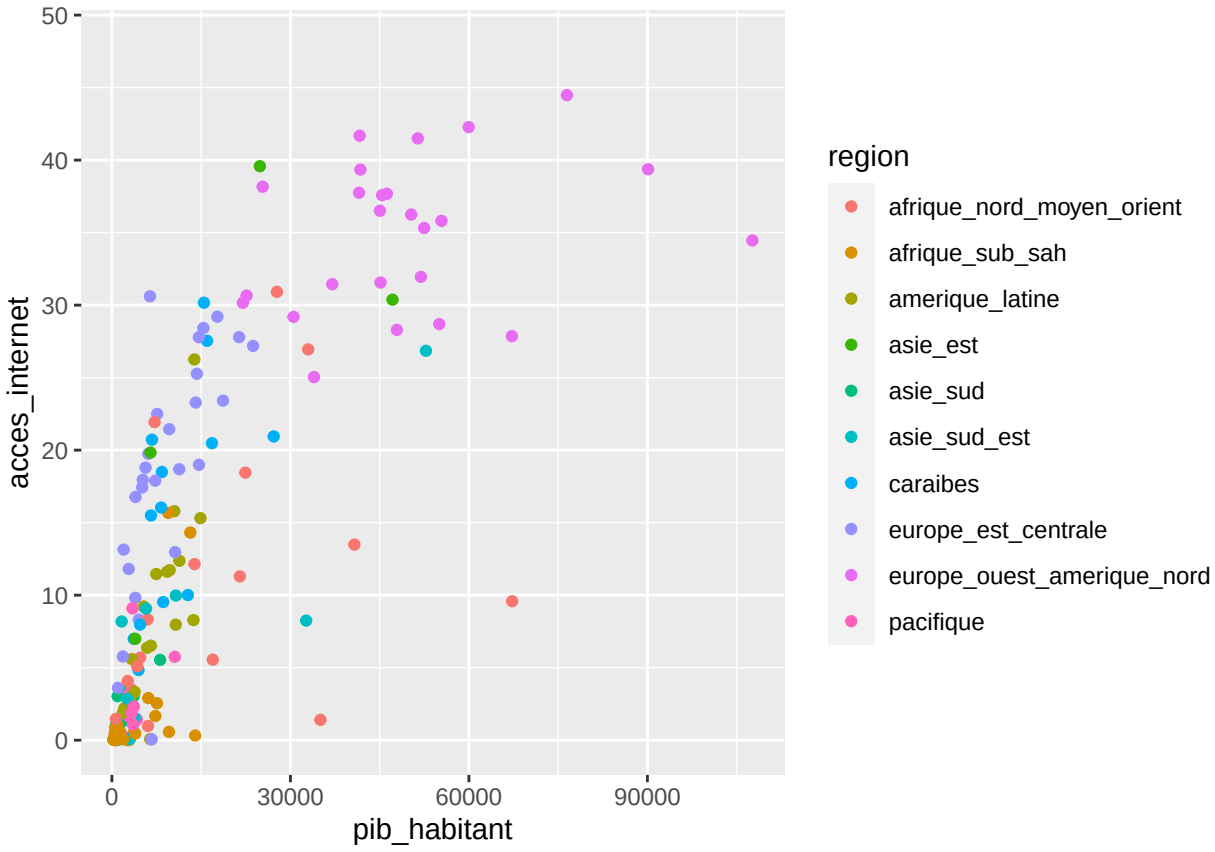
```
## Warning: Removed 12 rows containing missing values (geom_point).
```



6. Ajoutez au graphique précédent la représentation des régions en couleur pour chaque variable (`pib_habitant` (x) et `acces_internet` (y)):

```
graph_bi <- ggplot(quality_gov_a, aes(x= pib_habitant, y = acces_internet, color = region)) +
  geom_point()
graph_bi
```

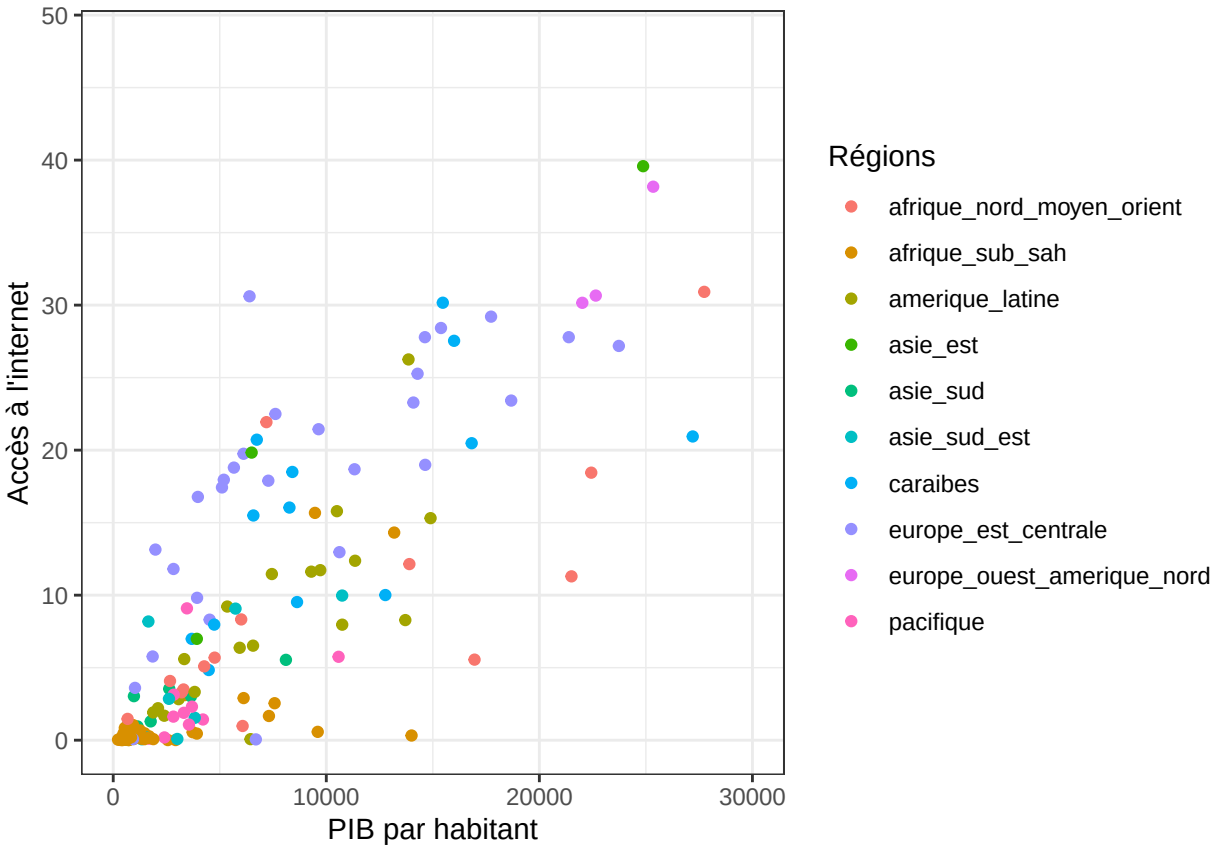
```
## Warning: Removed 12 rows containing missing values (geom_point).
```



7. Ajoutez les `scales`et un `thème`. 7.1 Limitez l'intervalle du PIB par habitant de 0 à 30000.

```
#Ajout des scales, limitation PIB et thème :
graph_bi <- ggplot(quality_gov_a, aes(x= pib_habitant, y = acces_internet, color = region)) +
  geom_point() +
  scale_x_continuous("PIB par habitant",
                     limits = c(0,30000)) + #limite
  scale_y_continuous("Accès à l'internet") +
  scale_color_discrete("Régions") +
  theme_bw() #thème

graph_bi
```

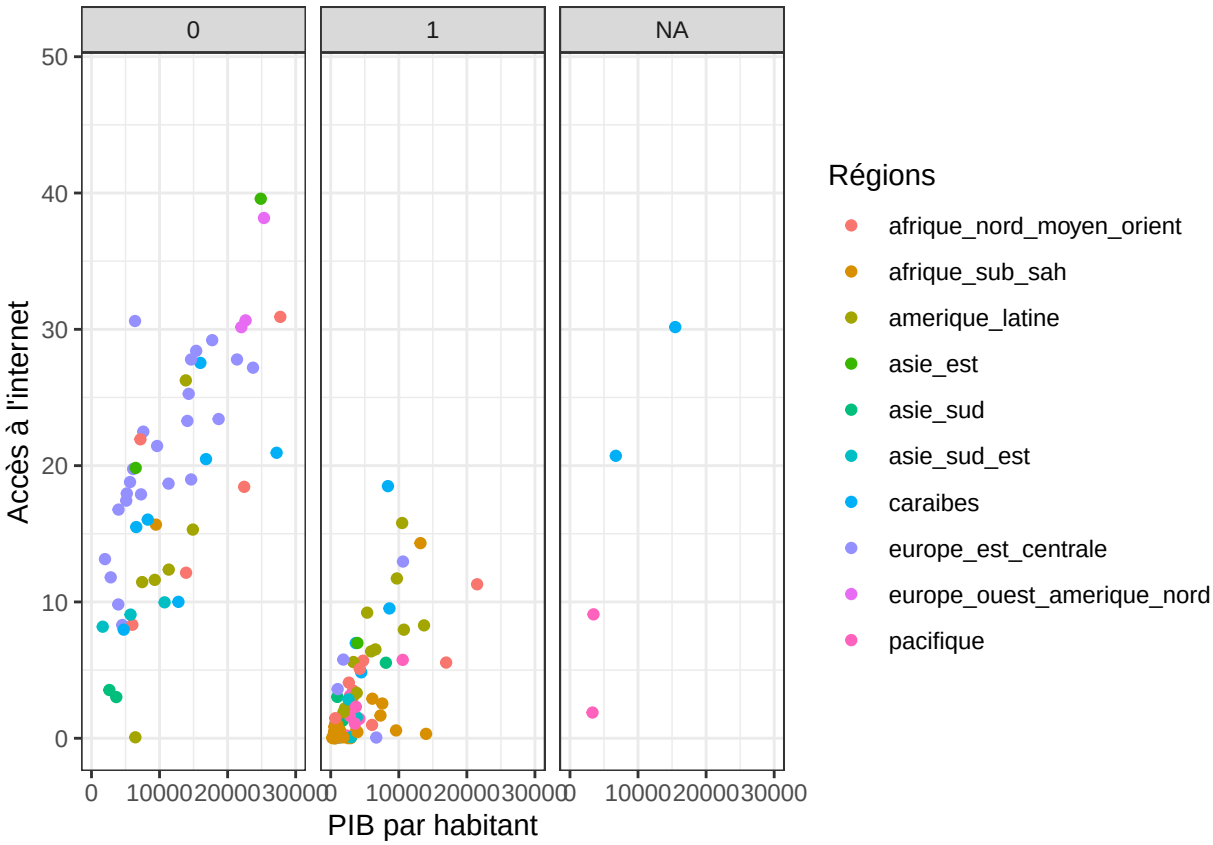
```
## Warning: Removed 41 rows containing missing values (geom_point).
```



8 Créez deux graphiques distincts de nos variables précédente en fonction du genre (`facet_wrap`) (Un graphique qui ne contient que les hommes et un autre graphique qui ne détient que les femmes).

```
#on veut enlever les NA. une solution possible :
quality_gov_a <- quality_gov_a %>%
  drop_na(fecondite)

graph_bi_a <- graph_bi +
  facet_wrap(~fecondite)

graph_bi_a
```

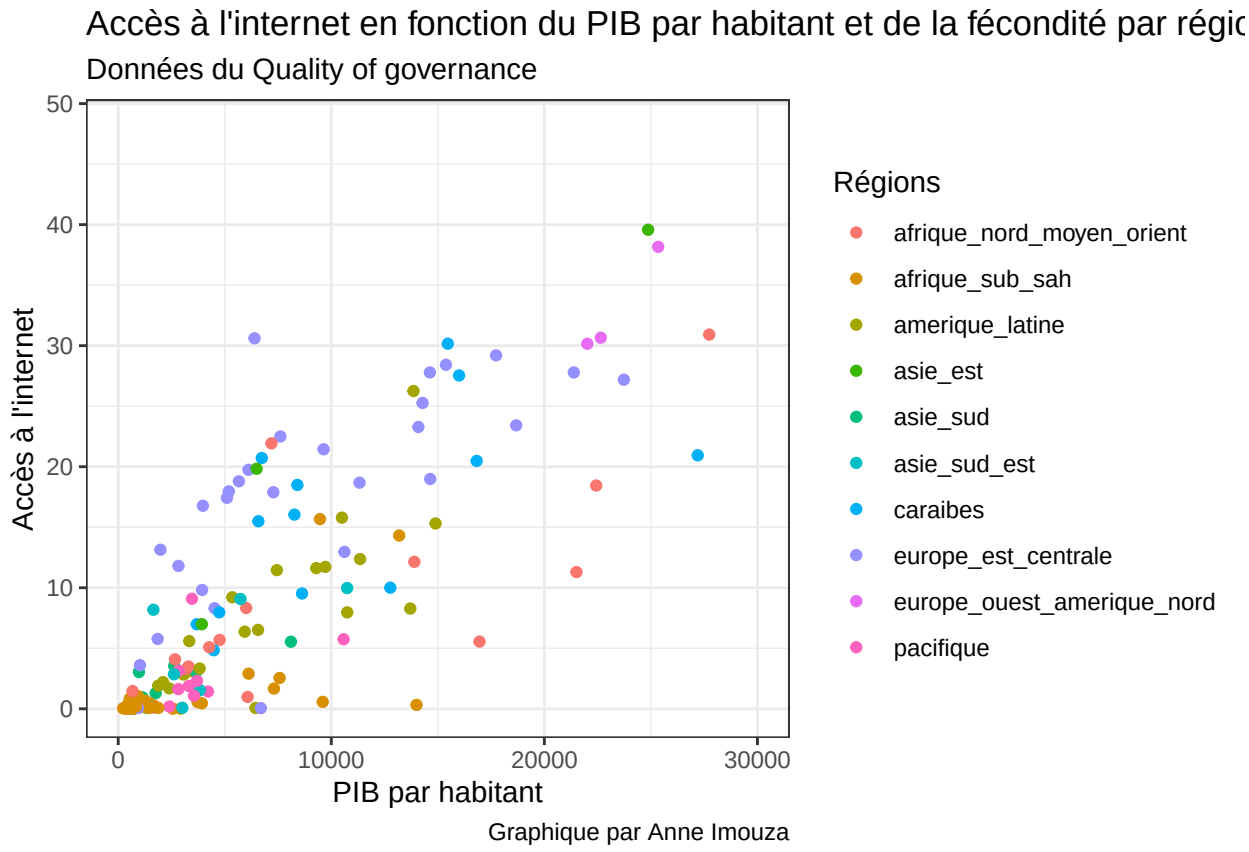
```
## Warning: Removed 41 rows containing missing values (geom_point).
```



9. En utilisant la commande `labs`, ajoutez un titre, un sous titre, renommez les variables et indiquez votre prénom et nom.

```
graph_bi_b <- graph_bi +
    labs(title = "Accès à l'internet en fonction du PIB par habitant et de la fécondité par région",
       subtitle = "Données du Quality of governance",
       x = "Pib par habitant",
       y = "Accès à l'internet",
       caption = "Graphique par Anne Imouza")
graph_bi_b
```

```
## Warning: Removed 41 rows containing missing values (geom_point).
```



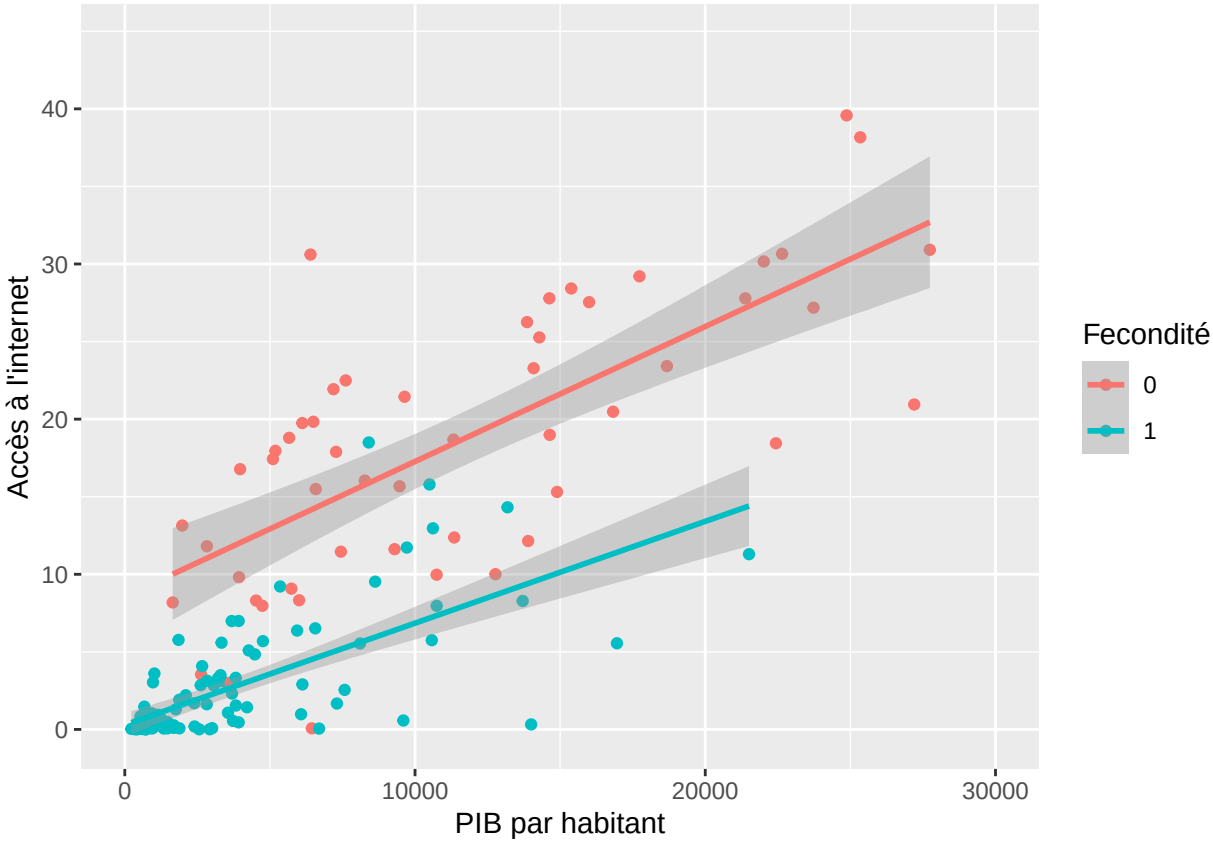
10. Faites un graphique avec une régression linéaire entre les variables `pib_habitant` et `acces_internet` en tenant compte de la fécondité :

```
graph_ai <- ggplot(quality_gov_a, aes(x= pib_habitant, y = acces_internet, color = as.factor(fecondite))
  geom_point() +
  scale_x_continuous("PIB par habitant",
                     limits = c(0,30000)) +
  scale_y_continuous("Accès à l'internet") +
  scale_color_discrete("Fecondité") +
  geom_smooth(method = "lm")

graph_ai
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 36 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 36 rows containing missing values (geom_point).
```



11. Enregistrez votre graphique univarié et votre graphique bivarié sur votre ordinateur.

```
ggsave(filename = "graphique_regression_lineaire2.png", plot = graph_ai, width = 9, height = 6)
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 36 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 36 rows containing missing values (geom_point).
```

Ressources très utiles :

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

https://ggplot2.tidyverse.org

https://datacarpentry.org/R-ecology-lesson/04-visualization-ggplot2.html

https://sicss.io

https://www.icpsr.umich.edu/web/pages/