clustering hiérarchique

Importer les données:

```
data2 <-  read.csv('C:/Users/asus/OneDrive/Desktop/Projet/data/lung_cancer.csv',
                   sep=';', row.names='id_sample')
dim(data2)
```

```
## [1] 150  53
```

```
head(data2)
```

```
##                      APOC1    MERTK    CMTM3     XPR1    H2BC5    MSMO1   TMEM97
## TCGA-43-6647-01A 3.692305 2.569732 4.155590 4.094676 2.792685 4.901656 2.290414
## TCGA-22-4593-01A 2.949213 1.027134 4.030874 2.677671 3.339568 6.229210 5.446595
## TCGA-92-7341-01A 5.400119 3.897991 3.961812 4.805674 3.608529 6.396105 4.997865
## TCGA-68-8251-01A 3.774515 2.150148 4.081626 3.183550 1.947876 4.221295 4.161457
## TCGA-85-8479-01A 2.022175 0.795053 2.062744 3.044048 5.256811 4.986559 2.568166
## TCGA-85-8666-01A 5.081326 1.847408 3.472309 5.064267 2.332815 4.554978 3.791437
##                      IFFO2   FAM89A     CTSH     DOK4  ARFGEF3    FOXA1     NCF2
## TCGA-43-6647-01A 4.404727 2.749062 6.560401 2.916253 0.645437 0.305448 3.777752
## TCGA-22-4593-01A 4.415215 2.714060 2.739495 2.806878 0.873398 3.780116 2.238145
## TCGA-92-7341-01A 3.701787 4.218262 4.566433 2.870315 1.356950 0.632262 3.154392
## TCGA-68-8251-01A 2.794275 3.145032 4.488327 2.390135 1.460426 3.601031 3.475740
## TCGA-85-8479-01A 4.760918 3.918480 5.046272 3.395144 0.731007 2.367649 3.425198
## TCGA-85-8666-01A 3.437406 3.527964 4.553245 2.121254 1.059957 3.131171 4.866393
##                  SNORD62B     PYGB     H3P6 DGUOK.AS1     APOE IGLV7.46
## TCGA-43-6647-01A 1.109690 5.036269 3.427798  0.812759 6.289549 3.081413
## TCGA-22-4593-01A 0.801689 6.715492 3.967429  1.528098 4.525605 3.328695
## TCGA-92-7341-01A 1.013383 5.952289 2.778527  2.534483 6.279737 3.029898
## TCGA-68-8251-01A 2.172688 4.648795 2.819027  0.800143 5.729022 4.264692
## TCGA-85-8479-01A 2.643929 4.354768 4.215500  3.342880 3.142600 0.205359
## TCGA-85-8666-01A 1.084348 6.636762 3.282881  1.402210 6.347396 2.191062
##                       NES     STOM      ELN   ERRFI1    IL1RN     FZD5    LIMK2
## TCGA-43-6647-01A 3.712585 5.715155 2.494779 5.658543 5.352689 1.603194 5.315352
## TCGA-22-4593-01A 3.019713 5.588161 2.317363 5.340114 3.064744 1.109417 4.509576
## TCGA-92-7341-01A 2.460793 6.238754 1.642679 4.437130 4.665649 1.180638 4.455959
## TCGA-68-8251-01A 3.298517 5.682438 2.823099 4.680525 3.418981 1.557129 4.214350
## TCGA-85-8479-01A 1.864746 6.854498 1.099852 4.923351 5.439573 2.236906 5.158902
## TCGA-85-8666-01A 4.111301 6.649276 1.385599 4.360964 6.486299 1.996883 3.758357
##                       IER5 RNU6.850P     CTSW CHCHD2P9   LGALS2  RPS2P46
## TCGA-43-6647-01A 3.347011  0.634589 1.153725 2.292747 6.170941 2.258906
## TCGA-22-4593-01A 2.617084  1.095964 1.444990 6.232879 5.031688 3.905070
## TCGA-92-7341-01A 2.639902  0.360089 1.418589 3.269786 5.849802 2.854741
## TCGA-68-8251-01A 2.940741  1.676786 2.255189 3.238307 5.364434 2.594649
```

```
## TCGA-85-8479-01A 4.865372  2.208319 0.508483 3.901956 7.173845 1.959871
## TCGA-85-8666-01A 4.823241  0.952701 1.460307 2.059909 5.898441 1.822573
##                  KDM7A.DT   FLVCR2     PRC1    FOXM1   MICAL1   MHENCR   CELSR2
## TCGA-43-6647-01A 2.416775 1.468010 3.113603 3.987042 2.455274 1.687131 3.301152
## TCGA-22-4593-01A 2.935570 0.614639 4.742355 4.809701 2.043815 1.987484 4.374073
## TCGA-92-7341-01A 3.364594 0.994641 4.657988 4.644909 1.487677 2.450438 2.748732
## TCGA-68-8251-01A 1.306755 0.979042 4.164329 4.724946 2.828199 2.095127 2.900037
## TCGA-85-8479-01A 3.025608 0.434705 4.540502 4.039842 1.532035 3.500577 5.417462
## TCGA-85-8666-01A 3.622431 1.358447 4.000246 5.434557 2.603257 1.890106 4.295918
##                    SEZ6L2     SORD    SNAI1     MELK    MMP12     SOX9    APBA2
## TCGA-43-6647-01A 5.913647 2.713180 2.952726 2.044266 7.120371 4.692437 2.630496
## TCGA-22-4593-01A 5.466573 2.730374 1.408844 3.737576 2.693306 5.017445 1.988419
## TCGA-92-7341-01A 3.144082 3.960228 1.417569 3.668505 6.228964 3.631651 2.902791
## TCGA-68-8251-01A 3.212341 3.244686 2.836101 3.278803 6.731229 4.789792 2.956461
## TCGA-85-8479-01A 5.758853 1.904738 1.699228 2.593756 1.851387 3.478012 0.346543
## TCGA-85-8666-01A 3.229790 2.845487 2.992420 3.765791 7.134901 5.574143 1.137086
##                      NFIB     GRK5 TMEM150B tissue_status    source class
## TCGA-43-6647-01A 1.996536 1.927014 1.280060       tumoral TCGA-LUSC   SQC
## TCGA-22-4593-01A 4.082883 1.961332 0.320457       tumoral TCGA-LUSC   SQC
## TCGA-92-7341-01A 2.988236 1.996756 0.459460       tumoral TCGA-LUSC   SQC
## TCGA-68-8251-01A 2.591611 1.337750 0.842107       tumoral TCGA-LUSC   SQC
## TCGA-85-8479-01A 0.749769 1.143228 0.171289       tumoral TCGA-LUSC   SQC
## TCGA-85-8666-01A 2.380319 2.111310 1.441280       tumoral TCGA-LUSC   SQC
```

Extraire les données d'expression et les étiquettes

```
#Données d'expression de 50 gènes
X <- subset(data2, select = sapply(data2, is.numeric))
print(paste("X", dim(X)[1], dim(X)[2], sep=" "))
```

```
## [1] "X 150 50"
```

```
#Etiquettes des échantillons
y <- data2$class
cat("y", length(y), length(unique(y)),
    "['", paste(unique(y), collapse = "' '"), "']", sep=" ")
```

```
## y 150 3 [' SQC' 'ADK' 'NTL ']
```

Appliquer une normalisation centrée-réduite:

Une normalisation de données est nécessaire pour ramener les niveaux d'expression de gènes à la même échelle. Pour des raisons pédagogiques, on réalise ici une normalisation centrée-réduite explicitement dans une étape séparée. Il est néanmoins possible de l'intégrer à la dernière étape du pipeline, simplement en option de la méthode **scale()** de la librairie **stats()**

```
#Charger la bibliothèque
library(stats)
```

```r
#Normalisation centrée-réduite
X_scaled <- scale(X)
X_scaled <- as.data.frame(X_scaled)
rownames(X_scaled) <- rownames(X)
colnames(X_scaled) <- colnames(X)
print(paste("X_scaled", dim(X_scaled)))
```

```
## [1] "X_scaled 150" "X_scaled 50"
```

Définir les couleurs pour chaque classe:

```r
#Définir les couleurs pour chaque classe
class_color <- c('NTL' = 'aquamarine',
                 'ADK' = 'darkslateblue',
                 'SQC' = 'darkorange')

#Etiquettes des échantillons
y <- data2$class
#print(paste("y", length(y), unique(y)))

# Convertir les étiquettes en couleurs
y_color <- class_color[y]
```

Réaliser un clustering hiérarchique:

```r
#Installer la bibliothèque si ce n'est pas déjà fait
if (!requireNamespace("pheatmap", quietly = TRUE)) {
  install.packages("pheatmap")
}

#Charger la bibliothèque
library(pheatmap)

#Paramètres du clustering hiérarchique
metric <- 'euclidean'
method <- 'ward.D2'

sample_rows <- sample(rownames(X_scaled), size = 25)
sample_cols <- sample(colnames(X_scaled), size = 15)
X_scaled_ <- X_scaled[sample_rows, sample_cols]
#Créer une matrice de données transposée (car pheatmap utilise les colonnes pour le clustering)
X_scaled_transposed <- t(X_scaled)

#Assurer que y_color a la bonne longueur (le nombre de colonnes dans X_scaled)
y_color <- rep(y_color, each = ncol(X_scaled))

#Créer le heatmap avec clustering hiérarchique
```

```
#clustergrid <- pheatmap(
#  X_scaled_transposed,
#  clustering_distance_rows = metric,
#  clustering_method = method,
#  col = y_color,
#  width = 18,
#  height = 17
#)
```

```
# Récupérer la figure
#fig <- clustergrid

# Sauvegarder la figure en tant qu'image PNG
#ggsave(file = "C:/Users/asus/OneDrive/Desktop/Projet/hierarchical_clustering2.png",
#plot = fig,dpi = 300, width = 8, height = 7, units = "in", device = "png")
```