Fiche TD avec le logiciel R : tdr1b

Manipuler des données calendaires

D. Chessel

La fiche contient quelques exercices élémentaires pour lire, éditer et manipuler des dates d'événements. On pourra y revenir pour des questions plus évoluées de traitement des séries chronologiques.

Contents

Note liminaire

POUR une version actualisée[1] on pourra consulter pour les objets de la classe `Date` (données calendaires) la fiche « [m]anipulation de données calendaires appliquée au suivi hebdomadaire de la croissance de dix chênes pendant sept ans[2] » et pour les objets de la classe `POSIXt` (horodatage) la fiche « [m]anipulation de données temporelles appliquée au suivi horaire de la croissance de dix chênes pendant un an[3] ».

1 Implanter des dates calendaires

Dans un grand quotidien (*Le Monde, 2003)* , on pouvait lire ce qui suit.

> L'Iran a été frappé à de nombreuses reprises par des séismes de grande magnitude, qui ont fait de très nombreuses victimes. Voici les principaux :
>
> - ⋆ 1er septembre 1962 : 12000 morts, 200 villages détruits dans le district de Qazvin (ouest de Téhéran).
> - ⋆ 31 août 1968 : environ 10000 morts dans la province de Khorassan (Nord-Est), magnitude 7,4 sur l'échelle de Richter.
> - ⋆ 10 avril 1972 : 5 044 morts, 45 villages détruits dans la région de Ghir (province de Fars, Sud).
> - ⋆ 16 septembre 1978 : 25 000 morts dans l'Est. La ville de Tabass est entièrement détruite par ce séisme de magnitude 7,5, et compte à elle seule 15 000 morts.
> - ⋆ 11 juin 1981 : 1 028 morts et 950 blessés dans la province de Kerman (Sud-Est). La ville de Golbaf est la plus touchée. Le 28 juillet, un nouveau séisme fait 1300 morts dans la même région.
> - ⋆ 21juin 1990 : 37 000 morts, et plus de 100 000 blessés dans les provinces de Ghilan et de Zandjan (Nord-Ouest). Ce tremblement de terre d'une magnitude de 7,7 - le plus meurtrier en Iran - a dévasté en quelques secondes 2100 km2, comprenant 27 villes et 1871 villages.
> - ⋆ 28 février 1997: un millier de morts et 2 600 blessés dans la région d'Ardabil (Nord-Ouest)
> - ⋆ 10 mai 1997 : environ 1600 morts et 3700 blessés (7,1) dans la région de Birjand (Est). - (AFP.)

On veut calculer en jours les temps d'attente entre deux catastrophes. Pour implanter des données *dates et/ou instants*, en particulier les données calendaires, on saisit habituellement les dates par chaînes de caractères dans un format précis mais arbitraire. Pour ce faire :

[1] La dernière mise à jour du contenu de cette fiche date du 2009-07-07.
[2] `https://pbil.univ-lyon1.fr/R/pdf/dendroCHS57.pdf`
[3] `https://pbil.univ-lyon1.fr/R/pdf/microdendroCHS57.pdf`

1. Définir d'abord le format. Nous avons besoin de renseigner l'année, le jour et le mois. Les éléments pouvant être introduit dans un tel format sont entre autres (voir la documentation de `strptime`) :

   - ⋆ `%d` le jour du mois (01-31)
   - ⋆ `%H` l'heure du jour (00-23)
   - ⋆ `%m` le mois de l'année (01-12)
   - ⋆ `%M` les minutes de l'heure (00-59)
   - ⋆ `%Y` l'année sur 4 caractères (éviter systématiquement les deux caractères)

   Par exemple, on peut utiliser le format `"%d/%m/%Y"` pour lire "25/12/2002" ou `"%Y-%d-%m"` pour lire "2002-25-12".

2. Faire ensuite un vecteur de chaînes de caractères conformément au format choisi :

```
w1 <- c("1/09/1962", "31/8/1968", "10/04/1972", "16/09/1978",
"11/06/1981", "21/06/1990", "28/02/1997", "10/05/1997")
w1
```

```
[1] "1/09/1962"  "31/8/1968"  "10/04/1972" "16/09/1978" "11/06/1981" "21/06/1990"
[7] "28/02/1997" "10/05/1997"
```

3. Lire ce vecteur de chaînes de caractères avec la fonction **`strptime`** qui lit les chaînes de caractères et les transforme au format `POSIXlt` (local time) :

```
w2 <- strptime(w1,"%d/%m/%Y")
class(w2)
```

```
[1] "POSIXlt" "POSIXt"
```

```
unclass(w2)
```

```
$sec
[1] 0 0 0 0 0 0 0 0
$min
[1] 0 0 0 0 0 0 0 0

$hour
[1] 0 0 0 0 0 0 0 0

$mday
[1]  1 31 10 16 11 21 28 10

$mon
[1] 8 7 3 8 5 5 1 4

$year
[1] 62 68 72 78 81 90 97 97

$wday
[1] 6 6 1 6 4 4 5 6

$yday
[1] 243 243 100 258 161 171  58 129

$isdst
[1] 0 0 0 1 1 1 0 1

$zone
[1] "CET"  "CET"  "CET"  "CEST" "CEST" "CEST" "CET"  "CEST"

$gmtoff
[1] NA NA NA NA NA NA NA NA
```

```
apropos("POSIXlt")
```

```
 [1] "[.POSIXlt"             "[[.POSIXlt"            "[[<-.POSIXlt"
 [4] "[<-.POSIXlt"           "anyNA.POSIXlt"         "as.data.frame.POSIXlt"
 [7] "as.Date.POSIXlt"       "as.double.POSIXlt"     "as.list.POSIXlt"
[10] "as.matrix.POSIXlt"     "as.POSIXct.POSIXlt"    "as.POSIXlt"
[13] "as.POSIXlt.character"  "as.POSIXlt.Date"       "as.POSIXlt.default"
[16] "as.POSIXlt.factor"     "as.POSIXlt.numeric"    "as.POSIXlt.POSIXct"
[19] "as.vector.POSIXlt"     "c.POSIXlt"             "duplicated.POSIXlt"
[22] "format.POSIXlt"        "is.na.POSIXlt"         "length.POSIXlt"
[25] "length<-.POSIXlt"      "mean.POSIXlt"          "names.POSIXlt"
[28] "names<-.POSIXlt"       "print.POSIXlt"         "rep.POSIXlt"
[31] "sort.POSIXlt"          "summary.POSIXlt"       "Summary.POSIXlt"
[34] "unique.POSIXlt"        "xtfrm.POSIXlt"
```

La classe `POSIXlt` est la plus accessible des classes de données pour les datations. On utilisera la classe `POSIXct` pour les `data.frame`. Noter dans la liste qui précède successivement :

- ⋆ `sec` les secondes dans la minutes, par défaut 0
- ⋆ `min` les minutes dans l'heure, par défaut 0
- ⋆ `hour` les heures dans la journée, par défaut 0
- ⋆ `mday` le jour du mois
- ⋆ `mon` le mois de l'année codé `0-11`
- ⋆ `year` l'année sur 2 caractères
- ⋆ `wday` le jour de la semaine 0 pour dimanche
- ⋆ `yday` le jour de l'année
- ⋆ `isdst` le décalage de l'horaire d'été

On notera l'usage de la fonction `unclass` pour examiner l'intérieur de l'objet et la liste des fonctions qui contiennent la chaîne `POSIXlt` dans leur nom fournie par `apropos`.

2 Premières utilisations des données calendaires

Les premières fonctions utilisent qui utilisent la classe d'objets `POSIXlt` sont :

- ⋆ `weekdays` pour avoir le jour de la semaine en clair ;
- ⋆ `months` pour avoir le mois en clair ;
- ⋆ `quarters` pour avoir le trimeste codé `Q1-Q4` ;
- ⋆ `julian` pour avoir la distances au 01/01/1970 en jours ;

```
weekdays(w2)
months(w2)
quarters(w2)
julian(w2)
as.numeric(julian(w2))
class(julian(w2))
```

On peut maintenant répondre à la question posée au paragraphe précédent : la mesure est en jours, les décimales intégrant heures, minutes et secondes, à la seconde la plus proche.

```
w3 <-
diff(as.numeric(julian(w2)))
```

Pour vérifier la signification des décimales :

```
 julian(strptime("1970-01-01","%F"))
Time difference of -0.04166667 days
 julian(strptime("1970-01-01 01:00","%F %R"))
Time difference of 0 days
 julian(strptime("1970-01-01 02:00","%F %R"))
Time difference of 0.04166667 days
 julian(strptime("1970-01-01 01:01","%F %R"))
Time difference of 0.0006944444 days
```

Noter que :

- `%F` équivaut à `%Y-%m-%d` ;
- `%R` équivaut à `%H:%M` ;
- 0.04167 jour vaut bien une heure ;
- une journée commence exactement à 1 heure.

Ceci permet toute précision exigée par le problème étudié.

3 Éditer les dates

La fonction `strftime` lit les données au format POSIXlt (local time) et les édite avec un choix du format. On l'utilisera par son synonyme plus simple à mémoriser de `as.character` (en fait, `as.character.POSIXt`, évidemment).

```
 strftime(w2,"%A %d %m %y")
[1] "Samedi 01 09 62"   "Samedi 31 08 68"   "Lundi 10 04 72"    "Samedi 16 09 78"
[5] "Jeudi 11 06 81"    "Jeudi 21 06 90"    "Vendredi 28 02 97" "Samedi 10 05 97"
 as.character(w2,"%A %d %m %y")
[1] "Samedi 01 09 62"   "Samedi 31 08 68"   "Lundi 10 04 72"    "Samedi 16 09 78"
[5] "Jeudi 11 06 81"    "Jeudi 21 06 90"    "Vendredi 28 02 97" "Samedi 10 05 97"
```

On pourra utiliser, en outre :

- `%a` le nom du jour abrégé
- `%A` le nom du jour complet
- `%b` le nom du mois abrégé
- `%B` le nom du mois complet
- `%j` le numéro du jour de l'année
- `%U` le numéro de la semaine dans l'année

Comment obtient-on cela ?

```
[1] "Samedi 01 septembre 62" "Samedi 31 août 68"      "Lundi 10 avril 72"
[4] "Samedi 16 septembre 78" "Jeudi 11 juin 81"       "Jeudi 21 juin 90"
[7] "Vendredi 28 février 97" "Samedi 10 mai 97"
[1] "Sam 01 sep 62" "Sam 31 aoû 68" "Lun 10 avr 72" "Sam 16 sep 78" "Jeu 11 jui 81"
[6] "Jeu 21 jui 90" "Ven 28 fév 97" "Sam 10 mai 97"
```

Qu'attendez-vous de l'ordre suivant ?

```
as.character(strptime("04-07-1993","%d-%m-%Y"),"%A %Y/%m/%d")
```

Éditer avec un maximum de détail le jour de votre anniversaire (celui que je préfère . . .)

Éviter de confondre le format de stockage et le format d'édition. Le 25 décembre 2003 à minuit pile est `"25 12 2003 24 00 00"`. Mais :

```
wcar <- "25 12 2003 24 00 00"
wlt <- strptime (wcar, "%d %m %Y %H %M %S")
wlt
[1] "2003-12-26 CET"
```

Minuit n'existe pas, essayer le 26 à 00:00:00 ! Examiner ce qui suit : quelles différences voyez-vous entre l'objet et sa réalisation ?

```
wcar <- "25 12 2003 23 59 59"
wlt <- strptime(wcar, "%d %m %Y %H %M %S")
wlt
[1] "2003-12-25 23:59:59 CET"
unlist(unclass(wlt))
   sec    min   hour   mday    mon   year   wday   yday  isdst   zone gmtoff
  "59"   "59"   "23"   "25"   "11"  "103"    "4"  "358"    "0"  "CET"     NA
```

Il existe un autre format de stockage :

```
wct <-  as.POSIXct(wlt)
wct
[1] "2003-12-25 23:59:59 CET"
attributes(wct)
$class
[1] "POSIXct" "POSIXt"
$tzone
[1] ""
```

On pourrait penser n'avoir pas changé grand chose. En fait on ne voit que la représentation de l'objet. Mais cette fois, il y a loin entre l'objet et sa représentation. Dans la classe `POSIXct` le contenu est le nombre secondes à partir de 01/01/1970 01:00:00. Retournons à l'école primaire (il y a 60 secondes par minutes et 60 minutes par heure, 24 heures par jour) :

```
unclass(wct)
[1] 1072393199
attr(,"tzone")
[1] ""
print(julian(wlt),dig=20)
Time difference of 12411.958321759258979 days
print(1072393199/60/60/24,dig=20)
[1] 12411.958321759258979
```

Pour travailler en numérique, le format POSIXct est plus pratique. Si ce calcul a déjà été assuré par un logiciel quelconque, le transfert sous forme numérique est le plus simple possible. Dans ces formats de données présents dans R mais universels, on travaille à la seconde. Il faut faire attention en permanence au travail –si bien fait– des fonctions d'édition :

```
 wlt
[1] "2003-12-25 23:59:59 CET"
 wlt+1
[1] "2003-12-26 CET"
 wlt+2
[1] "2003-12-26 00:00:01 CET"
 c(wlt, wlt+1,wlt+2)
[1] "2003-12-25 23:59:59 CET" "2003-12-26 00:00:00 CET" "2003-12-26 00:00:01 CET"
 c(wct,wct+1,wct+2)
[1] "2003-12-25 23:59:59 CET" "2003-12-26 00:00:00 CET" "2003-12-26 00:00:01 CET"
```

En bref : les fonctions `strptime` et `julian` suffisent à manipuler des dates simplement. On travaille à la seconde près dès qu'on introduit l'heure sinon on travaille en jours. La fonction `as.character` permet les éditions les plus claires.

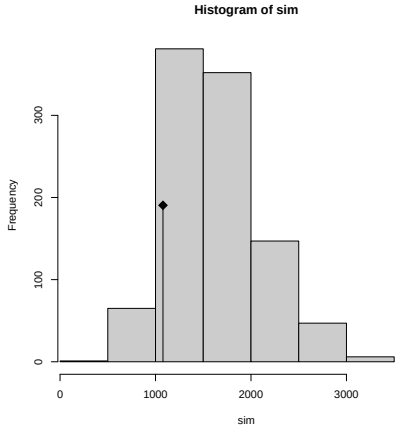
Calculer le temps qui sépare le 15 septembre 2004 et le 3 juillet 2005.

Noter encore qu'on peut saisir directement au format POSIXct par `ISOdatatime` ou `ISOdate` :

```
 ISOdate(2004,9,15)
[1] "2004-09-15 12:00:00 GMT"
 ISOdate(2004,9,15)-ISOdate(2005,7,3)
Time difference of -291 days
```

A propos des catastrophes en Iran, on pourra étudier ce problème (difficile) :

```
 library(ade4)
 f1 <- function(k) {
     w <-sort(c(-2679.1,runif(6,-2671.1,9990.9),9990.9))
     return(sd(diff(w)))
 }
 wt <- as.randtest(sim = unlist(lapply(1:999,f1)),sd(diff(julian(w2))))
 wt
 plot(wt,nclass=20)
```



Exécuter et expliquer l'intention et le résultat de cette opération.

4 L'arrivée des patients

On trouve dans l'ouvrage classique de D.R. Cox et P.A.W. Lewis [1] traduit en français [2] (1969 p.245-246) le jeu de données reproduit en fin de fiche (ğ 6). Implanter ces données. On pourra s'aider de :

`http://pbil.univ-lyon1.fr/R/donnees/coxlewisan1.txt`

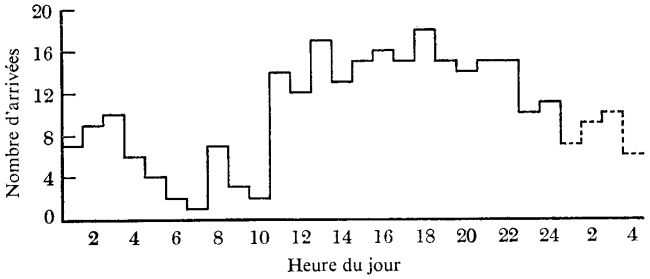
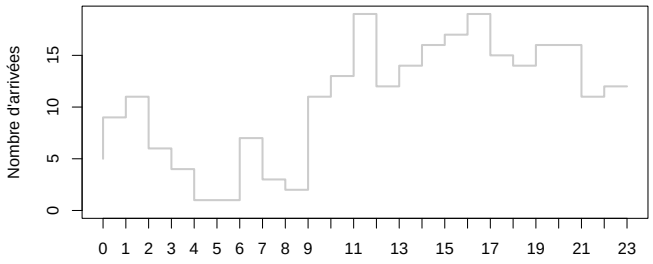


FIG. 1.8 *a* Arrivée des patients.
Graphique horaire du nombre d'arrivées destiné à l'étude de l'effet journalier.

Retrouver la description de l'effet journalier :

```
download.file("http://pbil.univ-lyon1.fr/R/donnees/coxlewisan1.txt","coxlewisan1.txt",mode="wb")
w <- scan("coxlewisan1.txt",character(),sep="\n")
w1 <- strptime(w,"%Y/%d/%m %H:%M")
plot(table(w1$hour),type="S",lwd=2,col=grey(0.8),ylab="Nombre d'arrivées")
```



Peut-on déceler un effet du jour de la semaine dans le nombre d'arrivées de patients ?

```
chisq.test(as.numeric(table(w1$wday)),p=rep(1/7,7))
        Chi-squared test for given probabilities
data:  as.numeric(table(w1$wday))
X-squared = 15.803, df = 6, p-value = 0.01485
```

```
table(w1$wday)
 0  1  2  3  4  5  6
28 37 54 34 27 30 44
```

```
table(weekdays(w1)) # présentation décevante !
Dimanche    Jeudi    Lundi    Mardi Mercredi   Samedi Vendredi
      28       27       37       54       34       44       30
```

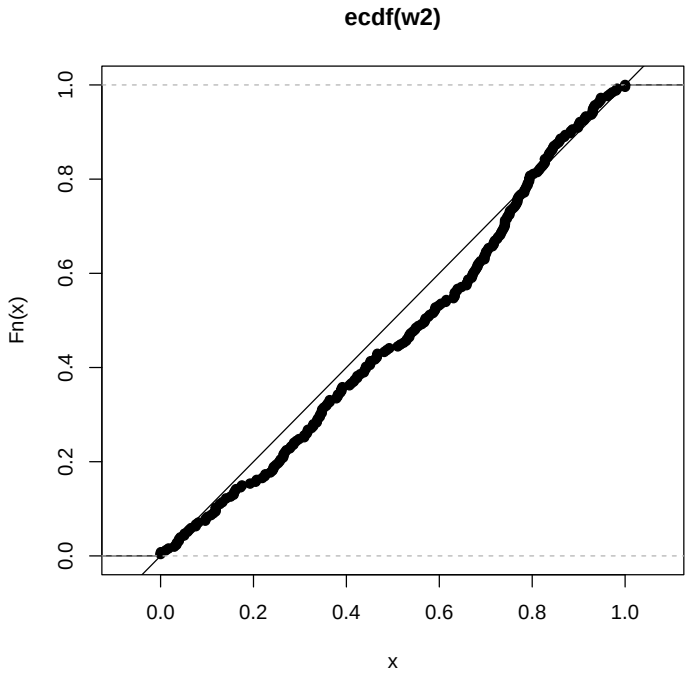
```
#d0 <- c("dimanche","lundi","mardi","mercredi","jeudi","vendredi","samedi")
#table(weekdays(w1))[d0]
#Marche pas.
```

Extraire la série des douze mois consécutifs et juger si on peut parler d'effet saisonnier ?

```
table(w1[1:230]$mon)
chisq.test(table(w1[1:230]$mon),p=rep(1/12,12))
```

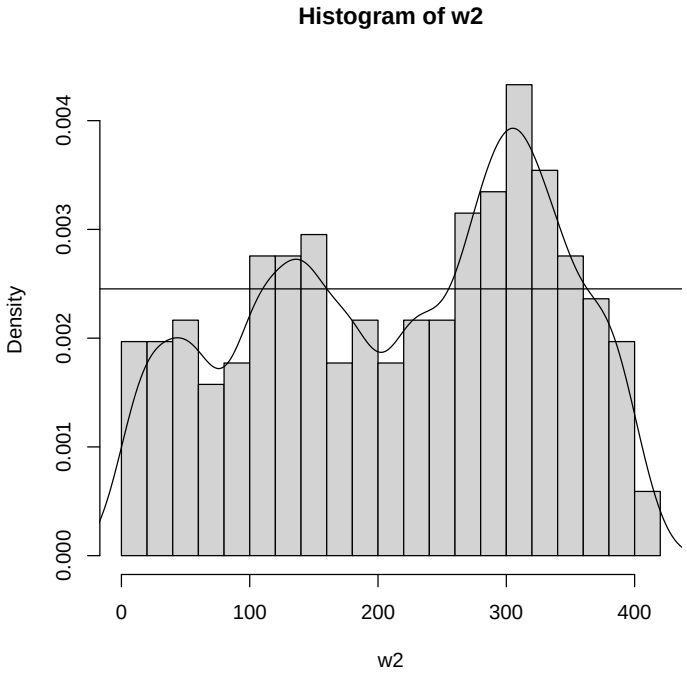
Tracer la fonction de répartition de la date d'arrivée des patients ramenées sur l'intervalle [0,1]. Tester l'écart avec le modèle d'une distribution uniforme (voir `ks.test`) :

```
w2 <- julian(w1)
w2 <- as.numeric(w2)
w2 <- (w2-min(w2))/(max(w2)-min(w2))
plot(ecdf(w2))
abline(c(0,1))
ks.test(jitter(w2),"punif") # n'aime pas les ex-aequo
```



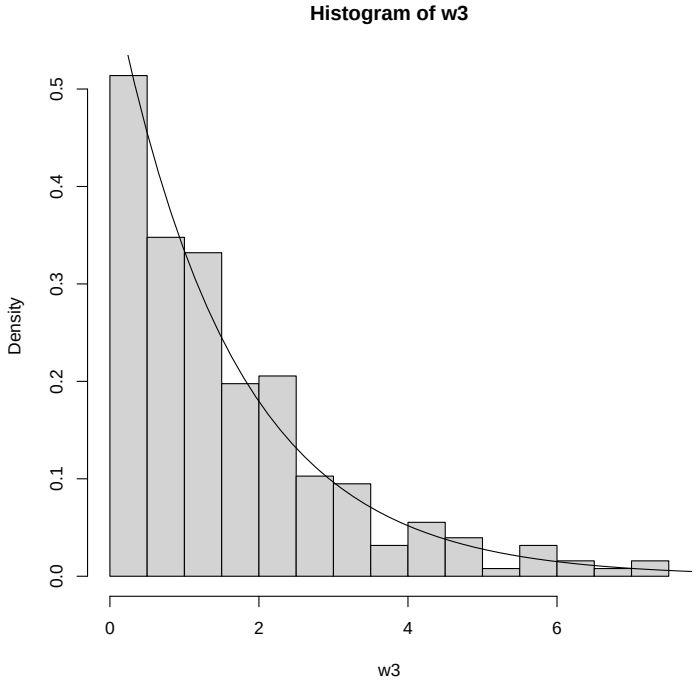
Tracer l'histogramme des dates d'arrivée des patient. Ajouter une estimation non paramétrique de la densité. La densité est-elle uniforme ?

```
w2 <- julian(w1)
w2 <- as.numeric(w2)
w2 <- w2-min(w2)
hist(w2,nclass=20,prob=T)
lines(density(w2,ad=0.5))
abline(h=1/range(w2))
a <- max(w2)
m <- a/2
s <- sqrt(a^2/12/length(w2))
mean(w2)-m/s
```



Tracer l'histogramme du temps d'attente du patient suivant. Ajuster une densité exponentielle.

```
w3 <- diff(w2)
hist(w3,proba=T,nclass=20)
lines(seq(0,8,le=50),dexp(seq(0,8,le=50),1/mean(w3)))
```



Tracer la distribution du nombre d'arrivées par jour et ajuster une loi de Poisson.

```
 table(table(as.integer(w2)))
  1   2   3   4
147  38   9   1
 obs <- c(211,150,38,8,1)
 calc <- dpois(0:3,254/408)
 calc <- c(calc,1-sum(calc))
 calc <- 408*calc
 plot(0:4,calc,type="n")
 points(0:4,obs,type="h",lwd=10,col=grey(0.8))
 points(0:4,calc,pch=20,cex=3)
 obs
[1] 211 150  38   8   1
 calc
[1] 218.922581 136.290038  42.423615   8.803593   1.560173
 sum((obs-calc)^2/calc)
[1] 2.401591
 pchisq(2.402,3)
[1] 0.5067376
```

Résumer les connaissances acquises.

5 Radio-tracking et sangliers

C. Calenge propose le fichier de données `puechabon.txt` issu des travaux de D. Maillard [3] et qui forme un objet de la librairie `adehabitat` de R. Lire ce fichier dans un objet `w` en conservant les chaînes de caractères telles quelles (voir `as.is` dans `read.table`). On utilisera le fichier :

`http://pbil.univ-lyon1.fr/R/donnees/puechabon.txt`

Les variables du fichier concernent le nom de l'animal (`Name`), son sexe (`Sex`), son âge (`Age`), les coordonnées X et Y des localisations de l'animal (en Lambert III) et la date à laquelle les localisations ont été relevées. Pour chaque animal, donner la date du premier enregistrement, la date du dernier, la durée de l'expérience et le nombre de localisations. Attention les localisations ne sont pas dans l'ordre chronologique !

On s'aidera de la remarque :

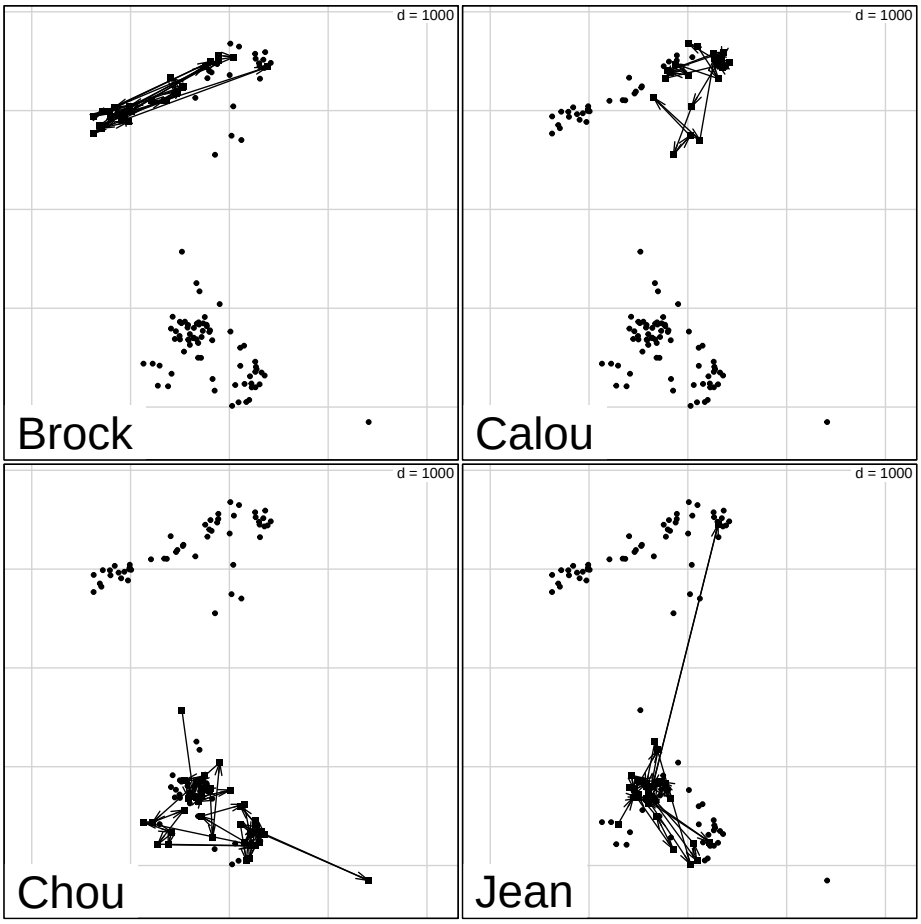
```
provi <-c("930807","930813","930831","930816","930728","930819", "930723","930713","930720","930724","930829",
strptime(provi,"%y%m%d")
```

```
 [1] "1993-08-07 CEST" "1993-08-13 CEST" "1993-08-31 CEST" "1993-08-16 CEST"
 [5] "1993-07-28 CEST" "1993-08-19 CEST" "1993-07-23 CEST" "1993-07-13 CEST"
 [9] "1993-07-20 CEST" "1993-07-24 CEST" "1993-08-29 CEST" "1993-07-29 CEST"
[13] "1993-07-01 CEST" "1993-07-30 CEST" "1993-07-14 CEST" "1993-07-15 CEST"
[17] "1993-07-06 CEST" "1993-08-27 CEST" "1993-08-01 CEST" "1993-08-10 CEST"
[21] "1993-08-30 CEST" "1993-07-08 CEST" "1993-08-03 CEST" "1993-07-09 CEST"
[25] "1993-07-26 CEST" "1993-07-07 CEST" "1993-08-09 CEST" "1993-08-22 CEST"
[29] "1993-08-21 CEST"
```

```
strptime(provi,"%y%m%d")[order(julian(strptime(provi,"%y%m%d")))]
```

```
 [1] "1993-07-01 CEST" "1993-07-06 CEST" "1993-07-07 CEST" "1993-07-08 CEST"
 [5] "1993-07-09 CEST" "1993-07-13 CEST" "1993-07-14 CEST" "1993-07-15 CEST"
 [9] "1993-07-20 CEST" "1993-07-23 CEST" "1993-07-24 CEST" "1993-07-26 CEST"
[13] "1993-07-28 CEST" "1993-07-29 CEST" "1993-07-30 CEST" "1993-08-01 CEST"
[17] "1993-08-03 CEST" "1993-08-07 CEST" "1993-08-09 CEST" "1993-08-10 CEST"
[21] "1993-08-13 CEST" "1993-08-16 CEST" "1993-08-19 CEST" "1993-08-21 CEST"
[25] "1993-08-22 CEST" "1993-08-27 CEST" "1993-08-29 CEST" "1993-08-30 CEST"
[29] "1993-08-31 CEST"
```

Tracer les trajectoires suivies par chacun des animaux :


d = 1000
Brock
d = 1000
Calou
d = 1000
Chou
d = 1000
Jean

6 Les données de Cox et Lewis

Tableau A 1.2 *Heures d'arrivée des patients à une unité de soin intensive**
(*A lire en colonne*)

1963		1963		1963		1963	
4 févr.	11,00 h	6 avr.	22,05	5 juin	22,30	23 juill.	21,45
	17,00	9 avr.	12,45	10 juin	12,30	24 juill.	21,30
8 févr.	23,15		19,30		13,15	27 juill.	0,45
11 févr.	10,00	10 avr.	18,45	12 juin	17,30		2,30
16 févr.	12,00	11 avr.	16,15	13 juin	11,20	29 juill.	15,30
18 févr.	8,45	15 avr.	16,00		17,30	1 août	21,00
	16,00	16 avr.	20,30	16 juin	23,00	2 août	8,45
20 févr.	10,00	23 avr.	23,40	18 juin	10,55	3 août	14,30
	15,30	28 avr.	20,20		13,30		17,00
21 févr.	20,20	29 avr.	18,45	21 juin	11,00	7 août	3,30
25 févr.	4,00	4 mai	16,30		18,30		15,45
	12,00	6 mai	22,00	22 juin	11,05		17,30
28 févr.	2,20	7 mai	8,45	24 juin	4,00	11 août	14,00
1 mars	12,00	11 mai	19,15		7,30	13 août	2,00
3 mars	5,30	13 mai	15,30	25 juin	20,00		11,30
7 mars	7,30	14 mai	12,00		21,30		17,30
	12,00		18,15	26 juin	6,30	19 août	17,10
9 mars	16,00	16 mai	14,00	27 juin	17,30	21 août	21,20
15 mars	16,00	18 mai	13,00	29 juin	20,45	24 août	3,00
16 mars	1,30	19 mai	23,00	30 juin	22,00	31 août	13,30
17 mars	11,05	20 mai	19,15	2 juill.	20,15	2 sept.	23,00
20 mars	16,00	22 mai	22,00		21,00	5 sept.	20,10
22 mars	19,00	23 mai	10,15	8 juill.	17,30	7 sept.	23,15
24 mars	17,45		12,30	9 juill.	19,50	8 sept.	20,00
	20,20	24 mai	18,15	10 juill.	2,00	10 sept.	16,00
	21,00	25 mai	21,05	12 juill.	1,45		18,30
28 mars	12,00	28 mai	21,00	13 juill.	3,40	11 sept.	21,00
	12,00	30 mai	0,30		4,15	13 sept.	21,10
30 mars	18,00	1 juin	1,45		23,55	15 sept.	17,00
2 avr.	22,00		12,20	20 juill.	3,15	16 sept.	13,25
	22,00	3 juin	14,45	21 juill.	19,00	18 sept.	15,05

* Nous sommes très reconnaissants à M. A. Barr, Oxford Regional Hospital Board, de nous avoir permis d'utiliser ces données.

1963		1963		1963-64		1964	
21 sept.	14,10 h	12 nov.	7,45	15 déc.	1,15	25 janv.	13,55
23 sept.	19,15	15 nov.	15,20	16 déc.	1,45	29 janv.	21,00
24 sept.	14,05		18,40	17 déc.	18,00	30 janv.	7,45
	22,40		19,50	20 déc.	14,15	31 janv.	22,30
27 sept.	9,30	16 nov.	23,55		15,15	5 févr.	16,40
28 sept.	17,30	17 nov.	1,45	21 déc.	16,15		23,10
1 oct.	12,30	18 nov.	10,50	22 déc.	10,20	6 févr.	19,15
2 oct.	17,30	19 nov.	7,50	23 déc.	13,35	7 févr.	11,00
3 oct.	14,30	22 nov.	15,30		17,15	11 févr.	0,15
	16,00	23 nov.	18,00	24 déc.	19,50		14,40
6 oct.	14,10		23,05		22,45	12 févr.	15,45
8 oct.	14,00	24 nov.	19,30	25 déc.	7,25	17 févr.	12,45
12 oct.	15,30	26 nov.	19,00		17,00	18 févr.	17,00
13 oct.	4,30	27 nov.	16,10	28 déc.	12,30		18,00
19 oct.	11,50	29 nov.	10,00	31 déc.	23,15		21,45
20 oct.	11,55	30 nov.	2,30	2 janv.	10,30	19 févr.	16,00
	15,20		22,00	3 janv.	13,45	20 févr.	12,00
	15,40	1 déc.	21,50	5 janv.	2,30	23 févr.	2,30
22 oct.	11,15	2 déc.	19,10	6 janv.	12,00	24 févr.	12,55
23 oct.	2,15	3 déc.	11,45	7 janv.	15,45	25 févr.	20,20
26 oct.	11,15		15,45		17,00	25 févr.	10,30
30 oct.	21,30		16,30		17,00	2 mars	15,50
31 oct.	3,00		18,30	10 janv.	1,30	4 mars	17,30
1 nov.	0,40	5 déc.	10,05		20,15	6 mars	20,00
	10,00		20,00	11 janv.	12,30	10 mars	2,00
4 nov.	9,45	7 déc.	13,35	12 janv.	15,40	11 mars	1,45
	23,45		16,45	14 janv.	3,30	18 mars	1,45
5 nov.	10,00	8 déc.	2,15		18,35		2,05
6 nov.	7,50	9 déc.	20,30	15 janv.	13,30		
7 nov.	13,30	11 déc.	14,00	17 janv.	16,40		
8 nov.	12,30	12 déc.	21,15	19 janv.	18,00		
9 nov.	13,45	13 déc.	18,45	20 janv.	20,00		
	19,30	14 déc.	14,05	21 janv.	11,15		
11 nov.	0,15		14,15	24 janv.	16,40		

Remarque : on utilisera 20h 20 et 20h 30 pour les arrivées du 25 février 1964.

Google ... la Science n'a pas de prix !

References

[1] D.R. Cox and P.A.W. Lewis. *The statistical analysis of series of events.* Methuen & Co LTD, London, 1966.

[2] D.R. Cox and P.A.W. Lewis. *L'analyse statistique des séries d'évènements.* Traduction de Larrieu, J. Dunod, Paris, 1969.

[3] D. Maillard. *Occupation et l'utilisation de la garrigue et du vignoble méditerranéens par le Sanglier.* PhD thesis, Université d'Aix-Marseille III, 1996.