Application breast cancer division par première valeur singulière

28 novembre, 2019

Contents

Changements depuis le dernier script : singular value au carré. 27/09/19 : Re-changement : c'est la racine carré de l'eigenvalue par laquelle il faut diviser, pas par le carré !

```
rm(list = ls())
```

```
library(kableExtra)
library(aricode)
library(devtools)
library(mergeTrees)
library(ggplot2)
library(gridExtra)
library(RColorBrewer)
library(dendextend)
library(tidyverse)
library(viridis)
library(rsvd)
library(svd)
```

1 Settings

```r
dist_arg = "euclidean"
linkage_arg = "ward.D2"
# linkage_arg = "single"
new_plot_window = FALSE
par(mar = c(2,2,2,2))
par(mfrow = c(1,1))

# Figure settings
cex.main_arg = 1.2
cex.axis_arg = 1.2
cex.rowlabels_arg = 1.1
height_arg = 3.92
width_arg = 5.4
mar_arg = c(3,4.5,1,0)
```

1.1 Scale/center settings

```r
center_arg = TRUE
scale_arg = FALSE
```

1.2 SVD settings

```r
k_svd = 5
```

2 Fonctions

Piquées à Julien

```r
directClustering <- function(dataSets) {
  hclust(dist(do.call("cbind", dataSets), method = "euclidean"), method = "ward.D2")
}

averagedClustering <- function(dataSets) {
  AD <- Reduce("+", lapply(dataSets, dist, method = "euclidean")) / length(dataSets)
  hclust(AD, method = "ward.D2")
}

mergeTreesWard <- function(dataSets) {
  hc_list <- lapply(dataSets, FUN = function(x) {
    univarclust::ward_1d(x)
  })
  mergeTrees::mergeTrees(hc_list)
}
```

3 Chargement des données

```r
load("tcga_brca_data.RData")
load("clinical.RData")

clinic1 = clinic1[order(clinic1$bcr_patient_barcode),]
zmethyl = zmethyl[order(rownames(zmethyl)),]
zmirna = zmirna[order(rownames(zmirna)),]
```

```r
zmutation = zmutation[order(rownames(zmutation)),]
zprotein = zprotein[order(rownames(zprotein)),]
zrna = zrna[order(rownames(zrna)),]
ztumor = ztumor[order(rownames(ztumor)),]

dataSets = list(
                "methyl" = zmethyl,
                "mirna" = zmirna,
                "protein" = zprotein,
                "rna" = log2(zrna+1))
```

```r
dataSets_0 = dataSets = lapply(dataSets, scale, center = center_arg, scale = scale_arg)
```

```r
dataSets = lapply(dataSets, FUN = function(dat){
  dat = dat/svd(dat,  nu = 0, nv = 0)$d[1] # division par premiere valeur singuliere
  })
```

4 Traitement des données cliniques

```r
clinical = clinic1
rownames(clinical) = clinical$bcr_patient_barcode
clinical = clinical[,-which(colnames(clinical)=="bcr_patient_barcode")]
```

4.1 Légende pour les clustering

```r
# Legende:
the_bars = data.frame(apply(clinical, 2, as.character))
the_bars$subtype = as.character(the_bars$subtype)

the_bars$ER_status = ifelse(the_bars$ER_status=="Positive", "grey88", "black")
the_bars$PR_status = ifelse(the_bars$PR_status=="Positive", "grey88", "black")
the_bars$subtype[the_bars$subtype=="Basal-like"] = "dimgray" ; the_bars$subtype[the_bars$subtype=="HER2-enriched
the_bars$subtype[the_bars$subtype=="Luminal A"] = "mistyrose3" ; the_bars$subtype[the_bars$subtype=="Luminal B"]

the_bars = the_bars[,which(colnames(the_bars)%in%c("subtype", "ER_status", "PR_status"))]
```

5 Application des méthodes MC, AD, DC

```r
hc_list_methods = list()
```

5.1 Méthodes multivariées

```r
hc_list = lapply(dataSets, FUN = function(x) hclust(dist(x, method = dist_arg), method = linkage_arg))
```

```r
hc_list_methods$AD = averagedClustering(dataSets)
hc_list_methods$DC = directClustering(lapply(dataSets, scale, center = TRUE, scale = FALSE))
hc_list_methods$MC = mergeTrees(hc_list)
```

5.2 Méthodes spectrales multivariées

5.2.1 Spectral sur tables séparées

```
rSVD <- lapply(dataSets, rsvd, k = k_svd)
rSVD_dataSets = lapply(rSVD, FUN = function(svd_res) svd_res$u%*% diag(svd_res$d))
```

```
dist_sp_list = lapply(rSVD_dataSets, FUN = function(dat) dist(dat, method = dist_arg))
hc_sp_list = lapply(dist_sp_list, FUN = function(dist_mat) hclust(dist_mat, method = linkage_arg))
```

```
hc_list_methods$SDC = directClustering(rSVD_dataSets)
hc_list_methods$SAD = averagedClustering(rSVD_dataSets)
hc_list_methods$SMC = mergeTrees(hc_sp_list)
```

5.3 Méthodes univariees

```
dataSets_univar = as.list(data.frame(Reduce("cbind", dataSets)))
```

```
hc_list_methods$ADuni = averagedClustering(dataSets_univar)
hc_list_methods$MCuni = mergeTreesWard(dataSets_univar)
```

5.4 Méthodes spectrale univariees

5.4.1 Spectral sur tables concaténées

```
rSVD <- rsvd(do.call("cbind", dataSets), k = k_svd)
dataSets_spectral <- as.list(as.data.frame(rSVD$u %*% diag(rSVD$d)))
```

```
hc_list_methods$ScDC = hclust(dist(as.data.frame(rSVD$u %*% diag(rSVD$d)), method = "euclidean"), method = "ward
hc_list_methods$ScADuni = averagedClustering(dataSets_spectral)
hc_list_methods$ScMCuni = mergeTreesWard(dataSets_spectral)
```

5.5 Sectral sur tables de base

```
rSVD <- lapply(dataSets, rsvd, k = k_svd)
rSVD_dataSets = lapply(rSVD, FUN = function(svd_res) svd_res$u%*% diag(svd_res$d))
names(rSVD_dataSets) = paste0(names(rSVD_dataSets), "sp")

hc_list = c(hc_list, lapply(rSVD_dataSets, FUN = function(x) hclust(dist(x, method = dist_arg), method = linkage
```

6 Comparaison des arbres des tables de base

```
NID_compare = function(tree_1, tree_2, cut_index_max = NULL){
  if(is.null(cut_index_max)) cut_index_max = length(tree_1$order)
  unlist(lapply(2:cut_index_max, FUN = function(cut_index) NID(cutree(tree_1, k = cut_index),
                                                               cutree(tree_2, k = cut_index))))
}

mat_NID_compare = sapply(hc_list, function(x) sapply(hc_list, function(y) min(NID_compare(x,y, cut_index_max = 2
```

7 NID exploration fonction @ Julien

7.1 Parametre pour niveau de coupure maximum

```
cutree_index_max = 104
```

7.2 Fonction get_nid

```
get_nid <- function(clustering, reference) {
  clusterings <- cutree(clustering, seq.int(1:cutree_index_max)) %>% as.data.frame() %>% as.list()
  nid <- map_dbl(clusterings, ~NID(., reference))
  nid
}
```

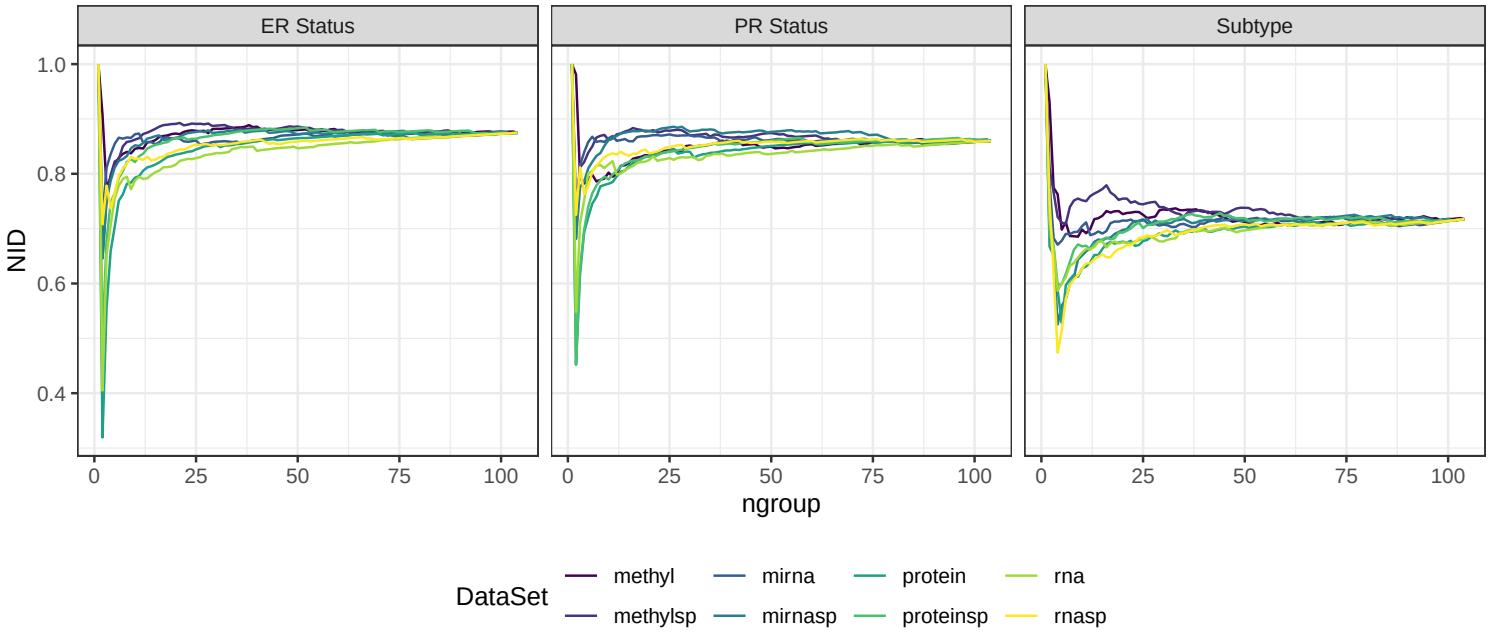
7.3 Fonction figure

```
plot_method = function(arbres_liste){
  nids_ER_status <- map_df(arbres_liste, get_nid, clinical$ER_status) %>%
  add_column(ngroup = seq.int(1:cutree_index_max)) %>% gather(key = "DataSet", value = "NID", -ngroup) %>%
  add_column(clinical = "ER Status")
nids_PR_status <- map_df(arbres_liste, get_nid, clinical$PR_status) %>%
  add_column(ngroup = seq.int(1:cutree_index_max)) %>% gather(key = "DataSet", value = "NID", -ngroup) %>%
  add_column(clinical = "PR Status")
nids_subtype <- map_df(arbres_liste, get_nid, clinical$subtype) %>%
  add_column(ngroup = seq.int(1:cutree_index_max)) %>% gather(key = "DataSet", value = "NID", -ngroup) %>%
  add_column(clinical = "Subtype")
nids <- rbind(nids_ER_status, nids_PR_status, nids_subtype)

nids %>% group_by(DataSet) %>%
  ggplot(aes(x = ngroup, y = NID, color = DataSet))  + geom_line() + facet_grid(.~clinical) + theme_bw() + theme
   scale_color_viridis(discrete = TRUE) -> plot_data

return(list(nids = nids, plot_data = plot_data))
}
```

```
res = plot_method(hc_list)
print(res$plot_data)
```



7.3.1 Meilleur NID possible et nombre de groupes associé

```
nids_df = as.data.frame(res$nids)
mat_res_best_nids = matrix(NA, ncol = 6, nrow = length(hc_list))
compteur_clinique = 1
for(clinique in unique(nids_df$clinical)){
  compteur_methode = 1
  for(dataset in unique(nids_df$DataSet)){
    subset_df = nids_df[nids_df$DataSet==dataset & nids_df$clinical==clinique,]
    mat_res_best_nids[compteur_methode, compteur_clinique:(compteur_clinique+1)] = c(subset_df$ngroup[which.min(
    compteur_methode = compteur_methode + 1
  }
  compteur_clinique = compteur_clinique+2
}
rownames(mat_res_best_nids) = names(hc_list)
mat_res_sp_data = mat_res_best_nids
```

```
knitr::kable(round(mat_res_best_nids,2), "latex",booktabs = T, escape = TRUE, linesep = "", row.names = TRUE)%>%
  add_header_above(c("", rep(c("Nb groupes", "NID"), 3)))%>%
  add_header_above(c("", "ER status" = 2, "PR status" = 2, "Subtype" = 2))%>%
  kable_styling(latex_options =c("repeat_header"), font_size = 10)
```

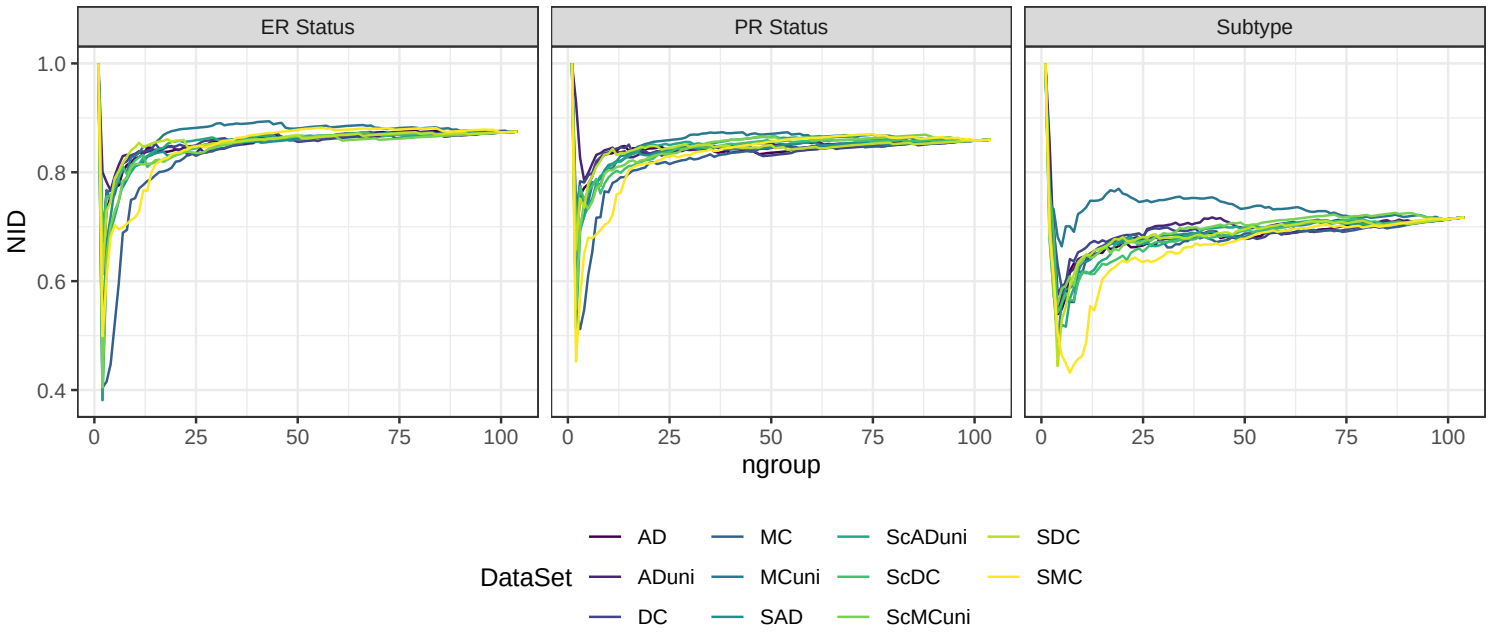
	ER status		PR status		Subtype	
	Nb groupes	NID	Nb groupes	NID	Nb groupes	NID
methyl	3	0.77	4	0.78	9	0.69
mirna	2	0.72	2	0.71	4	0.67
protein	2	0.32	2	0.45	5	0.53
rna	2	0.40	2	0.55	4	0.59
methylsp	2	0.65	2	0.74	6	0.71
mirnasp	2	0.65	2	0.68	4	0.53
proteinsp	2	0.50	2	0.45	4	0.60
rnasp	2	0.71	2	0.73	4	0.47

7.4 Méthodes d'agrégation

7.4.1 Figures (code Julien)

7.4.1.1 Toutes les méthodes ensemble

```
res = plot_method(hc_list_methods)
print(res$plot_data)
```




```
nids_df = data.frame(res$nids)
nids_df$Method = nids_df$DataSet
mat_res_best_nids = matrix(NA, ncol = 6, nrow = length(hc_list_methods))
compteur_clinique = 1
for(clinique in unique(nids_df$clinical)){
  compteur_methode = 1
  for(method in unique(nids_df$Method)){
    subset_df = nids_df[nids_df$Method==method & nids_df$clinical==clinique,]
    mat_res_best_nids[compteur_methode, compteur_clinique:(compteur_clinique+1)] = c(subset_df$ngroup[which.min(
    compteur_methode = compteur_methode + 1
  }
  compteur_clinique = compteur_clinique+2
}
rownames(mat_res_best_nids) = unique(nids_df$Method)
mat_res_methods = mat_res_best_nids
```

```
knitr::kable(round(mat_res_best_nids,2), "latex",booktabs = T, escape = TRUE, linesep = "", row.names = TRUE)%>%
  add_header_above(c("", rep(c("Nb groupes", "NID"), 3)))%>%
  add_header_above(c("", "ER status" = 2, "PR status" = 2, "Subtype" = 2))%>%
  kable_styling(latex_options =c("repeat_header"), font_size = 10)
```

	ER status		PR status		Subtype	
	Nb groupes	NID	Nb groupes	NID	Nb groupes	NID
AD	2	0.61	2	0.66	4	0.54
DC	2	0.68	2	0.70	4	0.57
MC	2	0.40	3	0.51	8	0.56
SDC	2	0.61	2	0.66	4	0.44
SAD	2	0.50	2	0.59	4	0.49
SMC	2	0.50	2	0.45	7	0.43
ADuni	4	0.77	4	0.79	4	0.54
MCuni	2	0.66	2	0.73	5	0.66
ScDC	2	0.68	2	0.67	4	0.54
ScADuni	2	0.38	2	0.51	4	0.49
ScMCuni	2	0.40	2	0.55	4	0.56