

# EI ST4 Big Data & Santé - Analyse et Intégration de données du projet « Milieu Intérieur » coordonné par l’Institut Pasteur

Alexis-Raja BRACHET, Eudoxie SYLVESTRE, Yvan THOREL, Tony WU, Claire ZHAO

2020-06-14

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Prise en main et exploration des données mises à disposition</b>	<b>2</b>
2.1	Figure synthétique des relations entre les caractéristiques des patients de la cohorte . . . . .	2
2.2	Influence du tabagisme sur les populations de cellules . . . . .	3
2.3	Corrélation des variables d'étude . . . . .	9
2.4	Decision Tree . . . . .	10
<b>3</b>	<b>Réduction de dimensions</b>	<b>14</b>
3.1	Analyse en composantes principales . . . . .	14
3.2	ACP en 3D . . . . .	20
3.3	Habillage avec l'eCRF . . . . .	21
3.4	t-SNE . . . . .	28
<b>4</b>	<b>Identification de gènes spécifiquement exprimés dans la réponse à un stimulus</b>	<b>30</b>
4.1	Identification de gènes par leur intensité d'expression . . . . .	30
4.2	Identification de gènes par leur différence d'expression . . . . .	39
<b>5</b>	<b>Recherche d'individus présentant des réponses immunitaires similaires pour plusieurs stimuli</b>	<b>59</b>
5.1	Réalisation d'une classification hiérarchique ascendante sur les individus pour chaque stimulation choisie . . . . .	59
5.2	Recherche des intersections entre les clusters à l'aide du package UpSetR . . . . .	62
<b>6</b>	<b>Comparaison des variances expliquées par les critères de l'eCRF pour les populations cellulaires</b>	<b>64</b>
<b>7</b>	<b>Association des niveaux d'ARNm avec les données de l'eCRF</b>	<b>66</b>
7.1	Régression linéaire pour expliquer le niveau d'ARNm des gènes et visualisation des p-valeurs	66
7.2	Variance des niveaux d'ARNm expliquée par les critères de l'eCRF et la composition en cellules CD45, pour stimuli variables . . . . .	69
7.3	Variance expliquée pour l'expression des gènes . . . . .	75
7.4	Random Forest . . . . .	78
<b>8</b>	<b>Conclusion</b>	<b>80</b>

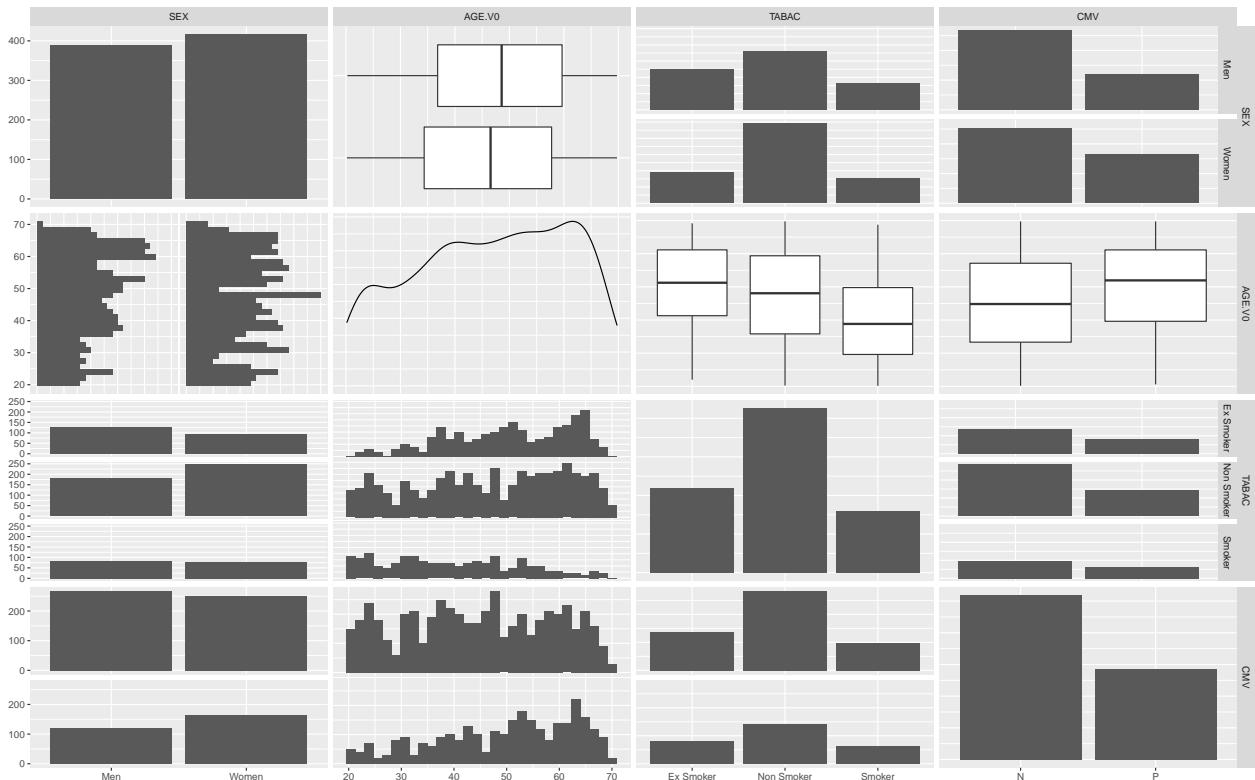
# 1 Introduction

Ce rapport a pour but de vous présenter le travail réalisé sur une semaine, en collaboration avec l’Institut Pasteur. Les données mises à notre disposition ont été récupérées à l’issu du projet “Milieu Intérieur”. Ce projet, impliquant des centaines de sujets test, regroupe des données issues de questionnaires remplis par ces derniers, et des données d’expression des cellules et des 560 gènes impliqués dans la réponse immunitaire, en réponse à 5 stimuli, notamment le virus de la grippe (Influenza), le vaccin contre la tuberculose (BCG) et le staphylocoque doré (*S.aureus*). Notre but au cours de cette semaine a été de mettre en pratique nos connaissances théoriques en statistique sur un sujet concret: la réponse immunitaire du corps humain. Ainsi, à l’aide de nos connaissances, nous avons analysé le grand nombre de données dont nous disposions afin d’identifier des facteurs influant la réponse immunitaire, des groupes de gènes caractérisant un stimulus donné...

## 2 Prise en main et exploration des données mises à disposition

### 2.1 Figure synthétique des relations entre les caractéristiques des patients de la cohorte

```
ggpairs(patients[, -c(1,4)]) # on retire SUBJID et la colonne AGEVAT (car redondant avec AGE.V0)
```



En regardant les données sur la diagonale du graphique, on observe que la cohorte est telle que : - il y a à peu près autant de femmes que d’hommes dans la cohorte - les personnes âgées sont en plus grand nombre que les jeunes - il y a plus environ 2 fois plus de personnes non-fumeurs que de personnes ayant déjà été ou qui sont actuellement fumeurs - il y a plus environ 2 fois plus de personnes ayant déjà contracté le CMV que de personnes ne l’ayant jamais eu.

## 2.2 Influence du tabagisme sur les populations de cellules

```
l.cells <- names(facs)[-1] # on enlève la 1ère colonne qui est SUBJID
```

On initialise df puis on sépare df selon la consommation de tabac du patient et on en profite pour enlever les colonnes ‘SUBJID’ et ‘TABAC’ qui ne sont plus utiles pour la suite.

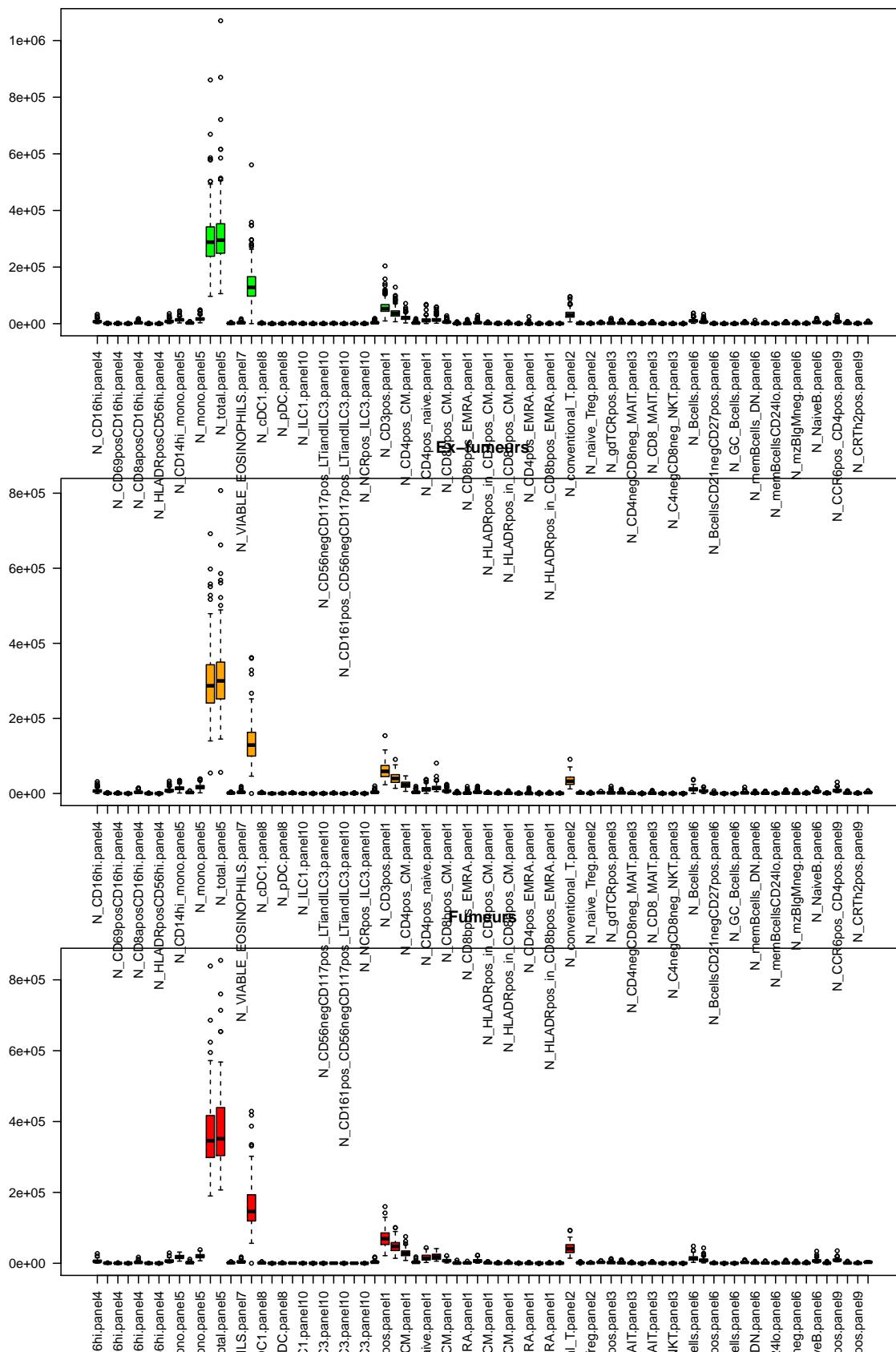
```
df <- merge(x=patients[c('SUBJID', 'TABAC')], y=facs, by='SUBJID')
df <- df[complete.cases(df), ]

df_smoker <- df[df$TABAC == 'Smoker', -(1:2)]
df_non_smoker <- df[df$TABAC == 'Non Smoker', -(1:2)]
df_ex_smoker <- df[df$TABAC == 'Ex Smoker', -(1:2)]
```

On trace maintenant 3 boxplot les uns au-dessus des autres pour avoir une vision d’ensemble de l’influence du tabagisme sur les populations cellulaires d’étude.

```
par(mfrow=c(3, 1))
boxplot(df_non_smoker, las=2, add=FALSE, col='green', main='Non-fumeurs')
boxplot(df_ex_smoker, las=2, add=FALSE, col='orange', main='Ex-fumeurs')
boxplot(df_smoker, las=2, col='red', main='Fumeurs', main='Fumeurs')
```

### **Non-fumeurs**

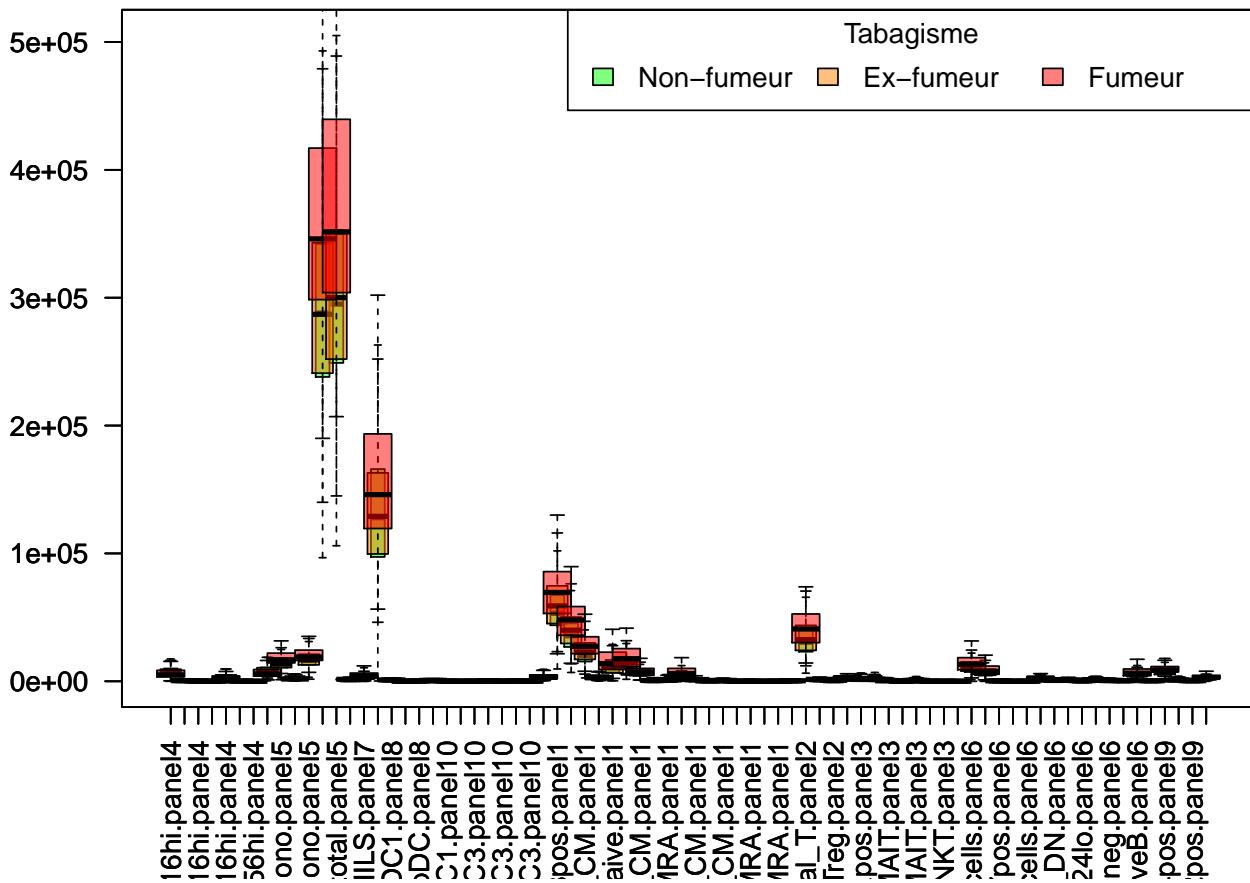


Comme la figure est un peu grande, on va maintenant superposer les 3 graphiques.

```
boxplot(df_non_smoker, las=2, add=FALSE, boxwex=1, col= rgb(0, 1, 0, alpha=0.5),
        outline = FALSE, main='Comparaison des quantités de cellules en fonction
        du tabagisme du patient')
boxplot(df_ex_smoker, las=2, add=TRUE, boxwex=1.5,
        col= rgb(1, 0.5, 0, alpha=0.5), outline = FALSE)
boxplot(df_smoker, las=2, add=TRUE, boxwex=2,
        col=rgb(1, 0, 0, alpha=0.5), outline = FALSE)

legend("topright", title="Tabagisme", c("Non-fumeur","Ex-fumeur","Fumeur"),
       fill=c(rgb(0, 1, 0, alpha=0.5), rgb(1, 0.5, 0, alpha=0.5),
              rgb(1, 0, 0, alpha=0.5)), horiz=TRUE, cex=1)
```

**Comparaison des quantités de cellules en fonction  
du tabagisme du patient**



Pour pouvoir étudier plus finement les données du graphique, du fait des grandes différences d'ordre de grandeur selon les populations de cellules, nous allons regrouper celles "nombreuses" et celles qui le sont moins pour essayer d'y voir plus clair.

On va donc créer des listes d'index regroupant ces 2 sous-ensembles de population.

```
l.means <- colMeans(df_smoker)

l.huge <- c()
l.rest <- c()
```

```

idx = 0

for (mean.val in l.means){
  idx <- idx + 1
  if (mean.val > 1e+04){
    l.huge <- c(l.huge, idx)
  } else{
    l.rest <- c(l.rest, idx)
  }
}

df_non_smoker_huge <- df_non_smoker[, l.huge]
df_smoker_huge <- df_smoker[, l.huge]
df_ex_smoker_huge <- df_ex_smoker[, l.huge]

df_non_smoker_rest <- df_non_smoker[, l.rest]
df_smoker_rest <- df_smoker[, l.rest]
df_ex_smoker_rest <- df_ex_smoker[, l.rest]

```

Première figure pour les cellules qui sont en moyenne à un nombre supérieur à 1e+4 :

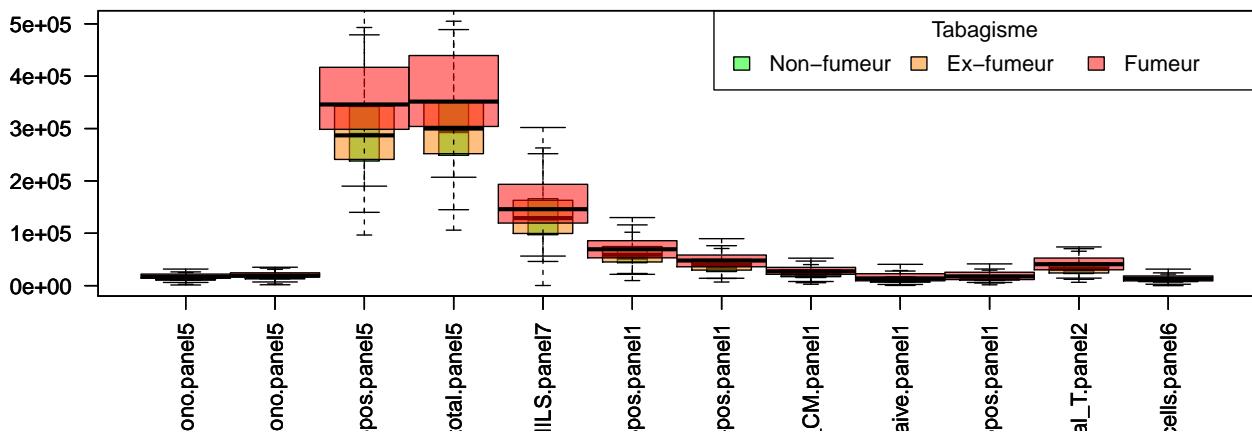
```

boxplot(df_non_smoker_huge, las=2, add=FALSE, boxwex=1/3,
        col= rgb(0, 1, 0, alpha=0.5), outline = FALSE,
        main='Comparaison des quantités de cellules en fonction du tabagisme
        du patient')
boxplot(df_ex_smoker_huge, las=2, add=TRUE, boxwex=2/3,
        col= rgb(1, 0.5, 0, alpha=0.5), outline = FALSE)
boxplot(df_smoker_huge, las=2, add=TRUE, boxwex=3/3, col=rgb(1, 0, 0, alpha=0.5),
        outline = FALSE)

legend("topright", title="Tabagisme", c("Non-fumeur", "Ex-fumeur", "Fumeur"),
       fill=c(rgb(0, 1, 0, alpha=0.5), rgb(1, 0.5, 0, alpha=0.5),
              rgb(1, 0, 0, alpha=0.5)), horiz=TRUE, cex=1)

```

**Comparaison des quantités de cellules en fonction du tabagisme  
du patient**



Deuxième figure pour les cellules qui sont en moyenne à un nombre inférieur à 1e+4 :

```

nb_cells_rest <- dim(df_smoker_rest)[2]

col_per_plot <- 5
nb_iter <- ceiling(ncol(df_smoker_rest) / col_per_plot)
par(mfrow=c(ceiling(nb_iter/2), 2))

for (i in 0:(nb_iter-1)){
  # Distinction de cas selon qu'on est ou non à la dernière itération
  if (i < (nb_iter - 1) & i!=0){
    l.idx <- (i*col_per_plot + 1):( (i+1)*col_per_plot)
  } else if(i == 0){
    l.idx <- 1:( (i+1)*col_per_plot)
  } else{
    l.idx <- (i*col_per_plot):(ncol(df_smoker_rest))
  }

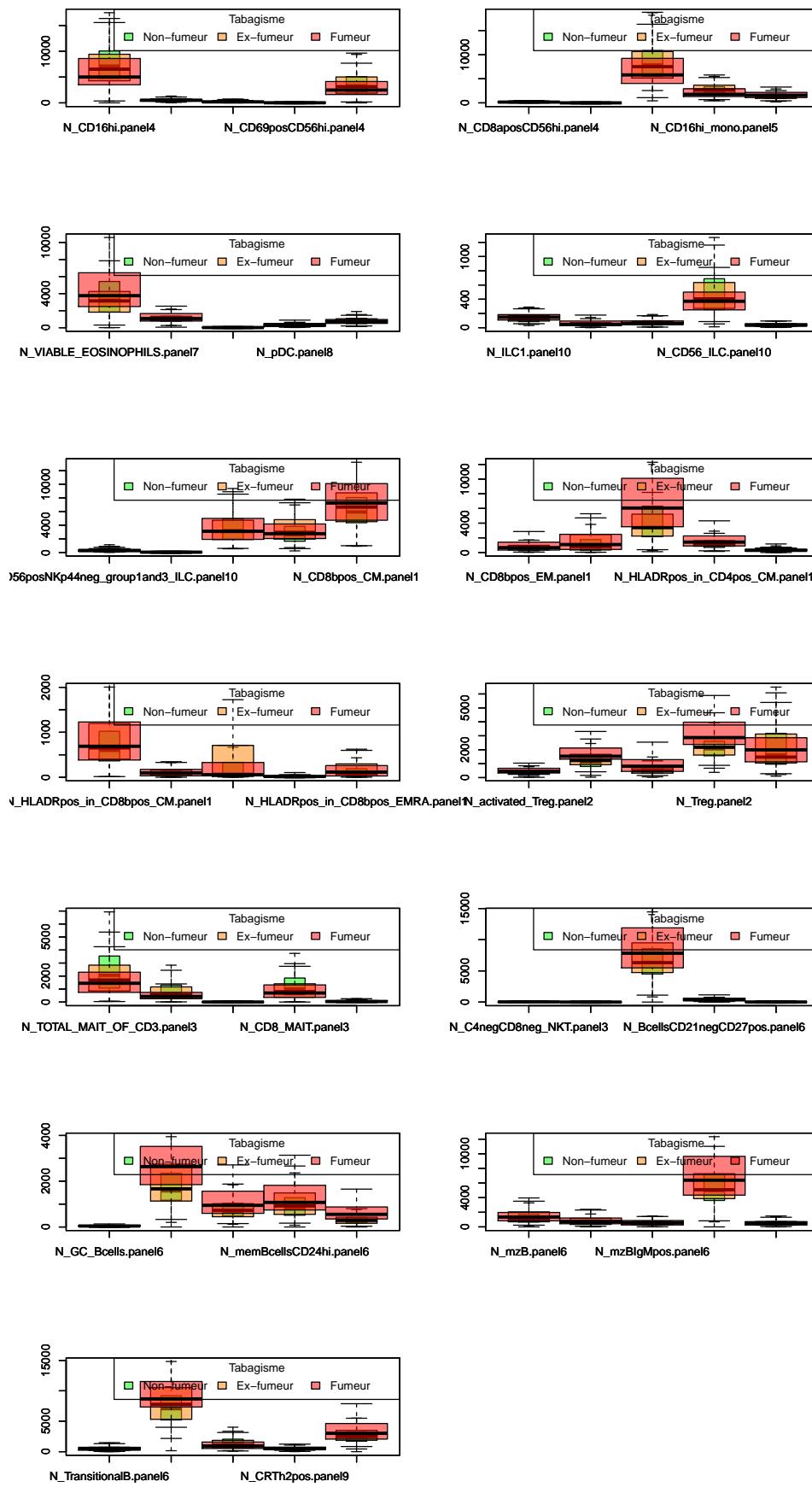
  #print(i)
  #print(l.idx)

  boxplot(df_non_smoker_rest[,l.idx], add=FALSE, boxwex=1/3,
          col= rgb(0, 1, 0, alpha=0.5), outline = FALSE)
  boxplot(df_ex_smoker_rest[,l.idx], add=TRUE, boxwex=2/3,
          col= rgb(1, 0.5, 0, alpha=0.5), outline = FALSE)
  boxplot(df_smoker_rest[,l.idx], add=TRUE, boxwex=3/3,
          col=rgb(1, 0, 0, alpha=0.5), outline = FALSE)

  legend("topright", title="Tabagisme", c("Non-fumeur","Ex-fumeur","Fumeur"),
         fill=c(rgb(0, 1, 0, alpha=0.5), rgb(1, 0.5, 0, alpha=0.5),
                rgb(1, 0, 0, alpha=0.5)), horiz=TRUE, cex=1)
}

mtext('Comparaison des quantités de cellules en fonction du tabagisme du patient',
      outer = TRUE, cex = 1.5)

```



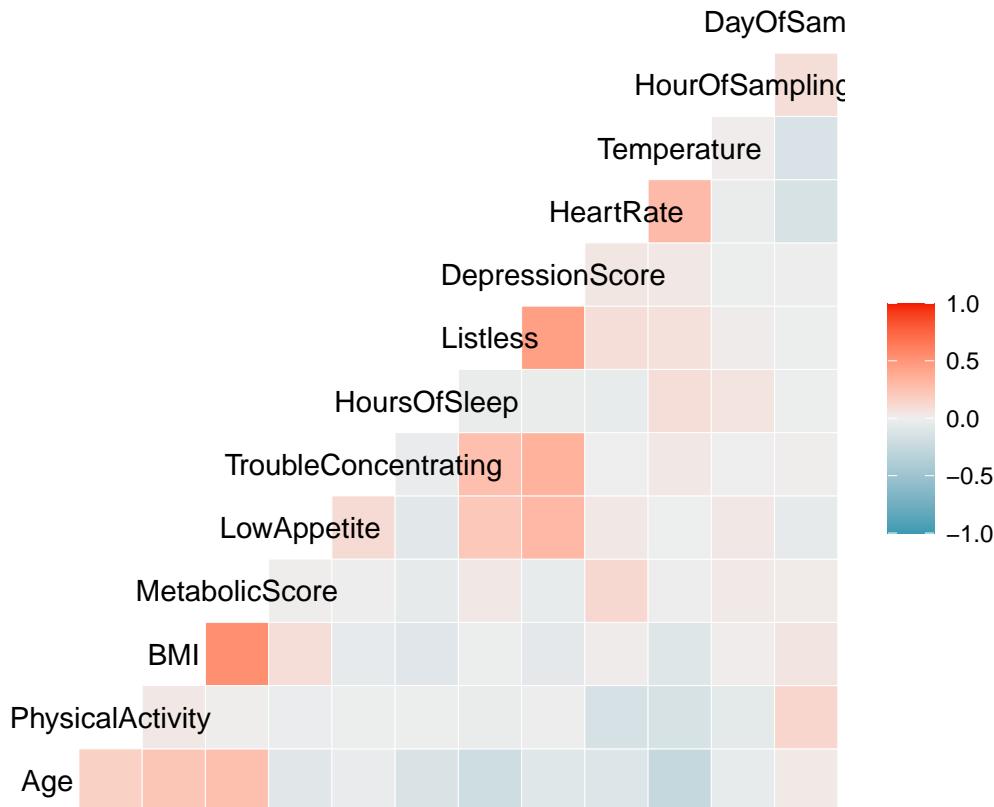
## 2.3 Corrélation des variables d'étude

On cherche à manipuler les données du questionnaire eCRF et à visualiser leur corrélation.

```
eCRF <- mutate(eCRF, OwnsHouse = (OwnsHouse == 'Yes'))
eCRF <- mutate(eCRF, LivesWithPartner = (LivesWithPartner == 'Yes'))
eCRF <- mutate(eCRF, LivesWithKids = (LivesWithKids == 'Yes'))
eCRF <- mutate(eCRF, BornInCity = (BornInCity == 'Yes'))
eCRF <- mutate(eCRF, CMVPositiveSerology = (CMVPositiveSerology == 'Yes'))
eCRF <- mutate(eCRF, UsesCannabis = (UsesCannabis == 'Yes'))
eCRF <- mutate(eCRF, RecentPersonalCrisis = (RecentPersonalCrisis == 'Yes'))
eCRF <- mutate(eCRF, Employed = (Employed == 'Yes'))
eCRF <- mutate(eCRF, HadMeasles = (HadMeasles == 'Yes'))
eCRF <- mutate(eCRF, HadRubella = (HadRubella == 'Yes'))
eCRF <- mutate(eCRF, HadChickenPox = (HadChickenPox == 'Yes'))
eCRF <- mutate(eCRF, HadMumps = (HadMumps == 'Yes'))
eCRF <- mutate(eCRF, HadTonsillectomy = (HadTonsillectomy == 'Yes'))
eCRF <- mutate(eCRF, HadAppendectomy = (HadAppendectomy == 'Yes'))
eCRF <- mutate(eCRF, VaccineHepA = (VaccineHepA == 'Yes'))
eCRF <- mutate(eCRF, VaccineMMR = (VaccineMMR == 'Yes'))
eCRF <- mutate(eCRF, VaccineTyphoid = (VaccineTyphoid == 'Yes'))
eCRF <- mutate(eCRF, VaccineWhoopingCough = (VaccineWhoopingCough == 'Yes'))
eCRF <- mutate(eCRF, VaccineYellowFever = (VaccineYellowFever == 'Yes'))
eCRF <- mutate(eCRF, VaccineHepB = (VaccineHepB == 'Yes'))
eCRF <- mutate(eCRF, VaccineFlu = (VaccineFlu == 'Yes'))

eCRF <- eCRF[,-c(35)] #on enlève les var qui ne conviennent pas (SUBJID, sexe...)
eCRF <- eCRF[,-c(21)]
eCRF <- eCRF[,-c(20)]
eCRF <- eCRF[,-c(19)]
eCRF <- eCRF[,-c(17)]
eCRF <- eCRF[,-c(4)]

#Visualisation de la corrélation
ggcorr(eCRF, method = c("everything", "pearson"))
```



On a ainsi un aperçu des corrélations entre les différentes caractéristiques, qui paraissent cohérentes. Certaines sont intéressantes : - La température et l'âge sont anti-correlés : la température corporelle diminue avec l'âge pour des raisons physiologiques (diminution de l'action de l'hypothalamus, glande dont le rôle est de maintenir la température du corps humain à 37°, métabolisme moins actif) - l'IMC et le score métabolique sont positivement corrélés - un score de dépression élevé est lié à une indolence/paresse élevée ('listless') et à une difficulté à se concentrer.

On réimporte eCRF comme on a précédemment écrit sur le DataFrame.

```
eCRF <- read.table(file=".~/data/eCRF.txt", header=TRUE, sep='\t', stringsAsFactors = FALSE)
```

## 2.4 Decision Tree

Encodage des données qualitatives

```
drop <- c("SUBJID") # drop contient la liste des colonnes à supprimer
df_1 <- eCRF[, !(names(eCRF) %in% drop)]
```

On stocke maintenant les index des colonnes contenant des variables qualitatives dans l'optique de les transformer avec une sorte de *One-Hot Encoding*.

```
1.idx_chr <- sapply(df_1, is.character)
```

Sorte de LabelEncoder de Scikit mais pour R. Pour des raisons de simplicité, pour savoir à quoi correspond chaque label, lancer `View(eCRF)` (données brutes) et `View(df_1)`(après encodage) et comparer.

**NE SURTOUT PAS ESSAYER DE DEVINER LES SENS DES ENCODAGES SANS CETTE ETAPPE PRELIMINAIRE !!**

```

for (i in 1:length(l.idx_chr)){
  if (l.idx_chr[i]==TRUE){ # Si la colonnes présente des variables qualitatives...
    # print(i)
    df_1[, i] <- LabelEncoder$new()$fit_transform(df_1[, i]) # Encodage
  }
}

```

**Construction de l'arbre de décision** On essaie de prédire MetabolicScore en fonction des autres données en indiquant à rpart la formule MetabolicScore ~ ..

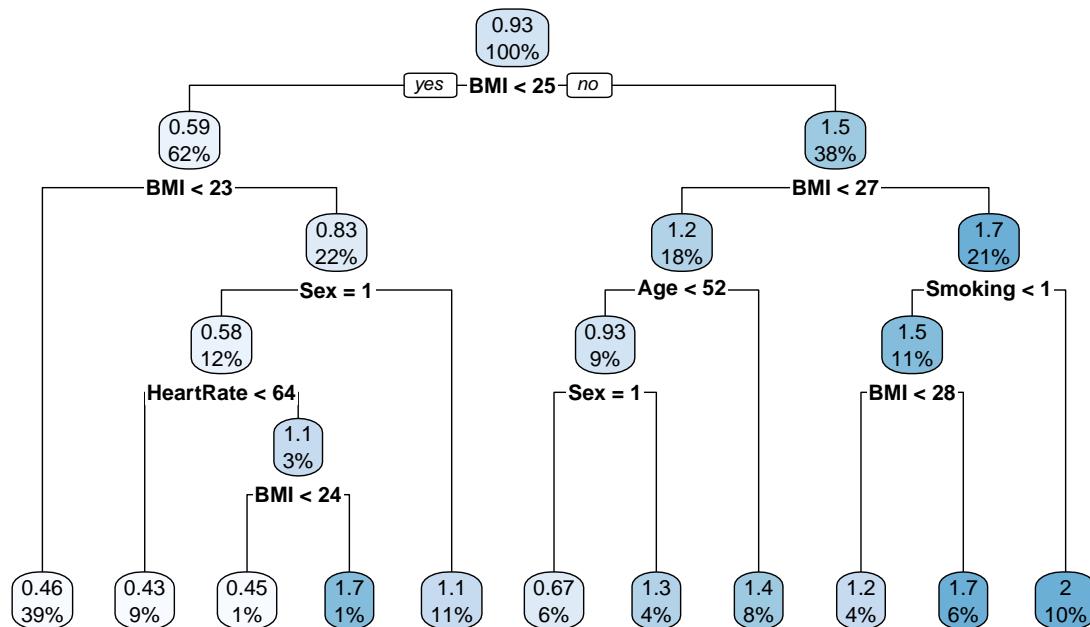
```

fit.tree <- rpart(
  MetabolicScore ~ .,
  data = df_1)

rpart.plot(fit.tree, main="Arbre de décision pour MetabolicScore (régression)")

```

## Arbre de décision pour MetabolicScore (régression)



**Interprétation :** On remarque que le premier critère influençant sur le *MetabolicScore* est le *BMI*.

Voici un extrait de [https://www.nhlbi.nih.gov/health/educational/lose\\_wt/BMI/bmicalc.htm](https://www.nhlbi.nih.gov/health/educational/lose_wt/BMI/bmicalc.htm) : BMI Categories: - Underweight = <18.5 - Normal weight = 18.5–24.9 - Overweight = 25–29.9 - Obesity = BMI of 30 or greater

Ceci n'est pas si surprenant car il est maintenant connu que le surpoids est une des principales causes de problèmes cardio-vasculaires. Ou plus directement, il est possible de se référer à *The Milieu Intérieur study — An integrative approach for study of human immunological variance* de Thomas et al. aux pages 288 et 289 pour voir que le *BMI* est étroitement lié au *MetabolicScore*.

On remarque ensuite que les 2 autres principaux facteurs sont l'*âge* et le *sexe*, caractéristiques qui ont déjà été explicitées dans les thèses fournies !

On fait de même pour d'autres features intéressantes.

**Pour HoursOfSleep :**

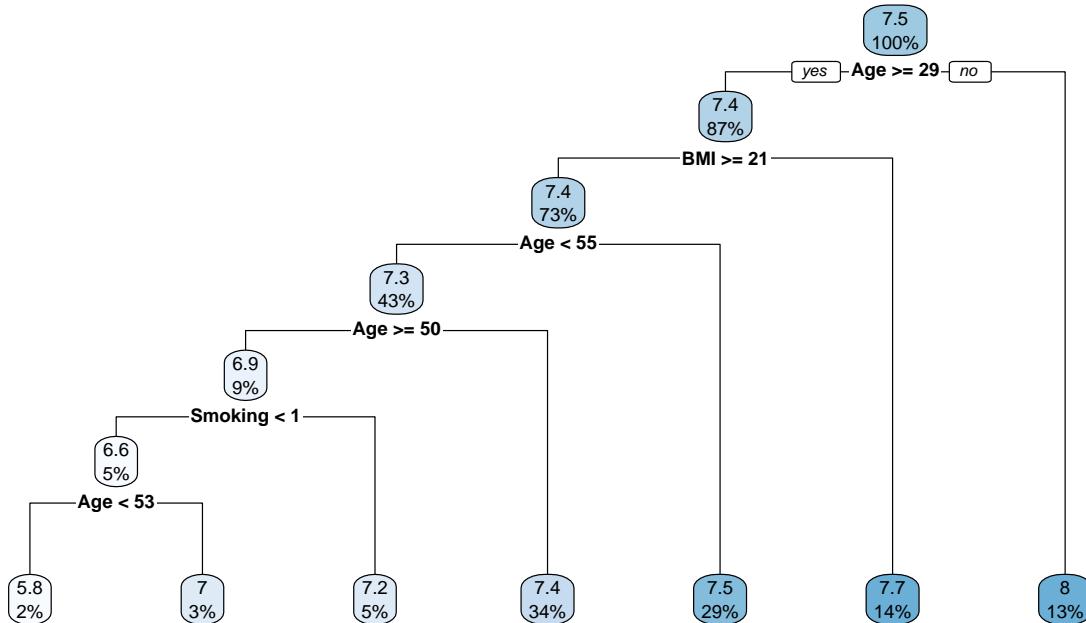
```

fit.tree <- rpart(
  HoursOfSleep ~ .,
  data = df_1)

rpart.plot(fit.tree, main="Arbre de décision pour HoursOfSleep (régression)")

```

### Arbre de décision pour HoursOfSleep (régression)



Pour DepressionScore :

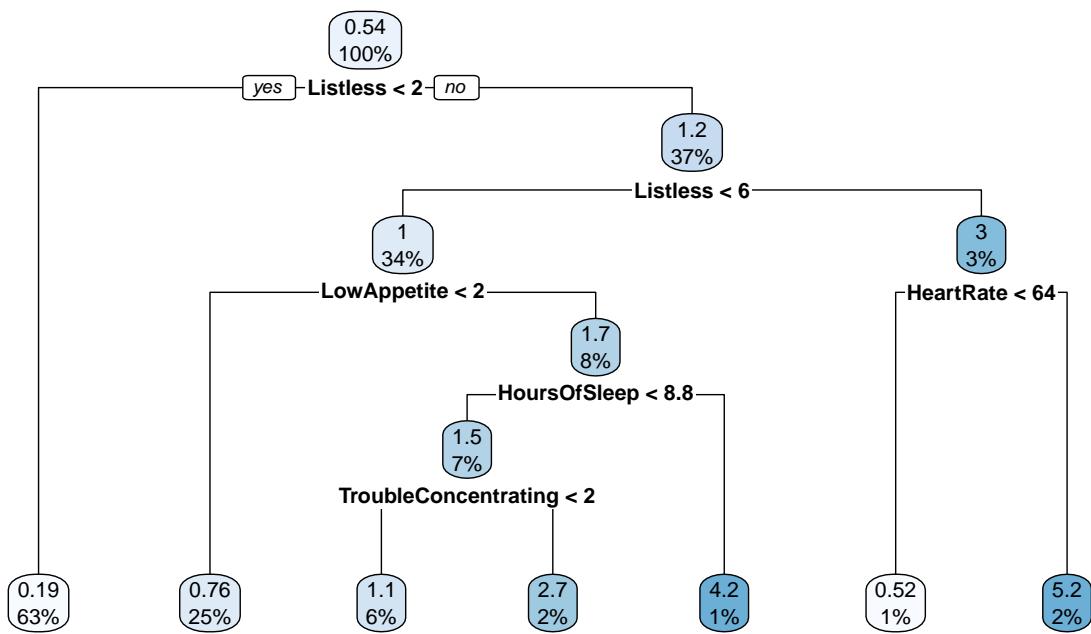
```

fit.tree <- rpart(
  DepressionScore ~ .,
  data = df_1)

rpart.plot(fit.tree, main="Arbre de décision pour DepressionScore (régression)")

```

## Arbre de décision pour DepressionScore (régression)



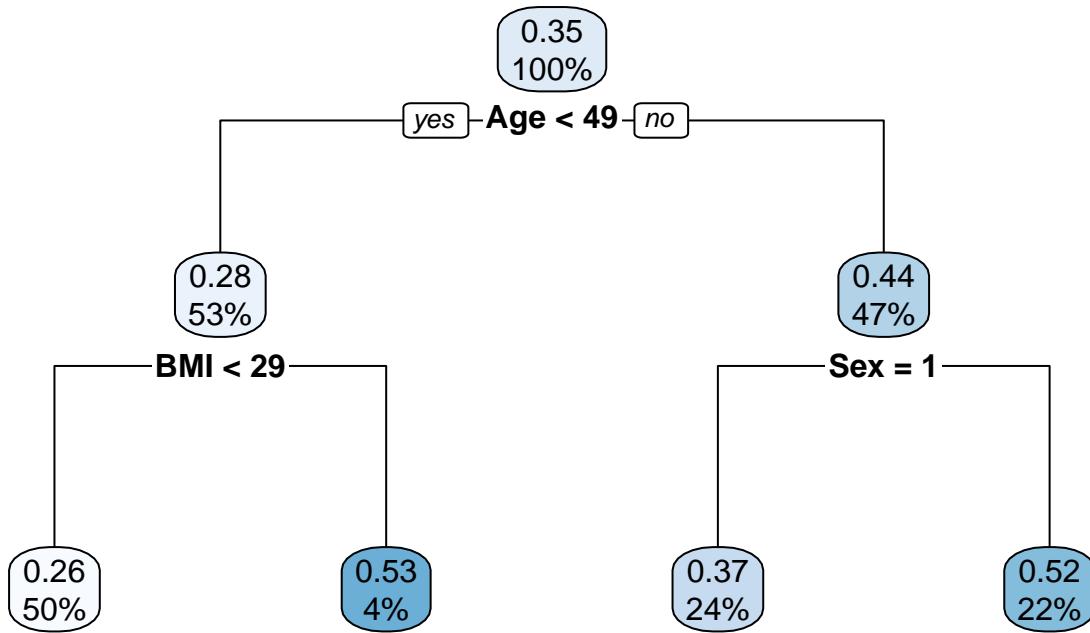
Pour CMVPositiveSerology :

```

fit.tree <- rpart(
  CMVPositiveSerology ~ .,
  data = df_1)

rpart.plot(fit.tree, main="Arbre de décision pour CMVPositiveSerology (classification)")
  
```

## Arbre de décision pour CMVPositiveSerology (classification)



**Interprétation :** Avant tout, on rappelle que le cytomégalovirus (ou CMV) est un virus responsable d'infections et donc dangereux. On remarque sur la figure ci-dessus, l'âge est le premier facteur contribuant à l'apparition du CMV. Néamoins, pour les moins de 50 ans, on peut critiquer la séparation obtenue car un BMI supérieur à 29 est synonyme d'obésité et il semble peu probable qu'être obèse rende moins susceptible de contracter le virus...

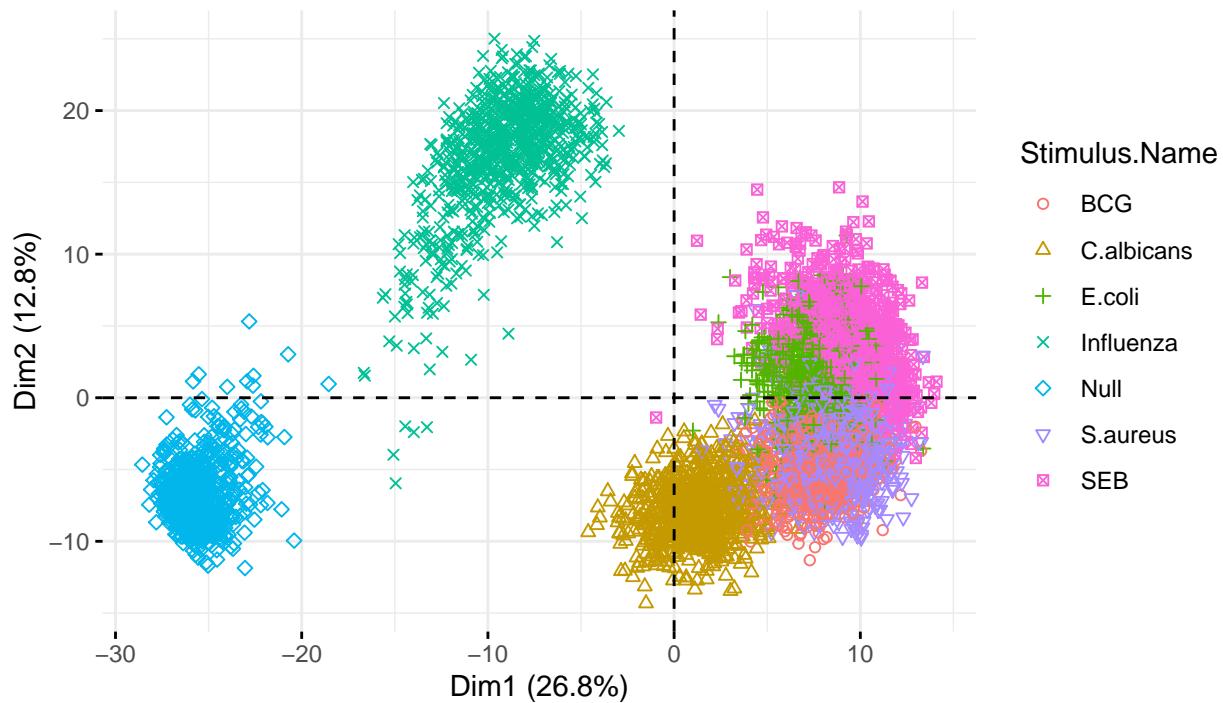
## 3 Réduction de dimensions

### 3.1 Analyse en composantes principales

```
respca = PCA(X=expr, scale.unit=TRUE, graph=F, ncp=5, quali.sup=1:2)

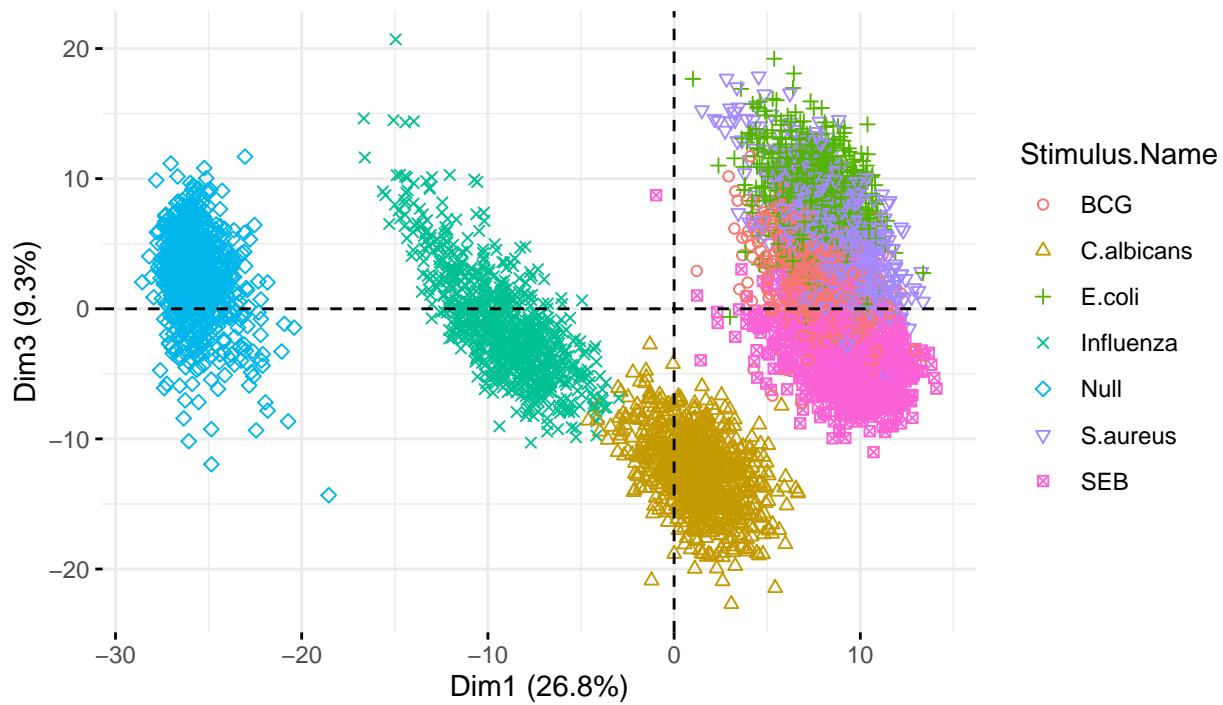
fviz_pca_ind(respca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label ="none")
```

Individuals – PCA

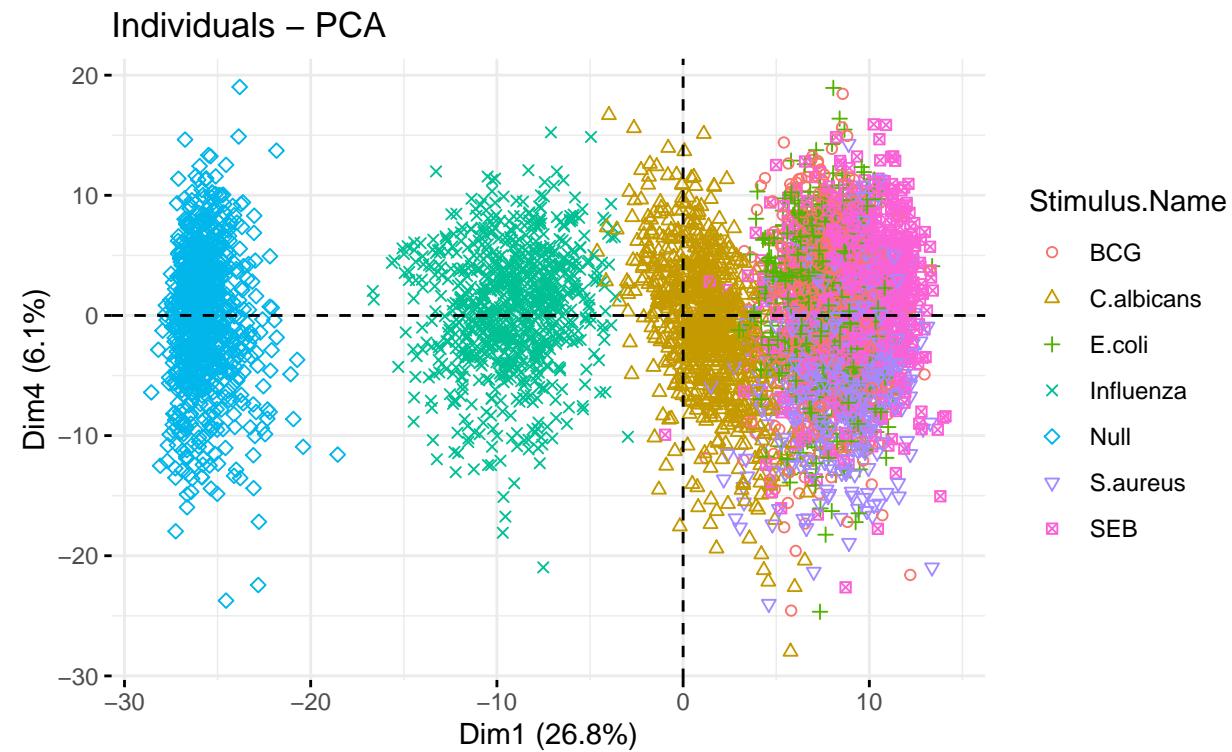


```
fviz_pca_ind(respca,
              axes = c(1,3),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label = "none")
```

Individuals – PCA



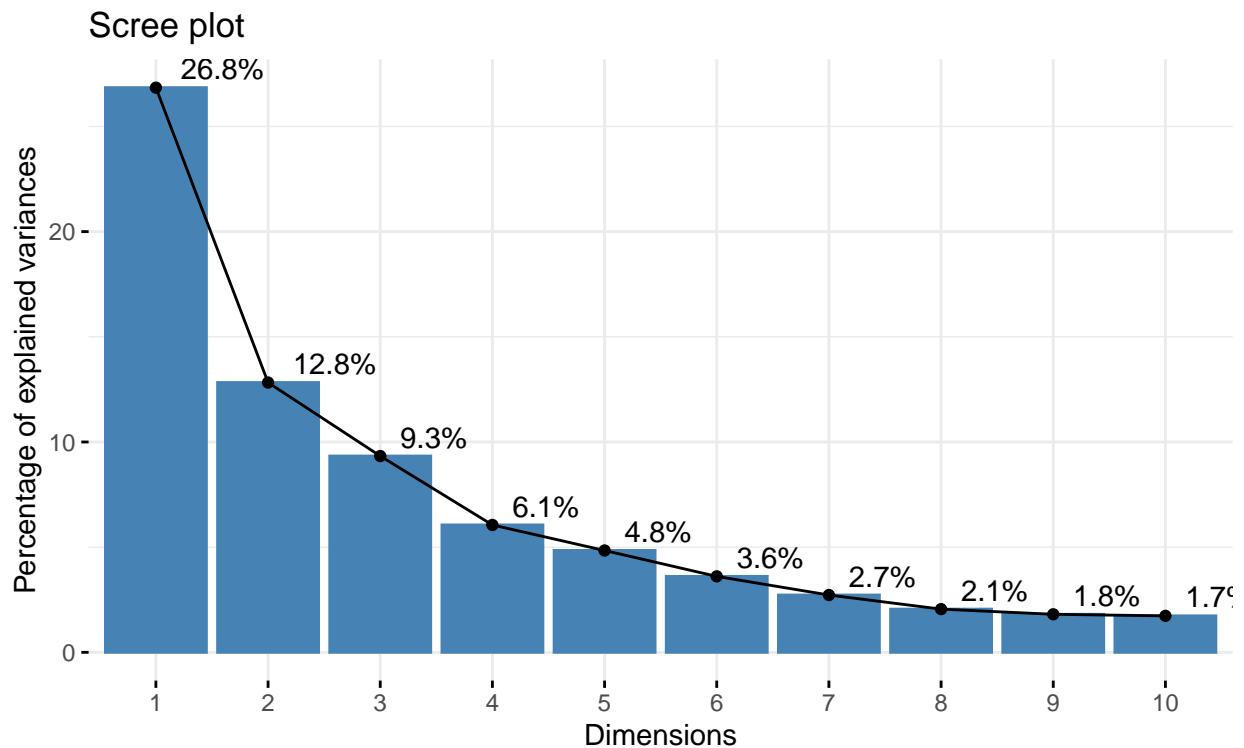
```
fviz_pca_ind(respca,
              axes = c(1,4),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label = "none")
```



L'analyse en composantes principales nous indique que c'est la stimulation à influenza qui engendre la réponse immunitaire la plus distincte. On voit aussi que l'absence de stimulation se distingue nettement sur l'axe 1. Tous ces résultats sont en accord avec l'ACP de Piasecka.

### 3.1.1 Pourcentage de variance capturée par chaque composante

```
fviz_screepplot(respca, addlabels=TRUE, hjust=-0.3)
```



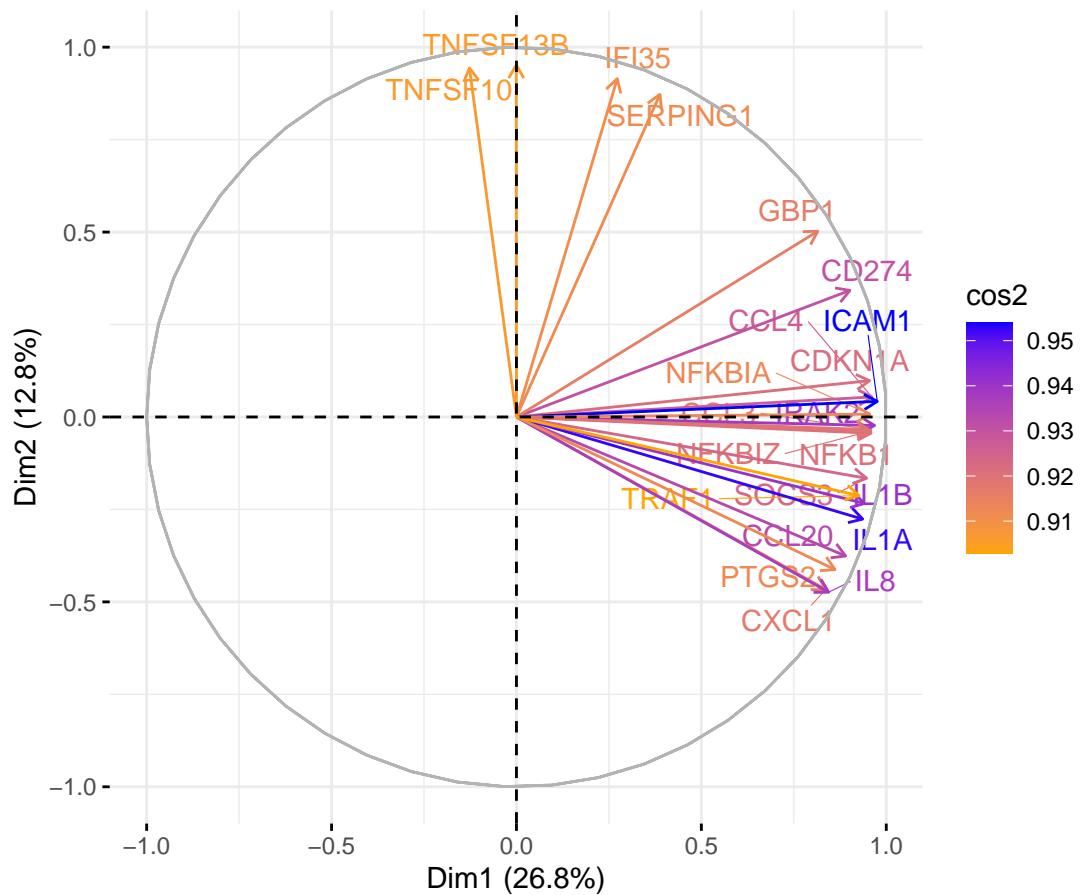
La 3ème composante capture près de 10% de la variance, il est donc mieux de s'intéresser aux trois premières composantes.

### 3.1.2 Cercle des corrélations

On sélectionne les covariables les plus corrélées avec les deux premiers axes, c'est-à-dire de cosinus carré supérieur à 0.9.

```
fviz_pca_var(respca, axes = c(1,2), select.var = list(cos2 = 0.9), col.var="cos2", repel=TRUE, gradient=TRUE)
```

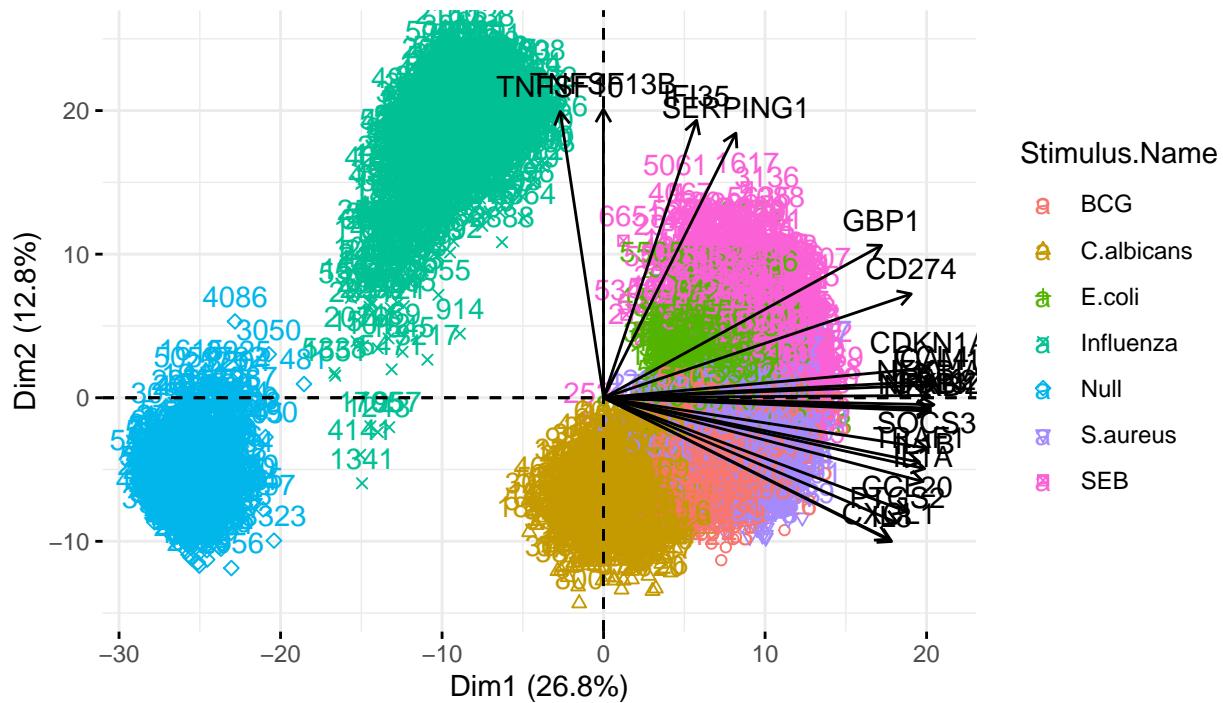
Variables – PCA



### 3.1.3 Biplot

```
fviz_pca_biplot(respca, axes = c(1, 2), habillage = "Stimulus.Name",
  select.var = list(cos2 = 0.9), col.var = "black",
  invisible = c("quali"), repel = FALSE)
```

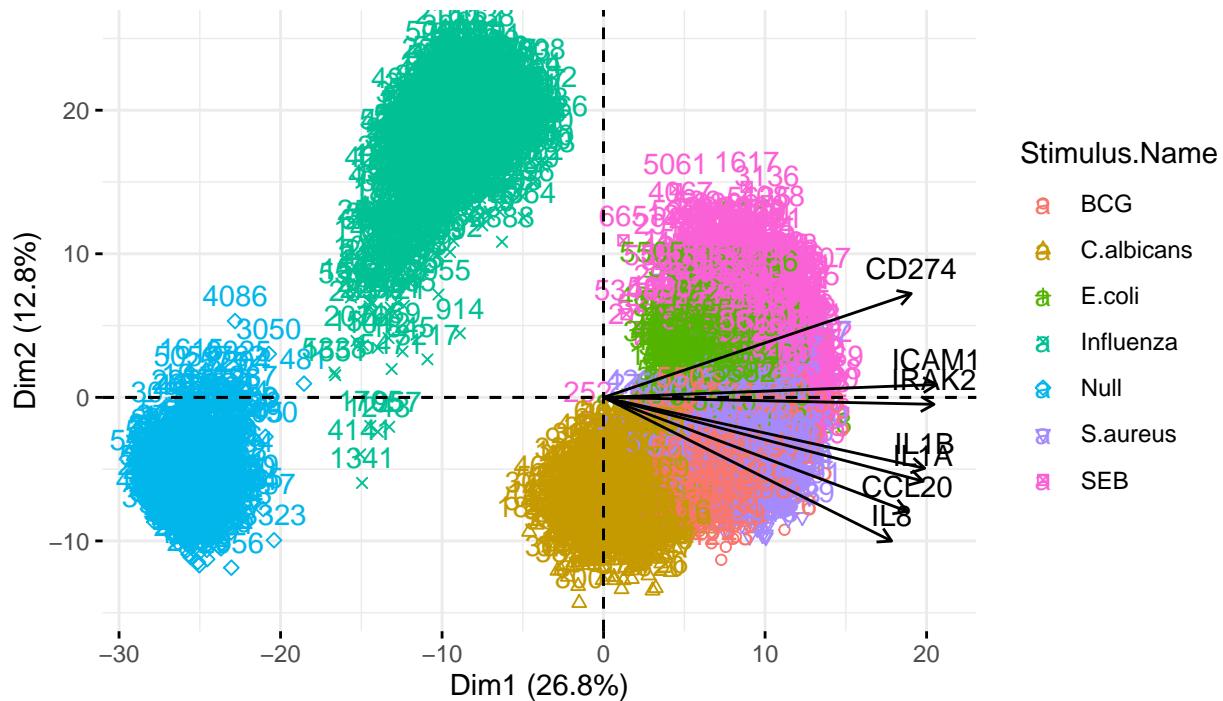
## PCA – Biplot



Le niveau d'ARNm du gène TNFSF10 semble fortement associé à la stimulation à l'influenza.

```
fviz_pca_biplot(respca, axes = c(1, 2), habillage = "Stimulus.Name",
                 select.var = list(cos2 = 0.93), col.var = "black", invisible = c("quali"))
```

## PCA – Biplot



Le niveau d'ARNm des gènes IL1A et IL1B semble fortement associé à la stimulation à S.aureus, et le niveau d'ARNm du gène CD274 semble fortement associé à la stimulation à SEB et E.coli.

## 3.2 ACP en 3D

```
# Code pour faire l'ACP  
acp = dudi.pca(expr[, -c(1, 2)], scannf= F, scale=FALSE, nf=3)
```

On trace le graphique 3D, avec les mêmes couleurs que celles utilisées dans l'article Piasecka. Pour accélérer la création du rapport, nous avons déjà enregistré le résultat obtenu que nous avons affiché ci-dessous.

```
#stimulus BCG  
plot3d(acp$li[, 1][expr$Stimulus.Name=="BCG"],  
       acp$li[, 2][expr$Stimulus.Name=="BCG"],  
       acp$li[, 3][expr$Stimulus.Name=="BCG"],  
       col="yellow",  
       radius=0.4,  
       type="p",  
       xlab="Dim1", ylab="Dim2", zlab="Dim3",  
       )  
  
#pas de stimulus IAV dans les données expr  
  
#Ajout stimulus C.albicans  
spheres3d(acp$li[, 1][expr$Stimulus.Name=="C.albicans"],  
          acp$li[, 2][expr$Stimulus.Name=="C.albicans"],  
          acp$li[, 3][expr$Stimulus.Name=="C.albicans"],  
          col="red",  
          radius=0.4)  
  
#Ajout stimulus E.coli  
spheres3d(acp$li[, 1][expr$Stimulus.Name=="E.coli"],  
          acp$li[, 2][expr$Stimulus.Name=="E.coli"],  
          acp$li[, 3][expr$Stimulus.Name=="E.coli"],  
          col="blue",  
          radius=0.4)  
  
#Ajout stimulus S.aureus  
spheres3d(acp$li[, 1][expr$Stimulus.Name=="S.aureus"],  
          acp$li[, 2][expr$Stimulus.Name=="S.aureus"],  
          acp$li[, 3][expr$Stimulus.Name=="S.aureus"],  
          col="green",  
          radius=0.4)  
  
#Ajout stimulus SEB  
spheres3d(acp$li[, 1][expr$Stimulus.Name=="SEB"],  
          acp$li[, 2][expr$Stimulus.Name=="SEB"],  
          acp$li[, 3][expr$Stimulus.Name=="SEB"],  
          col="purple",  
          radius=0.4)  
  
#Ajout stimulus NS  
spheres3d(acp$li[, 1][expr$Stimulus.Name=="Null"],  
          acp$li[, 2][expr$Stimulus.Name=="Null"],  
          acp$li[, 3][expr$Stimulus.Name=="Null"],
```

```

    col="grey",
    radius=0.4)

#Ajout légende

legend3d("topleft",
  legend = c("BCG", "C.albicans", "E.coli", "S.aureus", "SEB", "Non stimulé"),
  pch = 16,
  col = c("yellow", "red", "blue", "green", "purple", "grey"),
  cex=1,
  inset=c(0.02))

```

On voit bien 3 groupes distincts selon les stimulations d'origine virale (Influenza) et les 5 autres.

### 3.3 Habillage avec l'eCRF

#### 3.3.1 Nanostring

Il peut être intéressant, après avoir réalisé une réduction de dimension, d'habiller les données avec les données de l'eCRF afin de voir lesquelles engendrent une forte différence que ce soit dans l'expression des gènes stimulés, ou dans la composition cellulaire.

```
df <- merge(x=eCRF, y=expr, by.x ='SUBJID', by.y ='SUBJID')
```

On réalise une ACP sur les données de nanostring.

```
res.pca <- PCA(df, scale.unit=TRUE, graph=F, ncp=5, quali.sup=1:(ncol(eCRF)+1))
```

On colorie cet ACP selon chacun des critères de l'eCRF, et on affiche quelques résultats.

```

lcol = names(df)[2:40]
for(col in lcol){
  fviz_pca_ind(res.pca,
    axes = c(1,2),
    habillage=col,
    invisible = 'quali',
    label ="none")
  ggsave(paste(col, ".png", sep=""))
}

```

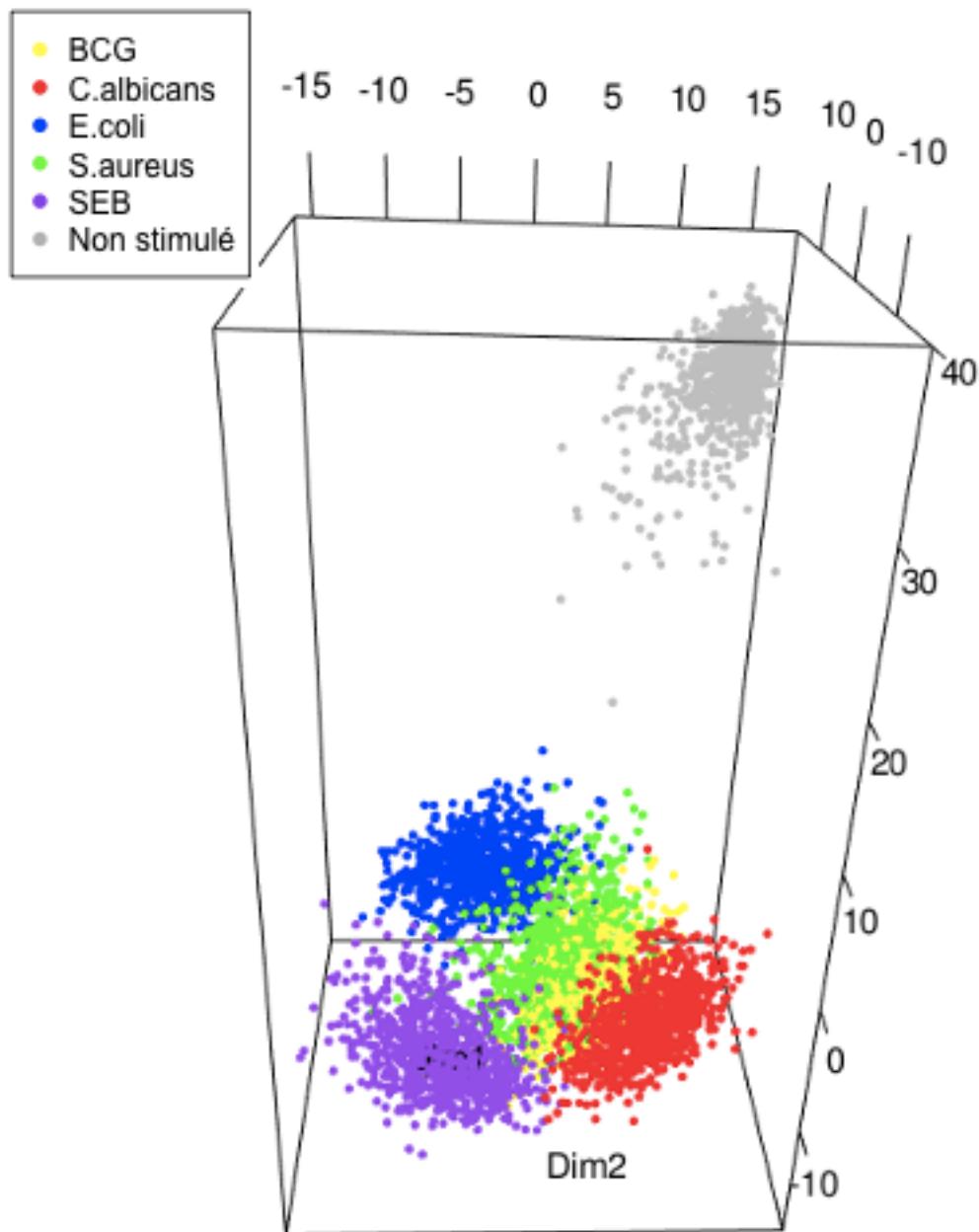
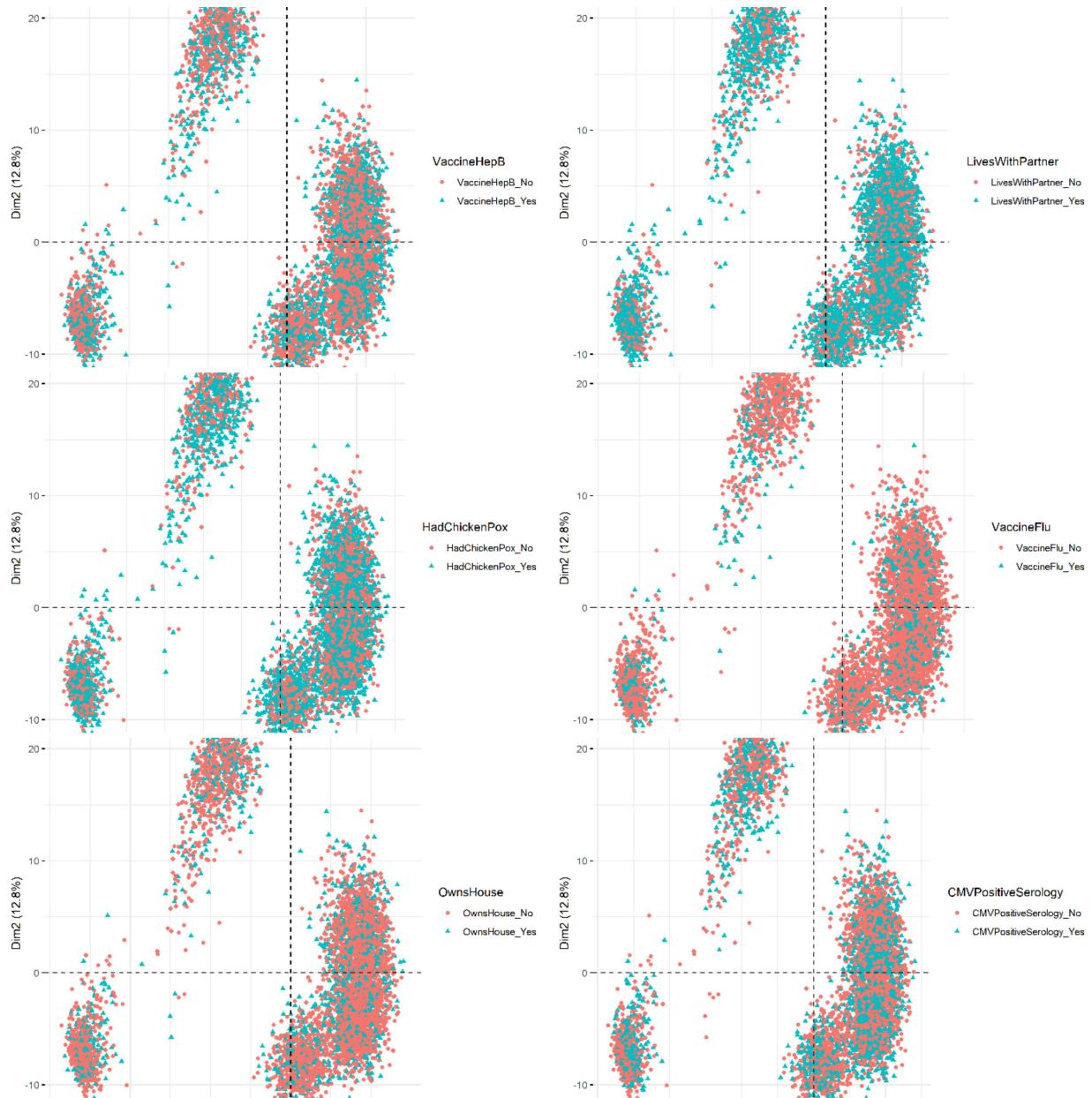


Figure 1: ACP sur les 3 premières composantes principales

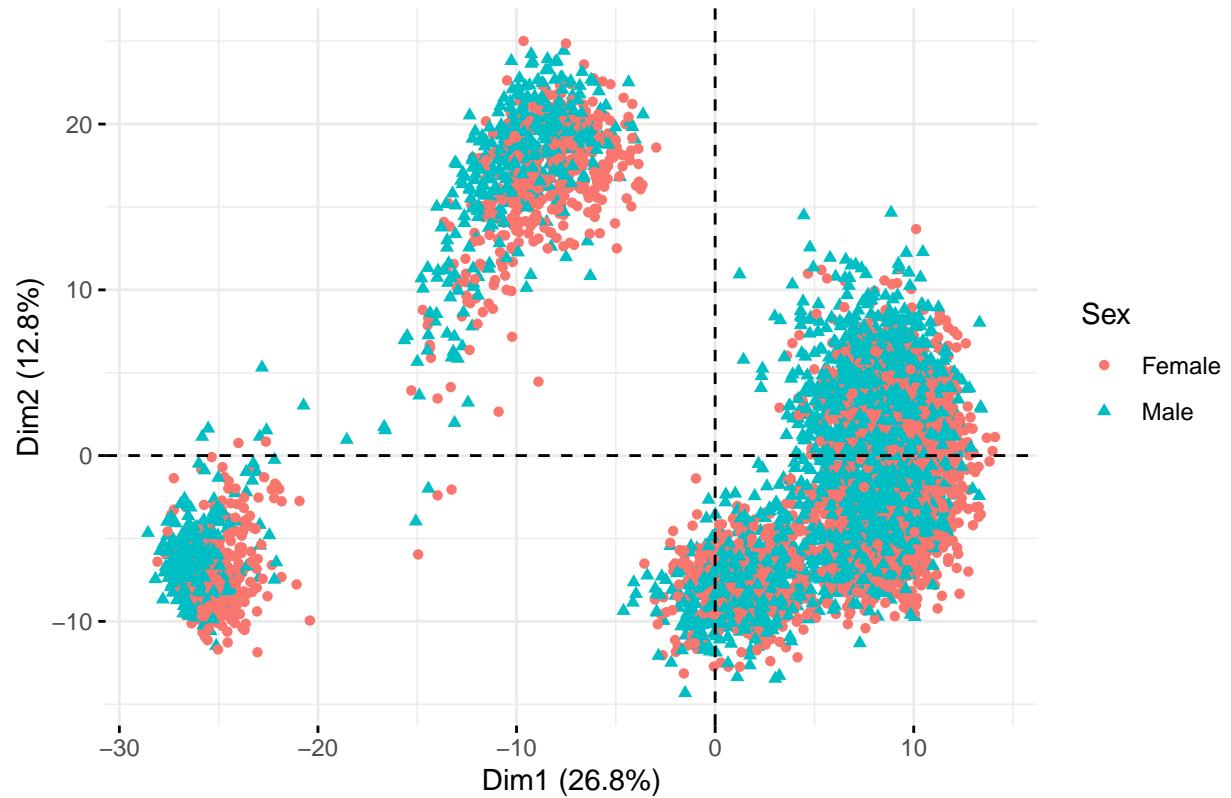


On remarque que pour la grande majorité des critères de l'eCRF, aucune différence n'est visible dans l'ACP. Cela nous indique que la plupart des critères, pris séparément, n'influe que peu sur la composition cellulaire.

On affiche le résultat pour le sexe, qui est un des seuls critères où la différence est marquée.

```
fviz_pca_ind(res.pca,
             axes = c(1,2),
             habillage='Sex',
             invisible = 'quali',
             label ="none")
```

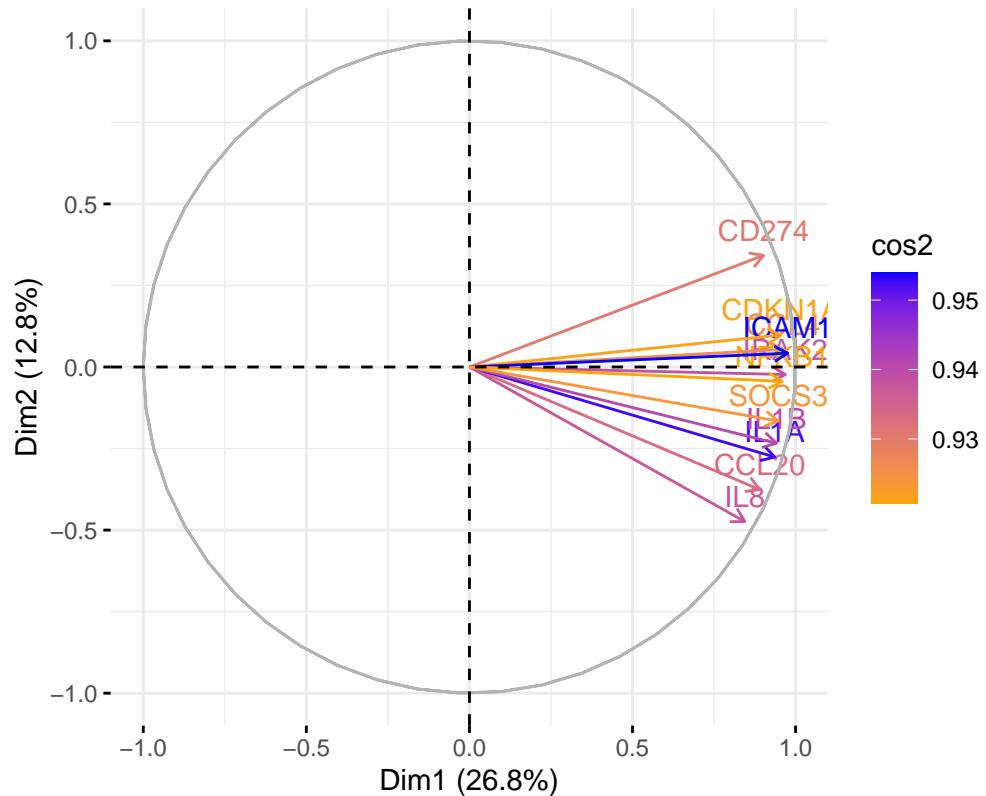
## Individuals – PCA



Il est intéressant de voir que l'expression des gènes varie de façon assez importante selon le sexe. On peut voir quels sont les gènes les plus fortement associés au sexe, ce sont les gènes orientés selon la diagonale descendante dans le cercle des corrélations :

```
fviz_pca_var(res.pca, axes = c(1,2), select.var = list(cos2 = 0.92),
             col.var="cos2", gradient.cols = c("orange", "blue"))
```

## Variables – PCA



L'expression du gène IL8 par exemple semble plus associée au sexe féminin.

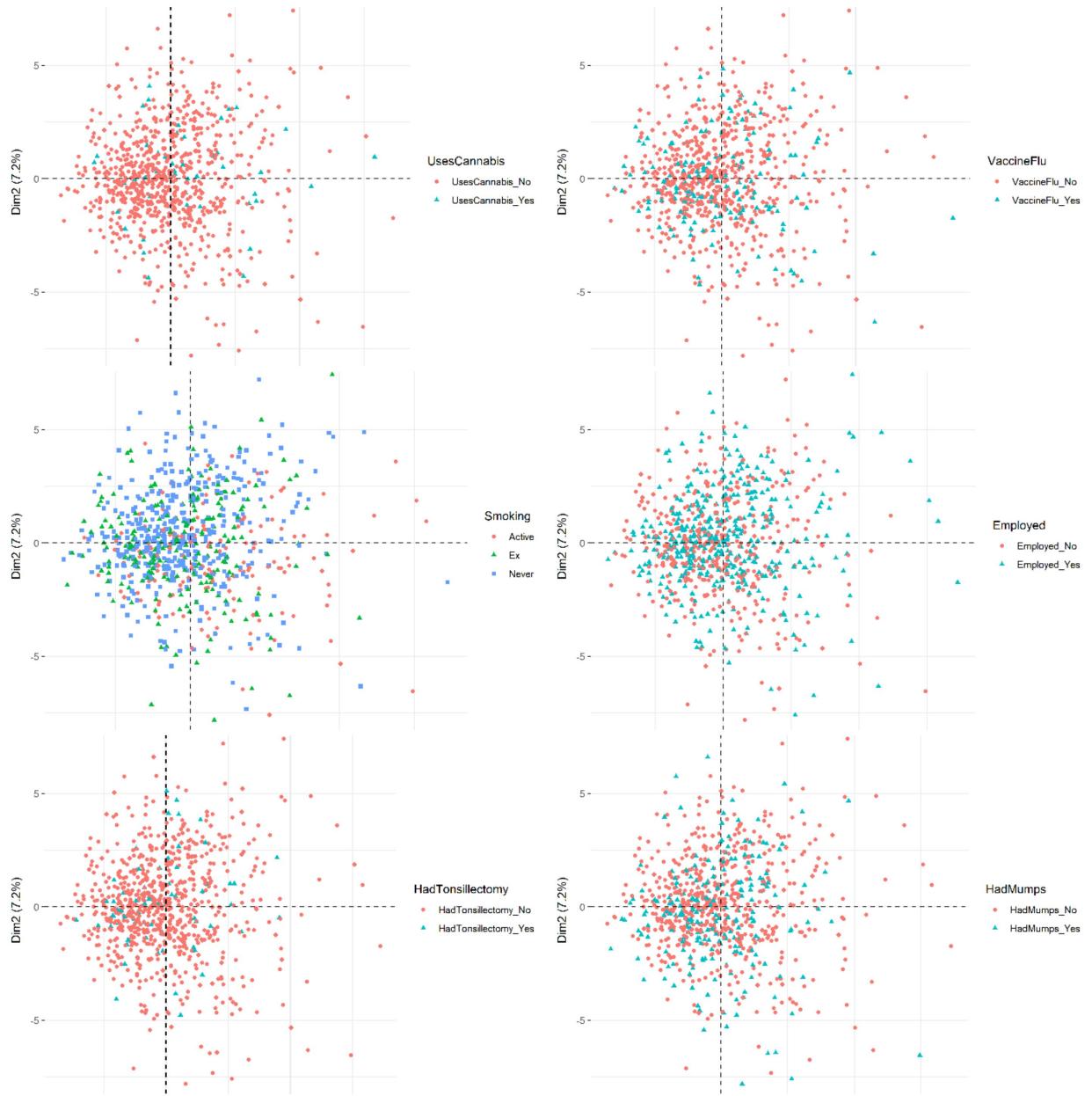
### 3.3.2 Facs

```
df <- merge(x=eCRF, y=facs, by='SUBJID')

res.pca <- PCA(df, scale.unit=TRUE, graph=F, ncp=5, quali.sup=1:40)
```

Dans le chunk suivant, nous avons coloré l'ACP précédente en fonction des différents critères de l'eCRF.

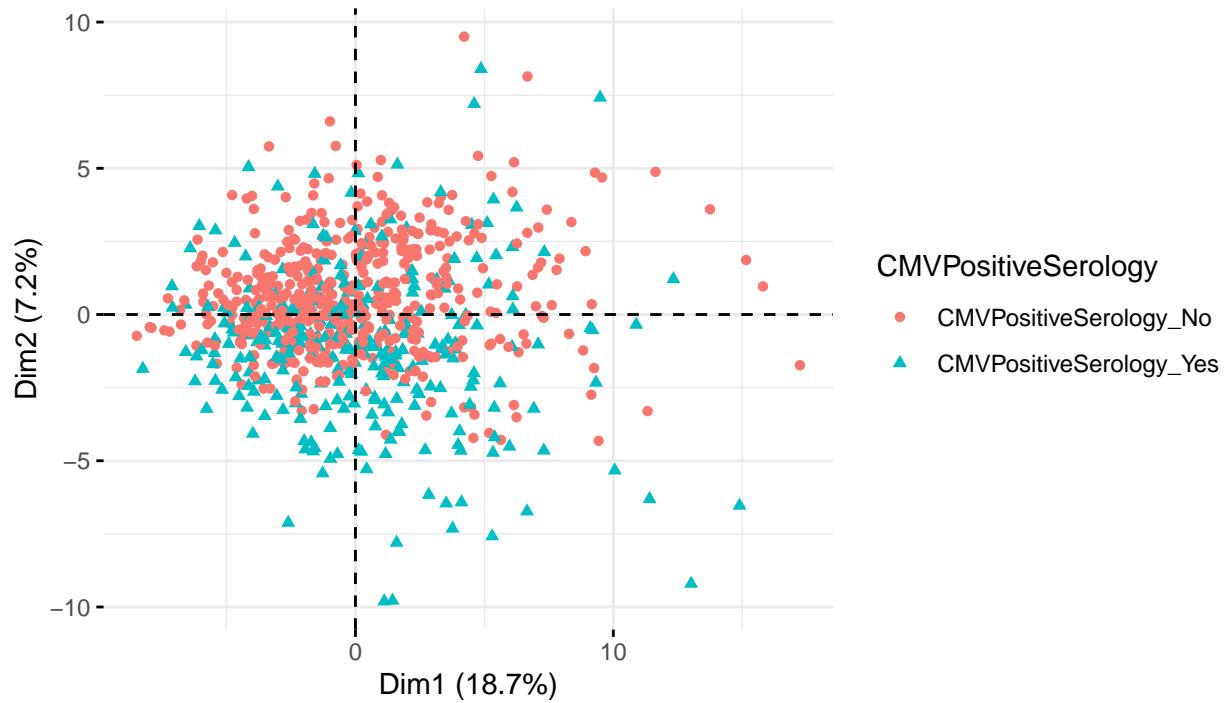
```
lcol = names(df)[2:40]
for(col in lcol){
  fviz_pca_ind(res.pca,
    axes = c(1,2),
    habillage=col,
    invisible = 'quali',
    label =none")
  ggsave(paste("PCA facs/", col, ".png", sep=''))
}
```



On remarque comme pour nanostring que peu de critères sont différenciés. La positivité au CMV l'est :

```
fviz_pca_ind(res.pca,
              axes = c(1,2),
              habillage='CMVPositiveSerology',
              invisible = 'quali',
              label ="none")
```

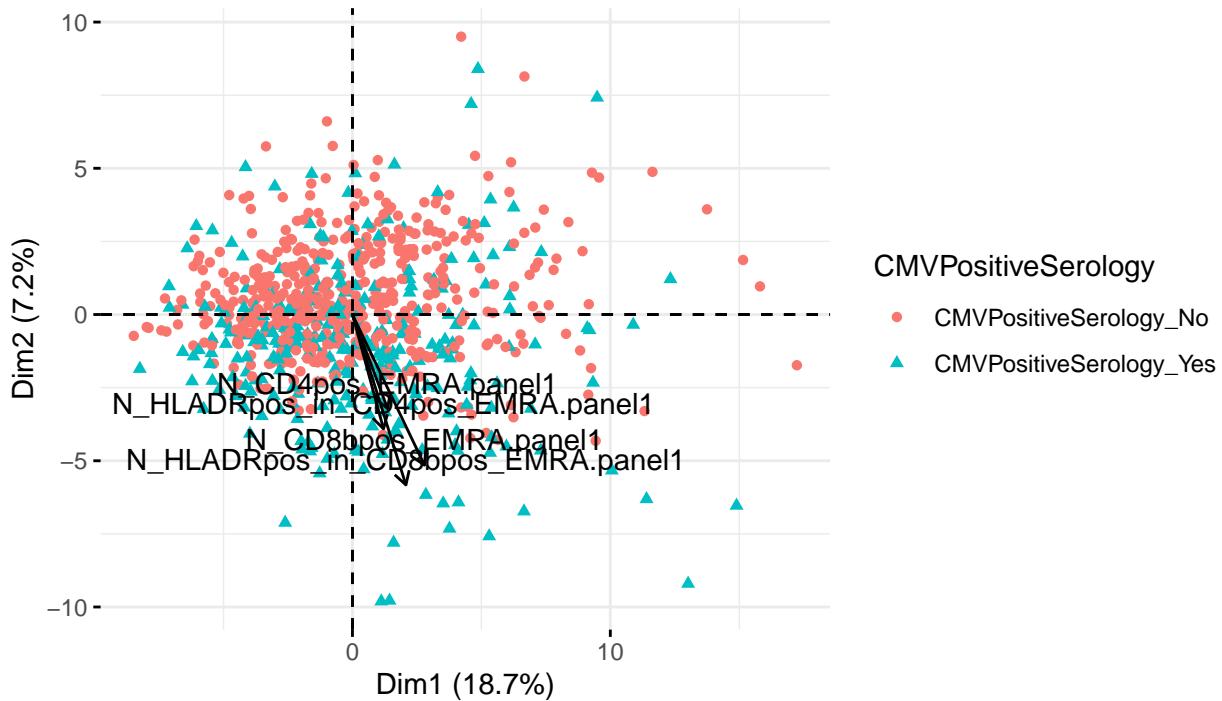
## Individuals – PCA



On visualise les cellules de la réponse immunitaire au CMV, pour vérifier qu'elles sont bien selon l'axe 2 qui différencie la positivité au CMV, vers le bas.

```
fviz_pca_biplot(res.pca, axes = c(1, 2), habillage = "CMVPositiveSerology",
                 select.var = list(name = c("N_CD4pos_EMRA.panel1",
                                             "N_HLADRpos_in_CD4pos_EMRA.panel1",
                                             "N_CD8bpos_EMRA.panel1",
                                             "N_HLADRpos_in_CD8bpos_EMRA.panel1")),
                 col.var = "black", invisible = c("quali"), label = c("var"))
```

PCA – Biplot



### 3.4 t-SNE

Le t-SNE est une autre technique de réduction de dimension, mais non linéaire. Le principe est que l'on cherche à garder la même distance entre les points dans l'espace de départ et dans l'espace de petite dimension. Le résultat conserve donc mieux les distances que le résultat de l'ACP.

```
label = expr$Stimulus.Name
data = subset(expr, select = - c(SUBJID, Stimulus.Name)) # on récupère les données dans expr
tsne <- Rtsne(data, dims = 2, perplexity=30, verbose=TRUE, max_iter = 500)

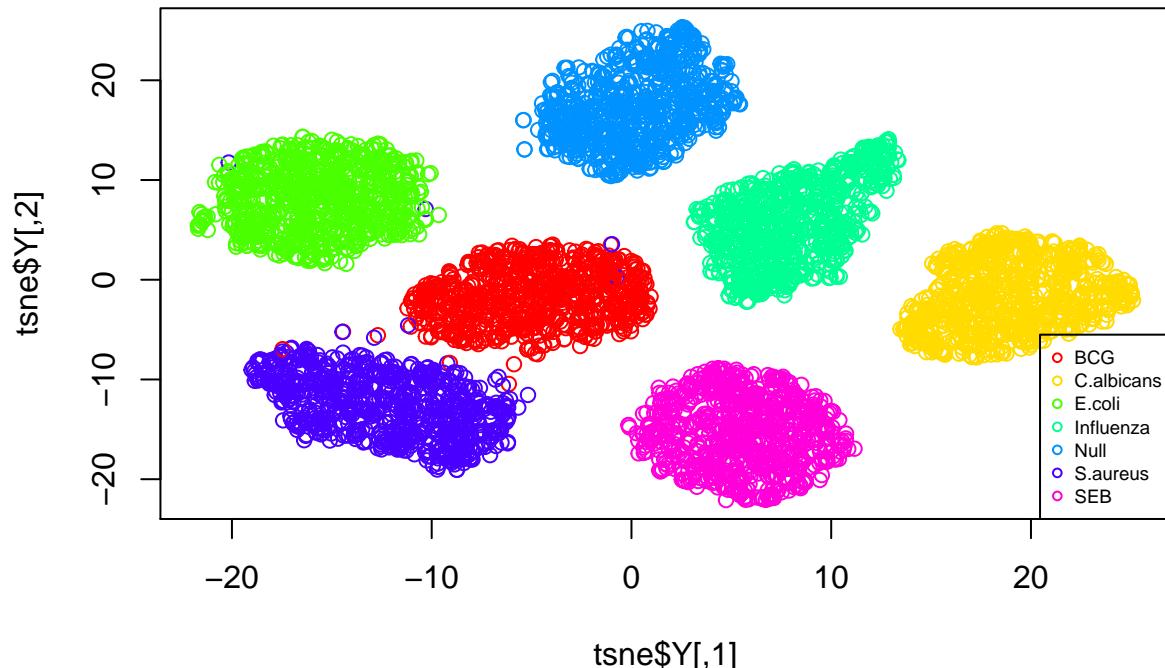
## Performing PCA
## Read the 5635 x 50 data matrix successfully!
## Using no_dims = 2, perplexity = 30.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 1.51 seconds (sparsity = 0.022785)!
## Learning embedding...
## Iteration 50: error is 89.817499 (50 iterations in 1.14 seconds)
## Iteration 100: error is 73.090591 (50 iterations in 1.13 seconds)
## Iteration 150: error is 70.692952 (50 iterations in 1.04 seconds)
## Iteration 200: error is 69.897934 (50 iterations in 1.05 seconds)
## Iteration 250: error is 69.486245 (50 iterations in 1.07 seconds)
## Iteration 300: error is 2.545206 (50 iterations in 1.00 seconds)
## Iteration 350: error is 2.281025 (50 iterations in 0.98 seconds)
## Iteration 400: error is 2.115564 (50 iterations in 0.96 seconds)
## Iteration 450: error is 2.013819 (50 iterations in 0.96 seconds)
## Iteration 500: error is 1.944737 (50 iterations in 0.96 seconds)
## Fitting performed in 10.29 seconds.
```

```

colors = rainbow(7)
names(colors) = unique(label)
plot(tsne$Y, main="t-SNE, perplexité = 30", col = colors)
legend("bottomright", unique(label), col = colors, pch=1, cex=0.6)

```

## t-SNE, perplexité = 30



```
tsne <- Rtsne(data, dims = 2, perplexity=100, verbose=TRUE, max_iter = 500)
```

```

## Performing PCA
## Read the 5635 x 50 data matrix successfully!
## Using no_dims = 2, perplexity = 100.000000, and theta = 0.500000
## Computing input similarities...
## Building tree...
## Done in 3.28 seconds (sparsity = 0.070550)!
## Learning embedding...
## Iteration 50: error is 76.008289 (50 iterations in 1.82 seconds)
## Iteration 100: error is 59.418040 (50 iterations in 1.68 seconds)
## Iteration 150: error is 57.594710 (50 iterations in 1.52 seconds)
## Iteration 200: error is 56.955499 (50 iterations in 1.51 seconds)
## Iteration 250: error is 56.642413 (50 iterations in 1.49 seconds)
## Iteration 300: error is 1.611559 (50 iterations in 1.43 seconds)
## Iteration 350: error is 1.423371 (50 iterations in 1.38 seconds)
## Iteration 400: error is 1.319985 (50 iterations in 1.56 seconds)
## Iteration 450: error is 1.258894 (50 iterations in 1.51 seconds)
## Iteration 500: error is 1.220367 (50 iterations in 1.38 seconds)
## Fitting performed in 15.27 seconds.

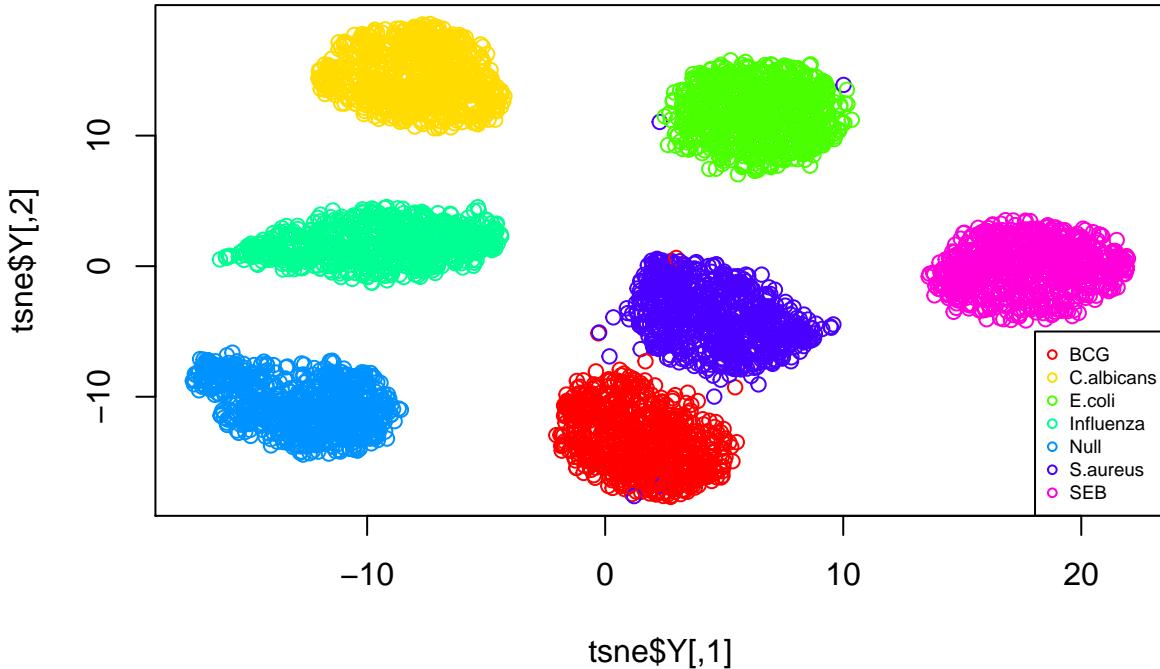
```

```

colors = rainbow(7)
names(colors) = unique(label)
plot(tsne$Y, main="t-SNE, perplexité = 100", col = colors)
legend("bottomright", unique(label), col = colors, pch=1, cex=0.6)

```

## t-SNE, perplexité = 100



On remarque que le t-SNE permet de séparer les différentes stimulations, bien mieux que l'ACP. On remarque aussi que lorsque la perplexité augmente, l'algorithme tend à rapprocher les stimulations BCG et S.aureus, ce qui pourrait indiquer une similitude dans l'expression des gènes pour ces deux stimulations.

## 4 Identification de gènes spécifiquement exprimés dans la réponse à un stimulus

### 4.1 Identification de gènes par leur intensité d'expression

Notre but ici est d'identifier l'expression de gènes caractéristiques dans la réponse à un stimulus à l'aide du clustering hiérarchique. Dans un premier temps, nous allons identifier ces gènes à l'aide de la visualisation sur une heatmap. Puis nous allons vérifier que des gènes exprimés ne se retrouvent pas dans la réponse à aucun stimulus. Enfin, nous allons effectuer une ACP uniquement sur les gènes spécifiquement identifiés et conclure si ces gènes permettent de séparer le groupe stimulé au virus d'intérêt. Nous allons détailler cette méthode pour le virus de la grippe : Influenza.

#### 4.1.1 Description détaillée de la méthode pour le virus de la grippe Influenza

##### Extraction des données

```
#Extraction stimulation Influenza
expI <- expr[expr$Stimulus.Name=='Influenza',]
expIgenes <- expI[,-c(1:2)] #Suppression des données qualitatives
```

##### Création d'une heatmap

```
myPalette <- colorRampPalette(rev(brewer.pal(11, "Spectral")))
```

```

## Fonction pour réaliser le dendrogramme
hclusfun <- function(x) hclust(dist(x), method="ward.D")
## Conversion du data.frame en matrix
expIgenes <- as.matrix(expIgenes)

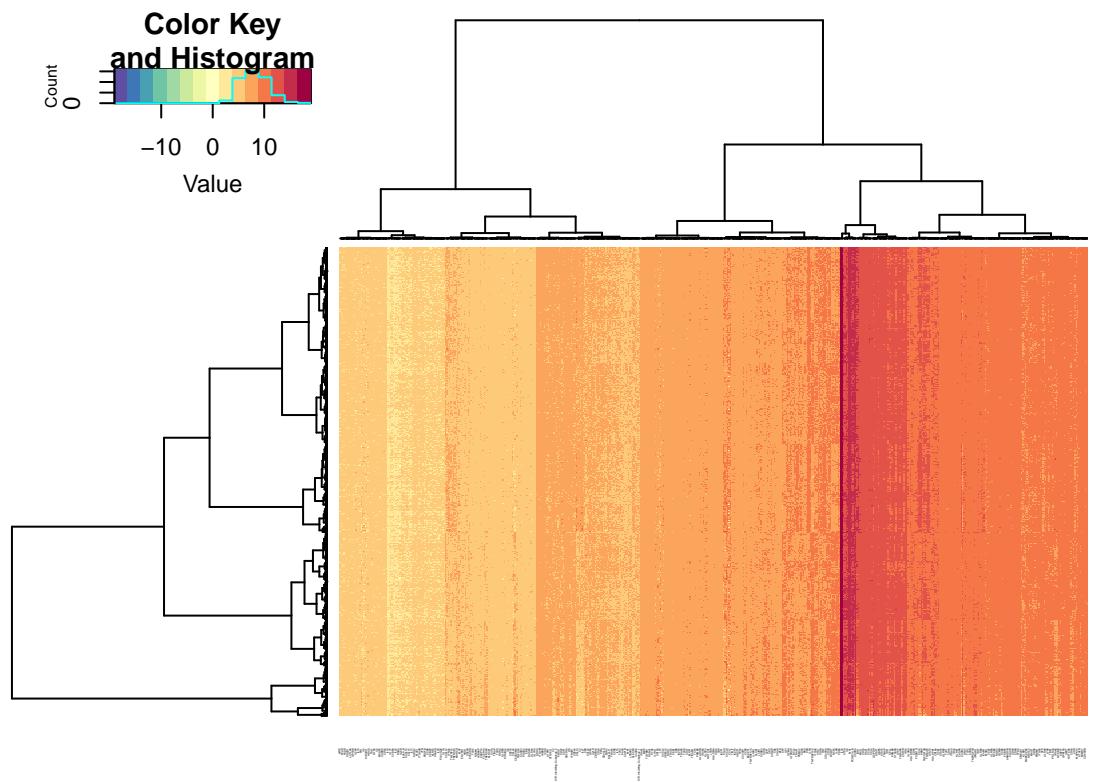
```

### Affichage de la heatmap

```

##Heatmap Influenza
heatmapFlu <- heatmap.2(expIgenes, dendrogram="both", scale="none",
trace="none", key=TRUE, col=myPalette, hclustfun = hclusfun,
cexRow = 0.1, cexCol=0.1)

```



Nous pouvons remarquer un groupe de gènes dont la coloration est bien plus foncée que les autres. Allons extraire ce groupe de gènes particulier. Dans un premier temps, nous avons extrait les gènes les plus foncés. Ils sont au nombre de 2-3, qui sont B2M (Beta-2-Microglobulin) et IL32 (Interleukin 32), pour chaque stimulus et il se trouve que ces gènes sont aussi très exprimés dans les échantillons sans stimulus. Ces deux gènes jouent des rôles essentiels dans la réponse immunitaire (d'après le site [genecards.org](http://genecards.org)), donc leur forte expression pour chacune des situations étudiées est justifiée.

Ainsi, ces gènes ne sont pas très intéressants pour caractériser la réponse immunitaire à la grippe. Ainsi, dans un second temps, nous avons extrait le deuxième groupe de gènes le plus exprimé ne comprenant pas ces 2-3 gènes cités précédemment.

Pour ce faire, nous avons extrait le dendrogramme des gènes de la heatmap. Ensuite, nous avons dessiné les clusters à l'aide de la fonction `rect.dendrogram()`, en testant plusieurs valeurs de `k` jusqu'à notre cluster d'intérêt soit isolé et nous avons coloré ces groupes pour identifier le numéro du cluster d'intérêt, comprenant les gènes qui sont le plus exprimés, à part les deux gènes cités précédemment.

```

dendroFlu <- heatmapFlu$colDendrogram
dendroFlu <- as.dendrogram(dendroFlu)
clustersFlu <- cutree(dendroFlu, k=16)

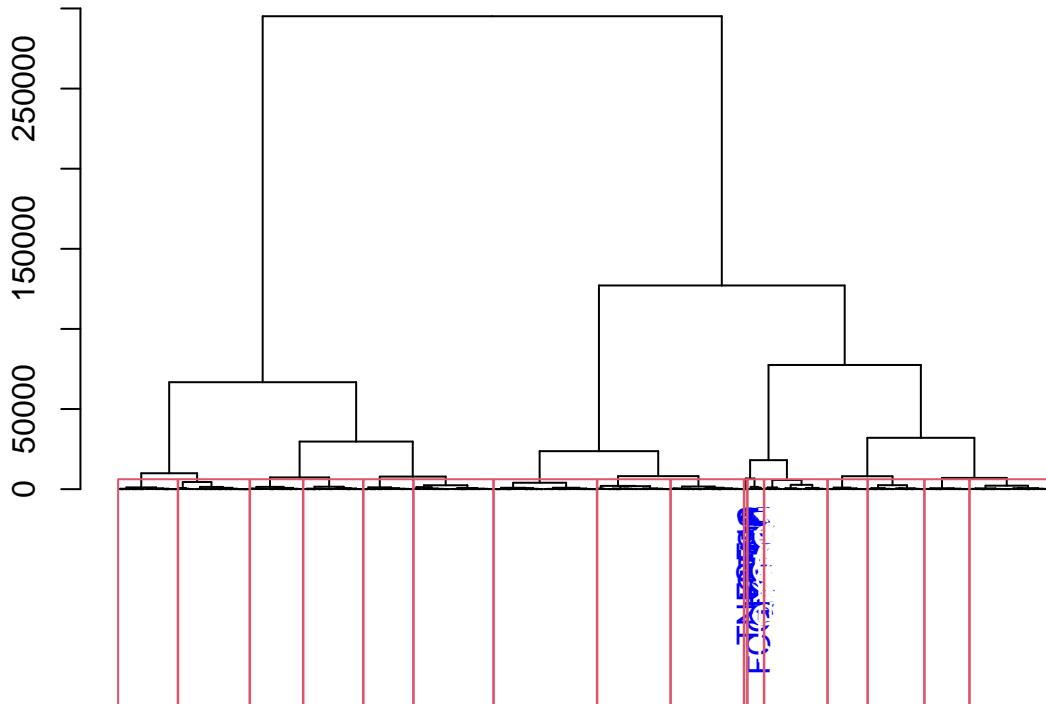
```

```

groupCodes <- clustersFlu
colorCodes <- c("white", "white", "white")

labels_colors(dendroFlu) <- colorCodes[groupCodes][order.dendrogram(dendroFlu)]
plot(dendroFlu)
rect.dendrogram(dendroFlu, k=16)

```



```

exprMaxHeatmapFlu <- names(clustersFlu)[clustersFlu==16]
exprMaxHeatmapFlu

```

```

## [1] "CTSS"      "CXCL10"     "CXCR4"      "FCGR3A_B"    "HLA.A"      "IFIT2"
## [7] "IFITM1"    "IL8"        "MX1"        "TNFSF10"

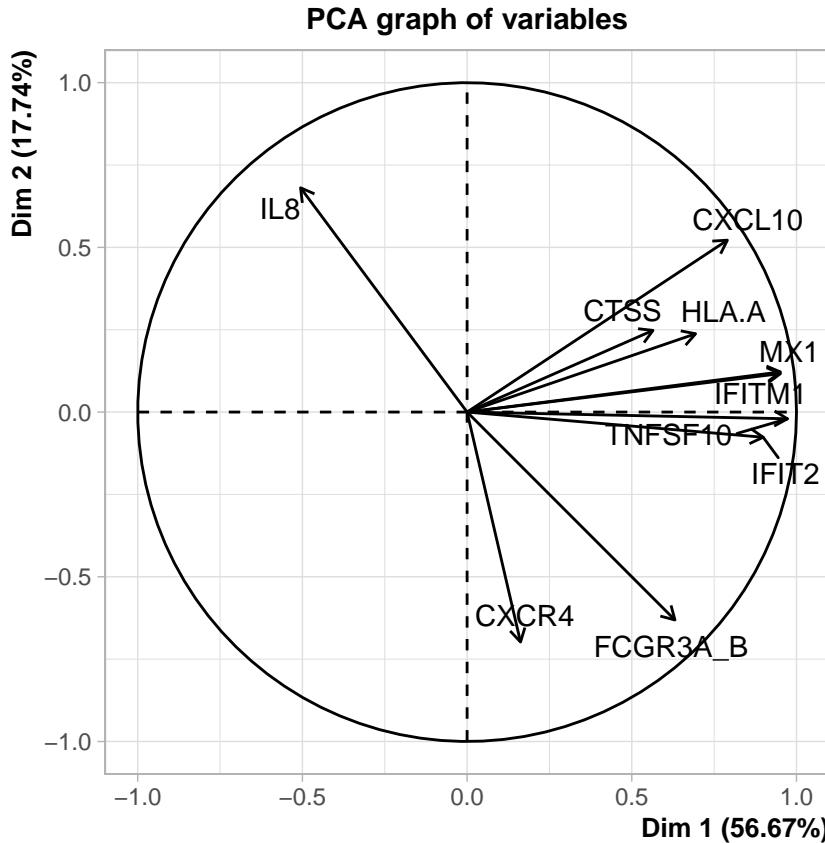
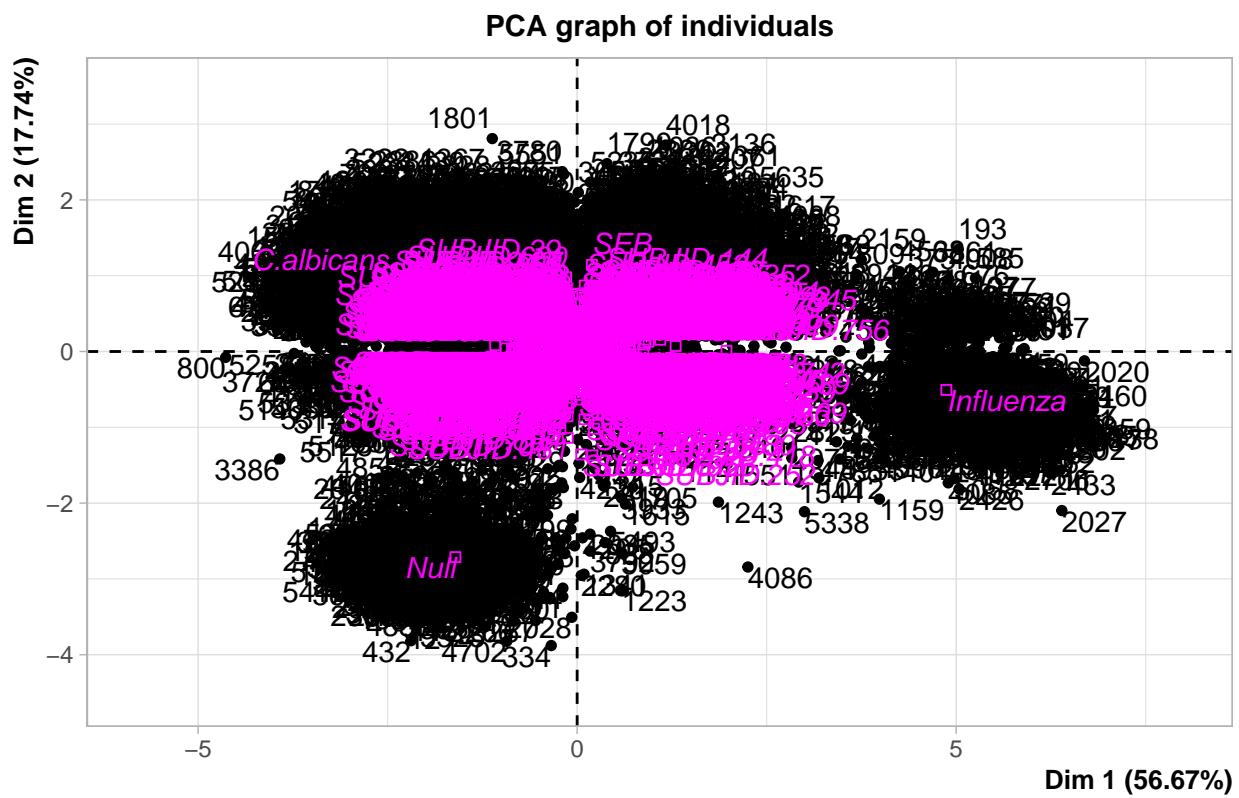
```

Ces gènes identifiés, nous les avons ensuite utilisés pour réaliser l'ACP, en ne les sélectionnant que eux, et voir si l'expression de ces gènes peut caractériser la réponse au virus de la grippe.

```

fit.pca <- PCA(X=expr[,c("SUBJID", "Stimulus.Name", exprMaxHeatmapFlu)], scale.unit = TRUE, quali.sup = 1:1)

```



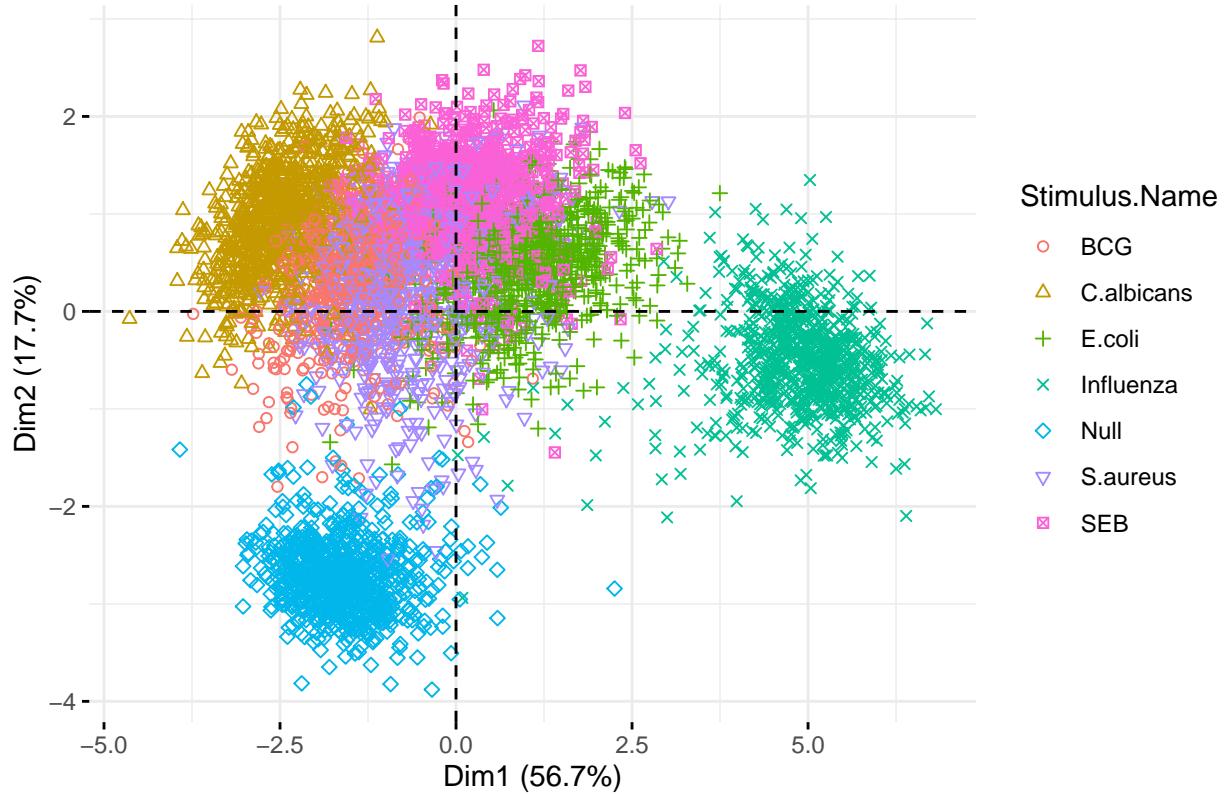
```
fviz_pca_ind(fit.pca,  
             axes = c(1,2),
```

```

habillage = 'Stimulus.Name',
invisible = 'quali',
label ="none")

```

Individuals – PCA



Nous pouvons voir que les gènes identifiés à l'aide du clustering hiérarchique discriminent bien les stimulations au virus de la grippe. Le groupe soumis à Influenza se distingue des autres selon le premier axe principal. Ces gènes sont donc caractéristiques de la réponse immunitaire à ce virus. De plus, l'ACP est très efficace ici car l'axe 1 couvre 56.7% de variance, ce qui est très correct. Ce résultat est ainsi très intéressant, et nous pouvons étudier plus en détail ces gènes à l'aide de méthodes différentes, ce que nous ferons par la suite.

#### 4.1.2 Travail similaire sur les cinq autres stimuli

Nous allons effectuer le même type d'étude et voir si nous pouvons identifier des gènes caractéristiques dans la réponse immunitaires contre les stimuli étudiés.

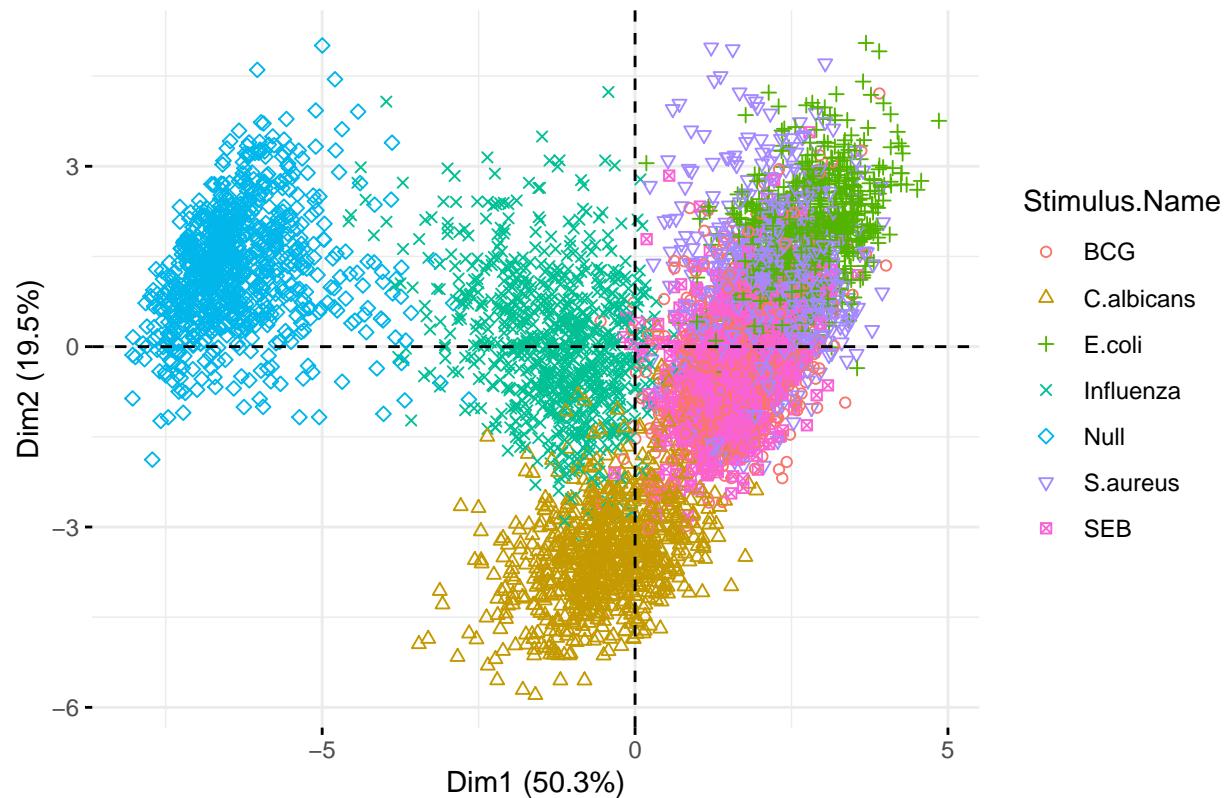
##### E.coli

```

fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label ="none")

```

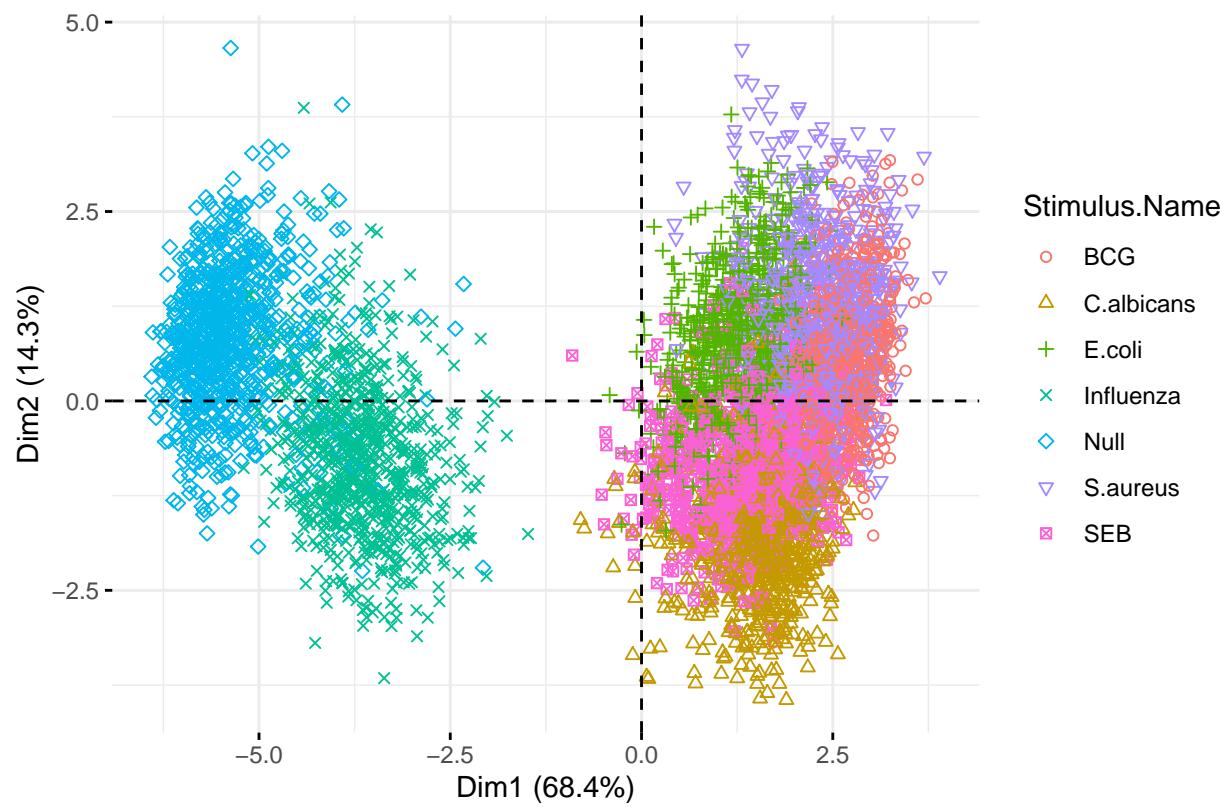
## Individuals – PCA



BCG

```
fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label = "none")
```

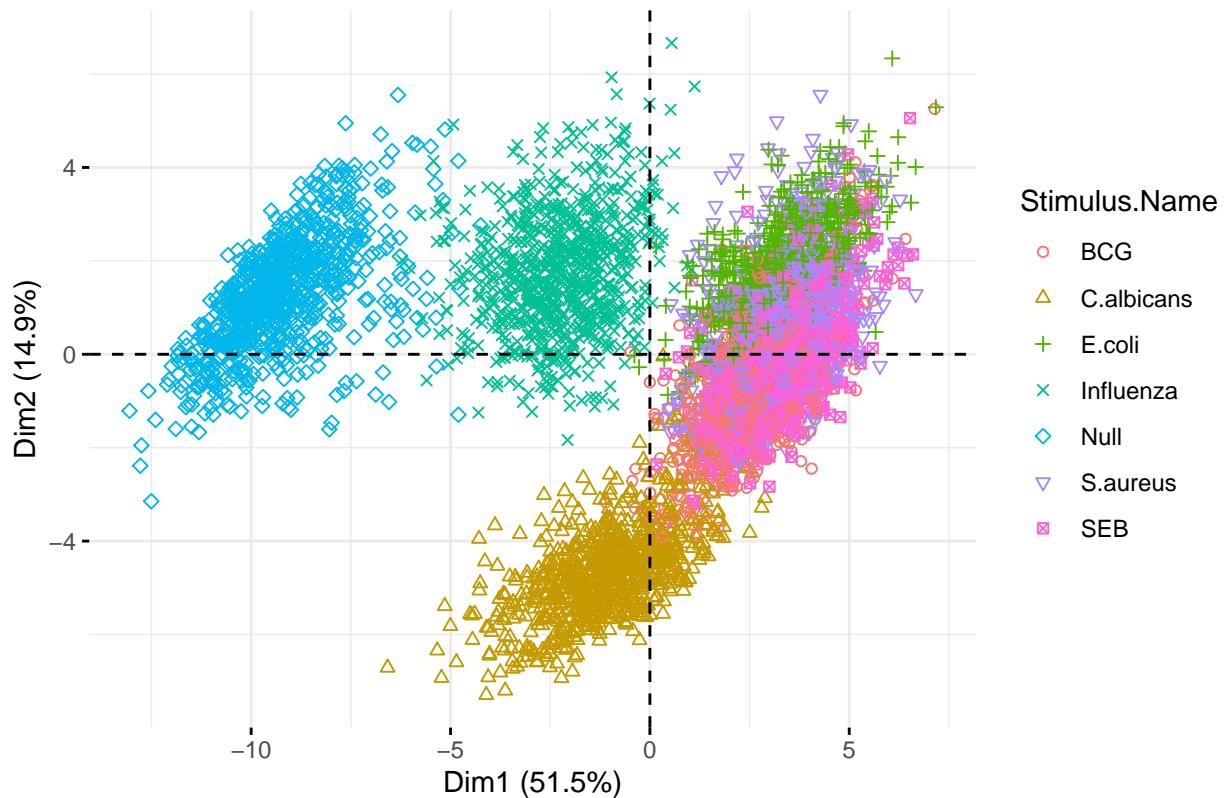
Individuals – PCA



SEB

```
fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label = "none")
```

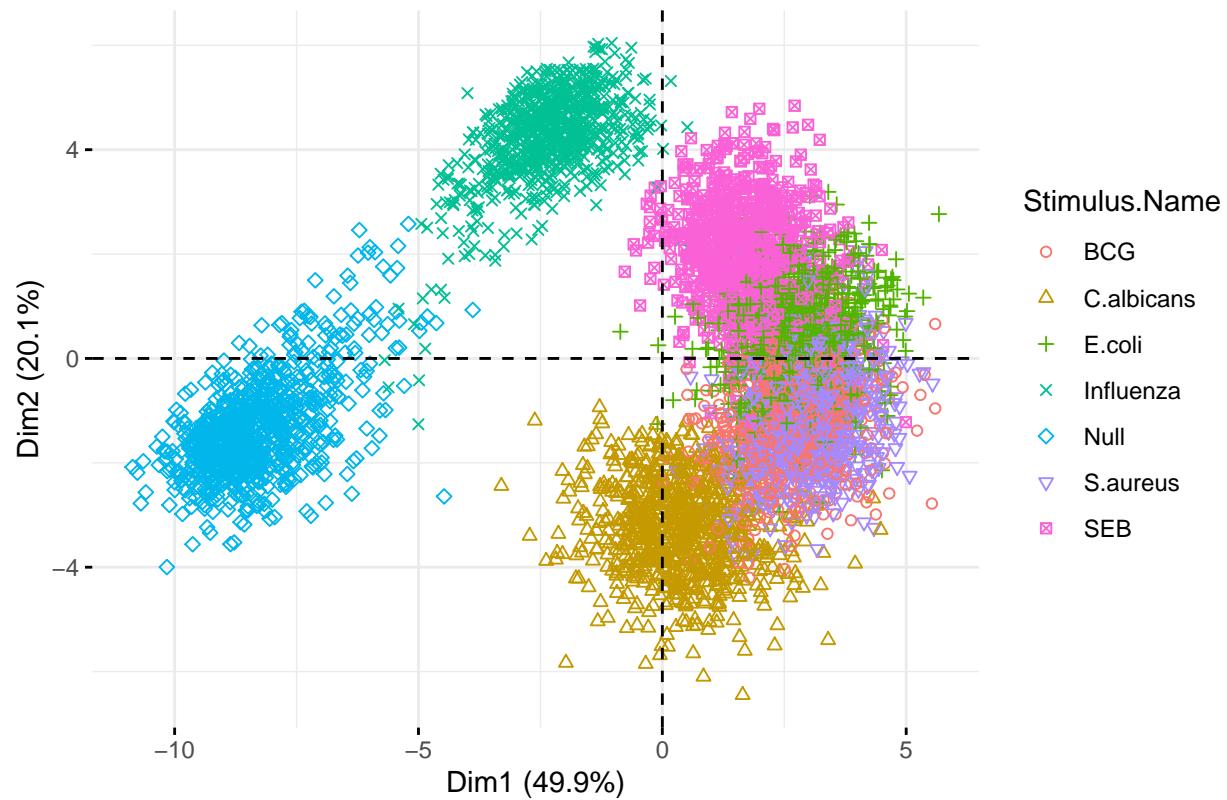
## Individuals – PCA



S.aureus

```
fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label = "none")
```

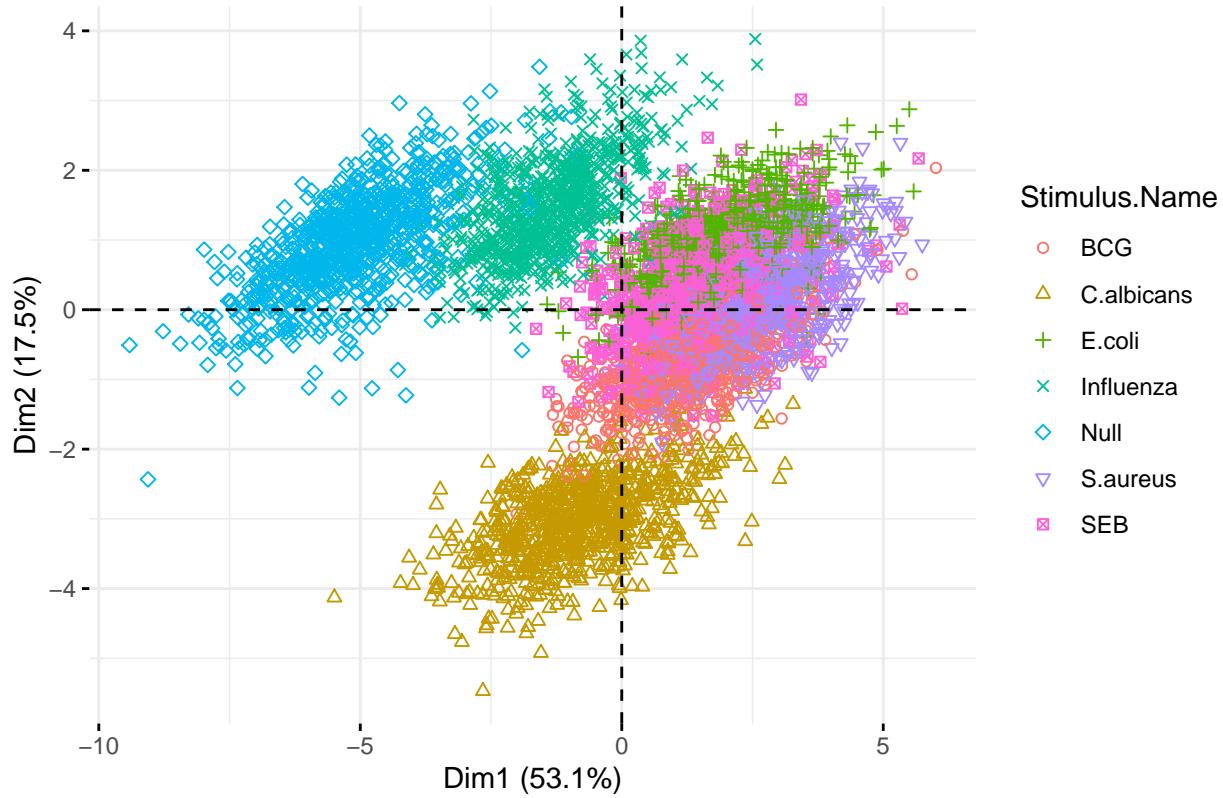
## Individuals – PCA



### C.albicans

```
fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label = "none")
```

## Individuals – PCA



### 4.1.3 Analyses

Même si la méthode employée n'a pas été concluante pour les stimuli E.coli, BCG, S.aureus et SEB car ces stimuli restent fortement regroupés peu importe les gènes utilisés, la méthode a été efficace pour les stimuli Influenza et C.albicans car ces groupes se détachent souvent des autres, peu importe les gènes que l'on utilise pour réaliser l'ACP.

N'ayant pas réussi à séparer les stimuli E.coli, BCG et S.aureus (SEB étant une stimulation à part, nous n'allons pas la considérer dans notre étude), nous essayons d'effectuer une analyse plus précise de ces stimuli afin d'essayer de les séparer.

Nous avons essayé d'identifier des gènes caractéristiques de la réponse aux stimuli étudiés en s'intéressant à leur intensité d'expression. Il est naturel d'étudier désormais la différence d'expression des gènes par rapport aux stimuli. Nous allons l'étudier en effectuant un test d'anova à un facteur, qui est le stimulus, qui permettra de dire si oui ou non, les réactions aux stimuli sont bien différentes. Puis, nous visualiserons les profils d'expression des gènes sélectionnés précédemment à l'aide d'un boxplot afin de choisir les gènes qui seront susceptibles d'être les plus discriminant entre les stimuli. Enfin, nous allons revenir sur l'ACP et l'effectuer sur les gènes sélectionnés après toutes ces méthodes et voir si oui ou non, nous arrivons à séparer ces groupes.

## 4.2 Identification de gènes par leur différence d'expression

### 4.2.1 Anova (Analysis of variance)

Nous allons effectuer le test d'hypothèses suivant :  $H_0 = \{\text{Le gène ne s'exprime pas différemment de manière significative quelque soit le stimulus}\}$  contre  $H_1 = \{\text{Dans au moins un des stimuli, le gène s'exprime}\}$

différemment de manière significative}. Nous posons notre seuil de rejet  $\alpha = 0.05$ , qui est un niveau de risque souvent utilisé.

Comme nous allons effectuer ce test sur beaucoup de gènes, nous aurons une proportion non négligeable de Faux Positifs (FP). Ainsi, nous contrôlons ce taux à l'aide de la correction de Benjamini-Hochberg cherchant à borner à 5% (car  $\alpha = 0.05$ ) le False Discovery Rate (FDR) défini par :

$$\text{FDR} = \mathbb{E} \left[ \frac{V}{R} \right]. \quad (1)$$

Nous appliquerons cette correction à notre tableau de p\_values déterminé par l'anova.

Soumettons désormais nos trois stimuli à ce test. Étudions ces stimuli deux à deux.

### E.coli contre BCG

Voici le code employé :

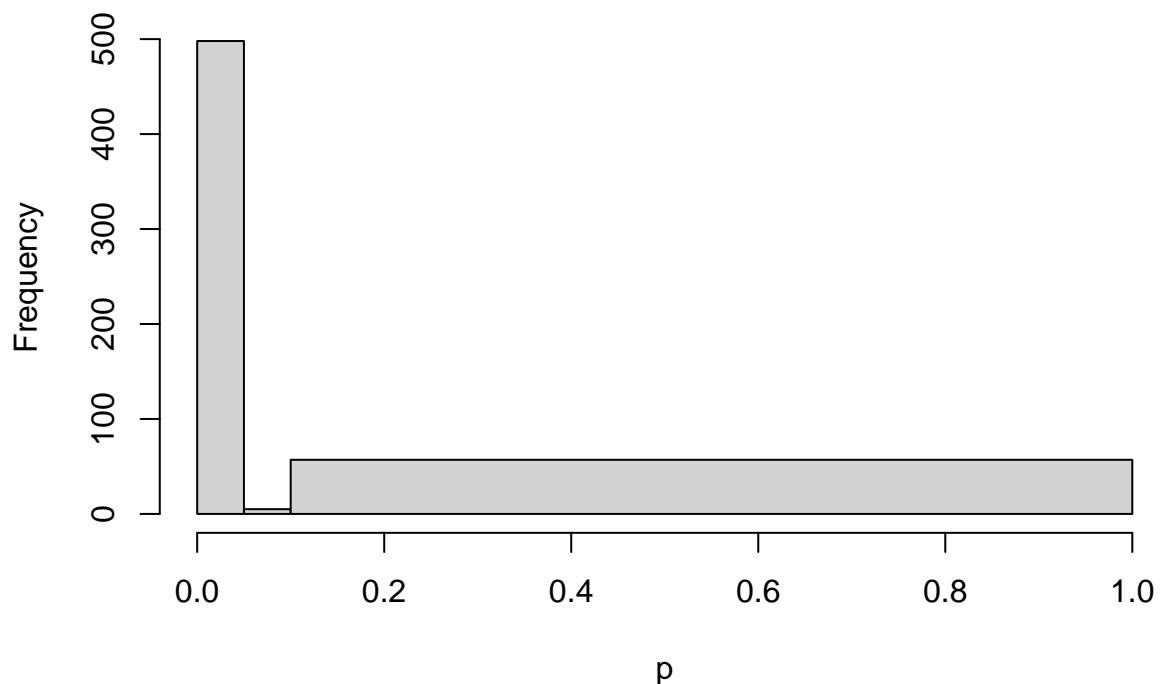
```
1.stimuli <- c("E.coli", "BCG") # Nous ne gardons que les lignes correspondant
                                # aux stimuli BCG et E.coli
expr3 <- expr[expr$Stimulus.Name %in% 1.stimuli,]
expr3genes <- expr3[, -c(1, 2)] # Suppression des colonnes inutiles à l'analyse
p = apply(expr3genes, 2, function(x){resultat = lm(x ~ factor(expr3$Stimulus.Name));
analyse<-anova(resultat);analyse$`Pr(>F)`[1]})
```

La dernière ligne de code ci-dessus mérite un peu plus de détails. Notons p le tableau qui contiendra les p-values des tests effectués. En lisant l'application de apply() de gauche à droite, nous appliquons à chaque colonne du jeu de donnée ne contenant que l'expression des gènes, et donc à chaque gène, le test d'anova à facteur, ce dernier étant le stimulus. Enfin, nous récupérons la valeur de la p-value.

```
p <- p.adjust(p, method ="fdr") # Correction de Benjamini-Hochberg
hist(p, main = "p-values ajustées de l'expression des gènes, BCG contre E.coli",
      breaks = c(0, 0.05, 0.1, 1), prob = FALSE)

## Warning in plot.histogram(r, freq = freq1, col = col, border = border, angle =
## angle, : the AREAS in the plot are wrong -- rather use 'freq = FALSE'
```

## p-values ajustées de l'expression des gènes, BCG contre E.coli



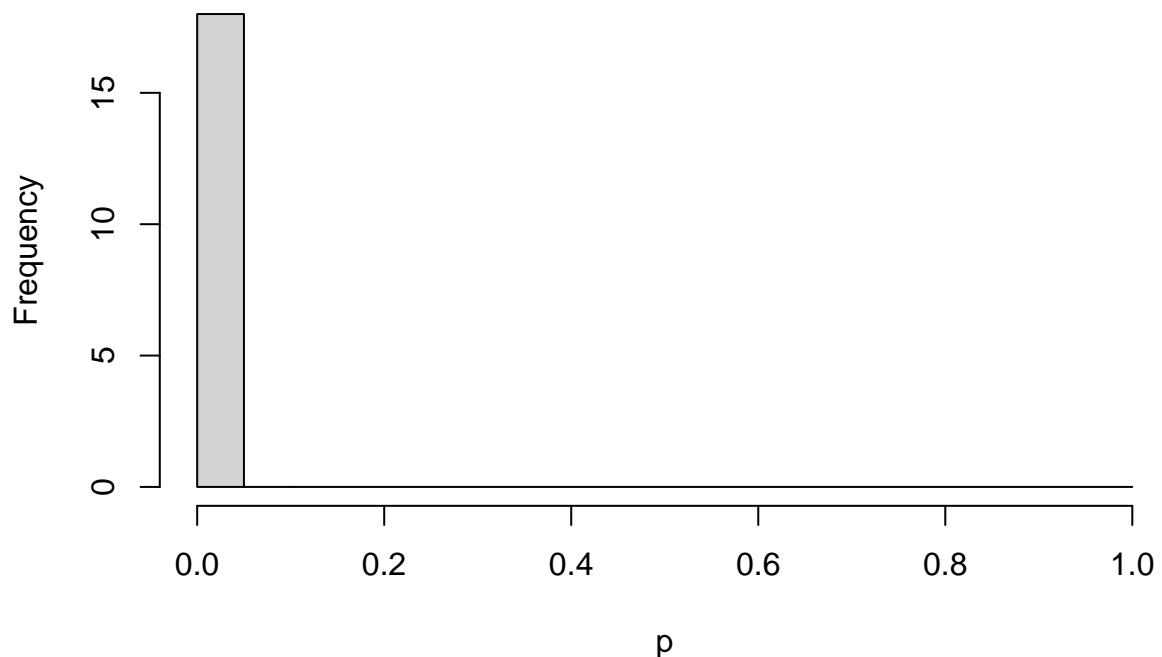
```
# le découpage de l'histogramme est réalisé ainsi pour visualiser facilement  
# les hypothèses rejetées ou conservées.
```

Nous pouvons remarquer que la majorité des gènes s'expriment différemment de manière significative entre les stimuli E.coli et BCG. Cela nous laisse donc penser que nous pouvons parvenir à trouver des gènes qui pourront les différencier.

### E.coli contre S.aureus

```
hist(p, main = "p-values ajustées de l'expression des gènes, E.coli contre S.aureus",  
      breaks = c(0, 0.05, 0.1, 1), prob = FALSE)  
  
## Warning in plot.histogram(r, freq = freq1, col = col, border = border, angle =  
## angle, : the AREAS in the plot are wrong -- rather use 'freq = FALSE'
```

## p-values ajustées de l'expression des gènes, E.coli contre S.aureus

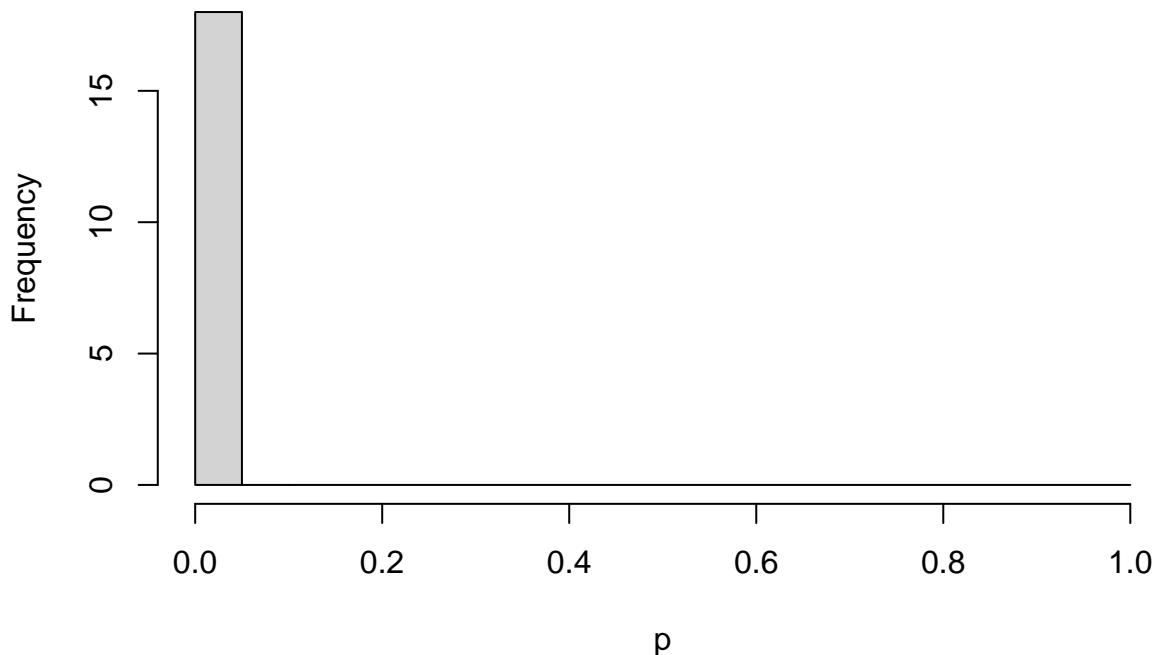


BCG contre S.aureus

```
hist(p, main = "p-values ajustées de l'expression des gènes, BCG contre S.aureus",
     breaks = c(0, 0.05, 0.1, 1), prob = FALSE)

## Warning in plot.histogram(r, freq = freq1, col = col, border = border, angle =
## angle, : the AREAS in the plot are wrong -- rather use 'freq = FALSE'
```

## p-values ajustées de l'expression des gènes, BCG contre S.aureus



Dans les deux dernières analyses, nous pouvons voir que toutes les hypothèses sont rejetées. Ainsi, le stimulus S.aureus semble être plus facilement séparable des deux autres. Essayons désormais de sélectionner les gènes qui pourront différencier ces st

Le nombre de gènes à étudier étant élevé, nous allons nous restreindre aux groupes de gènes déterminés à l'aide du clustering hiérarchique. Même sur cet ensemble restreint de gènes, nous pourrons obtenir des résultats intéressants.

Nous avons tester toutes les combinaisons possibles de confrontations de stimuli pour aboutir à la meilleure discrimination possible.

Nous allons vous présenter notre résultat le plus concluant, ce qui permettre de vous détailler la démarche suivie.

### 4.2.2 Travail sur un plus petit ensemble de gènes

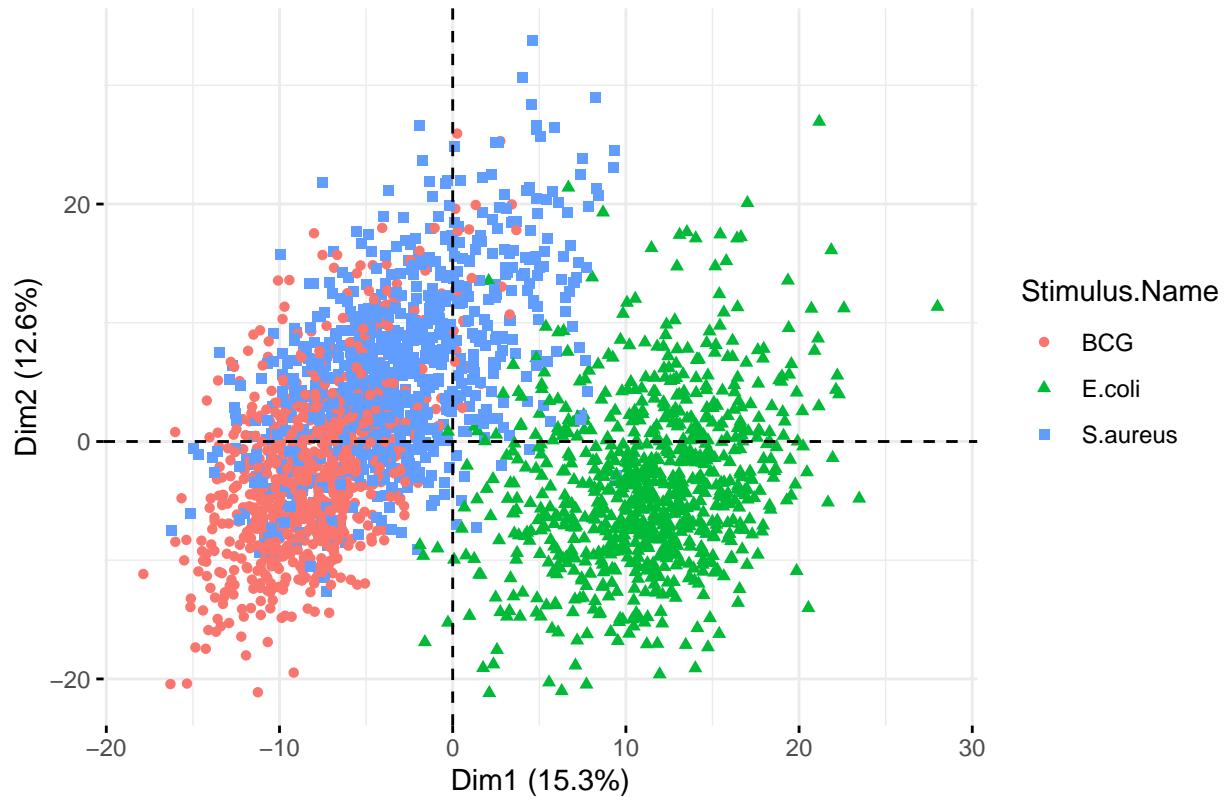
#### 4.2.2.1 Enjeu de départ

Nous allons restreindre notre ACP aux trois stimuli étudiés.

Voici ce que l'on obtient en considérant les 560 gènes.

```
fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label ="none")
```

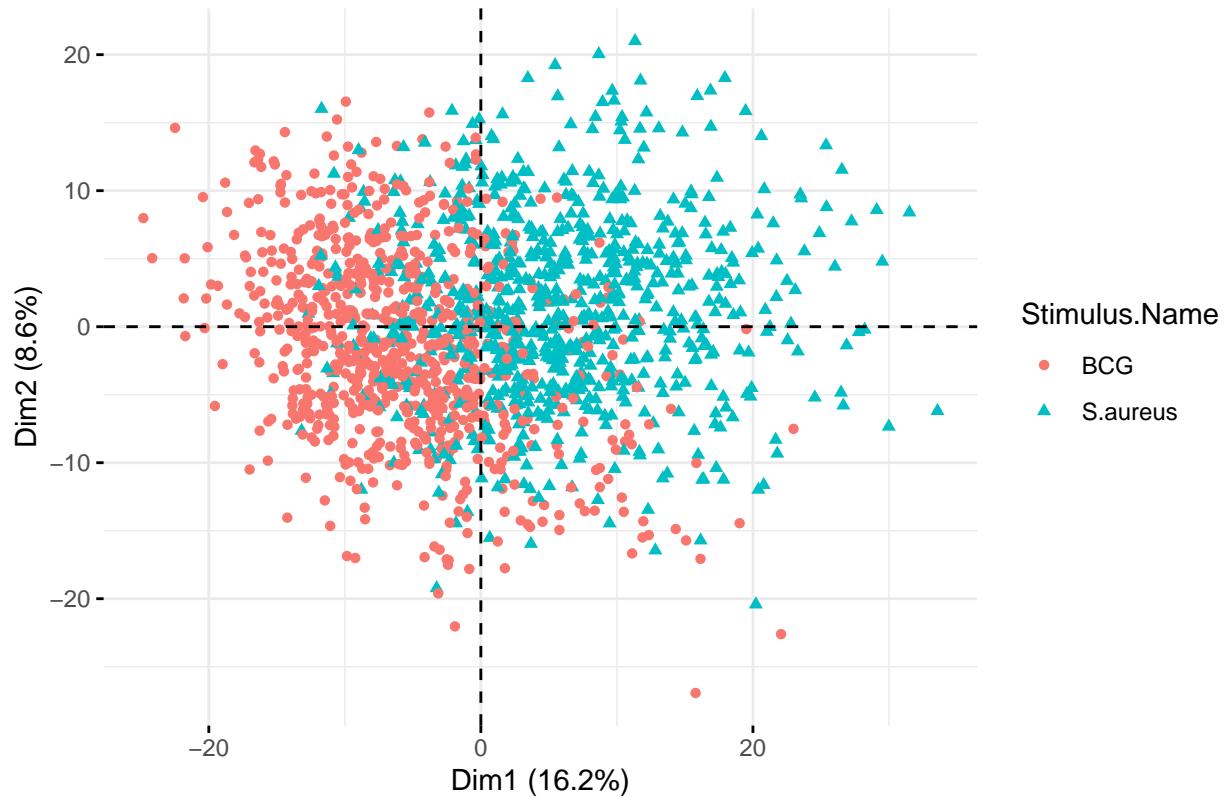
## Individuals – PCA



En ne considérant que ces trois stimuli, nous pouvons remarquer que la stimulation à E.coli se démarque bien. Les deux autres stimuli, à savoir BCG et S.aureus, se superposent.

```
fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
              label = "none")
```

## Individuals – PCA



Nous allons essayer de les séparer en se focalisant sur ces deux stimuli.

### 4.2.2.2 Boxplot

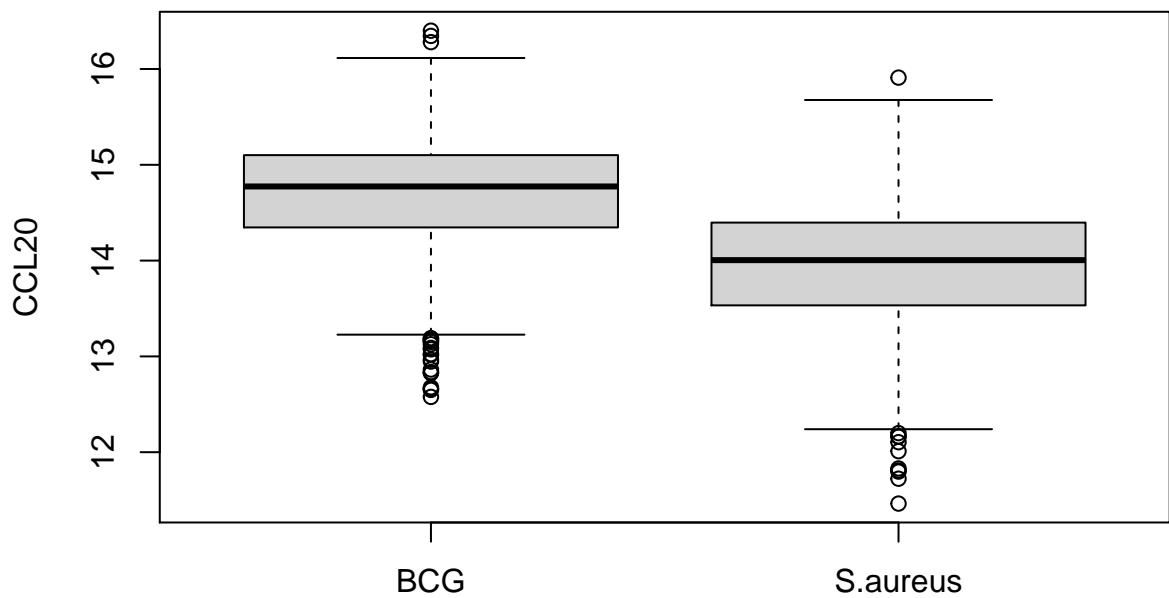
L'outil boxplot est très efficace pour comparer visuellement l'expression d'un même gène sous différents stimuli. Nous allons nous en servir sur les gènes déterminés par le clustering hiérarchique au stimulus BCG. Cela nous a permis d'en retenir 13. Les voici :

```
exprMaxHeatmapBCG
```

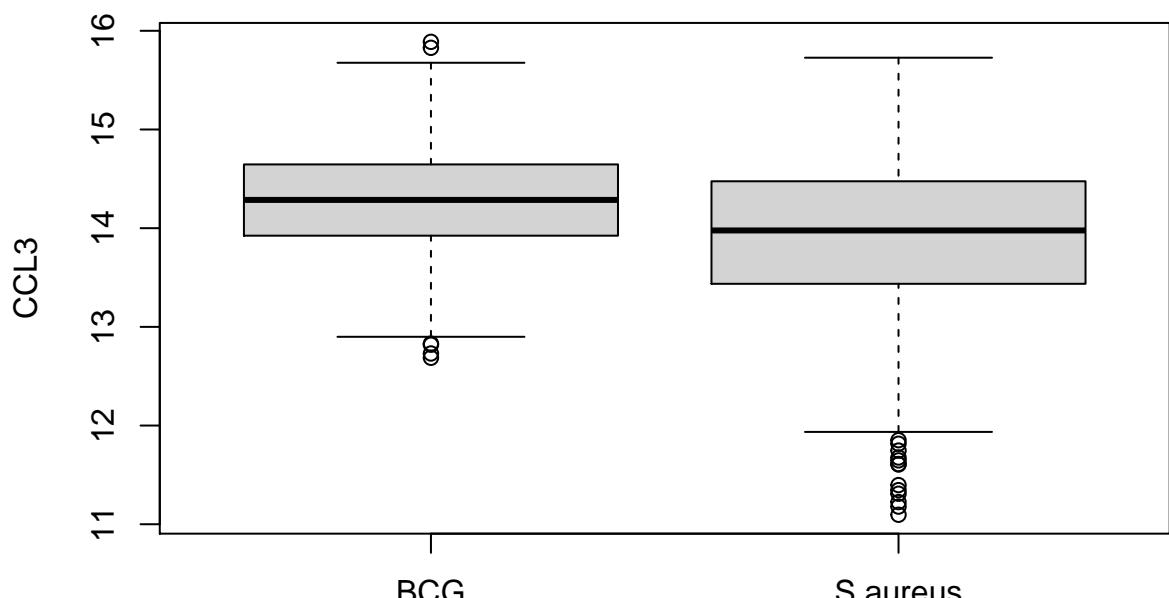
```
## [1] "CCL20"      "CCL3"       "CCL4"       "CXCL1"      "CXCL2"      "CXCR4"
## [7] "GAPDH"      "IL1A"       "NFKBIA"     "PTGS2"      "PTPRC_all"  "RPL19"
## [13] "SPP1"
```

Comparons désormais les expressions de ces 13 gènes entre les stimuli BCG et S.aureus à l'aide des boxplots.

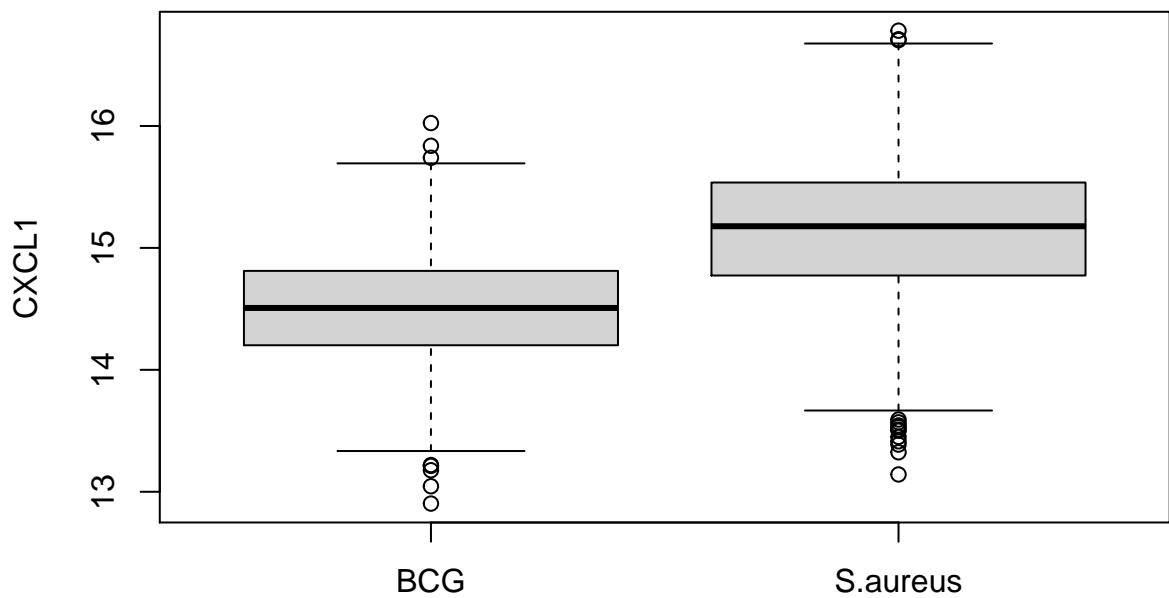
```
1.stimuli <- c("S.aureus", "BCG")
expr3 <- expr[expr$Stimulus.Name %in% 1.stimuli,]
boxplot(CCL20~Stimulus.Name, data = expr3) #
```



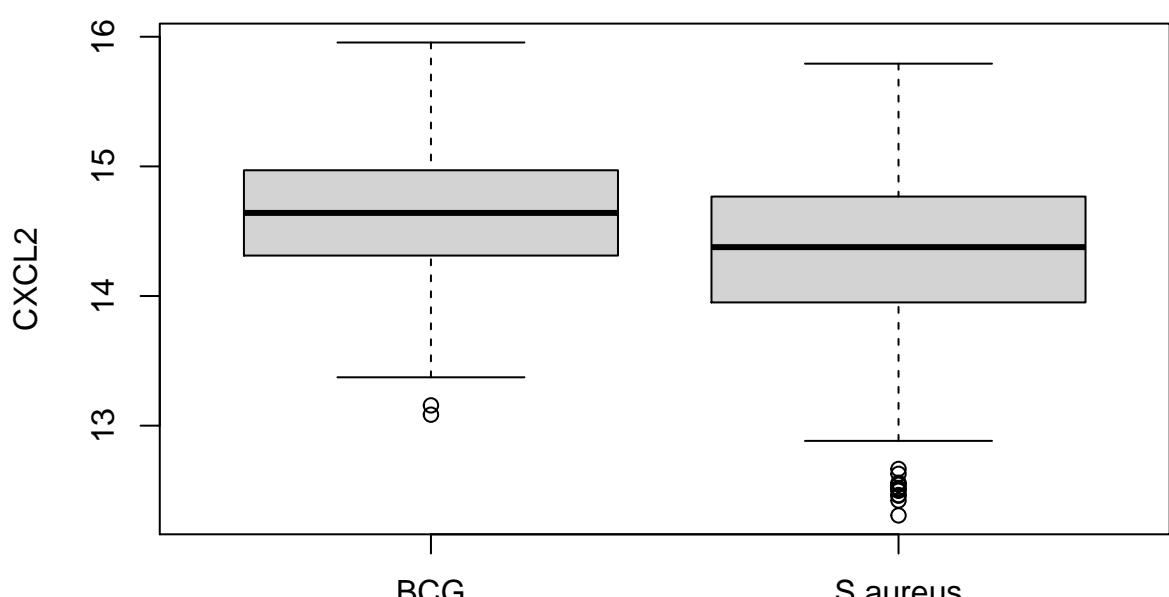
```
boxplot(CCL3~Stimulus.Name,data = expr3)
```



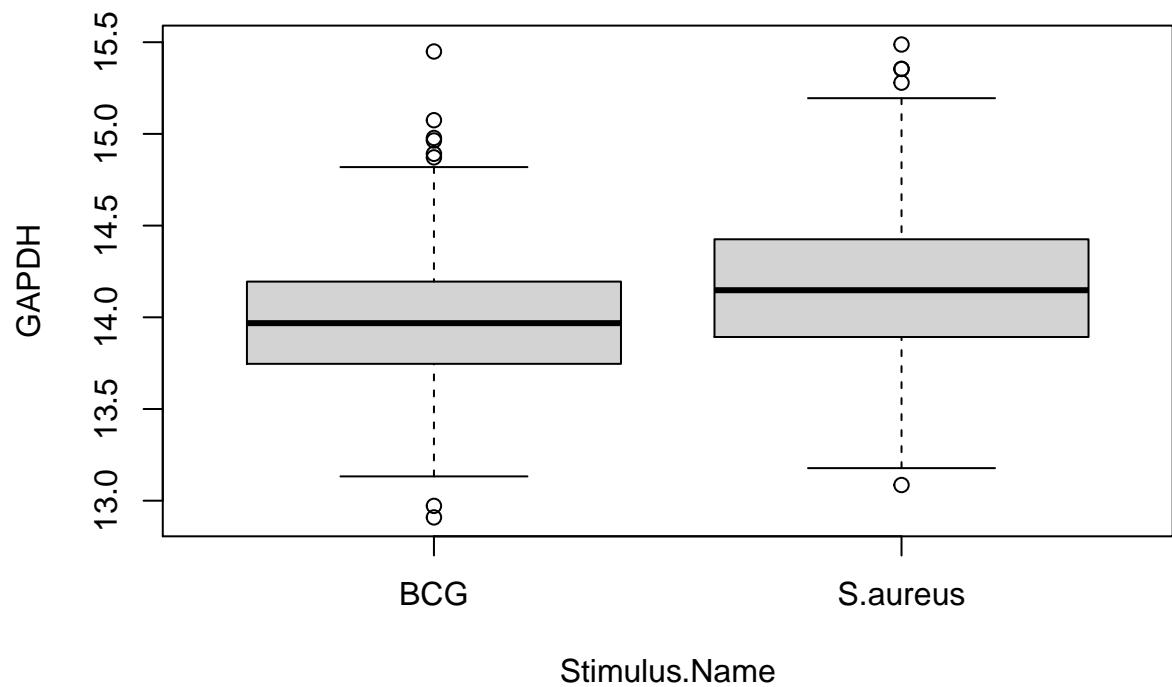
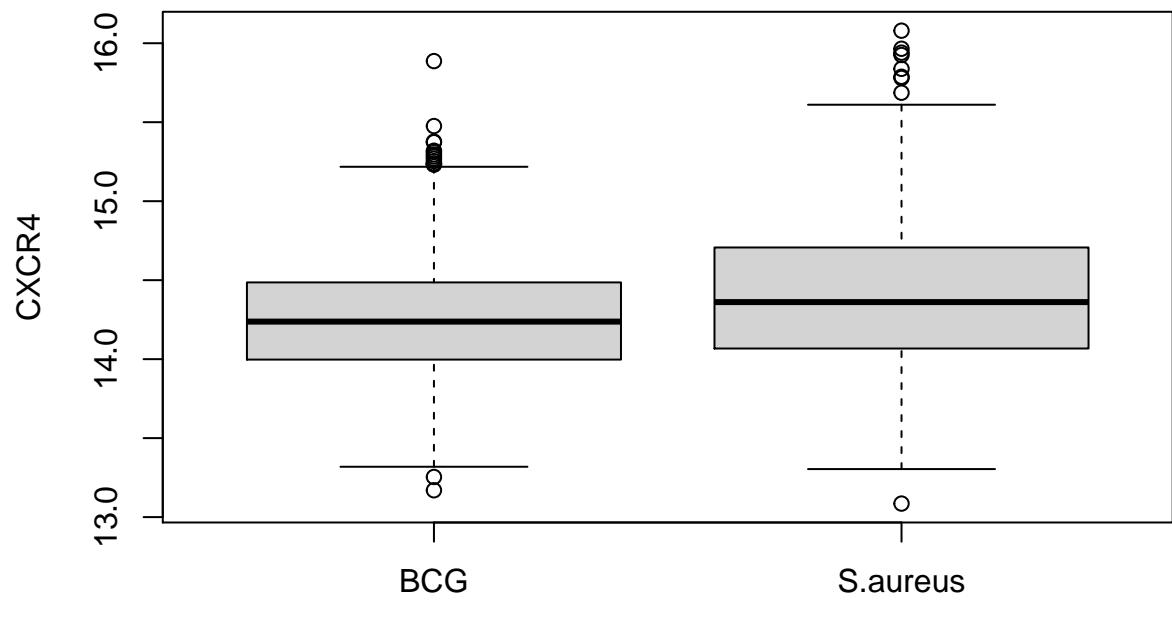
```
boxplot(CXCL1~Stimulus.Name,data = expr3) #
```

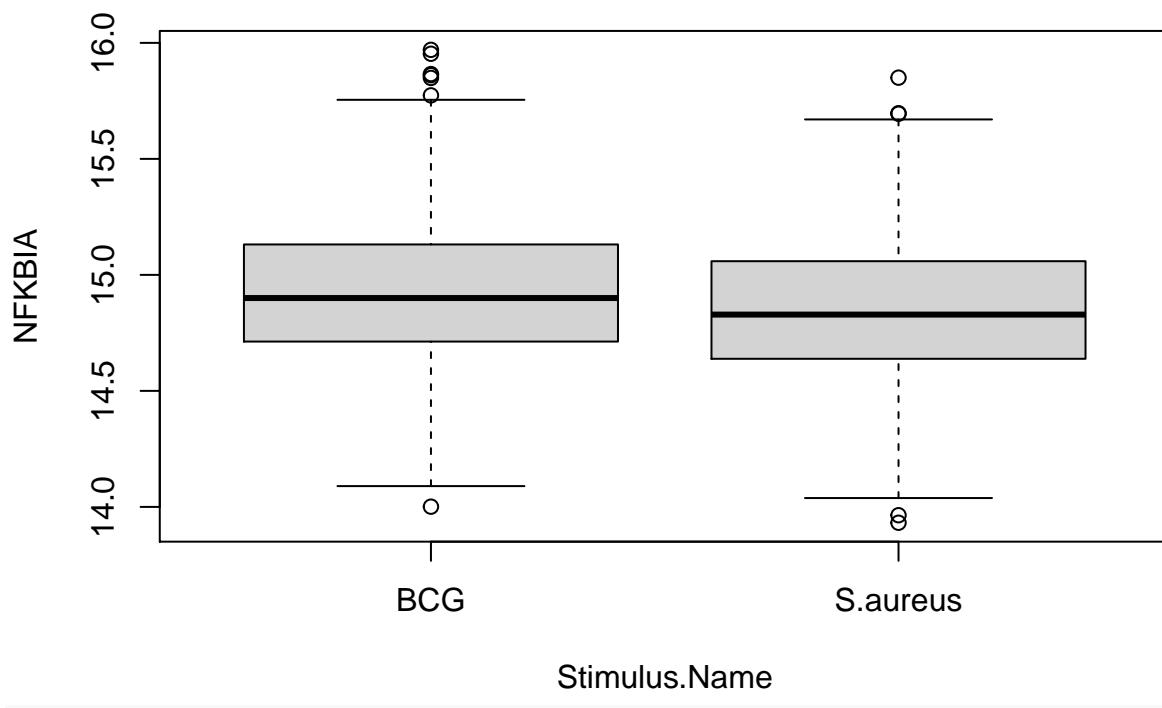
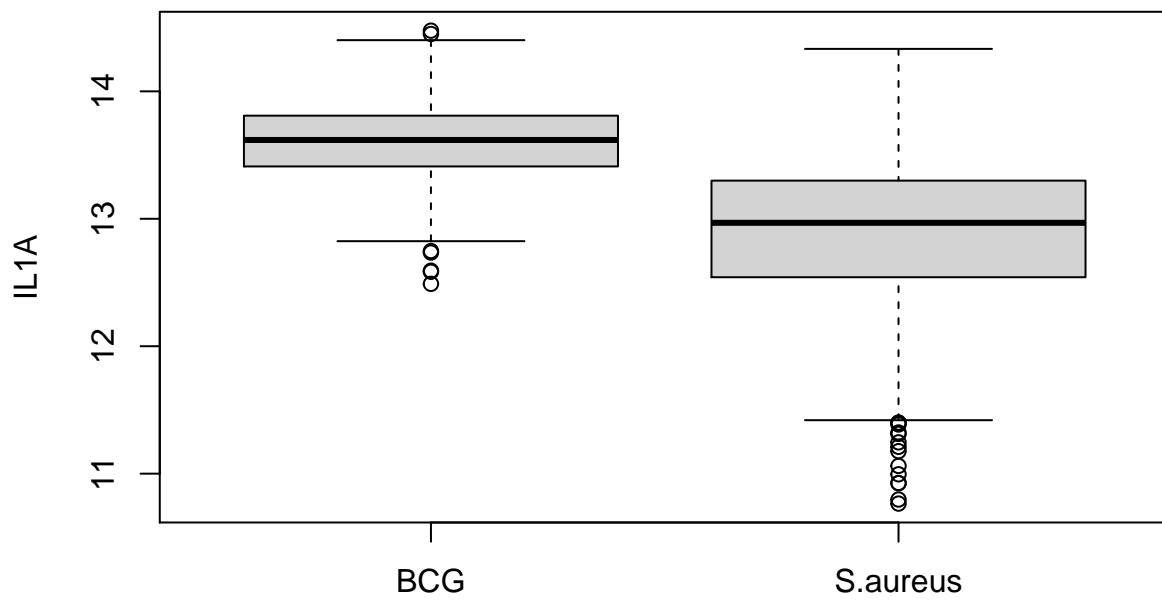


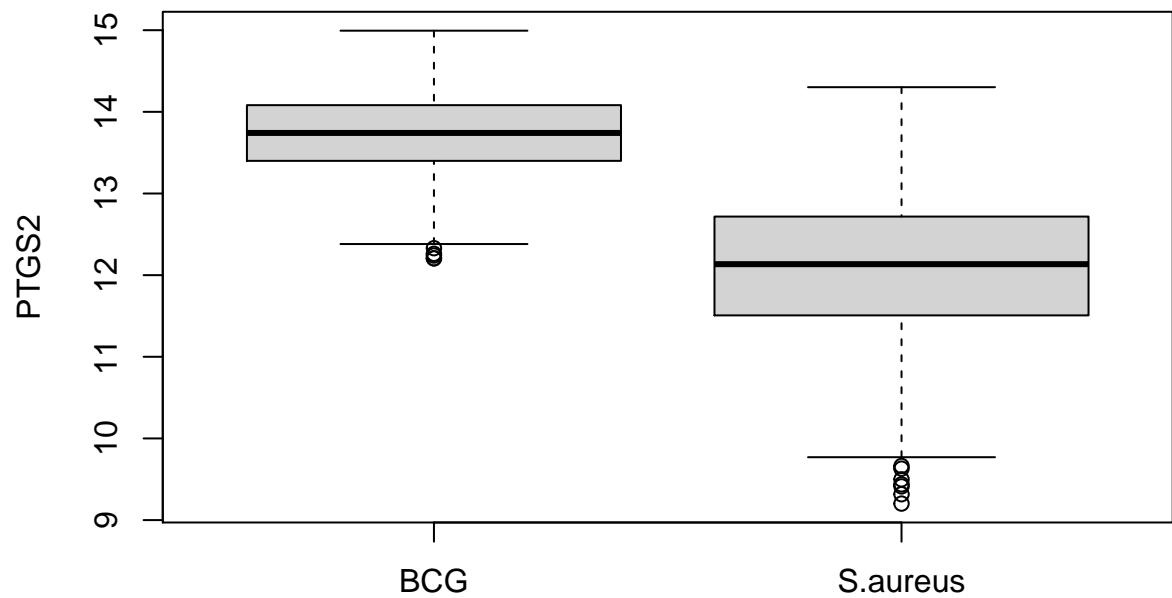
```
boxplot(CXCL1~Stimulus.Name,data = expr3)
```



```
boxplot(CXCR4~Stimulus.Name,data = expr3)
```

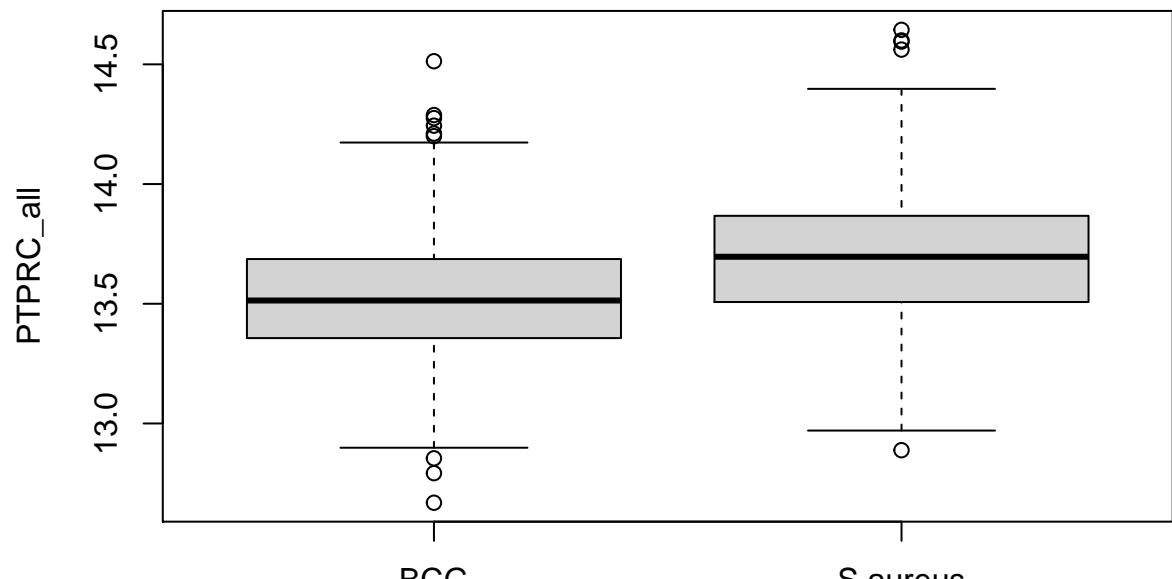






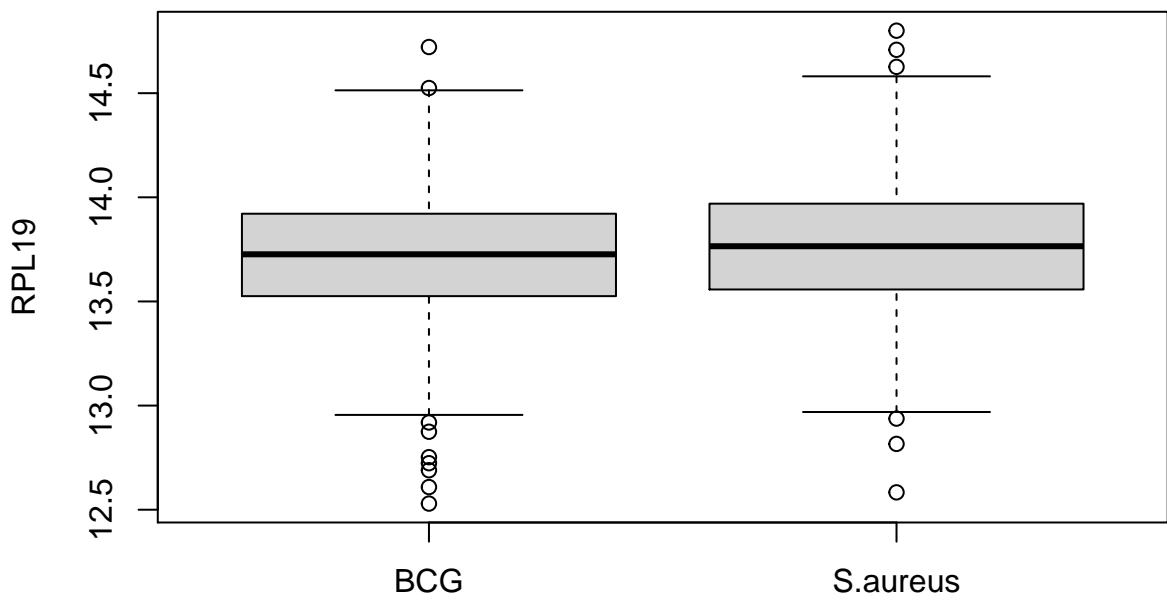
Stimulus.Name

```
boxplot(PTPRC_all~Stimulus.Name,data = expr3)
```



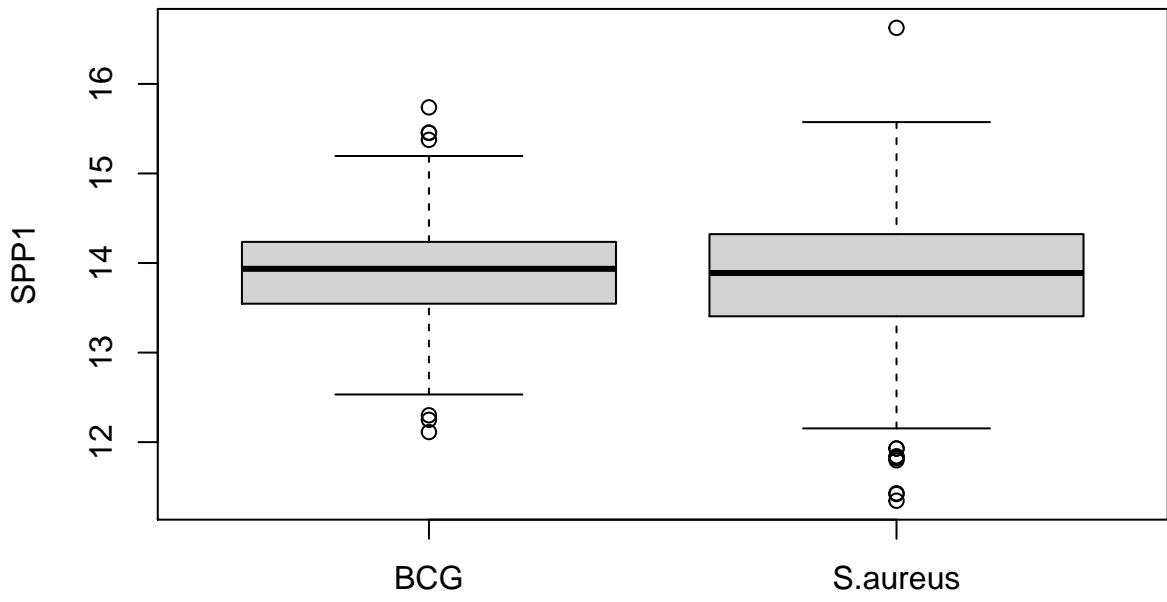
Stimulus.Name

```
boxplot(RPL19~Stimulus.Name,data = expr3)
```



Stimulus.Name

```
boxplot(SPP1~Stimulus.Name,data = expr3) #
```



Stimulus.Name

Nous sélectionnons les gènes les plus différenciés. Ils sont marqués par des #. Nous les stockons dans la liste suivante:

```
exprDiffMaxBCGSau <- c("CCL20", "CXCL1", "IL1A", "PTGS2", "SPP1")
```

Enfin, nous effectuons l'ACP sur ces gènes uniquement.

Nous obtenons ceci :

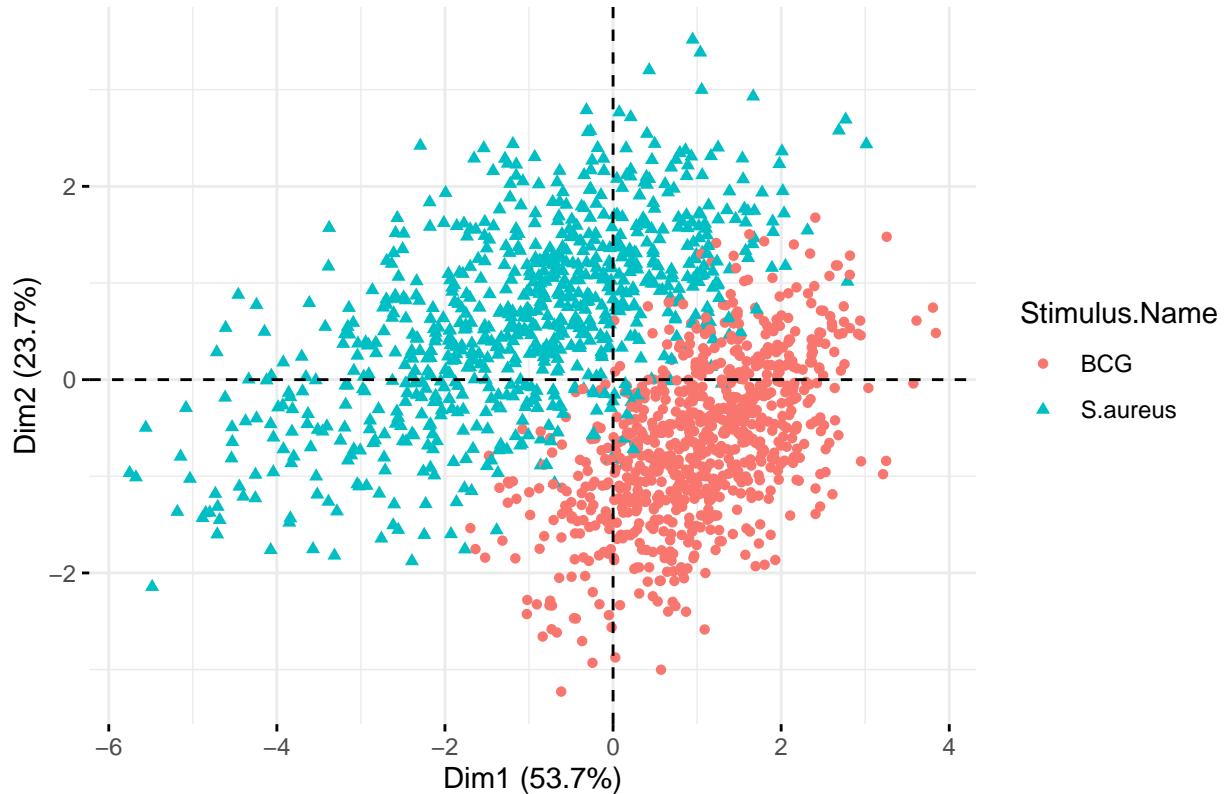
```
fviz_pca_ind(fit.pca,
  axes = c(1,2),
```

```

habillage = 'Stimulus.Name',
invisible = 'quali',
label ="none")

```

## Individuals – PCA



Nous avons réussi à séparer les deux stimuli !

Jetons un œil au site **genecards** pour analyser d'un peu plus près ces gènes. Il se trouve que le IL1A, CXCL1 et CCL20 sont directement liés à des infections touchant les poumons. Le fait qu'ils puissent discriminer la réponse contre le Staphylocoque Doré, et contre le vaccin contre la tuberculose, qui lui touche directement les voies respiratoires, n'est donc pas surprenant.

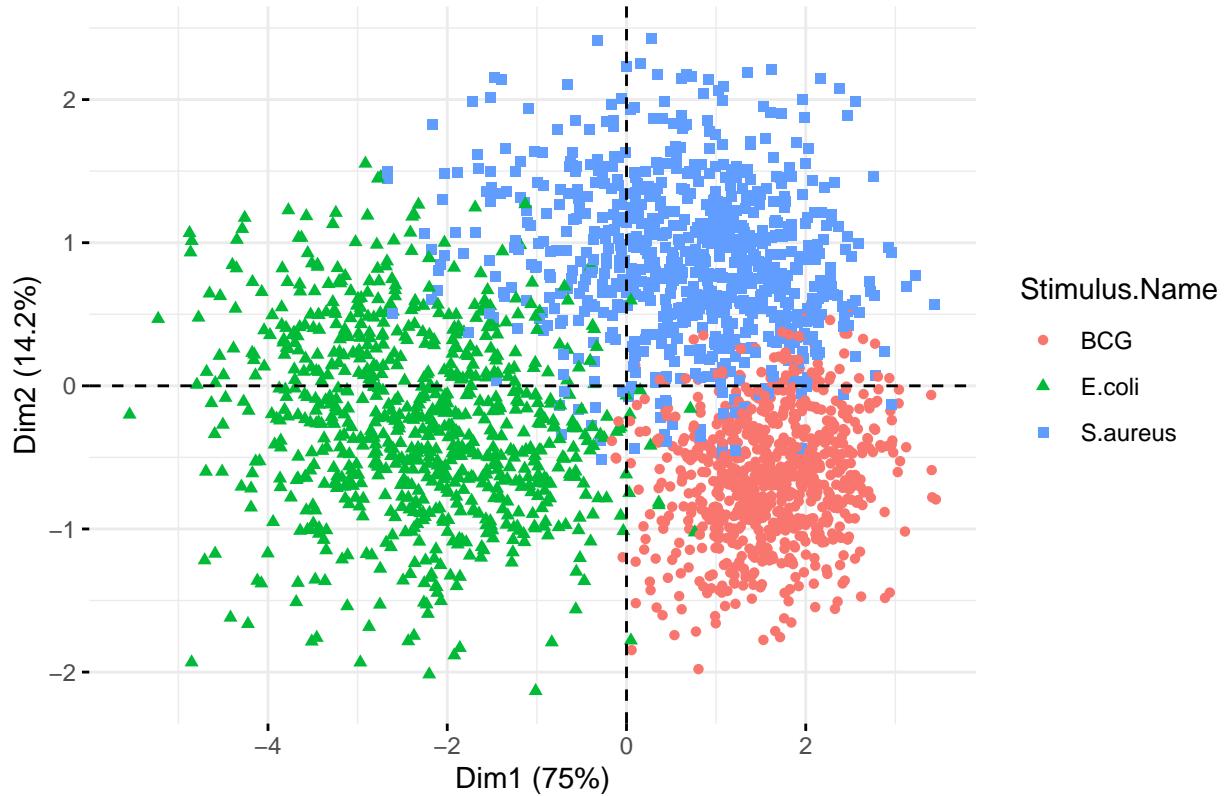
En réalisant l'ACP sur les trois stimuli, nous obtenons finalement ceci :

```

fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali', label = 'none')

```

## Individuals – PCA



Les trois stimuli sont enfin séparés de manière significative. Ainsi, ce groupe de gènes permet de différencier la réponse à ces trois stimuli.

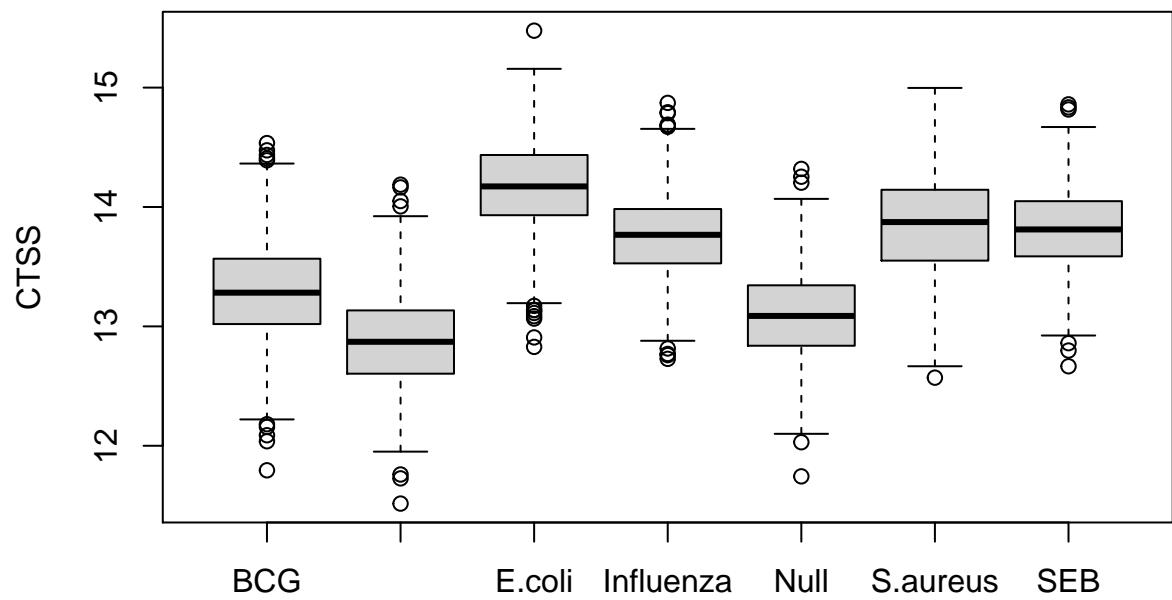
### 4.2.3 Retour sur le virus de la grippe

En reprenant le type d'étude que nous venons d'effectuer, nous pouvons identifier un plus petit groupe de gènes caractérisant la réponse au virus de la grippe. Pour rappel, nous avions identifié les gènes ci-dessous : `exprMaxHeatmapFlu`

```
## [1] "CTSS"      "CXCL10"     "CXCR4"      "FCGR3A_B"    "HLA.A"      "IFIT2"
## [7] "IFITM1"    "IL8"        "MX1"        "TNFSF10"
```

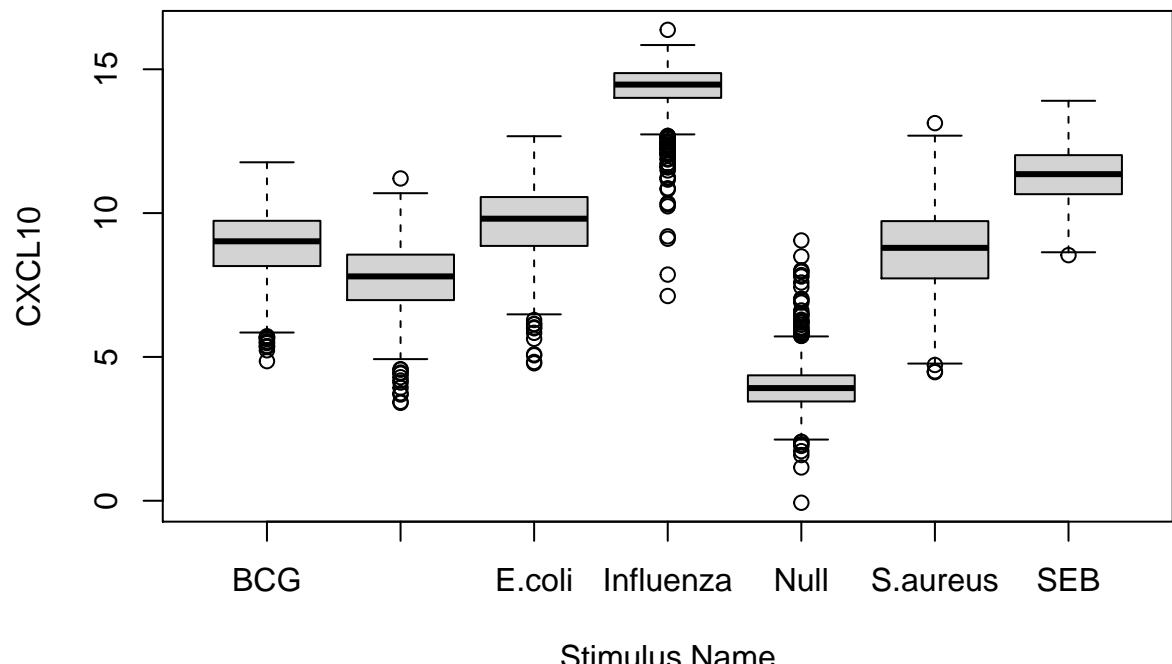
Effectuons un boxplot comprenant tous les stimuli, afin de choisir les gènes se différenciant le plus dans la liste ci-dessus.

```
boxplot(CTSS~Stimulus.Name, data = expr)
```



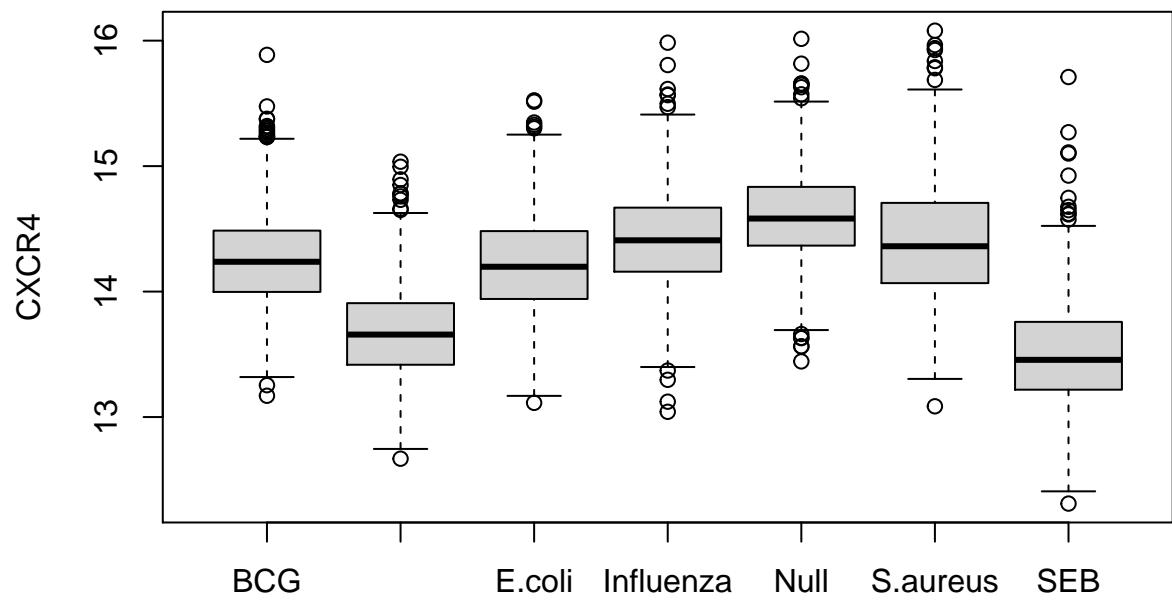
Stimulus.Name

```
boxplot(CXCL10~Stimulus.Name,data = expr) #
```

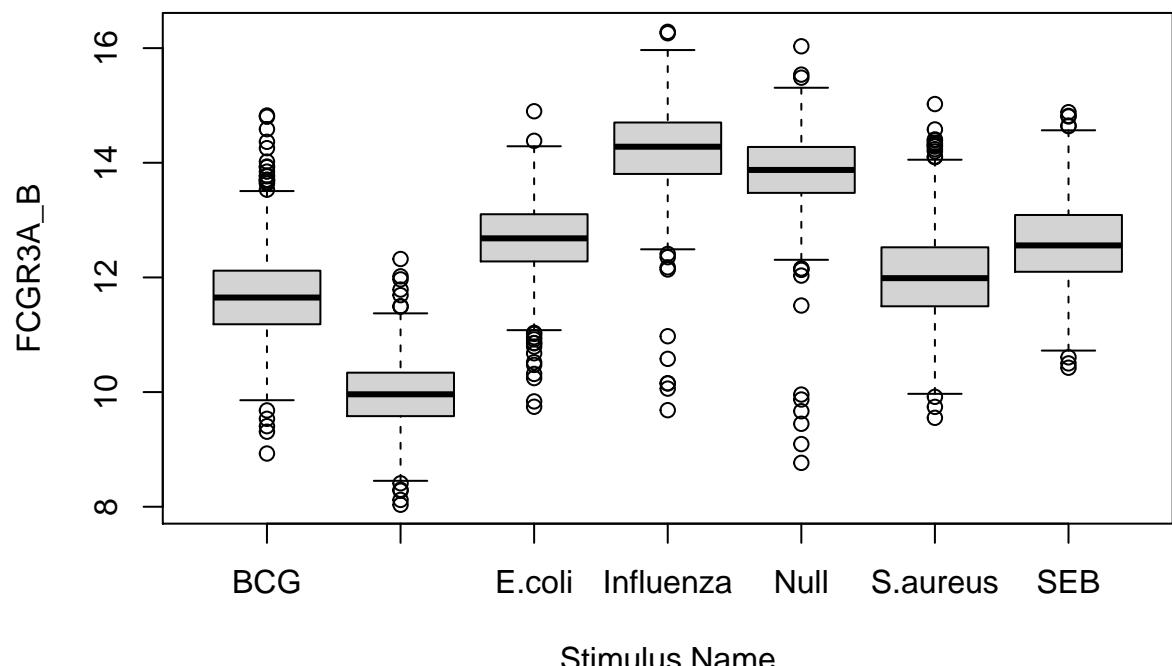


Stimulus.Name

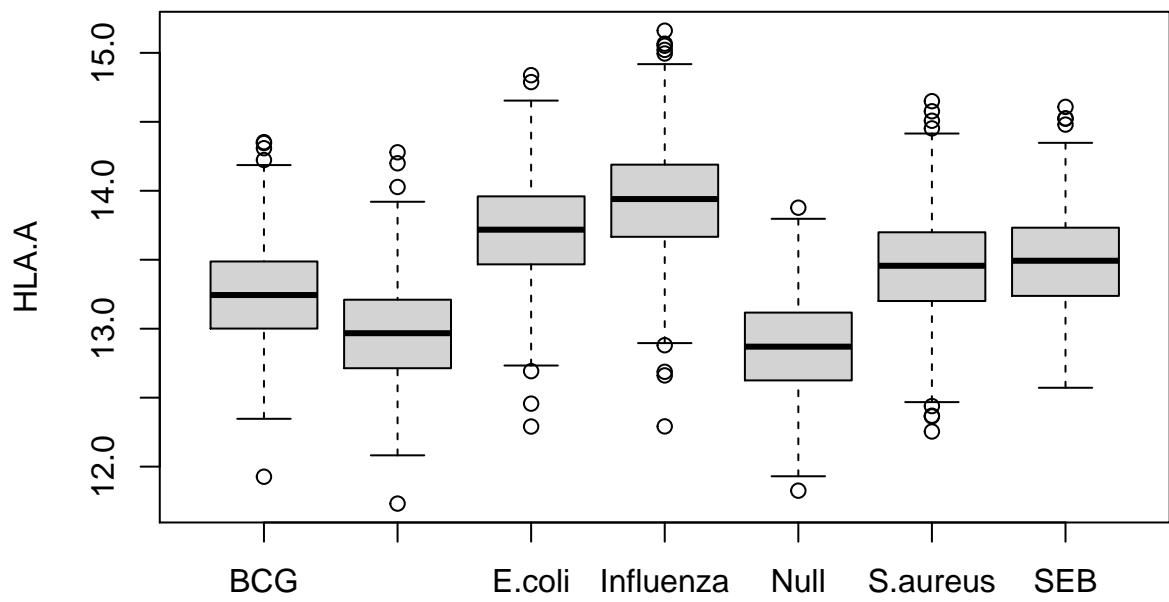
```
boxplot(CXCR4~Stimulus.Name,data = expr)
```



```
boxplot(FCGR3A_B~Stimulus.Name,data = expr)
```

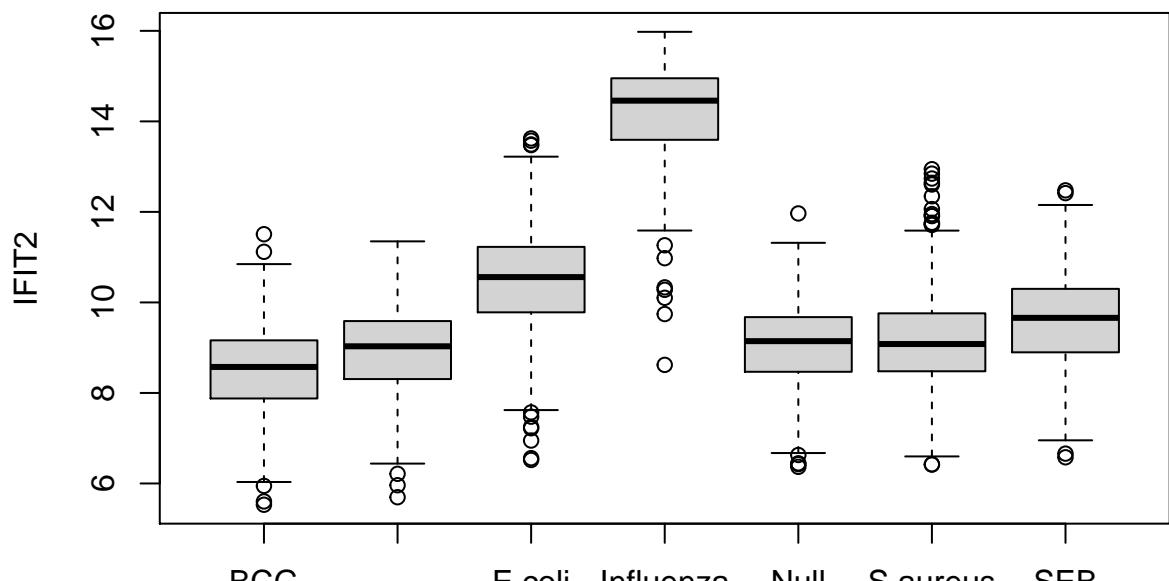


```
boxplot(HLA.A~Stimulus.Name,data = expr)
```



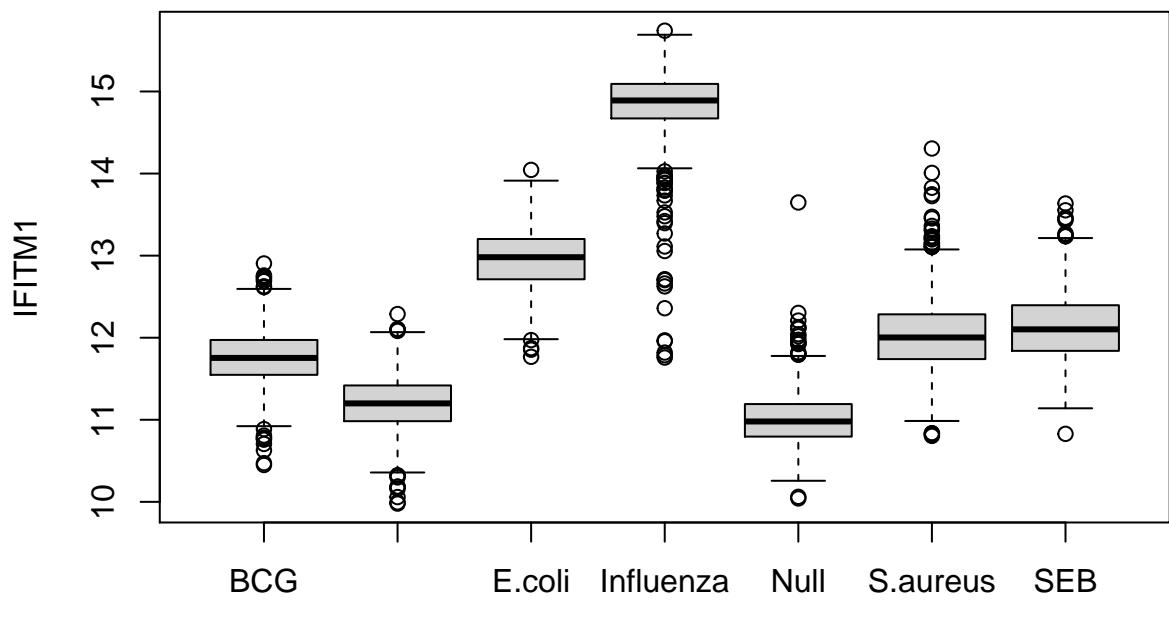
Stimulus.Name

```
boxplot(IFIT2~Stimulus.Name,data = expr) #
```

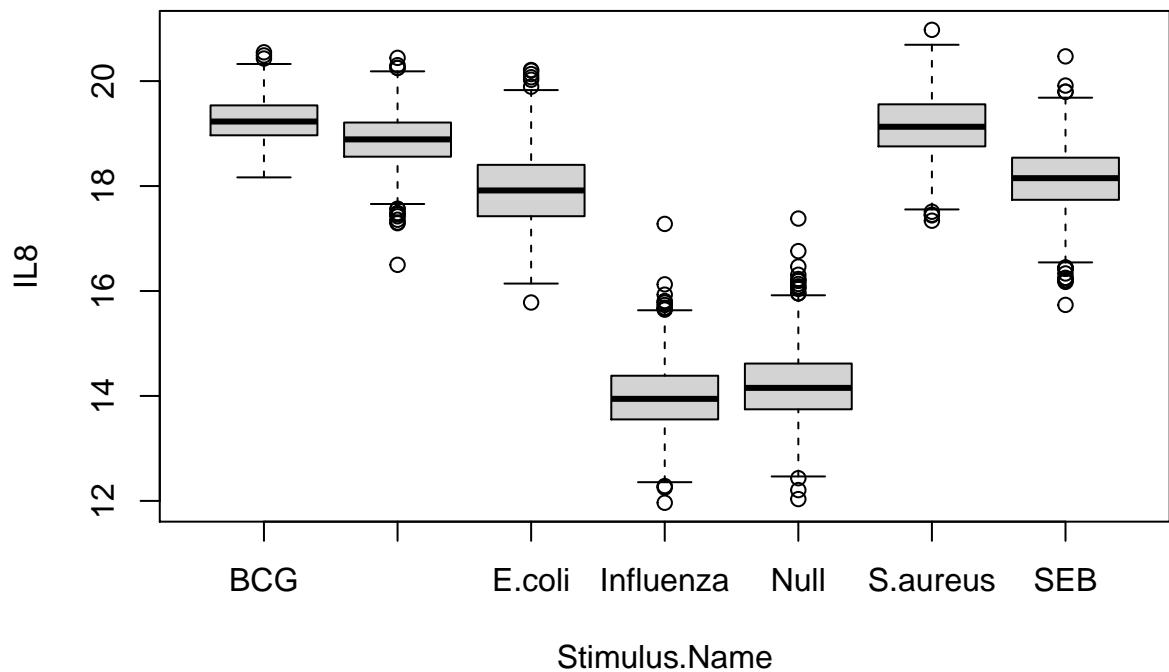


Stimulus.Name

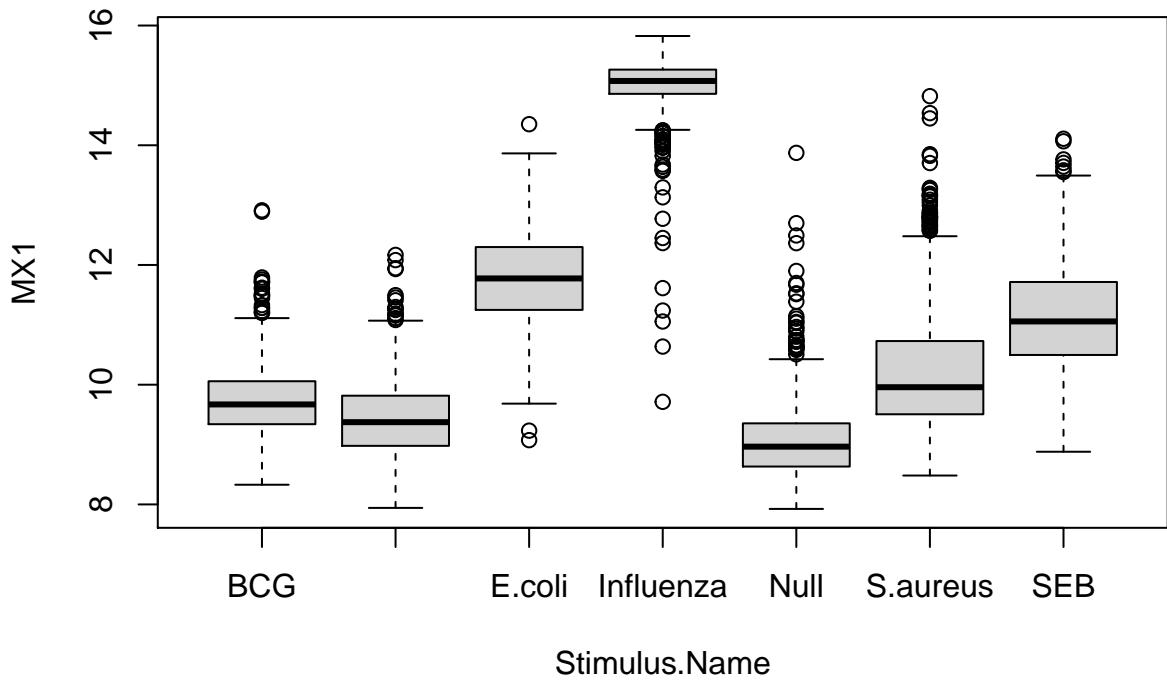
```
boxplot(IFITM1~Stimulus.Name,data = expr) #
```



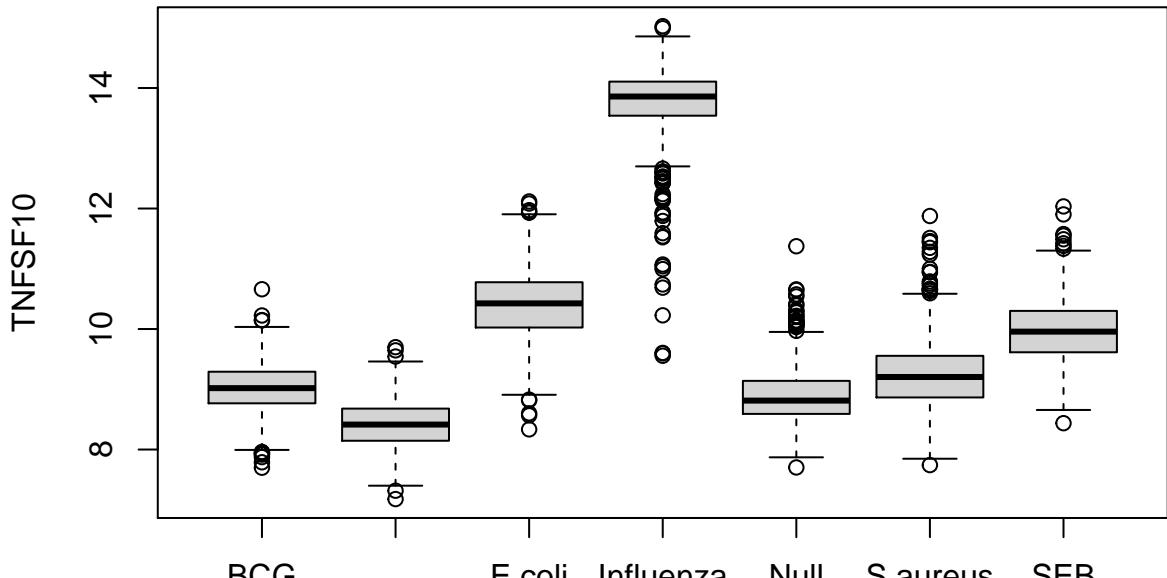
```
boxplot(IL8~Stimulus.Name,data = expr) #
```



```
boxplot(MX1~Stimulus.Name,data = expr) #
```



```
boxplot(TNFSF10~Stimulus.Name,data = expr) #
```

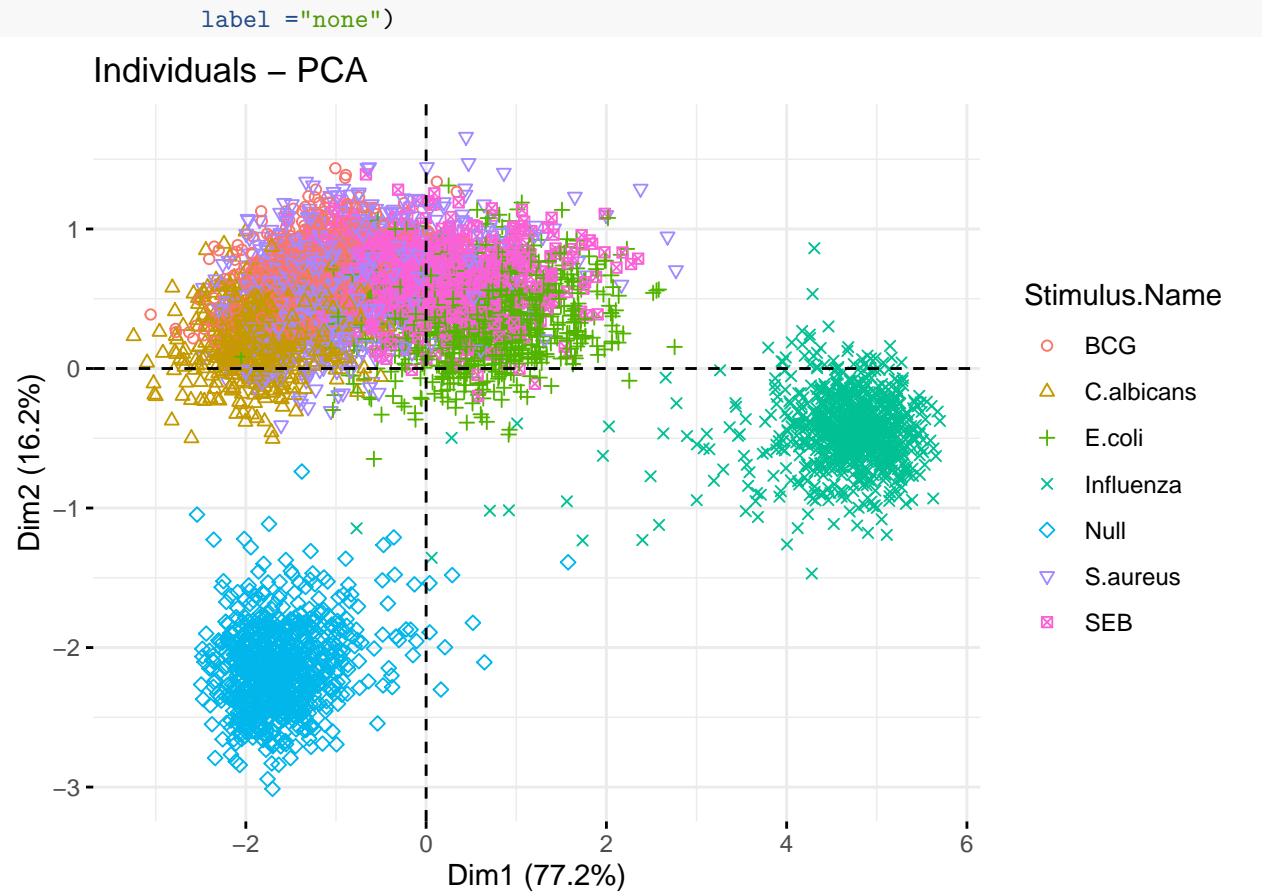


Nous marquons les gènes sélectionnés par un #. Voici la liste retenue.

```
exprDiffMaxFlu <- c("CXCL10", "IFIT2", "IFITM1", "IL8", "MX1", "TNFSF10")
```

Effectuons désormais l'ACP sur ces gènes:

```
fviz_pca_ind(fit.pca,
              axes = c(1,2),
              habillage = 'Stimulus.Name',
              invisible = 'quali',
```



Nous pouvons remarquer que le stimulus au virus Influenza est encore plus isolé ! La variance expliquée sur l'axe 1, axe où se discrimine notre stimulus d'intérêt, est élevée, valant 77,2 %, contre 56,7 %, valeur obtenue en premier lieu.

Si nous regardons l'influence des données de l'eCRF sur l'expression des gènes, nous pouvons voir qu'être vacciné contre la grippe influe beaucoup sur l'expression de nos six gènes (voir partie 4.2 sur la variance des critères de l'eCRF plus bas).

Enfin, jetons un œil au site [genecards](#) pour vérifier si ces gènes sont bien impliqués dans le mécanisme de réaction contre la grippe. Il se trouve que les gènes TNFSF10, MX1 et CXCL10 sont référencés comme impliqués directement dans le mécanisme de réaction contre la grippe. Nous pouvons donc conclure que le travail effectué sur le virus de la grippe a été très concluant.

## 5 Recherche d'individus présentant des réponses immunitaires similaires pour plusieurs stimuli

### 5.1 Réalisation d'une classification hiérarchique ascendante sur les individus pour chaque stimulation choisie

L'affichage de l'ACP a montré que l'on pouvait différencier facilement les stimuli d'origine virale (Influenza) et les stimuli d'origine bactérienne. Ici, on comparera les réponses immunitaires des stimuli de la seconde catégorie (E.Coli, BCG, C.Albicans, S.Aureus et SEB).

```

#on sélectionne seulement les lignes avec la stimulation choisie
expColi <- expr[expr$Stimulus.Name=='E.coli',]
expBCG <- expr[expr$Stimulus.Name=='BCG',]
expAlbi <- expr[expr$Stimulus.Name=='C.albicans',] #champignon
expAur <- expr[expr$Stimulus.Name=='S.aureus',]
expSEB <- expr[expr$Stimulus.Name=='SEB',] # stimulation un peu différente des précédentes

# on enlève les 2 premières colonnes avec n° patients et stimuli étudiés
expColigenes <- expColi[,-c(1:2)]
expBCGgenes <- expBCG[,-c(1:2)]
expAlbigenes <- expAlbi[,-c(1:2)]
expAurgenes <- expAur[,-c(1:2)]
expSEBgenes <- expSEB[,-c(1:2)]

# calcul de la distance euclidienne entre les individus avec la fonction dist()
eucl_EColi <- dist(expColigenes) #matrices des distances
eucl_BCG <- dist(expBCGgenes)
eucl_Albi <- dist(expAlbigenes)
eucl_Aur <- dist(expAurgenes)
eucl_SEB <- dist(expSEBgenes)

# On réalise un regroupement hiérarchique avec la distance euclidienne et la méthode
# de liaison "ward"
hc.ward_Coli <- hclust(eucl_EColi, method = 'ward.D')
hc.ward_BCG <- hclust(eucl_BCG, method = 'ward.D')
hc.ward_Albi <- hclust(eucl_Albi, method = 'ward.D')
hc.ward_Aur <- hclust(eucl_Aur, method = 'ward.D')
hc.ward_SEB <- hclust(eucl_SEB, method = 'ward.D')

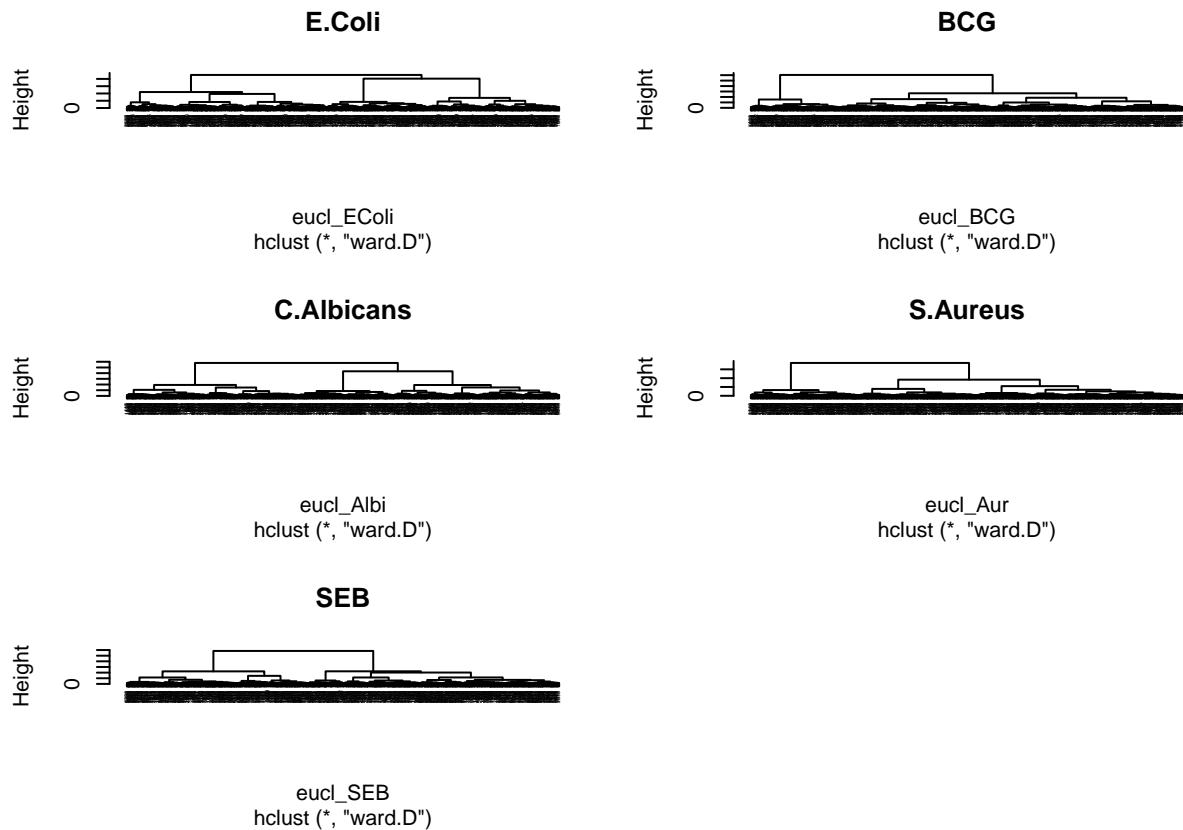
```

On affiche les dendogrammes correspondants

```

par(mfrow = c(3,2))
plot(hc.ward_Coli, main = 'E.Coli',cex=0.2)
plot(hc.ward_BCG, main = 'BCG',cex=0.2)
plot(hc.ward_Albi, main = 'C.Albicans',cex=0.2)
plot(hc.ward_Aur, main = 'S.Aureus',cex=0.2)
plot(hc.ward_SEB, main = 'SEB',cex=0.2)

```



L'arbre de classification nous indique en effet qu'il existe des groupes d'individus qui se ressemblent. On extrait cette information pour chaque stimulus. On choisit arbitrairement le nombre de clusters :  $k=5$  paraît un compromis entre une bonne classification et un nombre relativement restreint compte tenu des manipulations que l'on réalise ensuite. Il aurait sans doute été possible de rendre le code plus élégant et plus court, mais notre maîtrise du R reste récente et le temps relativement restreint pour réaliser cette étude nous a poussé à parfois privilégier la facilité afin de pouvoir réaliser plus de graphiques. Comme on cherchera à les regrouper pour 5 stimuli différents, 5 clusters différents paraissent suffire et permettent d'obtenir des graphiques lisibles. On pourra commenter ce choix à l'aide des conclusions de l'étude des caractéristiques (homogènes ou pas) des individus que l'on a regroupé.

```
# On récupère les groupes
k = 5 #on veut 3 groupes
clustersColi <- cutree(hc.ward_Coli, k)
clustersBCG <- cutree(hc.ward_BCG, k)
clustersAlbi <- cutree(hc.ward_Albi, k)

clustersAur <- cutree(hc.ward_Aur, k)
clustersSEB <- cutree(hc.ward_SEB, k)

# On affiche le nombre d'individus dans chaque groupe
table(clustersColi)

## clustersColi
##   1   2   3   4   5
## 189 128 245 152  91
```

```



```

## 5.2 Recherche des intersections entre les clusters à l'aide du package UpSetR

Le code étant parfois long et redondant, on laisse apparent les manipulations pour la première stimulation, le traitement étant exactement le même pour les 4 autres.

On a maintenant accès à la liste des SUBJID des individus appartenant à chaque cluster, pour chacun des 5 stimuli.

*#liste des SUBJID des individus appartenant à chaque cluster*

```

liste1 <- names(clustersColi[clustersColi==1])
liste2 <- names(clustersColi[clustersColi==2])
liste3 <- names(clustersColi[clustersColi==3])
liste4 <- names(clustersColi[clustersColi==4])
liste5 <- names(clustersColi[clustersColi==5])
indColi_1 <- expr$liste1]$SUBJID
indColi_2 <- expr$liste2]$SUBJID
indColi_3 <- expr$liste3]$SUBJID
indColi_4 <- expr$liste4]$SUBJID
indColi_5 <- expr$liste5]$SUBJID

```

On traite les données afin d'afficher le diagramme des intersections : on crée un tableau de booléens pour montrer l'appartenance d'un individu ou non à chaque cluster.

```

#on récupère les SUBJID des individus (différents des n° de lignes)
id =c()
for (x in expr$SUBJID)
  id <- append(id, x)

# df de booléens
df_clusters = tibble("SUBJID" = id,
                      "E.Coli_cluster1" = 0, "E.Coli_cluster2" = 0, "E.Coli_cluster3" = 0,
                      "E.Coli_cluster4" = 0, "E.Coli_cluster5" = 0, "BCG_cluster1" = 0,

```

```

"BCG_cluster2" = 0, "BCG_cluster3" = 0, "BCG_cluster4" = 0,
"BCG_cluster5" = 0, "Albi_cluster1" = 0, "Albi_cluster2" = 0,
"Albi_cluster3" = 0, "Albi_cluster4" = 0, "Albi_cluster5" = 0,
"Aur_cluster1" = 0, "Aur_cluster2" = 0, "Aur_cluster3" = 0,
"Aur_cluster4" = 0, "Aur_cluster5" = 0, "SEB_cluster1" = 0,
"SEB_cluster2" = 0, "SEB_cluster3" = 0, "SEB_cluster4" = 0,
"SEB_cluster5" = 0)

df_clusters <- df_clusters %>% distinct() #on enlève les individus dupliqués du
# dataframe df_clusters

for (x in indColi_1)
  df_clusters$E.Coli_cluster1[df_clusters$SUBJID==x] = 1
for (x in indColi_2)
  df_clusters$E.Coli_cluster2[df_clusters$SUBJID==x] = 1
for (x in indColi_3)
  df_clusters$E.Coli_cluster3[df_clusters$SUBJID==x] = 1
for (x in indColi_4)
  df_clusters$E.Coli_cluster4[df_clusters$SUBJID==x] = 1
for (x in indColi_5)
  df_clusters$E.Coli_cluster5[df_clusters$SUBJID==x] = 1

```

### Lecture du diagramme des intersections obtenu avec le package UpSetR

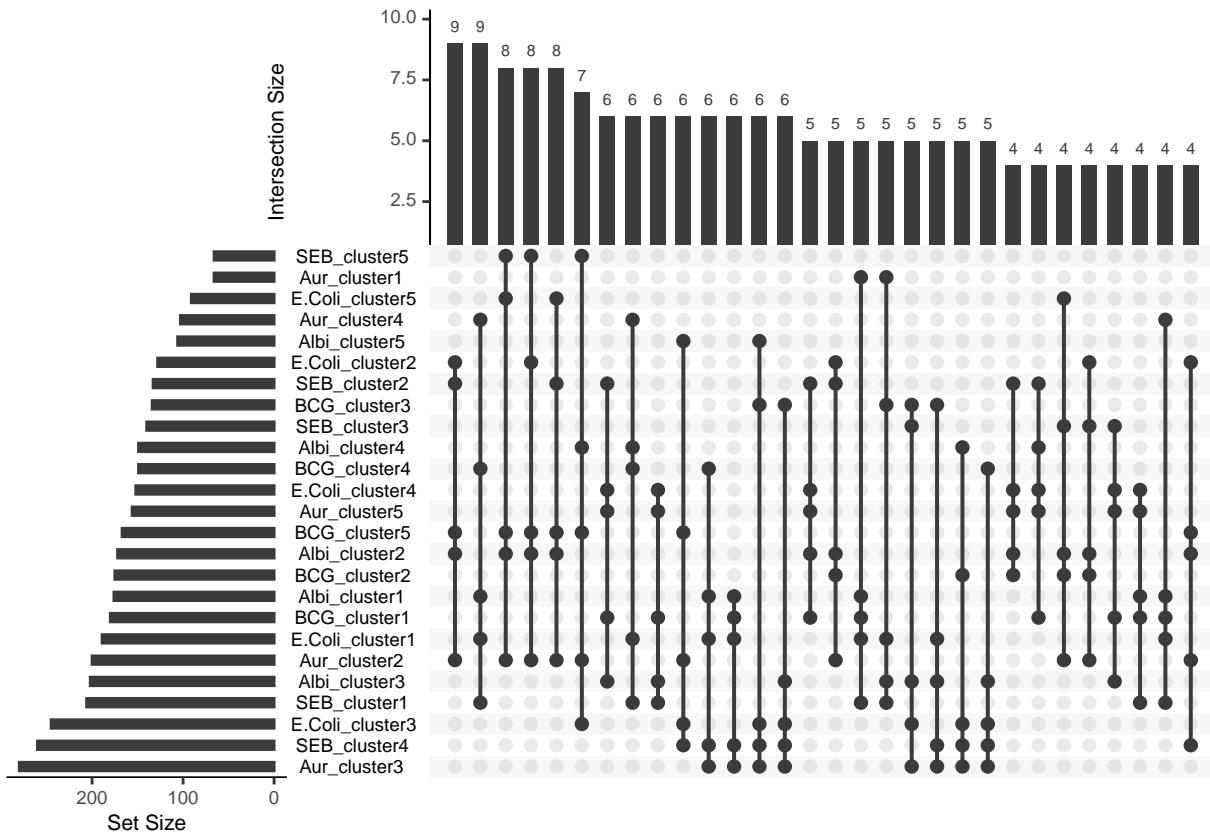
On a sur l'axe vertical la liste des 25 sous-groupes créées (5 clusters pour chacun des 5 stimuli). Lorsqu'un point est coloré, le sous-groupe considéré appartient au cluster. On lit sur l'axe horizontal le nombre d'individu appartenant à un sous-groupe donné.

```
df_2 <- df_clusters[,-c(1)] #on enlève les SUBJID pour pouvoir exécuter le plot
```

```

upset(as.data.frame(df_2), order.by = "freq", group.by = "degree",
      sets = c("E.Coli_cluster1", "E.Coli_cluster2", "E.Coli_cluster3",
              "E.Coli_cluster4", "E.Coli_cluster5", "BCG_cluster1", "BCG_cluster2",
              "BCG_cluster3", "BCG_cluster4", "BCG_cluster5", "Albi_cluster1",
              "Albi_cluster2", "Albi_cluster3", "Albi_cluster4", "Albi_cluster5",
              "Aur_cluster1", "Aur_cluster2", "Aur_cluster3", "Aur_cluster4",
              "Aur_cluster5", "SEB_cluster1", "SEB_cluster2", "SEB_cluster3",
              "SEB_cluster4", "SEB_cluster5"),
      nintersects = 30, decreasing = TRUE, mb.ratio = c(0.3, 0.7))

```



On discerne ainsi des sous-groupe de plusieurs individus qui appartiennent aux mêmes 5 clusters. Cependant, le nombre d'individus de chaque sous-groupe reste faible (8 ou 9 sur 805 individus étudiés). On peut ainsi dire que 5 clusters paraissent suffisants pour trier les individus (si l'on considère le choix des 5 stimuli pertinent). Il paraîtrait intéressant d'étudier les caractéristiques génétiques de chaque sous-groupe par rapport au reste des individus afin d'éventuellement discerner des traits spécifiques au groupe. L'étude que nous avons tenté par comparaison statistique et affichage de boxplot à partir des données de l'eCRF d'un sous-groupe donné ne donne pas de résultats intéressants (sur l'âge, l'activité physique ou le BMI par exemple).

## 6 Comparaison des variances expliquées par les critères de l'eCRF pour les populations cellulaires

On va calculer le pourcentage de variance expliquée, dans le stimulus Null, par les critères de l'eCRF pour les populations cellulaires.

```
expNull <- expr[expr$Stimulus.Name=='Null',]
y <- facets
df <- merge(eCRF, y, by.x="SUBJID", by.y="SUBJID" )
df <- df[complete.cases(df), ]
# head(df)

matrixvar <- matrix(0, nrow = 39, ncol = 76)
colnames(matrixvar) <- names(facets)[-1]
rownames(matrixvar) <- names(df)[2:40]

for (cell in 41:NCOL(df)) {
```

```

# Effectuer une régression linéaire avec tous les critères de l'eCRF
fitm <- lm(df[,cell] ~., data = df[,2:40],
            na.action=na.omit) # régression linéaire de l'expression des
                     # gènes en fonction des critères

fit <- anova(fitm, test = 'Chisq')
sumvar <- sum(fit$`Sum Sq`) # fit$`Sum Sq` : variance expliquée par chacun
                           # des critères de l'eCRF. En faisant la somme,
                           # on trouve la variance totale.
varexp <- fit$`Sum Sq` / sumvar # Pourcentages de variance expliquée.
varexp <- varexp[1:39] # La dernière valeur correspond à la variance
                     # non expliquée, on n'en a pas besoin.
matrixvar[,cell-40] <- varexp
}

```

On retient 20 critères importants pour faire les barplots.

```

Colors=brewer.pal(12,"Paired")

par(mfrow=c(2, 2), oma = c(0, 0, 2, 0))
nb.iter <- ceiling(ncol(matrixvar) / 20)

for (i in 0:(nb.iter-1)){
  #print(i)
  # Distinction de cas selon qu'on est ou non à la dernière itération
  if (i < (nb.iter - 1) & i!=0){
    m = matrixvar[, (i*20):((i+1)*20)]
  } else if(i == 0){
    m = matrixvar[,1:((i+1)*20)]
  } else{
    m = matrixvar[, (i*20):ncol(matrixvar)]
  }

  if (i==0){
    max_values = apply(m, 1, max) # pour chaque critère de l'ecrf,
                                # calcul de sa contribution maximale
    max_values_sorted = rev(sort(max_values))
    important = names(max_values_sorted[1:20]) #on prend les 20 contributions maximales
    matrix = m[important,]
  }

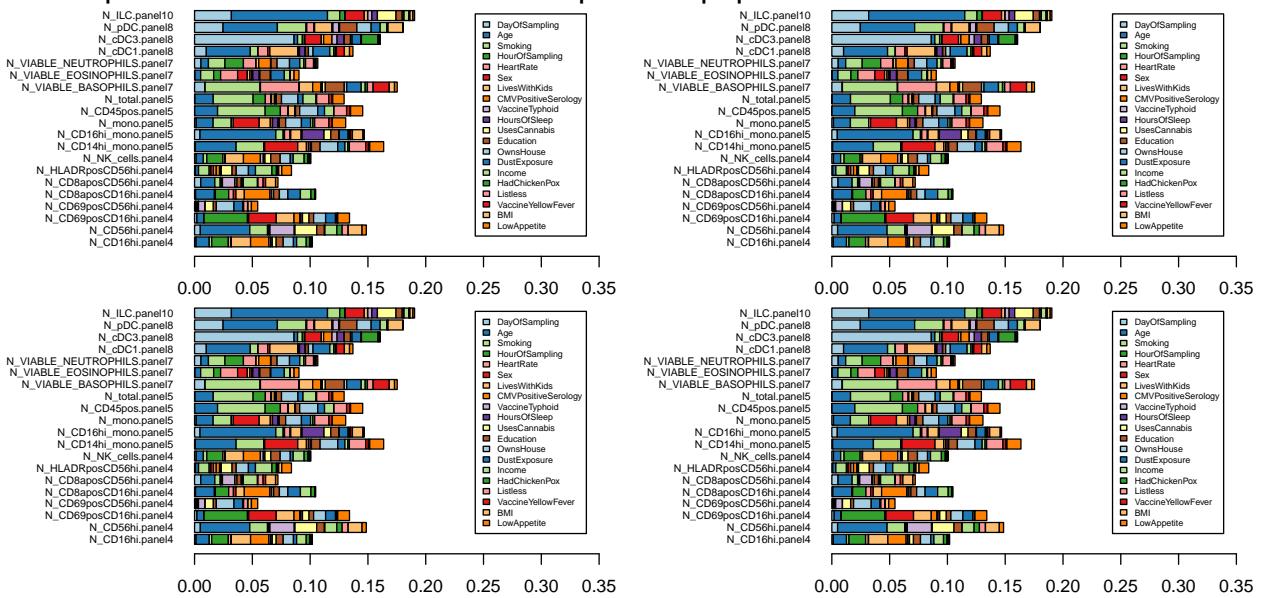
  # avec les critères importants
  par(mar=c(2,10,0,1)) #ajout des marges pour voir les labels
  barplot(matrix, legend.text = TRUE, beside = FALSE, horiz = TRUE, col = Colors,
          xlim = c(0,0.35), args.legend = c(cex=0.5), cex.names = 0.6, las = 1,
          mar=c(10,10,10,10))

}

mtext("Importances des critères de l'eCRF pour les populations de cellules de l'étude",
      outer = TRUE, cex = 1.5)

```

## Importances des critères de l'eCRF pour les populations de cellules de l'étude



## 7 Association des niveaux d'ARNm avec les données de l'eCRF

### 7.1 Régression linéaire pour expliquer le niveau d'ARNm des gènes et visualisation des p-valeurs

On effectue une régression linéaire pour expliquer le niveau d'ARNm des gènes de l'immunité en fonction des critères du questionnaire eCRF. On calcule ensuite les p-valeurs associées à chaque critère pour chaque gène et on les affiche à l'aide d'une heatmap.

Dans cette partie, le stimulus étudié est fixé. On prend ici l'exemple du stimulus nul.

```
expNull <- expr[expr$Stimulus.Name=='Null',] #Sélection du stimulus étudié
expNull <- expNull[,c(1,3:NCOL(expNull))] #On retire la colonne des stimuli
df <- merge(eCRF, expNull, by.x="SUBJID", by.y="SUBJID" ) #Concaténation avec l'eCRF
```

Nous avons fait le choix de stocker les p-valeurs dans une matrice `matrixPval` qui contient les gènes en colonne et les critères en ligne.

```
matrixPval <- matrix(0, nrow = 39, ncol = 560) #Génération d'une matrice qui contiendra
#les p-valeurs
colnames(matrixPval) <- names(expNull)[-1]
rownames(matrixPval) <- names(df)[2:40]

# Pour tous les gènes
for(gene in 41:NCOL(df)){
  # Pour tous les critères
  for (crit in 2:40) {
    # Régression linéaire avec le critère d'intérêt et les co-variables age (colonne 2)
    # et sexe (colonne 5)
    fitm <- lm(df[,gene] ~ df[,crit] + df[,2] + df[,5], na.action=na.omit)
    # Régression linéaire sans le critère d'intérêt
    fitm2 <- lm(df[,gene] ~ df[,2] + df[,5], na.action=na.omit)
    # tester si l'ajout du critère dans la régression entraîne une différence
```

```

# significative dans l'explication de la réponse
fit <- anova(fitm, fitm2, test = 'Chisq')
# récupérer les p-valeurs associées à chaque critère
matrixPval[crit-1,gene-40] <- fit[2,'Pr(>Chi)']
}
}

```

Les p-valeurs obtenues par anova sont ensuite ajustées grâce à la correction de Benjamini-Hochberg dans un but de réduire le False Discovery Rate (FDR). Notons  $R$  la variable aléatoire associée au nombre d'hypothèses rejetées. La correction de Benjamini-Hochberg cherche à borner à 5% le FDR défini par :

$$\text{FDR} = \mathbb{E} \left[ \frac{V}{R} \right]. \quad (2)$$

On trace alors la heatmap des p-valeurs ajustées.

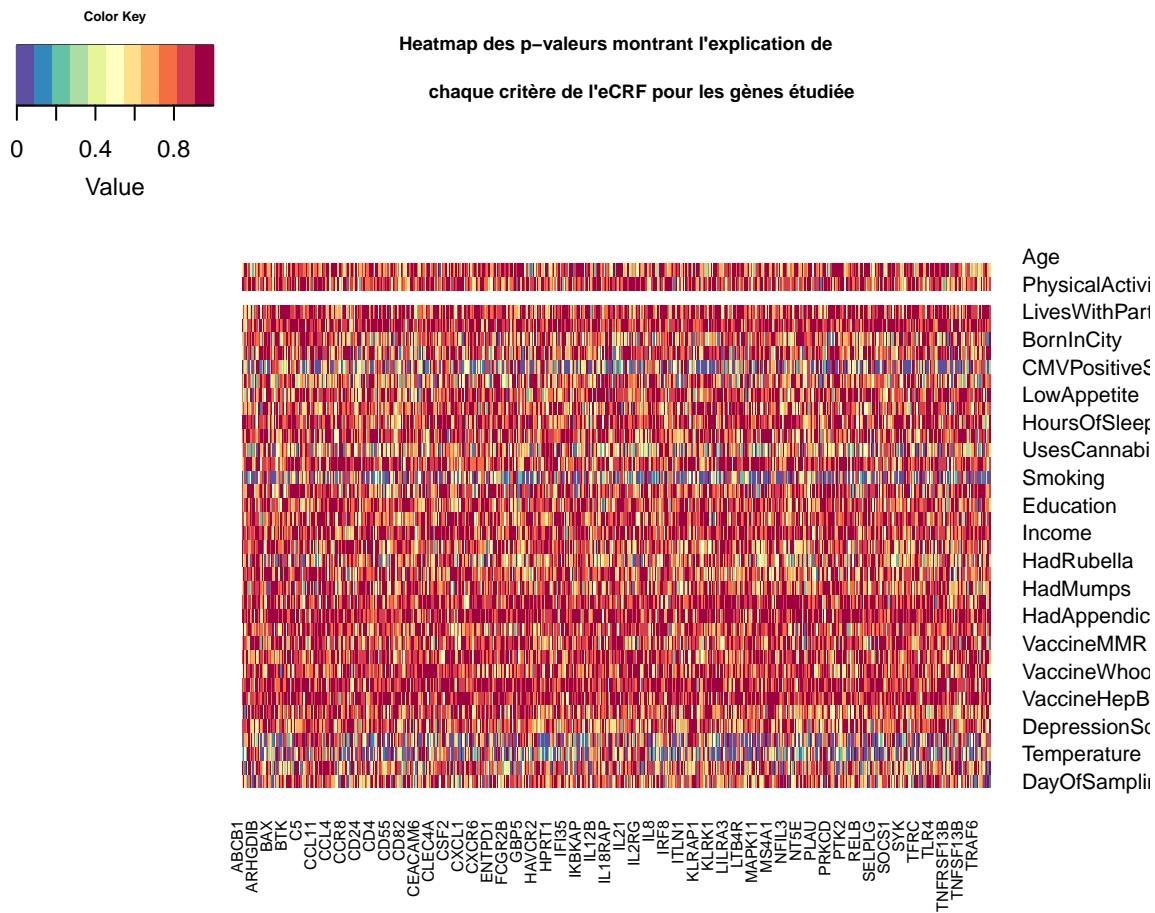
```

# ajuster les p-valeurs au nombre de tests réalisés
matPvalFDR <- matrix(nrow=nrow(matrixPval), ncol=ncol(matrixPval),
                      data=p.adjust(matrixPval, method='fdr'))
colnames(matPvalFDR) <- colnames(matrixPval)
rownames(matPvalFDR) <- rownames(matrixPval)

Colors=rev(brewer.pal(11,"Spectral"))
par(cex.main=0.5) # permet de réduire la taille du titre dans le rapport PDF

heatmap.2(matPvalFDR, col=Colors, density.info="none", dendrogram='none', Rowv=FALSE,
          Colv=FALSE,trace='none',
          main="Heatmap des p-valeurs montrant l'explication de\nchaque critère de l'eCRF pour les gènes étudiée")

```

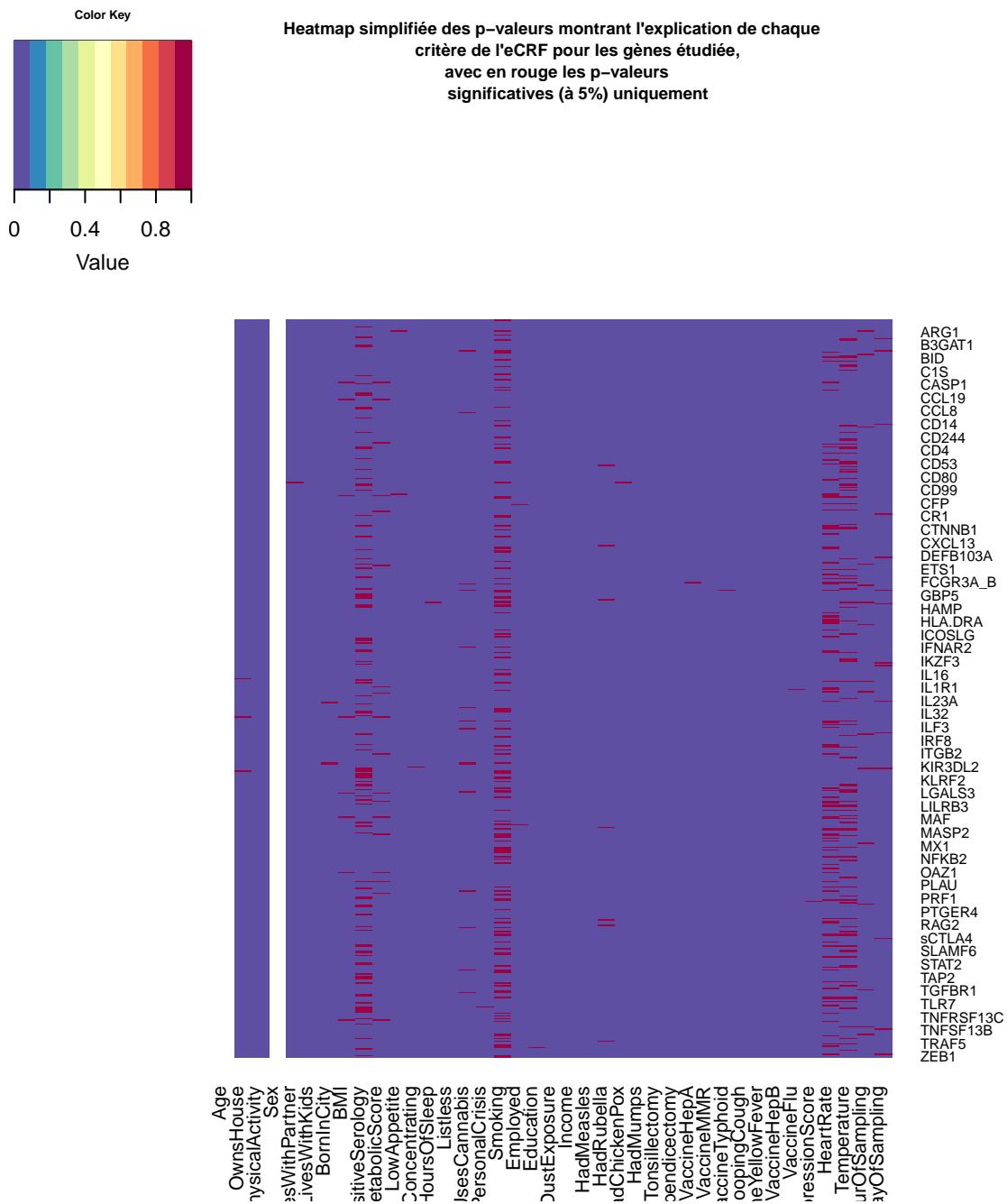


La heatmap étant très dense et peu lisible, on va la simplifier en sélectionnant les p-valeurs ajustées les plus significatives, c'est-à-dire inférieures à 0.05.

```
#selectionner les critères qui ont les p-valeurs ajustées les plus significatives
matPvalFDRselect <- apply(matPvalFDR, 1, function (x) (x< 0.05))
matPvalFDRselect [matPvalFDRselect == TRUE] <- 1
# heatmap(matPvalFDRselect, Colv=NA, Rowv=NA)

par(cex.main=0.5) # permet de réduire la taille du titre dans le rapport PDF

heatmap.2(matPvalFDRselect,col=Colors, density.info="none",dendrogram='none', Rowv=FALSE,
          Colv=FALSE,trace='none',
          main="Heatmap simplifiée des p-valeurs montrant l'explication de chaque critère de l'eCRF pour les gènes étudiée,\n avec en rouge les p-valeurs significatives (à 5%) uniquement")
```



On observe que les critères qui ont le plus d'influence (en dehors de l'âge et du sexe utilisés en co-variables) sont la positivité au test du Cytomegalovirus, le fait de fumer, la fréquence cardiaque et la température.

Nous allons ensuite évaluer l'association des niveaux d'ARNm avec les critères de l'eCRF grâce à leur variance expliquée.

## 7.2 Variance des niveaux d'ARNm expliquée par les critères de l'eCRF et la composition en cellules CD45, pour stimuli variables

On fait maintenant l'étude pour différents stimuli. Après une étude préliminaire dans la partie 3 (cf 2ème axe d'étude), nous avons choisi de nous restreindre à un sous-ensemble de critères qu'on a trouvé significatifs

et qui sont rassemblés dans la liste `l.criteres`.

```
# On crée des listes sur les variables sur lesquelles on itérera dans le chunk suivant
# dans la triple boucle. Ceci nous permettra d'avoir un code plus clair.
l.stimuli = c('Null', 'BCG', 'C.albicans', 'E.coli', 'Influenza', 'S.aureus', 'SEB')
l.genes.all <- names(expr)[-c(1:2)]
l.criteres <- c('Age', 'Sex', 'BMI', 'CMVPositiveSerology', 'Smoking', "HadMeasles",
                 "HadRubella", "HadChickenPox", "HadMumps", "HadTonsillectomy",
                 "HadAppendectomy", "VaccineHepA", "VaccineMMR", "VaccineTyphoid",
                 "VaccineWhoopingCough", "VaccineYellowFever", "VaccineHepB", "VaccineFlu")
```

On va stocker les données dans un tenseur 3D au lieu d'une matrice puisque l'on fait aussi varier les stimuli.

On initialise d'abord le tenseur `tensor.var` avec la bonne taille et que des zéros pour l'instant. On choisit pour la suite d'avoir : - les critères en lignes - les gènes en colonnes - les stimulus selon la 3ème dimension du tenseur.

```
tensor.var <- array(0, c(length(l.criteres)+1, ncol(expr)-2, length(l.stimuli)))
# rempli de 0 pour l'instant
# nombre de lignes = nombre de critère + 1 pour la cellule CD45
# nombre de colonnes = nombre de gènes (expr contient en plus SUBJID et stimulus)
```

On complète ensuite le tenseur progressivement, pour chaque stimulus, à l'aide d'une boucle selon la 3ème dimension de ce premier.

```
idx.z = 0
for (stimulus in l.stimuli){ # On traite stimulus par stimulus...
  idx.z = idx.z + 1
  #print(stimulus)
  # Génération du dataframe df :
  currentExpr <- expr[expr$Stimulus.Name==stimulus,]
  df <- merge(eCRF, currentExpr, by.x="SUBJID", by.y="SUBJID")
  facs.CD4 <- facs[names(facs) %in% c('SUBJID', "N_CD45pos.panel5")]
  df <- merge(facs.CD4, df, by.x="SUBJID",
              by.y="SUBJID")
  df <- df[c(l.criteres,l.genes.all, "N_CD45pos.panel5")]
  #
  # Le code qui suit est le même que précédemment sauf que les dataframes
  # ne contiennent pas les mêmes données à chaque itération de la boucle principale.

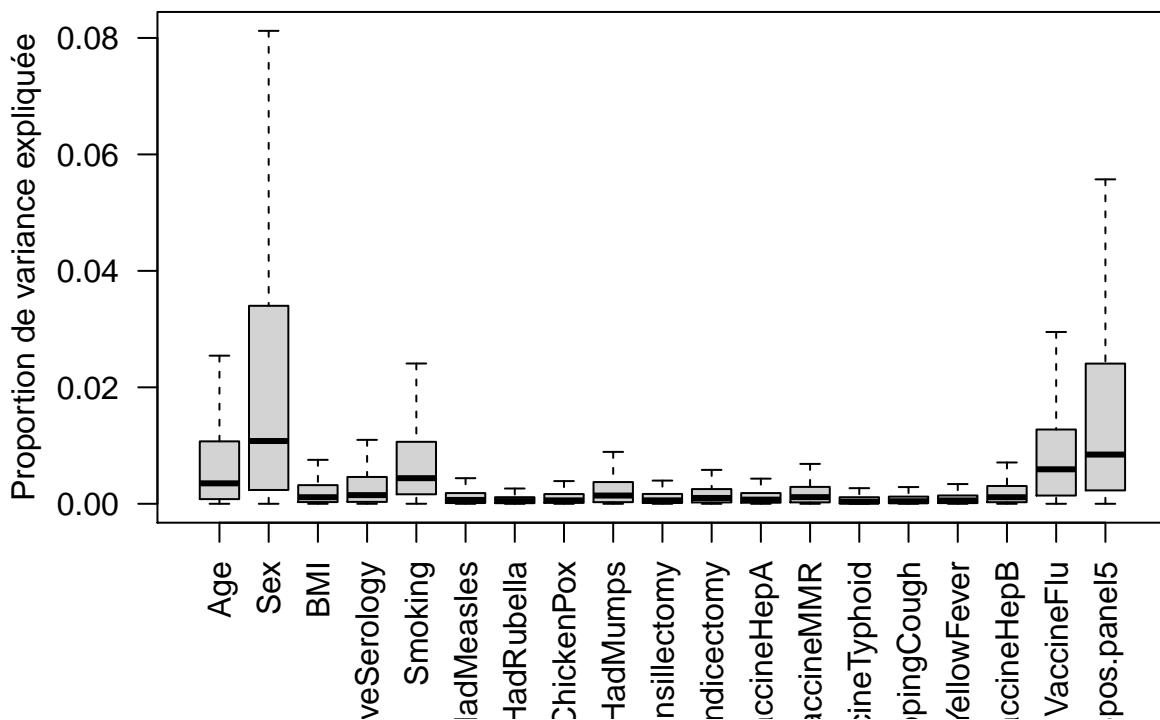
  n_gene = 0 # contient l'index du gène
  for(gene in l.genes.all){
    n_gene = n_gene + 1
    fitm <- lm(df[, gene] ~., data = df[c(l.criteres, "N_CD45pos.panel5")],
                na.action=na.omit)
    fit <- anova(fitm, test = 'Chisq')
    sumvar <- sum(fit$`Sum Sq`)
    varexp <- fit$`Sum Sq` / sumvar
    varexp <- varexp[1:(length(l.criteres) + 1)]
    tensor.var[,n_gene,idx.z] <- varexp
  }
  #
}
```

On va maintenant afficher pour chaque stimulus un boxplot de la variance des niveaux d'ARNm expliquée par chaque critère.

Par exemple, pour le virus de la grippe, on observe sur le graphe *Influenza* que le vaccin contre la grippe a une influence qui n'apparaît pas pour les autres stimuli, ce qui est cohérent avec l'efficacité du vaccin.

```
for (stimulus in 1:7){
  if (l.stimuli[stimulus]== "Influenza"){ #On affiche seulement la figure pour le virus Influenza par souci de lisibilité
    matrix.var <- tensor.var[, , stimulus]
    rownames(matrix.var) <- c(l.criteres, "N_CD45pos.panel5")
    colnames(matrix.var) <- l.genes.all
    boxplot.matrix(matrix.var, use.cols = FALSE, outline = FALSE, las = 2,
                  ylab = "Proportion de variance expliquée",
                  main = l.stimuli[stimulus])
  }
}
```

## Influenza



Plus généralement, nous observons que pour la grande majorité des stimuli étudiés, les critères les plus significatifs sont l'âge, le sexe, la positivité au test du Cytomegalovirus, le fait de fumer, et la composition en cellules CD45.

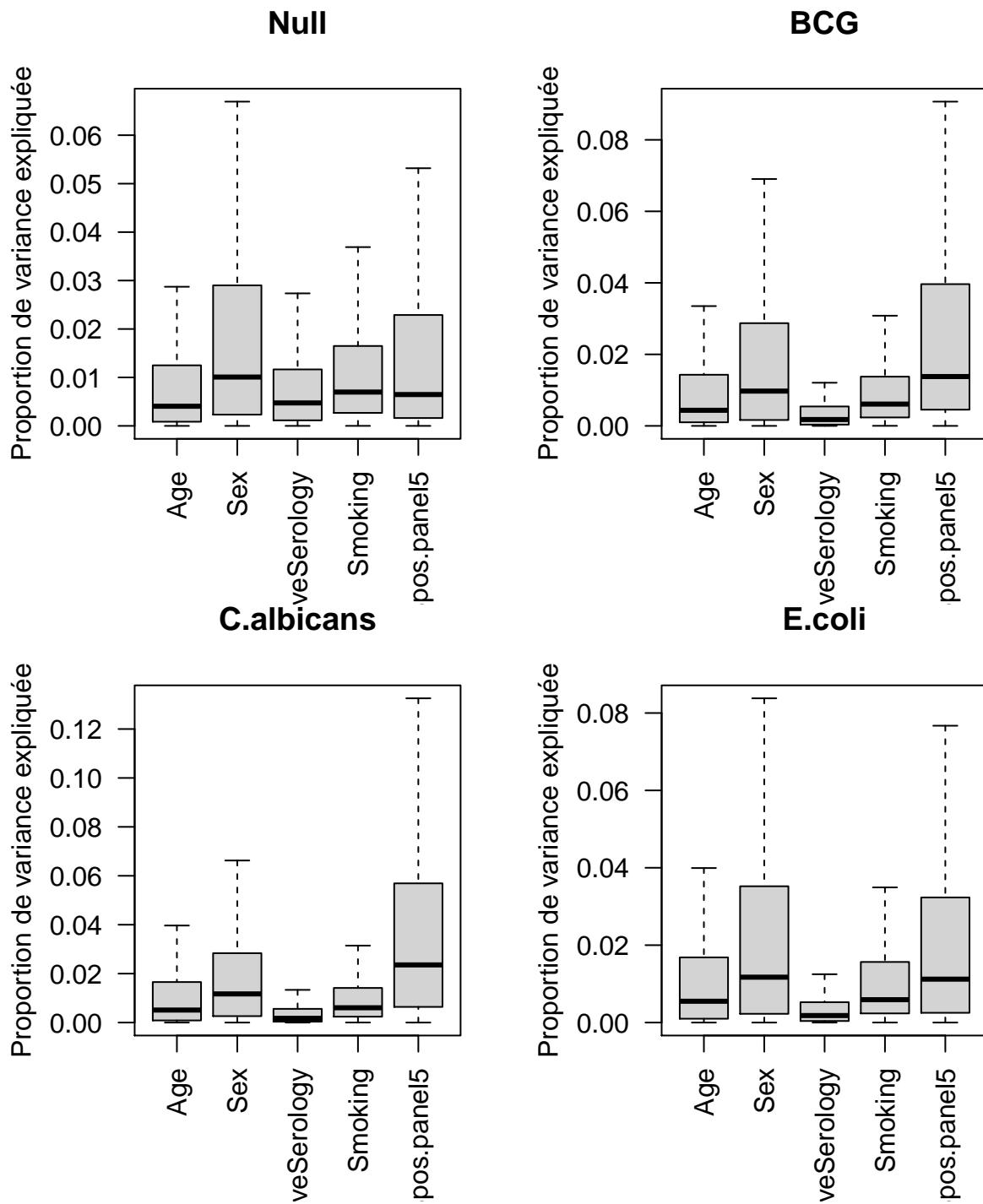
Afin de moins surcharger les graphes, nous avons retracé les figures précédentes en n'affichant que ces critères et les affichant côté à côté pour en faciliter la comparaison.

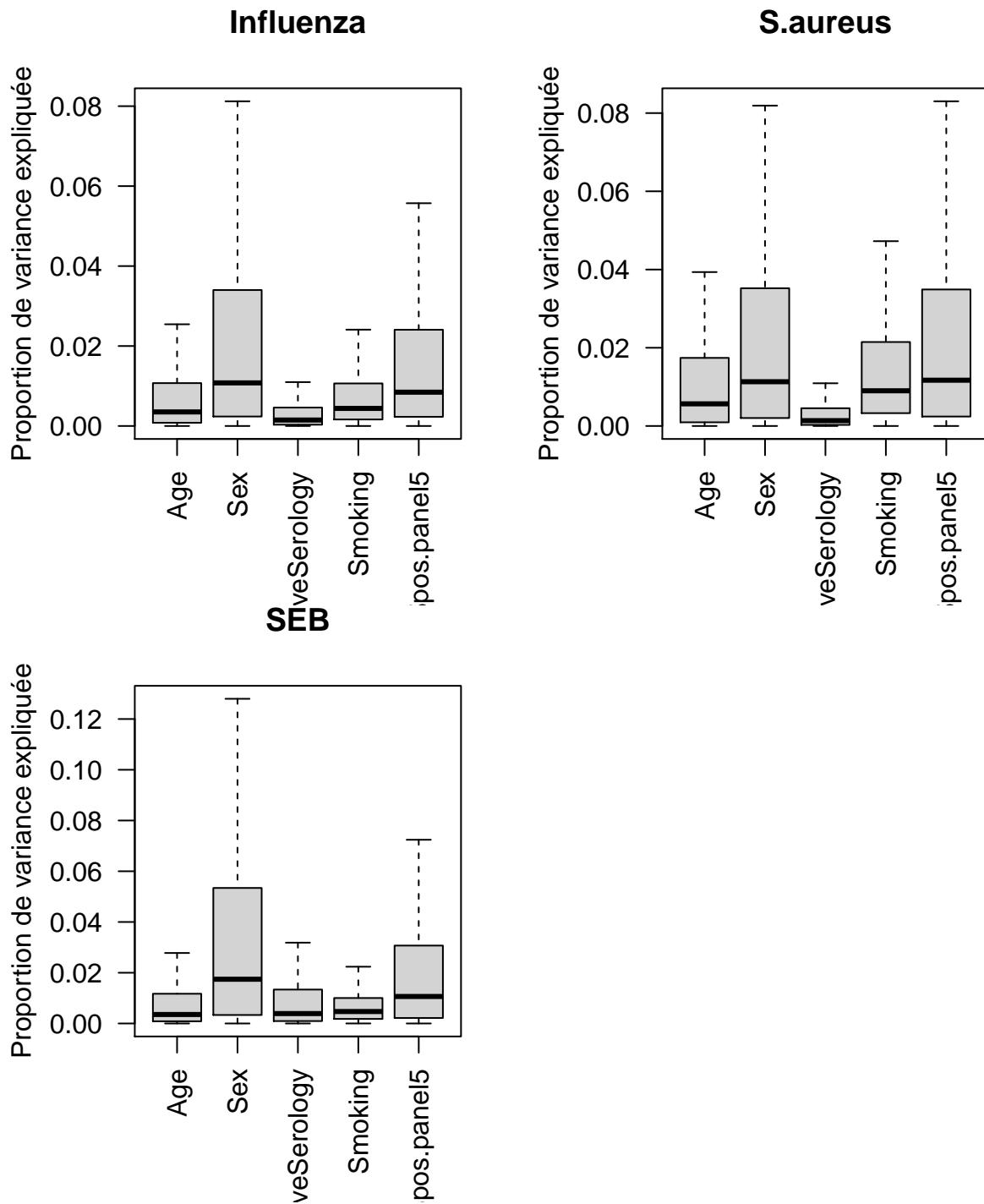
```
par(mfrow = c(1,2))
l.maincrit <- c('Age', 'Sex', 'CMVPositiveSerology', 'Smoking', "N_CD45pos.panel5")
# Critères les plus significatifs
for (stimulus in 1:7){
  matrix.var <- tensor.var[, , stimulus]
  rownames(matrix.var) <- c(l.criteres, "N_CD45pos.panel5")
  matrix.var <- matrix.var[l.maincrit,]
  # On affiche seulement les critères les plus significatifs
  colnames(matrix.var) <- l.genes.all
}
```

```

boxplot.matrix(matrix.var, use.cols = FALSE, outline = FALSE, las = 2,
               ylab = "Proportion de variance expliquée", main = l.stimuli[stimulus])
}

```





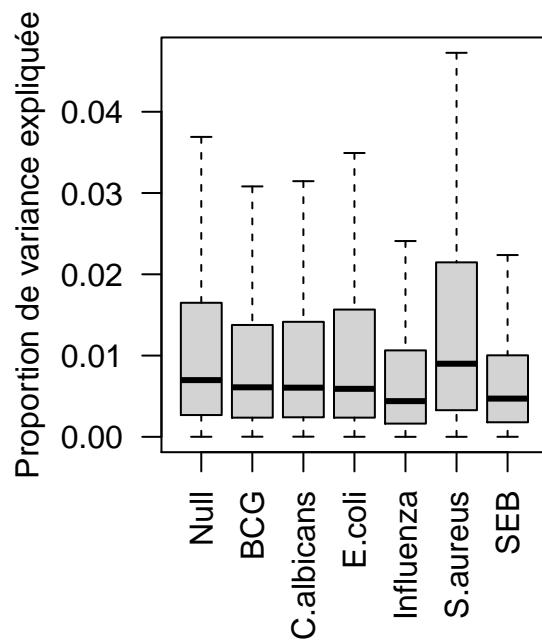
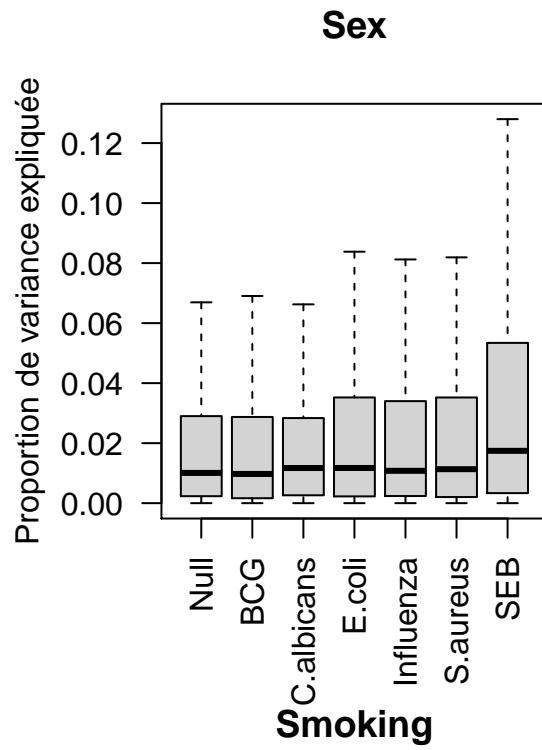
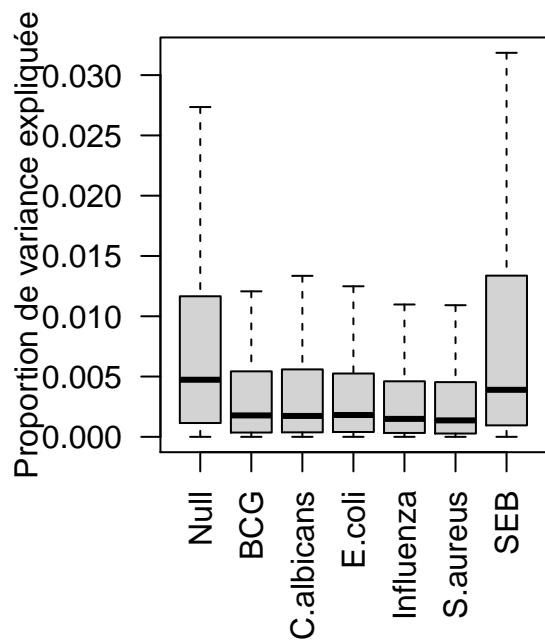
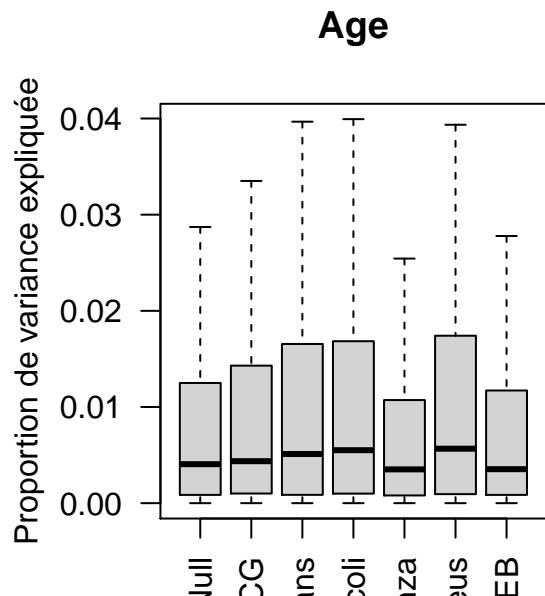
A l'inverse, nous pouvons afficher pour chaque critère la variance des niveaux d'ARNm expliquée par ce critère en fonction du stimulus considéré.

```
par(mfrow = c(1,2))
l.critCD = c(l.criteres,"N_CD45pos.panel5")
l.maincrit <- c('Age', 'Sex', 'CMVPositiveSerology', 'Smoking', "N_CD45pos.panel5")
for (crit in 1:19){
  if (l.critCD[crit] %in% l.maincrit){ #On va seulement afficher les critères sélectionnés précédemment
    matrix.var <- tensor.var[crit,,]
```

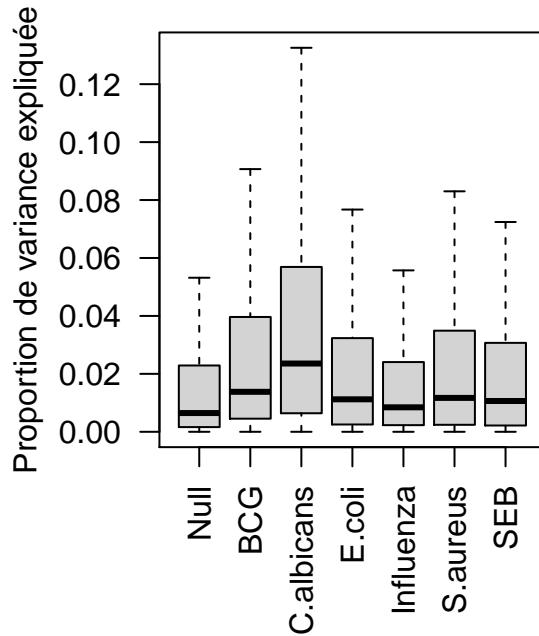
```

    rownames(matrix.var) <- l.genes.all
    colnames(matrix.var) <- l.stimuli
    boxplot.matrix(matrix.var, use.row = FALSE, outline = FALSE, las = 2, ylab = "Proportion de variance expliquée")
  }
}

```



## N\_CD45pos.panel5



### 7.3 Variance expliquée pour l'expression des gènes

On va maintenant chercher à calculer, pour une liste de gènes donnée, dans différents stimuli, la variance de l'expression du gène expliquée par différents critères. On va pour cela effectuer une régression linéaire de l'expression d'un gène en fonction des critères.

```
l.stimuli = c('Null', 'BCG', 'C.albicans', 'E.coli', 'Influenza', 'S.aureus', 'SEB')
# l.genes <- c('TLR1', 'TLR2', 'TLR4', 'MYD88', 'IRAK4', 'TRAF6', 'IRF5', 'IRF7')
l.genes <- c("CXCL10", "IFIT2", "IFITM1", "IL8", "MX1", "TNFSF10")
# Un groupe de gènes important dans la réponse au stimulus de la grippe, identifié
# dans l'axe 2
l.criteres <- c('Age', 'Sex', 'BMI', 'CMVPositiveSerology', 'Smoking', "HadMeasles",
               "HadRubella", "HadChickenPox", "HadMumps", "HadTonsillectomy",
               "HadAppendectomy", "VaccineHepA", "VaccineMMR", "VaccineTyphoid",
               "VaccineWhoopingCough", "VaccineYellowFever", "VaccineHepB", "VaccineFlu")
```

On va stocker les données des variances expliquées dans un tenseur 3D pour y avoir plus facilement accès. On ajoute 1 pour les critères car on va ajouter le nombre de cellules CD45+, qu'on sait être important dans l'expression des gènes.

```
tensor.var <- array(0, c(length(l.criteres)+1, length(l.genes), length(l.stimuli)))
# rempli de 0 pour l'instant
```

On procède maintenant au remplissage du tenseur `tensor.var` en procédant stimulus par stimulus, c'est-à-dire progressivement selon la 3ème dimension du tenseur.

```
idx.z = 0 # On va garder des indices pour les stimuli et les gènes à cause de
# l'impossibilité, à notre connaissance, de nommer les colonnes et les lignes d'un tenseur 3D
for (stimulus in l.stimuli){ # On traite stimulus par stimulus...
  idx.z = idx.z + 1
  # print(stimulus)
```

```

# Génération du dataframe df :
currentExpr <- expr[expr$Stimulus.Name==stimulus,]
df <- merge(eCRF, currentExpr, by.x="SUBJID", by.y="SUBJID")
facsf.CD4 <- facets[names(facets) %in% c('SUBJID', "N_CD45pos.panel5")]
df <- merge(facsf.CD4, df, by.x="SUBJID",
            by.y="SUBJID")
df <- df[c(l.criteres,l.genes, "N_CD45pos.panel5")]
# On ne garde que les données dont on a besoin.

n_gene = 0
for(gene in l.genes){
  n_gene = n_gene + 1
  fitm <- lm(df[, gene] ~ ., data = df[c(l.criteres, "N_CD45pos.panel5")],
              na.action=na.omit)
  # régression linéaire de l'expression des gènes en fonction des critères
  fit <- anova(fitm, test = 'Chisq')
  sumvar <- sum(fit$`Sum Sq`)
  # fit$`Sum Sq` : variance expliquée par chacun des critères de l'eCRF.
  # En faisant la somme, on trouve la variance totale.
  varexp <- fit$`Sum Sq` / sumvar # Pourcentages de variance expliquée.
  varexp <- varexp[1:(length(l.criteres) + 1)]
  # La dernière valeur correspond à la variance non expliquée, on n'en a pas besoin.
  tensor.var[,n_gene, idx.z] <- varexp
}
}

Colors=brewer.pal(12,"Paired")

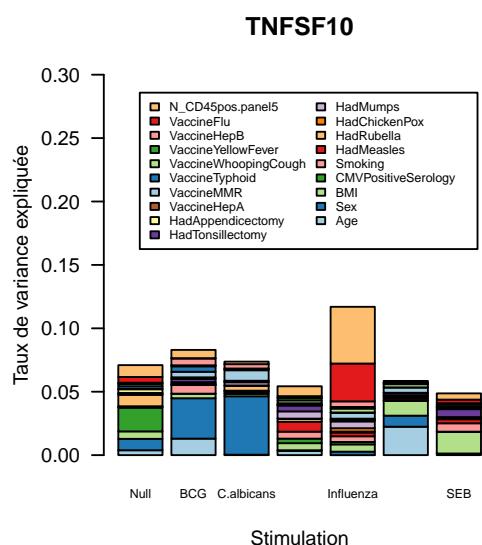
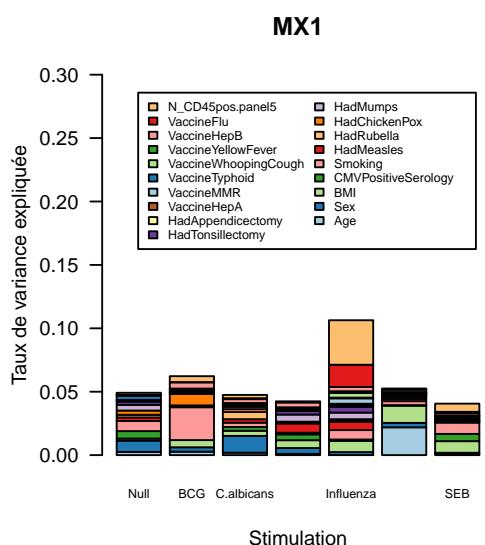
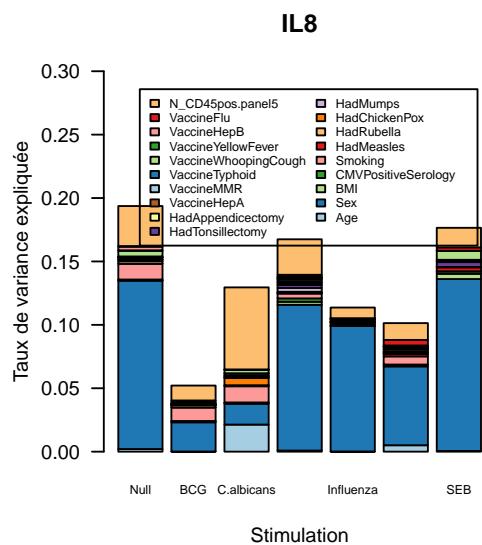
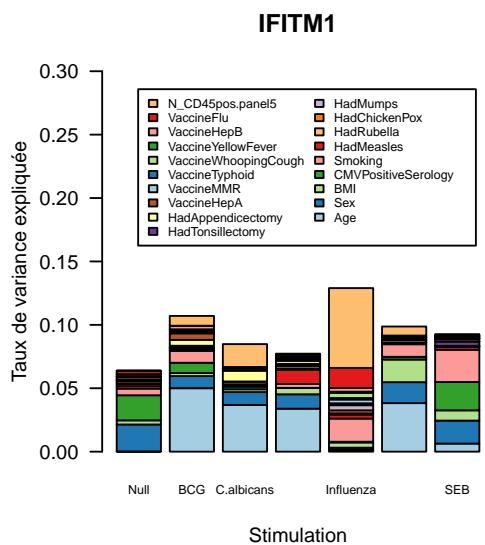
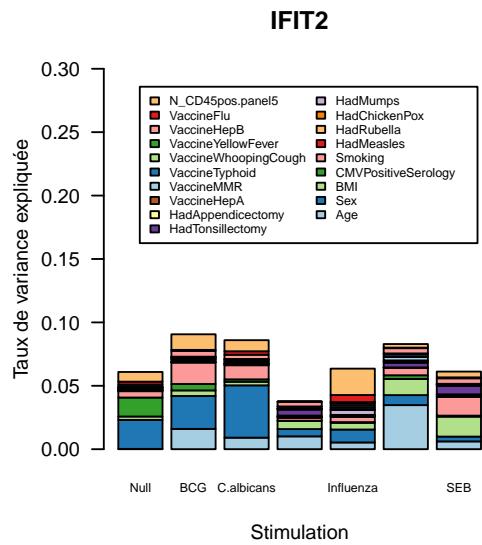
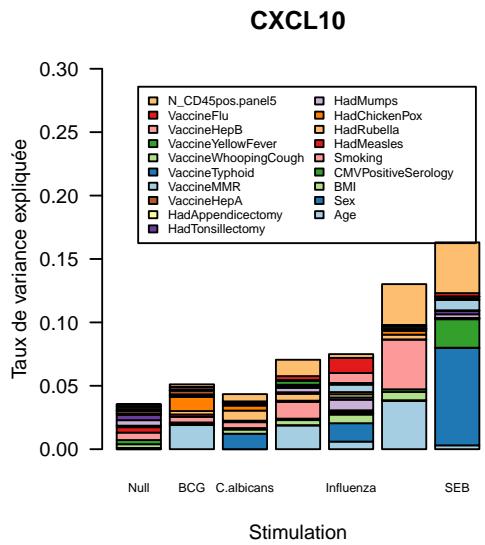
par(mfrow = c(3,2))

for (i in 1:length(l.genes)){
  m = tensor.var[,i,]
  colnames(m) <- names(l.stimuli)
  rownames(m) <- c(l.criteres, "N_CD45pos.panel5")

  # avec les critères importants
  barplot(m, legend.text = TRUE, beside = FALSE, horiz = FALSE, col = Colors,
          ylim = c(0,0.3), names.arg= l.stimuli, args.legend = c(cex=0.6, ncol=2),
          cex.names = 0.6, las = 1, main=l.genes[i], xlab = "Stimulation",
          ylab = "Taux de variance expliquée")
}

mtext("Variance expliquée du taux d'expression de gènes pour différents critères,
      dans différentes conditions de stimulation", side=3, outer=TRUE, cex = 0.9)

```



Nous pouvons voir que le vaccin contre la grippe joue un rôle important sur l'expression de ces gènes lors de la réponse à Influenza par rapport aux autres stimuli. Ce résultat confirme une nouvelle fois que ce groupe de gènes discrimine de manière significative la réponse au stimulus de la grippe.

## 7.4 Random Forest

Random Forest est un algorithme de machine learning particulièrement efficace pour prédire une variable quantitative ou qualitative. Elle permet également de mettre en avant les variables qui contribuent le plus à la prédiction de l'*outcome*. Random Forest est une méthode basée sur l'aggrégation d'arbres de décision.

On choisit ici de regarder les niveaux d'expression des gènes pour différents stimuli.

**1er exemple avec e-coli** Pour commencer, nous allons nous restreindre au cas sans stimulus.

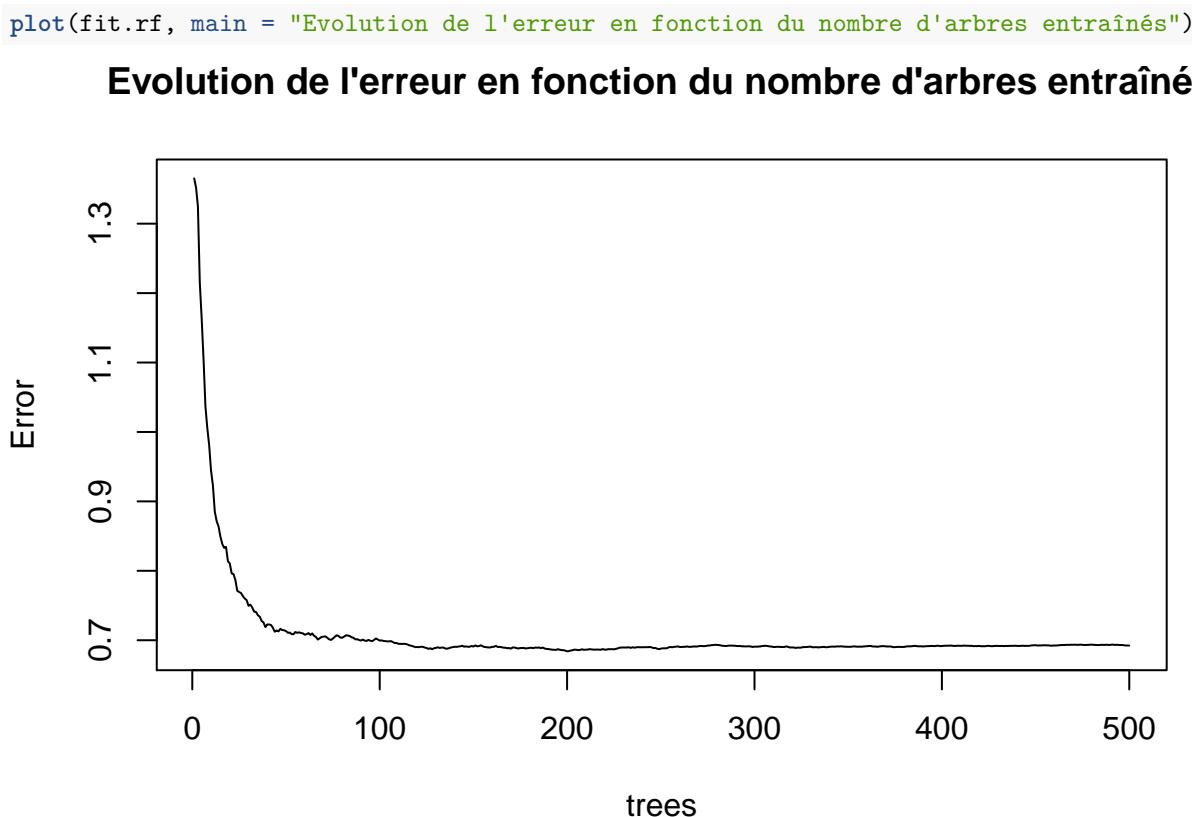
On étudiera ici le *MetabolicScore* et on essaiera de l'expliquer par une origine génétique.

```
df_2 <- merge(x=expr, y=eCRF, by='SUBJID')
df_2 <- df_2[df_2$Stimulus.Name == 'Null',]
df_2 <- df_2[,-which(names(df_2) %in% c('SUBJID', 'Stimulus.Name'))]
```

Le nombre d'arbre a été fixé à 500 (phase de stabilisation observée) et le nombre de variables testées à 24 (selon l'heuristique racine carrée du nombre de variables).

La valeur de 500 pour le nombre d'arbres est justifiée par le graphique suivant qui se stabilise autour de 500.

**Evolution de l'erreur en fonction du nombre d'arbres entraînés**



**Observations :** L'algorithme du *random forest* semble converger et atteindre sa valeur asymptotique dès 200 arbres entraînés. On choisira donc par la suite d'entraîner 200 arbres au lieu de 500 comme ici.

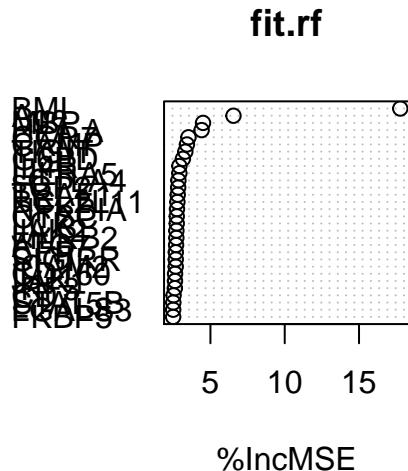
Par ailleurs, un résumé des qualités prédictives du Random Forest est reporté dans `str(fit.rf)`. Celui-ci n'est pas affiché ici pour des raisons de lisibilité du rapport.

En particulier, on obtient un MSE cross-validé de 0.6923035.

La méthode Random Forest permet également d'identifier les variables qui participent le plus à l'explication de la variable à expliquer. Le graphique suivant illustre le ranking basé sur la décroissante moyenne du MSE.

## Variables significatives dans l'explication du MetabolicScore

```
varImpPlot(fit.rf, type = 1)
```



**Interprétation :** On peut lire de haut en bas les features les plus importantes pour le *RandomForest*.

On observe tout d'abord que le BMI est nettement plus important que les autres covariables (25% de IncMSE environ contre moins de 10% pour les autres). Comme vu précédemment avec l'arbre de décision du MetabolicScore, cela vient du fait que le MetabolicScore est étroitement lié au BMI. Viennent ensuite l'âge et le MBP qui, sans surprise, ont une influence non négligeable sur le MetabolicScore.

### Résultats pour tous les stimuli de l'étude

```
par(mfrow=c(3, 2))
```

```

# On crée la figure qui contiendra les multiples plots.
for (stimulus in l.stimili){
  print(paste('Current stimulus : ', stimulus, sep=' '))

  # Mise en forme des données
  df <- merge(x=expr, y=eCRF, by='SUBJID')
  df <- df[df$Stimulus.Name == stimulus,]
  df <- df[,-which(names(df) %in% c('AGECAT','SUBJID'))]

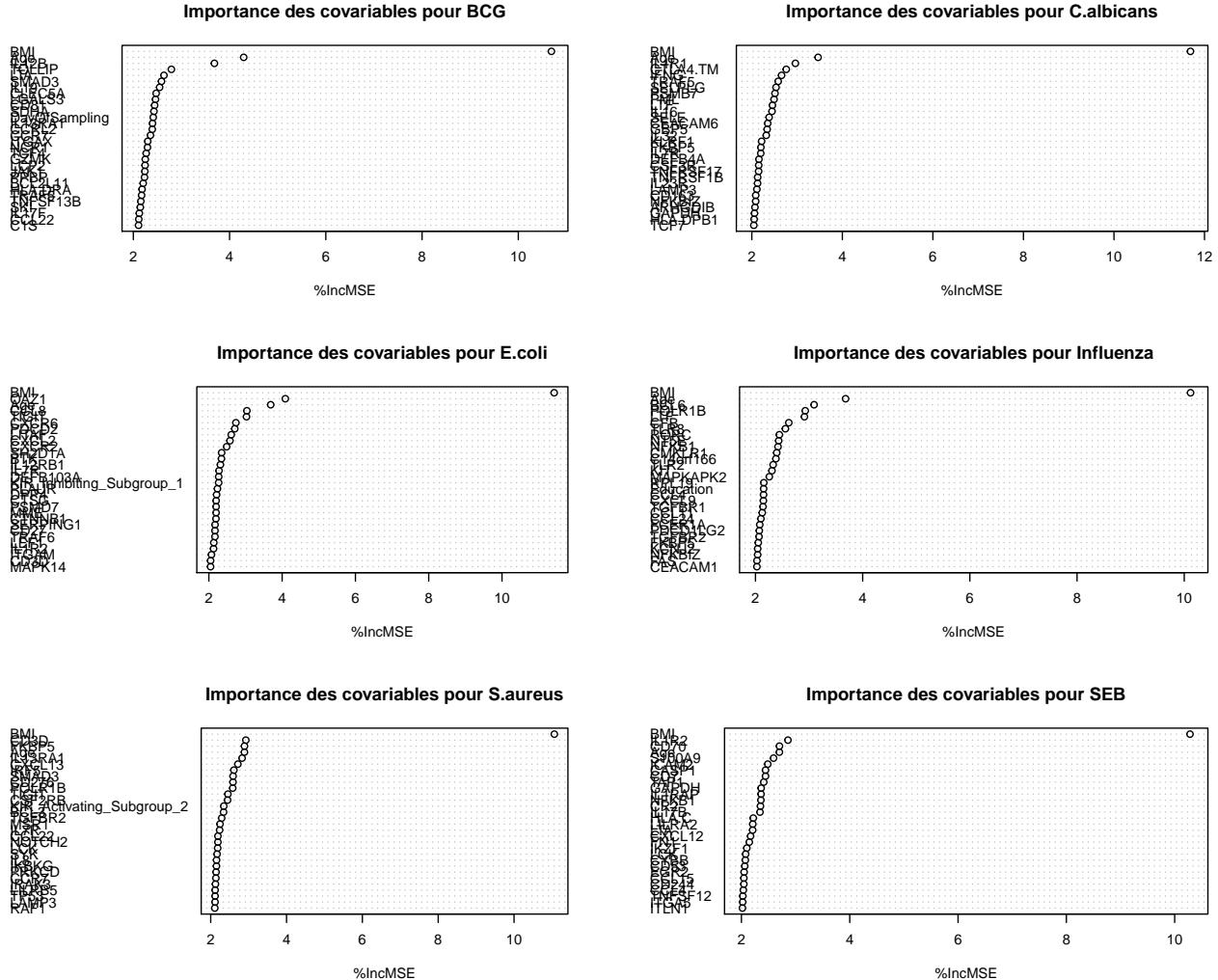
  # Random forest
  ntrees = 200
  mtry = floor(sqrt(NCOL(df_2)-1))
  fit.rf = randomForest(MetabolicScore ~ ., data = df, ntree = ntrees,
                        mtry = mtry, importance = TRUE)

  varImpPlot(fit.rf, type = 1, main=paste('Importance des covariables pour', stimulus, sep=' '))
}

## [1] "Current stimulus : BCG"
## [1] "Current stimulus : C.albicans"

```

```
## [1] "Current stimulus : E.coli"  
## [1] "Current stimulus : Influenza"  
## [1] "Current stimulus : S.aureus"  
## [1] "Current stimulus : SEB"
```



**Interprétation :** On retrouve encore une fois que l'âge et le BMI sont des covariables d'importance pour expliquer l'expression des gènes même si globalement, les gènes exprimés ont une importance variable pour différents stimuli.

## 8 Conclusion

Nous avons ainsi cherché à exploiter les données à notre disposition par différentes méthodes et à proposer une analyse statistique de la variabilité de la réponse immunitaire avec rigueur et précision. L'étude des niveaux d'expression des gènes d'une part et des individus du projet « Milieu Intérieur » d'autre part nous ont permis de mettre en pratique les outils étudiés au cours de la ST ‘Big Data & Santé’ mais aussi d'en découvrir de nouveaux. Nous maîtrisons bien mieux le langage de programmation R (ce mode de travail est complémentaire de l'étude plus théorique des cours de la séquence), et avons pu comprendre avec plus de précision les mécanismes étudiés dans le cadre de ce projet.

Nous tenions à remercier nos encadrants Violaine Saint-André, Vincent Guillemot et Arthur Tenenhaus qui nous ont aidé dans notre démarche malgré le travail en distanciel qui aurait pu compliquer nos échanges.