TD: Introduction et pratique de R et Bioconductor

Ghislain Bidaut, Plateforme Cibi, CRCM, Aix-Marseille Université

24/03/2022

1 Introduction

Ce TD va nous permettre d'aborder l'utilisation de R pour la manipulation de données et de fichiers. Nous aborderons également l'utilisation de Bioconductor.

Ce TD est librement inspiré de la formation R proposée par Justine Guégan (https://pf-bb.github.io/Formation-Rrrr/ (https://pf-bb.github.io/Formation-Rrrr/)), ainsi que d'un TP proposé par Jacques Van Helden (https://jvanheld.github.io/stat1/practicals/02_yeast_annotations/02_yeast_annotations.html (https://jvanheld.github.io/stat1/practicals/02_yeast_annotations/02_yeast_annotations.html))

2 Organisation

Il est souhaitable de réaliser ce TD dans un fichier **.R** (Dans RStudio, Fichier -> Nouveau Fichier -> R Script). Par exemple, dans `~/TP_R/TP_Intro_R.R` .

N'oubliez pas d'organiser votre code en sections avec des commentaires, comme, par exemple, ceci:

```
# Exercice 1
A <- 10
# Exercice 2
C <- 11
```

... et de commenter votre code

```
# Je génère un échantillon de 20 valeurs tirées d'une loi normale
x <- rnorm(20, 0, 1)
```

3 Manipulation d'objets

3.1 Exercice: Extraction d'éléments d'un vecteur

Soit `genel <- c("ALK", "HRAS", "BRAD", "AKT1", "EGFR")`

1. Quelle est la longueur du vecteur ? indice : `length`
2. Essayer de faire `genel[1:3]`. Qu'obtenez-vous ?
3. Créer un nouveau vecteur `gl2` ne contenant que "HRAS" et "EGFR".
4. Essayer de faire `genel[-1]`. Qu'obtenez-vous ?
5. Trier par ordre alphabétique. indice : `sort`
6. Ajouter les gènes "PIK3CA" et "KIT" au vecteur `gl2`

3.2 Exercice: Suites de nombres

1. Créer une séquence de nombre entiers de 1 à 100 et stockez la dans le vecteur `a`
2. Ajouter à ce vecteur les valeurs de 200 à 300
3. Créer une séquence de nombre entiers pairs de 2 à 100

3.3 Exercice: Suites et répétitions

1. Créer le vecteur `c1` contenant tous les multiples de 2 compris entre 1 et 50.
2. Créer le vecteur `c2` contenant 3 fois chacun des 10 chiffres (soit 0, 0, 0 ; 1, 1, 1 ; 2, 2, 2 ; 3, 3, 3 ; etc.). Indice: `rep`
3. Créer le vecteur `c3` contenant une fois la lettre A, deux fois la lettre B, trois fois la lettre C . . . et 26 fois la lettre Z. Quelle est la longueur de cette suite ? Indice: `LETTERS; length`

3.4 Exercice: Facteurs

Soit un facteur `cell_type <- factor(c("C1","C2","C2","C2","C1","C2","C2","C1"))`.

1. Afficher les "niveaux" de cell_types
2. Calculer le nombre de "C1" et de "C2" dans `cell_type` en utilisant les fonctions `which`, `length` et des opérateurs binaires (==).
3. Que permet de faire la fonction `table` ? Appliquer la à `cell_type`.

3.5 Exercice: Chaines de caractères

La commande `paste` permet de concaténer du texte.

1. Essayer la commande `paste("chr", 1, sep="")`.
2. Créer, en une seule ligne de commande, le vecteur `chrs` contenant les noms suivants : chr1, chr2, ..., chr22, chrX, chrY. Indice : `paste`.

3. Créer une chaîne unique `chrsj` contenant tous les champs de `chrs` séparés par un espace " ". Indice : `paste` .
4. Recréer un vecteur `chrs2` à partir de la chaine `chrsj` . Indice : `strsplit` . Vérifier s'il est identique à `chrs` .

3.6 Exercice: Matrices

1. Exécuter la commande `s <- rep(1:10, 10)`
2. Utiliser `s` pour construire une matrice `M` à 10 lignes et 10 colonnes, comportant des `1` sur la première ligne, des `2` sur la seconde, etc... Indice : `matrix`
3. Afficher les dimensions de cette matrice. Indices : `dim` , `ncol` , `nrow`
4. Utiliser la fonction `t` sur cette matrice pour créer une matrice `MT` . Que contient `MT` ?
5. Effectuer le **produit matriciel** entre `MT` et `M` (dans cet ordre). Qu'obtient-on ?
6. Les commandes `M[1:5, ]` et `MT[, 1:5]` permettent de récupérer respectivement les 5 premières lignes de M et les 5 premières colonnes de MT. Inspirez-vous de ces commandes pour récupérer les lignes dont les valeurs sont comprises entre 1 et 2 de `M` et les colonnes dont les valeurs sont comprises entre 1 et 2 de `MT` .

3.7 Exercice: Tableaux de données

Créer une liste `x` contenant :

- une variable aléatoire gaussienne de taille 10 appelée `a`
- un vecteur contenant uniquement des 1, de taille 10 également, appelé `b` .

On peut accéder aux deux éléments de cette liste avec les commandes `x[[i]]` ou `x$nom_de_la_variable` . Indices : `list` , `rnorm` .

2. Créer un objet `y` qui est la transformation de cette liste en *data frame*. On peut maintenant parcourir les éléments de chaque objet comme pour une matrice avec la commande `y[i,j]` . indice `= data.frame` .
3. Créer deux objets `z1` et `z2` contenant respectivement les 3 premières et les 3 dernières lignes de `y` .

Quelle est la classe de ces deux objets ?

4. Rajouter à la liste x un vecteur numérique contenant les entiers de 1 à 26.
5. Essayer de transformer de nouveau `x` en data.frame. Que se passe-t-il ?

4 Lecture et écriture de fichiers

4.1 Exercice 1: Fichier de données températures

Nous allons utiliser les fichiers textes suivants: `temperatures_ville_original.txt` et `temperatures_ville_comment.csv` .

Ils sont récupérables à cet adresse:

https://mycore.core-cloud.net/index.php/s/pvmUZqNjTpYybCc?path=%2Fdata_R (https://mycore.core-cloud.net/index.php/s/pvmUZqNjTpYybCc?path=%2Fdata_R)

Je vous laisse voir les fichiers dans un éditeur de texte préalable pour déterminer le séparateur entre champs, le séparateur décimal utilisé pour les nombres à virgule, la présence de commentaires dans le fichier, la présence de valeurs manquantes et de quelle manière elle sont encodées.

1. En utilisant la fonction `read.table` , importer dans une variable nommée `tville_orig` le jeu de données nommé `temperatures_ville_original.txt` . Quel est le mode des objets créés par la fonction `read.table()` ?
2. Importez dans une variable nommée `tville_comment` le jeu de données `temperatures_ville_comment.csv` . Combien de valeurs manquantes sont contenues dans le fichier ? Quel est le mode des objets créés par la fonction `read.table()` ?
3. Recalculez la moyenne et l'amplitude des températures et ajoutez les comme nouvelles colonnes dans `tville_orig` . Utiliser la fonction `apply` . Trouvez s'il y a une différence dans les résultats à l'aide des fonctions `identical` et `all.equal`
4. Faites un plot au cours du temps des températures pour les villes ayant la plus forte (en rouge) et la plus faible amplitude (en bleu). (Fonction `plot`).

4.2 Exercice 2: Ecriture d'une matrice dans un fichier

1. Créer la matrice suivante :

$$M = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}$$

à l'aide des fonctions `seq` et `matrix`

2. Ecrire la matrice `M` dans un fichier nommé `matrice.txt` . Que remarquez-vous?
3. Ajouter des arguments à la commande précédente pour retirer des noms aux lignes et aux colonnes du fichier créé.
4. Sauver la matrice `M` au format **.RData** dans le fichier `matriceM.RData` grâce à la fonction `save` .
5. Que donne la commande `load("matriceM.RData")` ?
6. Sauver la session complète dans un fichier nommé `session.RData`

5 Etude de données génomique

Dans ce TP, nous allons faire des statistiques sur les gènes du génome humain. Pour cela, nous allons utiliser un fichier GFF d'annotations, disponible sur le site **GenCode** à l'adresse http://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_38/gencode.v38.annotation.gtf.gz (http://ftp.ebi.ac.uk/pub/databases/gencode/Gencode_human/release_38/gencode.v38.annotation.gtf.gz) .

Nous enregistrerons les manipulations dans un fichier RScript (extension `.R`).

5.1 Chargement des données

0. Télécharger, décompresser le fichier et stockez le dans votre répertoire de travail.
1. Charger le fichier dans R dans l'objet `human.gene.annot` avec la fonction appropriée ou avec la fonction d'importation de RStudio.

Il n'y a pas de nom de colonnes prédéfinis pour ce fichier, nous allons donc les ajouter à l'objet `human.gene.annot` avec la fonction `names` , sur la base du format de fichier décrit https://www.gencodegenes.org/pages/data_format.html (https://www.gencodegenes.org/pages/data_format.html)

Note: Un fichier GTF est une instance spéciale des fichiers GFF et donc partagent le même format.

2. Afficher les dimensions du tableau `human.gene.annot`
3. Afficher les premières lignes, les dernières lignes, puis ouvrez l'objet avec la fonction View, disponible dans RStudio.

5.2 Sélection d'un sous ensemble de données

1. Afficher la ligne 12. Afficher le nom du gène de la ligne 12.
2. Afficher les lignes 100 à 105, incluant les positions chromosomique, strand et le type de *feature*.
3. Afficher les types de *features* existantes.

5.3 Sélection d'un sous ensemble de données basé sur les valeurs du tableau

1. Sélectionner les lignes correspondantes aux gènes et comptez les (fonction `subset`).
2. Afficher le nombre d'entrées pour chaque *feature* avec `table` .
3. Afficher le nombre d'entrées pour chaque chromosome.
4. Faire une table de contingence en affichant le nombre de chaque feature pour chaque chromosome (Utiliser la fonction `table`)

5.4 Manipulation d'un tableau

Ajouter une colonne `length` au tableau `human.gene.annot` , définie comme étant la différence entre les colonnes `start` et `end` .

5.5 Plot de données

1. Faire un histogramme représentant la distribution des longueurs des gènes (fonction `hist`).
2. Faire un plot représentant le nombre de gènes par chromosome (fonction `barplot`).

5.6 Paramètres descriptifs

1. Extraire la moyenne, écart type, valeurs min et max des longueurs de gènes pour le génome total (`feature == "gene"`)

6 Solution