TP: Manipulation et représentation de données

2021/2022

Contents

Objectif du TP

Ce TP a pour objectif de vous initier à une composante fondamentale de l'analyse de données: **la manipulation et le nettoyage d'une base de données.** Nous allons voir comment effectuer de manière rapide et propre des opérations standards quand on a affaire à un nouveau jeu de données. Ce TP doit se faire avec l'aide du document `Tutoriel_manipulation_donnees_avec_R.html`.

Librairies à charger

On utilisera la suite de packages `tidyverse`:

```
rm(list = ls()) # Nettoyage de l'environnement de travail
library(tidyverse) # Chargement de la librairie tidyverse
```

Données

Source

Differences in soil properties among contrasting soil types in northern Borneo, G. Sellan et al (2020).

Caractéristiques des sols

On s'intéresse aux caractéristiques physico-chimiques de sols tropicaux à Borneo. On a mesuré, sur 180 sites, en 3 profondeurs différentes, des caractéristiques chimiques sur 539 échantillons (180×3, auquels se soustrait une mesure manquante). Les 18 caractéristiques mesurées sont les suivantes:

- Teneurs en sodium, Magnésium, Calcium, Potassium échangeables (`Exc.Na`, `Exc.Mg`, `Exc.Ca`, `Exc.K`)

- Teneur en eau (colonne `MC`),
- Phosphore `P`, Carbone `C` et Azote `N`
- `pH` et Aluminium et Acidité échangeable `Exc.Al` `Exc.Ac`, Saturation en bases(`BS`)
- *Effective Cation Exchange Capacity* (`ECEC`)
- Nitrates `NO3`, Ammonium `NH4`
- Pourcentage d'argile `Clay`, de limon `Silt` et de sable `Sand` (dont la somme vaut 100).

En plus de ces caractéristiques, on connaît le type de sol (dans la colonne `Soil`), qui est soit Alluvial, dunaire (*heath*), ou grès (*sandstone*)) et la profondeur à laquelle a été faite le prélèvement (colonne `Depth`) à (0-5cm, 5-20 cm ou 25-30 cm).

En plus des ces colonnes sont enregistrées les noms des sites sur lesquels on a échantillonné les sols. Toutes les infos sont regroupées dans les colonnes `Plot`, `Subplot`, `Block`, `Name1`, `Name` et `Names2`.

Ces données sont disponibles dans le jeu de données `donnees_soil_characteristics.csv`. On les chargera ainsi:

```
sol_initial <- read.table("donnees_sols_chimies.csv",
                          sep = ",", # Séparateur de champs
                          header = TRUE) %>% # La 1ere ligne donne le nom des colonnes
  as_tibble() # Facilite l'affichage ensuite
sol_initial
```

Abondance d'espèces d'arbres par site

Parallèlement à cela, on dispose de l'abondance de 639 espèces d'arbres sur nos sites, dans le fichier `donnees_abondance.csv`. Dans ce tableau de 900 lignes et 640 colonnes, chaque ligne est donc associée à un site, dont le nom est renseigné dans la colonne `Name`. Les 639 colonnes restantes correspondent aux 639 espèces. Pour un site et une espèce donnés, on a renseigné dans le tableau le nombre d'individus recensé.

```
abondance_initial <- read.table("donnees_abondance.csv",
                                sep = ",", # Separateur de champ
                                header = TRUE, # 1ere ligne donne le nom des colonnes
                                row.names = 1 # La 1ere colonne est le numero de ligne
                                ) %>%
  as_tibble() # Facilite l'affichage ensuite
abondance_initial
```

Objectif

Notre objectif sera, à terme, de regarder si la richesse d'un site (en termes d'espèces et d'individus) pourra être expliquée par ses caractéristiques. Pour cela, nous allons devoir:

- Nettoyer les données;
- Effectuer des transformations des données basées sur des choix:
  - On a les caractéristiques par site et par profondeur. Qu'est ce que **LA** caractéristique d'un site? Une des trois profondeurs? Une moyenne sur les trois profondeurs?
  - Il va falloir créer un variable de richesse par site.
  - Il va falloir ensuite joindre les deux tableaux.

1 Nettoyage et manipulation des données

Dans cette section, on utilisera le tutoriel `Tutoriel_manipulation_donnees_avec_R.html`.

1.1 Données de sols

1. A partir du jeu de données `sol_initial`. On veut effectuer les nettoyages suivants:

a. On veut se débarasser des colonnes `Plot`, `Subplot`, `Block`, `Name1`, et `Names2`. On utilisera la fonction `select` (voir la section 5.4 du tutoriel de manipulation de données). Dans un premier temps, le tableau résultant sera stocké dans un objet `sol_intermediaire_1`.

b. Ensuite, on veut renommer les colonnes (en utilisant la fonction `rename`, section 5.6 du tutoriel):

- `MC` en `Eau`
- `ECEC` en `Exc.Cations`
- `Soil` en `Sol`
- `Name` en `Site`
- `Depth` en `Profondeur`
- `BS` en `SatBase`
- `Clay` en `Argile`
- `Silt` en `Limon`
- `Sand` en `Sable` Dans un premier temps, le tableau résultant sera stocké dans un objet `sol_intermediaire_2`.

c. Ensuite, en utilisant les sections 9.2 et 9.3 du tutoriel:

- Transformez la nouvelle colonne `Sol` en facteurs pour que les niveaux soient en Français, dans cet ordre (`Alluvial`, `Grès` (`Sandstone`) et `Dunaire` (`Heath`)).
- Transformez la colonne `Profondeur` en facteurs pour que les niveaux soient dans l'ordre "0-5 cm", "5-20 cm", "20-35 cm". Dans un premier temps, le tableau résultant sera stocké dans un objet `sol_intermediaire_3`.

d. Enfin, on a remarqué qu'il existait des valeurs manquantes dans les colonnes `pH` et `Av.P`. Pour les lignes correspondantes, seules ces informations manquent. Afin d'éviter de supprimer la ligne entière (et donc toutes les autres mesures pour ce site), utiliser la section 7.7.2 pour remplacer dans `sol_intermediaire3` chaque valeur manquante par la moyenne de sa quantité, **pour le type de sol correspondant** (on groupera donc selon la variable `Sol`. Stockez le résultat dans un objet `sol_intermediaire4`.

2. A l'aide de l'opérateur `%>%` décrit dans la section 4, faites toutes ces modifications en **un seul traitement séquentiel** et stockez le résultat dans un jeu de données `sol_propre`. **De manière générale, tout faire d'un seul tenant permet d'éviter la création de trop d'objets intermédiaires inutiles.** On pourra dans la suite supprimer les objets intermédiaires:

```
# On supprime les tableaux intermediaires inutiles
rm(sol_intermediaire1, sol_intermediaire2,
   sol_intermediaire3, sol_intermediaire4)
```

3. On veut maintenant créer un tableau où il y a une seule ligne par site, afin de le joindre avec la table des abondances. Pour cela, plusieurs options s'offrent à nous:

- **Option choix d'une profondeur:** Avec la fonction `filter` (section 5.3), stockez dans un tableau `sol_superficiel` le tableau ne comprenant que la profondeur `0-5 cm`. Dans ce tableau, on aura supprimé la colonne `Profondeur` (avec `select(-Profondeur)`, section 5.4)
- **Option moyenne** Avec les fonctions `group_by` et `summarise_if` (section 5.10 avec l'aide de la section 5.9.1) créez un tableau `sol_moyen` où chaque ligne est la moyenne (pour les colonnes numériques) des caractéristiques par site.

Créez ces deux tableaux à partir de `sol_propre`.

1.2 Données d'abondance

4. A partir du jeu de données `abondance_initial`, on veut effectuer les nettoyages suivants (vous ferez ces nettoyages dans un seul traitement séquentiel grâce au `%>%` vu en section 4, et stockerez le résultat dans un jeu de données `abondance_propre`):

a. On veut renommer La colonne `Name` en `Site` (section 5.6 du tutoriel)

b. Cette colonne `Site` contient le nom des sites, qui est commun avec le tableau `sol` traité dans la partie précédente. Malheureusement, ici, la lettre finale est séparée des chiffres par un espace au lieu d'un `_`. On veut donc modifier cette colonne `Site` en remplaçant les espaces par des `_` (function `str_replace`, section 8.2 du tutoriel).

5. A l'aide la fonction `pivot_longer` (section 7.4), transformez `abondance_propre` en un tableau au format long (que vous nommerez `abondance_long`), c'est à dire comportant 3 colonnes:

   - La colonne `Site` (qui est celle initiale)
   - Une nouvelle colonne `Espece` qui contiendra le noms de toutes les espèces présentes auparavant
   - Une nouvelle colonne `NbIndividus` qui contiendra, pour un site et une espèce donnée, le nombre d'individus correspondant.

6. On va créer un tableau `abondance_genre_metasite` qui compte le nombre d'individus de différents genres obtenus sur chaque metasite. Dans `abondance_long`, la colonne `Site` est du type: `MetaSite_Bloc` et la colonne `Espece` est codée `Genre.Espece`. A l'aide de la fonction `separate` (section 7.2), dans le tableau `abondance_long`:

   - Séparez la colonne `Site` en deux colonnes `MetaSite` et `Bloc` comprenant les deux informations.
   - Séparez la colonne `Espece` en deux colonnes `Genre` et `Espece`. **Attention** Quand le séparateur est un `"."`, on mettra comme argument `sep = "[.]"`
   - Ensuite, à l'aide des fonctions `group_by` et `summarise` (section 5.10), stockez dans un colonne `NbArbres` le nombre total d'invidus de chaque genre par meta-site. **Attention**: le `group_by` devra être fait sur les variables `MetaSite` **et** `Genre` (on fera `group_by(MetaSite, Genre)`).
   - Vous effectuerez ces trois traitements de manière séquentielle grâce à , et stockerez le résultat dans `abondance_genre_metasite`.

7. A partir du tableau `abondance_long`, créer (en un seul traitement séquentiel) le tableau `richesse_par_site` de la manière suivante:

   a. En utilisant la fonction `filter` (section 5.3), sélectionnez simplement les lignes où le nombre d'individus comptés est supérieur à 0;
   b. De cette sélection, en utilisant les fonctions `group_by` et `summarise` (section 5.10), vous compterez, pour chaque site, la densité d'arbres sur ce site, à savoir le nombre d'individus total (toutes espèces confondues), divisé par 400 (m^2), la surface de chaque site. Cette information sera stockée dans une colonne `densite_arbre`. De plus, on veut compter le nombre d'espèces recensées pour lesquelles on a compté **au moins un individu** (dans une colonne `richesse_specifique`).

1.3 Jointure des données

8. On veut créer un tableau `donnees_richesses` qui, pour chaque site présent à la fois dans `richesse_par_site` et `sol_superficiel`, donne toutes les caractéristiques du site. Faites cette jointure à l'aide de la fonction `left_join` et `inner_join` (section 6.3). Quelle jointure doit on garder? Stockez le résultat dans `donnees_richesses`.

2 Représentation graphique des données

Un des grands avantages de Tidyverse est qu'il permet de mettre en forme les données sous un format que l'on peut combiner avec **ggplot**.

1. A partir de `donnees_richesses`, représentez graphiquement `densite_arbre` en fonction de `richesse_specifique`. Vous colorirez selon le type de sol (colonne `Sol`).

2. Sur le graphique précédent, renommez les axes en français, et mettez un titre.

3. Dans le jeu de données `sol_propre`, faites un boxplot de la distribution du total de cations échangeables `Exc.Cations` en fonction des différents types de sol (`Sol`).

4. Sur ce même graphique, dans la commande `aes()`, rajoutez la commande `fill = Profondeur` pour inclure l'information de la profondeur dans la distribution.

5. A l'aide de la section 8, sur trois graphiques différents (grâce à `facet_wrap`), représentez les histogrammes (grâce à `geom_histogram`) de la distribution du phospore (colonne `NO3`).

6. Sur la graphique précédent, rajoutez dans la commande `aes` l'argument `fill = Profondeur` pour voir apparaitre l'information profondeur. Dans `geom_histogram` vous rajouterez:

   - `position = "identity"` pour que les histogrammes soient supperposés,
   - `alpha = 0.5` pour qu'il y ait de la transparence

7. Sur ce dernier graphique, renommez les axes pour qu'ils soient compréhensibles (section 6) et mettez un fond blanc avec `theme_bw` (section 9).

8. **Heat map** Avec les données `abondance_genre_metasite` on va faire une représentation de la matrice d'abondance. On aura en abscisse les espèces, en ordonnée les sites, et dans les cases une couleur donnant le nombre d'arbres pour chaque espè. Séquentiellement:

   - Représentez `abondance_genre_metasite` avec `ggplot(abondance_genre_metasite)`
   - Ajoutez `aes(x = Genre, y = MetaSite, fill = NbArbres)` (on remplira une case, d'où fill)
   - Ajoutez la commande `geom_raster()`.

9. Sur le graphique précédent, ajoutez (comme dans le section 9.1):

```
theme(legend.position = "none", # pas de légende
      axis.text.x = element_text(angle = 90, size = 6), # Le texte en absisses est est vertical et peti
      axis.text.y = element_text(size = 7)) # On rapetisse la taille du texte en y
```