Département
Licence

MOSE2014 Probabilités et statistiques
Mathématiques TP machine 1
Ph. Thieullen

TP machine I
(Première approche du logiciel `R`)

Les étudiants peuvent choisir de travailler en binôme ou en trinôme. Les TP font partie de la note de contrôle continu. Cependant ce premier TP ne donnera pas lieu à une note.

1 Environnement de travail

`R` est un logiciel de calcul scientifique orienté vers l'analyse des données en statistique. `R` est un logiciel du domain public comparable au logiciel professionnel `S-Plus` utilisé courament dans l'industrie et les laboratoires de recherche.

Environnement sur Unix : Avant toute utilisation de `R`, il est nécessaire de bien s'organiser.

– Ouvrir le terminal de commandes en ligne Unix.

– Créer un répertoire de travail : `mkdir MOSE2014`

– Changer de répertoire : `cd MOSE2014`

– Lancer alors le logiciel en allant dans `Menu des applications` → `Science` → `RKWard`. Cliquer sur `Terminal R` pour ouvrir la console. Vérifier bien le nom du répertoire courant par `getwd()`. Eventuellement pour changer de répertoire, utiliser `setwd()` ; pour lister touts les fichiers, `list.files()`.

– `R` peut être utilisé interactivement depuis la console, mais il est plus commode d'écrire un script séparément et de l'exécuter par la suite sur la console. C'est ce qu'on fera par la suite. Aller dans le menu : `Fichier -> Nouveau script` pour créer un fichier vierge.

– Ecrire quelques lignes de commentaires : nom du tp, nom des étudiants en binôme ou trinôme

```
# TP1
# noms, prenoms : ...
```

Remarquer bien qu'un commentaire commence toujours par `#` ; le reste de la ligne n'est alors pas exécutée.

– Enregistrer le fichier sous le nom `TP1.R` en n'oubliant pas l'extension `.R`. Vérifier qu'il se trouve dans votre répertoire courant.

– Tester une opération simple : d'abord sur la console en tapant par exemple `choose(4,2)` puis retour a la ligne. La commande calcule le coefficient binomial $\binom{4}{2} = \frac{4!}{2!(4-2)!} = \frac{4*3}{2*1} = 6$. Le résultat 6 s'affiche.

– Recommencer cette opération dans le script `TP1.R` : écrire sur deux lignes séparées sans mettre les commentaires

```
x <- choose(4,2) # affectation d'une variable x
cat(x) # affichage de x
```

Exécuter maintenant ce script en tapant sur la console

```
source("TP1.R")
```

Constater bien que le résultat 6 est affiché. On notera que, sur la console, la touche ↑ rappelle la dernière commande et que → permet de completer automatiquement les noms de chemins et de fichiers.

– Pour obtenir de l'aide en ligne sur une commande : écrire sur la console `?`*`nom_commande`*. Par exemple taper

```
?choose
```

Un autre méthode plus simple (Unix seulement) consiste, dans le fichier script, à mettre le pointeur sur le nom de la fonction puis à cliquer sur `F2`.

– Pour obtenir de l'aide en ligne en général, depuis le menu, exécuter `Aide -> Aide HTML`. Puis cliquer sur `Search Engine & Keywords` et écrire `Choose` dans le champ `Search`. Comparer avec la méthode précédente.

Environnement sous Windows (à faire à la maison) : Il est impératif que chaque étudiant expérimente le logiciel `R` chez lui sur son ordinateur personnel.

– Télécharger le logiciel (en mode administrateur) : le site officiel du Projet `R` est

`http://www.r-project.org/`

– Créer un répertoire de travail `MOSE2014`

– Dans le menu `Démarrer -> Tous les programmmes -> R`, par un clic droit sur `Envoyer vers Bureau`, créer un raccourci sur le bureau d'une version de `R` puis déplacer ce raccourci dans le répertoire `MOSE2014`.

– Modifier les propriétés de ce raccourci par un clic droit sur `Propriété -> Raccourci -> Démarrer dans`, et effacer les informations de ce champ. Rendre permanent la modification en cliquant sur `Appliquer`, puis sur `OK`. En démarrant `R` dans le répertoire `MOSE2014`, celui-ci devient le répertoire de travail courant.

2 Les bases de R

Taper les commandes dans le fichier script ; puis exécuter les en revenant sur la console et en tapant `source("TP1.R")` *ou an utilisant la touche ↑ pour rappeler les anciennes commandes.*

– L'affectation d'une variable se fait avec `<-`. Expérimenter

```
n1 <- pi
n2 <- exp(1)
n3 <- 1/4
n4 <- 3^3
n5 <- "Aa"
cat(n1, "\n", n2, "\n", n3, "\n", n4, "\n", n5, "\n")
```

Pour afficher les valeurs des variables sur la console, on utilise dans la script la commande `cat` (sur la console, il aurait suffi de taper le nom de la variable). Le caractère `\n` désigne un retour à la ligne. Le type de la variable s'appelle `mode`. Taper sur la console

```
mode(pi)
mode("Aa")
```

– La structure de données la plus simple est la structure `vector`

```
m1 <- c(-1,1,-2,2) ; m3 <- 1/m1
m2 <- c("A", "a", "B", "b", "C", "c")
m4 <- c(m1, m3, m1*m3)
cat(m1, "\n", m2, "\n", m3, "\n", m4, "\n" )
```

A chaque fois, il faut bien comprendre le sens de l'opération. Que fait `m3` ? Que fait `m4` ? Remarquer aussi qu'on peut séparer deux instructions par `;`

– Les opération arithmétiques se font composante par composante sur chaque élément des vecteurs. Les deux opérateurs suivant `:` et `seq` créent des suites de nombres de pas constant (pour `x:y` le pas est égal a 1). Expérimenter

```
p1 <- ((-10):10)/10
p2 <- seq(from=-1, to=1, by=0.1)
p3 <- p1-p2
p4 <- p1/p2
p5 <- sum(p1)
cat(p1, "\n", p2, "\n", p3, "\n", p4, "\n", p5, "\n")
```

C'est le moment d'aller chercher de l'information : taper sur la console

```
?sum
?":"
?seq
```

– On peut arranger une série de nombres sous forme de tableau multidimensionnel appelé `array`. Expérimenter sans écrire les commentaires

```
q0 <- array(1:9, dim=c(3,3)) # un tableau 3x3
q1 <- rep(c(1,2),times=3) # répétition de c(1,2)
q2 <- array(q1, dim=c(2,3)) # un tableau 2x3
q3 <- array(q1, dim=c(2,3),
      dimnames=list(c("L1","L2"),c("C1","C2","C3")))
q4 <- q2[2,3] # indexation par des nombres
q5 <- q3["L1",] # indexation par des noms
print(q0) ; print(q1) ; print(q2)
print(q3) ; print(q4) ; print(q5)
# l'affectation de la colonne C3
q3[,"C3"] <- c(0,0)
print(q3)
```

Remarquer bien comment le vecteur `1:9` a été aligné dans le tableau `q0`. On constate aussi que l'indexation peut se faire aussi bien avec des nombres qu'avec des noms `q3["L2","C3"]` (à condition qu'on ait nommé les colonnes et lignes auparavant). Remarquer enfin qu'on peut extraire des lignes complètes `q2[1,]` ou des colonnes complètes `q2[,3]`.

– La fonction de base pour afficher un graphique est `plot`. On commence par un exemple simple de superposition de 3 densités de loi normale

```
x <- seq(from=-10, to=10, by=0.01)
y <- dnorm(x, mean=1, sd=1)
z <- dnorm(x, mean=2, sd=1.5)
w <- dnorm(x, mean=3, sd=2)
plot(x,y, type="l", col="green")
lines(x,z, type="l", lty=2, col="blue")
lines(x,w, type="l", lty=3, col="red")
legend(-10, 0.4,
  legend=c("mean=1, sd=1","mean=2, sd=1.5","mean=3, sd=2"),
  lty=c(1,2,3), col=c("green", "blue", "red"))
```

Rechercher d'abord l'aide en ligne de `?dnorm` puis celle de `?plot`, `?lines` et `?legend`. Dans `type`, il s'agit de ℓ. Si vous manquez d'idée pour une couleur, taper sur la console `colors()`.

– Expérimenter maintenant l'exemple plus compliqué suivant sans écrire les commentaires. Il servira au TP prochain.

```
r1 <- seq(from=-4, to=4, by=0.01)
r2 <- dnorm(r1) # densité de la loi normale
r3 <- dt(r1,6) # densité de la loi de Student : ddl=6
r4 <- dt(r1,60) # densité de la loi de Student : ddl=60
```

```
r5 <- dbinom(0:10,10,0.5) # loi binomiale : n=10 et p=0.5
oldpar <- par(no.readonly = TRUE) # ancienne configuration
par(mfrow=c(1,3)) # 3 figures sur une même ligne
# dans le plot suivant type="l" ; ell pour ligne
plot(r1,r2, type="l", col="green",
    main="Loi Z normale",
    xlab="x", ylab="densité en x")
plot(r1,r3, type="l", # un tracé en ligne
    main="Loi T de Student",
    xlab="x", ylab="densité en x")
par(new=TRUE) # superposition des figures
plot(r1,r4, type="l", col="blue",
    xlab="", ylab="", xaxt="n", yaxt="n")
par(new=FALSE) # suppression de la superposition
plot(0:10,r5, type="h", # h pour histogramme
    main="loi B binomiale", xlab="k", ylab="P(B=k)", lwd=3)
par(oldpar) # on récupère la configuration
```

Il est important ici de lire la documentation concernant les graphiques :

```
?plot ?plot.default ?par ?hist ?barplot.
```

On devra trouver la figure 1.

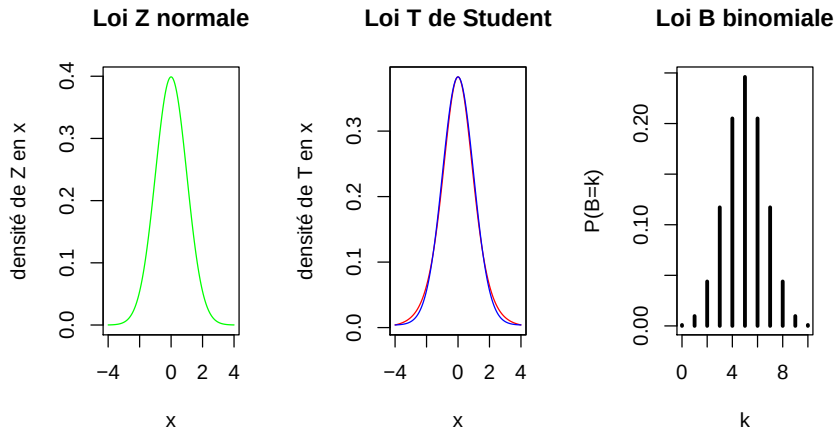

FIGURE 1 – Différentes distributions et leur loi

3 En résumé

Opérateurs :

?base::Arithmetic		?base::Comparison		?base::Logic		?base::tilde
`x + y`	addition	`x - y`	soustraction	`x * y`	multiplication
`x / y`	division	`x ^ y`	puissance	`x %% y`	reste
`x %/% y`	dividende	`x < y`	inférieur	`x > y`	supérieur
`x <= y`	inf ou égal	`x >= y`	sup. ou égal	`x = y`	égal
`x != y`	différent	`! x`	NON logique	`x & y`	ET logique
`x | y`	OU logique	`x : y`	suite	`x ~ y`	formule

Fonctions usuelles mathématiques :

?base::Trig ?Special					
`abs(x)`	`sqrt(x)`	`log(x)`	`exp(x)`	`cos(x)`	`sin(x)`
`tan(x)`	`acos(x)`	`asin(x)`	`atan(x)`	`gamma(x)`	`beta(x,y)`
`choose(n,k)`	`factorial(x)`				

Manipulation de vecteurs et de tableaux :

?S4groupGeneric ?complex ?round					
`sign`	`ceiling`	`floor`	`trunc`	`round`	`signif`
`sum`	`colSums`	`rowSums`	`cumsum`	`prod`	`cumprod`
`min`	`pmin`	`max`	`pmax`	`range`	`which`
`which.min`	`which.max`	`any`	`all`	`length`	`mean`
`colMeans`	`rowMeans`	`median`	`quantile`	`var`	`cor`
`Re`	`Im`	`Mod`	`Arg`	`Conj`	

Distributions usuelles statistiques :

?stats::distribution				
stats::Binomial	dbinom	pbinom	qbinom	rbinom
stats::Chisquare	dchisq	pchisq	qchisq	rchisq
stats::Exponential	dexp	pexp	qexp	rexp
stats::Geometric	dgeom	pgeom	qgeom	rgeom
stats::Normal	dnorm	pnorm	qnorm	rnorm
stats::Poisson	dpois	ppois	qpois	rpois
stats::TDist	dt	pt	qt	rt
stats::Uniform	dunif	punif	qunif	runif