Mise en forme des données de floraison

Objectif

Ce document présente la mise en forme des données brutes issus du fichier `donnees.floraison.csv`. L'objectif est d'obtenir pour une parcelle (bloc, sous-bloc) et pour chaque jour (entre le premier jour de relevés et le dernier) :

- le nombre d'inflorescences vivantes observées ;
- le nombre d'inflorescences mortes observées ;
- le nombre de nouvelles inflorescences (*i.e* qui sont apparues le jour de l'observation) ;
- le nombre théorique d'inflorescences au stade C/D ;
- le nombre théorique d'inflorescences au stade E ;
- le nombre théorique d'inflorescences au stade F ;
- le nombre théorique d'inflorescences vivantes ;
- le nombre théorique d'inflorescences mortes.

Les variables des données brutes utilisées sont :

- `Bloc` : le bloc dans lequel l'observation a été effectuée ;
- `Traitm` : la modalité de couverture du sol ;
- `ApdateC` : la date d'appartion du stade phénologique C pour le bourgeon apical ;
- `ApdateM` : la date de mort du bourgeon apical ;
- `Lat`i`dateC` : la date d'apparition du stade phénologique C pour le $i^{\text{ème}}$ bourgeon latéral, avec $i \in \{1, \dots, 5\}$;
- `Lat`i`dateM` : la date de mort du $i^{\text{ème}}$ bourgeon latéral, avec $i \in \{1, \dots, 5\}$.

Pour le calcul des observations théoriques, on estime que :

- une inflorescence est au stade C/D entre sa naissance observée jusqu'au jour 6 ;
- une inflorescence est au stade E du jour 7 au jour 15 ;
- une inflorescence est au stade F du jour 16 au jour 49 ;
- une inflorescence est morte au bout du $50^{\text{ème}}$ jour ;
- une inflorescence est théoriquement vivante si elle est au stade C/D, E ou F.

Réalisation

Chargement des librairies

```
library(tidyverse) ## Pour nettoyer les données
library(lubridate) ## Pour gérer les dates
library(magrittr)
```

Importation du fichier

```
data <- read_csv2("donnees.floraison.csv")
data %<>% mutate_at(c("ApdateC", "ApdateM", c(paste0("Lat",1:5,"dateC"),
                    paste0("Lat", 1:5, "dateM"))), list(as.Date), "%d/%m/%Y")
```

Fonction

La fonction suivante renvoie un fichier `csv` contenant les informations voulues. On fera l'hypothèse que si une date de mort n'est pas disponible, cela signifie que la mort est intervenue après la fin des relevés.

```r
obs_floraison <- function(bloc = 1, modalite = NULL, annee = 2017){
    ## Renvoie un fichier .csv contenant pour chaque date de relevés les inflorescences
    ## vivantes, mortes et nouvelles qui ont étés observées.

    ## Arguments : bloc : 1 ou 2, selon le bloc voulue
    ## modalité : "B", "Sn" ou "E". Par défaut, renoie pour le bloc entier.

    ## Séléction des données selon les critères
    data %<>% filter(Bloc == bloc & Annee == annee)
    if (!is.null(modalite))
        data %<>% filter(Traitm == modalite)

    ## On réorganise les données 1 ligne = 1 inflo
    da <- data %>% filter(!is.na(ApdateC)) %>% select(ApdateC, ApdateM) %>%
        rename(birth = ApdateC, death = ApdateM)
    dl1 <- data %>% filter(!is.na(Lat1dateC)) %>% select(Lat1dateC, Lat1dateM) %>%
        rename(birth = Lat1dateC, death = Lat1dateM)
    dl2 <- data %>% filter(!is.na(Lat2dateC)) %>% select(Lat2dateC, Lat2dateM) %>%
        rename(birth = Lat2dateC, death = Lat2dateM)
    dl3 <- data %>% filter(!is.na(Lat3dateC)) %>% select(Lat3dateC, Lat3dateM) %>%
        rename(birth = Lat3dateC, death = Lat3dateM)
    dl4 <- data %>% filter(!is.na(Lat4dateC)) %>% select(Lat4dateC, Lat4dateM) %>%
        rename(birth = Lat4dateC, death = Lat4dateM)
    dl5 <- data %>% filter(!is.na(Lat5dateC)) %>% select(Lat5dateC, Lat5dateM) %>%
        rename(birth = Lat5dateC, death = Lat5dateM)

    ## Dates d'appartition et de mort pour chacune des inflorescences observées
    inflo <- bind_rows(da,dl1,dl2,dl3,dl4,dl5) %>% arrange(birth) %>%
        mutate(stadeC_theo = birth, stadeE_theo = birth+7, stadeF_theo = birth+16,
               mortes_theo = birth + 50)

    ## Création d'un vecteur contenant chaque jour entre le premier jour de relevés
    ## et le dernier.
    days <- unique(c(inflo$birth, inflo$death))
    days <- days[which(!is.na(days))]
    days <- as_date(days[1]:days[length(days)])

    ## Calcul des valeurs voulues pour chaque jour
    alive <- rep(NA, length(days))
    new <- rep(NA, length(days))
    dead <- rep(NA, length(days))
    alive_theo <- rep(NA, length(days))
    stadeC_theo <- rep(NA, length(days))
    stadeE_theo <- rep(NA, length(days))
    stadeF_theo <- rep(NA, length(days))
    dead_theo <- rep(NA, length(days))
    for (day in 1:length(days)){
        alive[day] <- length(which(inflo$birth <= days[day] & inflo$death > days[day])) +
            length(which(inflo$birth <= days[day] & is.na(inflo$death)))
        new[day] <- length(which(inflo$birth == days[day]))
        dead[day] <- length(which(inflo$death <= days[day]))
        alive_theo[day] <- length(which(inflo$birth <= days[day] &
                                            inflo$mortes_theo > days[day]))
```

```
        stadeC_theo[day] <- length(which(inflo$stadeC_theo <= days[day] &
                                         inflo$stadeE_theo > days[day]))
        stadeE_theo[day] <- length(which(inflo$stadeE_theo <= days[day] &
                                         inflo$stadeF_theo > days[day]))
        stadeF_theo[day] <- length(which(inflo$stadeF_theo <= days[day] &
                                         inflo$mortes_theo > days[day]))
        dead_theo[day] <- length(which(inflo$mortes_theo <= days[day]))
    }

    ## Résultats
    res <- as_tibble(cbind(days, alive, dead, new, alive_theo, stadeC_theo, stadeE_theo,
                           stadeF_theo, dead_theo))
    res %<>% mutate_at("days", list(as_date))

    write_csv2(res, paste0("Bloc", bloc,"_", modalite,".csv"))
    return(res)

}
```

Différences avec Laurie

Toutes les observations du jeu de données ne sont pas de 2017. Par défaut, on ne récupère ici que les données de 2017.