TP : Initiation à R[1]

ISIFC 2ème année, Statistiques pour l'ingénieur, 2020-2021

Le TP dure 4h et vous êtes invités à le faire en binôme. On ne vous demande pas de compte-rendu, mais prenez des notes qui pourront vous servir ultérieurement. Lisez bien l'énoncé car vous y trouverez beaucoup d'indications.

Le logiciel R de traitement statistique de données est basé sur un langage fonctionnel et interprété. Il est proche de Matlab dans son utilisation. À la différence de Matlab, il est uniquement dédié à la statistique ; c'est de plus un logiciel libre. L'IDE RStudio [2], lui aussi libre et gratuit pour un usage non commercial, facilite l'utilisation de R (interface qui regroupe invite de commande, éditeur de script, fenêtre graphique, etc). Pour commencer ce TP, lancez RStudio.

1 Prise en main de R

1.1 L'environnement de travail

Attention : R crée un répertoire .RData et un fichier .Rhistory dans le répertoire courant. Il est conseillé de se placer dans un nouveau répertoire de travail (`TPStat` par exemple). Pour ce faire depuis RStudio, cliquez sur "Files" dans la partie en bas à droite de la fenêtre pour afficher l'explorateur de fichier ; puis créez un nouveau dossier, cliquez dessus et enfin sur le bouton "More → Set as working directory".

Dans RStudio, la documentation s'affiche aussi dans la partie en bas à droite de l'écran (onglet "Help"). Vous pouvez également obtenir de l'aide sur une commande particulière en tapant `?`*nom de la commande*. Par exemple, testez

```
> ?sum
```

1.2 Les commandes élémentaires

Dans l'interface du logiciel, vous avez accès à une fenêtre qui vous permet d'entrer interactivement des commandes et qui vous affiche les résultats. Les commandes élémentaires sont des expressions comme par exemple :

```
> 3 + 4
```

La structure de données élémentaire, comme pour Matlab, est le vecteur. La syntaxe change par contre par rapport à Matlab. Pour définir un vecteur `a` qui contient pour valeurs $3, 5, 7$ et 11, on procède comme suit :

```
> a <- c(3,5,7,11)
```

Notez l'instruction d'affectation, qui est `<-`.

Pour savoir ce que contient la variable `a` :

```
> a
```

1. https://www.r-project.org/
2. https://www.rstudio.com/

Exercice 1. Quels résultats obtenez-vous avec les commandes suivantes ? `a[2]`, `a[1:3]`, `a[a>5]`, `a^2`, `a**2`, `sum(a)`, `min(a)`, `max(a)`, `b<-2:5`, `a*b`, `2*a-b`.

Outre l'utilisation interactive, vous pouvez aussi entrez des commandes dans des scripts (c'est ce que l'on vous conseille). Pour créer un script : "File → New File → R Script". Lorsque vous enregistrez le script, il est sauvé par défaut dans votre répertoire courant. N'oubliez pas l'extension `.R`.

Dans l'éditeur de RStudio :

- Pour charger tout le script en mémoire, cliquer sur le bouton "Source".
- Pour exécuter la ligne sur laquelle se trouve votre curseur, utilisez `Ctrl+Entrée`.
- Pour exécuter un bloc de texte, surlignez-le et utilisez `Ctrl+Entrée`.

Exercice 2. On peut aussi créer ses propres fonctions. Créer un script qui contient la fonction suivante :

```
mafonction <- function(x)
{
  n <- length(x)
  s <- 0
  for(i in 1:n) { s <- s + x[i] }
  # on pourrait remplacer les lignes précédentes par s <- sum(x)
  s/n
}
```

`mafonction` est le nom de la fonction, `x` est le paramètre, `s/n` est la valeur renvoyée par la fonction. Essayez cette fonction avec le vecteur `a` précédemment crée. Quel indicateur statistique vous renvoie cette fonction ?

Note : utilisez # pour mettre des commentaires dans les fonctions.

2 Génération de données aléatoires

Pour entrer un échantillon statistique à la main :

```
> echantillon <- c(x1, x2, ..., xn)
```

Vous pouvez aussi générer un échantillon aléatoire via R. Par exemple, pour générer un échantillon de 7 réels qui suivent la loi normale de moyenne 4 et d'*écart-type* 3 :

```
> echantillon <- rnorm(7, 4, 3)
```

Exercice 3. Générez 9 échantillons de tailles différentes (voir l'aide de `rnorm`, `rexp` et `runif`) :

a) avec la loi normale centrée réduite (tailles n=10,50,500).

b) avec la loi exponentielle de paramètre lambda=0.1 (tailles n=10,50,500).

c) avec la loi uniforme sur l'intervalle (0,1) (tailles n=10,50,500).

Gardez bien les échantillons obtenus. Ils serviront par la suite.

Il est bien sûr possible d'importer des échantillons statistiques de "grande taille" provenant par exemple d'un fichier texte ou Excel. Néanmoins, c'est un peu plus acrobatique qu'il n'y paraît. La documentation de R contient quelques pistes, que vous pourrez explorer chez vous si vous le souhaitez.

3 Représentation graphique de données

3.1 Données discrètes

On reporte ici la consommation en décembre 2015 de produits régionaux en Franche-Comté, en kilogrammes (ou en litres) :

Cancoillotte	1950	Saucisse de Morteau	920
Saucisse de Montbéliard	470	Mont d'or	750
Edel de Clairon	670	Vin jaune	715

Exercice 4. Représentez ces données à l'aide d'un diagramme sectoriel. La commande pour tracer un tel diagramme est :

```
> pie(x, labels = produits)
```

où la variable x contient les quantités et la variable `produits` contient les noms (entre guillemets).

Exercice 5. Tracez aussi le diagramme à colonnes correspondant :

```
> barplot(x, names = produits)
```

Exercice 6. Reprenez l'exemple :

Nombre d'enfants	0	1	2	3	4	5	6	>6
Fréquence absolue	235	183	285	139	88	67	3	0

Tracez le diagramme à colonnes associé.

3.2 Variables continues et histogramme

On reprend les données sur la durée de vie des dispositifs électroniques :

```
91.6, 35.7, 251.3, 24.3, 5.4, 67.3, 170.9, 9.5, 118.4, 57.1
```

Exercice 7.

a) Ordonnez l'échantillon (en utilisant la commande `sort`).

b) Représentez ces données à l'aide d'un histogramme à pas fixe. Indication :

```
> hist(x, prob=T, breaks = seq(..., ..., ...))
```

Comment utilise-t-on `breaks` et `seq` ? Que fait l'option `prob=T` ?

c) Faites de même avec un histogramme à pas variable.

d) On voudrait tracer le polygone des fréquences. Pour ce faire, testez la commande

```
> h <- hist(x, ...)
```

Que contient alors la variable `h` ? Et `h$mids` ? Pour afficher une ligne brisée en superposition sur le graphique courant, vous disposez de la commande :

```
> lines(liste_abscisses, liste_ordonées, options)
```

Exercice 8. Tracez les histogrammes associés aux échantillons obtenus dans l'exercice 3. Sauvez les résultats obtenus. Pour sauvegarder un graphique, on utilise par exemple

```
> pdf("image.pdf", width=6, height=5) # on peut remplacer pdf par png ou jpeg
> # suite de commandes qui permettent de tracer un graphique (hist, plot, etc...)
> dev.off()
```

Faites attention à respecter l'ordre dans lequel ces commandes doivent être utilisées.

4 Calcul d'indicateurs

Un logiciel comme R est bien sûr capable de calculer les indicateurs les plus courants pour un échantillon donné : `mean(x)`, `var(x)`, `median(x)`, `quantile(x, p)`, etc.

Exercice 9.

a) Que font les commandes ci-dessus ? Expliquez en particulier les commande `median` et `quantile`.

b) Appliquer ces commandes aux échantillons générés à l'exercice 3.

c) Que fait la commande `summary` ?

d) Que remarquez-vous concernant la commande `var` ? (faites le calcul à la main pour comparer...)

e) Que remarquez-vous lorsque vous faites varier la taille de l'échantillon ?

5 Régression linéaire

Exercice 10.

a) D'abord, générer un jeu de données (x, y) qui vous serve ensuite à effectuer une régression linéaire. On pourra utiliser ce qui suit :

```
> x <- seq(0,10,1)
> y <- -4*x + 1 + rnorm(length(x), 0, val);
```

où `val` est la valeur du bruit sur les données : essayez un bruit faible, moyen et fort pour comparer.

b) Commencez par visualiser les données, en faisant par exemple :

```
> plot(x, y)
```

c) Faire une régression linéaire sur les 3 jeux de données (bruit faible, moyen et fort). Faites le d'abord "manuellement" à l'aide des formules du cours et des commandes `var`, `cov` et `abline`. Ensuite, regardez comment faire avec la commande `lm` de R.

d) Évaluez la qualité de votre régression dans chacune des 3 situations (bruit faible, moyen et fort). Pour cela, calculez le coefficient de corrélation empirique (calculer à la main, puis tester la commande `cor`) et l'erreur quadratique minimale.

6 Bonus

Exercice 11. Que font les commandes :

```
> plot(density(ech))
> plot(ecdf(ech))
```

avec `ech` un échantillon.