Paris School of Business (PSB)

Compréhension et utilisation de 5 packages de R: evir, evd, R.miner, graphics, regtest

MSc Data Management

Projet : **R**

par :

Aufrere THUY, Nina ZOUMANIGUI, Arnaud Bruel YANKO

Sous la direction de :

M. Henri Laude

Enseignant

Année académique 2020-2022

♣ Table des matières ♣

0.1 Introduction

Dans une session R, nous avons accès à un bon nombre de fonctions et de jeux de données. Les objets accessibles sont ceux contenus dans les packages R chargés à l'ouverture de la session. Un package R est simplement un regroupement de fonctions et de données documentées. Ce tutoriel à pour but de vous faire une presentation de certains packages qui ont été choisis à savoir evir, evd, R.miner, graphics, regtest, packages qui nous permettrons :

— D'explorer le monde de la théorie des valeurs extrêmes (Cours intéressant sur la Statisque des valeurs extrêmes) en modélisant les risques extrêmes ;

— Travailler d'une manière générale sur la Régression.

0.2 Installation et chargement d'un package R

Il est simple de charger en R des packages supplémentaires à ceux chargés par défaut. Il suffit d'utiliser la fonction library comme dans cet exemple :

library(evir)

Cette commande fonctionne uniquement si le package a été préalablement installé sur notre ordinateur. L'installation d'un package et le chargement d'un package sont deux étapes distinctes. Certains packages R sont installés automatiquement lors de l'installation de R.

La fonction *installed.packages* retourne des informations à propos des packages R installés sur l'ordinateur local.

regardons ensemble un exemple obtenu sur mon ordinateur.

```
> Arnaud <- installed.packages()
> head(Arnaud, n = 3)
          Package      LibPath                                        Version Priority Depends        Imports
abind     "abind"      "C:/Users/yanko/Documents/R/win-library/3.6" "1.4-5" NA       "R (>= 1.5.0)" "methods, utils"
actuar    "actuar"     "C:/Users/yanko/Documents/R/win-library/3.6" "2.3-3" NA       "R (>= 3.3.0)" "stats, graphics, expint"
assertthat "assertthat" "C:/Users/yanko/Documents/R/win-library/3.6" "0.2.1" NA       NA             "tools"
          LinkingTo Suggests         Enhances License       License_is_FOSS License_restricts_use OS_type MD5sum NeedsCompilation
abind     NA        NA               NA       "LGPL (>= 2)" NA              NA                    NA       NA     "no"
actuar    "expint"  "MASS"           NA       "GPL (>= 2)"  NA              NA                    NA       NA     "yes"
assertthat NA       "testthat, covr" NA       "GPL-3"       NA              NA                    NA       NA     "no"
          Built
abind     "3.6.0"
actuar    "3.6.2"
assertthat "3.6.3"
```

0.3 Accéder au contenu d'un package R chargé

Une fois un package chargé en R avec la commande library, son contenu est accessible dans la session R. Nous avons vu dans des notes précédentes comment fonctionne l'évaluation d'expressions en R. Nous savons donc que le chargement d'un nouveau package ajoute un environnement dans le chemin de recherche de R, juste en dessous de l'environnement de travail. Le chargement de certains packages provoque aussi le chargement de packages dont ils dépendent. Ainsi, parfois plus d'un environnement est ajouté au chemin de recherche de R lors du chargement d'un package. L'environnement d'un package contient les fonctions publiques et les données du package.

0.3.1 Jeux de données

Souvent, les jeux de données inclus dans un package se retrouvent directement dans l'environnement d'un package dans le chemin de recherche. C'est le cas, par exemple, des jeux de données du package datasets.

head(ls("package :datasets"), n = 8)

```
> head(ls("package:datasets"), n = 8)
[1] "ability.cov"   "airmiles"      "AirPassengers" "airquality"    "anscombe"      "attenu"        "attitude"      "austres"
>
```

Dans notre cas espèce seulement les 8 premiers éléments de la liste sont affichés ici, car cette liste compte normalement 104 éléments.

Cependant, les jeux de données sont parfois cachés. Ils sont alors traités différemment des fonctions privées et ne se retrouvent même pas dans l'espace de noms du package.

La fonction data est très utile dans ce cas. Cette fonction a en fait plusieurs utilités.

Premièrement, elle permet d'énumérer tous les jeux de données contenus dans un package.

data(package = "evir")

Dans notre cas espèce, nous allons à partir du packages "evir", afficher le jeux de données $nidd.annual$, ces données représentent les niveaux maximaux annuels de la *rivière Nidd* dans le Yorkshire..

On a :

```
1 install.packages("evir")
2 library(evir)
3 donnes<-data(nidd.annual)
4 donnes<-nidd.annual
5 donnes
6
```

Et on a la sortie suivante :

```
> data(nidd.annual)
> nidd<-nidd.annual
> nidd
 [1]  65.08  65.60  75.06  76.22  78.55  81.27  86.93  87.76  88.89  90.28  91.80  91.80  92.82  95.47 100.40 111.54 111.74 115.52
[19] 131.82 138.72 148.63 149.30 151.79 153.04 158.01 162.99 172.92 179.12 181.59 189.04 213.70 226.48 251.96 261.82 305.75
> donnes<-nidd.annual
```

(−) Format

Nous obtenons en sortir un vecteur numérique contenant 35 observations.

0.4 Applications

Pour application on utilise les données des crues annuelles de la rivière $Nidd$, dans le Yorkshire.

0.4.1 Contexte

La hauteur d'une rivière est modélisée par une variable aléatoire X.
On dispose de $\{X_1, \cdots, X_n\}$ un échantillon de hauteurs d'eau annuelles. On note $X_{1,n} \leq X_{2,n} \leq, \cdots, \leq X_{n,n}$ l'échantillon ordonné, avec $n \in \mathbb{N}$.

Deux problèmes complémentaires :

* Calculer la probabilité p d'une hauteur d'eau h extrême
  $p = \mathbf{P}(X \geq h)$ avec $h > X_{n,n}$.
* Calculer le niveau d'eau h qui est atteint ou dépassé une seule fois sur $T > n$, i.e. résoudre
  $1/T = \mathbf{P}(X \geq h)$

Modélisation : Le but ici étant d'écrire un programme permettant de calculer la fonction empirique des excès moyens et de tracer les différents seuils :

Allons y :

$(-)$ On Charge au préalable nos deux packages, les bouts de code suivant permettent de faire ce travail :

```
library(evd)
library(evir)
```

$(-)$ Nous allons commencer par lire les données du fichier $Nidd$ et les classé par ordre décroissant dans un vecteur noté $nidd$. On creé un vecteur $seuil$ ordonnés par ordre.

```
data(nidd.annual)
nidd<-nidd.annual
nidd

nidd2<-sort(nidd, decreasing = TRUE)
nidd2
seuil=seq(70,300,by =5)
#seuil<-70:300
seuil
taille_seuil=length(seuil)
taille_nidd2=length(nidd2)
```

(−) Afin de calculer la moyenne des excès pour chaque seuil, on crée une matrice dont le nombre de lignes correspond à la taille du vecteur "$seuil$" et le nombre de colonnes correspond à la taille du vecteur "$nidd$" telle que chaque ligne i correspond aux excès au delà de $seuil[i]$.

Puisque certains excès peuvent être négatifs, nous mettons les zéros à la place des termes négatifs :

```
X=matrix(0,nrow=length(seuil),ncol=length(nidd2))
X
#View(X)

for(i in 1:length(seuil))
{for(j in 1:length(nidd2))
{X[i,j]=max(nidd2[j]-seuil[i],0)}
}
X
```

(−) On calcule pour chaque seuil la moyenne des excès :

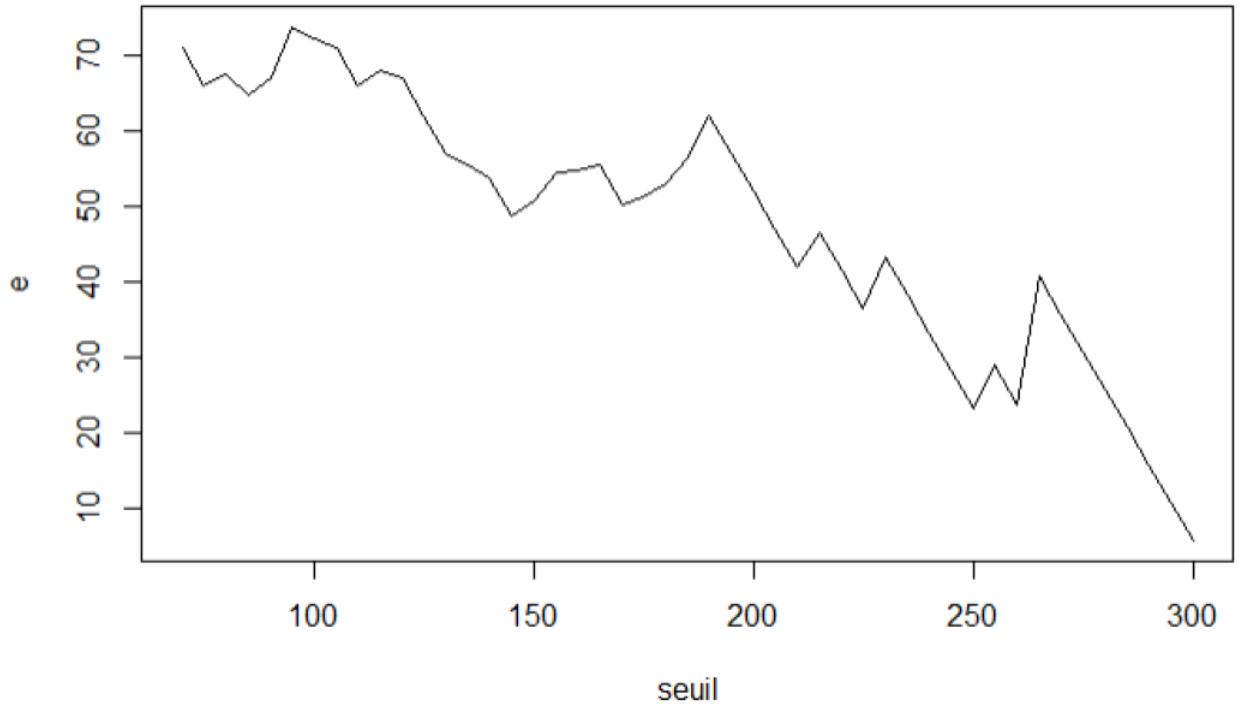
```
somme=rep(0,length(seuil))
somme
Compteur=rep(0,length(seuil))
Compteur
e=c()
e

for(i in 1:length(seuil))
{
  for(j in 1:length(nidd2))
  {
    if ( X[i,j]>0 )
    {
    somme[i]=somme[i]+X[i,j]
    Compteur[i]=Compteur[i]+1
    }
  }
  e[i]=somme[i]/Compteur[i]
}
e
```

(−) La fonction moyenne des excès est une méthode permettant de répondre à l ?une des dificultés de la modélisation des sinistres extrêmes qui est la détermination du seuil. Ainsi, graphiquement il est possible de déterminer le seuil le plus adéquat, en prenant la valeur à partir de laquelle la fonction moyenne des excès est linéaire.

Regardons ensemble ce que cela produit :

```
plot(seuil,e,type='l')
```



Pour aller plus loin l'idéal serait d'appliquer cela aux différentes lois usuelles : La loi Pareto, Weibull, Frechet, Gamma, Cauchy, Normal et bien d'autres : (Les differents lois en Statisque des valeurs extrêmes)

♣ Bibliographie ♣

[1] Site le CRAN *https ://cran.r-project.org/web/packages/available packages by name.html available-packages-E*

[2] *https ://cran.r-project.org/web/packages/*

[3] Sidney I. Resnick *Extreme Values, Regular Variation and Point Processes*

[4] Stuart Coles *An Introduction to Statistical Modeling of Extreme Values*

[5] Laurens de Haan *Extreme value theory*