TP 5 : AFD et Classement

Quelques ouvrages :

Comment interpréter une AFD. Tomassone 1988, ITCF, 1988
Discrimination et classement. Tomassone et al. Masson, 1988

Quelques sites :

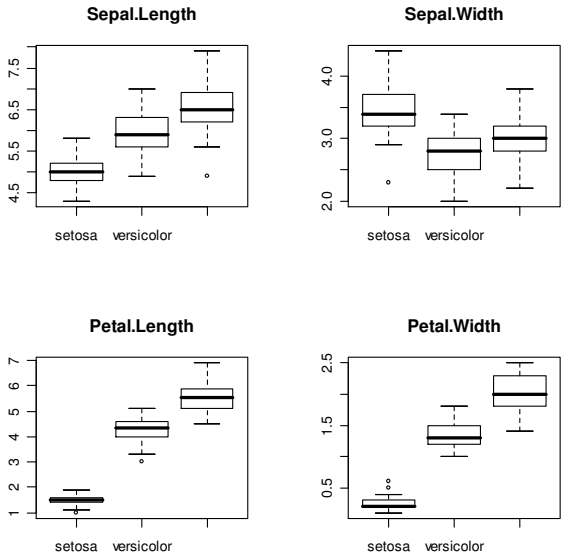
http://biom3.univ-lyon1.fr/R/fichestd/tdr63.pdf (dont est tirée l'introduction)
http://cedric.cnam.fr/~saporta/discriminante.pdf
http://www.lsp.ups-tlse.fr/Besse/enseignement.html
http://www.lsp.ups-tlse.fr/Fp/Bigot/ens_fr.html

I Introduction

1/ Etude de l'exemple historique iris data(iris)

Une dimension

Utiliser la fonction boxplot pour étudier les différences entre groupes et effectuer une anova pour chaque variable.

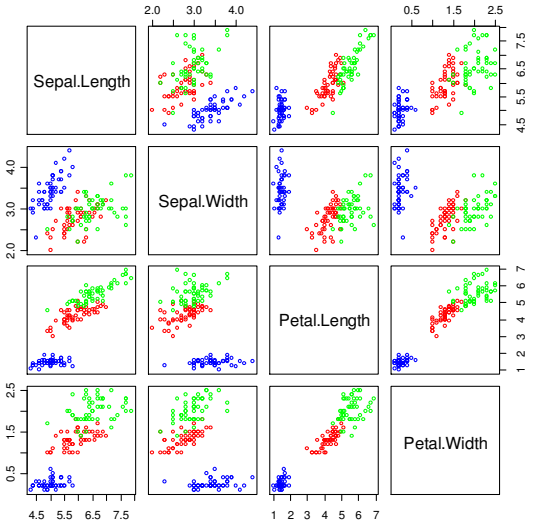



```
library(MASS);data(iris);attach(iris)
anova(lm(Sepal.Length~Species))

[1] "Sepal.Length"
             Df Sum Sq Mean Sq F value    Pr(>F)
Species       2 63.212  31.606  119.26 < 2.2e-16
Residuals   147 38.956   0.265

[1] "Sepal.Width"
             Df  Sum Sq Mean Sq F value    Pr(>F)
Species       2 11.3449  5.6725   49.16 < 2.2e-16
Residuals   147 16.9620  0.1154

[1] "Petal.Length"
             Df Sum Sq Mean Sq F value    Pr(>F)
Species       2 437.10  218.55  1180.2 < 2.2e-16
Residuals   147  27.22    0.19

[1] "Petal.Width"
             Df Sum Sq Mean Sq F value    Pr(>F)
Species       2 80.413  40.207     960 < 2.2e-16
Residuals   147  6.157   0.042
```

Deux dimensions

Examiner la discrimination des groupes en 2 dimensions en utilisant des nuages de points.

```
>plot(iris[,1],iris[,2],col=c("blue","red"
,"green")[iris$Species])
```



Trois dimensions

Etudier les différentes possibilités de représentation des nuages dans l'espace

```
> library(rgl)
>
plot3d(iris[,1],iris[,2],iris[,3],col=c("blue",
"red","green")[iris$Species],type="s")
```

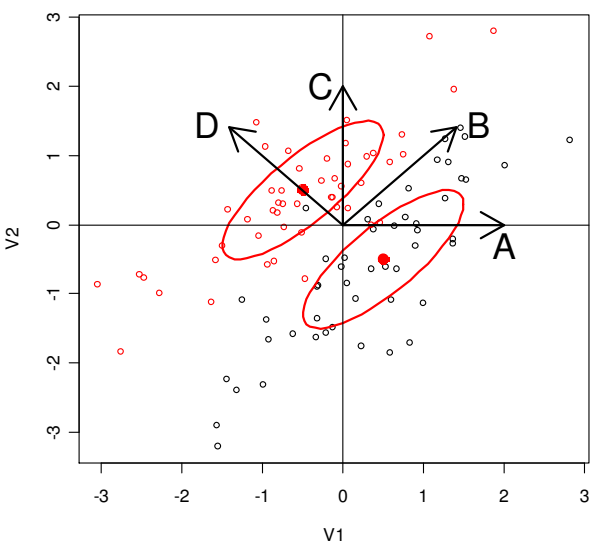
2/ Vers l'afd

On simule deux lois normales à deux dimensions et on projette les points sur les axes (OA), (OB), (OC), (OD).

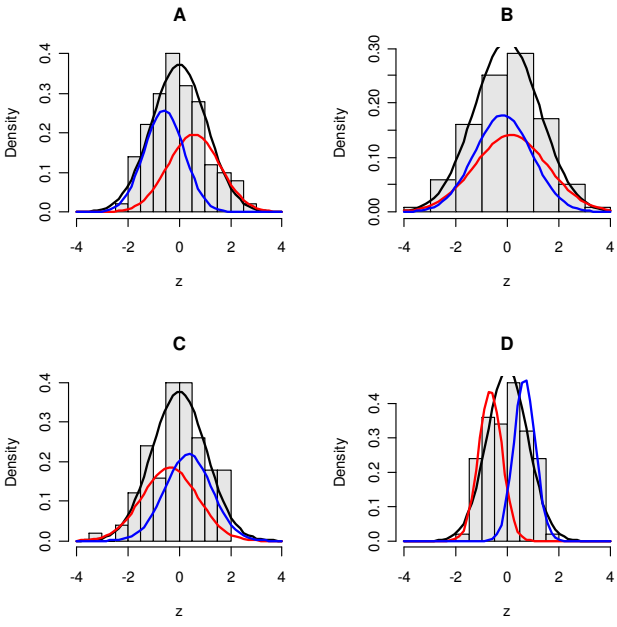
Quelle direction permet la meilleure discrimination ?

Quel critère permet de déterminer le meilleur axe de discrimination possible ?

```
library(MASS)
s <- matrix(c(1, 0.8, 0.8, 1), 2)
x1 <- mvrnorm(50, c(0.5, -0.5), s)
x2 <- mvrnorm(50, c(-0.5, 0.5), s)
x <- rbind.data.frame(x1, x2)
x <- scale(x, scale = F)
fac <- factor(rep(1:2, rep(50, 2)))
plot(x, col = as.integer(fac))
library(car)
ellipse(c(0.5, -0.5), s,1)
ellipse(c(-0.5, 0.5), s,1)
abline(h=0);abline(v=0)
arrows(0, 0, 2, 0, lwd = 2)
text(2, 0, "A", pos = 1, cex = 2)
arrows(0, 0, sqrt(2), sqrt(2), lwd = 2)
text(sqrt(2), sqrt(2), "B", pos = 4, cex =
2)
arrows(0, 0, 0, 2, lwd = 2)
text(0, 2, "C", pos = 2, cex = 2)
arrows(0, 0, -sqrt(2), sqrt(2), lwd = 2)
text(-sqrt(2), sqrt(2), "D", pos = 2, cex
= 2)
```




```
par(mfrow = c(2, 2))
f1 <- function(a, b, cha) {
z <- a * x[, 1] + b * x[, 2]
z0 <- seq(-4, 4, le = 50)
z1 <- z[fac == 1]
z2 <- z[fac == 2]
hist(z, proba = TRUE, col = grey(0.9),
xlim = c(-4, 4), main = cha)
lines(z0, dnorm(z0, mean(z), sd(z)), lwd =
2)
lines(z0, 0.5 * dnorm(z0, mean(z1),
sd(z1)), col = "red", lwd = 2)
lines(z0, 0.5 * dnorm(z0, mean(z2),
sd(z2)), col = "blue", lwd = 2)
}
f1(1, 0, "A")
f1(1/sqrt(2), 1/sqrt(2), "B")
f1(0, 1, "C")
f1(-1/sqrt(2), 1/sqrt(2), "D")
```



A quoi peut servir la fonction suivante ?

```
f2 <- function(a,b) {
 z=a*x[,1]+b*x[,2]
 a1 = var(z) * 99/100
 a2=var(predict(lm(as.numeric(z)~fac)))*99/100
 a3=a2/a1
 round(c(a1,a2,a3),2)
}
```

II Un exemple simple : calcul manuel.

Soit le tableau suivant décrivant deux populations décrites par deux variables quantitatives .

	pop 1					pop 2			
x_1	0	2	2	4		5	7	7	9
x_2	3	5	7	9		0	2	4	6

1/ Notations

Déterminer les coefficients suivants

Effectif total des individus: n=
Nombre de classes : q=
Effectif des individus de la population I : n_1 =
Effectif des individus de la population I : n_2 =
Nombre de variables : p =

2/ Nuage des individus

a. Centrer les données et calculer les moyennes des différentes variables classe par classe.

$x_1^{(1)}$ = $x_2^{(1)}$ = $x_1^{(2)}$ = $x_2^{(2)}$ =

b. En déduire les centres de gravités des nuages de chaque classe.

c. Construire ces nuages et placer les centres de gravité.

3/ Recherche de la fonction discriminante

a. Calculer les matrices des sommes des carrés intra-groupes W_1, W_2 et W, et inter-groupes B.

b. Calculer W^{-1}

c. Déterminer les valeurs propres et vecteurs propres normés de $W^{-1}B$.

d. Comment retrouver directement ce résultat?

e. En déduire la fonction discriminante.

4/ Projection des individus et classement.

a. Calculer les coordonnées des individus et des centres de gravité sur l'axe discriminant.

b. Placer ces projections sur l'axe.

c. En déduire la distance des individus aux centres et leur classement.

5/ Calculs avec R

a. Calculer la matrice des données centrées X_c et en déduire la matrice de variance totale $T= {}^tX_c X_c$.

b. Calculer la matrice des centres de gravité $g\begin{pmatrix} g_1 \\ g_2 \end{pmatrix}$ et en déduire la matrice de variance interclasses B

c. Calculer la matrice de variance intra-classe avec la matrice des variables centrées par groupe X_g $W={}^tX_g X_g$. Vérifier l'égalité T= B + W.

d. Calculer la matrice $W^{-1}B$ et en déduire les vecteurs propres W* normés.

e. En déduire la fonction discriminante et les projections des individus F1 sur l'axe.

f. Afficher les résultats. Effectuer une anova sur F1 : anova(lm(F1~as.factor(c(rep(1,5),rep(2,5)))

g. Utiliser la fonction lda de la librairie MASS et comparer les résultats.

II Exemple avec 3 classes et 3 variables

On suppose que l'on a observé 3 variables quantitatives notées Y1, Y2, Y3 et une variable quantitative T à K = 3 modalités sur un échantillon de 7 individus. Les résultats obtenus sont les suivants :

$individus$(i)	y_i^1	y_i^1	y_i^3	t_i
1	7	26	84	1
2	9	28	84	1
3	8	27	81	2
4	10	23	81	2
5	11	25	80	3
6	12	24	79	3
7	13	29	78	3

On souhaite réaliser l'AFD de ce jeu de données où tous les poids des individus sont pris égaux.

1. A l'aide de R, calculer :
– la matrice des données centrées, et en déduire la matrice de variance totale T* ,
– la matrice des barycentres des classes centrées, et en déduire la matrice de variance inter-classes B*,
– calculer la matrice de variance intra-classes W* et vérifier que T = B + W,
– diagonaliser la matrice W^{*-1} B*,
– déterminer la part d'inertie expliquée par les deux premières valeurs propres. Que pouvez en déduire sur l'importance du troisième axe discriminant ?
– normaliser les vecteurs propres pour qu'ils soient W*-orthonormés
– calculer les projections des individus sur les deux premiers axes discriminants et commenter graphiquement le résultat
– calculer les corrélations entre les axes discriminants et les variables initiales puis réprésenter graphiquement le cercle des corrélations. Comment interpréter les deux premiers axes discriminants en fonction des variables initales ?

2. L'AFD peut s'implémenter sous R à l'aide de la fonction lda() de la librairie MASS. Comparer les résultats donnés par la fonction lda() avec ceux de la question précédente.

3. Effectuer l'ACP de ce jeu de données. Comparer les résultats avec l'AFD.

IV Etude de deux exemples concrets : chazeb et iris

Les exemples sont tirés des ouvrages de Tomassone (Comment interpréter …, ITCF, 1988) et Tomassone et al. (Discrimination et classement, Masson, 1988).

Le fichier chazeb est constitué de deux populations (charolais :cha et zebu :zeb) sur lesquelles sont mesurées 6 poids en kg: vif, carcasse, première qualité, viande totale, gras, os.
Le fichier pommier décrit trois variables quantitatives, nombre d'inflorescence, longueur et diamètre des rameaux mesurées chacune à un et deux ans sur quatre variétés.

Analyse des résultats
1. Etude des variables

- **a.** Décrire à l'aide de paramètres statistiques et de graphique les variables.
- **b.** Parmi les variables initiales, quelle est celle qui discrimine le mieux les populations ? Préciser la méthode utilisée.
- **c.** Tester globalement l'influence de la variété sur l'ensemble des variables.

2. Analyse discriminante

a. Réaliser l'analyse discriminante.

b. Comment évaluer l'intérêt de l'AFD?

c. Montrer que les populations sont significativement différentes. Quelle est l'hypothèse testée et les hypothèses nécessaires au test?

d. Déterminer le nombre d'axes discriminants significatifs.

e. Déterminer la meilleure fonction discriminante.

f. Calculer la projection des centres de gravité sur cet axe avec lda et discrimin.

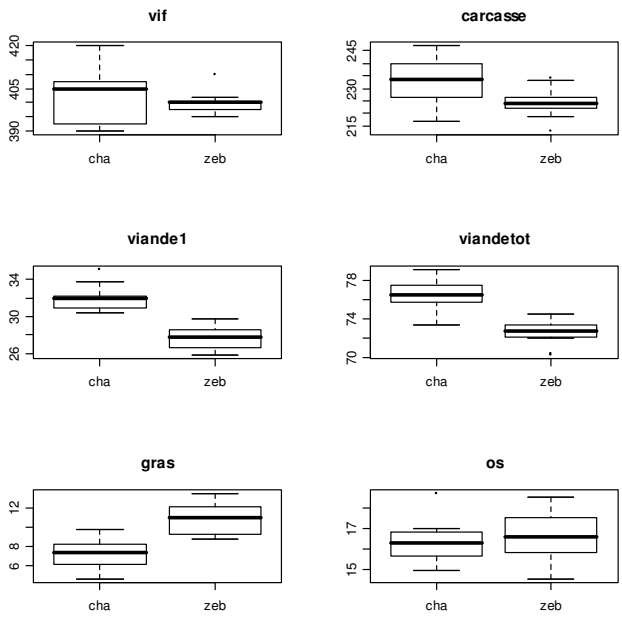
h. Afficher les projections.

i. Etudier la qualité du classement.

Exemple chazeb :

1.

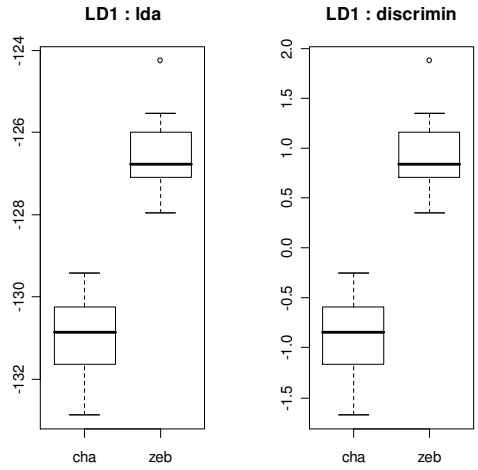
```
library(ade4)
data(chazeb);chazeb
tab=chazeb$tab
cla=chazeb$cla
par(mfrow=c(3,2))
for (k in 1:6)
boxplot(tab[,k]~cla,main=names(tab)[k])
tapply(tab[,1],cla,mean)
lda(cla~.,tab)$means
> tapply(tab[,1],cla,mean)
     cha      zeb
402.5000 399.7273
> round(lda(cla~.,tab)$means,1)
      vif carcasse viande1 viandetot gras   os
cha 402.5    233.0    32.0      76.6  7.3 16.3
zeb 399.7    224.3    27.7      72.6 10.8 16.5
for (k in 1:6)  {
  print(names(tab)[k])
  print(anova(lm(tab[,k]~cla))) }

 "viande1"
          Df  Sum Sq Mean Sq F value    Pr(>F)
cla        1 107.505 107.505  59.513 1.467e-07
Residuals 21  37.935   1.806
```




```
> summary(manova(as.matrix(tab)~cla),test="Wilks")
          Df   Wilks approx F num Df den Df    Pr(>F)
cla        1  0.1549  14.5468      6     16 1.110e-05
Residuals 21
```

2. Fonction lda library(MASS)

```
>afd1=lda(cla~.,tab)

>round(afd$scaling,2)  (coefficients des
variables)
      vif carcasse viande1 viandetot  gras
os
LD1 -0.07    -0.04   -0.72     -0.79 -0.62
-0.49
```



Signification des axes

```
> anova(lm(as.matrix(tab)%*%afd1$scaling~cla))
          Df Sum Sq Mean Sq F value
Pr(>F)
cla        1 114.56  114.56  114.56 5.819e-10
Residuals 21  21.00    1.00

> round(afd1$svd^2,2) (F*)
[1] 114.56
```

Fonction discrimin library(ade4)

```
>library(ade4); afd2=discrimin(dudi.pca(tab,scan=FALSE),cla,scan=FALSE) (avec ade4)
> anova(lm(as.matrix(afd2$li)~cla)) (on retrouve le F*)

> afd2$eig;1-afd2$eig;afd2$eig/(1-afd2$eig)
       [1] 0.845083  (valeur propre de discrimin)
       [1] 0.1549170 (on retrouve le critère de Wilks)
       [1] 5.455068  (On retrouve la valeur propre de lda)
```

```
> afd2$fa(coefficients)
                  DS1
vif       -0.2012722
carcasse  -0.1319587
viande1   -0.7467706
viandetot -0.7878439
gras      -0.6038732
os        -0.2212261

> afd2$va (corrélation)
                  CS1
vif       -0.2006132
carcasse  -0.5600174
viande1   -0.9352399
viandetot -0.9048107
gras       0.8276005
os         0.1164899

> afd2$cp (corrélation)
             DS1
RS1  0.94954572
RS2 -0.20780580
RS3  0.07505468
RS4 -0.17661015
RS5  0.07905467
RS6 -0.11002585

> afd2$gc
             DS1
cha -0.8801474
zeb  0.9601608
```

```
> plot(as.matrix(afd2$li),as.matrix(as.matrix(tab)%*%afd1$scaling))
```

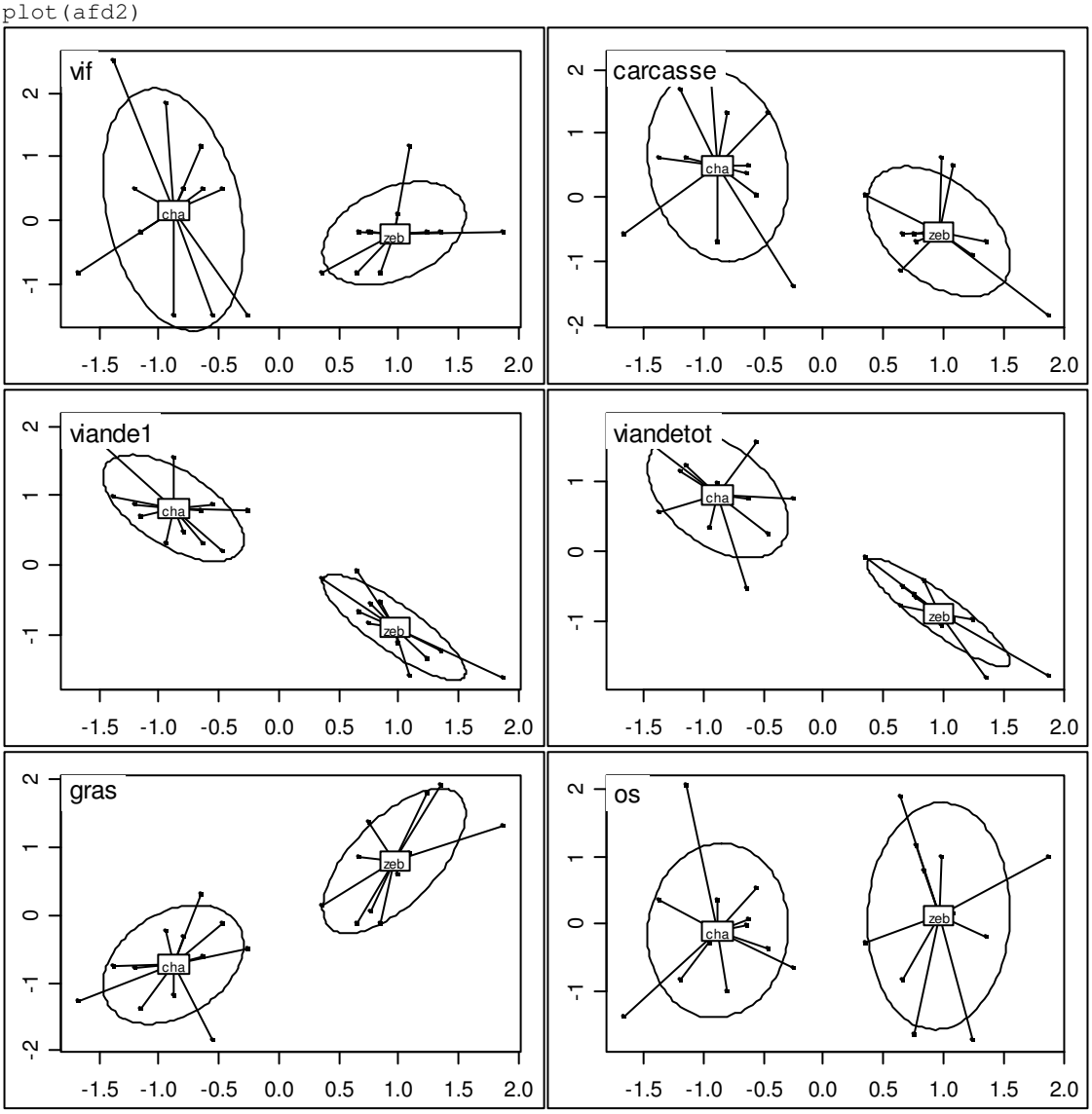
Signification des axes (bis) (Statistique de wilks corrigé par Bartlett)

```
        wilks = function(afd,n,p,g,k) {
        eig=afd$eig
        if (k>=1) eig =afd$eig[-(1:k)]
        Wilks=prod(1-eig)
        khi2=-(n-0.5*(p+g)-1)*log(Wilks)
        ddl= (p-k)*(g-k-1)
        prob=1-pchisq(khi2,ddl)
        print(round(c(Wilks,khi2,ddl,prob),2))}

> wilks(afd2,23,6,2,0)
[1] 1.549170e-01 3.356758e+01 6.000000e+00 8.152345e-06
```

Classement

```
> predict(afd1,tab)$class
cha cha cha cha cha cha cha cha cha cha cha zeb zeb zeb zeb zeb zeb zeb zeb zeb zeb zeb
> table(cla,predict(afd1,tab)$class)
cla   cha zeb
  cha  12   0
  zeb   0  11
```

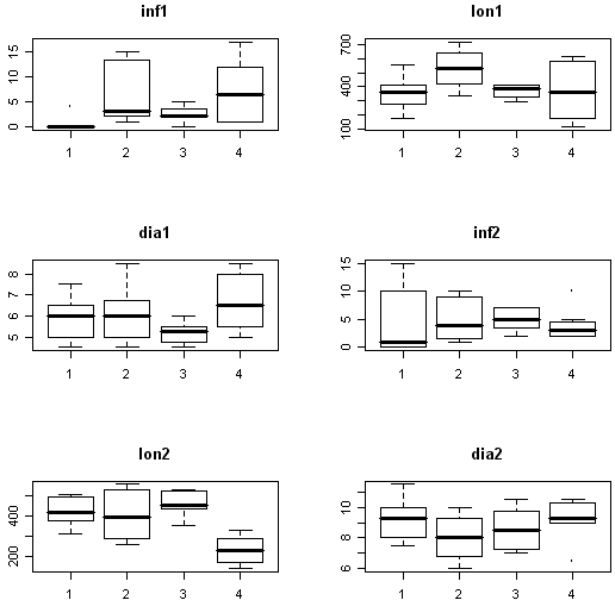
```
plot(afd2)
```



Exemple pommier :

Moyennes

```
  inf1   lon1 dia1 inf2   lon2 dia2
1 0.50 352.50 5.88 4.62 423.12 9.19
2 6.62 533.75 6.06 5.00 403.75 8.00
3 2.50 370.62 5.19 5.00 461.25 8.56
4 7.12 370.62 6.69 3.88 228.75 9.25
```



Anova

```
[1] "lon2"
          Df Sum Sq Mean Sq F value    Pr(>F)
cla        3 255196   85065  11.669 3.882e-05
***
Residuals 28 204109    7290
```

Manova

```
             Df  Wilks approx F num Df den Df
Pr(>F)
cla        3.000 0.0436   7.3768 18.000 65.539
7.243e-10 ***
Residuals 28.000
```

```
> afd2$eig;          1-afd2$eig;          afd2$eig/(1-afd2$eig);     prod(1-afd2$eig)
[1] 0.88 0.49 0.29   [1] 0.12 0.51 0.71   [1] 7.28 0.96 0.41          [1] 0.04364072
```

coefficients (lda)

	LD1	LD2	LD3
inf1	0.10	-0.10	-0.33
lon1	-0.02	-0.01	0.00
dia1	1.03	0.16	2.74
inf2	-0.03	-0.10	0.14
lon2	-0.01	0.01	0.00
dia2	0.94	0.44	-1.23

coefficient (discrimin)

	DS1	DS2
inf1	0.19	-0.40
lon1	-0.96	-0.64
dia1	0.45	0.15
inf2	-0.04	-0.28
lon2	-0.53	0.58
dia2	0.48	0.46

corrélation variables

	CS1	CS2
inf1	0.12	-0.74
lon1	-0.31	-0.51
dia1	0.33	-0.39
inf2	-0.12	0.03
lon2	-0.68	0.56
dia2	0.34	0.25

corrélation acp

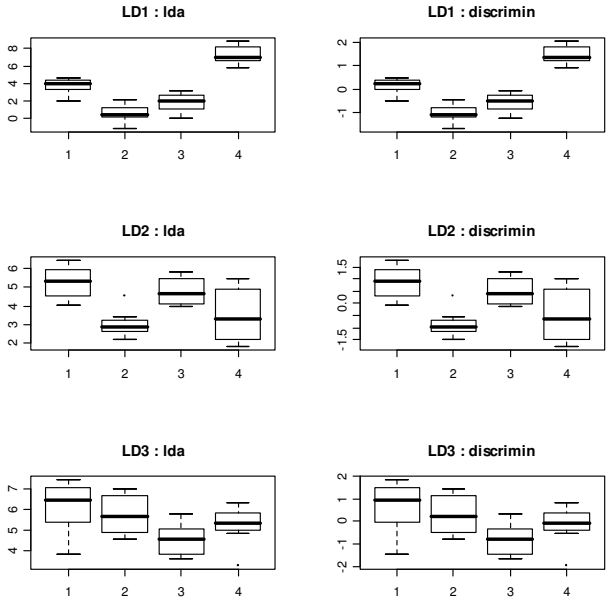
	DS1	DS2
RS1	-0.17	0.46
RS2	0.40	-0.44
RS3	0.11	-0.67
RS4	-0.87	-0.33
RS5	-0.20	-0.16
RS6	-0.07	-0.08

Signification des axes

F*

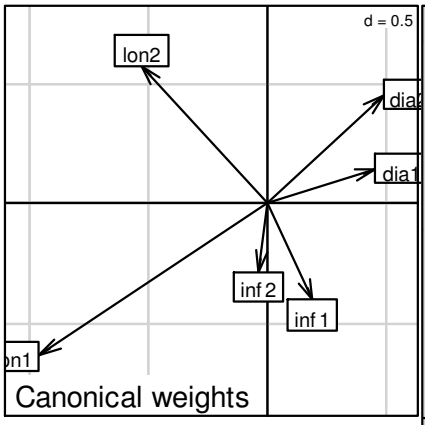
```
round(afd1$svd^2,2)
[1] 67.92  9.01  3.82
```

```
> wilks(afd2,32,6,4,0)
[1]  0.04 81.43 18.00  0.00
> wilks(afd2,32,6,4,1)
[1]  0.36 26.48 10.00  0.00
> wilks(afd2,32,6,4,2)
[1] 0.71 8.91 4.00 0.06
```

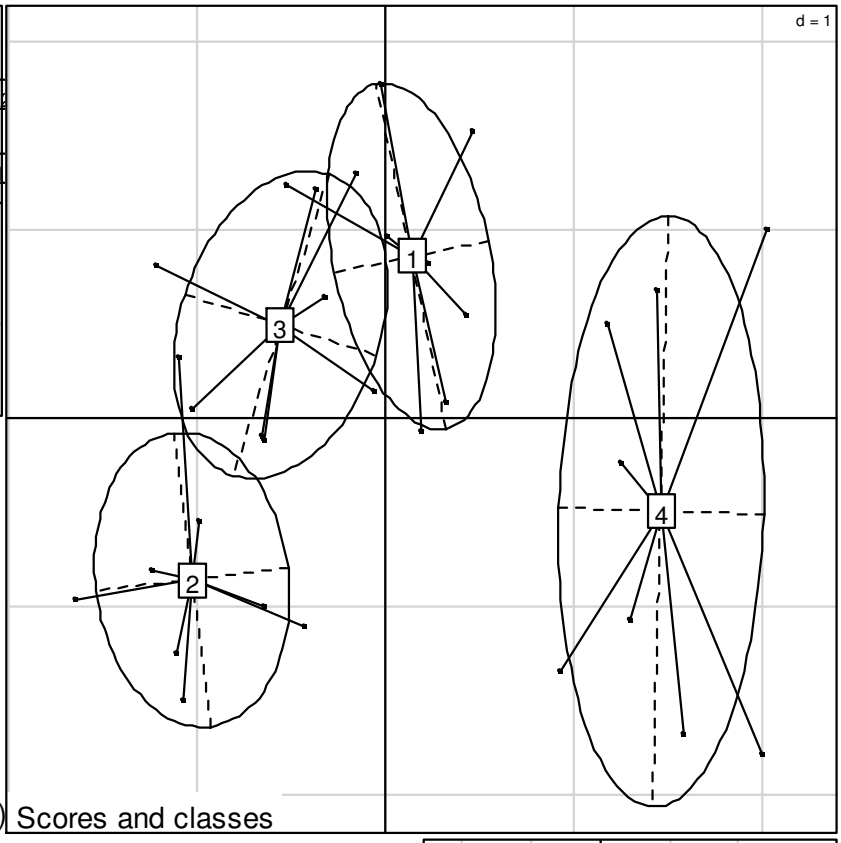


Prédiction

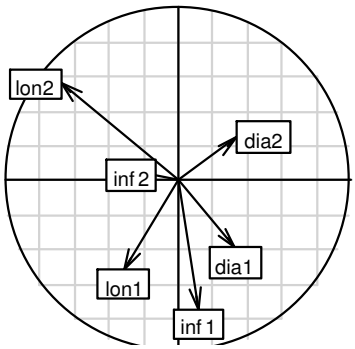
```
> table(cla,predict(afd1,tab)$class)

cla 1 2 3 4
  1 7 0 1 0
  2 0 7 1 0
  3 1 1 6 0
  4 0 0 0 8
```



Canonical weights

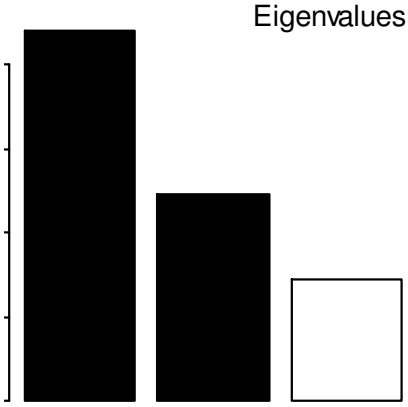


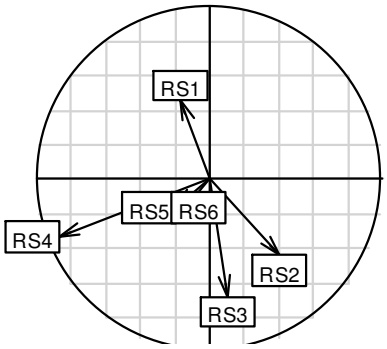
Scores and classes



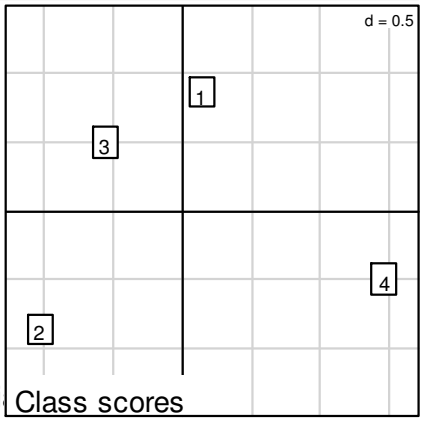
Cos(variates,canonical variates)

Eigenvalues

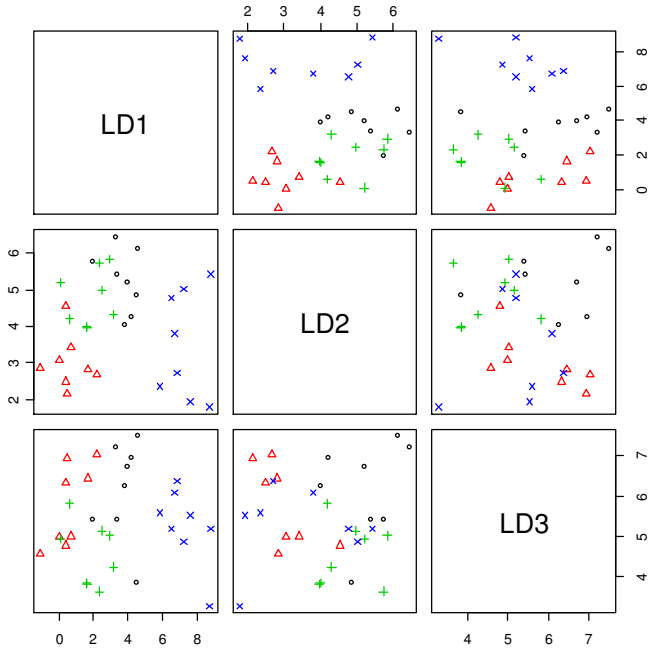




Cos(components,canonical vari



Class scores



Exemple V : Analyse discriminante et classement

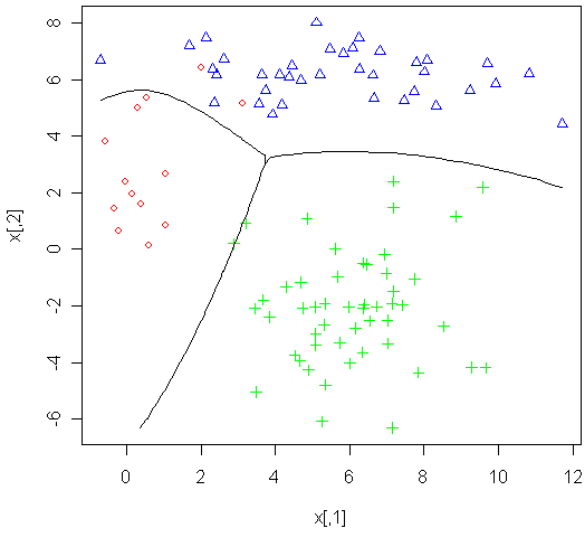
TP tiré de http://www.lsp.ups-tlse.fr/Fp/Bigot/Ens/Classif/tpclassif03.pdf

1 Analyse d'un mélange de vecteurs Gaussien

1.1 Simulation des données

Dans cette partie, on va simuler des données dans $\mathbb{R}^2$ issues d'un mélange de 3 vecteurs gaussien. On suppose que les probabilités a priori des classes sont p_1, p_2 et p_3. Pour générer ces données, il suffit de simuler une variable C qui prend les valeurs 1, 2 ou 3 avec probabilité p_1, p_2 et p_3 respectivement. Puis, conditionnellement à la valeur de C, on simule un vecteur qui suit une loi normale en dimension 2 (i.e. si C = k, on simule $x \sim N(\mu_k, \Sigma_k)$ pour k = 1, 2, 3). Ici les covariances sont nulles.

```
p1=0.2; p2=0.3; p3=0.5;n=100

s1=c(1,2); s2=c(3,1); s3=c(1.5,2)
s <- rbind(s1,s2,s3)

m1 =c(1,2); m2=c(6,6); m3=c(6,-2)
m <- rbind(m1,m2,m3)

c=sample(c(1,2,3), size=n,
prob=c(p1,p2,p3),replace=TRUE)

x = cbind(rnorm(n,m[c,1],s[c,1]),
rnorm(n,m[c,2],s[c,2]))
couleur <- rep("red",n)
couleur[c==2] <- "blue"
couleur[c==3] <- "green"
plot(x,col=couleur)
```



1.2 Règle de Bayes

Afin de représenter les frontières de décision données par la règle de Bayes, on discrétise le plan sous forme d'une grille de points régulièrement espacés dans le plan contenant les observations. Pour pouvoir classer chaque point x de la grille, on calcule le maximum de $p_k\, f_k(x)$ où $f_k(x)$ est la densité de la loi $N(\mu_k, \Sigma_k)$ pour k = 1, 2, 3. Les frontières de Bayes correspondent aux points x de la grille tels qu'il y ait égalité entre deux probabilités.

Afin de limiter le temps de calcul, on choisit une grille carrée de taille 50 sur 50 points (i.e. 2500 points) dont les extrémités sont données par les minima et les maxima des observations (en abscisse et ordonnée).

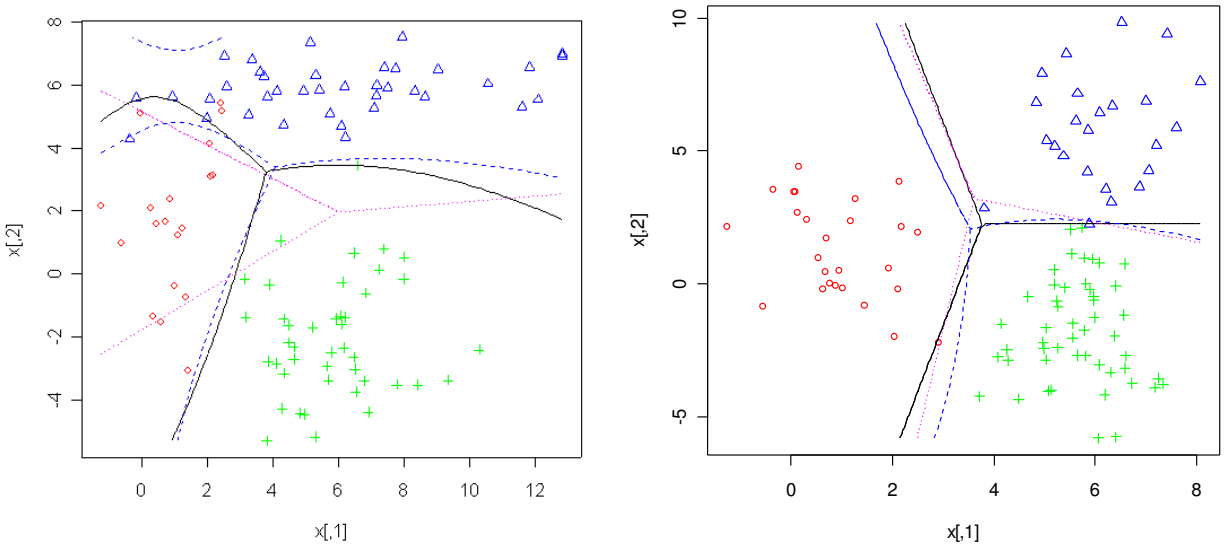
```
len <- 50
xp <- seq(min(x[,1]),max(x[,1]), length=len); yp <- seq(min(x[,2]),max(x[,2]), length=len)
grille <- expand.grid(z1=xp,z2=yp)
Z <- p1*dnorm(grille[,1],m[1,1],s[1,1])*dnorm(grille[,2],m[1,2],s[1,2])
Z <- cbind(Z,p2*dnorm(grille[,1],m[2,1],s[2,1])*dnorm(grille[,2],m[2,2],s[2,2]))
Z <- cbind(Z,p3*dnorm(grille[,1],m[3,1],s[3,1])*dnorm(grille[,2],m[3,2],s[3,2]))
zp <- Z[,3] - pmax(Z[,2], Z[,1])
contour(xp, yp, matrix(zp, len), add=TRUE, levels=0, drawlabels=FALSE)
zp <- Z[,1] - pmax(Z[,2], Z[,3])
contour(xp, yp, matrix(zp, len), add=TRUE, levels=0, drawlabels=FALSE)
```

1.3 Analyse discriminante linéaire et quadratique

La fonction lda (resp. qda) de la librairie MASS permet de réaliser l'analyse discriminante linéaire (resp. quadratique) d'un jeu de données à partir de l'estimation empirique des probabilités a priori, des moyennes et de la matrices de variance des classes. Si x est une matrice n×d qui représente n observations caractérisées par d variables explicatives, et si T est un vecteur de taille n qui représente la classe de chaque observation, alors l'analyse discriminante linéaire de x peut être effectuée à l'aide des instructions suivantes :

```
library(MASS)
T <- as.factor(couleur)
x.lda <- lda(x,T)
len <- 50
xp <- seq(min(x[,1]),max(x[,1]), length=len)
yp <- seq(min(x[,2]),max(x[,2]), length=len)
grille <- expand.grid(z1=xp,z2=yp)
Z <- predict(x.lda,grille)
zp <- as.numeric(Z$class)

zp <- Z$post[,3] - pmax(Z$post[,2], Z$post[,1])
contour(xp, yp, matrix(zp, len), add=TRUE, levels=0, drawlabels=FALSE,col="magenta")
zp <- Z$post[,1] - pmax(Z$post[,2], Z$post[,3])
contour(xp, yp, matrix(zp, len), add=TRUE, levels=0, drawlabels=FALSE,col="magenta")
```

matrices de variance-covariance différentes **matrices de variance-covariance égales**



Utiliser la fonction front.lda qui construit les frontières déterminées par les 3 méthodes dans les deux cas suivants: front.lda() où les matrice de variance covariance sont distinctes et front.lda(sigma="eq") où les matrices sont égales.

2 Evaluation du nombre de points mal-classés

On utilise les données crabs de la library MASS:

```
library(MASS)
data(crabs);?crabs
```

L'analyse discriminante linéaire utilisée porte sur la couleur et a pour variables le logarithme de FL et RW.

 # Separation selon la couleur

```
x <- log(cbind(crabs$FL,crabs$RW))
n <- dim(x)[1]
couleur <- rep("black",n)
couleur[crabs$sp == "O"] <- "magenta"
plot(x,col=couleur)
T <- as.factor(couleur)
x.lda <- lda(x,T)
```

 # Visualisation des frontieres de decision- Choix d'une grille

```
len <- 50
xp <- seq(min(x[,1]),max(x[,1]), length=len)
yp <- seq(min(x[,2]),max(x[,2]), length=len)
grille <- expand.grid(z1=xp,z2=yp)
Z <- predict(x.lda,grille)
Z <- Z$post
zp <- Z[,1] - Z[,2]
contour(xp, yp, matrix(zp, len), add=TRUE, levels=0, drawlabels=FALSE,col='green')
```

 # Evaluation du taux de points classés dans un ensemble de test

```
test <- x
col <- couleur
M <- dim(test)[1]
C <- rep(1,M)
C[col == "magenta"] <- 2
Z <- predict(x.lda,test)
erreur.lda <- sum(Z$class != col)/M
```

Remarque : attention, dans les exemples ci-dessus, l'erreur de classification est estimée à partir des données de l'ensemble d'apprentissage qui ont servi à l'estimation des paramètres des différents modèles. Vous pouvez constater que cette erreur est beaucoup trop optimiste.

3/ Méthode de l'échantillon test

Les instructions suivantes permettent de partitionner un ensemble de n individus en un ensemble d'apprentissage et un ensemble de test :

Partition des individus en deux sous-ensembles: apprentissage et test

```
choix <-
sample(c(1,2),size=n,prob=c(0.5,0.5),replace=
TRUE)
appr <- x[choix==1,]
test <- x[choix==2,]
appr.col <- couleur[choix==1]
test.col <- couleur[choix==2]
M <- dim(test)[1]
C <- rep(1,M)
C[test.col == "magenta"] <- 2
T <- as.factor(appr.col)
appr.lda <- lda(appr,T)
Z <- predict(appr.lda,test)
erreur.lda <- sum(Z$class != test.col)/M
appr.qda <- qda(appr,T)
Z <- predict(appr.qda,test)
erreur.qda <- sum(Z$class != test.col)/M
list(erreur.lda,erreur.qda)
```

4/ Validation croisée

Les instructions suivantes permettent de partitionner un ensemble de n individus en un ensemble d'apprentissage et un ensemble de test :

Partition des individus en deux sous-ensembles: apprentissage et test

```
x <- log(cbind(crabs$FL,crabs$RW))
n <- dim(x)[1]
couleur <- rep("black",n)
couleur[crabs$sp == "O"] <- "magenta"
erreur.lda <- 0
erreur.qda <- 0
for (i in 1:n) {
appr <- x[-i,]
test <- x[i,]
appr.col <- couleur[-i]
test.col <- couleur[i]
C <- test.col
T <- as.factor(appr.col)
appr.lda <- lda(appr,T)
Z <- predict(appr.lda,test)
erreur.lda <- erreur.lda +
sum(Z$class!=test.col)
appr.qda <- qda(appr,T)
Z <- predict(appr.qda,test)
erreur.qda <- erreur.qda +
sum(Z$class!=test.col)
}
erreur.lda <- erreur.lda/n
erreur.qda <- erreur.qda/n
list(erreur.lda,erreur.qda)
```

Linear Discriminant Analysis : lda

Arguments

formula	A formula of the form groups ~ x1 + x2 + ... That is, the response is the grouping factor and the right hand side specifies the (non-factor) discriminators.
data	Data frame from which variables specified in formula are preferentially to be taken.
x	(required if no formula is given as the principal argument.) a matrix or data frame or Matrix containing the explanatory variables.
grouping	(required if no formula principal argument is given.) a factor specifying the class for each observation.
prior	the prior probabilities of class membership. If unspecified, the class proportions for the training set are used. If present, the probabilities should be specified in the order of the factor levels.
tol	A tolerance to decide if a matrix is singular; it will reject variables and linear combinations of unit-variance variables whose variance is less than tol^2.
subset	An index vector specifying the cases to be used in the training sample. (NOTE: If given, this argument must be named.)
na.action	A function to specify the action to be taken if NAs are found. The default action is for the procedure to fail. An alternative is na.omit, which leads to rejection of cases with missing values on any required variable. (NOTE: If given, this argument must be named.)
method	"moment" for standard estimators of the mean and variance, "mle" for MLEs, "mve" to use cov.mve, or "t" for robust estimates based on a *t* distribution.
CV	If true, returns results (classes and posterior probabilities) for leave-one-out cross-validation. Note that if the prior is estimated, the proportions in the whole dataset are used.
nu	degrees of freedom for method = "t".
...	arguments passed to or from other methods.

Details

The function tries hard to detect if the within-class covariance matrix is singular. If any variable has within-group variance less than tol² it will stop and report the variable as constant. This could result from poor scaling of the problem, but is more likely to result from constant variables.

Specifying the prior will affect the classification unless over-ridden in predict.lda. Unlike in most statistical packages, it will also affect the rotation of the linear discriminants within their space, as a weighted between-groups covariance matrix is used. Thus the first few linear discriminants emphasize the differences between groups with the weights given by the prior, which may differ from their prevalence in the dataset.

If one or more groups is missing in the supplied data, they are dropped with a warning, but the classifications produced are with respect to the original set of levels.

Value

If CV = TRUE the return value is a list with components class, the MAP classification (a factor), and posterior, posterior probabilities for the classes. Otherwise it is an object of class "lda" containing the following components:

prior	the prior probabilities used.
means	the group means.
scaling	a matrix which transforms observations to discriminant functions, normalized so that within groups covariance matrix is spherical.
svd	the singular values, which give the ratio of the between- and within-group standard deviations on the linear discriminant variables. Their squares are the canonical F-statistics.
N	The number of observations used.
call	The (matched) function call.

Note

This function may be called giving either a formula and optional data frame, or a matrix and grouping factor as the first two arguments. All other arguments are optional, but subset= and na.action=, if required, must be fully named.

If a formula is given as the principal argument the object may be modified using update() in the usual way.

References

Venables, W. N. and Ripley, B. D. (2002) *Modern Applied Statistics with S.* Fourth edition. Springer.

Ripley, B. D. (1996) *Pattern Recognition and Neural Networks*. Cambridge University Press.

See Also

predict.lda, qda, predict.qda

Classify Multivariate Observations by Linear Discrimination

Description

Classify multivariate observations in conjunction with lda, and also project data onto the linear discriminants.

Usage

```
predict(object,newdata,prior=object$prior,dimen,method=c("plug-in","predictive","debiased"))
```

Arguments

object	object of class "lda"
newdata	data frame of cases to be classified or, if object has a formula, a data frame with columns of the same names as the variables used. A vector will be interpreted as a row vector. If newdata is missing, an attempt will be made to retrieve the data used to fit the lda object.
prior	The prior probabilities of the classes, by default the proportions in the training set or what was set in the call to lda.
dimen	the dimension of the space to be used. If this is less than min(p, ng-1), only the first dimen discriminant components are used (except for method="predictive"), and only those dimensions are returned in x.
method	This determines how the parameter estimation is handled. With "plug-in" (the default) the usual unbiased parameter estimates are used and assumed to be correct. With "debiased" an unbiased estimator of the log posterior probabilities is used, and with "predictive" the parameter estimates are integrated out using a vague prior.
...	arguments based from or to other methods

Value a list with components

class	The MAP classification (a factor)
posterior	posterior probabilities for the classes
x	the scores of test cases on up to dimen discriminant variables

Examples

```
data(iris3); tr <- sample(1:50, 25); train <- rbind(iris3[tr,,1], iris3[tr,,2], iris3[tr,,3])
test <- rbind(iris3[-tr,,1], iris3[-tr,,2], iris3[-tr,,3])
cl=factor(c(rep("s",25),rep("c",25),rep("v",25))); z=lda(train, cl); predict(z, test)$class
```

Linear Discriminant Analysis (descriptive statistic)

Usage `discrimin(dudi, fac); plot.discrimin (x, xax = 1, yax = 2) print.discrimin (x)`

Arguments

dudi	a duality diagram, object of class dudi
fac	a factor defining the classes of discriminant analysis
scannf	a logical value indicating whether the eigenvalues bar plot should be displayed
nf	if scannf FALSE, an integer indicating the number of kept axes
x	an object of class 'discrimin'
xax	the column number of the x-axis
yax	the column number of the y-axis
...	further arguments passed to or from other methods

Value returns a list of class 'discrimin' containing :

nf	a numeric value indicating the number of kept axes
eig	a numeric vector with all the eigenvalues
fa	a matrix with the loadings: the canonical weights
li	a data frame which gives the canonical scores
va	a matrix which gives the cosines between the variables and the canonical scores
cp	a matrix which gives the cosines between the components and the canonical scores
gc	a data frame which gives the class scores

Author(s)

Daniel Chessel chessel@biomserv.univ-lyon1.fr Anne B Dufour dufour@biomserv.univ-lyon1.fr

Examples

```
data(chazeb);
dis1 <- discrimin(dudi.pca(chazeb$tab, scan = FALSE), chazeb$cla,scan = FALSE);dis1
plot(dis1);data(skulls);plot(discrimin(dudi.pca(skulls, scan = FALSE), gl(5,30),scan = FALSE))
```