

Aide mémoire fonctions R

environnement

répertoire de travail	<code>setwd('chemin/rep')</code>
exécuter script	<code>source('file.R')</code>
charger package	<code>library(NomPackage)</code>
aide	<code>?NomFonction</code>
liste variables	<code>ls()</code>
effacer toutes variables	<code>rm(list=ls(all=TRUE))</code>

graphiques

nouveau graphe	<code>plot()</code>
plusieurs graphes	<code>matplot()</code>
ajout points	<code>points()</code>
ajout lignes	<code>lines()</code>
histogramme	<code>hist()</code>
boxplot	<code>boxplot()</code>
tracer fonction	<code>curve()</code>
variables 2 à 2	<code>pairs(M)</code>

commandes de base

commentaire	<code>#</code>
assignation	<code>x <- 3</code>
booléens	<code>TRUE FALSE T F</code>

fonctions de bases

entiers de x à y	<code>x:y</code>
vecteur répétant x	<code>rep(x,n)</code>
séquence de x à y	<code>seq(x,y,by=length=)</code>
tri	<code>sort(v,index.return=)</code>
matrice diagonale	<code>diag(v)</code>
éléments diagonaux	<code>diag(M)</code>
somme lignes	<code>rowSums(M)</code>
fonctions	<code>exp(), sin(), sqrt()</code>
statistiques	<code>mean(), var(), sd()</code>
variables aléatoires	<code>rnorm(), runif()</code>
densités	<code>dnorm(), dunif()</code>
transposée	<code>t()</code>
inverse matrice	<code>solve(M)</code>
déc. valeurs propres	<code>eigen(M)</code>
déc. Cholesky	<code>chol(M)</code>

structures de données

vecteur	<code>c(1,3,2)</code>
taille	<code>length(v)</code>
i^{eme} element	<code>v[i]</code>
privé élément i	<code>v[-i]</code>
éléments i à j	<code>v[i:j]</code>
taille	<code>length(v)</code>
matrice	<code>matrix(v,ncol=,nrow=)</code>
coefficient i, j matrice	<code>M[i,j]</code>
i^{eme} ligne matrice	<code>M[i,]</code>
taille	<code>dim(M)</code>
liste (objets \neq)	<code>list(nom1=,nom2=)</code>
élément "nom1"	<code>liste\$nom1</code>
i^{eme} element	<code>liste[[i]]</code>
taille	<code>length(list)</code>

syntaxe

```
fonction :
NomFonction <- function(param1,...,paramp){
  z <- ...
  return(z)
}

boucle for :
for(i in 1:10){
  ...
}

boucle while :
while(condition){
  ...
}

condition :
if(condition){
  ...
}else{
  ...
}
```

opérateurs

produit terme à terme	<code>*</code>
puissance	<code>^</code>
produit matriciel	<code>%%</code>
division entière	<code>/%</code>
modulo	<code>%%</code>
est égal	<code>==</code>
est inférieur ou égal	<code><=</code>
est différent	<code>!=</code>
et	<code>&</code>
ou	<code> </code>