

5 - Quelques fonctions

Résumé

La fiche illustre diverses fonctions de base utiles en biostatistique. Sont abordées l'analyse en composante principale (`princomp`), la classification automatique (`mcclust`), l'estimation des modèles non linéaires (`nls`, `logistic`), les ajustements à des distributions de probabilités (`ms`), l'estimation des modèles autorégressifs (`arma`, `arima`). On utilise dans chaque cas les illustrations d'un ouvrage de référence. La richesse du langage tant statistique que graphique fait de S-PLUS un outil incomparable.

Plan

1 - Analyse en composantes principales	2
2 - Classification	10
3 - Modèle exponentiel	12
4 - Courbe gamma	17
5 - Modèle logistique	19
6 - Mélange de lois normales	21
7 - Séries temporelles	32

1 - Analyse en composantes principales

Lancer S-PLUS et s'assurer qu'on est bien dans le dossier de travail désiré :

Working data will be in D:\Data\DEA4_Data

Implanter le tableau V ¹ :

```
> read.table("D:\\Data\\Dea4\\V.txt",header=T,row.names=1)
      cyl  pui vit  poi lon lar
Honda.Civic 1396  90 174  850 369 166
Renault.19 1721  92 180  965 415 169
Fiat.Tipo 1580  83 170  970 395 170
Peugeot.405 1769  90 180 1080 440 169
Renault.21 2068  88 180 1135 446 170
Citroen.BX 1769  90 182 1060 424 168
  BMW.530i 2986 188 226 1510 472 175
Rover.827i 2675 177 222 1365 469 175
Renault.25 2548 182 226 1350 471 180
Opel.Omega 1998 122 190 1255 473 177
Peugeot.405.break 1905 125 194 1120 439 171
Ford.Sierra 1993 115 185 1190 451 172
  BMW.325iX 2494 171 208 1300 432 164
Audi.90.Quattro 1994 160 214 1220 439 169
Ford.Scorpio 2933 150 200 1345 466 176
Renault.Espace 1995 120 177 1265 436 177
Nissan.Vanette 1952  87 144 1430 436 169
VW.Caravelle 2109 112 149 1320 457 184
Ford.Fiesta 1117  50 135  810 371 162
Fiat.Uno 1116  58 145  780 364 155
Peugeot.205 1580  80 159  880 370 156
Peugeot.205.Rallye 1294 103 189  805 370 157
Seat.Ibiza.SX 1461 100 181  925 363 161
Citroen.AX.Sport 1294  95 184  730 350 160
> V<-read.table("D:\\Data\\Dea4\\V.txt",header=T,row.names=1)
```

On cherche à reproduire les résultats décrits dans l'ouvrage cité (recommandé) pp. 115-144 ce qui permet la comparaison avec les sorties de Statgraphics et SPAD.N. les variables sont la cylindrée, la puissance, la vitesse, le poids, la longueur et la largeur.

Statistiques élémentaires

```
> apply(V,2,mean)
      cyl  pui  vit  poi  lon  lar
1906 113.7 183.1 1111 421.6 168.8
> apply(V,2,var)
      cyl  pui  vit  poi  lon  lar
278688 1504 635.8 53034 1709 58.58
```

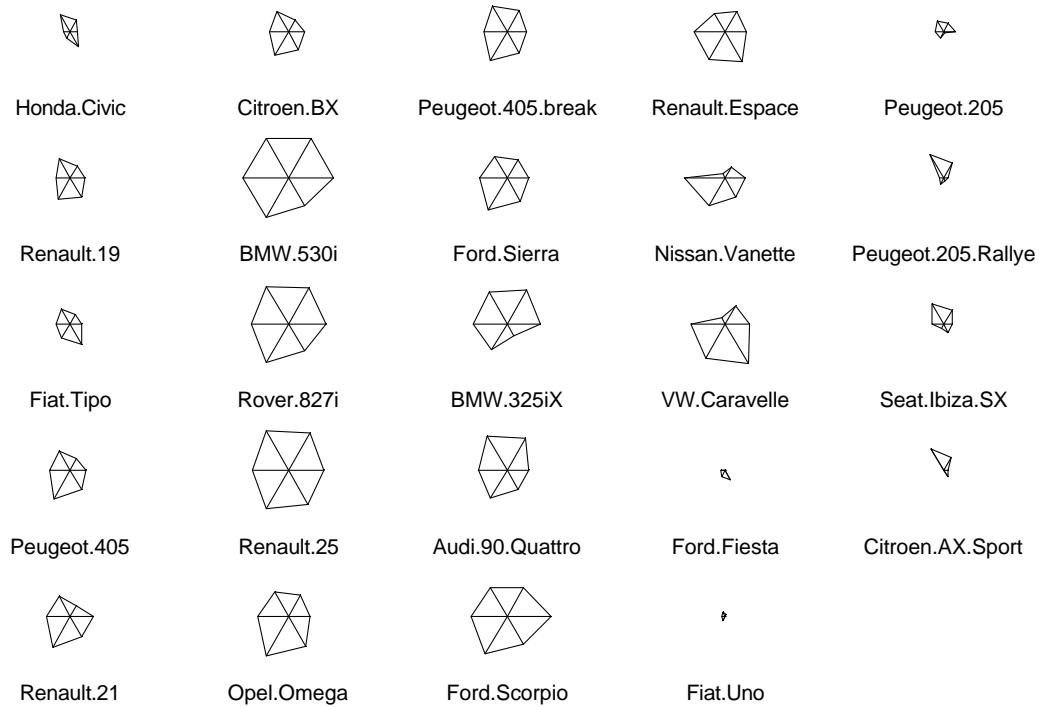
Graphiques en étoiles (p. 117) :

stars

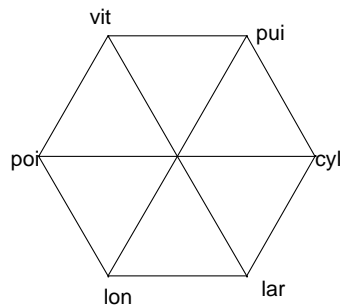
Each star represents one row of the input x. Variables (columns) start on the right and wind counterclockwise around the star. The size of a (scaled) column is shown by the distance from the center to the point on the star representing the variable.

¹ Tenenhaus, M. (1994) Méthodes statistiques en gestion. Dunod, Paris. 1-373.

```
> stars(as.matrix(V))
```



```
> starsymb(matrix(1,nrow=1,ncol=6))
> starsymb(matrix(1,nrow=1,ncol=6),collab=names(V))
```



Comparer avec Figures 1 et 2 op.cit. p.116-117

Variations et biais

```
> apply(V,2,var,unbiased=F)
  cyl pui vit poi lon lar
267076 1442 609.3 50824 1638 56.14
> apply(V,2,min)
  cyl pui vit poi lon lar
1116 50 135 730 350 155
> apply(V,2,max)
  cyl pui vit poi lon lar
2986 188 226 1510 473 184

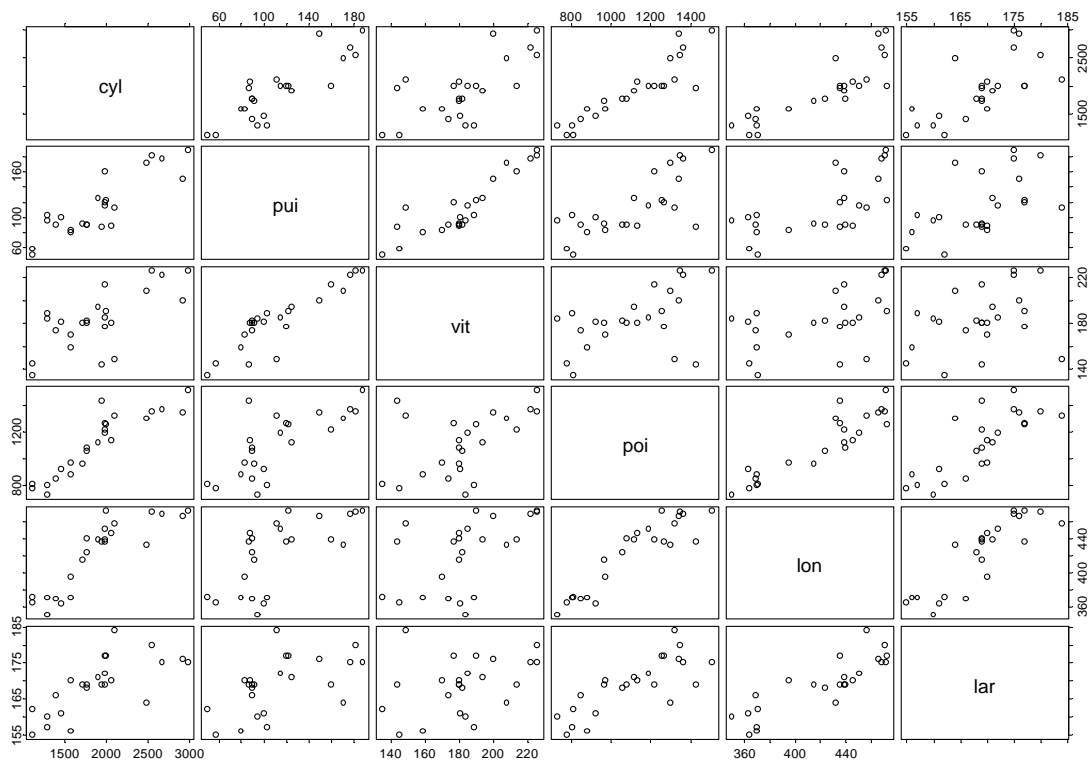
> cor(V)
      cyl      pui      vit      poi      lon      lar
cyl 1.0000 0.8610 0.6933 0.9049 0.8642 0.7091
pui 0.8610 1.0000 0.8940 0.7461 0.6885 0.5523
vit 0.6933 0.8940 1.0000 0.4914 0.5319 0.3632
poi 0.9049 0.7461 0.4914 1.0000 0.9171 0.7909
lon 0.8642 0.6885 0.5319 0.9171 1.0000 0.8638
```

```
lar 0.7091 0.5523 0.3632 0.7909 0.8638 1.0000
```

Comparer avec Tableau 2 op.cit. p.118

pairs

```
> pairs(V)
```



Comparer avec Figure 3 op.cit. p.119

princomp

```
> princomp(V,cor=T)
```

Standard deviations:

```
Comp. 1 Comp. 2 Comp. 3 Comp. 4 Comp. 5 Comp. 6  
2.158 0.9567 0.4903 0.3205 0.2543 0.1448
```

The number of variables is 6 and the number of observations is 24

Component names:

```
"sdev" "loadings" "correlations" "scores" "center" "scale"
```

```
"n.obs" "call" "factor.sdev" "coef"
```

Call:

```
princomp(x = V, cor = T)
```

Une ACP est un objet liste

```
> pcal<-princomp(V,cor=T)
```

Axes principaux

```
> pcal$loadings
```

	Comp. 1	Comp. 2	Comp. 3	Comp. 4	Comp. 5	Comp. 6
cyl	0.444		-0.401		-0.799	
pui	0.414	-0.421		-0.490	0.307	-0.562
vit	0.344	-0.663	0.370	0.320		0.450
poi	0.430	0.255	-0.484	-0.123	0.473	0.526
lon	0.430	0.296		0.712	0.166	-0.437
lar	0.378	0.478	0.681	-0.365	-0.131	0.119

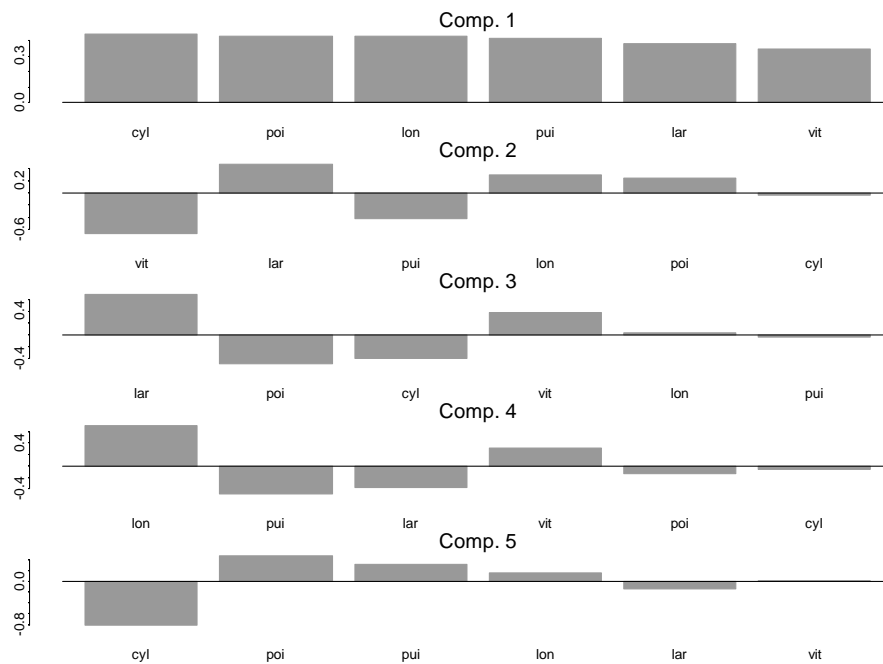
```
> print.matrix(pcal$loadings) à méditer !
```

	Comp. 1	Comp. 2	Comp. 3	Comp. 4	Comp. 5	Comp. 6
cyl	0.4442	-0.03396	-0.40143	-0.05002	-0.798600	-0.01086
pui	0.4145	-0.42122	-0.03956	-0.48972	0.306665	-0.56155
vit	0.3435	-0.66344	0.36993	0.31991	0.007185	0.45010
poi	0.4303	0.25517	-0.48446	-0.12315	0.472589	0.52584
lon	0.4302	0.29558	0.04398	0.71185	0.165972	-0.43743
lar	0.3776	0.47832	0.68103	-0.36529	-0.131360	0.11880

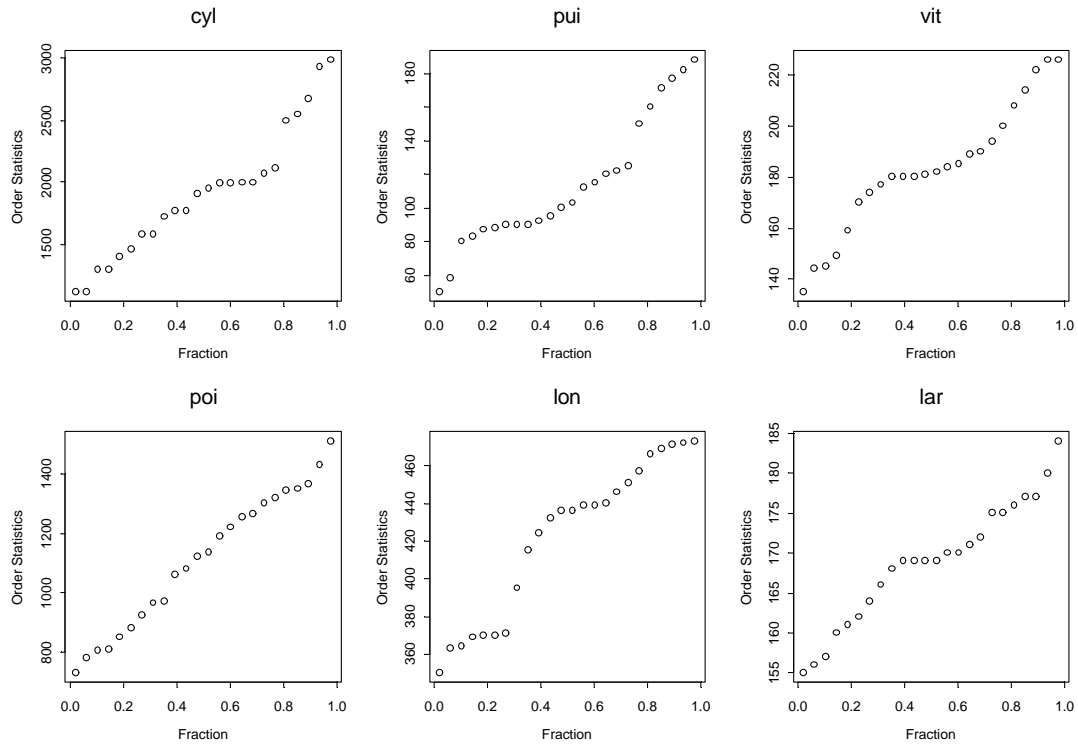
Comparer avec Tableau 4 op.cit. p.124

plot est une fonction générique : voir ? plot.xxx

```
> plot(loadings(pca1))
> plot(pca1$loadings)
> plot.loadings(pca1$loadings)
```



```
> plot.dataframe(V)
```

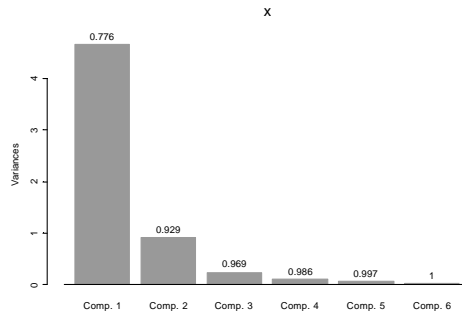


Valeurs propres et valeurs singulières

```

> names(pcal)
[1] "sdev"          "loadings"      "correlations"  "scores"
[5] "center"        "scale"         "n.obs"        "call"
[9] "factor.sdev"  "coef"
> pcal$sdev
Comp. 1 Comp. 2 Comp. 3 Comp. 4 Comp. 5 Comp. 6
  2.158  0.9567  0.4903  0.3205  0.2543  0.1448
> pcal$sdev2
NULL
> pcal$sdev^2
Comp. 1 Comp. 2 Comp. 3 Comp. 4 Comp. 5 Comp. 6
  4.656  0.9152  0.2404  0.1027  0.06466  0.02096
> var(pcal$scores,unbiased=F)
          Comp. 1      Comp. 2      Comp. 3      Comp. 4
Comp. 1  4.656e+000 -1.665e-016  3.516e-016 -6.800e-016
Comp. 2 -1.665e-016  9.152e-001  8.789e-017 -2.012e-016
Comp. 3  3.516e-016  8.789e-017  2.404e-001 -1.272e-016
Comp. 4 -6.800e-016 -2.012e-016 -1.272e-016  1.027e-001
Comp. 5 -1.041e-016 -7.517e-017 -7.864e-017  1.457e-016
Comp. 6 -4.684e-016  1.318e-016 -2.220e-016 -3.601e-016
...
          Comparer avec Tableau 4 op.cit. p.124
> plot(pcal)
[1] 0.7 1.9 3.1 4.3 5.5 6.7

```



Cercle des corrélations

```
> cor(V,pcal$scores)
      Comp. 1  Comp. 2  Comp. 3  Comp. 4  Comp. 5  Comp. 6
cyl  0.9585 -0.03249 -0.19684 -0.01603 -0.203065 -0.001573
pui  0.8944 -0.40297 -0.01940 -0.15695  0.077978 -0.081301
vit  0.7413 -0.63469  0.18139  0.10253  0.001827  0.065164
poi  0.9285  0.24411 -0.23755 -0.03947  0.120168  0.076130
lon  0.9283  0.28278  0.02157  0.22814  0.042203 -0.063330
lar  0.8148  0.45759  0.33393 -0.11707 -0.033402  0.017199
```

Comparer avec Tableau 6 op.cit. p.125

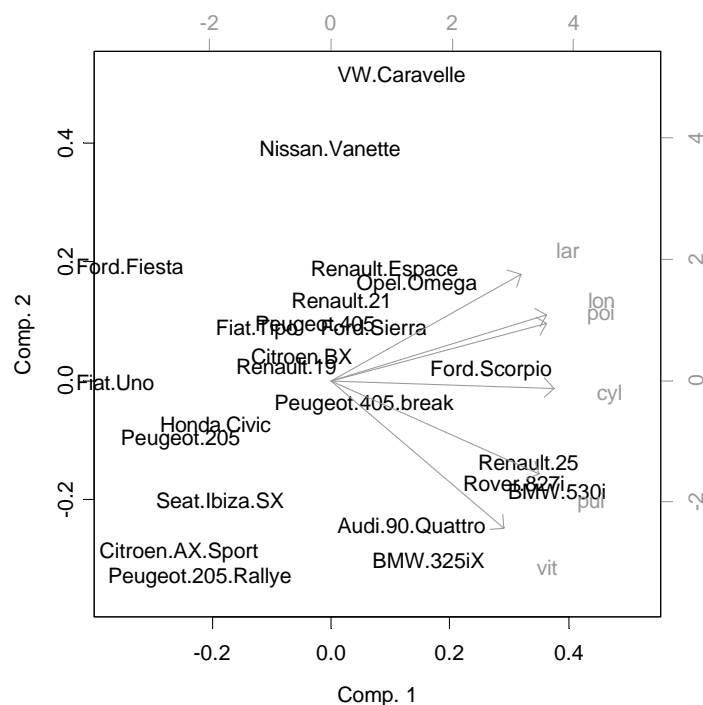
Coordonnées factorielles

```
> pcal$scores
      Comp. 1  Comp. 2  Comp. 3  Comp. 4  Comp. 5
Honda.Civic -2.02290 -0.31994  0.53062 -0.407377 -0.11826
Renault.19 -0.77851  0.13290  0.44155  0.213162 -0.22545
Fiat.Tipo -1.28979  0.43421  0.46900 -0.189979 -0.17224
Peugeot.405 -0.27382  0.46470  0.18639  0.611229  0.02782
Renault.21  0.18051  0.63715 -0.06456  0.634822 -0.32801
Citroen.BX -0.50465  0.20761  0.15106  0.415396 -0.06161
BMW.530i  3.94563 -0.83539 -0.51558 -0.138997 -0.12044
Rover.827i  3.19390 -0.77144 -0.01415  0.007588 -0.04612
...
Peugeot.205 -2.61879 -0.41507 -0.79900 -0.001865 -0.24542
Peugeot.205.Rallye -2.28877 -1.47879  0.10085  0.110174  0.21629
Seat.Ibiza.SX -1.93307 -0.90142 -0.04755 -0.354670  0.08438
Citroen.AX.Sport -2.65026 -1.29510  0.44636 -0.308521 -0.14162
```

Comparer avec Tableau 5 op.cit. p.126

biplot (Version Windows)

```
> biplot(pcal)
```



Comparer avec Figure 10 op.cit. p.137

DESCRIPTION

Produces a biplot corresponding to the model.

USAGE

```
biplot.princomp(x, choices=1:2, scale=1, ...)
biplot.factanal(x, choices=1:2, scale=1, ...)
```

REQUIRED ARGUMENTS

x an object of class "princomp" or "factanal".

OPTIONAL ARGUMENTS

choices length 2 vector stating which components or factors to plot.
scale a number between 0 and 1, inclusive that tells what kind of scaling to perform. The basic scaling factors, lambda, are the singular values of the data in the case of principal components and the square root of the sum of squared loadings in the case of factor analysis. The variables are scaled by λ^{scale} and the observations are scaled by $\lambda^{(1-\text{scale})}$. See Gabriel (1971).

... optional arguments to biplot.default may be given.

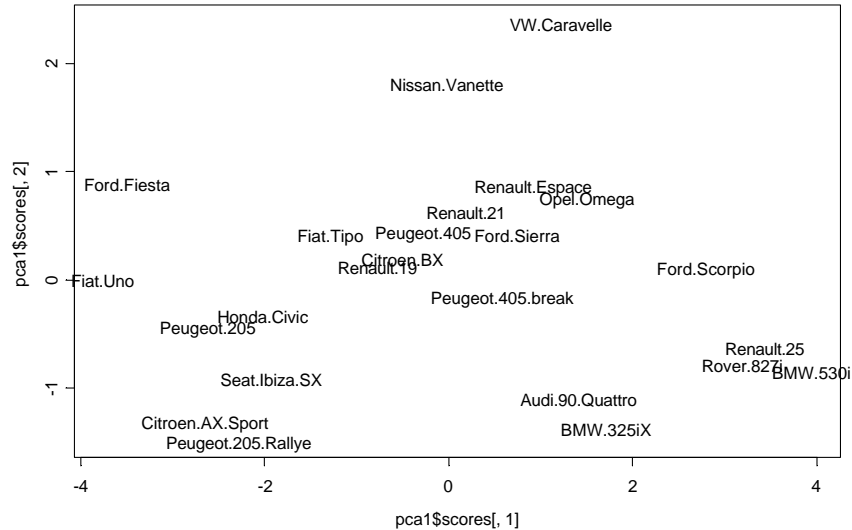
REFERENCES

Gabriel, K. R. (1971). The biplot graphical display of matrices with applications to principal component analysis. *Biometrika* 58 453-467.

Moralité : *S-PLUS* représente toujours une école de statistique.

Cartes factorielles

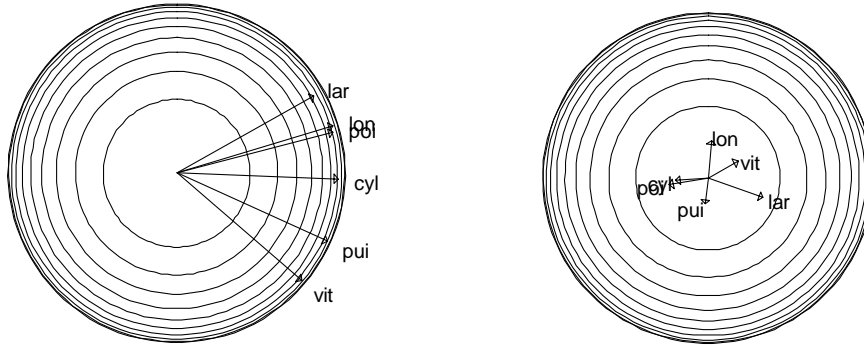
```
> plot(pca1$scores[,1],pca1$scores[,2],type="n")
> text(pca1$scores[,1],pca1$scores[,2],row.names(V))
```



Par exemple :

```
corcircle<-function (pca.object, axex=1,axey=2) {
  if (class(pca.object)!="princomp") stop("Illegal call")
  cool<-pca.object$coef[,axex]*pca.object$sdev[axex]
  coo2<-pca.object$coef[,axey]*pca.object$sdev[axey]
  old.par<-par()
  par(fty="s")
  par(mai=c(0,0,0,0))
  x0<-sin(seq(0,2*pi,le=100))
  y0<-cos(seq(0,2*pi,le=100))
  plot(x0,y0,xlim=c(-1,1),ylim=c(-1,1),type="l",axes=F)

  arrows(0,0,cool,coo2)
  for(i in cos(asin(seq(0.1,1,by=0.1)))) {
    lines( i*x0,i*y0)
  }
  nomvar<-names(eval(pca.object$call[[2]]))
  for (i in 1:length(cool)) {
    if (cool[i]<0) posi<-1 else posi<-0
    text(1.1*cool[i],1.1*coo2[i],nomvar[i],cex=2,adj=posi)
  }
  par(old.par)
  return("ok")
}
> corcircle(pca1,1,2)
> corcircle(pca1,3,4)
```



2 - Classification

Nous avons réalisé une classification ascendante hiérarchique des données centrées-réduites de l'exemple à l'aide du critère de Ward (op. cit. p. 139) :

```
> norma<-function (x) {
+ return ((x-mean(x))/sqrt(var(x,unbiased=T)))
+ }
> apply(V,2,norma)
           cyl      pui      vit      poi      lon      lar
Honda.Civic -0.966313 -0.61021 -0.36023 -1.1326 -1.27196 -0.37019
Renault.19  -0.350676 -0.55864 -0.12228 -0.6333 -0.15925  0.02178
Fiat.Tipo    -0.617768 -0.79070 -0.51886 -0.6115 -0.64303  0.15243
Peugeot.405 -0.259751 -0.61021 -0.12228 -0.1339  0.44549  0.02178
...
> Vnor<-apply(V,2,norma)
```

mclust

DESCRIPTION

Performs hierarchical clustering via a wide range of clustering options, calculates a Bayesian criterion for choosing the number of clusters, and optionally allows for noise or "outliers".

USAGE

```
mclust(x, method = "S*", signif = rep(0, dim(x)[2]), noise = F,
       scale = rep(1, dim(x)[2]), shape = c(1, rep(0.2, (dim(x)[2]-1))),
       workspace = <<see below>>)
```

REQUIRED ARGUMENTS

x n by p matrix containing n p-dimensional data points (NAs not allowed).

OPTIONAL ARGUMENTS

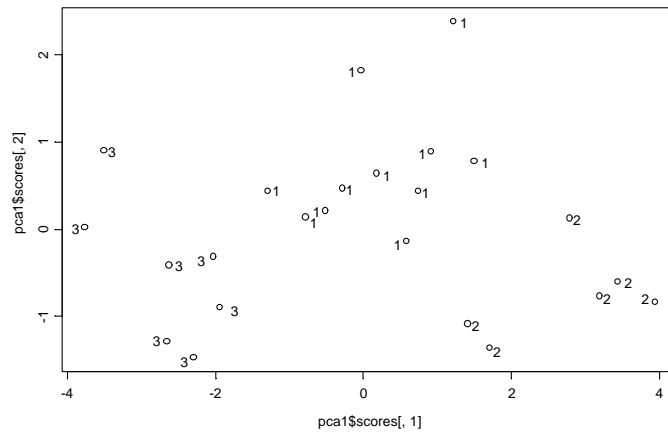
method a character string to select the clustering criterion. Possible values are : "S*", "S", "spherical" (with varying sizes), "sum of squares" or "trace" (Ward's method), "unconstrained", "determinant", "centroid", "weighted average link", "group average link", "complete link" or "farthest neighbor", "single link" or "nearest neighbor". Only enough of the string to determine a unique match is required.


```

[22] -507.415 -530.126 -565.490

> cutree(clust0$tree,k=3)
[1] 3 1 1 1 1 1 2 2 2 1 1 1 2 2 2 1 1 1 3 3 3 3 3 3
attr(,"height"):
[1] 21 20 19
> numgroup<-cutree(clust0$tree,k=3)
> numgroup
[1] 3 1 1 1 1 1 2 2 2 1 1 1 2 2 2 1 1 1 3 3 3 3 3 3
attr(,"height"):
[1] 21 20 19
plot(pca1$scores[,1],pca1$scores[,2])
> identify(pca1$scores[,1],pca1$scores[,2],labels=as.vector(numgroup))

```



Comparer avec Figure 12 op.cit. p.143

```

> apply(V[numgroup==1,],2,mean)
cyl   pui   vit   poi   lon   lar
1896 102.2 175.5 1163 437.5 172.4
> apply(V[numgroup==2,],2,mean)
cyl   pui vit   poi   lon   lar
2605 171.3 216 1348 458.2 173.2
> apply(V[numgroup==3,],2,mean)
cyl   pui   vit   poi   lon   lar
1323 82.29 166.7 825.7 365.3 159.6

```

Comparer avec Tableau 8 op.cit. p.141

3 - Modèle exponentiel

Lancer S-PLUS et s'assurer qu'on est bien dans le dossier de travail désiré :

Working data will be in D:\Data\DEA3_Data

Le problème est posé par Didier Pont². Dans plusieurs stations (STA) de Camargue, à plusieurs dates (DAT en semaines), est mesuré le taux (DEG, pourcentage, sur [0,1]) de dégradation des feuilles d'une espèce de Salicornes. Les erreurs de mesure peuvent conduire à une valeur plus grande que 1 :

² Rioual P., Torre, F. & D. Pont (1996) - Primary production of Salicornia-type vegetation within the Rhone delta. MEDDELT Program "Impact of Climate Change on Northwestern Mediterranean Deltas. Final report to the Commission of the European Communities (DG XII, contract EV5V-CT94-0465). 17 p.

STA	DAT	DEG	STA	DAT	DEG	STA	DAT	DEG	STA	DAT	DEG
PAL	0	1	DS	0	1	DN	0	1	TDV	0	1
PAL	1	1.018	DS	1	0.941	DN	1	0.976		TDV	1
PAL	4	0.921	DS	4	0.836	DN	4	0.86	TDV	4	
PAL	8	0.648	DS	8	0.652	DN	8	0.566		TDV	8
PAL	16	0.392	DS	16	0.409	DN	16	0.471		TDV	16
PAL	43	0.166	DS	43	0.249	DN	43	0.278		TDV	43
PAL	0	1	DS	0	1	DN	0	1	TDV	0	1
PAL	1	1.038	DS	1	0.962	DN	1	0.977		TDV	1
PAL	4	0.93	DS	4	0.886	DN	4	0.852		TDV	4
PAL	8	0.657	DS	8	0.601	DN	8	0.628		TDV	8
PAL	16	0.356	DS	16	0.357	DN	16	0.509		TDV	16
PAL	43	0.159	DS	43	0.244	DN	43	0.241		TDV	43

On veut estimer un modèle de la cinétique de dégradation du type :

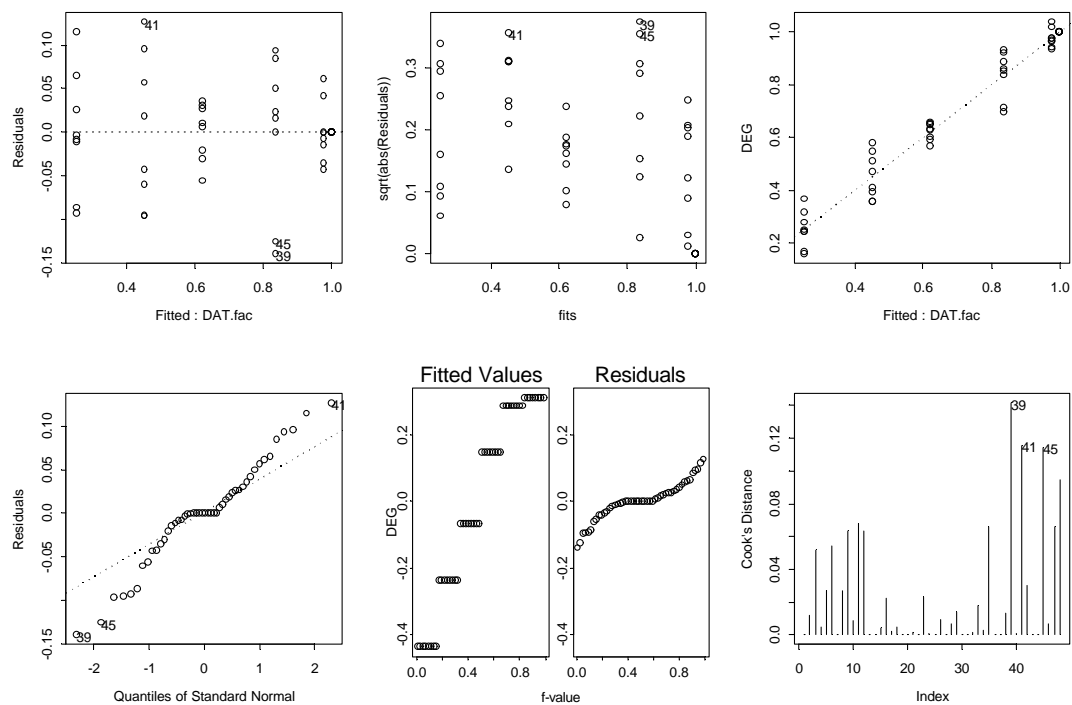
$$y = pe^{kt} + (1 - p)$$

Implanter le data.frame :

```
> salicor
  STA DAT  DEG
1 PAL  0 1.000
2 PAL  1 1.018
...
47 TDV 16 0.548
48 TDV 43 0.367
```

attach

```
> attach(salicor) Les variables de salicor sont directement accessibles
> search()
[1] "D:\\Data\\Dea3\\_Data"
[2] "salicor"
[3] "d:\\asplus\\splus\\_Functio"
...
> is.factor(STA)
[1] T
> levels(STA)
[1] "DN" "DS" "PAL" "TDV"
```



Graphique standard associé à un modèle linéaire

- 1) Résidus en fonction des valeurs prédites
- 2) Racine des valeurs absolues des résidus en fonction des valeurs prédites
- 3) Valeurs observées en fonction des valeurs prédites

4) Graphique quantile-quantile normal des résidus (normalité des résidus). *N.B.* Chacun des graphiques proposés est issu d'une recherche approfondie. Le qq-plot est de Wilk M.B. & Gnanadesikan R. (1968). *Probability plotting methods for the analysis of data*. Biometrika, 55, 1-17 validé par Cleveland W.S. (1994) *The elements of graphing data*. Hobart Press, Summit, New Jersey, p. 143. Les modes de lecture sont décrits dans des ouvrages célèbres comme Tuckey J.W. (1977) *Exploratory data analysis*, Adison-Wesley, Reading, Massachussets. Ici, les résidus sont sur-dispersés par rapport à une loi normale (cf. du Toit S.H.C., Steyn A.G.W. & Stumpf R.H. (1986) *Graphical Exploratory data analysis*, Springer-Verlag, , New-York, p. 49). Ouvrages classiques : Chambers J.M., Cleveland W.S., Kleiner B. & Tukey P.A. (1983) *Graphical methods for data analysis*, Wadsworth, Belmont, California. Cleveland W.S. (1993) *Visualizing data*, Hobart Press, Summit, New Jersey.

5) graphique r-f (r pour residuals, f pour fitted). A gauche, en abscisse le rang des valeurs prédites sur [0,1], en ordonnée les valeurs prédites centrées (fonction de répartition inversée des prédictions). A droite à la même échelle en abscisse le rang des résidus sur [0,1], en ordonnée les valeurs observées des résidus (fonction de répartition inversée des résidus). Le couple permet de comparer l'étendue de la distribution des observations qu'on espère beaucoup plus grande que celle des résidus. Ce graphe exprime le rapport variance expliquée - variance résiduelle.

6) Graphe des distances de Cook. Donne pour chacun des points de mesure la distance entre les paramètres estimés par la régression avec et sans ce point. Si l'importance du rôle de chaque point est concentré sur quelques valeurs, la régression n'est pas bonne (prise en compte de points aberrants). Voir Cook, R. D. and Weisberg, S. (1982). *Residuals and Influence in Regression*. Chapman and Hall, New York.

La question est bien posée :

```
> DAT.fac<-factor(DAT)
> lm1<-lm(DEG~STA*DAT)
> anova(lm1)
Analysis of Variance Table
```

```

Response: DEG

Terms added sequentially (first to last)
      Df Sum of Sq Mean Sq F Value Pr(F)
    STA  3     0.003   0.001    0.0 0.9856
    DAT  1     3.005   3.005  170.6 0.0000
 STA:DAT  3     0.063   0.021    1.2 0.3246
Residuals 40     0.704   0.018

```

```

> lml<-lm(DEG~DAT*STA)

> anova(lml)
Analysis of Variance Table

```

```

Response: DEG

Terms added sequentially (first to last)
      Df Sum of Sq Mean Sq F Value Pr(F)
    DAT  1     3.005   3.005  170.6 0.0000
    STA  3     0.003   0.001    0.0 0.9856
 DAT:STA  3     0.063   0.021    1.2 0.3246
Residuals 40     0.704   0.018

```

```

> lml<-lm(DEG~DAT.fac)
> anova(lml)
Analysis of Variance Table

```

```

Response: DEG

Terms added sequentially (first to last)
      Df Sum of Sq Mean Sq F Value Pr(F)
 DAT.fac  5     3.617   0.7235  192.7    0
Residuals 42     0.158   0.0038

```

tapply

```

> tapply(DEG,list(DAT,STA),mean)
      DN      DS      PAL      TDV
0 1.0000 1.0000 1.0000 1.0000
1 0.9765 0.9515 1.0280 0.9515
4 0.8560 0.8610 0.9255 0.7040
8 0.5970 0.6265 0.6525 0.6115
16 0.4900 0.3830 0.3740 0.5635
43 0.2595 0.2465 0.1625 0.3420
> datsta<-tapply(DEG,list(DAT,STA),mean)

```

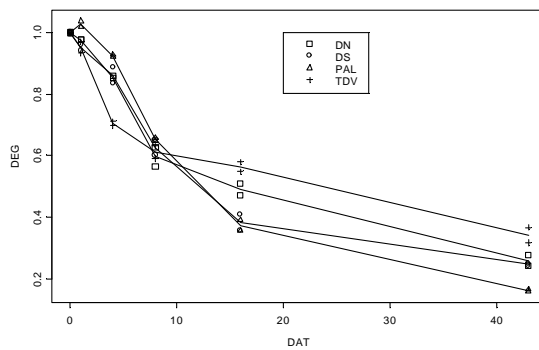
function

```

f1<- function () {
  plot(DAT,DEG,type="n")
  for (i in 1:4) {
    nomsta<-levels(STA)[i]
    points(DAT[STA==nomsta],DEG[STA==nomsta],pch=i-1)
    lines(unique(DAT),datsta[,i])
  }
  legend(20,1,levels(STA),marks=c(0,1,2,3))
}

> f1()

```



Mettre en place la formule (*objet*) du modèle :

```
> formule<-DEG~p*exp(a*DAT)+(1-p)
> formule
DEG ~ p * exp(a * DAT) + (1 - p)
```

Associer aux données une liste de paramètres :

```
> parameters(salicor)<-list(a=-1,p=0.8)
> parameters(salicor)
$a:
[1] -1

$p:
[1] 0.8
> class(salicor)
[1] "pframe"      "data.frame"
```

salicor est un tableau de données paramétré (*pframe*) sous-classe des *data.frame*.

Estimer les paramètres par la procédure nls :

nls

DESCRIPTION

Fits a nonlinear regression model via least squares.

USAGE

```
nls(formula, data, start=<<see below>>, control=<<see below>>,
    algorithm="default", trace=F)
```

REQUIRED ARGUMENTS

formula a formula which specifies the nonlinear regression model.

data a data frame in which to do the computations. In addition to the usual data variables, the data frame may contain parameters (set, typically, by using the assignment form of parameters or param) that establish initial values for the model parameters.

OPTIONAL ARGUMENTS

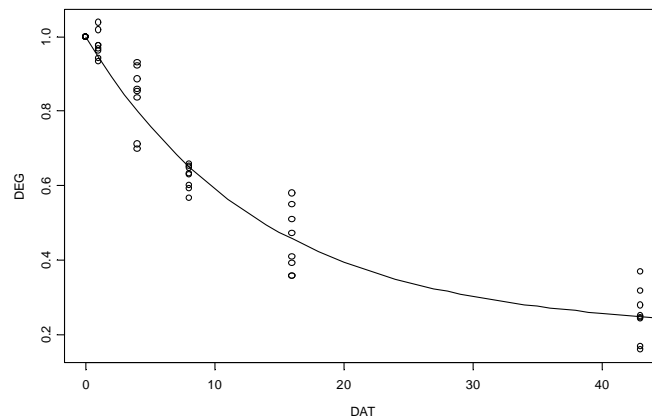
start a list or numerical vector. Although it is optional, use of start is recommended for unambiguous specification of the parameters. If start is omitted, the assumption is that any names occurring in formula that are not variables in the data frame are parameters. The list form of start allows the individual parameter names to refer to subsets of the parameters of arbitrary length. If a numeric starting vector is supplied, the named parameters must each be of length 1. In the case of partially

linear models (algorithm = "plinear"), only the nonlinear parameters should be supplied.

```
> nls(formule, salicor)
Residual sum of squares : 0.1839
parameters:
      a      p
-0.07371 0.7844
formula: DEG ~ p * exp(a * DAT) + (1 - p)
48 observations
> nls1<-nls(formule, salicor)
> class(nls1)
[1] "nls"
> names(nls1)
[1] "parameters"      "formula"          "call"             "residuals"
[5] "R"                "fitted.values"   "assign"
```

```
> x0<-seq(from=0, to=43, by=1)
> y0<-0.7844 * exp(-0.07369 * x0) + (1 - 0.7844)
> plot(DAT,DEG)
> lines(x0,y0)
```

On obtient une première approche satisfaisante :



4 - Courbe gamma

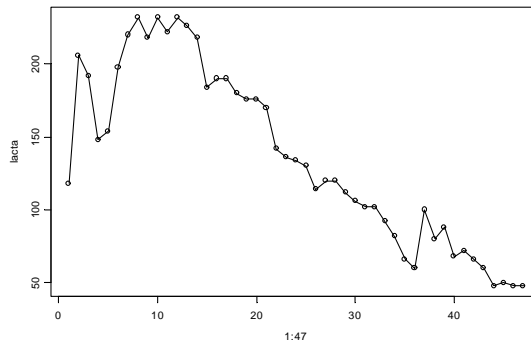
L'exercice est proposé par R. Tomassone & al. ³ p. 380. Il porte sur une courbe de lactation (47 semaines) :

```
> lacta
[1] 118 206 192 148 154 198 220 232 218 232 222 232 226 218 184 190
[17] 190 180 176 176 170 142 136 134 130 114 120 120 112 106 102 102
[33] 92 82 66 60 100 80 88 68 72 66 60 48 50 48 48
```

On veut ajuster aux données une fonction Gamma :

$$y = ax^d e^{-cx}$$

³ Tomassone R., Dervin C. & Masson J.P. (1993) Biométrie Modélisation des phénomènes biologiques, Masson, Paris, 553 p.



lacta n'est pas un data.frame :

```
> is.data.frame(lacta)
[1] F
```

Modifier la structure des données :

```
> lacta<-data.frame(lacta)
> lacta<-cbind.data.frame(lacta,1:47)
> names(lacta)
[1] "lacta" "X2"
> names(lacta)<-c("lacta","t")
> names(lacta)
[1] "lacta" "t"
```

Obtenir une première estimation des paramètres par une régression linéaire :

```
> loglacta<-log(lacta$lacta)

> lm(loglacta~t+log(t),data=lacta)
Call:
lm(formula = loglacta ~ t + log(t), data = lacta)
Coefficients:
 (Intercept)          t log(t)
      4.851    -0.0616  0.4929
> exp(4.851)
[1] 127.9
> 0.4929/0.0616
[1] 8.002 (Ok, p. 380)
```

Implanter la formule :

```
> lacta.form<-lacta~a*t^d*exp(c*t)
```

Paramétrer le data.frame. Quelle est la classe du résultat ?

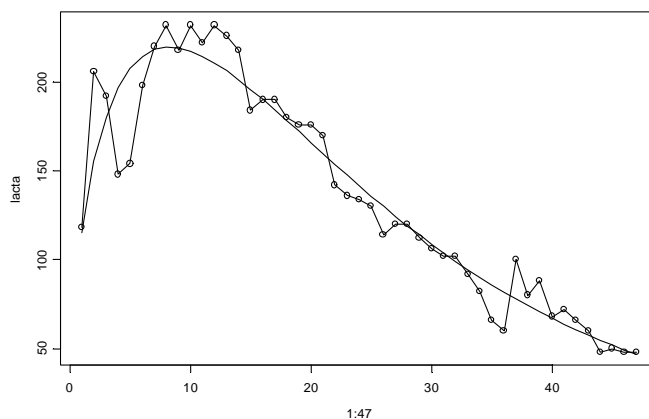
```
> parameters(lacta)<-list(a=127.88,d=0.4929,c=-0.0616)
```

Estimer le modèle :

```
> nls2<-nls(lacta.form,lacta)
> nls2
Residual sum of squares : 12772
parameters:
      a      d      c
 122.7 0.5243 -0.06347
formula: lacta ~ a * t^d * exp(c * t)
47 observations
```

Tracer le modèle :

```
> lines(lacta$t,predict(nls2))
```



Donner la précision des paramètres fournie par le jackknife.

5 - Modèle logistique

Pour le modèle logistique, il n'est même pas la peine de donner une valeur initiale aux paramètres. S-PLUS s'en occupe pour vous. Pourtant c'est un problème réel :

*Le problème, que nous n'avons pas abordé jusqu'à présent, est délicat car un bon comportement d'un algorithme de minimisation dépend fortement de la qualité des estimations initiales. Pour le modèle exponentiel on peut proposer de prendre les estimations obtenues par régression linéaire sur les logarithmes, par contre pour des modèles plus compliqués on est souvent réduit à des bricolages plus ou moins avouables.*⁴ (p. 513)

Un exemple est proposé par D. Nandris (Cf. ⁵). Dans une parcelle de caféier, lors de 9 visites en 1992, une note de synthèse de développement de la rouille est enregistrée.

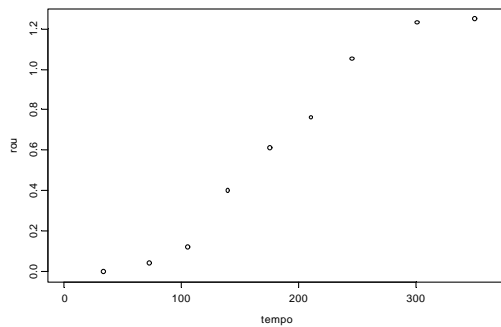
```
> tempo
[1] 34 73 106 140 176 211 246 302 351
> rou
[1] 0.00 0.04 0.12 0.40 0.61 0.76 1.05 1.23 1.25
```

Représenter les données en imposant le rang sur l'axe des x :

```
> plot(tempo,rou,xlim=c(0,364))
```

⁴ Pavé A. (1994) Modélisation en biologie et écologie. Aléas, Lyon, 559 p.

⁵ Lamouroux, N., Pellegrin, F., Nandris, D. & Kohler, F. (1995) The *Coffea arabica* fungal pathosystem in New Caledonia: interactions at two different spatial scales. *Journal of Phytopathology* : 143, 403-413.



```
> ?logistic
```

DESCRIPTION

A self-starting nonlinear regression function for the three-parameter logistic function. The parameters to be fit are *Asym*, *T50*, and *scal*. An expression for the model is

$$y = \frac{Asym}{1 + \exp\left(\frac{T50 - t}{scal}\right)}$$

USAGE

```
logistic(time, Asym, T50, scal)
```

REQUIRED ARGUMENTS

- time** the "independent" variable - usually time or concentration.
- Asym** the parameter giving the asymptotic response as time goes to infinity. It has the same units as the response.
- T50** the parameter giving the time at which the response reaches 50% of *Asym*. It has the same units as the time variable.
- scal** a scale parameter. When the time is *T50* + *scal* the response is roughly 75% of *Asym*. This parameter has the same units as the time variable.

Ceci correspond à l'écriture :

$$y(t) = \frac{K}{1 + e^{-r(t-t_m)}} \text{ avec } r = 1/scal$$

```
> rou.lis<-nlsList(rou ~ logistic(tempo, Asym, T50,
scal),cluster=rep(1,9),data=rou)
> rou.lis
$"1":
Residual sum of squares : 0.01828
parameters:
  Asym  T50  scal
  1.283 185.1 42.19
formula: rou ~ logistic(tempo, Asym, T50, scal)
9 observations

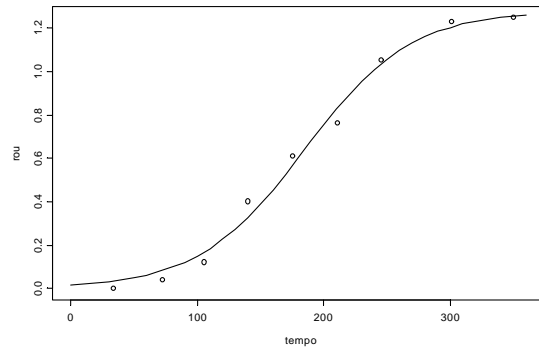
attr(,"class"):
[1] "nlsList" "lmList"
attr(,"call"):
nlsList(formula = rou ~ logistic(tempo, Asym, T50, scal), data = rou,
  cluster = rep(1, 9))

> provi<-seq(0,365,10)
> provi
```

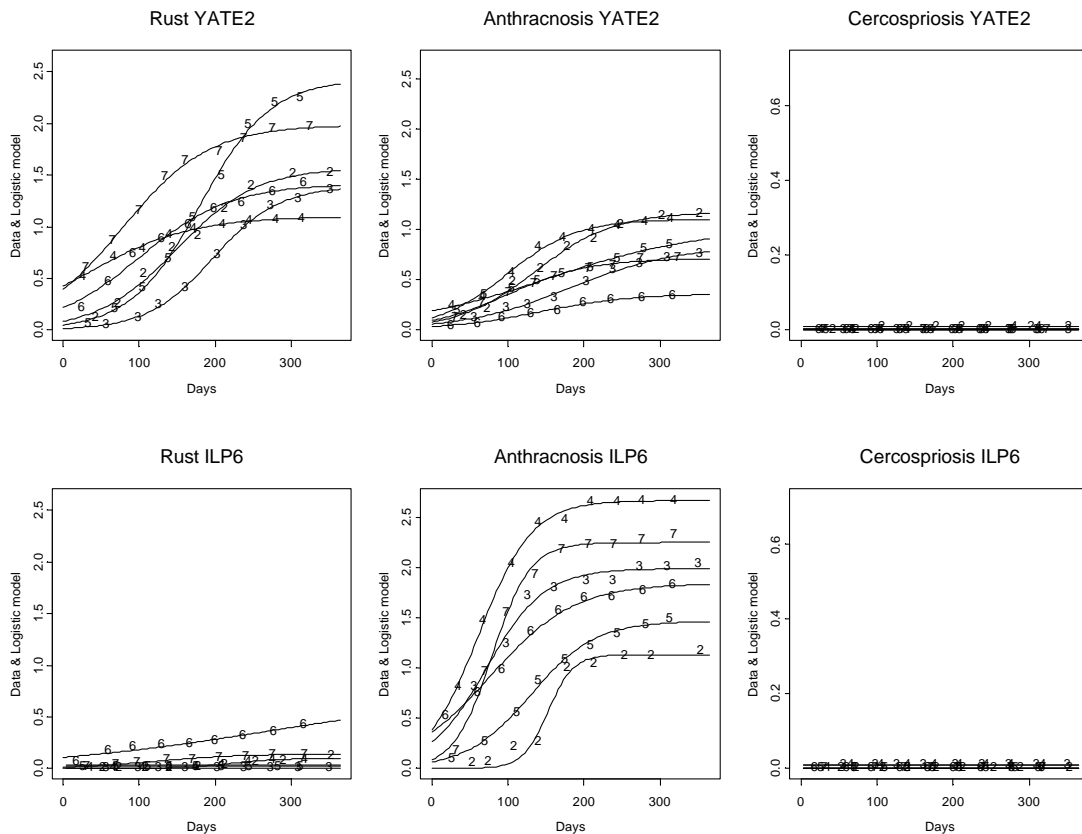
```

[1] 0 10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160
170 180 190 200 210 220
[24] 230 240 250 260 270 280 290 300 310 320 330 340 350 360
> lines(provi,1.283/(1+exp((185.1-provi)/42.19)))

```



Le paramètre `cluster` permet de calculer un modèle par groupe de lignes. On peut ainsi estimer des familles de modèles, par exemple pour plusieurs années, plusieurs stations et plusieurs maladies :



6 - Mélange de lois normales

```

> read.table("D:\\Data\\Dea4\\donax.txt",header=F,row.names=1)
V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16

```

```

      Mars  0  0  0  0  3  2  4  7  12  21  22  6  1  1  0
      Avril 0  0  0  0  0  1  3 20  28  42  79  47 18  2  1
      Mai   25 18  5  5  1  0  6  6  16  53  93  83 20  6  0
      Juin  9 11  5  7  6  7  5  2  12  34  92  77 24  1  0
      Juillet 11 9  5  2  6  8  4  1  2  9  35  30 16  6  0
      Aout  235 79 26  6  5  7  4  4  7  6  29  52 23  3  0
      Septembre 21 13 24 14 26 19 13  3  10  13  10  28 13  1  2
      Octobre 10  5  6 11 12  3  6  8  3  7  7  17 18  8  0
      Novembre 5  4 13  6  6  8 11  5  6  14  12  26  7  2  0
> donax<-read.table("D:\\Data\\Dea4\\donax.txt",header=F,row.names=1)

```

Les lignes sont des échantillons de *Donax Trunculus*⁶ d'une population échantillonnée chaque mois. En colonnes, on a des classes de tailles d'amplitude 2 mm et de centre respectif 3, 5, ..., 31 mm. On veut représenter la distribution en classes de taille et son évolution.

Transposition

```

> donax.t<-t(donax)
> donax.t
      Mars Avril Mai Juin Juillet Aout  Septembre  Octobre  Novembre
V2      0      0  25   9      11  235          21      10       5
V3      0      0  18  11      9   79          13       5       4
V4      0      0   5   5      5   26          24       6      13
V5      0      0   5   7      2    6          14      11       6
V6      3      0   1   6      6    5          26      12       6
V7      2      1   0   7      8    7          19       3       8
V8      4      3   6   5      4    4          13       6      11
V9      7      20  6   2      1    4           3       8       5
V10     12     28 16  12      2    7          10       3       6
V11     21     42 53  34      9    6          13       7      14
V12     22     79 93  92     35  29          10       7      12
V13      6     47 83  77     30  52          28      17      26
V14      1     18 20  24     16  23          13      18       7
V15      1      2  6   1      6    3           1       8       2
V16      0      1  0   0      0    0           2       0       0

```

Extraction

```

> as.vector(donax.t[,1])
[1] 0 0 0 0 3 2 4 7 12 21 22 6 1 1 0

```

Une liste d'échantillons

```

> donax.list<-as.list(1:9)
> tref<-seq(3,31,by=2)
> tref
[1] 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31
> rep(tref,as.vector(donax.t[,1]))
[1] 11 11 11 13 13 15 15 15 15 17 17 17 17 17 17 17 17 19 19 19 19 19
[22] 19 19 19 19 19 19 19 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21
[43] 21 21 21 21 21 21 21 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23
[64] 23 23 23 23 23 23 23 25 25 25 25 25 25 25 25 25 25 27 29
> for (i in 1:9) {
+ donax.list[[i]]<-rep(tref,as.vector(donax.t[,i]))
+ }
> donax.list
[[1]]:
[1] 11 11 11 13 13 15 15 15 15 17 17 17 17 17 17 17 17 19 19 19 19 19
[22] 19 19 19 19 19 19 19 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21
[43] 21 21 21 21 21 21 21 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23
[64] 23 23 23 23 23 23 23 25 25 25 25 25 25 25 25 25 27 29

[[2]]:
[1] 13 15 15 15 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17

```

⁶ Moueza, M. & Chessel, D. (1976) Contribution à l'étude de la biologie de *Donax trunculus* L. dans l'algérois: analyse statistique de la dispersion le long d'une plage en baie de Bou-Ismaïl. *Journal of Experimental Marine Biology and Ecology* : 21, 211-221.

```

[22] 17 17 17 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19
[43] 19 19 19 19 19 19 19 19 19 19 21 21 21 21 21 21 21 21 21 21
[64] 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21
[85] 21 21 21 21 21 21 21 21 21 21 21 23 23 23 23 23 23 23 23 23
[106] 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23
[127] 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23
[148] 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23
[169] 23 23 23 23 23 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
[190] 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
[211] 25 25 25 25 25 25 25 25 25 25 25 27 27 27 27 27 27 27 27 27
[232] 27 27 27 27 27 27 27 29 29 31

```

```

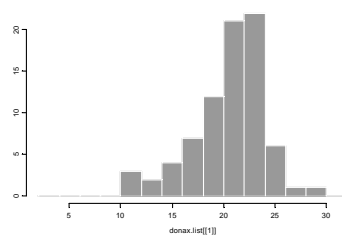
...
[[9]]:
 [1] 3 3 3 3 3 5 5 5 5 7 7 7 7 7 7 7 7 7 7 7 7
[22] 7 9 9 9 9 9 9 11 11 11 11 11 11 13 13 13 13 13 13 13 13
[43] 15 15 15 15 15 15 15 15 15 15 15 17 17 17 17 19 19 19 19 19
[64] 19 21 21 21 21 21 21 21 21 21 21 21 21 21 23 23 23 23 23 23
[85] 23 23 23 23 23 23 25 25 25 25 25 25 25 25 25 25 25 25 25 25
[106] 25 25 25 25 25 25 25 25 25 25 25 27 27 27 27 27 27 27 29 29

```

```

> break.donax<-seq(2,32,by=2)
> ?hist
> hist(donax.list[[1]],breaks=break.donax)

```

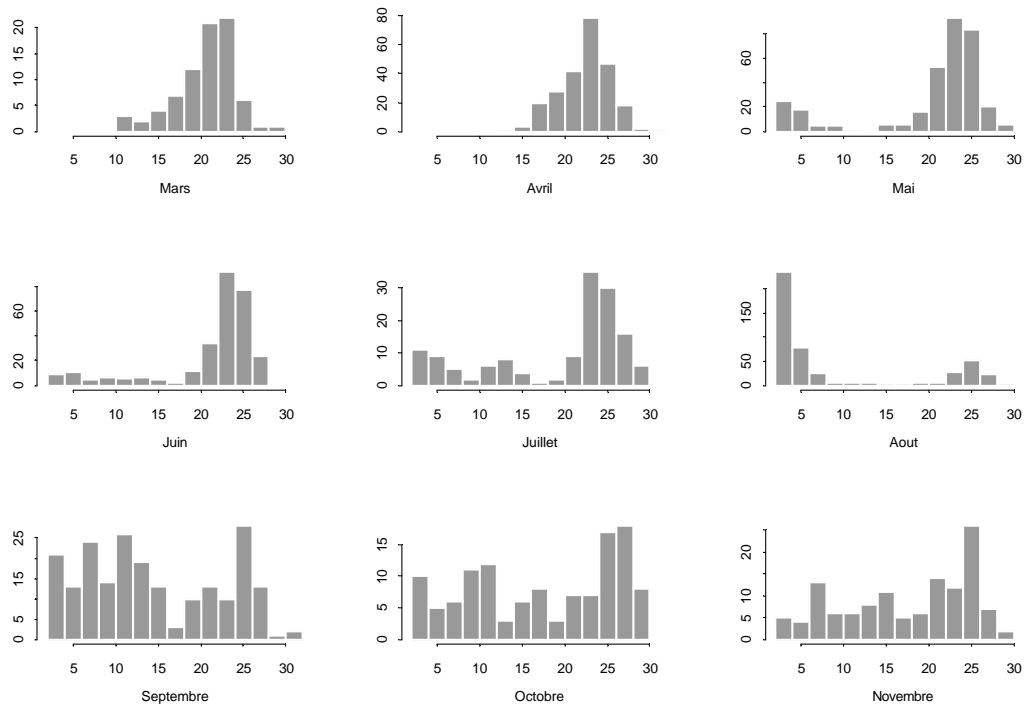


```

donax.fl<-function() {
  allpar<-par()
  par (mfrow=c(3,3))
  for (i in 1:9) {

hist(donax.list[[i]],breaks=break.donax,xlab=row.names(donax)[i])
  }
  par(allpar)
  return("ok")
}

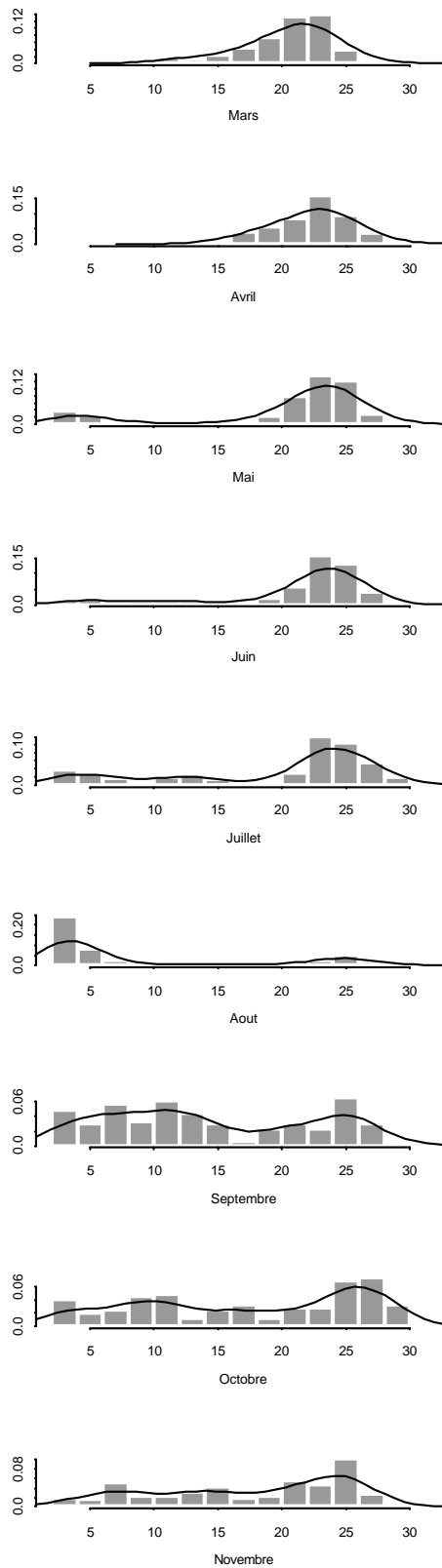
```



```

donax.fl<-function() {
  allpar<-par()
  par (mfrow=c(9,1))
  for (i in 1:9) {
    x0<-donax.list[[i]]
    hist(x0,breaks=break.donax,xlab=row.names(donax)[i],proba=T)
    lines(density(x0,width=8))
  }
  par(allpar)
  return("ok")
}

```

On se pose la question de l'estimation des mélanges de lois normales. On écrit une petite fonction pour générer des échantillons de mélanges de lois normales :

```

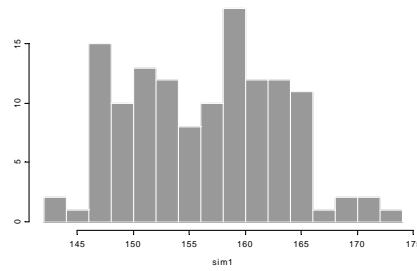
simmix<-function(n1 = 150, m1 = -1, v1 = 1, n2 = 50, m2 = 2, v2 = 4){
  x1 <- rnorm(n1, m = m1, sd = sqrt(v1))
  x2 <- rnorm(n2, m = m2, sd = sqrt(v2))
  xsim <- append(x1, x2)
  return (xsim)
}

```

```

> sim1<-simmix(n1=50,m1=150,v1=10,n2=80,m2=160,v2=20)
> hist(sim1,nclass=20)

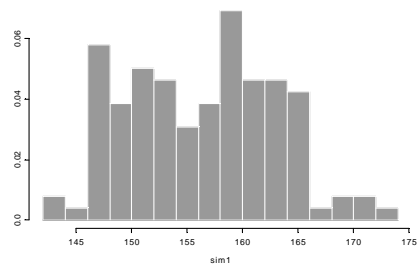
```



```

> hist(sim1,nclass=20,proba=T)

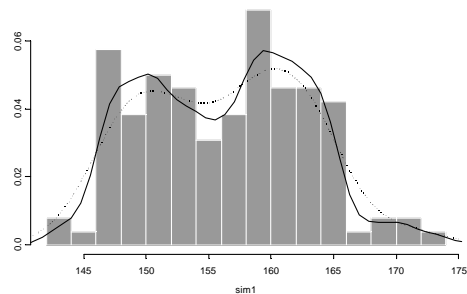
```



```

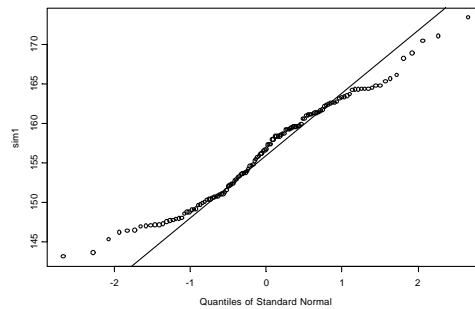
> lines(density(sim1,width=5),lty=1)
> lines(density(sim1,width=10),lty=2)

```



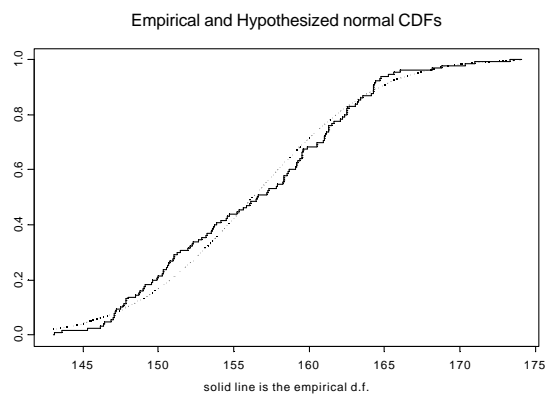
Le résultat dépend de la largeur de la fenêtre de lissage. La documentation donne les références bibliographiques incontournables sur les estimations non paramétriques de densité ⁷. Pour tracer un graphe quantile-quantile en référence à la loi normale ⁸ :

```
> qqnorm(sim1)
> qqline(sim1)
```



La déviation par rapport à la loi normale apparaît moins clairement dans le graphe de la fonction de répartition empirique de l'échantillon :

```
> cdf.compare(sim1,distri="normal",mean=mean(sim1),sd=sqrt(var(sim1)))
```



Pour tester la normalité en utilisant la fonction de répartition :

```
> ks.gof(sim1,distri="normal")
```

One sample Kolmogorov-Smirnov Test of Composite Normality

```
data: sim1
ks = 0.0809, p-value = 0.0362
alternative hypothesis:
 True cdf is not the normal distn. with estimated parameters
sample estimates:
```

⁷ Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London. Wegman, E. J. (1972). Nonparametric probability density estimation. *Technometrics*, 14, 533-546.

⁸ Chambers, J. M., Cleveland, W. S., Kleiner, B. and Tukey, P. A. (1983). *Graphical Methods for Data Analysis*. Wadsworth, Belmont, California. Hoaglin, D. C., Mosteller, F. and Tukey, J. W., editors (1983). *Understanding Robust and Exploratory Data Analysis*. Wiley, New York

mean of x standard deviation of x
156.3 6.547

Le logiciel propose une citation ⁹. Pour tester la normalité en utilisant le Khi2 :

```
> chisq.gof(sim1,distri="normal",n.param.est=2,mean=mean(sim1),
  sd=sqrt(var(sim1)))

Chi-square Goodness of Fit Test

data:  sim1
Chi-square = 21.85, df = 12, p-value = 0.0393
alternative hypothesis:
 True cdf does not equal the normal Distn. for at least one sample p
oint.
```

Le nombre de classes est optimisé automatiquement ¹⁰.

L'estimation au maximum de vraisemblance des 5 paramètres d'un mélange de deux gaussiennes n'est pas difficile. On écrit une fonction qui montre le fonctionnement de la procédure. La fonction de vraisemblance :

```
vrai <- function(z, p, m1, m2, v1, v2) {
  - log((p * exp((-0.5 * (z - m1)^2)/v1))/sqrt(v1) + ((1 - p) *
  exp((-0.5 * (z - m2)^2)/v2))/sqrt(v2))
}
```

La densité de probabilité :

```
dens <- function(z, p, m1, m2, v1, v2) {
  (p * exp((-0.5 * (z - m1)^2)/v1))/sqrt(2 * pi * v1) + ((1 - p) *
  exp((-0.5 * (z - m2)^2)/v2))/sqrt(2 * pi * v2)
}
```

La procédure d'optimisation :

```
normix<- function(n1 = 150, m1 = -1, v1 = 1, n2 = 50, m2 = 3, v2 = 4,
  trace = F)
{
  x1 <- as.data.frame(rnorm(n1, m = m1, sd = sqrt(v1)))
  x2 <- as.data.frame(rnorm(n2, m = m2, sd = sqrt(v2)))
  x3 <- rbind.data.frame(x1, x2)
  names(x3)[1] <- "z"
  minz <- min(x3$z)
  maxz <- max(x3$z)
  zz <- seq(from = minz, to = maxz, le = 100)
  p <- n1/(n1 + n2)
  n <- n1 + n2
  w0 <- vector(6, mode = "numeric")
  w0[1:6] <- c(n, p, m1, m2, v1, v2)
  names(w0) <- c("n", "p", "m1", "m2", "v1", "v2")
  parameters(x3) <- list(p = p, m1 = m1, m2 = m2, v1 = v1, v2 = v2)
  hist(x3$z, nclass = 20, proba = T)
  lines(zz, dens(zz, p, m1, m2, v1, v2), lty = 2)
  ms.fit <- ms( ~ vrai(z, p, m1, m2, v1, v2), data = x3, control = T,
  trace = trace)
```

⁹ Dallal, G. E. and Wilkinson, L. (1986). An analytic approximation to the distribution of Lilliefors's test for normality. *The American Statistician* 40, 294-296.

¹⁰ Moore, D. S. (1986). Tests of chi-squared type. In *Goodness-of-Fit Techniques*. R. B. D'Agostino and M. A. Stevens, eds. New York: Marcel Dekker.

```

w <- ms.fit$parameters
lines(zz, dens(zz, w[1], w[2], w[3], w[4], w[5]), lty = 1)
legend(locator(1), c("Estimation", "Observation"), lty = c(1, 2), cex
= 0.5)
list("simulation", w0, "estimation", w)
}
> normix()
[1] "simulation"

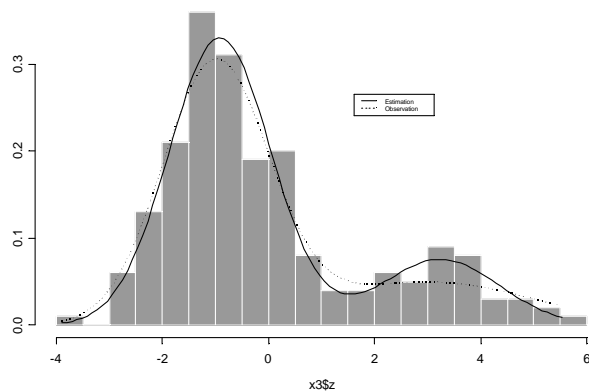
  n    p m1 m2 v1 v2
200 0.75 -1  3  1  4

[1] "estimation"

      p      m1      m2      v1      v2
0.7901 -0.9136 3.241 0.9135 1.224

```

Attention au locator



La fonction `ms`¹¹ fait l'essentiel du travail.

On peut essayer des situations variées (*à quoi sert cex ?*).

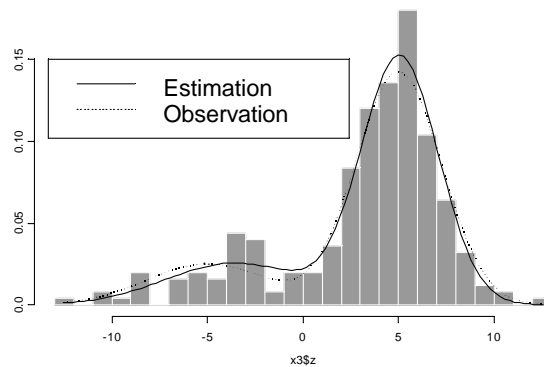
```

> normix(n1=50, n2=200, m1=-5, m2=5, v1=10, v2=5)
[1] "simulation"
  n    p m1 m2 v1 v2
250 0.2 -5  5 10  5

[1] "estimation"
      p      m1      m2      v1      v2
0.2337 -3.566 5.071 13.19 4.091

```

¹¹ Chambers, J. M., and Hastie, T. J. (eds) (1992). *Statistical Models in S*, Chapter 10, "Nonlinear Models". Pacific Grove, CA.: Wadsworth & Brooks/Cole.

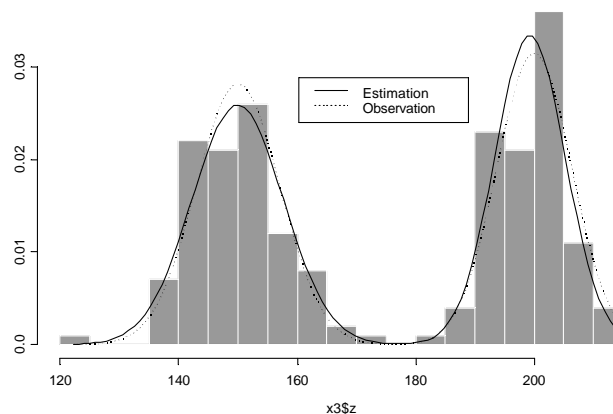


```
> normix(n1=100,n2=100,m1=150,m2=200,v1=50,v2=40)
[1] "simulation"

  n  p  m1  m2  v1  v2
200 0.5 150 200 50 40

[1] "estimation"

  p  m1  m2  v1  v2
0.5 150 199.3 59.36 35.54
```



On peut maintenant aborder le problème :

```
"estmix" <- function(z, p, m1, m2, v1, v2, minz = 0, maxx = 32, break0 =
  break.donax, trace = F)
{
  if(is.vector(z) == F) # Une précaution
    stop("Vector need")
  x3 <- as.data.frame(z) # Installation d'un pframe depuis un data.frame
  names(x3) <- "z" # Nom de la variable dans le pframe
  if(is.na(minz)) # Pour introduire à la main des valeurs de départ
    minz <- min(z)
  if(is.na(maxx))
    maxx <- max(z)
  n <- length(z)
  zz <- seq(from = minz, to = maxx, le = 100)
  if(is.na(p))
    p <- 0.5
  if(is.na(m1))
    m1 <- mean(z[z <= quantile(z, p)])
  if(is.na(m2))
    m2 <- mean(z[z >= quantile(z, p)])
  if(is.na(v1))
```

```

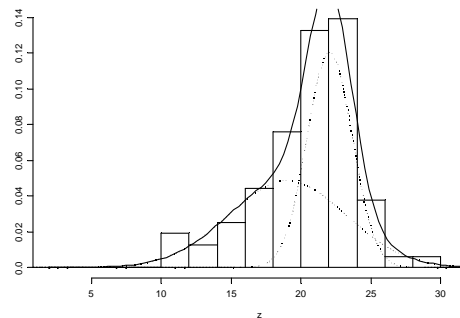
    v1 <- var(z[z <= quantile(z, p)])
  if(is.na(v2))
    v2 <- var(z[z >= quantile(z, p)])
  if(is.na(break0[1]))
    break0 <- quantile(z, probs = seq(0, 1, by = 0.05))
  w0 <- vector(6, mode = "numeric")
  w0[1:6] <- c(n, p, m1, m2, v1, v2)
  names(w0) <- c("n", "p", "m1", "m2", "v1", "v2")
  parameters(x3) <- list(p = p, m1 = m1, m2 = m2, v1 = v1, v2
    = v2) # Paramétrage du pframe
  hist(z, breaks = break0, proba = T,density=-1) # Tracer de
l'histogramme
  ms.fit <- ms( ~ vrai(z, p, m1, m2, v1, v2), data = x3,
    control = T, trace = trace) # Estimation des paramètres
  w <- ms.fit$parameters # Récupération des valeurs
  lines(z, dens(z, w[1], w[2], w[3], w[4], w[5]), lty = 1)
  lines(z, w[1] * dnorm(z, mean = w[2], sd = sqrt(w[4])),
    lty = 2) # Tracer des composantes du modèle
  lines(z, (1 - w[1]) * dnorm(z, mean = w[3], sd = sqrt(w[5]
  )), lty = 2)
  w # Retour des valeurs des estimations
}

```

```

> estmix(donax.list[[1]],NA,NA,NA,NA,NA)
p    m1    m2    v1    v2
0.5118 19.02 22.04 17.68 2.6

```

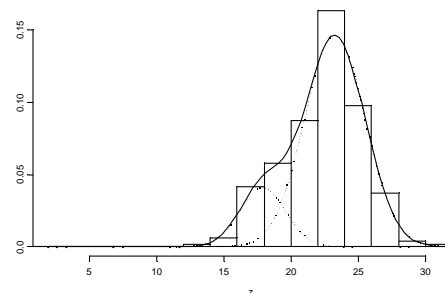


Mars

```

> estmix(donax.list[[2]],NA,NA,NA,NA,NA)
p    m1    m2    v1    v2
0.1748 17.88 23.26 2.886 5.041

```

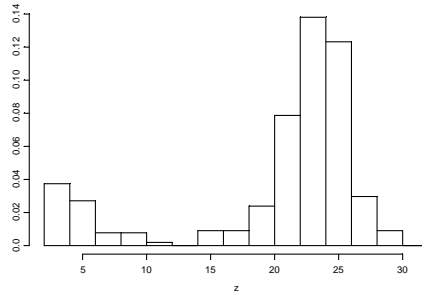


Avril

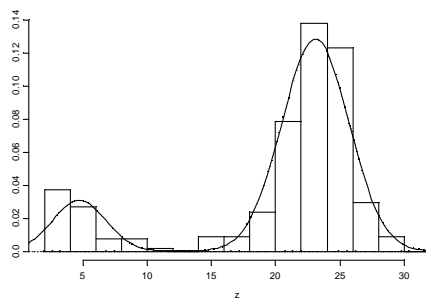
```

> estmix(donax.list[[3]],NA,NA,NA,NA,NA)
Error in ms( ~ vrai(z, p, m1, m2, v1, v2), d..: 337 NAs found in value
Dumped
Premier échec

```

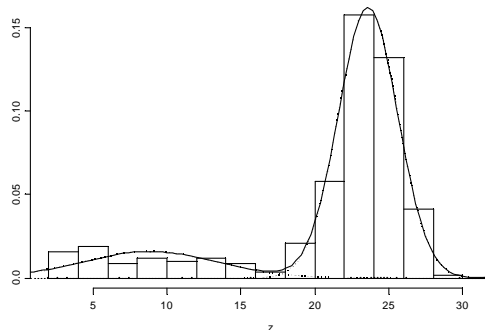


Premier éssai à vue :
 > estmix(donax.list[[3]],0.15,3,23,5,15)
 p m1 m2 v1 v2
 0.1602 4.74 23.1 4.299 6.835
 Gagné :



> estmix(donax.list[[4]],NA,NA,NA,NA,NA)
 p m1 m2 v1 v2
 0.1821 8.84 23.6 20.99 4.035

Mai



Juin

...

7 - Séries temporelles

Implanter les données de la chronique proposée dans le texte de référence Tomassone R., Charles-Bajard S. & Bellanger L. (1998) Introduction à la planification expérimentale, DEA « Analyse et modélisation des systèmes biologiques ». Sous Windows, utiliser la fonction Import du menu File. Sous UNIX, utiliser la fiche de J. Thioulouse.

```
> Class1
   V1  V2  V3  V4  V5  V6  V7  V8  V9  V10 V11 V12
1 0.66 0.70 0.74 0.63 0.70 0.66 0.61 0.52 0.60 0.61 0.70 1.10
```



```

2 1.17 1.23 0.85 0.78 0.71 0.55 0.56 0.74 0.80 0.75 0.74 0.79
3 0.78 1.00 1.05 1.09 1.05 0.75 0.73 0.77 0.77 0.84 0.66 0.68
4 0.67 0.56 0.62 0.73 0.70 0.74 0.93 1.00 1.50 1.30 1.18 1.15
5 1.34 1.37 1.13 1.04 0.92 1.15 0.99 1.32 1.46 1.24 1.01 1.04
6 1.08 0.94 0.81 1.00 0.98 1.02 1.16 0.96 1.23 1.10 1.02 1.08
7 1.30 0.97 0.96 0.80 0.62 0.51 0.56 0.84 0.87 0.87 0.76 0.86
8 0.81 0.77 0.74 0.80 0.78 0.72 0.66 0.92 0.99 0.98 0.70 0.65
9 0.78 0.57 0.41 0.61 0.85 0.85 1.11 1.05 0.96 1.31 1.49 1.35
10 1.32 1.24 1.47 1.32 1.23 1.33 1.48 1.49 1.48 1.49 1.55 1.73
11 1.70 1.43 1.44 1.37 1.20 1.19 1.39 1.41 1.40 1.39 1.62 1.59
12 1.36 1.31 0.99 0.89 0.87 0.94 1.03 1.27 1.20 1.10 0.93 1.00
13 1.04 1.10 1.10 1.09 1.05 0.70 0.88 0.81 1.08 1.39 1.16 0.49

```

t

```

> t(Class1)
      1      2      3      4      5      6      7      8      9     10     11     12     13
V1 0.66 1.17 0.78 0.67 1.34 1.08 1.30 0.81 0.78 1.32 1.70 1.36 1.04
V2 0.70 1.23 1.00 0.56 1.37 0.94 0.97 0.77 0.57 1.24 1.43 1.31 1.10
V3 0.74 0.85 1.05 0.62 1.13 0.81 0.96 0.74 0.41 1.47 1.44 0.99 1.10
V4 0.63 0.78 1.09 0.73 1.04 1.00 0.80 0.80 0.61 1.32 1.37 0.89 1.09
V5 0.70 0.71 1.05 0.70 0.92 0.98 0.62 0.78 0.85 1.23 1.20 0.87 1.05
V6 0.66 0.55 0.75 0.74 1.15 1.02 0.51 0.72 0.85 1.33 1.19 0.94 0.70
V7 0.61 0.56 0.73 0.93 0.99 1.16 0.56 0.66 1.11 1.48 1.39 1.03 0.88
V8 0.52 0.74 0.77 1.00 1.32 0.96 0.84 0.92 1.05 1.49 1.41 1.27 0.81
V9 0.60 0.80 0.77 1.50 1.46 1.23 0.87 0.99 0.96 1.48 1.40 1.20 1.08
V10 0.61 0.75 0.84 1.30 1.24 1.10 0.87 0.98 1.31 1.49 1.39 1.10 1.39
V11 0.70 0.74 0.66 1.18 1.01 1.02 0.76 0.70 1.49 1.55 1.62 0.93 1.16
V12 1.10 0.79 0.68 1.15 1.04 1.08 0.86 0.65 1.35 1.73 1.59 1.00 0.49

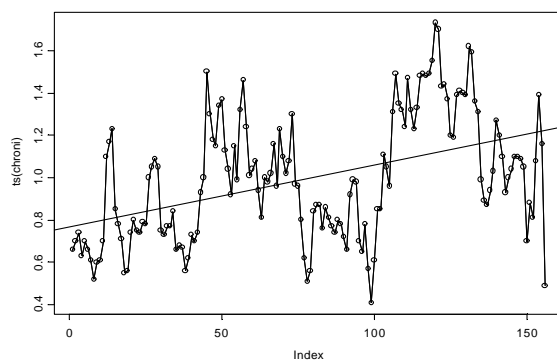
```

Transposer le tableau :

```

> chroni<-t(Class1)
> matrix(1:10,nrow=2)
      [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
> matrix(1:10,ncol=2)
      [,1] [,2]
[1,]    1    6
[2,]    2    7
[3,]    3    8
[4,]    4    9
[5,]    5   10
> matrix(1:10,ncol=2,byrow=T)
      [,1] [,2]
[1,]    1    2
[2,]    3    4
[3,]    5    6
[4,]    7    8
[5,]    9   10
> chroni<-matrix(chroni,ncol=1)
> chroni[1:10]
[1] 0.66 0.70 0.74 0.63 0.70 0.66 0.61 0.52 0.60 0.61
> is.vector(chroni)
[1] F
> chroni<-chroni[,1]
> is.vector(chroni)
[1] T
> lines(chroni)
> x<-1:156
> abline(lm(chroni~x))

```



```
> lm(chroni~x)
Call:
lm(formula = chroni ~ x)

Coefficients:
(Intercept)          x
    0.7666    0.002922

Degrees of freedom: 156 total; 154 residual
Residual standard error: 0.2635
> tsup<-c(157,158,159)
> 0.767+0.00292*tsup
[1] 1.23 1.23 1.23
> 0.767+0.00292*tsup
[1] 1.225 1.228 1.231
```

ar

```
> ?ar
DESCRIPTION

Fits a model of the form  $x(t) = a(1)x(t-1) + \dots + a(p)x(t-p) + e(t)$ .

USAGE

ar(x, aic=T, order.max=<<see below>>, method="yule-walker")

REQUIRED ARGUMENTS

x a univariate or multivariate time series, or a vector, or a matrix
with columns representing univariate components of a multivariate time
series. Missing values are allowed only at the beginning or end of
series.
```

arima

```
> ?arima.lme
DESCRIPTION

Returns a list representing a univariate ARIMA model estimated by
Gaussian maximum likelihood.

USAGE

arima.mle(x, model, n.cond=<<see below>>, xreg=NULL, ...)
...

WARNING
```

Unlike the `ar` function, the mean of the series will not be estimated by `arima.mle` unless you use the `xreg` argument. `arima.mle` assumes a zero mean series.

```
> ar1<-arima.mle(chroni,list(order=c(1,0,0)),xreg=1)
> ar1
$model:
$model$order:
[1] 1 0 0

$model$ar:
[1] 0.85

$model$ndiff:
[1] 0

$var.coef:
      ar(1)
ar(1) 0.00179

$method:
[1] "Maximum Likelihood"

$series:
[1] "chroni"

$aic:
[1] -131.4

$loglik:
[1] -135.4

$sigma2:
[1] 0.02445

$n.used:
[1] 155

$n.cond:
[1] 1

$converged:
[1] T

$conv.type:
[1] "relative function convergence"

$reg.coef:
[1] 0.9918

$reg.series:
[1] "intercept"
```

arima.forecast

DESCRIPTION

Forecasts a univariate time series using an ARIMA model. Under the assumption that the model is known, predicted values and their standard errors are computed for future values.

USAGE

```
arima.forecast(x, model, n, end, sigma2=<<see below>>, xreg=NULL,
              reg.coef=NULL)
```

```
> arima.forecast(t, model=ar1$model, n=3, xreg=rep(1,159),
```

```
reg.coef=ar1$reg.coef)
$mean:
[1] 0.5653 0.6293 0.6836

$std.err:
[1] 0.1564 0.2052 0.2342
```

Pour en savoir plus

<http://www.mathsoft.com/splus/doc/index.htm>

The following zipped documents are in pdf format.

S-PLUS Programmers Guide, Version 4.5:

In this book, we concentrate on describing how to use the language. As with any good book on programming, the goal of this book is to help you quickly produce useful S-PLUS functions, and then step back and delve more deeply into the internals of the S-PLUS language. Along the way, we will continually touch on those aspects of S-PLUS programming that are either particularly effective (such as vectorized arithmetic) or particularly troubling (memory use, for loops). This chapter aims to familiarize you with the language, starting with a comparison of interpreted and compiled languages. We then briefly describe object-oriented programming as it relates to S-PLUS, although a full discussion is deferred until chapter 10, Object-Oriented Programming in S-Plus. We then describe the basic syntax and data types in S-PLUS. Programming in S-PLUS does not require, but greatly benefits from, programming tools such as text editors and source control. We touch on these tools briefly in section The S-Plus Programming Environment (page 32). Finally we introduce the various graphics paradigms, and discuss when each should be used.

S-PLUS 4 Guide to Statistics:

Welcome to the S-PLUS Guide to Statistics. This book is designed as a reference tool for S-PLUS users wanting to use the powerful statistical techniques in S-PLUS. The Guide to Statistics covers a wide range of statistical and mathematical modeling; no one user is likely to tap all of these resources since advanced topics such as survival analysis and time series are complete fields of study in themselves. All examples in this guide are run using input through the Commands window—the traditional method of accessing the power of S-PLUS. Many of the functions can also be run through the Statistics menu and dialogs available in the graphical user interface. We hope you will find this book a valuable aid for exploring both the theory and practice of statistical modeling.

S-PLUS Users Guide, Version 4.5

S-PLUS Documentation Supplement, Version 4.5

Pour ceux qui se lève tôt ...