Guide d'introduction à la modélisation de dynamique de populations avec R

Louis Moisan

2024-08-22

Contents

1 Activation des librairies nécessaires **2**

2 Modèle de croissance exponentielle **2**
2.1 Sans stochasticité 2
2.2 Avec stochasticité environnementale 4
2.3 Avec stochasticité environnementale et démographique 6

3 Modèle de croissance avec densité-dépendance **8**
3.1 Compétition par exploitation (scramble) 8
3.2 Compétition par interférence (contest) 10
3.3 Compétition par interférence avec plafond (ceiling) 12
3.4 Compétition par exploitation (scramble) avec stochasticité 14

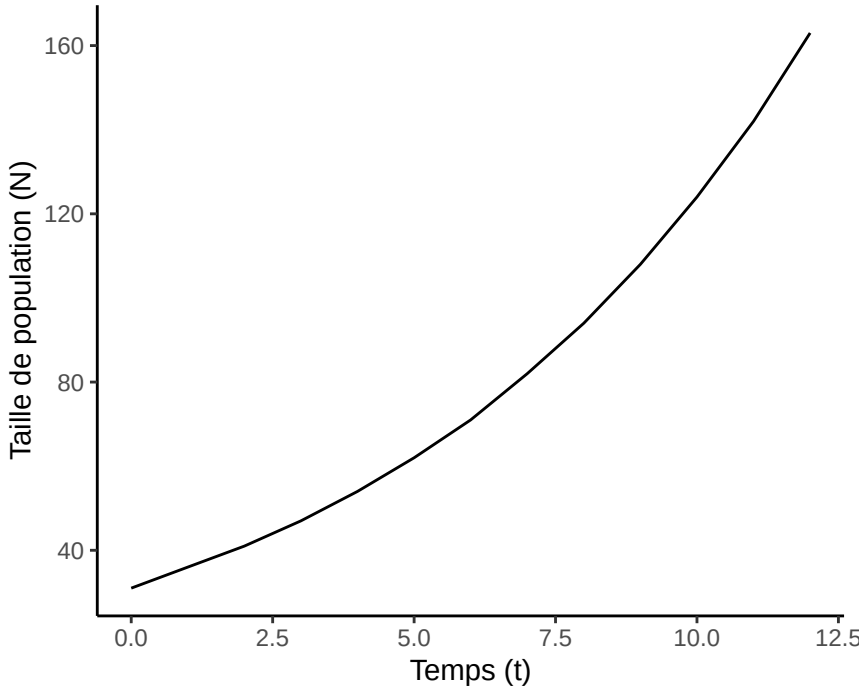
1 Activation des librairies nécessaires

```
library(stats)
library(ggplot2)
```

2 Modèle de croissance exponentielle

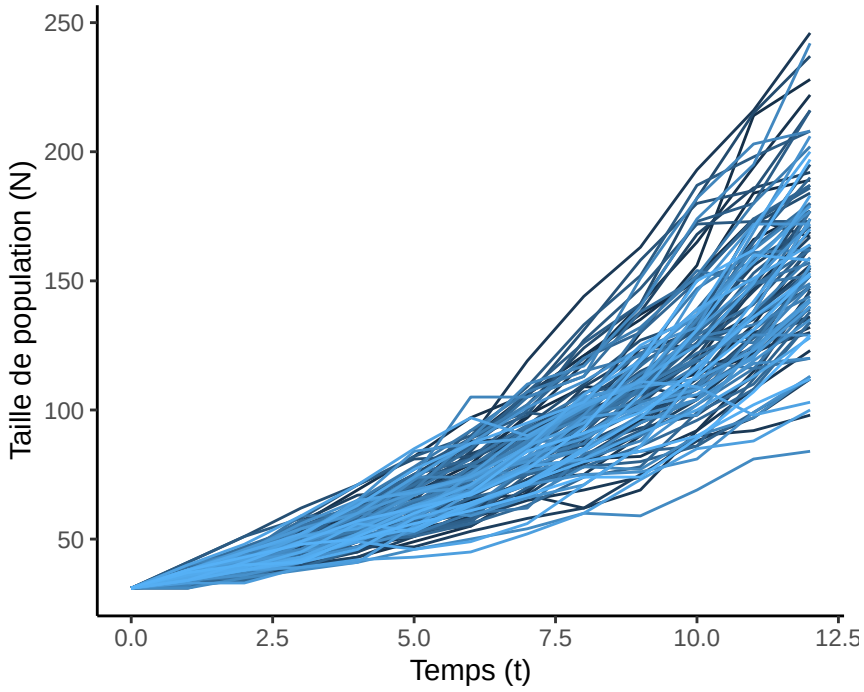
2.1 Sans stochasticité

```
#1. Déterminer les paramètres du modèle
taux_croissance= 1.148
temps=12
abondance_initiale = 31

#2. Créer un "data frame" vide pour enregistrer le résultat
df <- data.frame()

#3. Faire rouler la simulation
  #Pour chaque pas de temps
for (t in seq(0,temps)){

  #si le premier pas de temps, utiliser taille de pop. initiale
        if (t == 0) {
          taille_population <- abondance_initiale

  #sinon calculer la taille de la population avec la formule suivante (Nt+1= Nt * R)
        } else {
    taille_population <- round(taille_population * taux_croissance, digits=0)
        }

#4. Ajouter le résultat au "data frame"
  df <- rbind(df, data.frame(temps = t, taille_population = taille_population))
}

ggplot(df, aes(x = temps, y =  taille_population)) +
    geom_line() +
    labs(x = "Temps (t)", y = "Taille de population (N)") +
    theme_classic()
```


160
120
80
40
Taille de population (N)
0.0
2.5
5.0
7.5
10.0
12.5
Temps (t)

2.2 Avec stochasticité environnementale

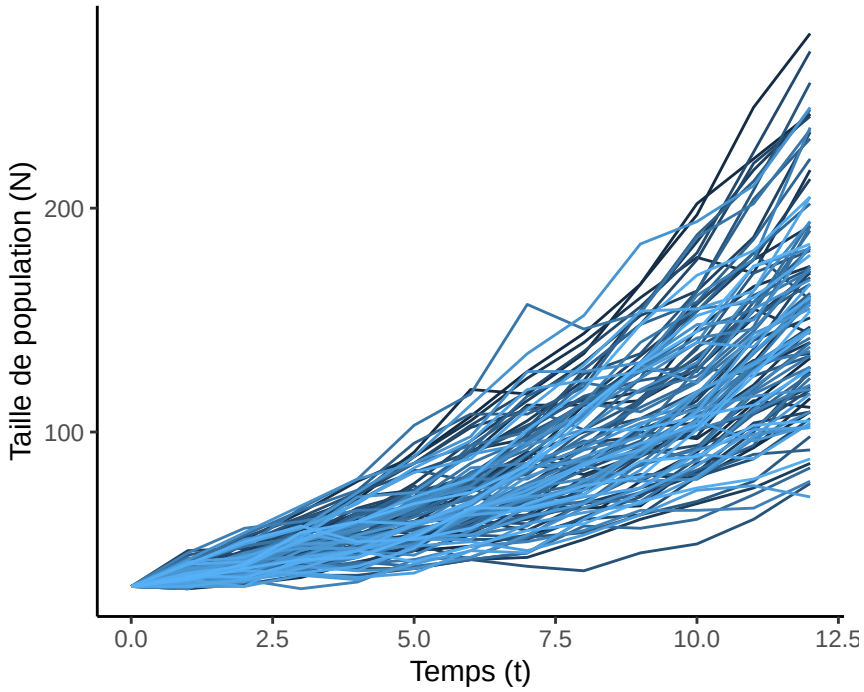
```r
#1. Déterminer les paramètres du modèle
taux_croissance= 1.148
ecart_type_R= 0.075 #Écart-type du taux de croissance (stochasticité environnementale)
temps=12
abondance_initiale = 31
nombre_replications=100

#2. Créer un "data frame" vide pour enregistrer le résultat
df <- data.frame()

#3. Faire rouler la simulation
#pour chaque réplication
for (r in 1:nombre_replications){

  #et pour chaque pas de temps
  for (t in seq(0,temps)){

  #si le premier pas de temps, utiliser taille de pop. initiale
        if (t == 0) {
          taille_population <- abondance_initiale

          #sinon
        } else {
  #Ajuster le taux de croissance moyen selon la stochasticité environnementale
  #Une nouvelle valeur de R est tirée d'une distribution normale
  #centrée sur le taux de croissance moyen et son écart-type.
  taux_croissance_stochas <-  rnorm(1, mean = taux_croissance, sd = ecart_type_R)

  #Calculer la taille de population avec la formule (Nt+1= Nt*R)
  taille_population <- round(taille_population * taux_croissance_stochas, digits=0)
        }

#4. Ajouter le résultat au "data frame"
  df <- rbind(df, data.frame(replication=r, temps = t, taille_population = taille_population))
  }
}

ggplot(df, aes(x = temps, y =  taille_population, group=replication, col=replication)) +
    geom_line() +
    labs(x = "Temps (t)", y = "Taille de population (N)") +
    theme_classic()+
    guides(col= "none")
```


250
200
150
100
50
Taille de population (N)
0.0
2.5
5.0
7.5
10.0
12.5
Temps (t)

2.3 Avec stochasticité environnementale et démographique

```
#1. Déterminer les paramètres du modèle
survie= 0.921
fecondite= 0.227
ecart_type_R= 0.075
temps=12
abondance_initiale = 31
nombre_replications=100

#2. Créer un "data frame" vide pour enregistrer le résultat
df <- data.frame()

#3. Faire rouler la simulation
#pour chaque réplication
  for (r in 1:nombre_replications){

  #et pour chaque pas de temps
    for (t in seq(0,temps)){

  #si le premier pas de temps, utiliser taille de pop. initiale
        if (t == 0) {
          taille_population <- abondance_initiale

          #sinon
        } else {

#4. Ajuster le taux de croissance selon la stochasticité environnementale
  taux_croissance_stochas <-  rnorm(1, mean = (survie+fecondite), sd = ecart_type_R)
  #Si le taux de croissance est négatif, le définir à 0
  taux_croissance_stochas <- ifelse(taux_croissance_stochas < 0, 0, taux_croissance_stochas)

  #Calculer le coefficient de changement du taux de croissance avec stochasticité
    coef_stochas_environ <- taux_croissance_stochas/taux_croissance

  #Ajuster le taux de survie proportionellement au changement du R
   survie_stochas <- survie*coef_stochas_environ
   #Si le taux de survie est négatif, le maintenir à 0
   #et si plus grand que 1, le maintenir à 1
    survie_stochas <- ifelse(survie_stochas < 0, 0, ifelse(survie_stochas > 1, 1, survie_stochas))

    #Faire la même chose pour le taux de fécondité
    fecondite_stochas<- fecondite*coef_stochas_environ
    #Toutefois, ici le taux de fécondité peut être plus grand que 1
    fecondite_stochas <- ifelse(fecondite_stochas < 0, 0, fecondite_stochas)

#5. Considérer la stochasticité démographique
  #Effectuer un tirage binomiale (0 ou 1) pour chaque individu
    #pour déterminer combien d'individus ont survécus
  survivants <- rbinom(taille_population, 1, survie_stochas)

  #Calculer ensuite combien de jeunes sont produit par individu
  #avec le taux de fécondité, on utilise ici une distribution de Poisson,
  #car certains individus peuvent faire plus qu'un jeune en une année
```

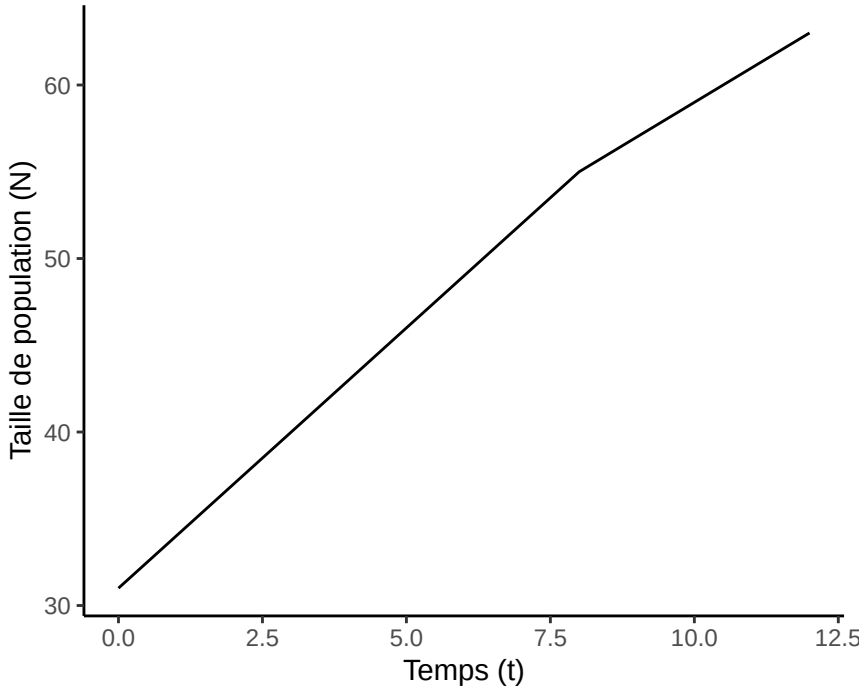
```
  jeunes <- rpois(taille_population, fecondite_stochas)

  #Faire la somme des survivants et jeunes produits
  taille_population <- sum(survivants) + sum(jeunes)
  }

#6. Ajouter le résultat au "data frame"
  df <- rbind(df, data.frame(replication=r, temps = t, taille_population = taille_population))
  }
}

ggplot(df, aes(x = temps, y =  taille_population, group=replication, col=replication)) +
    geom_line() +
    labs(x = "Temps (t)", y = "Taille de population (N)") +
    theme_classic()+
    guides(col= "none")
```



3 Modèle de croissance avec densité-dépendance

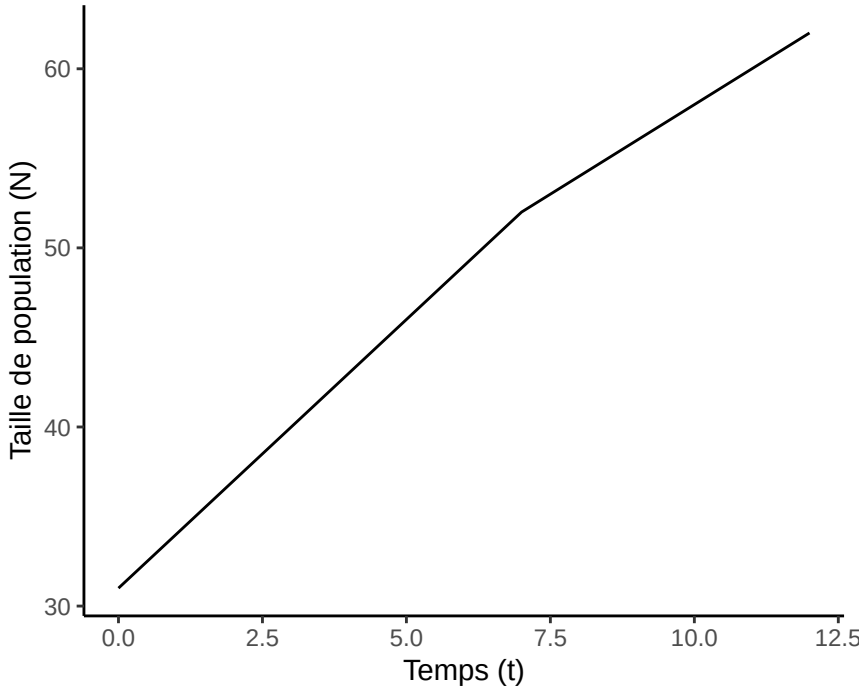
3.1 Compétition par exploitation (scramble)

```
#1. Déterminer les paramètres du modèle
taux_croissance= 1.148 #Taux de croissance maximal (faible abondance)
temps=12
abondance_initiale = 31
capacite_biotique= 80

#2. Créer un "data frame" vide pour enregistrer le résultat
df <- data.frame()

#3. Faire rouler la simulation
  #Pour chaque pas de temps
for (t in seq(0,temps)){

  #si le premier pas de temps, utiliser taille de pop. initiale
        if (t == 0) {
          taille_population <- abondance_initiale

        #sinon
        } else {
#4. Ajuster le taux de croissance selon la densité avec le modèle de Ricker
 taux_croissance_ajust <- taux_croissance^(1-(taille_population/capacite_biotique))

#5.Calculer la taille de la population avec la formule (Nt+1= Nt*R)
  taille_population <- round(taille_population * taux_croissance_ajust, digits=0)
        }

#6. Ajouter le résultat au "data frame"
  df <- rbind(df, data.frame(temps = t, taille_population = taille_population))
}

ggplot(df, aes(x = temps, y =  taille_population)) +
    geom_line() +
    labs(x = "Temps (t)", y = "Taille de population (N)") +
    theme_classic()
```


Taille de population (N)
60
50
40
30
0.0
2.5
5.0
7.5
10.0
12.5
Temps (t)

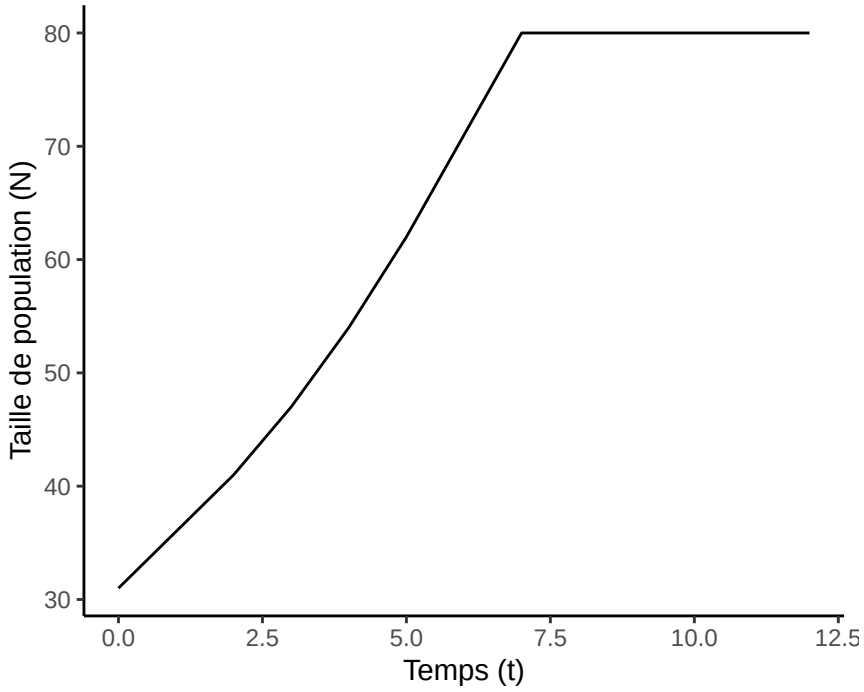
3.2 Compétition par interférence (contest)

```
#1. Déterminer les paramètres du modèle
taux_croissance= 1.148 #Taux de croissance maximal (faible abondance)
temps=12
abondance_initiale = 31
capacite_biotique= 80

#2. Créer un "data frame" vide pour enregistrer le résultat
df <- data.frame()

#3. Faire rouler la simulation
  #Pour chaque pas de temps
for (t in seq(0,temps)){

  #si le premier pas de temps, utiliser taille de pop. initiale
        if (t == 0) {
          taille_population <- abondance_initiale

        #sinon
        } else {
#4. Ajuster le taux de croissance selon la densité avec le modèle de Beverton-Holt
 taux_croissance_ajust <-(taux_croissance* capacite_biotique)/
  (taux_croissance* taille_population - taille_population + capacite_biotique)

#5.Calculer la taille de la population avec la formule (Nt+1= Nt*R)
  taille_population <- round(taille_population * taux_croissance_ajust, digits=0)
        }

#6. Ajouter le résultat au "data frame"
  df <- rbind(df, data.frame(temps = t, taille_population = taille_population))
}

ggplot(df, aes(x = temps, y =  taille_population)) +
    geom_line() +
    labs(x = "Temps (t)", y = "Taille de population (N)") +
    theme_classic()
```


Taille de population (N)
60
50
40
30
0.0
2.5
5.0
7.5
10.0
12.5
Temps (t)

3.3 Compétition par interférence avec plafond (ceiling)

```
#1. Déterminer les paramètres du modèle
taux_croissance= 1.148 #Taux de croissance maximal (faible abondance)
temps=12
abondance_initiale = 31
capacite_biotique= 80

#2. Créer un "data frame" vide pour enregistrer le résultat
df <- data.frame()

#3. Faire rouler la simulation
  #Pour chaque pas de temps
for (t in seq(0,temps)){

  #si le premier pas de temps, utiliser taille de pop. initiale
        if (t == 0) {
          taille_population <- abondance_initiale

        #sinon
        } else {
#4. Ajuster le taux de croissance selon la densité
    #si la taille de population est plus petite que K, alors croissance exponentielle sinon croissance
  if(taille_population<capacite_biotique){
    taux_croissance_ajust <-  taux_croissance
  #Sinon si taille de pop est égale ou plus grande que K, utilisé un R de 1
  }else {
    taux_croissance_ajust =  1
  }

#5.Calculer la taille de la population avec la formule (Nt+1= Nt*R)
  taille_population <- round(taille_population * taux_croissance_ajust, digits=0)

   #Avec un plafond la population ne peut dépasser la capacité biotique
  #donc limiter à la capacité biotique
  if(taille_population> capacite_biotique){
  taille_population= capacite_biotique
              }
        }

#6. Ajouter le résultat au "data frame"
  df <- rbind(df, data.frame(temps = t, taille_population = taille_population))
}

ggplot(df, aes(x = temps, y =  taille_population)) +
    geom_line() +
    labs(x = "Temps (t)", y = "Taille de population (N)") +
    theme_classic()
```


Taille de population (N)
80
70
60
50
40
30
0.0
2.5
5.0
7.5
10.0
12.5
Temps (t)

3.4 Compétition par exploitation (scramble) avec stochasticité

```
#1. Déterminer les paramètres du modèle
survie= 0.921
fecondite= 0.227
ecart_type_R= 0.075
temps=12
abondance_initiale = 31
capacite_biotique= 80
nombre_replications=100


#2. Créer un "data frame" vide pour enregistrer le résultat
df <- data.frame()

#3. Faire rouler la simulation
  #pour chaque réplication
  for (r in 1:nombre_replications){

    #Pour chaque pas de temps
    for (t in seq(0,temps)){

  #si le premier pas de temps, utiliser taille de pop. initiale
        if (t == 0) {
          taille_population <- abondance_initiale

        #sinon
        } else {
#4. Ajuster le taux de croissance selon la densité avec le modèle de Ricker
 taux_croissance_ajust <- (survie+fecondite)^(1-(taille_population/capacite_biotique))

#5. Ajuster le taux de croissance selon la stochasticité environnementale
  taux_croissance_stochas <-  rnorm(1, mean = taux_croissance_ajust, sd = ecart_type_R)
  #Si le taux de croissance est négatif, le définir à 0
  taux_croissance_stochas <- ifelse(taux_croissance_stochas < 0, 0, taux_croissance_stochas)

  #Calculer le coefficient de changement du taux de croissance avec stochasticité
    coef_stochas_environ <- taux_croissance_stochas/taux_croissance

  #Ajuster le taux de survie proportionellement au changement du R
   survie_stochas <- survie*coef_stochas_environ
   #Si le taux de survie est négatif, le maintenir à 0
   #et si plus grand que 1, le maintenir à 1
    survie_stochas <- ifelse(survie_stochas < 0, 0, ifelse(survie_stochas > 1, 1, survie_stochas))

    #Faire la même chose pour le taux de fécondité
    fecondite_stochas<- fecondite*coef_stochas_environ
    #Toutefois, ici le taux de fécondité peut être plus grand que 1
    fecondite_stochas <- ifelse(fecondite_stochas < 0, 0, fecondite_stochas)

#5. Considérer la stochasticité démographique
  #Effectuer un tirage binomiale (0 ou 1) pour chaque individu
    #pour déterminer combien d'individus ont survécus
  survivants <- rbinom(taille_population, 1, survie_stochas)
```

```r
  #Calculer ensuite combien de jeunes sont produit par individu
  #avec le taux de fécondité, on utilise ici une distribution de Poisson,
  #car certains individus peuvent faire plus qu'un jeune en une année
  jeunes <- rpois(taille_population, fecondite_stochas)

  #Faire la somme des survivants et jeunes produits
  taille_population <- sum(survivants) + sum(jeunes)
        }

#6. Ajouter le résultat au "data frame"
  df <- rbind(df, data.frame(replication=r, temps = t, taille_population = taille_population))
    }
  }

ggplot(df, aes(x = temps, y =  taille_population, group=replication, col=replication)) +
    geom_line() +
    labs(x = "Temps (t)", y = "Taille de population (N)") +
    theme_classic()+
    guides(col= "none")
```