TP Analyse des données n^0 2 : Initiation à Splus (suite)

Le premier TP a eu pour but de vous fournir un premier contact avec Splus et ses capacités graphiques. Il s'est terminé par l'apprentissage de l'utilisation des vecteurs et matrices et des méthodes d'indiçages. Ce second TP a pour but de vous familiariser avec les objets de type liste et les `data.frame`, à vous initier à la manipulation de données et aux entrées sorties (lecture/écriture de données, de fichiers de commande ou de fonctions).

Splus -e	Lancer Splus (avec l'option éditeur de ligne).
help.start()	Ouverture de la fenêtre d'aide (l'iconifier).
motif()	Ouverture d'une fenêtre graphique (la rétrécir et la placer dans un coin).
	————————————
	Manipulation de listes Une liste est un ensemble d'objets de nature quelconque. Si on met une suite d'indices entre crochets simples à la suite du nom d'une liste, on en extrait une sous-liste. Si on met un indice entre crochets doubles, on obtient un composant de la liste
	————————————
?evap	Aide sur le jeu de données "évaporation".
dimnames(evap.x)	Noms de dimension du tableau `evap.x`.
is.list(dimnames(evap.x))	Cet objet est-il une liste ?
length(dimnames(evap.x))	Longueur de cette liste (nombre de composants).
dimnames(evap.x)[[2]]	Extraction du second composant
is.vector(dimnames(evap.x)[[2]])	Ce second composant est-il un vecteur (de chaines de caractères)?
mode(dimnames(evap.x)[[2]])	de quel mode est cet objet?
length(dimnames(evap.x)[[2]])	Nombre d'éléments du vecteur.
is.vector(dimnames(evap.x)[2])	Cette sous-liste est-elle un vecteur?
mode(dimnames(evap.x)[2])	De quel mode est cet objet?
is.list(dimnames(evap.x)[2])	Autre façon de poser la question.
length(dimnames(evap.x)[2])	Nombre de composants dans la sous-liste.
	Rappel du TP précédent (mettre des noms de dimension à un tableau).
x_1:20	Un exemple de vecteur.
x	Pour voir.
dim(x)_c(4,5)	On donne des dimensions à `x`.
x	Pour voir.
nomi_letters[1:4]	Définition de noms de lignes.
nomi	Pour voir.
nomj_LETTERS[1:5]	Définition de noms de colonnes.
nomj	Pour voir.
dimnames(x)_list(nomi,nomj)	On attribue une liste à deux composants "noms de dimensions" à la matrice `x`.
x	Pour voir.
	————————————
	Création d'une liste
	————————————
ll_list(x=x,y=1:30,letters[1:5], "Ceci est une chaine")	Définition d'une liste (attention : commande sur deux lignes).
ll	Pour voir
length(ll)	Longueur de la liste.
names(ll)	Nom des composants de la liste.
ll$x	On adresse un composant par son nom.
ll$y	idem
ll[[2]]	On adresse ce composant par son numéro.
ll[[3]]	idem (mais ici on n'a pas le choix).

	————————————
	Liste, résultat d'une fontion Splus.
	————————————
dimnames(evap.x)[[2]]	Noms des variables du tableau `evap.x`.
avh_evap.x[,"avh"]	extraction de la variable `avh` du tableau de données "évaporation".
avh_evap.x[,9]	Autre façon d'extraire cette variable.
plot(avh,evap.y)	représentation de l'évaporation en fonction de `avh`.
res.reg_lm(evap.y ~ avh)	Régression linéaire simple de `evap.y` en fonction de `avh`.
mode(res.reg)	De quel mode est cet objet?
length(res.reg)	Nombre de composants de cette liste.
names(res.reg)	Noms des composants de cette liste.
res.reg$coef	On affiche le composant coefficient
res.reg$residuals	On affiche le composant "résidus".
res.reg$res	idem (économie : on ne donne que les caractères nécessaires).
a_res.reg$coef[2]	On stocke le second coefficient de la régression
round(a,3)	(c'est la pente) : affichage de la pente.
abline(res.reg)	On représente la droite de régression.
segments(avh,evap.y,	On représente les résidus
avh,res.reg$fit,lwd=2)	(attention : commande sur deux lignes).
	————————————
	Introduction aux `data.frame` Ce sont des tableaux rectangulaires, mais pas forcément homogènes (ils peuvent contenir simultanément des variable de type charactère, des facteurs, des variables numériques). Ce sont des listes particulières.
	————————————
? fran	Information sur le jeu de données des franchissements.
dim(fran)	Dimensions du jeu de données `fran` (c'est bien un tableau rectangulaire).
is.data.frame(fran)	Est-ce un `data.frame` ?
mode(fran)	Est-ce une liste ?
length(fran)	Nombre de composants.
names(fran)	Les noms des composants sont les noms des variables.
fran[,1]	Extraction de la première variable.
fran[,"VER"]	Autre façon d'extraire cette variable.
mode(fran[,"VER"])	De quel mode est cette variable?
fran[,13]	Extraction de la 13-éme variable.
fran[,"TrH"]	Autre façon d'extraire cette variable (attention majuscules/minuscules).
mode(fran[,"TrH"])	De quel mode est cette variable?
search()	Liste des répertoires de recherche (équivalent des PATH de DOS).
ls()	Liste des objets situés dans votre répertoire de travail.
rm(ll)	On efface `ll` du répertoire de travail.
ls()	Vérification.
find(TrH)	Peux-t'on trouver cet objet quelquepart?
TrH	Cette variable esst-elle connue en tant que telle?
attach(fran)	On rajoute `fran` dans la liste des répertoires.
search(fran)	On vérifie que `fran` est attaché en deuxième position.
TrH	Cette variable est maintenant connue par son nom.
find(TrH)	Où se trouve t'elle?
ls(pos=2)	liste des opbjets du deuxième "répertoire" (en réalité le `data.frame`).
table(DEB)	Nombre de relevés pour chaque classe de débit.
table(DEB,TrH)	Effectifs des cellules par classe de débit et par tranche horaire.

———————————————
Brève introduction aux entrées sorties
———————————————

fran	On voit défiler le tableau `fran`.
sink("monfichier")	Tout ce qui devrait apparaitre à l'écran est redirigé. vers le fichier "monfichier".
TrH	Rien n'apparait car les sorties sont redirigées.
sink()	On arrête la redirection.
!emacs monfichier	On contrôle.
sink("monfichier")	On rétablit la redirection (l'ancien fichier est écrasé)
fran	le tableau `fran` est envoyé dans dans le fichier.
sink()	On termine la redirection.
!emacs monfichier	Pour voir (Ctrl X Ctrl C pour quitter emacs).
fran2_read.table("monfichier")	Lecture du fichier ASCII : `fran2` contient le même tableau que `fran`. Remarquez qu'il faut parfois spécifier à Splus l'existence des noms de variables en première ligne, grâce à l'option `header=T`.
!emacs hiphop	On ouvre un fichier ASCII, puis on tappe les commandes qui suivent dans le fichier `hiphop`.
clear()	Effacement de la fenêtre graphique.
par(mfrow=c(2,2))	Découpage de la fenêtre.
nbrel_apply(fran[,1:11],1,sum)	Calcul du nombre de passages par relevé.
pvert_100*VER/nbrel	calcul du pourcentage de passages au vert par relevé.
troug_apply(fran[,5:11],1,sum)	calcul du nombre de passages au rouge par relevé.
proug_100*troug/nbrel	calcul du pourcentage de passage au rouge.
plot(proug,pvert,xlab="%rouges",	Diagramme de dispersion.
ylab="%verts")	Attention, commande sur deux lignes.
plot(DEB,proug,xlab="Débit",	Boxplots parallèles.
ylab="%rouges")	Attention, commande sur deux lignes.
plot(TrH,proug,	
xlab="Tranche Horaire",	
ylab="%rouges")	Attention, commande sur trois lignes.
plot(TYJ,proug,xlab="Type de Jour",	
ylab="%rouges")	Attention, commande sur deux lignes.
	On sauve le fichier `hiphop`.
source("hiphop")	On exécute les commandes du fichier. Après, on peut par exemple rajouter des titres dans le fichier `hiphop`. et relancer la série de commandes par un nouveau `source("hiphop")`.

Exercice Générer un vecteur x constitué de 20 réalisations d'une variable aléatoire gaussienne de moyenne 2 et de variance 4. Soit $y = x^2$. Faire la régression de y par x. Rajouter un bruit gaussien centré de variance 0.01 à y et refaire la régression.