Tutoriel Lubridate

Gaspard PALAY

9 Novembre 2020

1. Package Lubridate

Gérer les données de date et d'heure avec le package lubridate

Si vous avez déjà dû travailler sur des fichiers comportant des données indiquant le temps, avec des dates, ou des heures (et des minutes et des secondes), ou les deux à la fois, vous savez que ce n'est pas toujours facile à gérer. D'abord, parce que les dates ne sont pas toujours écrites de la même façon. Par exemple, pour le premier mars 2019, on peut trouver "2019-03-01" ou encore 1-mars-2019, ou encore 01/03/2019, etc... La même remarque peut être faite pour les heures, les minutes et les secondes. Et ensuite, parce qu'il faut convertir ces données, qui sont généralement considérées comme des chaînes de caractères, ou des modalités d'une variable catégorielle à l'issue de l'importation dans R, dans un format (ou classe) reconnu comme étant du temps dans R. Ces formats (ou classes) sont notamment le format **POSIXct** et le format Date.

Une fois que ces données indiquant le temps, sont correctement reconnues par R, il est alors possible de les manipuler, de réaliser des analyses descriptives (par année, mois, jour, heure, etc...), de les représenter visuellement, ou encore de réaliser des calculs. Le package lubridate a été spécialement conçu pour gérer ces données de temps et les rendre facilement manipulables.

A noter que lubridate est disponible avec la collection de package TidyVerse en open source. si vous souhaitez installer la totalité du package tidyverse (très utile) prenez la fonction suivante:

```
install.packages("tidyverse")
```

ou simplement le package

```
install.packages("lubridate")
```

1.1 Transformer une chaîne de caractères en date

1.1.1 transformation d'un vecteur de classe chaîne de caractères en un vecteur de classe date

```
class ("05 may 2020")
```

05 Mai 2020 est donc une chaine de charactère, je vais lui indiquer que les éléments composants cette chaine de caractère sont des Jours des Mois et des Années (DMY)

```
jourJ <- lubridate::dmy("30 may 2020")
class(jourJ)
```

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

grace au package Lubridate R transforme la chaine de caractère en date, si j exécute jourJ il le transformera :

```
jourJ
```

```
## [1] "2020-05-30"
```

1.1.2 transformation d'un vecteur de classe chaîne de caractères en un vecteur de classe heure

quand on a des vecteurs qui regroupent à la fois la date et l heure (comme "11/04/2019 14h37" par exemple) on parle de "date-time" Plusieurs fonction sont disponibles dans le package :

```
ymd("2019/04_11")
ymd_hm("2019.04.11 14h37")
ymd_hms("20190407143752")
hms("14h37min52s")
```

Lubridate nous permet aussi de récupérer seulement certains éléments compris dans une date grace aux plusieurs fonctions suivantes :

```
t <- ymd_hms("2020.11.09_17.56.32")
date(t)
hour(t)
minute(t)
second(t)
```

1.2 Manipulation des dates/heures

1.2.1 arrondir une date

Arrondir vers le haut avec la fonction **(ceiling_date())**,

```
t <- ymd_hms("2020.11.09_17.56.32")
ceiling_date(t,"hour")
```

vers le bas **(floor_date())**:

```
t <- ymd_hms("2020.11.09_17.56.32")
round_date(t,"hour")
```

ou vers le plus proche **(round_date())**:

```
t <- ymd_hms("2020.11.09_17.56.32")
round_date(t,"hour")
```

choisir à quelle unité se fait cet arrondi en précisant ainsi:

```
t <- ymd_hms("2019.04.11 14h37min52s")
round_date(t,"minute")
round_date(t,"hour")
round_date(t,"day")
round_date(t,"month")
round_date(t,"year")
```

1.2.2 Calcul des périodes de temps écoulés ou de durée

"diff" nous renseigne sur la "différence de temps" entre t1 et t2. Il s'agit d'un objet de classe difftime

```
t1 <- dmy("12/09/2020")
t2 <- dmy("30/01/2016")
diff <- t1-t2
as.duration(diff)
as.period(diff)
```

si l'on veut connaitre la date d'un évenement dont on connait la récurrence (tous les 38 ans par exemple) on utilisera la fonction suivante :

```
t0 <- dmy_h("12/09/2020 22h")
t0+years(38)
```

Vous remarquez que le résultat est inexacte. Il prédit un événemet le 12/09 sans compter les années bissextiles. Pour obtenir le bon résultat on fait :

```
t0+dyears(38)
```

si vous osuhaitez savoir si une année est bissextile, la fonction suivante le permet :

```
leap_year(2016)
leap_year(2017)
```

1.3 Calculs artithméiques

1.3.1 calculs arithmétiques sur des périodes ou des durées

Non pouvons inclure des calculs arithmétiques sur des périodes ou des durées. Les périodes correspondent au fonctions **xxx()** (par exemple **days()** ou **months()**) tandis que les durées correspondent aux fonctions **dxxx()** (par exemple **ddays()** ou **dyears()**)

```
t1+months(9) # t1 + 9 mois
t1+ddays(287) # t1 + exactement 287 jours
ddays(287)/dweeks(1) # combien de semaines (exactement) pour 287 jours?
t2-dweeks(7) # t2 - 7 semaines
```

1.3.2 Intervalles

Les intervals sont des intervalles de temps qui commencent et finissent à des instants bien déterminés (parce qu'ils sont liés à des dates spécifiques).

Les intervalles conservent des informations complètes sur un intervalle de temps. Les fonctions pour travailler avec l'objet intervals sont **is.interval**, **as.interval**, **interval**, **int_start**, **int_end,int_shift**, **int_flip**, **int_aligns**, **int_overlaps** et **%within%**.

Les intervalles peuvent également être manipulés avec les fonctions suivantes : **intersect, union,** et **setdif**

```
date_depart <- dmy_hm("27/12/2016 5:45", tz="Africa/Dakar")
date_arrive <- mdy_hm("dec 28, 2017  19:30", tz="America/Toronto")
duree <- interval(date_depart, date_arrive)
duree
```

```
## [1] 2016-12-27 05:45:00 GMT--2017-12-29 00:30:00 GMT
```

1.4 L'Heure POSIX ou l'heure Unix

L'heure Unix ou heure Posix (aussi appelée Unix Timestamp) est une mesure du temps basée sur le nombre de secondes écoulées depuis le 1er janvier 1970 00:00:00 UTC, hors secondes intercalaires. Elle est utilisée principalement dans les systèmes qui respectent la norme POSIX ISO 8601, dont les systèmes de type Unix, d'où son nom. C'est la représentation POSIX du temps.

1.4.1 Lubridate et Instants

Lubridate dispose d'un analyseur POSIX très rapide intégré. La plupart des formats strptime() et des diverses extensions sont pris en charge

Les instants sont des moments précis du temps. ***Date***, ***POSIXct*** et ***POSIXlt*** sont les trois classes d'objets que la base R reconnaît comme des instants.

Chaque objet POSIXct est enregistré comme le nombre de secondes où il s'est produit après le 1970-01-01 00:00:00.

is.Date() vérifie si un objet hérite de la classe Date.

```
is.Date(as.Date("2009-08-03"))
```

is.POSIXt() vérifie si un objet hérite des classes POSIXlt ou POSIXct. is.instant() vérifie si un objet hérite de l'une des trois classes.

```
is.POSIXt(as.Date("2009-08-03"))
is.POSIXt(as.POSIXct("2009-08-03"))
```

now() renvoie l'heure système actuelle sous la forme d'un objet POSIXct.

```
now(tzone = "")
```

today() renvoie la date système actuelle.

```r
today(tzone = "")
```