TP2 ◆ Instructions R de base

```
# 04 - Creation d'un vecteur d'entiers regulierement espaces de 1
a 100
x<-1:100

# 05 - Calcul
x^2
x+1
x^2/(x+1)

# 06 - Selection des composantes superieures ou egales a 50
x[x>=50]

# 07 - Calcul du quotient de la division euclidienne
x%/%3

# 08 - Selection des composantes paires
x[x%/%2==x/2]

# 09 - Selection des composantes divisibles par 7
x[x%/%7==x/7]

# 10 - Creation d'un vecteur de 100 valeurs entre -1 et 1
x<-seq(from=-1, to=1, by=0.02)

# 11 - Calcul et verification d'une formule de trigo
sin(x)
cos(x)
sin(x)^2
cos(x)^2
sin(x)^2+cos(x)^2

# 12 - Selection des composantes dont le sinus est positif
sin(x)[sin(x)>0]

# 13 - Selection de toutes les composantes sauf la premiere
x[-1]

# 14 - Creation d'un vecteur alternant T et F 10 fois
x<-rep(c(TRUE,FALSE),10)

# 15 - Creation d'un vecteur alternant F et T 10 fois
y<-rep(c(FALSE,TRUE),10)

# 16 - Calcul logique
x+y
x*y
x|y
x&y

# 17 - Calcul sur les nombres particuliers
1/0
NA+1
Inf-2
Inf*(-3)
Inf/0
0/0
Inf/Inf
Inf+NA
Inf+NaN

# 18 - Listage des variables actuelles
ls()

# 19 - Effacement de la variable x
rm(x)

# 20 - Affichage de x
x

# 21 - On 'surdefinit' c
c<-1:5

# 22 - Ca marche quand meme
c

# 23 - Declaration de quatre variables de types differents puis
concatenation simple
aa<-T
bb<-2
cc<-"coucou"
dd<-pi
c(aa,bb,cc,dd)

# 24 - Concatenation de aa,bb,cc,dd en conservant les types
list(aa,bb,cc,dd)

# 25 - Exploration du mode de cette liste
mode(list(aa,bb,cc,dd))

# 26 - Extraction de la premiere composante de la liste
list(aa,bb,cc,dd)[1]

# 28 - Nouveau vecteur
x<-1:10

# 29 - Creation d'une matrice 3x3
A<-matrix(data=1:9,nrow=3,ncol=3)

# 30 - Creation d'une matrice 3x4
B<-matrix(data=1:12,nrow=3,ncol=4)

# 31 - Modification d'un vecteur
x<-1:3

# 32 - Calcul matriciel
A*x
A*B

# 36 - Creation directe de deux matrices
A<-1:2%*%t(1:3)
B<-2:4%*%t(2:4)

# 37 - Extraction de la premiere ligne de A
A[1,]

# 38 - Extraction de la deuxieme ligne de A
A[2,]

# 39 - Extraction des lignes paires de A
A[seq(from=2, to=dim(A)[1], by=2),]

# 40 - Calcul matriciel
A^2
A*B
B^2
A%*%A
B%*%B

# 41 - Fonctions de calcul matriciel
mean(A)
sum(A)
prod(A)

# 42 - Construction d'un vecteur contenant la moyenne de chaque
ligne de A
apply(X=t(A), FUN=mean, MARGIN=1)

# 43 - Construction d'un vecteur contenant la moyenne de chaque
colonne de A
apply(X=A, FUN=mean, MARGIN=1)

# 44 - Extraction de la ligne de A ayant la somme la plus elevee
A[2,]

# 47 - Premiere boucle
for(i in 1:10) print(i)

# 48 - Somme de N premiers entiers avec boucle (plus lent)
s=0
unix.time(for(i in 1:1000000) s<-s+i)
s

# 49 - Somme des N premiers entiers sans boucle
unix.time(sum(seq(from=1, to=1000000, by=1)))

# 50 - Premiere fonction calculant l'ecart-type d'un vecteur
et<-function(vect)
          {
          sqrt(sum((mean(vect)-vect)^2)/length(vect))
          }

# 51 - Fonction calculant le produit de deux fonctions
tensor<-function(f,g,x,y) f(x)*g(y)

# 52 - Personnalisation de l'interface
.First<-function()
          {
          cat("Bonjour !")
          options(prompt=":)")
          }

# 56 - Lecture d'un vecteur dans un fichier enregistre sur le
disque
A<-scan("z:\profil/r/data.txt")

# 57 - Recuperation d'un tableau dans un fichier texte
B<-read.table("z:\profil/r/data.txt")

# 59 - Sauvegarde dans un fichier
write.table(A+B,"z:\profil/r/sauv.txt")

# 60 - Trace d'une fonction
plot(sin,-pi,pi)

# 61 - Rajout des axes sur le graphe
abline(h=0,v=0,col=3)

# 62 - Ajout de la premiere bissectrice
abline(0,1,col=5)

# 63 - Decoupage d'une fenetre en deux
par(mfrow=c(1,2))

# 64 - Trace dans une meme fenetre de deux courbes
plot(sin,-pi,pi)
plot(tan,-1,1)

# Fin du TP
```

```
# [A] Fonction sur R²

# 1 - Schema a un point : approximation par une fonction constante par morceaux
Integ1<-function(f,a,b,n)
        {
        pas<-1/n
        seg<-f(seq(from=a+pas,to=b,by=pas))
        sum(seg)*pas
        }


# 2 - Schema a deux points : approximation par une fonction lineaire par morceaux
Integ2<-function(f,a,b,n)
        {
        pas<-1/n
        seg<-f(seq(from=a+pas/2,to=b-pas/2,by=pas))
        sum(seg)*pas
        }


# 3 - Schema a trois points : approximation par une fonction polynomiale de degre 2 par morceaux (ne
marche pas tres bien)
Integ3<-function(f,a,b,n)
        {
        pas<-1/n
        somme=0
        for(i in 0:n-1)
                {
                x1<-a+i*pas
                x3<-a+pas*i+pas
                x2<-x1+pas/2
                M<-matrix(data=c(x1^2,x1,1,x2^2,x2,1,x3^2,x3,1),nrow=3,ncol=3)
                Y<-matrix(data=c(f(x1),f(x2),f(x3)),nrow=3,ncol=1)
                L<-solve(M,Y)
                X<-c((x1^3-x3^3)/3,(x1^2-x3^2)/2,x1-x3)
                somme=somme+L%*%X
                }
        somme
        }

# 4 - Application : densite gaussienne

N<-function(u) {1/sqrt(2*pi)*exp(-u^2/2)}
a<-1.959964
Integ1(N,-a,a,100)
Integ2(N,-a,a,100)
Integ3(N,-a,a,100)


# [B] Fonctions sur R²

# 1 - Algorithme a un point

IntegDouble1<-function(f,a,n)
        {
        pas<-2*a/n
        tmp<-seq(from=-a,to=a,by=pas)
        n=length(tmp)
        col1<-rep(tmp,n)
        col2<-as.vector(t(matrix(data=col1,nrow=n,ncol=n)))
        coord<-f(col1,col2)
        sum(coord)*pas^2
}

# 2 - Application numerique :

N<-function(x,y) {1/(2*pi)*exp(-(x^2+y^2)/2)}
IntegDouble1(N,1,100)
IntegDouble1(N,2,100)
```

```
# Calcul d'intégrales multiples par la Méthode de Monte-Carlo
# Christophe REQUIN & Jeremie OSMONT



# Question 3

montecarlo1 <- function(f,a,b,n)
  {
    x <- runif(n,a,b)
    sum(f(x))/n
  }

f <- function(x) exp(-x²/2)
montecarlo1(f,0,1,10000)



# Question 4

histmontecarlo <- function(f,a,b,n,e)
  {
    a <- f(matrix(data=runif(n*e),nrow=e,ncol=n))
    Estimation <- apply(X=a, FUN=mean, MARGIN=1)
    hist(Estimation,nclass=50)
  }



# Question 5

montecarlo2 <- function(n)
  {
    x <- runif(n,-1,1)
    y <- runif(n,-1,1)
    # Selectionne le nombre de points (x,y) qui sont dans le disque de rayon 1 :
    masque <- sqrt(x^2+y^2) < 1
    # Les compte, puis divise par le nombre de points, et multiplie par
    # l'aire du carre i.e. 4 :
    length(x[masque])*4/n
  }



# Question 6

montecarlon <- function(dim,n,rayon)
  {
    points <- matrix(data=runif(n*dim,-1,1),nrow=dim,ncol=n)
    # On applique la fonction somme pour chaque colonne (x,y,z,t,.....) :
    norme<-sqrt(apply(points^2,2,sum))
    # Calcul du hyper-volume : 2^dim :
    (sum(norme<rayon)*2^dim)/n
  }


# Annexe : Dessine moi un Disque...

x<-runif(n=80000,-1,1)
y<-runif(n=80000,-1,1)
r<-cbind(x,y)
plot(r[sqrt(x^2+y^2)<1,])
```

```
# RECONNAISSANCE DE FORMES
# Lundi 29/04/02 - 18:24:48


# 0 - Chargement des fichiers :

matvar <- read.table("z:/r/var.txt")
matvar <- as.matrix(matvar)
coord <- read.table("z:/r/coord.txt")


# 1 - Tracer quelques formes et observons les :

panorama <- function(na,nb)
  {
    x <- runif(na*nb,1,100)
    par(mfrow=c(na,nb))
    for(i in x) plot(coord[1:84,i],coord[85:168,i])
  }

# 2:4 - Calculs divers sur papier (SIGH).

# 5 - Estimation du type de chaque forme en maximisant la densité :

carrex=c(seq(from=-1, to=1, by=1/10),rep(1,21),seq(from=1,to=-1,by=-1/10),rep(-1,21))
carrey=c(rep(-1,21),seq(from=-1,to=1,by=1/10),rep(1,21),seq(from=1,to=-1,by=-1/10))
r=(1+sqrt(2))/2
cerclex=r*cos(seq(from=0,to=2*pi,by=2*pi/83.5))
cercley=r*sin(seq(from=0,to=2*pi,by=2*pi/83.5))


detsigma=prod(eigen(matvar)$values)

inverse <- function(matrice)
  {
    res <- eigen(matrice)
    res$vectors%*%diag(1/res$values)%*%solve(res$vectors)
  }

invsigma=inverse(matvar)

densiteCarre <- function(v,w)
  {
    n=84
    1/(sqrt(detsigma)*2*(pi)^(n/2))*exp(-1/2*t(v-carrex)%*%invsigma%*%(v-carrex))
    1/(sqrt(detsigma)*2*(pi)^(n/2))*exp(-1/2*t(w-carrey)%*%invsigma%*%(w-carrey))
  }

densiteCercle <- function(v,w)
  {
    n=84
    1/(sqrt(detsigma)*2*(pi)^(n/2))*exp(-1/2*t(v-cerclex)%*%invsigma%*%(v-cerclex))
    1/(sqrt(detsigma)*2*(pi)^(n/2))*exp(-1/2*t(w-cercley)%*%invsigma%*%(w-cercley))
  }

testforme <- function(i)
  {
    c(densiteCarre(coord[1:84,i],coord[85:168,i]),densiteCercle(coord[1:84,i],coord[85:168,i]))
  }
```