Université Rennes 2 — Année 2019-2020
Master 1 MAS — Madison Giacofci/Pierre Gloaguen

Importation de données et manipulation d'objets

De manière générale, on est rarement amené à saisir soi-même ses données dans R. D'une part, à des fins d'exemples et d'entraînement, R et ses nombreuses librairies possèdent de nombreux jeux de données. D'autre part, il est possible d'importer des jeux de données par le biais de fichiers externes. Les formats les plus courants sont les formats ASCII (fichiers `.txt`), le format CSV (fichiers `.csv`) et le format (binaire) propre à R (fichiers `.Rdata`). Les autres formats nécessitent en général de s'appuyer sur des fonctions issues de packages dédiés (comme `readxl` pour les fichiers Excel par exemple).
Ouvrez une session R avec Rstudio et placer vous dans votre répertoire personnel.

1 Chargement et manipulation d'un jeu de données de R

R propose de nombreux jeux de données, cela peut être de façon native (c'est-à-dire disponible dans la version basique) :

```
> women # affiche les donnees du jeu de donnees women
> data(women) # charge ces memes donnees dans votre session de travail
```

ou disponible au sein de packages particuliers. Dans ce cas, le chargement nécessite de préciser le package :

```
> data(Boston)
> data(Boston,package="MASS")
```

1.1 Data-frames et facteurs

Chargez le jeu de données `mtcars`.

1. Ce jeu de données est-il une matrice (`is.matrix`) ? Un data.frame (`is.data.frame`) ? Une liste ?
2. Quelle est sa dimension (`dim`) ?
3. Combien de variables contient-il ? Quel est leur nom (`names`) ?
4. Décrivez le jeu de données `mtcars` (`help` ou `??mtcars`).
5. Donnez le mode de chacune des variables à l'aide de la fonction `summary`.
6. Affectez à `x` la première colonne de `mtcars`, à l'aide du numéro de colonne (`[,i]`) et ensuite du nom de la colonne (`$`).
7. Affichez le 15^{e}, le 3^{e} et le 4^{e} élément de `x` dans cet ordre.
8. Affichez ses 10 premiers éléments. Idem dans l'ordre inverse (`rev`).
9. Affichez ses 10 derniers éléments (`tail`).
10. Transformez la variable Transmission (`am`) en un facteur (`as.factor`). Regardez à nouveau le résumé des données.
11. Affectez à y le facteur Transmission.
12. Créez un nouvel objet constitué de `x` et `y` (`cbind`).
13. Quel type d'objet a été crée ? Quelle est sa dimension ?

14. Affichez la 12^{e} ligne de cet objet.
15. Affichez les 8 premières lignes de cet objet.
16. Affichez la 2^{e} colonne de l'objet.
17. Affichez les consommations des véhicules à transmission automatique.
18. Créez un vecteur contenant les consommations moyennes pour les véhicules selon leur type de transmission.
19. Effectuer le même calcul (de consommations moyennes pour les véhicules selon leur type de transmission) sur le data frame `mtcars` et donner les types des objets obtenus :
    — en utilisant la fonction `subset`
    — en utilisant la fonction `by`
    — en utilisant la fonction `aggregate`
20. Créez un vecteur contenant les écarts types associés au consommations moyennes des véhicules en fonction de leur type de transmission.
21. Résumez les informations obtenues sur la consommation (moyenne et écart type pour chaque type de transmission) dans une liste.

2 Chargement de jeux de données externes

Le chargement de jeu de données externes peut être effectué au moyen de 3 fonctions principales : la fonction `read.table` pour les fichiers ASCII et CSV, la fonction `scan` qui représente une version plus flexible de `read.table` et la fonction `load` pour les fichiers `Rdata`.

1. Accédez à l'aide de la fonction `read.table`. A quoi correspondent les options `file`, `header`, `sep`, `dec`, `row.names`, `col.names`, `na.strings`, `colClasses`, `skip` ?
2. Accédez à l'aide de la fonction `scan`. A quoi correspond l'option `what` ?

2.1 Importation simple

1. Importez le jeu de données `cancerprostate`.
   — De quel objet s'agit-il ? Quelle est sa taille ?
   — A l'aide du résumé (`summary`), déterminer le type de chacune des variables. On peut aussi utiliser `str()`.
   — Modifier votre importation afin que les variables soient d'un mode approprié.
2. Importez le jeu de données `test1`
   — Que constatez-vous dans les données chargées ?
   — D'après-vous, que s'est-il passé ?
3. Importez le jeu de données `test2`.
   — Que constatez-vous dans les données chargées ?
   — Y a-t-il de données manquantes en Test2 ? Si oui, combien ? (on peut utiliser `is.na()`)
   — Extraire le data frame ne contenant pas de données manquantes.

2.2 Fusion de jeux de données

Les données d'intérêts peuvent parfois être contenues dans plusieurs jeux de données séparés.

1. Importez les jeux de données `etat1` et `etat2`.
   — Quelles sont leurs tailles respectives ? Que contiennent-ils ?
   — Fusionnez les deux jeux de données afin de réunir les informations dans 1 seul tableau :
      — à l'aide de la fonction `cbind()`

— à l'aide de la fonction `merge()`
Laquelle vous semble plus appropriée ?
— Que se passe-t-il si les individus diffèrent entre les 2 jeux de données ?

2. Création et manipulation de facteurs
— À l'aide de la fonction `cut`, créez un facteur `Richesse` séparant les états selon leur revenu moyen (`Income`) : Pauvre (inférieur ou égal à 4000), Aisé (entre 4001 et 5000) et Riche (supérieurs ou égal à 5001). Vérifiez si le résultat contient de `NA`'s.
— Effectuez le découpage de la variable `Income` cette fois-ci en 5 niveaux d'effectifs comparables (utilisez la fonction `quantile`).
— Afficher les listes des revenus appartenant à chaque niveaux du facteur (`split`).
— Renommez les facteurs en `R1`, `R2`, `R3`, `R4`, `R5` (on affecte à `levels(facteur)` les noms souhaités).
— Comment feriez-vous pour créer automatiquement le vecteur des noms `R1`, `R2`,... ? (commande `paste`)
— Pour fusionnez deux niveaux d'un vecteur (en cas de groupes à effectifs trop faibles par exemple), on peut renommer les niveaux des 2 classes à fusionner avec le même nom. Fusionnez les classes `R1` et `R2` ainsi que les classes `R3` et `R4` dans l'exemple.

2.3 Variables au format Date

Certains jeux de données contiennent parfois des variables correspondent à des jours, heures, minutes. Dans une certaine mesure, R peut traiter ces variables au sein d'un format de type `Date`.

1. Importez le jeu de données `ski` (vous pourrez utiliser l'option `skip`).
2. Importez le même jeu de données en utilisant un mode approprié pour chacune des variables (en particulier, on s'assurera que la date est au format `Date`).

Remarque. `Sys.time()` envoie l'heure et la date du système :

```
> t<-Sys.time()
> Sys.time()-t
```

2.4 Repérer des données manquantes

Repérer des données manquantes peut parfois s'avérer plus ardu que prévu : c'est pourquoi il est important de réaliser une inspection des données une fois celles-ci chargées.
Le jeu de données `bladder` contient les données génomiques d'individus atteints d'un cancer de la vessie.

1. Importez le jeu de données `bladder`. Décrivez brièvement ce dernier.
2. Contient-il des données manquantes ?
3. Prenez quelques variables au hasard et effectuez leur résumé et un boxplot de leurs valeurs (fonction graphique `boxplot`). Que remarquez-vous ?
4. Importez de nouveau le jeu de données en spécifiant les valeurs à considérer comme données manquantes `NA`.
5. Remplacer les données manquantes par la moyenne de la variable en question.

2.5 Importation depuis une URL

Il est aussi possible d'importer un jeu de données directement depuis une URL. Exécutez le commandes suivantes :

```
> Dwiki <- read.csv("http://tecfa.unige.ch/guides/R/data/edutechwiki-fr-gw-oct-6-2014.csv",
 header = TRUE, sep= ",")
> View(Dwiki)
```

2.6 Importation problématique

Cet exemple est tiré du livre *Statistiques avec R* de Cornillon et al.

1. Importez le jeu de données `donnees`.
   Pour une importation délicate, il est parfois nécessaire de passer par la fonction `scan` afin d'identifier (et résoudre) les problèmes.
2. Importez le fichier dans un vecteur de caractères à l'aide de la fonction `scan`.
3. Récupérez le nom des variables dans un objet `nomcol`.
4. Changez le séparateur décimal de "," à "." à l'aide de la fonction `gsub`.
5. Constituez une matrice de 4 lignes et 5 colonnes avec le vecteur de caractères importé, tel que la première ligne contient `nomcol`.
6. Affectez le nom des individus dans un objet `nomlign`
7. Créez une matrice contenant uniquement les données et la transformer en data frame. Affectez les noms des variables et des individus au data frame obtenu.
8. Vérifiez (et éventuellement corrigez) le type des colonnes.

3 Exportation de jeux de données

Plutôt que d'importer un jeu de données, on peut être amené à vouloir sauvegarder un jeu de données nouvellement créé, par exemple, après des opérations de nettoyage d'un jeu de données. Plusieurs solutions sont possibles : soit à l'aide de la fonction `write.table` pour une sauvegarde dans un format ASCII, soit à l'aide de la fonction `save` pour une sauvegarde dans le format binaire Rdata.

1. Reprenez le jeu de données résultant de la fusion entre `etat1` et `etat2`. Retirer la ligne correspondant à l'état d'Hawaï. On conservera cette ligne dans un objet `etat.supp`.
2. Sauvegarder ce jeu de données dans un fichier txt à l'aide de la fonction `write.table`.
3. Ajouter dans ce même fichier la ligne correspondant à l'état d'Hawaï. On pourra utiliser l'argument `append`.
4. Reprenez le jeu de données `bladder`. Quelle est sa taille sur le disque (en octets) ?
5. Sauvegarder ce même jeu de données au format binaire `.Rdata` à l'aide de la fonction `save`.
6. Quelle est à présent sa taille sur le disque (en octets) ?
7. Effacez les objets en mémoire dans votre environnement de travail à l'aide de la commande `rm`.
8. Ré-importez le jeu de données `bladder` depuis le fichier `.Rdata` avec la fonction `load`. Affichez la dimension de la matrice.

4 Jeu de données `accident`

Le site institutionnel `https://www.data.gouv.fr/` met à disposition des utilisateurs de nombreux jeux de données. L'un de ces projets concerne les accidents corporels survenus sur les routes de France Métropolitaine et d'Outre-Mer (adresse ici). Pour chaque année, 4 bases de données sont disponibles : une nommée **caractéristiques** qui décrit les circonstances générales de l'accident, une nommée **lieux** qui

décrit le lieu précis de l'accident, une nommée **véhicules** qui décrit les véhicules impliqués dans l'accident et une nommée **usagers** qui décrit plus particulièrement les victimes de l'accident. La description des bases de données est détaillée dans le document suivant : description
On souhaiterait effectuer une typologie des accidents de la route ayant eu lieu en 2016. Pour cela, nous allons construire un nouveau jeu de données à partir des 4 jeux de données cités précédemment.

1. Importez le jeu de données `caracteristiques_2016.csv` directement depuis son url. Quelle est la taille du jeu de données ?
2. À l'aide des variables `an`, `mois`, `jour` et `hrmn`, créez une variable `dateheure` donnant la date et l'heure de l'accident au format date. On pourra utiliser les fonctions `paste` et `strptime`.
3. Dans le jeu de données caractéristiques, on ne conservera que les variables `dateheure`, `int`, `atm` et `col`. Ces variables possèdent-elles des données manquantes ?
4. Charger le jeu de données `lieux_2016.csv`. Pour ce jeu de données, on ne conservera que les variables `catr` et `surf`.
5. Charger à présent le jeu de données `usagers_2016.csv`. Quelle est sa dimension ?
6. Dans le jeu de données `Usagers`, on s'intéresse à la variable `grav`. Quels sont ses différents niveaux ? Transformez cette variable en facteur.
7. Créez un `data.frame gravite` avec 4 variables correspondant aux 4 niveaux de gravité et comptant, par accident, le nombre de victimes correspondant à chaque degré de gravité (tableau de contingence, on peut utiliser `table()`).
8. Fusionnez à présent les jeux de données `caracteristiques`, `lieux` et `gravite` afin de créer un nouveau jeu de données `accidents`. Le sauvegarder.
9. Quelle est la catégorie de route qui occasionne le plus de tués ?
10. Y a-t-il eu plus d'accidents l'été ou l'hiver ?