

02 - Visualisation de données

PRO1036 - Analyse de données scientifiques en R

Tim Bollé

September 9, 2024

Qu'est ce qu'un dataset ?

Terminologie

En français, on parle de jeu de données. Concrètement, c'est un tableau:

- Chaque ligne correspond à une **observation**
- Chaque colonne correspond à une **variable**

```
| 1 starwars
# A tibble: 87 × 14
  name      height  mass hair_color skin_color eye_color birth_year sex   gender
  <chr>     <int> <dbl> <chr>       <chr>       <chr>       <dbl> <chr> <chr>
1 Luke Sk...    172     77 blond      fair        blue         19 male   masculin...
2 C-3PO        167     75 <NA>       gold        yellow       112 none   masculin...
3 R2-D2         96      32 <NA>       white, bl... red          33 none   masculin...
4 Darth V...    202     136 none       white        yellow       41.9 male   masculin...
5 Leia Or...    150      49 brown      light       brown        19 female feminin...
6 Owen La...    178     120 brown, gr... light       blue         52 male   masculin...
7 Beru Wh...    165      75 brown      light       blue         47 female feminin...
8 R5-D4         97      32 <NA>       white, red red          NA none   masculin...
9 Biggs D...    183      84 black      light       brown        24 male   masculin...
10 Obi-Wan...   182      77 auburn, w... fair        blue-gray      57 male   masculin...
# i 77 more rows
# i 5 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>
```

Obiwan Kenobi

- Taille = 182cm
- Poids = 77kg
- Couleur des cheveux = brun
- Couleur des yeux = gris-bleu
- Année de naissance = 57BBY
- Planète d'origine = Stewjon ...



Hello there.

Jetons un coup d'oeil

Nous pouvons avoir une vue d'ensemble avec `glimpse`:

```
| 1 glimpse(starwars)
```

```
Rows: 87
Columns: 14
$name      <chr> "Luke Skywalker", "C-3PO", "R2-D2", "Darth Vader", "Leia Or...
$height    <int> 172, 167, 96, 202, 150, 178, 165, 97, 183, 182, 188, 180, 2...
$mass      <dbl> 77.0, 75.0, 32.0, 136.0, 49.0, 120.0, 75.0, 32.0, 84.0, 77....
$hair_color <chr> "blond", NA, NA, "none", "brown", "brown, grey", "brown", N...
$skin_color <chr> "fair", "gold", "white, blue", "white", "light", "light", "...
$eye_color  <chr> "blue", "yellow", "red", "yellow", "brown", "blue", "blue", ...
$birth_year <dbl> 19.0, 112.0, 33.0, 41.9, 19.0, 52.0, 47.0, NA, 24.0, 57.0, ...
$sex        <chr> "male", "none", "none", "male", "female", "male", "female", ...
$gender     <chr> "masculine", "masculine", "masculine", "masculine", "femini...
$homeworld <chr> "Tatooine", "Tatooine", "Naboo", "Tatooine", "Alderaan", "T...
$species    <chr> "Human", "Droid", "Droid", "Human", "Human", "Human", "Huma...
$films      <list> <"A New Hope", "The Empire Strikes Back", "Return of the J...
$vehicles   <list> <"Snowspeeder", "Imperial Speeder Bike">, <>, <>, <>, "Imp...
$starships  <list> <"X-wing", "Imperial shuttle">, <>, <>, "TIE Advanced x1", ...
```

Pour en savoir plus...

Pour avoir le plus d'information possible, l'aide reste la meilleure option

The screenshot shows the R Documentation interface for the `starwars` dataset. The title bar says "R Documentation". The main title is "Starwars characters". The "Description" section states: "The original data, from SWAPI, the Star Wars API, <https://swapi.py4e.com/>, has been revised to reflect additional research into gender and sex determinations of characters." The "Usage" section shows the command `starwars`. The "Format" section describes it as "A tibble with 87 rows and 14 variables:". It lists two variables: "name" (Name of the character) and "height" (Height (cm)).

```
1 ?starwars
```

Files Plots Packages Help Viewer Presentation

R: Starwars characters ▾ Find in Topic

starwars {dplyr}

R Documentation

Starwars characters

Description

The original data, from SWAPI, the Star Wars API, <https://swapi.py4e.com/>, has been revised to reflect additional research into gender and sex determinations of characters.

Usage

```
starwars
```

Format

A tibble with 87 rows and 14 variables:

name
 Name of the character

height
 Height (cm)

Description des données

Une information importante à connaitre est le nombre de ligne et de colonnes dans le jeu de données

```
| 1 nrow(starwars) # nombre de lignes  
[1] 87  
| 1 ncol(starwars) # nombre de colonnes  
[1] 14  
| 1 dim(starwars) # dimensions du tableau: lignes x colonnes  
[1] 87 14
```

Exploration de données

Qu'est ce que c'est ?

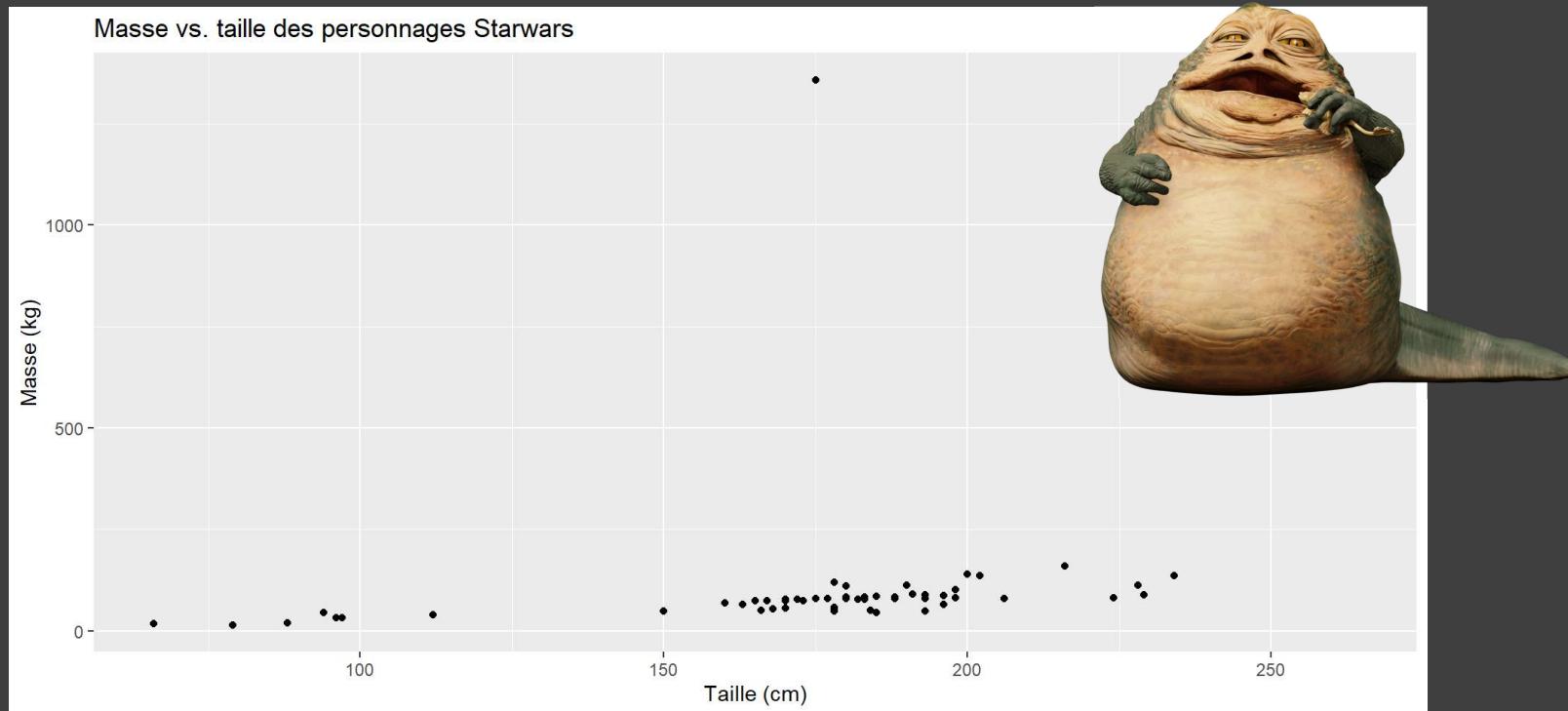
L'idée de l'exploration de données est de chercher à résumer les principales caractéristiques du jeu de données.

Souvent, cette analyse est visuelle: c'est de la visualisation de données !

Elle s'accompagne aussi d'analyses statistiques élémentaires.

Masse vs Taille

Comment décririez-vous la relation entre la masse et la taille ? Y a-t-il des outliers ?



Data Viz

Data Visualisation

“The simple graph has brought more information to the data analyst’s mind than any other device.” — John Tukey

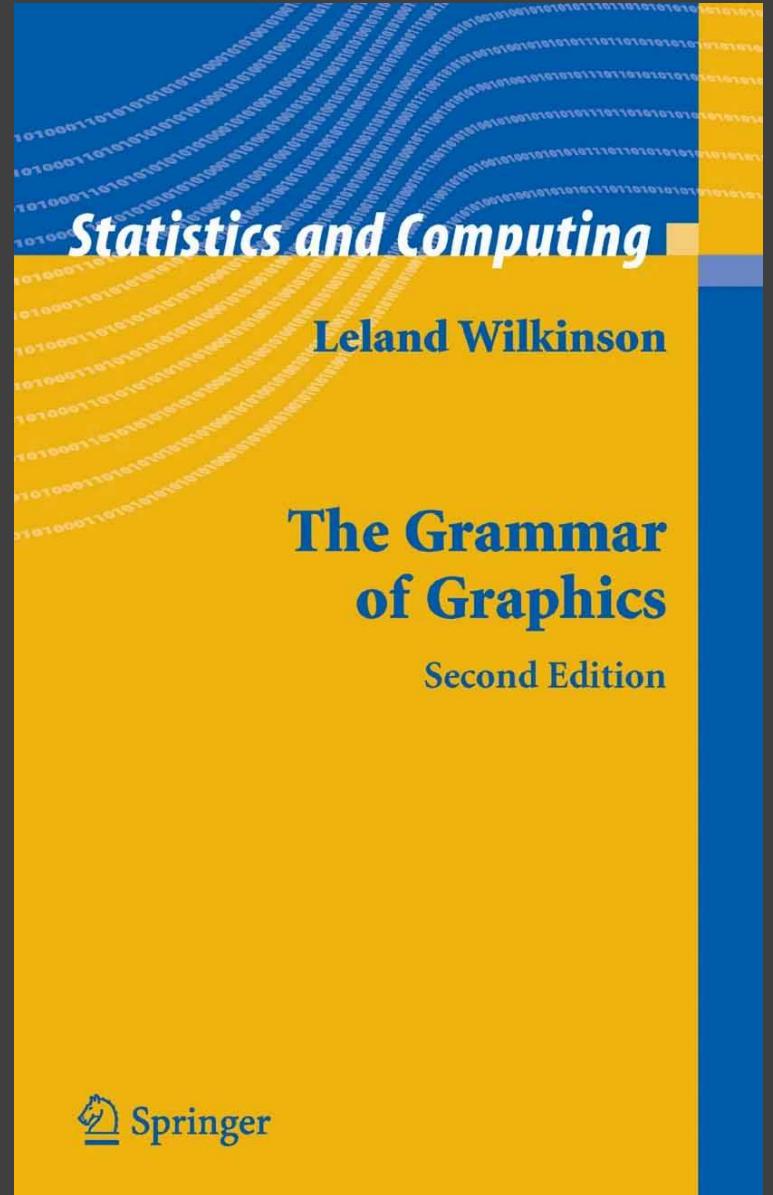
La visualisation de données cherche à étudier comment représenter les données.

Il y a de nombreux outils pour la DataViz

- R est l'un d'eux
- Il y a plusieurs modules pour visualiser les données dans R
 - Nous allons utiliser **ggplot2**

ggplot2

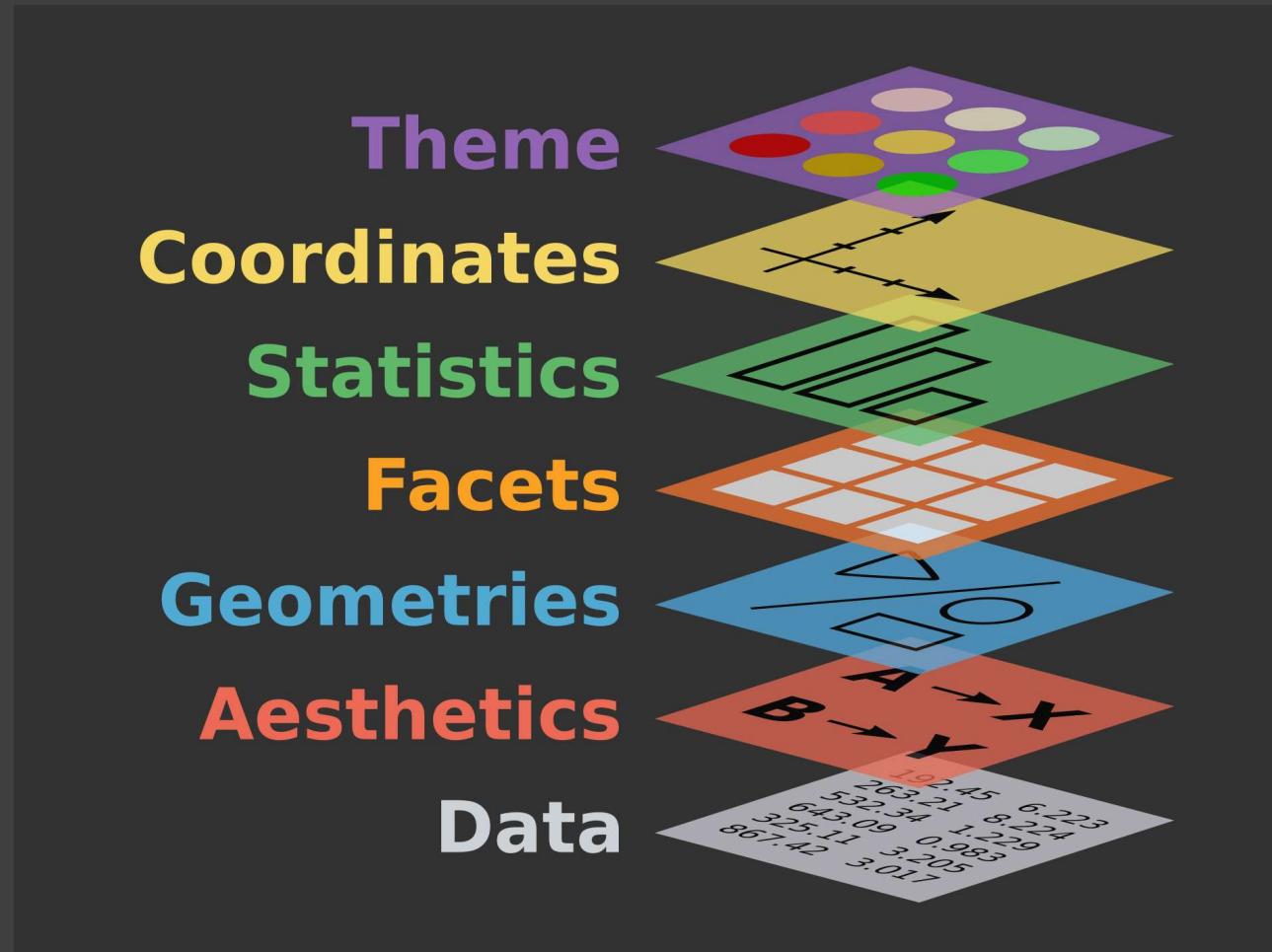
- `ggplot2` fait partie du Tidyverse et sert à la visualisation de données
- `gg` veut dire *Grammar of Graphics*
- Inspiré du livre *Grammar of Graphics* de Leland Wilkinson



(Wilkinson, 2005)

Grammar of Graphics

Permet de décrire toutes les briques d'un graphe

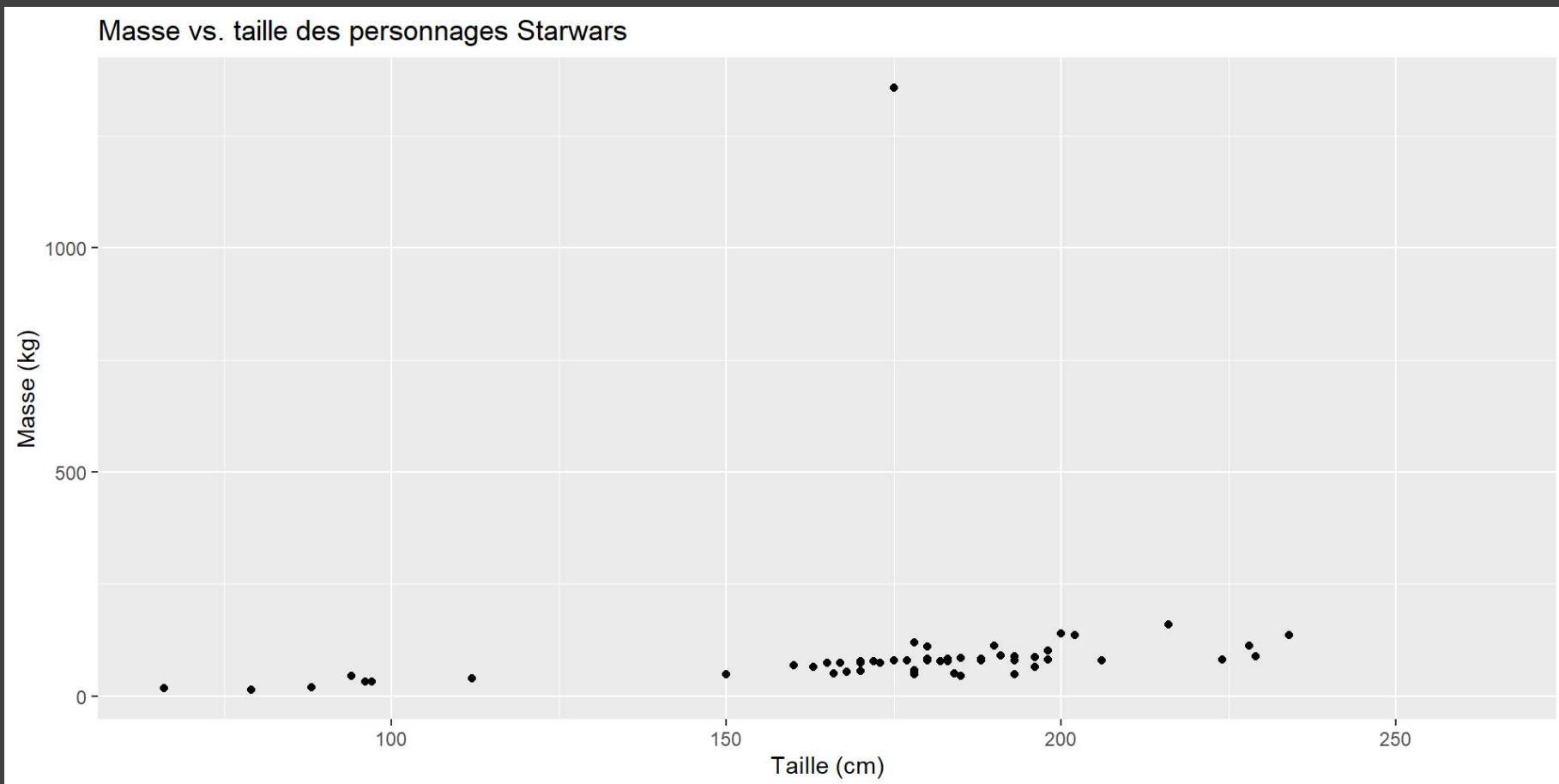


source

masse vs taille

```
1 ggplot(data = starwars, mapping = aes(x = height, y = mass)) +  
2   geom_point() +  
3   labs(title = "Masse vs. taille des personnages Starwars",  
4       x = "Taille (cm)", y = "Masse (kg)")
```

Warning: Removed 28 rows containing missing values or values outside the scale range
(`geom_point()`).



Kesako

- Quelle fonction s'occupe du plotting ?
- Quel jeu de données est visualisé ?
- Quelles variables sont projetées sur quels éléments (*aesthetics*) du graphe ?
- Que signifie le warning ?

```
1 ggplot(data = starwars, mapping = aes(x = height, y = mass)) +  
2   geom_point() +  
3   labs(title = "Masse vs. taille des personnages Starwars",  
4       x = "Taille (cm)", y = "Masse (kg)")
```

Hello ggplot2 !

ggplot() est la fonction principale dans ggplot2. Elle est toujours présente.

Les graphes sont faits en couches. Nous pouvons les résumer de la manière suivante:

```
ggplot(data = [dataset],  
       mapping = aes(x = [x-variable], y = [y-variable])) +  
  geom_xxx() +  
  other options
```

Pourquoi visualiser ?

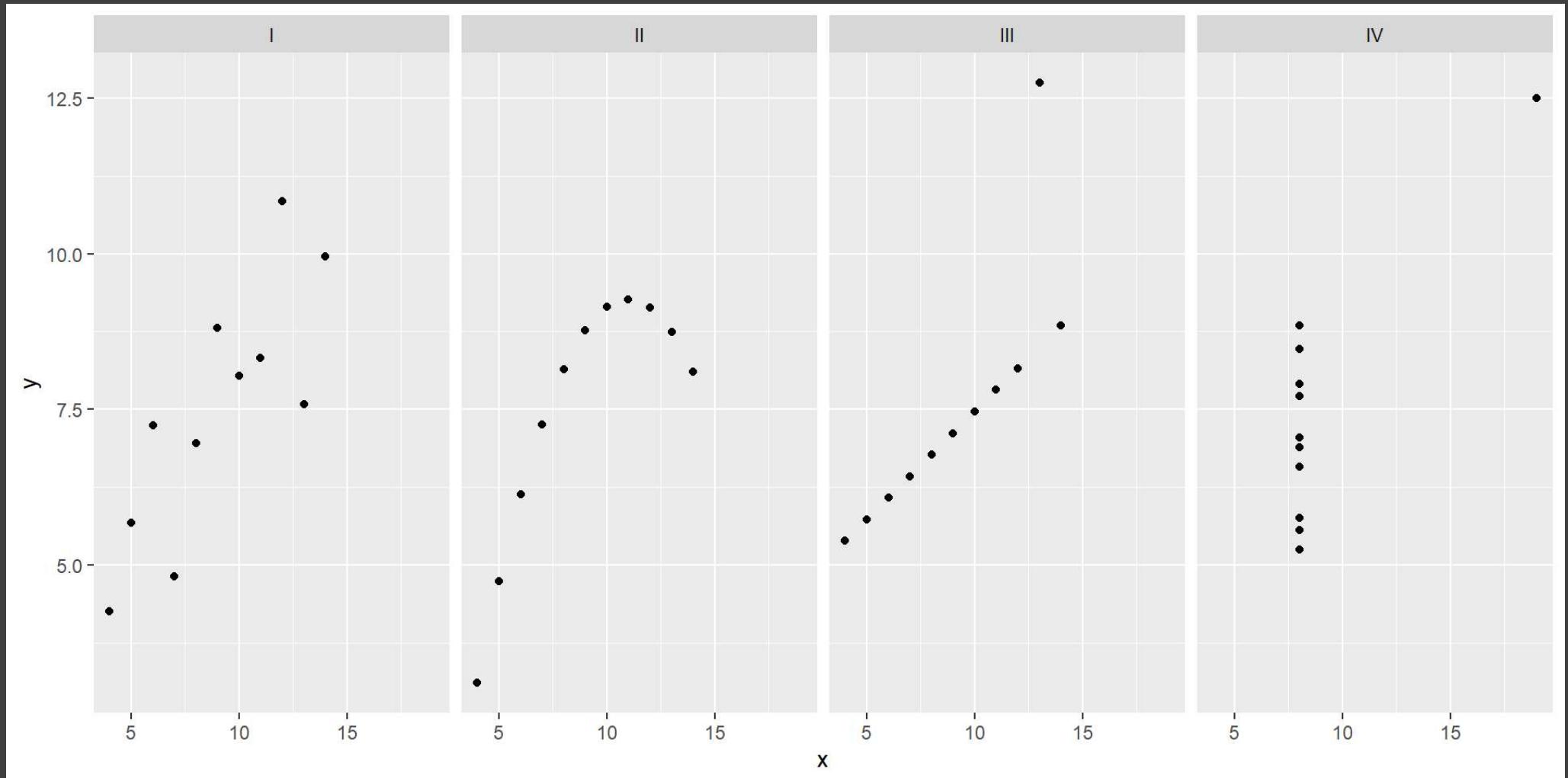
Le quartet d'Anscombe

	set	x	y		set	x	y	
1	I	10	8.04		12	II	10	9.14
2	I	8	6.95		13	II	8	8.14
3	I	13	7.58		14	II	13	8.74
4	I	9	8.81		15	II	9	8.77
5	I	11	8.33		16	II	11	9.26
6	I	14	9.96		17	II	14	8.10
7	I	6	7.24		18	II	6	6.13
8	I	4	4.26		19	II	4	3.10
9	I	12	10.84		20	II	12	9.13
10	I	7	4.82		21	II	7	7.26
11	I	5	5.68		22	II	5	4.74
	set	x	y		set	x	y	
23	III	10	7.46		34	IV	8	6.58
24	III	8	6.77		35	IV	8	5.76
25	III	13	12.74		36	IV	8	7.71
26	III	9	7.11		37	IV	8	8.84
27	III	11	7.81		38	IV	8	8.47
28	III	14	8.84		39	IV	8	7.04
29	III	6	6.08		40	IV	8	5.25
30	III	4	5.39		41	IV	19	12.50
31	III	12	8.15		42	IV	8	5.56
32	III	7	6.42		43	IV	8	7.91
33	III	5	5.73		44	IV	8	6.89

Descrivons chaque jeu de données

```
1 quartet %>%
2   group_by(set) %>%
3   summarise(
4     mean_x = mean(x),
5     mean_y = mean(y),
6     sd_x = sd(x),
7     sd_y = sd(y),
8     r = cor(x, y)
9   )
# A tibble: 4 × 6
  set    mean_x  mean_y   sd_x   sd_y      r
  <fct>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
1 I        9     7.50   3.32   2.03  0.816
2 II       9     7.50   3.32   2.03  0.816
3 III      9     7.5    3.32   2.03  0.816
4 IV       9     7.50   3.32   2.03  0.817
```

Maintenant visualisons !



Let's Dive in !

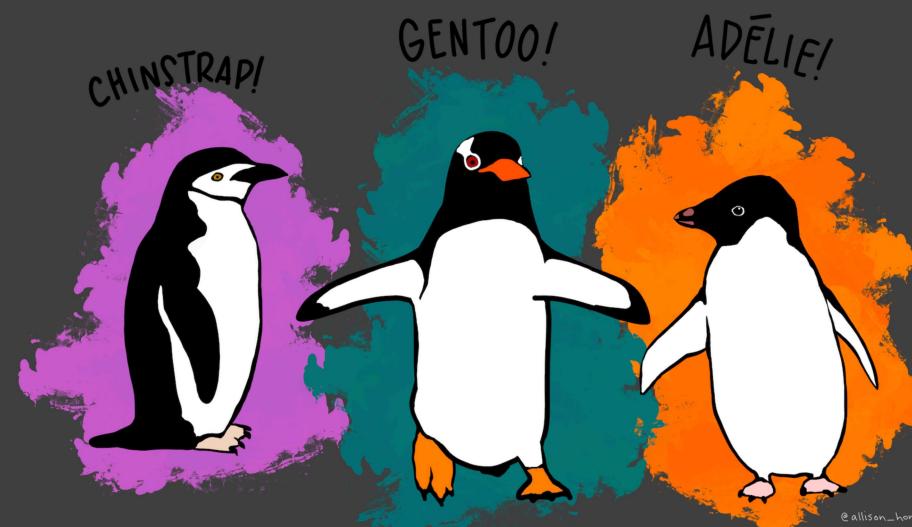
Dataset: Palmer Penguins

```

1 library(palmerpenguins)
2 glimpse(penguins)

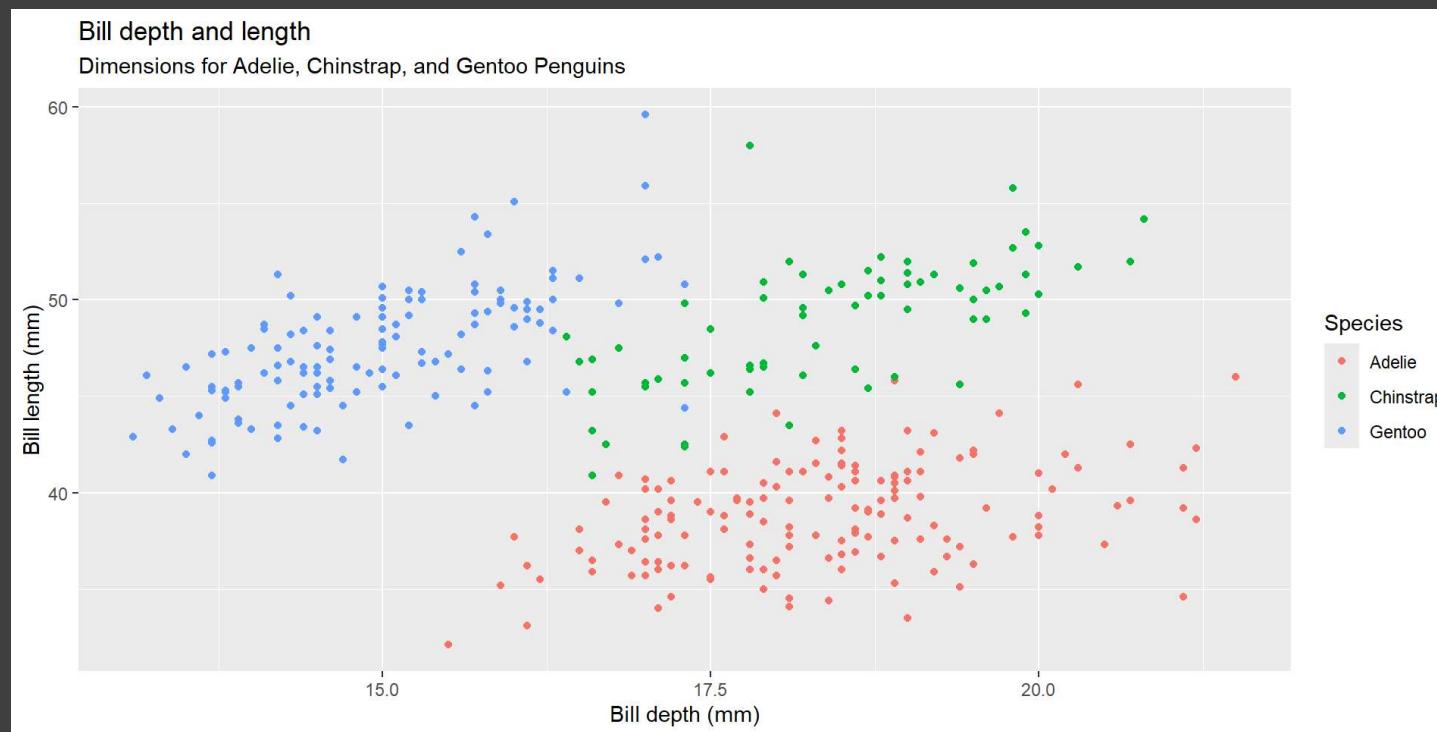
Rows: 344
Columns: 8
$ species           <fct> Adelie, Adelie, Adelie, Adelie, Adelie, Adel...
$ island            <fct> Torgersen, Torgersen, Torgersen, Torgersen, Torgers...
$ bill_length_mm    <dbl> 39.1, 39.5, 40.3, NA, 36.7, 39.3, 38.9, 39.2, 34.1, ...
$ bill_depth_mm     <dbl> 18.7, 17.4, 18.0, NA, 19.3, 20.6, 17.8, 19.6, 18.1, ...
$ flipper_length_mm <int> 181, 186, 195, NA, 193, 190, 181, 195, 193, 190, 186...
$ body_mass_g       <int> 3750, 3800, 3250, NA, 3450, 3650, 3625, 4675, 3475, ...
$ sex               <fct> male, female, female, NA, female, male, female, male...
$ year              <int> 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007, 2007...

```



[source](#)

Graphe Code

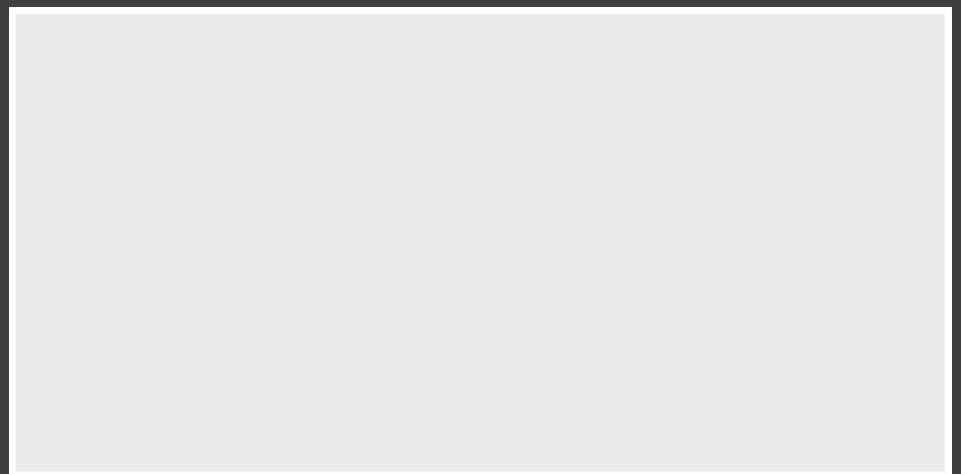


En détails

Détaillons tout ça ! (1)

| On commence avec le dataset `penguins`

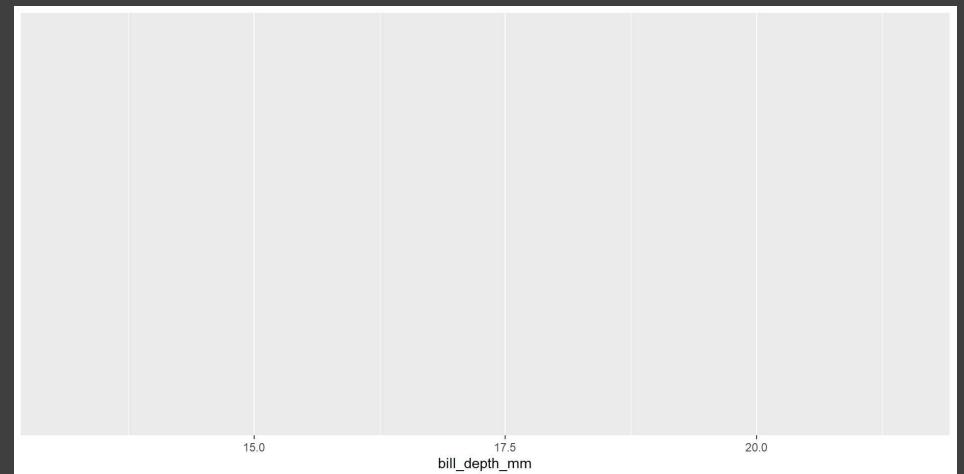
```
| 1 ggplot(data = penguins)
```



Détaillons tout ça ! (2)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x

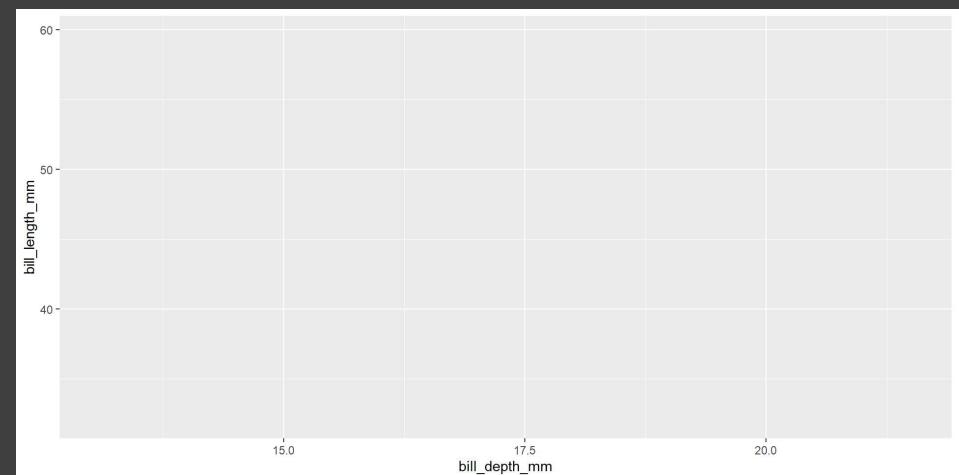
```
1 ggplot(data = penguins,  
2         mapping = aes(x = bill_depth_mm))
```



Détaillons tout ça ! (3)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y

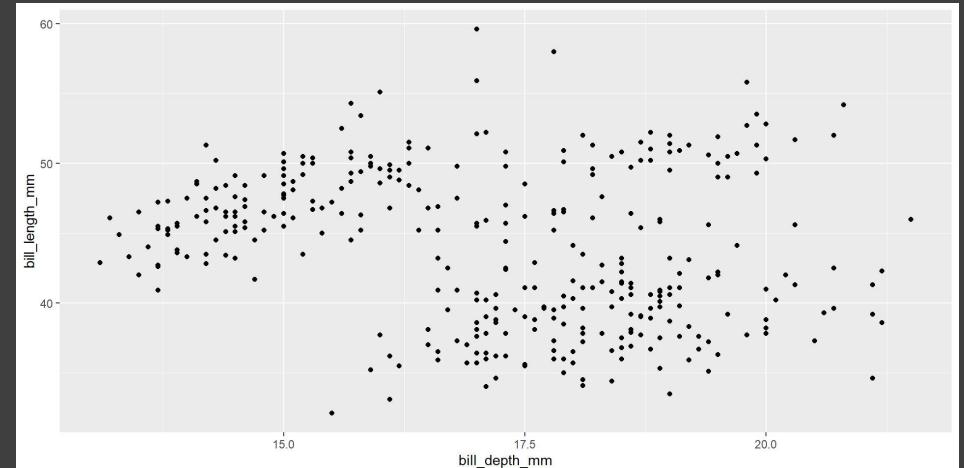
```
1 ggplot(data = penguins,  
2         mapping = aes(x = bill_depth_mm,  
3                           y = bill_length_mm))
```



Détaillons tout ça ! (4)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y. Représente chaque observation par un point

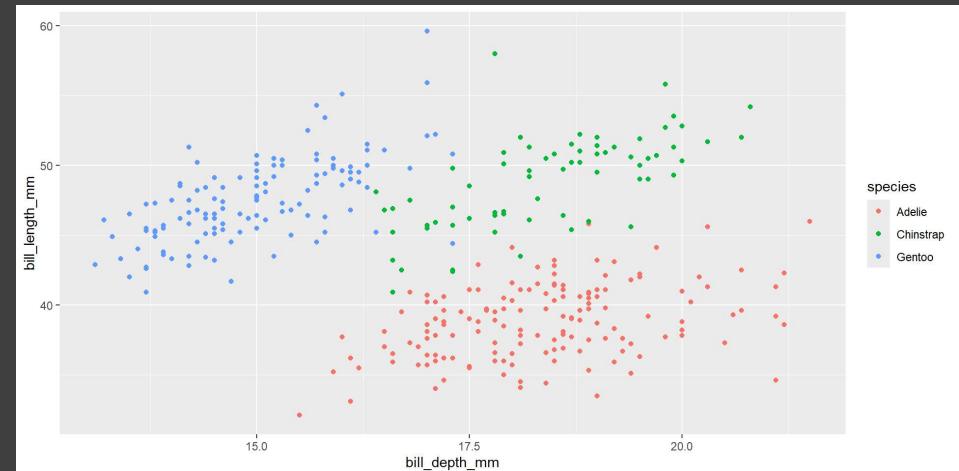
```
1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                         y = bill_length_mm)) +
4     geom_point()
```



Détaillons tout ça ! (5)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y. Représente chaque observation par un point et map la couleur des points en fonction de l'espèce

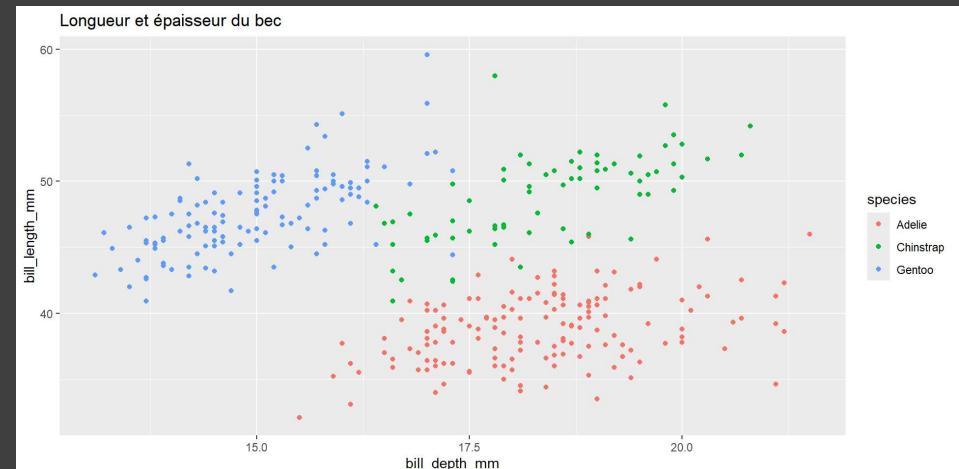
```
1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                          y = bill_length_mm,
4                          colour = species)) +
5   geom_point()
```



Détaillons tout ça ! (6)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y. Représente chaque observation par un point et map la couleur des points en fonction de l'espèce. Ajoute comme titre "Longueur et épaisseur du bec"

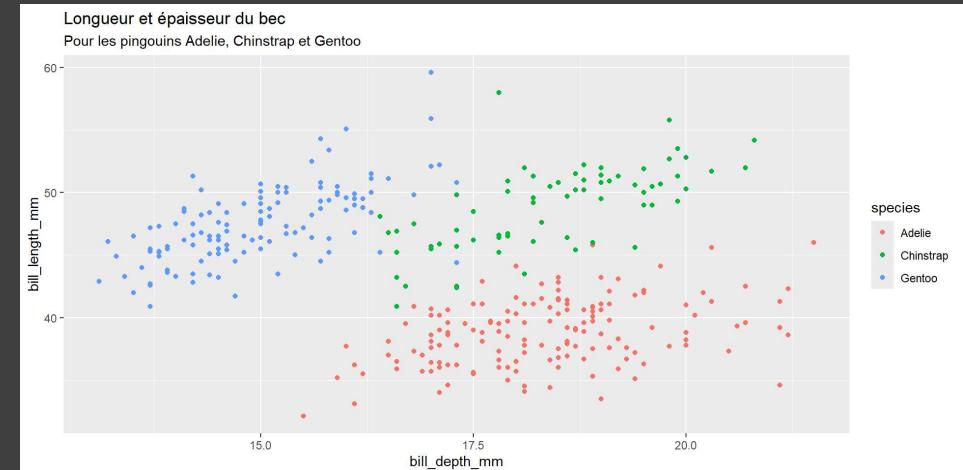
```
1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                          y = bill_length_mm,
4                          colour = species)) +
5   geom_point() +
6   labs(title = "Longueur et épaisseur du bec")
```



Détaillons tout ça ! (7)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y. Représente chaque observation par un point et map la couleur des points en fonction de l'espèce. Ajoute comme titre "Longueur et épaisseur du bec", et comme sous titre "Pour les pingouins Adelie, Chinstrap et Gentoo".

```
1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                          y = bill_length_mm,
4                          colour = species)) +
5   geom_point() +
6   labs(title = "Longueur et épaisseur du bec",
7        subtitle = "Pour les pingouins Adelie,"
```



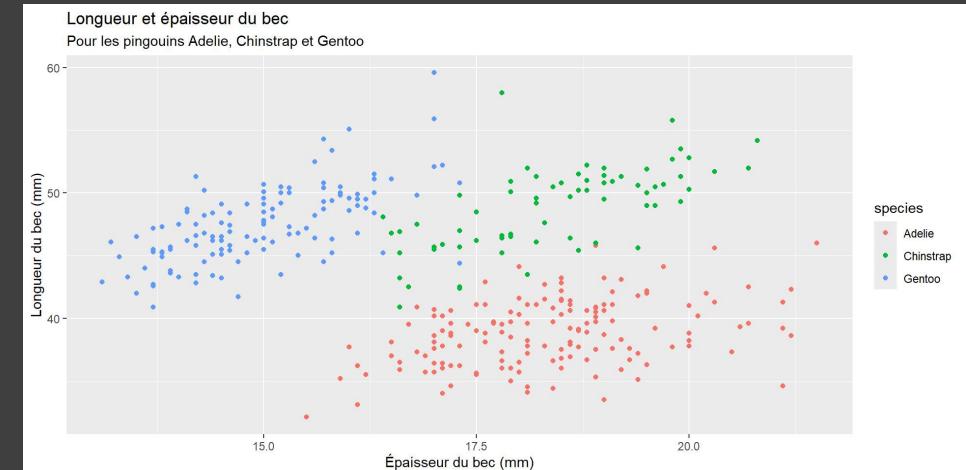
Détaillons tout ça ! (8)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y. Représente chaque observation par un point et map la couleur des points en fonction de l'espèce. Ajoute comme titre "Longueur et épaisseur du bec", et comme sous titre "Pour les pingouins Adelie, Chinstrap et Gentoo". Nomme les axes x et y, "Épaisseur du bec (mm)" et "Longueur du bec (mm)".

```

1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                          y = bill_length_mm,
4                          colour = species)) +
5   geom_point() +
6   labs(title = "Longueur et épaisseur du bec",
7        subtitle = "Pour les pingouins Adelie,
8        x = "Épaisseur du bec (mm)", y = "Long"

```



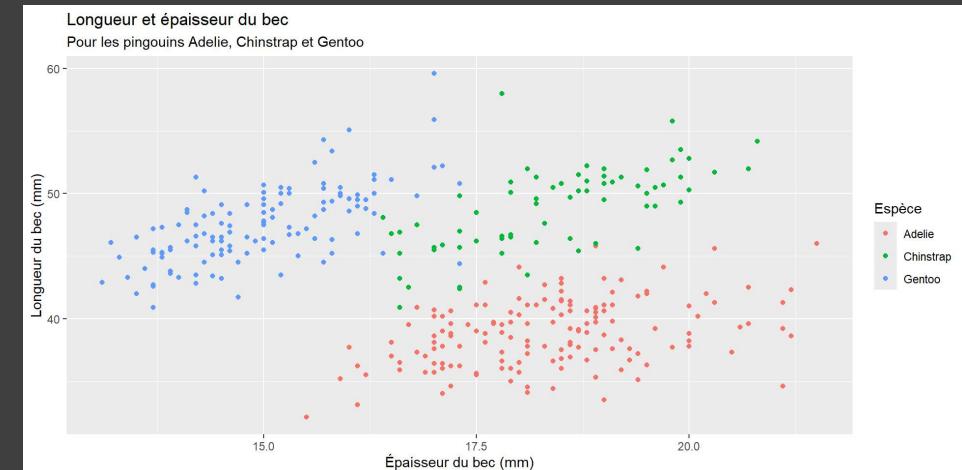
Détaillons tout ça ! (9)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y. Représente chaque observation par un point et map la couleur des points en fonction de l'espèce. Ajoute comme titre “Longueur et épaisseur du bec”, et comme sous titre “Pour les pingouins Adelie, Chinstrap et Gentoo”. Nomme les axes x et y, “Épaisseur du bec (mm)” et “Longueur du bec (mm)”, et la légende “Espèce”.

```

1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                          y = bill_length_mm,
4                          colour = species)) +
5   geom_point() +
6   labs(title = "Longueur et épaisseur du bec",
7        subtitle = "Pour les pingouins Adelie,
8        x = "Épaisseur du bec (mm)", y = "Long
9        colour = "Espèce")

```



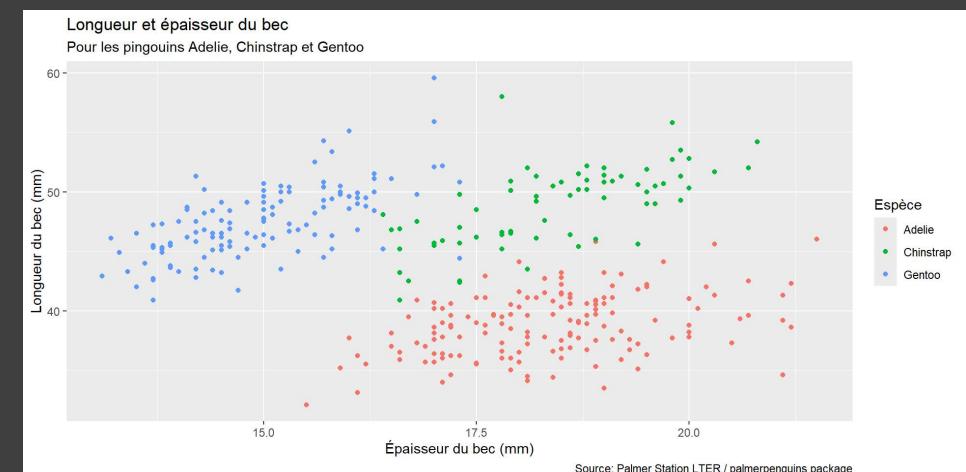
Détaillons tout ça ! (10)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y. Représente chaque observation par un point et map la couleur des points en fonction de l'espèce. Ajoute comme titre “Longueur et épaisseur du bec”, et comme sous titre “Pour les pingouins Adelie, Chinstrap et Gentoo”. Nomme les axes x et y, “Épaisseur du bec (mm)” et “Longueur du bec (mm)”, et la légende “Espèce”. Ajoute une phrase pour indiquer la source des données.

```

1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                         y = bill_length_mm,
4                         colour = species)) +
5   geom_point() +
6   labs(title = "Longueur et épaisseur du bec",
7        subtitle = "Pour les pingouins Adelie,
8        x = "Épaisseur du bec (mm)", y = "Long
9        colour = "Espèce",
10       caption = "Source: Palmer Station LTER"

```



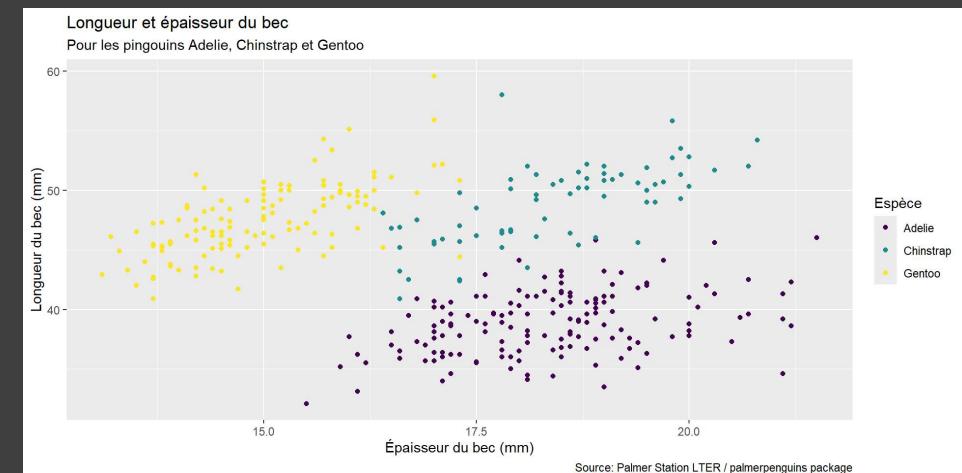
Détaillons tout ça ! (11)

On commence avec le dataset penguins, on map l'épaisseur du bec sur l'axe x, et la longueur du bec sur l'axe y. Représente chaque observation par un point et map la couleur des points en fonction de l'espèce. Ajoute comme titre “Longueur et épaisseur du bec”, et comme sous titre “Pour les pingouins Adelie, Chinstrap et Gentoo”. Nomme les axes x et y, “Épaisseur du bec (mm)” et “Longueur du bec (mm)”, et la légende “Espèce”. Ajoute une phrase pour indiquer la source des données. Pour finir, utilise une échelle de couleur adaptée aux personnes daltoniennes.

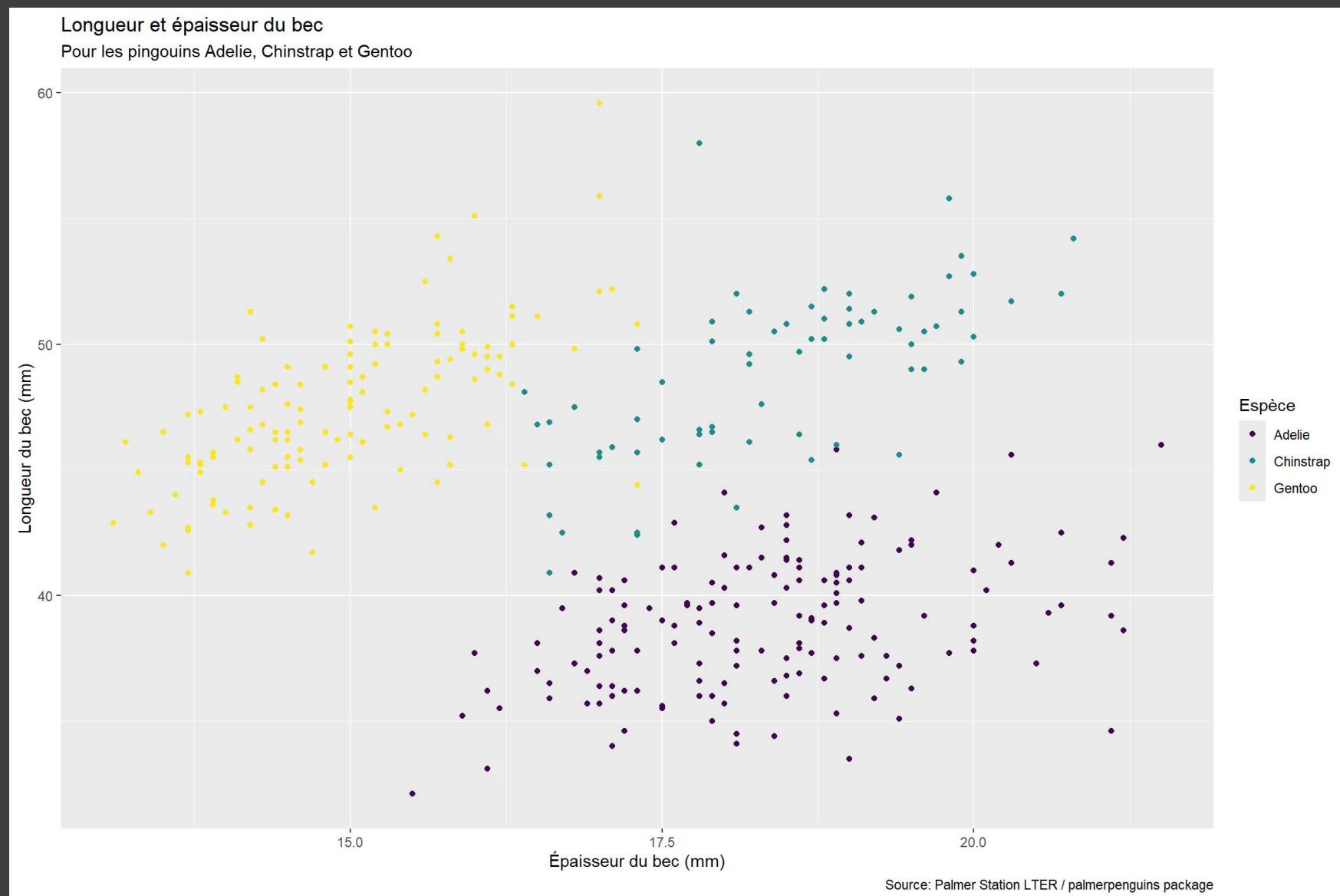
```

1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                         y = bill_length_mm,
4                         colour = species)) +
5   geom_point() +
6   labs(title = "Longueur et épaisseur du bec",
7        subtitle = "Pour les pingouins Adelie,
8        x = "Épaisseur du bec (mm)", y = "Long
9        colour = "Espèce",
10       caption = "Source: Palmer Station LTER
11       scale_colour_viridis_d()

```



Graphe Code



Arguments

Les deux premiers arguments, `data` et `mapping` peuvent être écrits directement pour alléger la notation

```
1 ggplot(data = penguins,  
2         mapping = aes(x = bill_depth_mm,  
3                           y = bill_length_mm,  
4                           colour = species)) +  
5   geom_point() +  
6   scale_colour_viridis_d()
```

```
1 ggplot(penguins,  
2         aes(x = bill_depth_mm,  
3                 y = bill_length_mm,  
4                 colour = species)) +  
5   geom_point() +  
6   scale_colour_viridis_d()
```

Aesthetics

Options

Les caractéristiques principales du graphe peuvent être **envoyées directement sur des variables** dans le jeu de données:

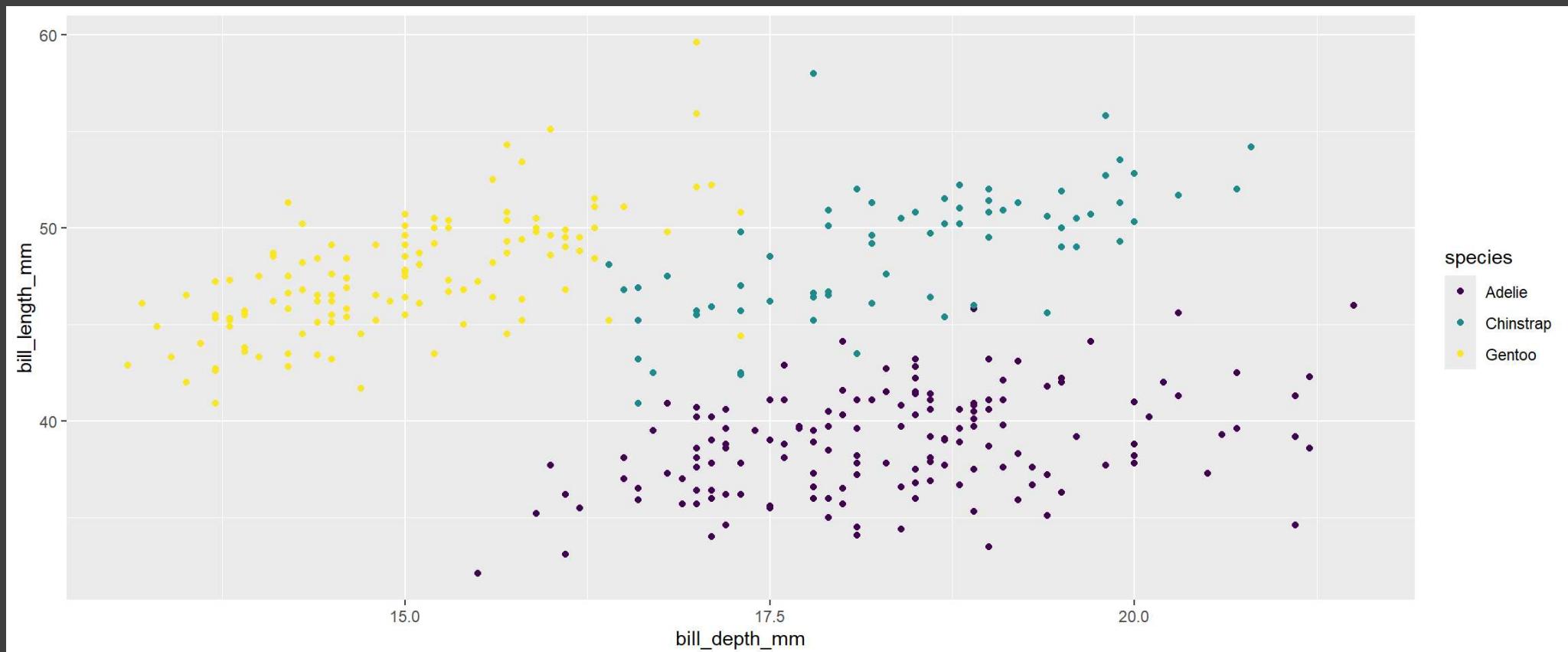
- colour
- shape
- size
- alpha

Colour

```

1 ggplot(data = penguins,
2         mapping = aes(x = bill_depth_mm,
3                          y = bill_length_mm,
4                          colour = species)) +
5   geom_point() +
6   scale_colour_viridis_d()

```

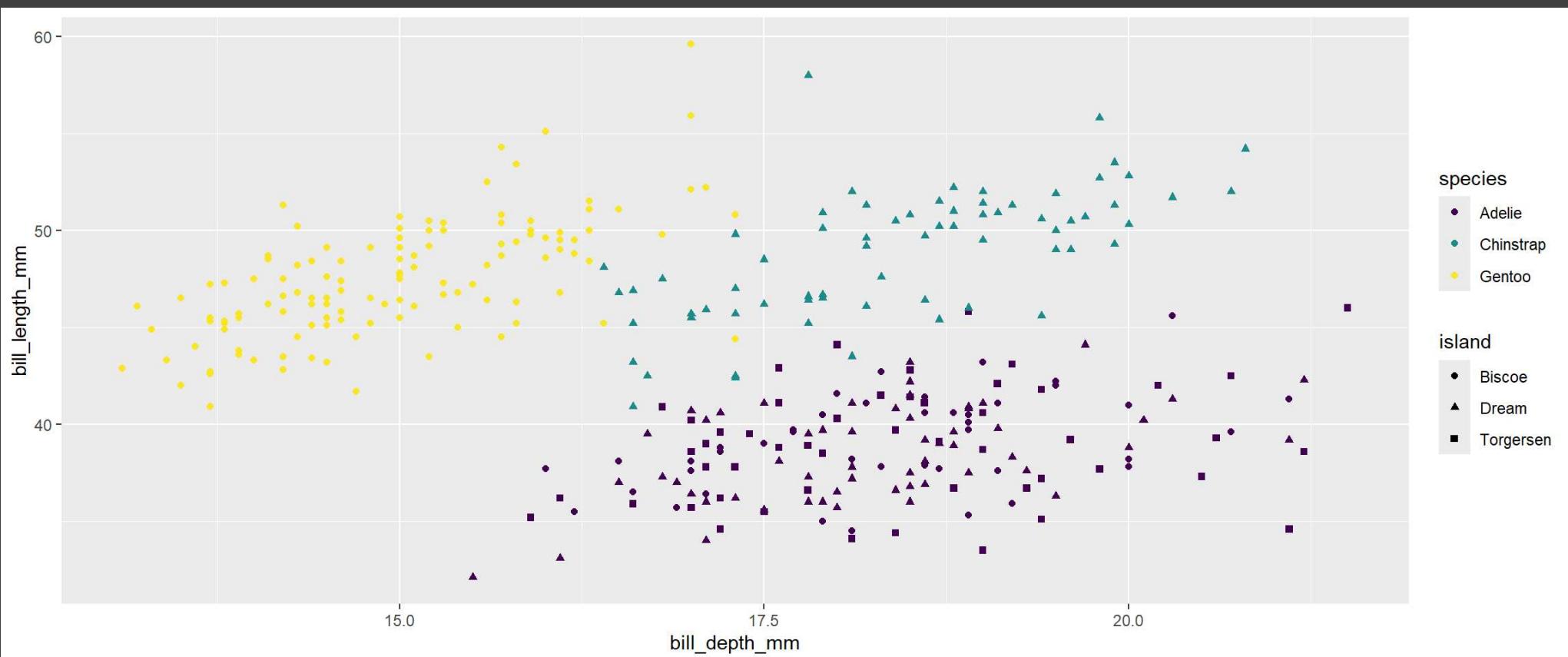


Shape

```

1 ggplot(penguins,
2   aes(x = bill_depth_mm,
3     y = bill_length_mm,
4     colour = species,
5     shape = island)) +
6   geom_point() +
7   scale_colour_viridis_d()

```

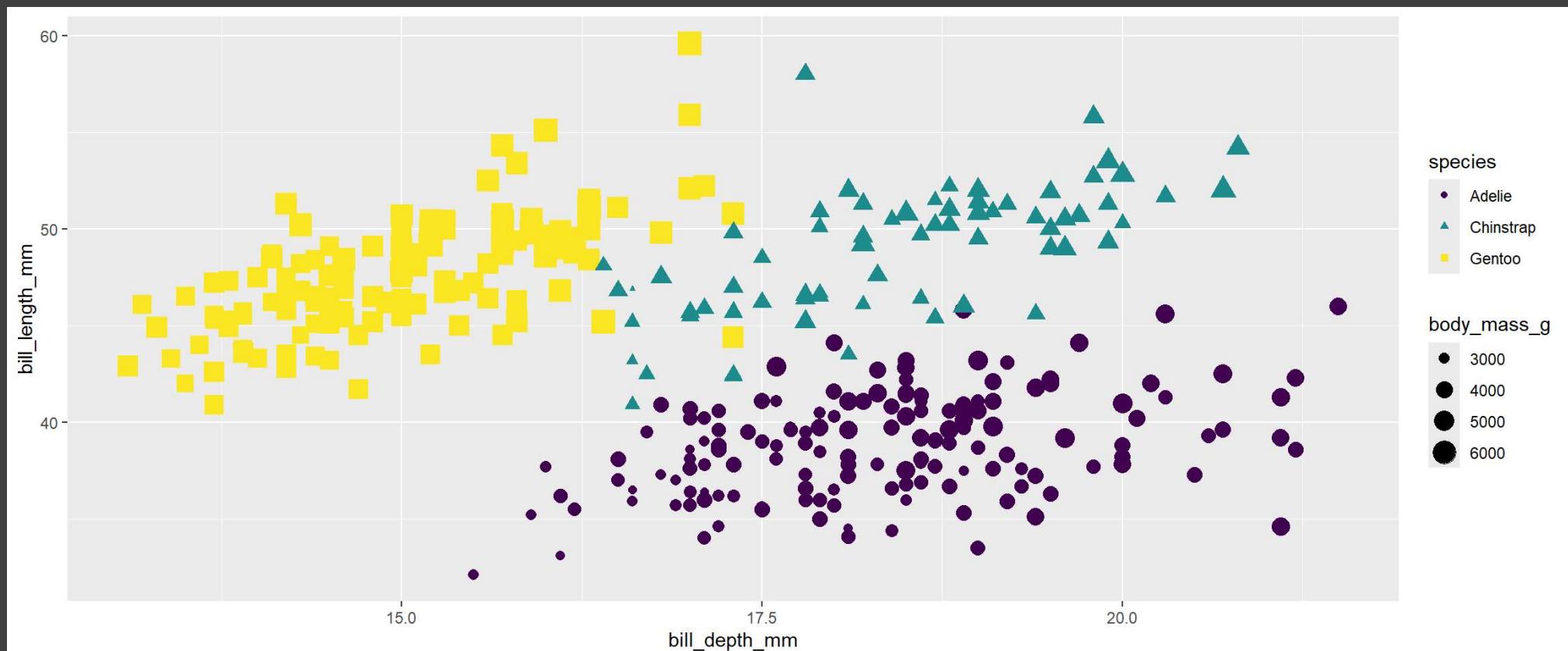


Size

```

1 ggplot(penguins,
2   aes(x = bill_depth_mm,
3       y = bill_length_mm,
4       colour = species,
5       shape = species,
6       size = body_mass_g)) +
7   geom_point() +
8   scale_colour_viridis_d()

```

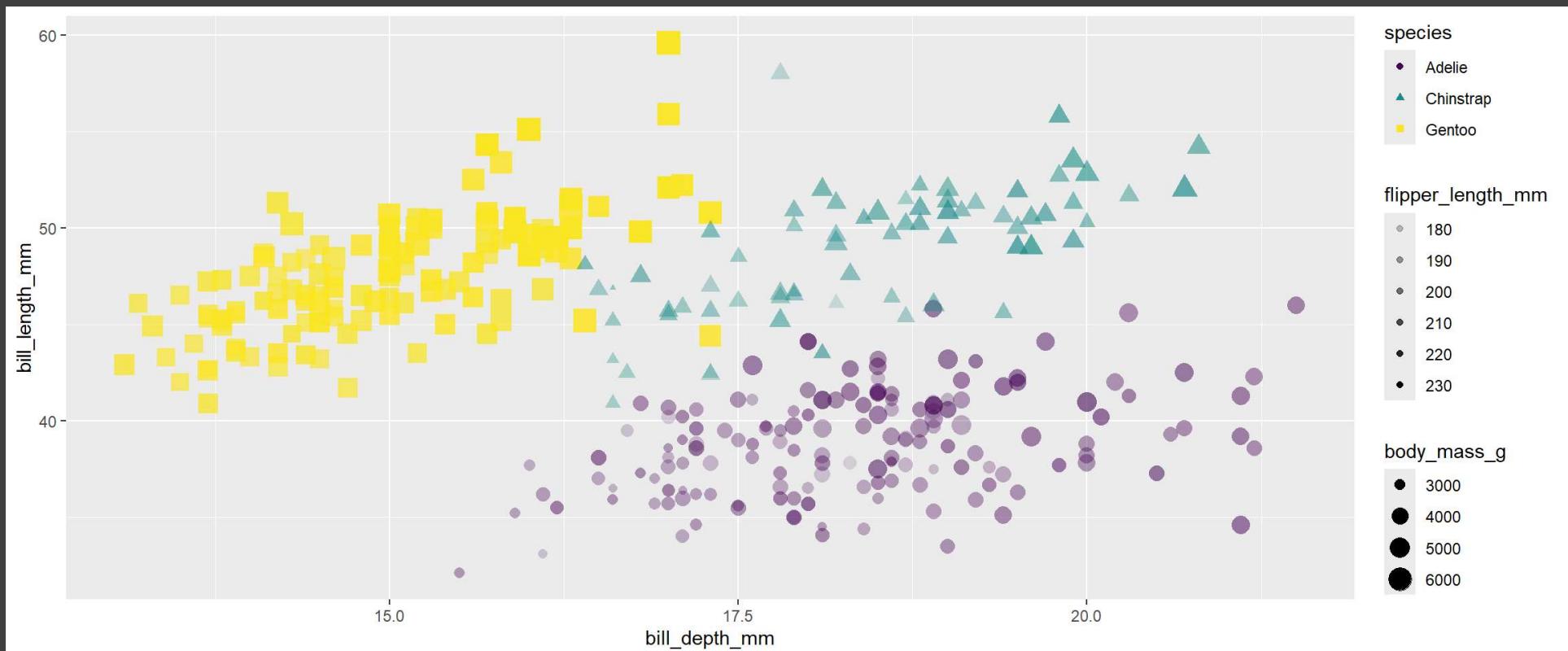


Alpha

```

1 ggplot(penguins,
2   aes(x = bill_depth_mm,
3       y = bill_length_mm,
4       colour = species,
5       shape = species,
6       size = body_mass_g,
7       alpha = flipper_length_mm) +
8   geom_point() +
9   scale_colour_viridis_d()

```

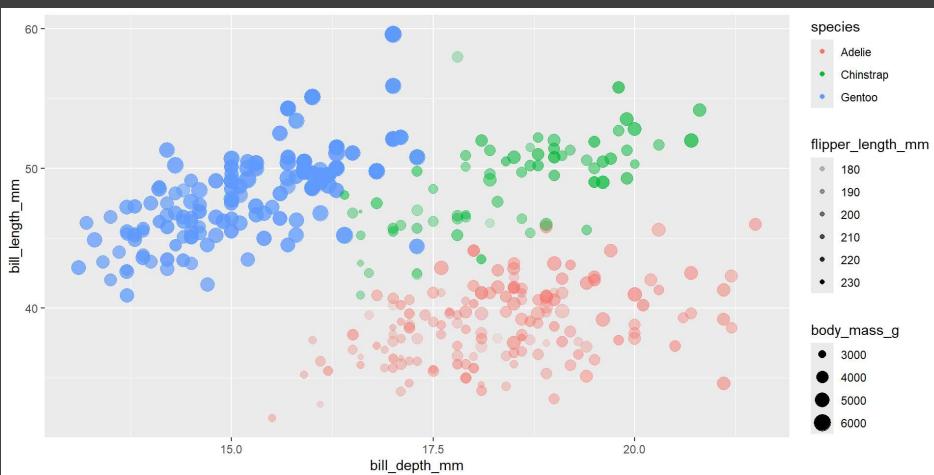


Mapping vs Setting

```

1 ggplot(penguins,
2   aes(x = bill_depth_mm,
3       y = bill_length_mm,
4       colour = species,
5       size = body_mass_g,
6       alpha = flipper_length_mm)) +
7   geom_point()

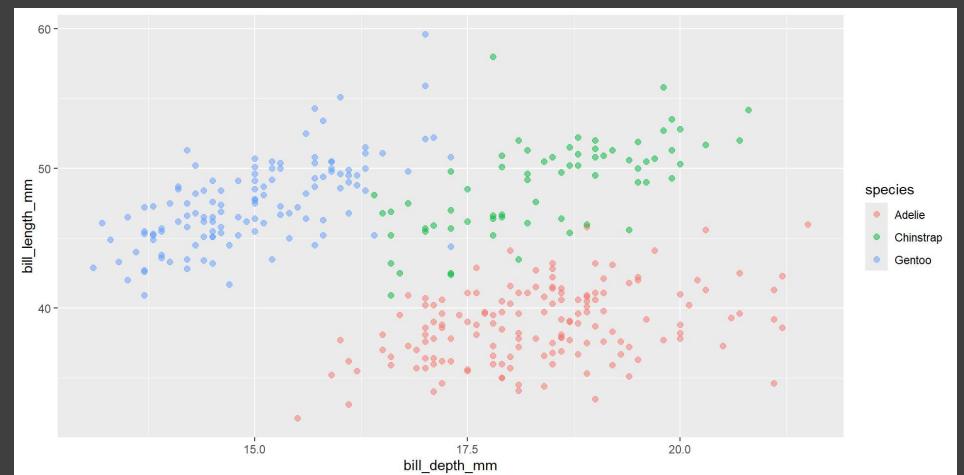
```



```

1 ggplot(penguins,
2   aes(x = bill_depth_mm,
3       y = bill_length_mm,
4       colour = species)) +
5   geom_point(size = 2, alpha = 0.5)

```



Mapping vs Setting

Mapping: Détermine la propriété (taille, alpha, forme, ...) en fonction d'une propriété des données

- Va dans la fonction `aes()`

Setting: Fixer une propriété, qui ne dépend pas d'une variable dans les données

- Va dans la fonction `geom_*`

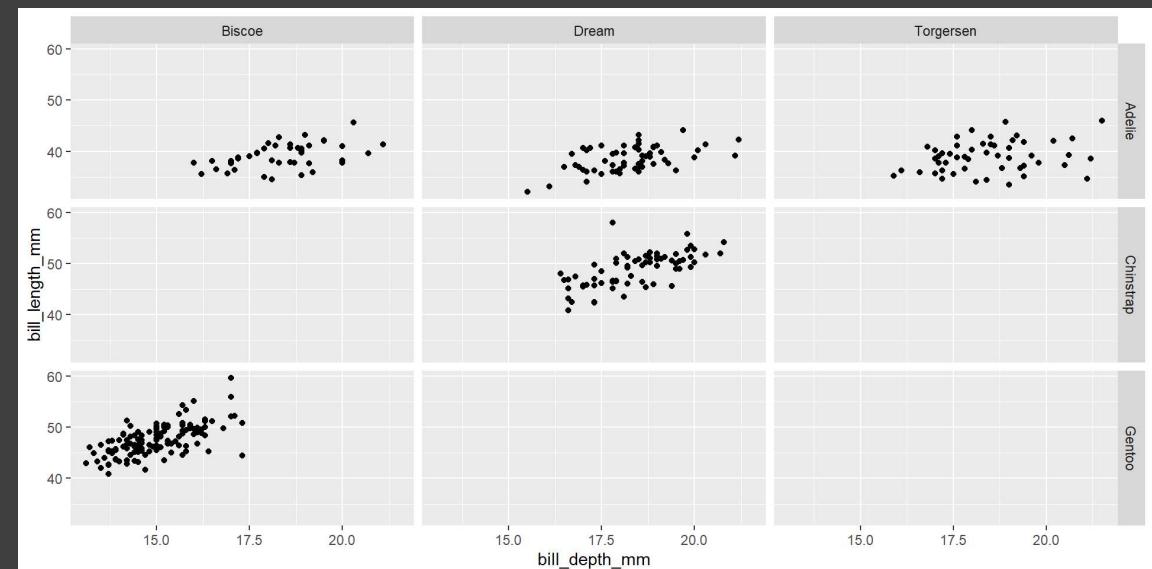
Faceting

Faceting

L'idée est de faire des sous graphes qui vont représentés des sous catégories dans les données.

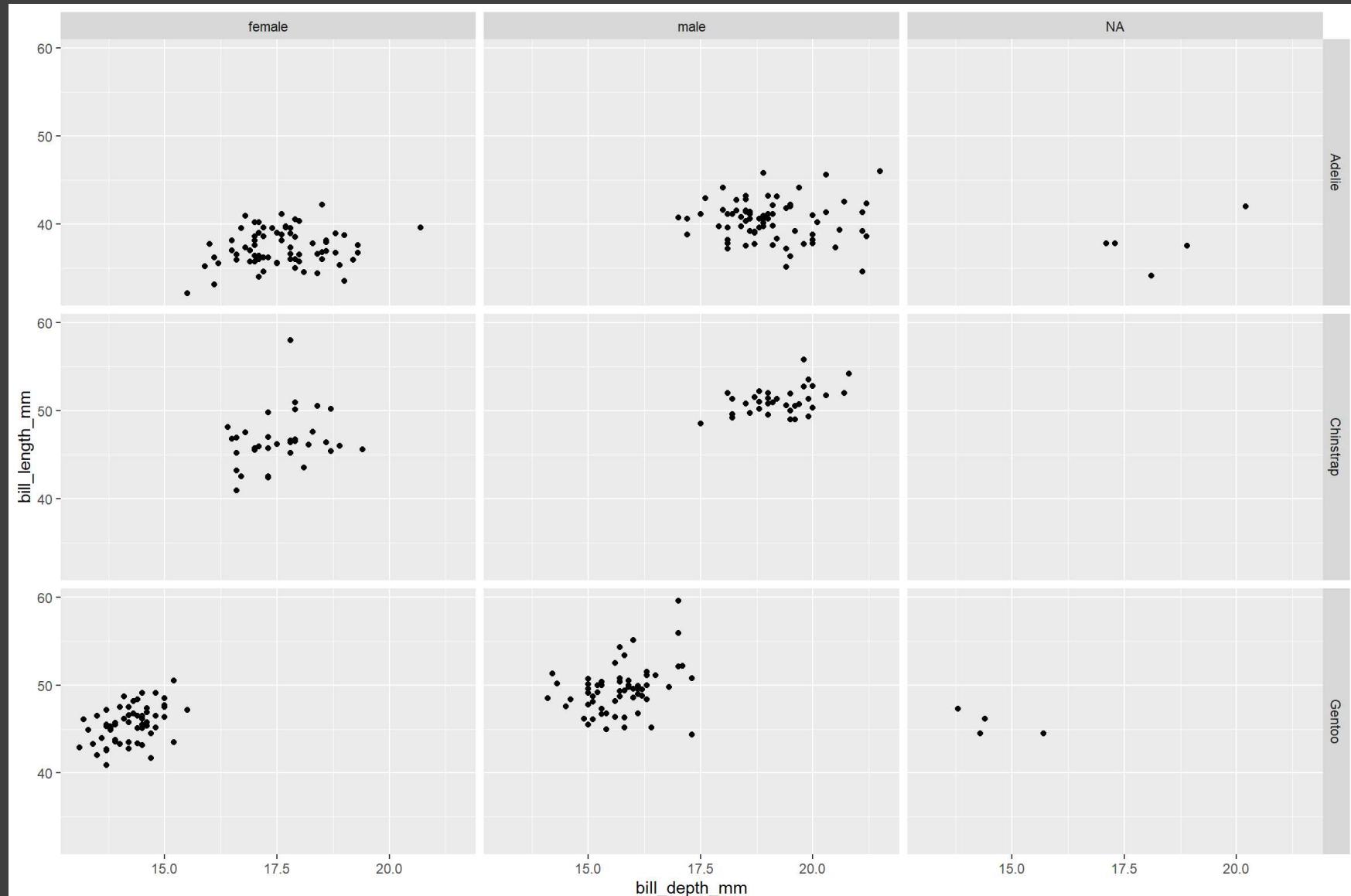
Utile pour explorer des relations conditionnelles dans les données.

```
1 ggplot(penguins,
2       aes(x = bill_depth_mm,
3             y = bill_length_mm)) +
4   geom_point() +
5   facet_grid(species ~ island)
```

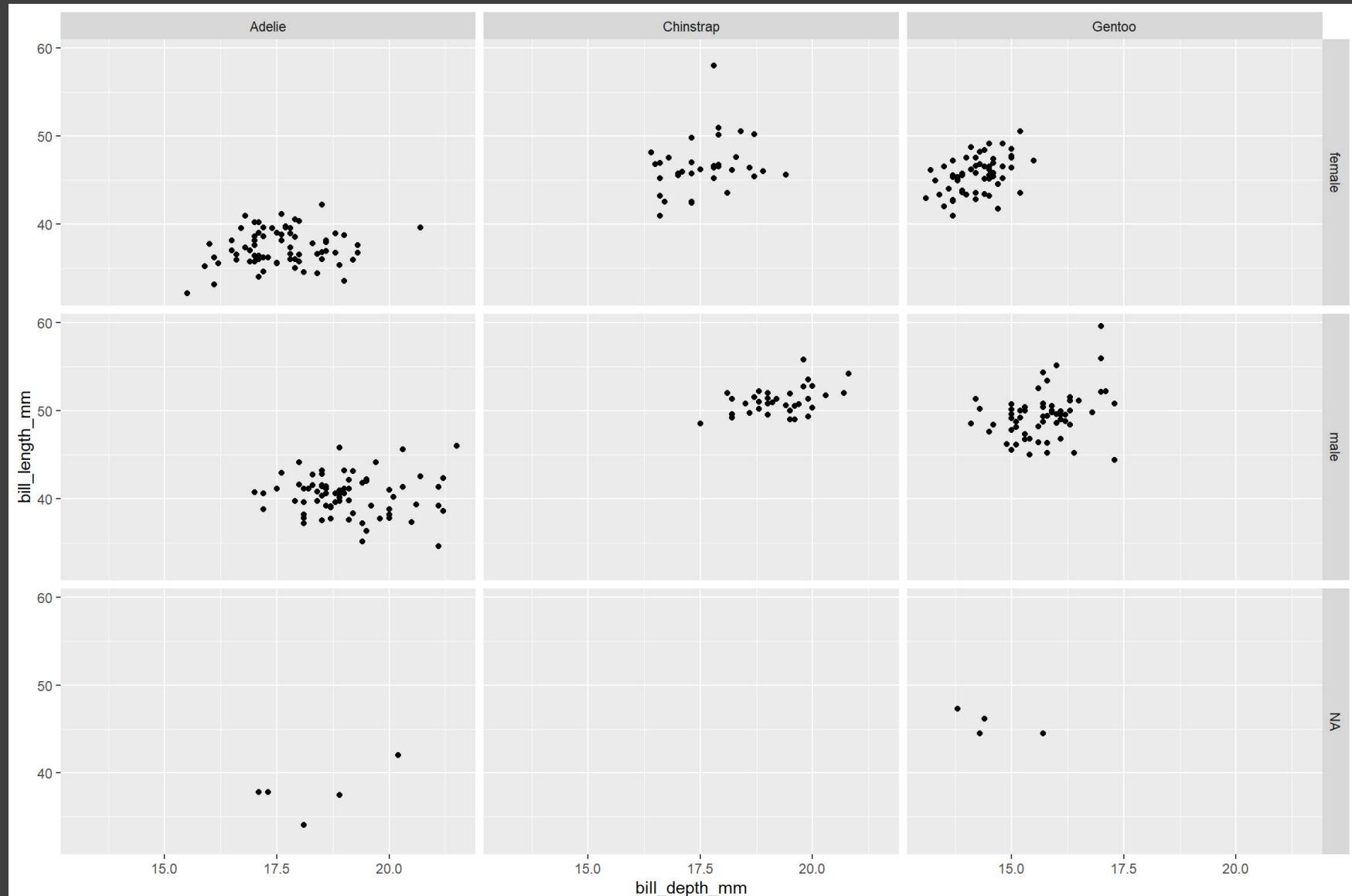


Décrivez ce qu'il se passe dans les graphes en fonction du code

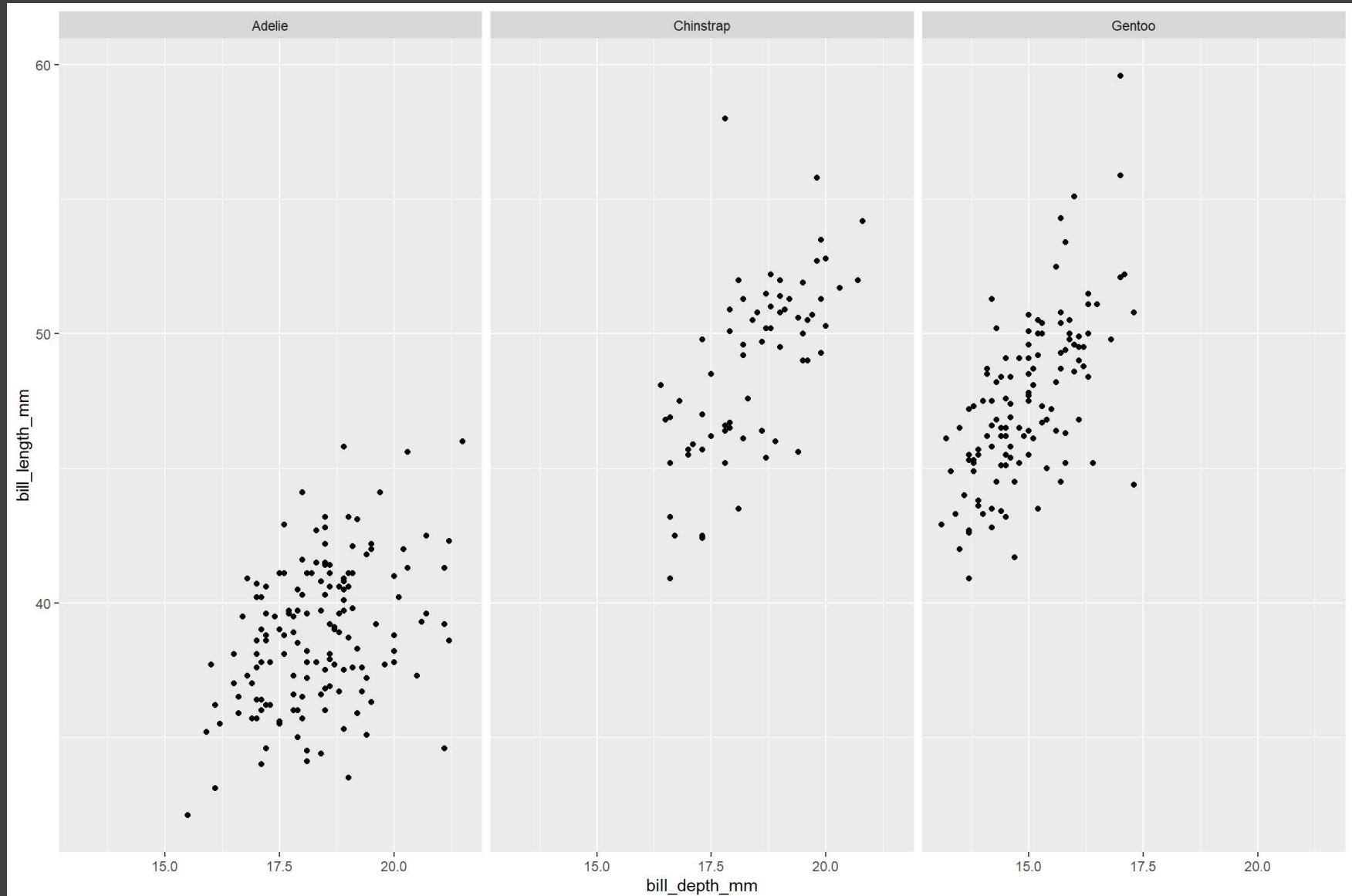
```
1 ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
2   geom_point() +
3   facet_grid(species ~ sex)
```



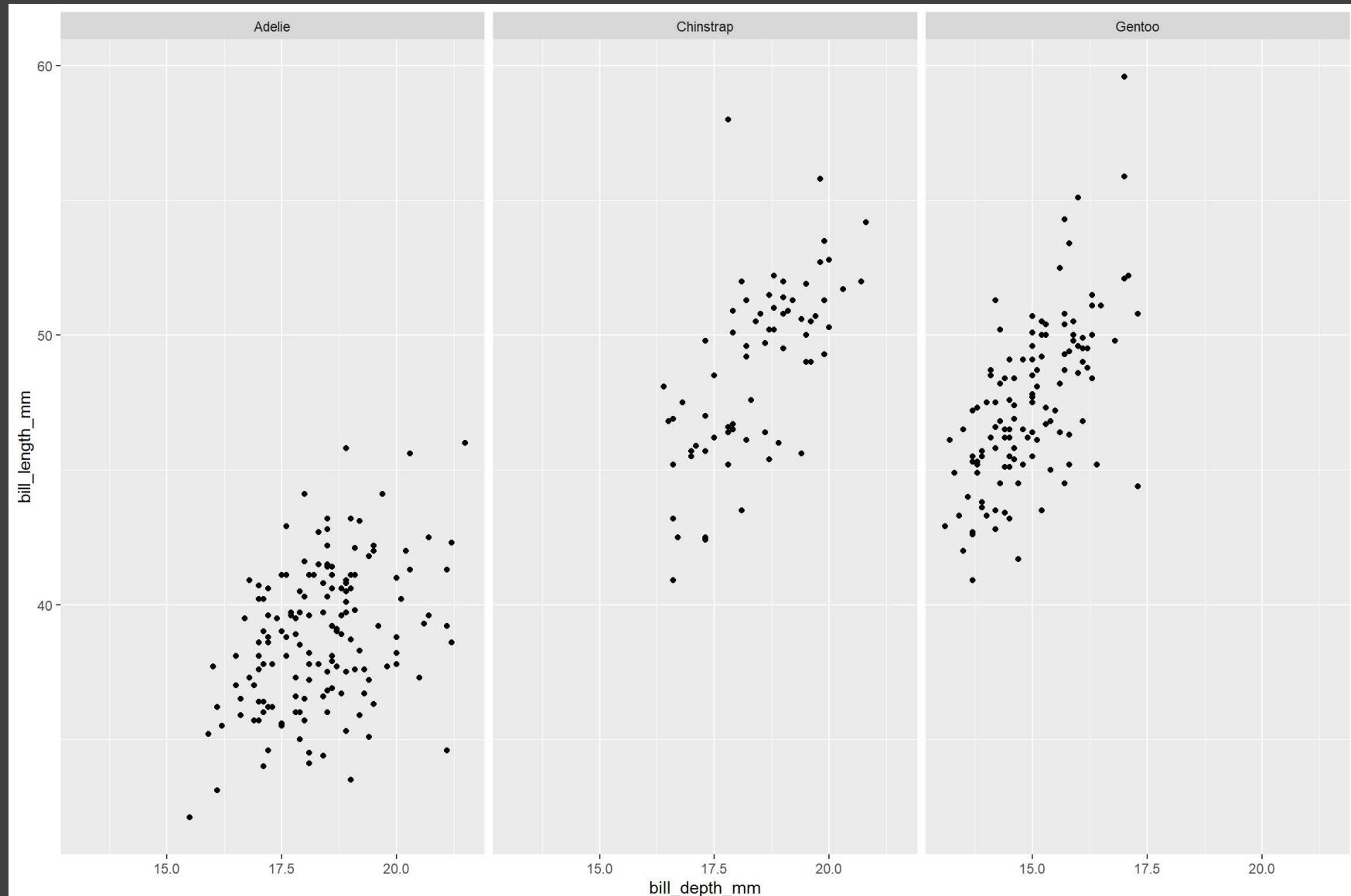
```
1 ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +  
2   geom_point() +  
3   facet_grid(sex ~ species)
```



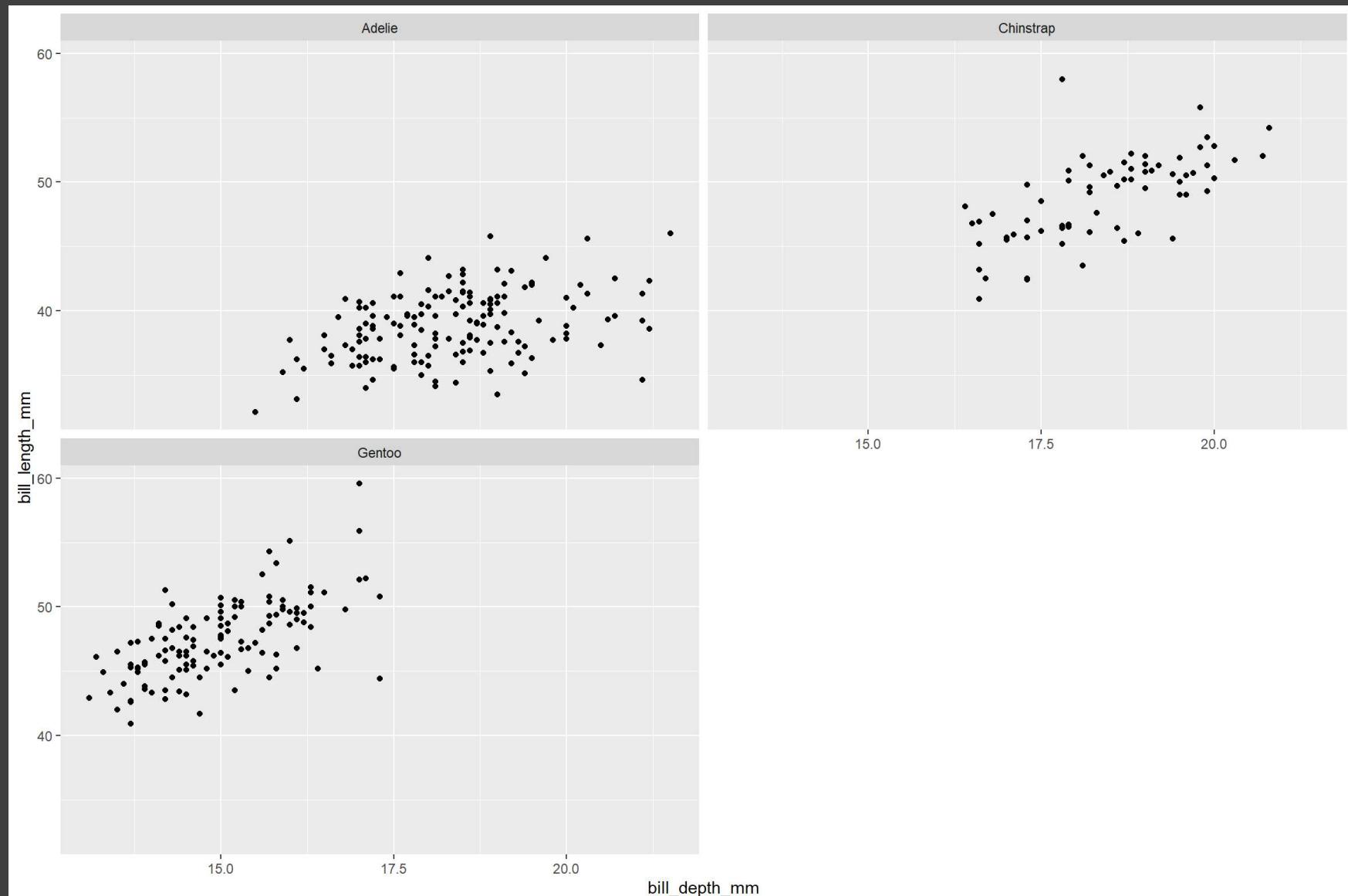
```
1 ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +  
2   geom_point() +  
3   facet_wrap(~ species)
```



```
1 ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +  
2   geom_point() +  
3   facet_grid(. ~ species)
```



```
1 ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +  
2   geom_point() +  
3   facet_wrap(~ species, ncol = 2)
```



Pour résumer

`facet_grid()`:

- Grille en 2D
- lignes ~ colonnes
- Utilisez le . pour faire du 1D

`facet_wrap()`:

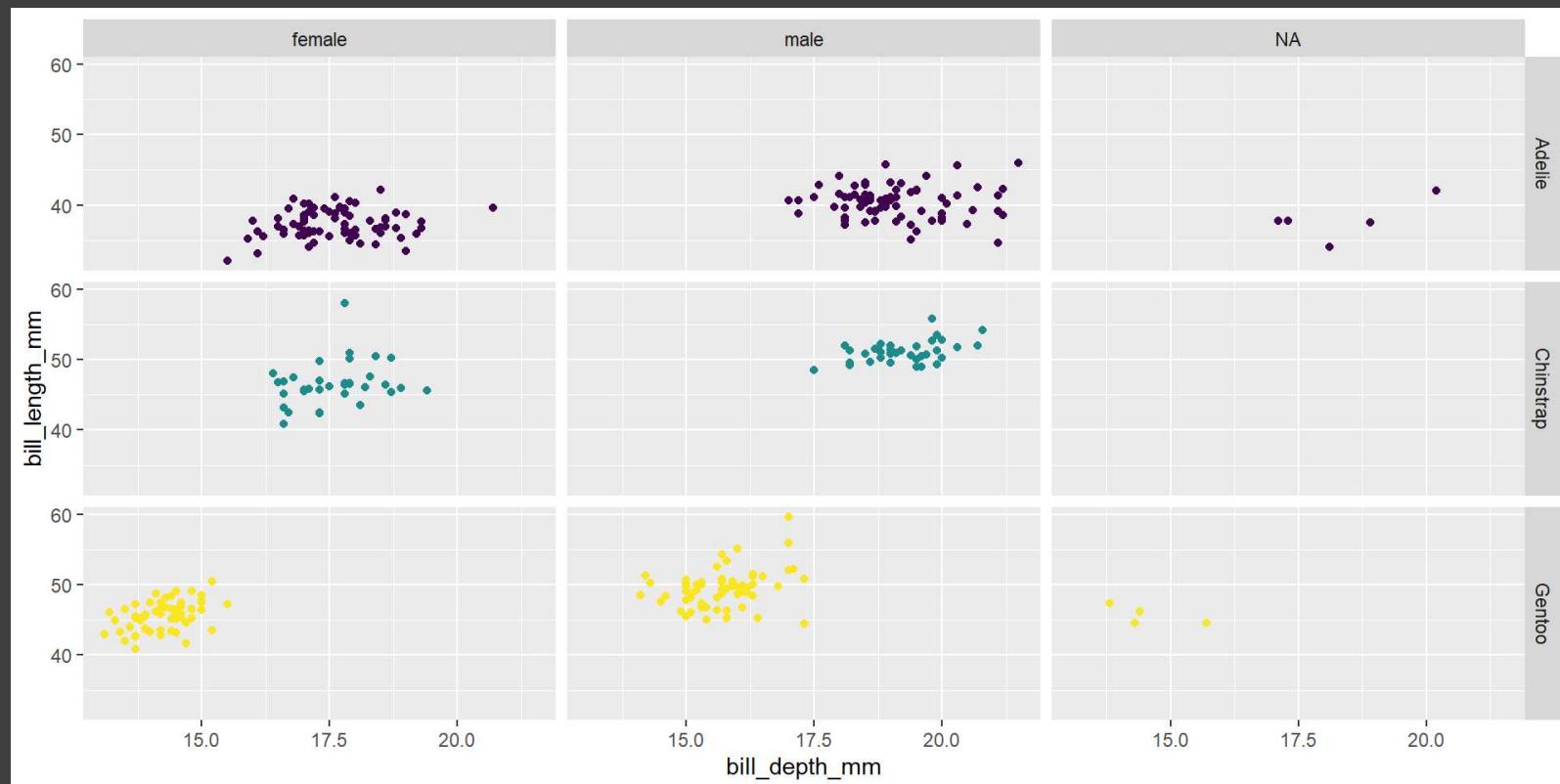
- Ruban 1D
- Déroulé selon les arguments spécifiés ou la place à disposition

Combinaison avec la couleur

```

1 ggplot(
2   penguins,
3   aes(x = bill_depth_mm,
4       y = bill_length_mm,
5       color = species)) +
6   geom_point() +
7   facet_grid(species ~ sex) +
8   scale_color_viridis_d() +
9   guides(color = "none")

```



Références

Wilkinson, L. (2005). *The Grammar of Graphics*. Springer-Verlag. <https://doi.org/10.1007/0-387-28695-0>