TD5 -Partial Least Squares - Discriminant Analysis

Courtenay Rebecca & Ducros Chloé & Lasson Marie

Contents

1 Introduction

Dans ce TD, nous allons faire un PLS-DA. Regardez votre cours partie 5.3 (page 43).

On utilise un dataset indiquant l'expression de 6830 gènes dans des échantillons de 64 cancers.

On importe les données et on retire du dataset les sous-types de cancer suivants : UNKNOWN, K562B-repro,K562A-repro, MCF7A-repro, MCF7D-repro (à faire).

```
dat.1 <- read.table(file="nci.data.txt")
names.data <- as.character(read.csv(file="names-sample.csv",header=FALSE,sep=" ")[,1])
```

2 Question

2.1 Retirer de dat.1 et de names.final les sous-types de cancer suivants : UNKNOWN, K562B-repro,K562A-repro, MCF7A-repro, MCF7D-repro.

```
ech <- c("UNKNOWN", "K562B-repro", "K562A-repro", "MCF7A-repro", "MCF7D-repro")
supp <- which(names.data%in%ech)
names.final <- names.data[-supp]
dat.1 <- dat.1[,-supp]
```

```
table(names.final)
```

```
## names.final
##            BREAST      CNS    COLON LEUKEMIA MELANOMA    NSCLC  OVARIAN
##         4        7        5        7        6        8        9        6
## PROSTATE    RENAL
##         2        9
```

Une fois les sous-types de cancer retirés, on travaille avec 59 échantillons.

On définit alors Y, une variable de réponse binaire :

- 1 pour les sous-types de cancer : colon, leukemia, prostate, NSCLC
- 0 pour : BREAST, SCNS, MELANOMA, OVARIAN, RENAL

```
Y <- rep(0,59)
group1 <- which(names.final%in%c("COLON","LEUKEMIA","PROSTATE","NSCLC"))
Y[group1] <- 1
table(Y)
```

```
## Y
##  0  1
## 35 24
```

On importe ensuite `mixOmics` et on définit notre X et Y.

```
library(mixOmics)
X <- t(dat.1)
Y <- factor(Y)
```

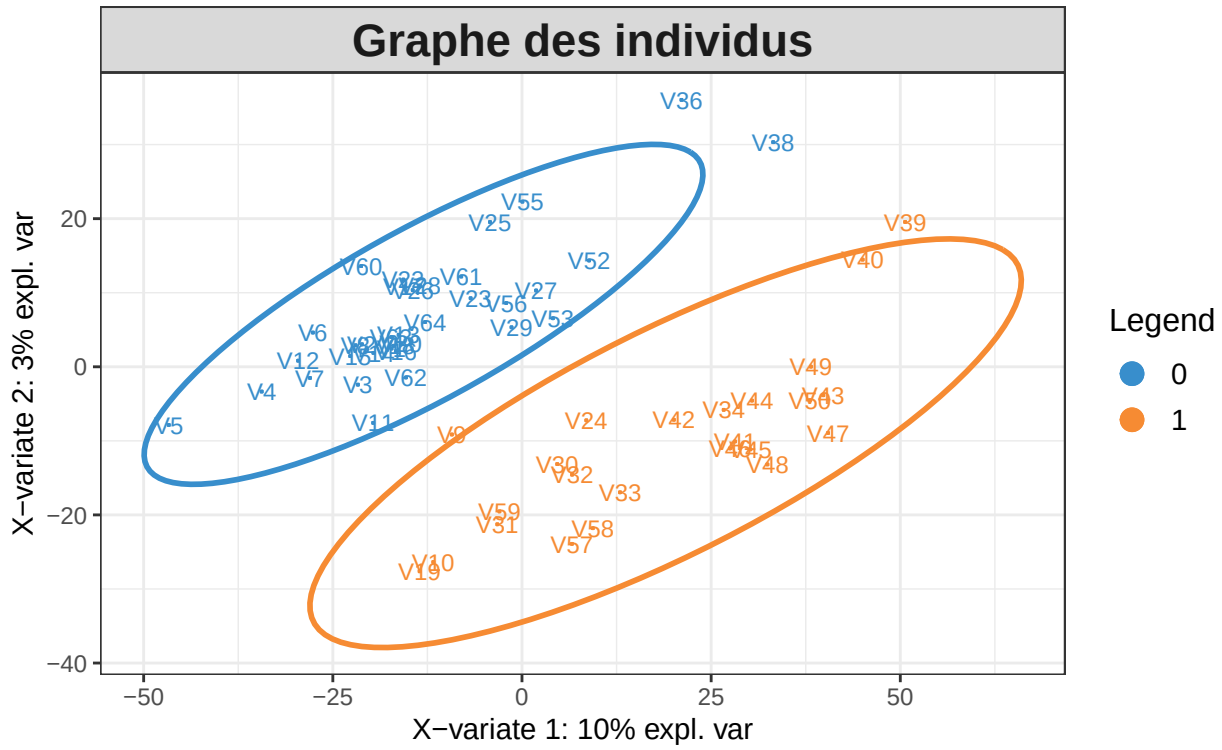
2.2 Expliquer pourquoi on effectue les deux lignes `X <- t(dat.1)` et `Y <- factor(Y)` (2 points)

Afin de pouvoir calculer faire des produits matricielles entre Y et X, il est nécessaire de prendre la transposé de X.

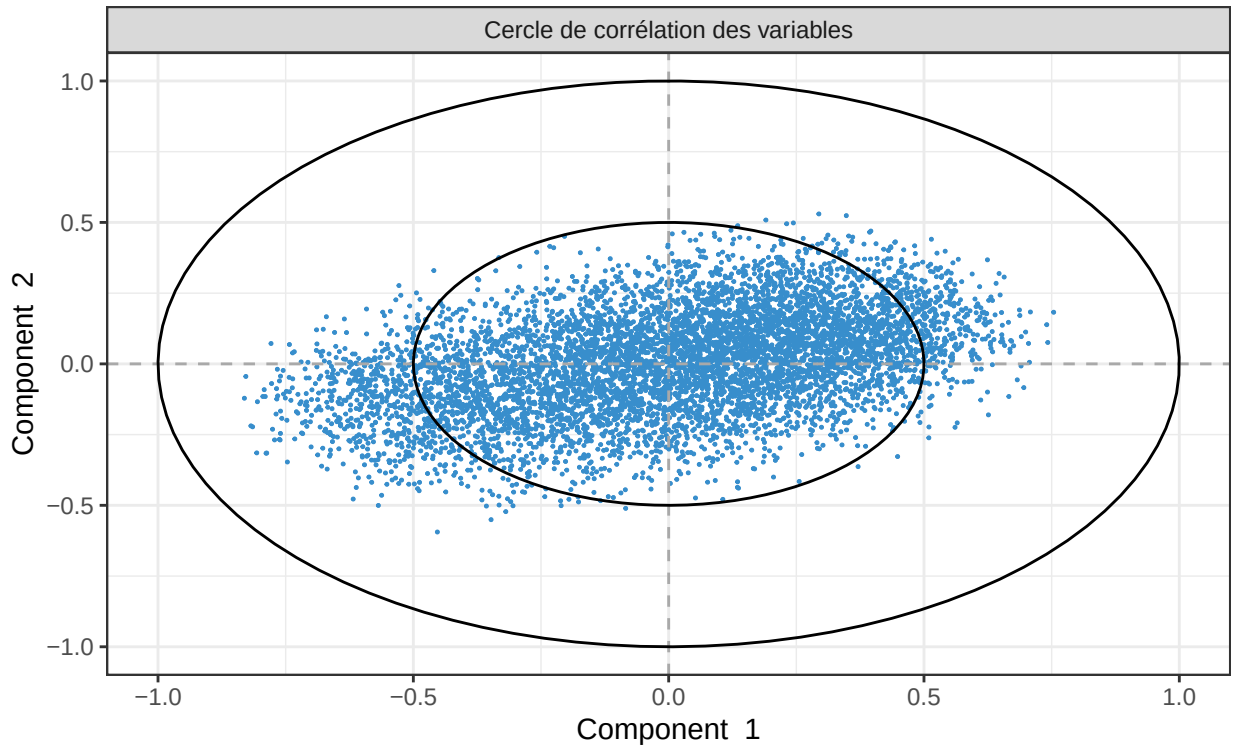
Concernant Y, comme il ne peut prendre que 1 ou 0, celui-ci est un vecteur dont les éléments ne peuvent prendre que des modalités prédéfinies (numérique : 0 et 1). C'est pour cela qu'on le converti en facteur afin qu'il définisse ces deux classes.

2.3 Effectuer la PLS-DA des données et tracer le graphe des individus et le cercle des corrélations des variables.

```
#Run the method
pls_da <- plsda(X, Y)
#Plot the samples
plotIndiv(pls_da,legend=TRUE, ellipse = TRUE, title = 'Graphe des individus')
```

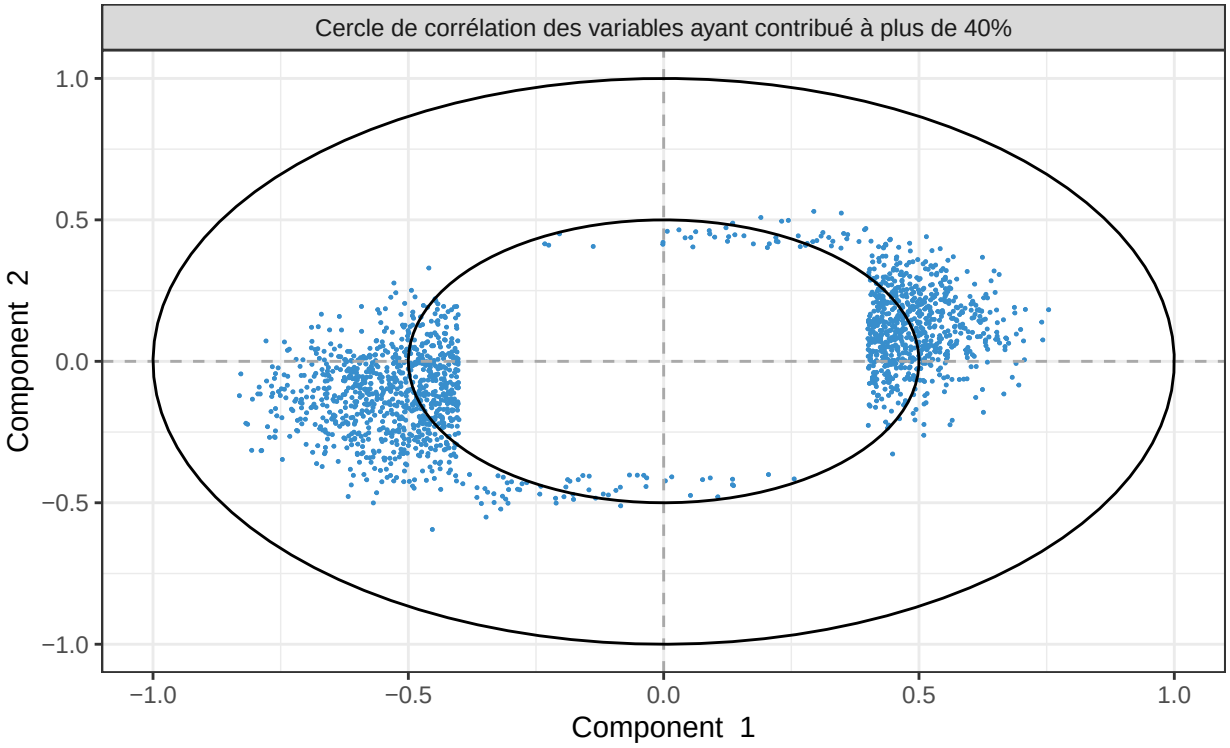



```
#Plot the variables
plotVar(pls_da, var.names=FALSE,cex=0.2, title="Cercle de corrélation des variables")
```



2.4 Tracer le cercle des variables avec seulement les variables ayant contribué à plus de 40% à la création des axes.

```
plotVar(pls_da, cutoff=0.4, var.names=FALSE, cex=0.2,
        title="Cercle de corrélation des variables ayant contribué à plus de 40%")
```

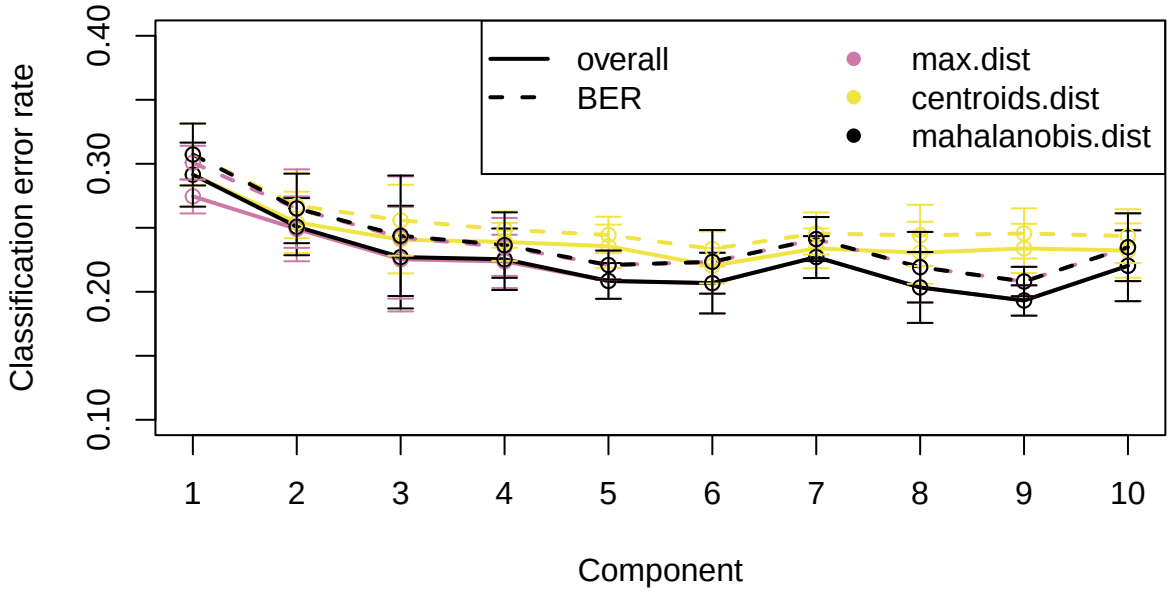


2.5 Effectuer la PLS-DA avec 10 composants, puis effectuer la classification avec la fonction `perf` et 10 répétitions.

Pour la reproductibilité de cette question, on définira la seed à 2.

Ensuite, afficher le résultat grâce à: `plot(résultat_de_perf, col = color.mixo(5:7), sd = TRUE, legend.position = "horizontal",ylim=c(0,0.4))`

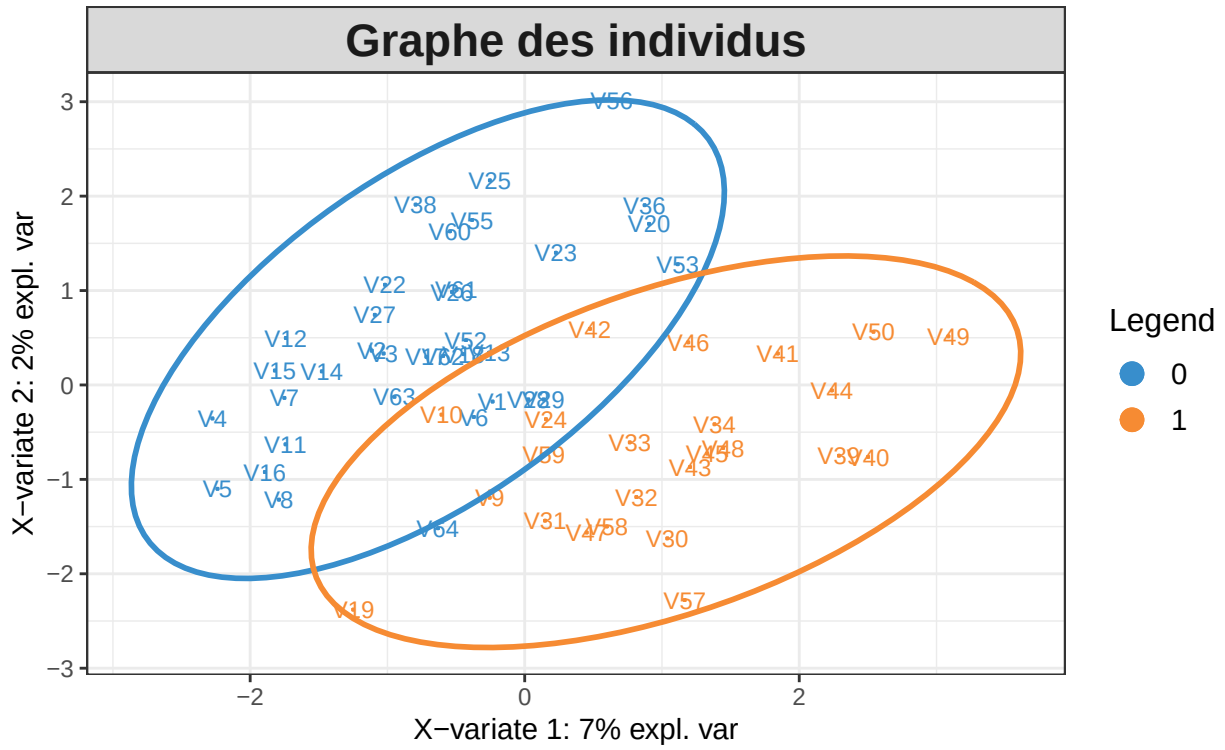
```
set.seed(2)
pls_da2 <- plsda(X,Y, ncomp=10)
résultat_de_perf <-  perf(pls_da2, nrepeat = 10)
plot(résultat_de_perf, col = color.mixo(5:7), sd = TRUE,
     legend.position = "horizontal",ylim=c(0.1,0.4))
```



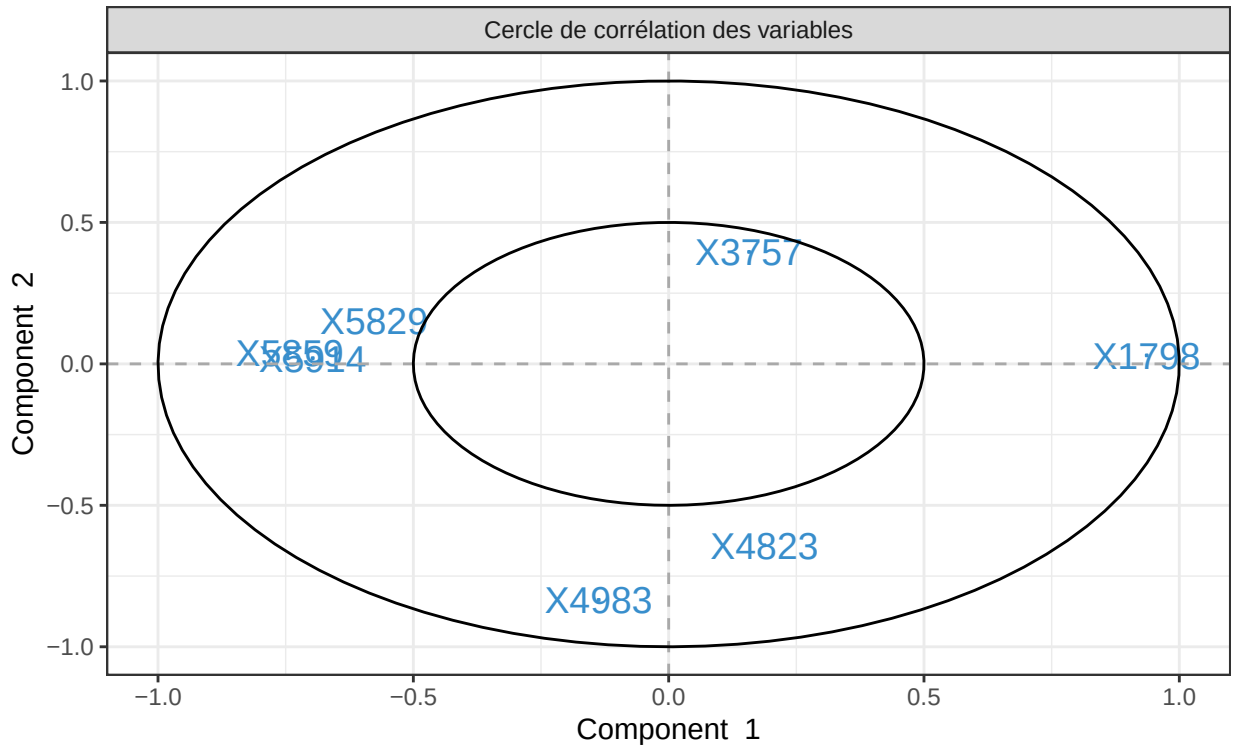
2.6 Lancer une sparse PLS-DA en ne gardant dans les loadings de X que 4 variables sur le premier axe et 3 sur le deuxième et afficher le graphe des individus et le cercle des variables.

Si besoin, se référer à la documentation : https://www.rdocumentation.org/packages/mixOmics/versions/6.3.2/topics/splsda (2 points)

```
spls_da <- splsda(X, Y, keepX = c(4,3))
#Plot the samples
plotIndiv(spls_da,legend=TRUE, ellipse = TRUE, title = 'Graphe des individus')
```




```
#Plot the variables
plotVar(spls_da,title="Cercle de corrélation des variables")
```



2.7 On cherche à trouver le nombre optimal de variables à garder. Pour cela on crée une liste de nombres qu'on va tester :

```
list.keepX <- c(5:10, seq(20, 200, 10))
list.keepX
```

```
##  [1]   5   6   7   8   9  10  20  30  40  50  60  70  80  90 100 110 120 130 140
## [20] 150 160 170 180 190 200
```

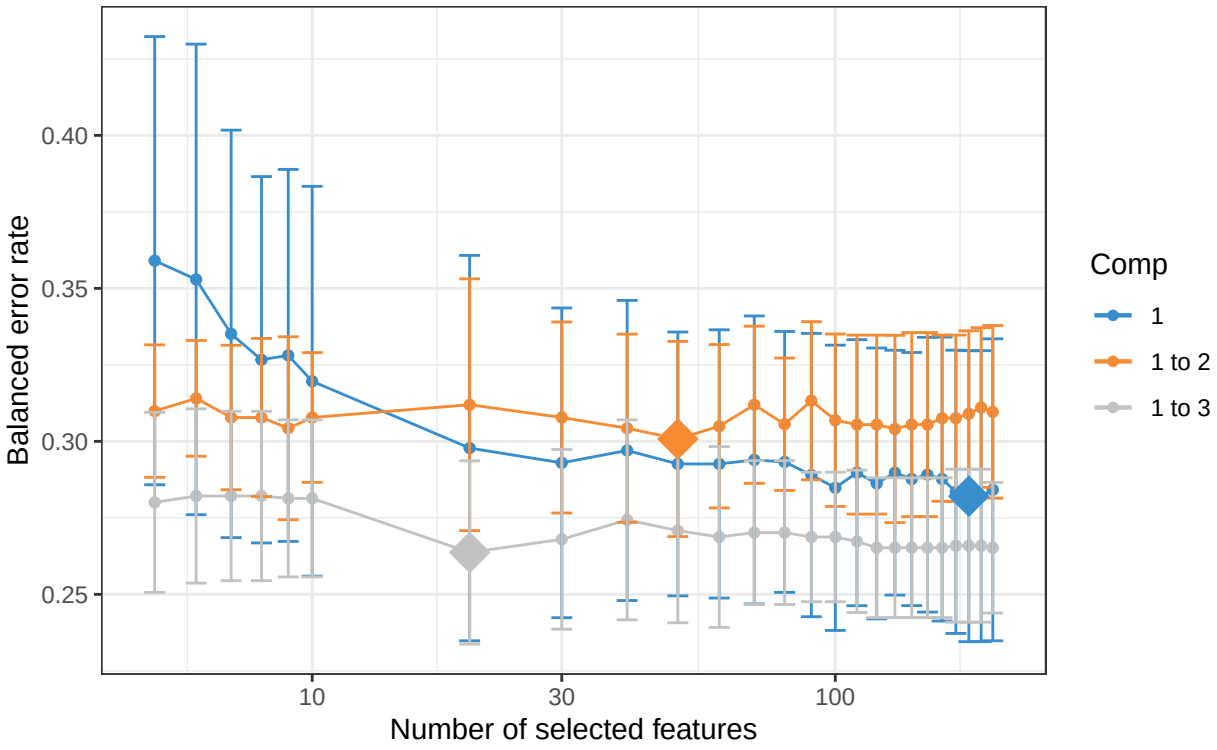
Fixer la seed à 2 et, grâce à la fonction `tune.splsda`, trouver le nombre optimal de variables à garder sur les deux premiers axes. Ensuite, afficher le résultat de l'appel grâce à la fonction 'plot' (4 points)

```
set.seed(2)
spls_da_tune <- tune.splsda(X, Y, ncomp = 3, dist = 'max.dist', progressBar = FALSE,
                            validation = 'Mfold', folds = 3, measure = "BER",
                            test.keepX = list.keepX, nrepeat = 10)

spls_da_tune$choice.keepX[1:2]
```

```
## comp1 comp2
##   180    50
```

```
plot(spls_da_tune)
```



2.8 Modèle final

Réaliser la PLS-DA avec le nombre optimal de variables sur les deux premiers axes, puis tracer le graphe des individus.

```
select.keepX <- spls_da_tune$choice.keepX[1:2]
pls_da_final <- splsda(X, Y, keepX = select.keepX)
```

```
plotIndiv(pls_da_final,legend=TRUE, ellipse = TRUE, title = 'Graphe des individus')
```