TD7 Modèle de Beverton-Holt

Le TD est en lien avec le **problème 6** de la matière « Introduction à la modélisation en SVT ». Les questions avec 🧩 sont notées uniquement pour les étudiants de la licence de mathématiques et d'informatique.

Exercice 7.1 Évolution temporelle

Soit un modèle de Beverton-Holt

$$n_{t+1} = \frac{r_o \times n_t}{1 + \frac{n_t}{m}} \tag{11}$$

décrivant l'évolution année par année.

7.1.1 Écrire une fonction qui permet de calculer la prochaine étape à partir de la valeur actuelle de la population, cette fonction s'appelle `modele` et elle prend en argument les éléments suivants :

- 🡆 **argument** `actuel`
- 🡆 **argument** `r_0` qui par défaut vaut 2
- 🡆 **argument** `m` qui par défaut vaut 10

retour 🡆 une valeur numérique qui est calculée avec (11)

7.1.2 🧩 En utilisant les fonctions `expand` et `arrange`, construire un tibble `parametre` qui contient :
- `annee` de 0 à 15 deux fois.
- `condition_initiale` qui vaut 0.1 seize fois et 15 seize fois.

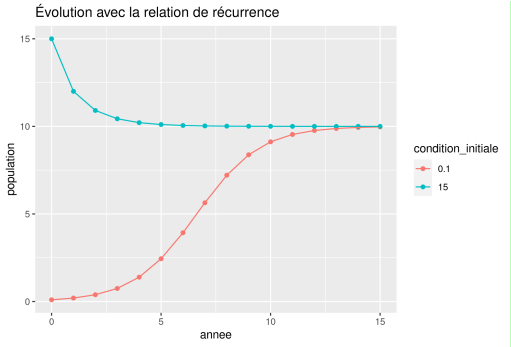
7.1.3 Avec l'algorithme A2 du cours, construire un tibble `modele_recurrence` qui contient les données de l'évolution de la population pour deux conditions initiales avec les *variables* :
- `annee` de 0 à 15 deux fois.
- `condition_initiale` qui vaut 0.1 seize fois et 15 seize fois.
- `population` qui contient le nombre d'individus

Il faut que le tibble `modele_recurrence` soit bien formaté pour pouvoir ensuite répondre à la question suivante.

7.1.4 Tracer l'évolution des deux population en fonction des années en donnant deux couleurs différentes selon les conditions initiales. Il faut ajouter le titre suivante à la figure « Évolution avec la relation de récurrence ».

Correction 7.1

```r
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))
library(tidyverse)

# Q 7.1.1
modele <- function(actuel, r_0 = 2, m = 10){
    return(r_0 * actuel / ( 1 + actuel / m))
}

# Q 7.1.2
parametre <- tibble(annee = 0:15) %>%
    expand(annee, condition_initiale = c(0.1, 15)) %>%
    arrange(condition_initiale)

# Q 7.1.3 Avec des boucles pour
modele_recurrence <- tibble()
for (pop_initiale in c(0.1, 15)){
    population <- c()
    population[1] <- pop_initiale
    for (annee in 1:15){
        population[annee+1] = modele(population[annee])
    }
    donnee <- tibble(annee = 0:15,
                     population,
                     condition_initiale = pop_initiale)
    modele_recurrence <- bind_rows(modele_recurrence, donnee)
}
modele_recurrence <- modele_recurrence %>%
    mutate(condition_initiale = factor(condition_initiale))

# Q 7.1.3 En programmation fonctionnelle
modele_recurrence <-  tibble(annee = 0:15) %>%
    expand(annee, condition_initiale = c(0.1, 15)) %>%
    arrange(condition_initiale) %>%
    mutate(r_0 = 2) %>%
    group_by(condition_initiale) %>%
    mutate(population = accumulate(.x = r_0[-1],
                                   .f = modele,
                                   .init = condition_initiale[1])) %>%
    mutate(condition_initiale = factor(condition_initiale))

# Q 7.1.4
ggplot(data = modele_recurrence) +
    aes(x = annee,
        y = population,
        color = condition_initiale) +
    geom_point() +
    geom_line() +
    labs(title = "Évolution avec la relation de récurrence")
ggsave("modele.png")
```



Exercice 7.2 Modèle général

Le modèle général est donné par

$$n(t) = \frac{k \times n_0}{n_0 + (k - n_0) \times r_0^t} \tag{12}$$

Le tibble `modele_recurrence` contient le tibble de l'exercice précédent.

7.2.1 Créer un tibble `modele_general` avec les calculs de l'évolution de la population en utilisant le modèle général et tous les paramètres de l'exercice précédent.

7.2.2 Tracer les valeurs numériques obtenues avec le modèle général. Il faut ajouter le titre suivante à la figure « Évolution avec le modèle général ».

7.2.3 Sélectionner uniquement les *variables* `annee`, `condition_initiale` et `population` dans les tibbles `modele_general` et `modele_recurrence`

7.2.4 Est-ce que les données du modèle général sont identiques à celles obtenues à l'exercice précédent ? Écrire le code qui répond à la question en donnant un booléen comme réponse.

7.2.5 Jusqu'à quelle décimale le modèle général et le modèle par récurrence donnent-ils exactement les mêmes valeurs ? Écrire le code qui permet de répondre à la question en utilisant les fonctions : `pull`, `round` et `setequal`. Le résultat doit être mis dans la variable numérique `decimale`.

Correction 7.2

```
1 # Q 7.2.1
2 modele_general <- tibble(annee = rep(0:15, 2))  %>%
3     mutate(n_0 = c(rep(0.1, 16), rep(15, 16))) %>%
4     mutate(r_0 = 2) %>%
5     mutate(m = 10) %>%
```

```r
    mutate(k = (r_0-1)*m) %>%
    mutate(population = (k*n_0)/(n_0 + (k-n_0)*r_0^(-annee))) %>%
    rename(condition_initiale = n_0) %>%
    mutate(condition_initiale = factor(condition_initiale))

# Q 7.2.2
ggplot(data = modele_general) +
    aes(x = annee,
        y = population,
        color = condition_initiale) +
    geom_point() +
    geom_line() +
    labs(title = "Évolution avec le modèle général")
ggsave("modele_2.png")

# Q 7.2.3
modele_general <- modele_general %>%
    select(annee, condition_initiale, population)

modele_recurrence <- modele_recurrence %>%
    select(annee, condition_initiale, population)

# Q 7.2.4
setequal(modele_general, modele_recurrence)
waldo::compare(modele_recurrence, modele_general)

# Q 7.2.5
decimale = 0
while(setequal(round(pull(modele_general, population), decimale+1),
               round(pull(modele_recurrence, population), decimale+1))){
```