Objets dans R

Fousseynou Bah

19-Jun-2020

Section 1

Introduction

Objectif de ce cours

Dans ce cours, nous allons :

- introduire la notion d'objet;
- présenter un certain nombre qui sont manipulables dans R;
- et illustrer avec quelques exemples.

Que nous faut-il?

- R (évidemment)
- RStudio (de préférence)

Section 2

La notion d'objet dans R

Qu'est-ce qu'un objet?

Un objet représente un concept, une idée. Dans R, il se matérialise par une entité qui possède sa propre identité. Dans celle-ci, l'on compte deux aspects majeurs:

- la structure interne;
- le comportement.

Illustrons pour comprendre

Création d'objets

```
# Imaginez que vous voulez créer et conserver des bouts d'information dans R
# sur les présidents qui se sont succédés à la tête de la République du Mali.
# Commençons par le premier président, Mobido Keïta.
# Créeons des objects relatifs à son nom et son prénom.
nom <- "Keïta"
prenom <- "Mobido"
# Désormais, ces informations sont stockées dans notre environnement.
# Pour vérifier appellons-les!
nom
```

```
## [1] "Keïta"
```

```
prenom
```

```
## [1] "Mobido"
```

Oranges et bananes

```
# Enrichissons notre environnement des objets additionnels
# Ajoutons l'année d'accession au pouvoir
# Appelons-le: annee_arrivee_pouvoir
annee_arrivee_pouvoir <- 1960
# Comme pour les objets précédent celui-ci aussi peut être invoqué:
annee_arrivee_pouvoir
```

```
## [1] 1960
```

```
# A l'instar de l'orange et de la banane, fort différentes bien que
# toutes les deux des fruits, ici aussi nos objets diffèrent.
# Peut-on les additionner?
nom + annee_arrivee_pouvoir
```

```
## Error in nom + annee_arrivee_pouvoir: non-numeric argument to binary operator
```

```
# Non, en l'occurence! On a message d'erreur.
# R, c'est comme la vraie vie!
```

Ce qui se ressemblent s'assemblent

```
# Les choses qui diffèrent ne s'assemblent pas
# Illustration d'une propriété des objets: le comportement
# Regardons les choses qui marchent
1 + 1
```

```
## [1] 2
```

```
# Stockons ce résultat dans un objet
objet1 <- 1 + 1
# Créons un autre
objet2 <- 2 + 2
# Amusons à faire diverses opérations avec ces objets
objet1 + objet2
```

```
## [1] 6
```

```
objet1 - objet2
```

```
## [1] -2
```

```
objet1 * objet2
```

```
## [1] 8
```

```
objet1 / objet2
```

```
## [1] 0.5
```

Quelques objets dans R

Bref, vous voyez l'idée! Les propriétés des objets déterminent les intéractions auxquelles elles se prêtent. Et ce sont justement ces intéractions qui constituent le coeur de l'analyse de données. D'où l'importance de la notion d'objet.

Dans R, l'on distingue plusieurs types d'objets. Nous en retiendrons ici 6:

```
# Les charactères (strings en anglais), que l'on déjà vu
nom <- "Keïta"
prenom <- "Mobido"
# Les nombres (entiers ou réels), dont on a vu un cas aussi
annee_arrivee_pouvoir <- 1960
# Les dates. Ajoutons la date de naissance.
date_naissance <- as.Date("1915-06-04")
# Les booléens: TRUE (vrai en anglais) et FALSE (faux en anglais)
# Ce président était-il militaire? Non, donc FALSE
parcours_militaire <- FALSE
# Les facteurs (factor), des variables catégorielles
# Considérons ici la région de naissance:
region_naissance <- as.factor("Bamako")
```

La notion de classe (1)

Dans R, l'on peut déterminer le type d'objet qu'on a avec la fonction `class()`. Regardons les classes des objets créés, pour bien confirmer les identités qu'on leur a attribuées.

```
class(nom)
```

```
## [1] "character"
```

```
class(prenom)
```

```
## [1] "character"
```

```
class(annee_arrivee_pouvoir)
```

```
## [1] "numeric"
```

```
class(date_naissance)
```

```
## [1] "Date"
```

La notion de classe (2)

```
class(parcours_militaire)
```

```
## [1] "logical"
```

```
class(region_naissance)
```

```
## [1] "factor"
```

Vers d'autres types d'objets

Ces objets peuvent, à leur tour, être agrégés pour former d'autres types d'objets.

Nous en verrons quatre types:

- le vecteur;
- la matrice;
- le data frame (cadre de données ou données rectangulaires);
- la liste.

Section 3

Vecteurs

Qu'est-ce qu'un vecteur?

De façon basique, un vecteur est un ensemble d'éléments de même nature. Revenons à notre exemple.

```
# A quoi sait-on qu'un objet est un vecteur?
# On demande tout simplement:
is.vector(nom)
```

```
## [1] TRUE
```

```
# Donc nous avons créé des vecteurs depuis longtemps et
# on voit qu'un objet d'un seul élément peut être un vecteur.
# On connait le nombre d'élément dans un objet à la fonction suivante:
length(nom)
```

```
## [1] 1
```

```
# C'est vraiment un singleton qu'on a là...pour le moment!
```

Créons-en, des vecteurs!

```
# Décidons d'étendre nos observations à tous les présidents de la République du Mali.
# Omettons les périodes de transition (la valeur pédagogique est ce qui est recherché ici!)
nom <- c("Keïta", "Traoré", "Konaré", "Touré", "Keïta")
prenom <- c("Modibo", "Moussa", "Alpha Oumar", "Amadou Toumani", "Ibrahim Boubacar")
date_naissance <- as.Date(c("1915-06-04", "1936-09-25", "1946-02-02", "1948-11-04", "1945-01-29"))
region_naissance <- as.factor(c("Bamako", "Kayes", "Kayes", "Mopti", "Koutiala"))
annee_arrivee_pouvoir <- c(1960, 1968, 1992, 2002, 2013)
parcours_militaire <- c(FALSE, TRUE, FALSE, TRUE, FALSE)
# Maintenant, expérimentons!
is.vector(nom)
```

```
## [1] TRUE
```

```
length(nom)
```

```
## [1] 5
```

```
class(nom)
```

```
## [1] "character"
```

```
typeof(nom)
```

```
## [1] "character"
```

```
# "nom" est un "vecteur", un ensemble de "5" éléments en "charactères"
```

Vrai pour un, vrai pour plusieurs (1)

```
# Aussi bien à l'échelle d'un élément que de plusieurs,
# les opérations dans R ne s'effectuent qu'entre les éléments de même nature.
# On sait par exemple que le "nom" et le "prénom" sont tous les deux des vecteurs de charactères.
# Collons le nom et le prénom avec une fonction de base dans R
# (Ne vous en faites pas! Vous ferez progressivement connaissance avec les fonctions!)
prenom_nom <- paste(prenom, nom)
# Maintenant imprimons le résultat en entrant tout simplement le nom du nouveau vecteur.
prenom_nom
```

```
## [1] "Modibo Keïta"           "Moussa Traoré"          "Alpha Oumar Konaré"
## [4] "Amadou Toumani Touré"   "Ibrahim Boubacar Keïta"
```

```
# Regardons ses caractéristiques
is.vector(prenom_nom)
```

```
## [1] TRUE
```

```
length(prenom_nom)
```

```
## [1] 5
```

```
class(prenom_nom)
```

```
## [1] "character"
```

Vrai pour un, vrai pour plusieurs (2)

```
# Un autre exemple!
# Cherchons à déterminer l'âge de ces hommes à leur arrivée au pouvoir.
# On a les éléments nécéssaires pour ce faire, la date de naissance et
# l'année d'arrivée au pouvoir, mais ces deux vectors ne sont pas de même nature
age_arrivee_pouvoir <- annee_arrivee_pouvoir - date_naissance
```

```
## Error in `-.Date`(annee_arrivee_pouvoir, date_naissance): can only subtract from "Date" objects
```

```
# L'opération n'est pas possible sans une transformation
annee_naissance <- as.numeric(format(date_naissance,'%Y'))
# Testons si le nouveau vecteur est de même nature de celui de "annee_arrivee_pouvoir"
class(annee_naissance)
```

```
## [1] "numeric"
```

```
# Maintenans, nous pouvons procéder à l'opération
age_arrivee_pouvoir <- annee_arrivee_pouvoir - annee_naissance
age_arrivee_pouvoir
```

```
## [1] 45 32 46 54 68
```

Nommer les éléments d'un vecteur

```
# Jusque là, ce sont des objets à part intégrale que nous avons nommées.
# Maintenant, nous allons donner un nom aux éléments de vecteur.
# Considerons que nous voulons associer à chaque date de naissance
# le nom du président en question.
names(date_naissance) <- prenom_nom
# Voyons ce que ça donne
date_naissance
```

```
##           Modibo Keïta          Moussa Traoré     Alpha Oumar Konaré
##           "1915-06-04"           "1936-09-25"           "1946-02-02"
##   Amadou Toumani Touré Ibrahim Boubacar Keïta
##           "1948-11-04"           "1945-01-29"
```

```
# On peut même conduire des opérations pendant que le vecteur a ses éléments nommés
# Le résultat issu deux vecteurs aux noms conformes hérite de ceux-ci
# Reprenons l'opération de déduction de l'âge à l'arrivée au pouvoir
names(annee_naissance) <- prenom_nom
names(annee_arrivee_pouvoir) <- prenom_nom
age_arrivee_pouvoir <- annee_arrivee_pouvoir - annee_naissance
age_arrivee_pouvoir
```

```
##           Modibo Keïta          Moussa Traoré     Alpha Oumar Konaré
##                     45                     32                     46
##   Amadou Toumani Touré Ibrahim Boubacar Keïta
##                     54                     68
```

Opérations sur vecteurs: logiques et sélections

```
# Généralement, l'analyse commence par des questions.
# Parmi celles-ci, celles de logique,
# celles auxquelles les réponses sont OUI/NON, VRAI/FAUX
# Explorons une question ici:
# Quels sont les présidents arrivés au pouvoir avant l'âge de 50 ans?
president_avant_50ans <- age_arrivee_pouvoir < 50
president_avant_50ans
```

```
##           Modibo Keïta          Moussa Traoré     Alpha Oumar Konaré
##                   TRUE                   TRUE                   TRUE
##   Amadou Toumani Touré Ibrahim Boubacar Keïta
##                  FALSE                  FALSE
```

```
# Ce résultat de logique peut aussi servir de critère de sélection.
# Les présidents qui répondent à ce critère, quel âge avaient-ils lors de l'accession au pouvoir?
age_arrivee_pouvoir[age_arrivee_pouvoir < 50]
```

```
##       Modibo Keïta      Moussa Traoré Alpha Oumar Konaré
##                 45                 32                 46
```

```
# Pendant qu'on y est, dans quelle région sont-ils nés?
names(region_naissance) <- prenom_nom # nommons d'abord les éléments
region_naissance[age_arrivee_pouvoir < 50]
```

```
##       Modibo Keïta      Moussa Traoré Alpha Oumar Konaré
##             Bamako              Kayes              Kayes
## Levels: Bamako Kayes Koutiala Mopti
```

Opérations sur vecteurs: sélection explicite

```
# Il est possible qu'on ne soit intéréssée que par un élément précis d'un vecteur.
# Peut-être l'on souhaite connaitre seulement l'âge du premier président lors de son
# accès au pouvoir. C'est le premier élément du vecteur.
age_arrivee_pouvoir[1]
```

```
## Modibo Keïta
##           45
```

```
# Peut-être nous voulons l'information pour le 1er et le 3ème présidents.
# Ce sont les 1er et 3ème éléments du vecteur
age_arrivee_pouvoir[c(1, 3)]
```

```
##       Modibo Keïta Alpha Oumar Konaré
##                 45                 46
```

```
# On peut aussi souhaiter exclure certains éléments
# Imaginons que l'on veuille seulement regarder:
#  les informations sans les deux derniers éléments du vecteur.
age_arrivee_pouvoir[-c(4, 5)]
```

```
##       Modibo Keïta      Moussa Traoré Alpha Oumar Konaré
##                 45                 32                 46
```

Opérations sur vecteurs: statistiques sommaires

```
# Une fois le vecteur constitué, il peut en lui-même faire l'objet d'opérations diverses.
# Posons diverses questions avec le vecteur "age_arrivee_pouvoir".
# 1. Quelle est la moyenne d'âge d'arrivée au pouvoir sur la base des éléments disponibles?
mean(age_arrivee_pouvoir)
```

```
## [1] 49
```

```
# ou cette formule qui donne le même résultat.
sum(age_arrivee_pouvoir)/length(age_arrivee_pouvoir)
```

```
## [1] 49
```

```
# 2. Quel est l'âge d'arrivée au pouvoir le plus bas ?
min(age_arrivee_pouvoir)
```

```
## [1] 32
```

```
# 3. Quel est l'âge d'arrivée au pouvoir le plus élevé ?
max(age_arrivee_pouvoir)
```

```
## [1] 68
```

Opérations sur vecteurs: ajustement et recyclage

```
# Maintenant, revenons-en un peu aux opérations entre deux vecteurs.
# Imaginez maintenant, que l'on veuille connaître l'âge auquel les présidents ont quitté le pouvoir.
# Rappellons d'abord le vecteur "age_arrivee_pouvoir" (on avait déjà calculé ça!)
age_arrivee_pouvoir <- c(45, 32, 45, 54, 68)
# Construisons ensuite un vecteur avec le nombre d'années passées au pouvoir.
annee_en_pouvoir <- c(8, 23, 10, 10)
# Maintenant calculons l'année de départ du pouvoir
age_depart_pouvoir <- age_arrivee_pouvoir + annee_en_pouvoir
```

```
## Warning in age_arrivee_pouvoir + annee_en_pouvoir: longer object length is not a
## multiple of shorter object length
```

```
# Examinons le nouvel objet créé
age_depart_pouvoir
```

```
## [1] 53 55 55 64 76
```

```
# Parvenez-vous à déceler l'erreur? Nous avons additionné une vecteur
# de 5 éléments (age_arrivee_pouvoir) avec un vecteur de 4 élément (annee_en_pouvoir).
# R a récyclé le premier élément du vecteur court (4) pour réitérér
# l'opération d'addition sur le 5ème élément du vecteur long
68 + 8
```

```
## [1] 76
```

```
# R avertit, mais conduit l'opération.
# De ce fait, même si les opérations entre les vecteurs de même nature
# s'exécute sans problème majeur, il reste utile de vérifier leur taille.
```

Section 4

Matrices

La matrice, un ensemble de vecteurs (1)

```
# De façon basique, une matrice n'est autre qu'une collection de vecteurs.
# De ce fait, la matrice hérite d'une propriété fondamentale du vecteur:
# ne peuvent former une matrice que des éléments de même nature.
# Retournons à notre exemple:
# Associons les noms et prénoms en une matrice car tous deux sont en charactères

# Solution 1: coller horizontalement les deux vecteurs
prenom_nom_hmatrix <- rbind(prenom, nom)
prenom_nom_hmatrix
```

```
##        [,1]     [,2]     [,3]          [,4]             [,5]
## prenom "Modibo" "Moussa" "Alpha Oumar" "Amadou Toumani" "Ibrahim Boubacar"
## nom    "Keïta"  "Traoré" "Konaré"      "Touré"          "Keïta"
```

```
# Solution 2: coller verticalement les deux vecteurs
prenom_nom_vmatrix <- cbind(prenom, nom)
prenom_nom_vmatrix
```

```
##      prenom             nom
## [1,] "Modibo"           "Keïta"
## [2,] "Moussa"           "Traoré"
## [3,] "Alpha Oumar"      "Konaré"
## [4,] "Amadou Toumani"   "Touré"
## [5,] "Ibrahim Boubacar" "Keïta"
```

```
# On voit que la matrice hérite des noms donnés aux différents vecteurs.
```

La matrice, un ensemble de vecteurs (2)

```r
# la même matrice peut-être crééée à partir de rien: horizontalement
prenom_nom_hmatrix <- matrix(c("Modibo", "Moussa", "Alpha Oumar", "Amadou Toumani", "Ibrahim Boubacar",
                               "Keïta",  "Traoré", "Konaré", "Touré", "Keïta"),
                             byrow = TRUE,
                             nrow = 2,
                             dimnames = list(c("prenom", "nom"), NULL)
                             )
# ou verticalement
prenom_nom_vmatrix <- matrix(c("Modibo", "Keïta",
                               "Moussa", "Traoré",
                               "Alpha Oumar", "Konaré",
                               "Amadou Toumani", "Touré",
                               "Ibrahim Boubacar", "Keïta"),
                             byrow = TRUE,
                             ncol = 2,
                             dimnames = list(NULL, c("prenom", "nom"))
                             )
# Il existe une fonction qui permet de créer directement une matrice.
# Il est toutefois utile de connaitre l'ordre de positionnement des éléments
# Dans la fonction "matrix", les arguments "nrow", "ncol", "byrow" et "bycol" servent à ça.
# (fonction, arguments...ne vous en faites pas! On y viendra).
```

La matrice, un objet bidimensionnel (1)

```
# La matrice n'est pas seulement un ensemble de vecteurs.
# Elle se distingue de par sa bidimensionnalité
# Elle est faite de lignes (rows) et de colonnes (columns)
# Nous avions noté que pour connaître le nombre d'éléments dans un vecteur on faisait:
length(prenom_nom)
```

```
## [1] 5
```

```
# La même chose marche-t-elle pour le vecteur?
length(prenom_nom_vmatrix)
```

```
## [1] 10
```

```
# En l'occurence, non! "length" ne rend pas compte de la bidimensionnalité
# Il y a une autre fonction pour ça
dim(prenom_nom_vmatrix)
```

```
## [1] 5 2
```

```
# La bidimensionnalité se lit aussi dans le nom des rangées
dimnames(prenom_nom_vmatrix)
```

```
## [[1]]
## NULL
##
## [[2]]
## [1] "prenom" "nom"
```

```
knitr::asis_output("\\normalsize")
```

La matrice, un objet bidimensionnel (2)

```
# Reprenons la création de "prenom_nom_vmatrix"
# Nommons toutes les rangées
prenom_nom_vmatrix <- matrix(c("Modibo", "Keïta",
                               "Moussa", "Traoré",
                               "Alpha Oumar", "Konaré",
                               "Amadou Toumani", "Touré",
                               "Ibrahim Boubacar", "Keïta"),
                             byrow = TRUE,
                             ncol = 2,
                             dimnames = list(c("1er", "2ème", "3ème", "4ème", "5ème"),
                                             c("prenom", "nom"))
                             )
# Imprimons la matrice
prenom_nom_vmatrix
```

```
##      prenom             nom
## 1er  "Modibo"           "Keïta"
## 2ème "Moussa"           "Traoré"
## 3ème "Alpha Oumar"      "Konaré"
## 4ème "Amadou Toumani"   "Touré"
## 5ème "Ibrahim Boubacar" "Keïta"
```

```
# Amusez-vous maintenant en regardant
# les dimensions et les noms des rangées avec
# les fonctions "dim()" et "dimnames()"
# Testez aussi les fonctions: "rownames()" et "colnames()".
# Qu'observez-vous?
```

Opérations sur matrices: une autre matrice

```
# Reprenons les années et les âges pour former une nouvelle matrice
# Considérons les années d'évènements majeurs: naissance, arrivée au pouvoir et départ du pouvoir
annee_evenement_matrix <- matrix(c(1915, 1936, 1946, 1948, 1945,
                                   1960, 1968, 1992, 2002, 2013,
                                   1968, 1991, 2002, 2012, NA),
                                 byrow = TRUE,
                                 ncol = 5,
                                 dimnames = list(c("Naissance", "Arrivée", "Départ"),
                                                 c("M. Keïta", "M. Traoré", "A.O. Konaré", "A.T. Touré", "I.B.
                                                 )
                                 )
annee_evenement_matrix
```

```
##           M. Keïta M. Traoré A.O. Konaré A.T. Touré I.B. Keïta
## Naissance     1915      1936        1946       1948       1945
## Arrivée       1960      1968        1992       2002       2013
## Départ        1968      1991        2002       2012         NA
```

```
# Nous venons d'introduire une nouvelle notion: "NA"
# NA: 'Non Available' en anglais
# Cette valeur rempli la cellule dont les valeurs nous sont inconnues.
```

Opérations sur matrices: questions logiques

```
# Maintenant que nous avons notre matrice, amusons-nous avec.
# Prenons un grand-père née vers 1949 (oui, il fait partie des "né vers").
# Marié à l'âge de 22 ans, père 1 an plus tard, grand-père 18 ans plus tard et décédé à l'âge de 61 ans.
# Quels sont les évènements qui se sont passés de son vivant?
ce_que_grandpa_a_vu <- annee_evenement_matrix > 1949 & annee_evenement_matrix < (1949 + 61)
ce_que_grandpa_a_vu
```

```
##           M. Keïta M. Traoré A.O. Konaré A.T. Touré I.B. Keïta
## Naissance    FALSE     FALSE       FALSE      FALSE      FALSE
## Arrivée       TRUE      TRUE        TRUE       TRUE      FALSE
## Départ        TRUE      TRUE        TRUE      FALSE         NA
```

```
# Apparemment, il en a vu beaucoup, mais tous les présidents le dépassent en âge
# On vient d'introduier ici la notion d'addition dans les critères.
```

Opérations sur matrices: extraction par position

```
# Comme pour les vecteurs, des éléments peuvent être explicitement sélectionnés à l'intérieur des matrices.
annee_evenement_matrix
```

```
##           M. Keïta M. Traoré A.O. Konaré A.T. Touré I.B. Keïta
## Naissance     1915      1936        1946       1948       1945
## Arrivée       1960      1968        1992       2002       2013
## Départ        1968      1991        2002       2012         NA
```

```
# Supposons que l'on veuille connaître l'élément qui est dans la cellule de la 3ème ligne et le 2ème colonne.
annee_evenement_matrix[3, 2]
```

```
## [1] 1991
```

```
# Et la 3ème ligne toute entière.
annee_evenement_matrix[3, ]
```

```
##    M. Keïta   M. Traoré A.O. Konaré  A.T. Touré  I.B. Keïta
##        1968        1991        2002        2012          NA
```

```
# Et la 2ème colonne toute entière.
annee_evenement_matrix[ , 2]
```

```
## Naissance   Arrivée    Départ
##      1936      1968      1991
```

Opérations sur matrices: extraction par nom

```
# Rappel
annee_evenement_matrix
```

```
##           M. Keïta M. Traoré A.O. Konaré A.T. Touré I.B. Keïta
## Naissance     1915      1936        1946       1948       1945
## Arrivée       1960      1968        1992       2002       2013
## Départ        1968      1991        2002       2012         NA
```

```
# Si les rangées sont nommés, alors il est aussi possible de passer par ces noms pour sélectionner
# Vous vous rappelez "rownames()" ou "colnames()"
# Si la réponse est non, je saurai que vous n'avez pas tout suivi!
# Utilisons "rownames()" pour voir la ligne sur les années de naissance.
annee_evenement_matrix[rownames(annee_evenement_matrix) == "Naissance", ]
```

```
##    M. Keïta   M. Traoré A.O. Konaré  A.T. Touré  I.B. Keïta
##        1915        1936        1946        1948        1945
```

```
# Et "colnames()" pour voir la colonne du premier président
annee_evenement_matrix[, colnames(annee_evenement_matrix) == "M. Keïta"]
```

```
## Naissance   Arrivée    Départ
##      1915      1960      1968
```

Opérations sur matrices: consolidation

```
# Il arrive qu'on souhaite consolider une matrice, y ajouter de nouvelles informations.
# Considérons ici les âges à l'arrivée au et au départ du pouvoir.
# Recalculons les à partir des matrices.
age_arrivee_pouvoir <- annee_evenement_matrix[rownames(annee_evenement_matrix) == "Arrivée", ] -
  annee_evenement_matrix[rownames(annee_evenement_matrix) == "Naissance", ]
age_depart_pouvoir <- annee_evenement_matrix[rownames(annee_evenement_matrix) == "Départ", ] -
  annee_evenement_matrix[rownames(annee_evenement_matrix) == "Naissance", ]
# Ajoutons maintenant c'est deux nouveaux vecteurs à notre matrice
annee_evenement_matrix_cons <- rbind(annee_evenement_matrix,
                                     "Âge d'arrivée au pouvoir" =  age_arrivee_pouvoir,
                                     "Âge de départ du pouvoir" = age_depart_pouvoir)
# Voyons la matrice
annee_evenement_matrix_cons
```

```
##                           M. Keïta M. Traoré A.O. Konaré A.T. Touré I.B. Keïta
## Naissance                     1915      1936        1946       1948       1945
## Arrivée                       1960      1968        1992       2002       2013
## Départ                        1968      1991        2002       2012         NA
## Âge d'arrivée au pouvoir        45        32          46         54         68
## Âge de départ du pouvoir        53        55          56         64         NA
```

```
# Nous sommes passés par la fonction "rbind()". Sachez qu'il y a plusieurs solutions!
# Remarquez-vous "NA" dans une nouvelle cellule? Pouvez-vous expliquer pourquoi?
```

Opérations sur matrices: calculs

```
# Comme pour les vecteurs, des calculs sont possibles sur les matrices.
# Pour ce faire, limitons-nous à deux informations de la matrice: les âges.
# Ajoutons aussi la durée de la période passée au pouvoir.
age_pouvoir_matrix <- rbind(annee_evenement_matrix_cons[c(4, 5),],
                            "Durée au pouvoir" = annee_evenement_matrix_cons[5, ] -
                              annee_evenement_matrix_cons[4, ])
age_pouvoir_matrix[, -c(1)] # -c(1): juste pour une commodité d'impression.
```

```
##                          M. Traoré A.O. Konaré A.T. Touré I.B. Keïta
## Âge d'arrivée au pouvoir        32          46         54         68
## Âge de départ du pouvoir        55          56         64         NA
## Durée au pouvoir                23          10         10         NA
```

```
# Calculons la moyenne pour chacune des lignes (âges moyens, durées moyennes)
rowMeans(age_pouvoir_matrix)
```

```
## Âge d'arrivée au pouvoir Âge de départ du pouvoir         Durée au pouvoir
##                       49                        NA                       NA
```

```
# On voit des "NA". R ne traite pas les valeurs inconnues de lui-même. On lui instruit de les ignorer.
rowMeans(age_pouvoir_matrix, na.rm = TRUE)
```

```
## Âge d'arrivée au pouvoir Âge de départ du pouvoir         Durée au pouvoir
##                    49.00                    57.00                    12.75
```

```
# Comme pour beaucoup de fonctions dans R, tout ce qu'il y avec "row" existe avec "col"
# Testez les fonctions suivantes: colSums(), rowSums(), colMeans() et rowMeans().
```

Section 5

Data frames

Le *data frame*, au-délà de la matrice (1)

```
# Jusque là, nous avons travaillé avec des éléments de même nature.
# Et pourtant le data scientist ne peut pleinement mener ses
# ses investigations avec une telle contrainte.
# Il a besoin d'explorer en même temps des informations de diverses natures.
# D'où le data frame. Qu'est-ce que c'est au juste?
# Un format d'organisation de données en forme rectangulaire.
# Toutefois, contrairement à la matrice, elle respecte la nature des données qu'elle contient.
# Explorons l'idée. Rassemblons verticalement les différents vecteurs que nous avons créés
presidents_df <- cbind(nom,
                       prenom,
                       date_naissance,
                       region_naissance,
                       parcours_militaire,
                       annee_arrivee_pouvoir,
                       annee_en_pouvoir)
# Qu'est-ce que ça donne?
presidents_df
```

```
##      nom      prenom             date_naissance region_naissance
## [1,] "Keïta"  "Modibo"           "-19935"       "1"
## [2,] "Traoré" "Moussa"           "-12151"       "2"
## [3,] "Konaré" "Alpha Oumar"      "-8734"        "2"
## [4,] "Touré"  "Amadou Toumani"   "-7728"        "4"
## [5,] "Keïta"  "Ibrahim Boubacar" "-9103"        "3"
##      parcours_militaire annee_arrivee_pouvoir annee_en_pouvoir
## [1,] "FALSE"            "1960"                "8"
## [2,] "TRUE"             "1968"                "23"
## [3,] "FALSE"            "1992"                "10"
## [4,] "TRUE"             "2002"                "10"
## [5,] "FALSE"            "2013"                NA
```

Le *data frame*, au-délà de la matrice (2)

```
# Nous avons vu que certaines informations ont été dénaturées.
# Certaines données ont été coercées à se transformer en autre chose
# Regardons la classe de l'objet "presidents_df"
class(presidents_df)
```

```
## [1] "matrix" "array"
```

```
typeof(presidents_df)
```

```
## [1] "character"
```

```
# Les vecteurs ont été rassemblés en matrice (class)
# Les éléments ont toutefois été coercés en charactères (typeof)
# C'est en celà que le data frame révèle sa place.
```

Le *data frame*, au-délà de la matrice (3)

```
# Reprenons l'opération
presidents_df <- data.frame(nom,
                            prenom,
                            date_naissance,
                            region_naissance,
                            parcours_militaire,
                            annee_arrivee_pouvoir,
                            annee_en_pouvoir,
                            stringsAsFactors = FALSE)
# Regardons à nouveau
presidents_df
```

```
##      nom           prenom date_naissance region_naissance parcours_militaire
## 1  Keïta           Modibo     1915-06-04           Bamako               FALSE
## 2 Traoré           Moussa     1936-09-25            Kayes                TRUE
## 3 Konaré      Alpha Oumar     1946-02-02            Kayes               FALSE
## 4  Touré   Amadou Toumani     1948-11-04            Mopti                TRUE
## 5  Keïta Ibrahim Boubacar     1945-01-29         Koutiala               FALSE
##   annee_arrivee_pouvoir annee_en_pouvoir
## 1                  1960               8
## 2                  1968              23
## 3                  1992              10
## 4                  2002              10
## 5                  2013              NA
```

```
# Qu'en est-il de la classe et du type
class(presidents_df)
```

```
## [1] "data.frame"
```

Le *data frame*, au-délà de la matrice (4)

```r
# Maintenant que nous savons à quoi ressemble un data frame, essayons de le définir.
# Un data frame est une forme d'organisation de données en format rectangulaire où
# les lignes sont des observations et les colonnes des attributs de ceux-ci.
# Ici par exemple, nous organisons diverses informations sur les individus qui
# ont assumé le poste de Président de la République du Mali.
# Chaque ligne sera dédiée à un président et rassemblera tous les informations sur lui (attributs).
# Chaque colonne sera dédiée à un seul attribut et couvrira tous les présidents (observations).
# Introduisons ici la fonction "str" qui permet de visualiser la structure d'un objet dans R
# Elle permettra de rendre compte de toute la richesse du concept de data frame.
str(presidents_df)
```

```
## 'data.frame':    5 obs. of  7 variables:
##  $ nom                 : chr  "Keïta" "Traoré" "Konaré" "Touré" ...
##  $ prenom              : chr  "Modibo" "Moussa" "Alpha Oumar" "Amadou Toumani" ...
##  $ date_naissance      : Date, format: "1915-06-04" "1936-09-25" ...
##  $ region_naissance    : Factor w/ 4 levels "Bamako","Kayes",..: 1 2 2 4 3
##  $ parcours_militaire  : logi  FALSE TRUE FALSE TRUE FALSE
##  $ annee_arrivee_pouvoir: num  1960 1968 1992 2002 2013
##  $ annee_en_pouvoir    : num  8 23 10 10 NA
```

```r
# On a une synthèse: nombre d'observations et nombre de variables - comme avec la fonction dim().
# On voit aussi que pour chaque variable, on a :
# - le nom
# - la classe
# - quelques observations
```

Opérations sur *data frame*: sélection de cellules

```
# En matière de sélection, la data frame hérite beaucoup de la matrice.
# Les principes demeurent
# Si l'on veut la ligne 2 de la colonne 4, on fait:
presidents_df[2, 4]
```

```
## [1] Kayes
## Levels: Bamako Kayes Koutiala Mopti
```

```
# Si l'on veut la ligne 5 (un président, une observation)
presidents_df[2, ]
```

```
##      nom prenom date_naissance region_naissance parcours_militaire
## 2 Traoré Moussa     1936-09-25            Kayes               TRUE
##   annee_arrivee_pouvoir annee_en_pouvoir
## 2                  1968               23
```

```
# Ou encore la colonne 4 (une variable, un attribut)
presidents_df[, 4]
```

```
## [1] Bamako   Kayes    Kayes    Mopti    Koutiala
## Levels: Bamako Kayes Koutiala Mopti
```

Opérations sur *data frame*: sélection de variables

```
# Comme pour la matrice, avec le data frame, on peut utiliser le nom des colonnes (variables)
# pour accéder aux éléments. Regardons juste la variable "date_naissance".
presidents_df[, "date_naissance"]
```

```
## [1] "1915-06-04" "1936-09-25" "1946-02-02" "1948-11-04" "1945-01-29"
```

```
# Le data frame offre en plus une alternative: les variables y sont accessibles avec le signe "$".
presidents_df$date_naissance
```

```
## [1] "1915-06-04" "1936-09-25" "1946-02-02" "1948-11-04" "1945-01-29"
```

```
# Cependant, la première solution se prête à la sélection de plusieurs variables.
presidents_df[, c("date_naissance", "region_naissance")]
```

```
##   date_naissance region_naissance
## 1     1915-06-04           Bamako
## 2     1936-09-25            Kayes
## 3     1946-02-02            Kayes
## 4     1948-11-04            Mopti
## 5     1945-01-29         Koutiala
```

Opérations sur *data frame*: création de variables

```r
# Comme avec les matrices, souvent, l'analyse peut souhaiter ajouter une nouvelle variable à son data frame.
# Procédons comme avec les matrices à la génération de deux nouvelles variables:
# l'âge d'arrivée au pouvoir et l'âge de départ du pouvoir.
# Pour commencer, générons l'année de naissance
presidents_df$annee_naissance <- as.numeric(format(presidents_df$date_naissance,'%Y'))
# Ensuite on génère l'âge d'arrivée au pouvoir
presidents_df$age_arrivee_pouvoir <- presidents_df$annee_arrivee_pouvoir - presidents_df$annee_naissance
# Ensuite l'âge de départ du pouvoir
presidents_df$age_depart_pouvoir <- presidents_df$age_arrivee_pouvoir + presidents_df$annee_en_pouvoir
# Regardons notre nouveau data frame
str(presidents_df)
```

```
## 'data.frame':    5 obs. of  10 variables:
##  $ nom                  : chr  "Keïta" "Traoré" "Konaré" "Touré" ...
##  $ prenom               : chr  "Modibo" "Moussa" "Alpha Oumar" "Amadou Toumani" ...
##  $ date_naissance       : Date, format: "1915-06-04" "1936-09-25" ...
##  $ region_naissance     : Factor w/ 4 levels "Bamako","Kayes",..: 1 2 2 4 3
##  $ parcours_militaire   : logi  FALSE TRUE FALSE TRUE FALSE
##  $ annee_arrivee_pouvoir: num  1960 1968 1992 2002 2013
##  $ annee_en_pouvoir     : num  8 23 10 10 NA
##  $ annee_naissance      : num  1915 1936 1946 1948 1945
##  $ age_arrivee_pouvoir  : num  45 32 46 54 68
##  $ age_depart_pouvoir   : num  53 55 56 64 NA
```

```r
# A travers cette création, on voit comment on peut mener des opérations entre des colonnes d'un data frame.
```

Opérations sur *data frame*: suppression de variables

```
# Dans notre exemple, nous avons créé l'année de naissance comme étape transitoire vers une
# autre variable. Sachant que nous avons la même information dans la date de naissance,
# l'on peut éviter la redondance, donc la supprimer.
# Comment s'y prend-on dans R?
presidents_df$annee_naissance <- NULL
# Vérifions si cette colonne est partie.
str(presidents_df)
```

```
## 'data.frame':    5 obs. of  9 variables:
##  $ nom                  : chr  "Keïta" "Traoré" "Konaré" "Touré" ...
##  $ prenom               : chr  "Modibo" "Moussa" "Alpha Oumar" "Amadou Toumani" ...
##  $ date_naissance       : Date, format: "1915-06-04" "1936-09-25" ...
##  $ region_naissance     : Factor w/ 4 levels "Bamako","Kayes",..: 1 2 2 4 3
##  $ parcours_militaire   : logi  FALSE TRUE FALSE TRUE FALSE
##  $ annee_arrivee_pouvoir: num  1960 1968 1992 2002 2013
##  $ annee_en_pouvoir     : num  8 23 10 10 NA
##  $ age_arrivee_pouvoir  : num  45 32 46 54 68
##  $ age_depart_pouvoir   : num  53 55 56 64 NA
```

```
# Mission accomplie!
```

Opérations sur *data frame*: sélection d'observations

```
# Nous avons vu que comme la matrice, les éléments du data frame sont accessibles grâce aux numéros de lignes.
# Ici, nous allons voir qu'il est aussi possible de passer par des critères spécifiques aux variables
# pour sélectionner des observations.
# Cherchons seulement la date de naissance des présidents nés dans la région de "Kayes".
presidents_df[presidents_df$region_naissance == "Kayes", c("nom", "prenom")]
```

```
##      nom      prenom
## 2 Traoré      Moussa
## 3 Konaré Alpha Oumar
```

```
# Il est possible d'aboutir au même résultat avec une fonction intégrée à R: "subset"
# Expérimentons
subset(x = presidents_df, subset = region_naissance == "Kayes", select = c(nom, prenom))
```

```
##      nom      prenom
## 2 Traoré      Moussa
## 3 Konaré Alpha Oumar
```

```
# Vous voyez?
# Avec R, tous les chemins mènent...à Roundé.
# (Rome est trop loin pour moi! Même s'il comment par R).
```

Opérations sur *data frame*: ordonner les observations

```
# On peut souvent souhaiter  ordonner son data frame selon une variable donnée.
# Rearrangeons nos données selon l'année de naissance des présidents
ordre_age <- order(presidents_df$date_naissance)
ordre_age
```

```
## [1] 1 2 5 3 4
```

```
# A l'aide de ce classement, regardons les nom, prénom et date de naissance
presidents_df[ordre_age, c("nom", "prenom", "date_naissance")]
```

```
##      nom           prenom date_naissance
## 1  Keïta           Modibo     1915-06-04
## 2 Traoré           Moussa     1936-09-25
## 5  Keïta Ibrahim Boubacar     1945-01-29
## 3 Konaré      Alpha Oumar     1946-02-02
## 4  Touré   Amadou Toumani     1948-11-04
```

Data frame: le meilleur reste à venir

Le *data frame* est la pièce maîtresse de l'analyse dans R, comme dans beaucoup d'autres langages. D'ailleurs, d'autres langages ont développé des concepts similaires. En prenant Python par exemple, on trouve la notion de *DataFrame*, une adaption du concept de *data frame* tel que défini dans R. Pour dire combien l'idée englobée dans le *data frame* est puissante. D'où son rôle capital dans le reste de ce cours.

C'est avec le *data frame* que nous:

- procéderons à des manipulations de données: du nettoyage à la transformation ;
- explorerons des données par la visualisation ;
- introduirons l'application de modèles à des données.

Section 6

Listes

Les listes, que faire de l'ordre et de la structure?

```
# La liste (list en anglais et dans R) apporte elle aussi sa particularité.
# Elle permet de créer un espace pour les données non structurées dans R
# Créons de nouveaux éléments
# Commençons par les pays voisins du Mali: un vecteur en caractères
voisins_vec_char <- c( "Algérie", "Burkina-Faso", "Côte d'Ivoire", "Guinée", "Mauritanie", "Niger", "Sénégal")
# Ajoutons des données sur le population (à partir de 1976, 1987, 1998 et 2009)
# ça nous fait une matrice en nombres (entiers).
population_matrix_int <- matrix(data = c(3123733, 3269185, 6392918,
                                         3760711, 3935638, 7696349,
                                         4856023, 4954889, 9810912,
                                         7204990, 7323672, 14528662),
                                byrow = TRUE,
                                nrow = 4,
                                dimnames = list(c(1976, 1987, 1998, 2009),
                                                c("Hommes", "Femmes", "Total")))
# Ajoutons un dernier élément: lesquels de nos présidents sont encore vivants?
# Mettons ça sous forme boléén.
presidents_en_vie_vec_logi <- c(FALSE, TRUE, TRUE, TRUE, TRUE)

# Nous avons là un beau monde. Rassemblons tout ça dans une liste!
mali_list <- list(presidents = presidents_df,
                  voisins = voisins_vec_char,
                  population = population_matrix_int,
                  presidents_en_vie = presidents_en_vie_vec_logi)
# De par leurs différences en nature, forme et taille, rien ne prédispose ses objets à être contenus
# dans le même objet! Et pourtant ça tient dans notre liste.
# Explorons-là!
```

Listes, un contenant de contenants (1)

```
# Commençons par la structure de la liste. Que voit-on?
str(mali_list)
```

```
## List of 4
##  $ presidents       :'data.frame':   5 obs. of  9 variables:
##   ..$ nom                 : chr [1:5] "Keïta" "Traoré" "Konaré" "Touré" ...
##   ..$ prenom              : chr [1:5] "Modibo" "Moussa" "Alpha Oumar" "Amadou Toumani" ...
##   ..$ date_naissance      : Date[1:5], format: "1915-06-04" "1936-09-25" ...
##   ..$ region_naissance    : Factor w/ 4 levels "Bamako","Kayes",..: 1 2 2 4 3
##   ..$ parcours_militaire  : logi [1:5] FALSE TRUE FALSE TRUE FALSE
##   ..$ annee_arrivee_pouvoir: num [1:5] 1960 1968 1992 2002 2013
##   ..$ annee_en_pouvoir    : num [1:5] 8 23 10 10 NA
##   ..$ age_arrivee_pouvoir : num [1:5] 45 32 46 54 68
##   ..$ age_depart_pouvoir  : num [1:5] 53 55 56 64 NA
##  $ voisins          : chr [1:7] "Algérie" "Burkina-Faso" "Côte d'Ivoire" "Guinée" ...
##  $ population       : num [1:4, 1:3] 3123733 3760711 4856023 7204990 3269185 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:4] "1976" "1987" "1998" "2009"
##   .. ..$ : chr [1:3] "Hommes" "Femmes" "Total"
##  $ presidents_en_vie: logi [1:5] FALSE TRUE TRUE TRUE TRUE
```

```
# Qu'en est-il des noms
names(mali_list)
```

```
## [1] "presidents"        "voisins"           "population"
## [4] "presidents_en_vie"
```

```
# Les noms assignés aux objets sont bien reconduits
# Voyons voir si à l'instar des matrices et des data frames, ces noms peuvent être utilisés
# pour accéder aux éléments qui y sont stockés.
```

Listes, un contenant de contenants (2)

```
# Prenons le vecteur sur les pays voisins.
mali_list[["voisins"]]
```

```
## [1] "Algérie"       "Burkina-Faso"  "Côte d'Ivoire" "Guinée"
## [5] "Mauritanie"    "Niger"         "Sénégal"
```

```
# Le même résultat doit être possible par l'ordre de l'objet dans la liste, le 2ème.
mali_list[[2]]
```

```
## [1] "Algérie"       "Burkina-Faso"  "Côte d'Ivoire" "Guinée"
## [5] "Mauritanie"    "Niger"         "Sénégal"
```

```
# Peut-on utiliser le signe "$" comme avec les data frame
mali_list$voisins
```

```
## [1] "Algérie"       "Burkina-Faso"  "Côte d'Ivoire" "Guinée"
## [5] "Mauritanie"    "Niger"         "Sénégal"
```

```
# Donc, on a l'embarra du choix.
```

Listes, un contenant de contenants (3)

```
# Maitenant qu'on peut accéder aux objets à l'intérieur d'un liste,
# qu'en est-il des éléments stockés à l'intérieur de cet objet lui-même.
# Cherchons le 2ème élément du vecteur des pays voisins
mali_list[["voisins"]][2]
```

```
## [1] "Burkina-Faso"
```

```
# Qu'en est-il de "mali_list[[2]]" et de "mali_list$voisins[2]"?
# Testez avec eux pour voir.

# Un autre exemple: la 3ème colonne de la matrice sur la population
mali_list[["population"]][, 3]
```

```
##     1976     1987     1998     2009
##  6392918  7696349  9810912 14528662
```

```
# La ligne suivante aussi marche
mali_list[["population"]][, "Total"]
```

```
##     1976     1987     1998     2009
##  6392918  7696349  9810912 14528662
```

Section 7

Conclusion

Et ce n'est que le début

Avec une introduction à ces objets, on pose les bases de l'analyse de données dans R. Bien que, pour des raisons pédagogiques, chaque objet ait été présenté par rapport aux limites du précédent, ils demeurent tous utiles, chacun avec ses avantages (compétitifs). Il revient au data scientist de connaître quand, où et comment faire intervenir un au lieu des autres. Contribuer à vous outiller pour faire ces choix - parmi tant d'autres - est l'un des objectifs de ce cours.