

# LITO



Présentation Flash : Graphiques sous R

# LITO

- Librairie ggplot2
  - Set de données (ggplot)
  - Définition échelles, légendes et thème
  - Type de graphique
  - Subplot avec gridExtra
- Graphique ACP
- Heatmap
  - Corrplot
  - aheatmap

**ggplot2** v3.3.0 Other versions ▾ NaN ⓘ Monthly downloads > 99.99th Percentile

by [Hadley Wickham](#)

[View Source](#)



<https://www.rdocumentation.org/packages/ggplot2>

Copy

<https://www.rdocumentation.org/packages/ggplot2>

**Initialisation du graphique**  
**Set de données**



**Type de graphique**



**Echelles / Légendes /  
Thème**

# Initialisation / Set de données

Fonction ggplot :

- Entrée = set de données (format dataframe)
- Initialise le graphique

---

## Exemple :

**DATA =**

| n° patient | CONVENTIONAL .min | CONVENTIONAL .min harmonized | ATRX |
|------------|-------------------|------------------------------|------|
| 1,00       | 11,70             | 11,47                        | 0    |
| 2,00       | 42,70             | 40,09                        | 0    |
| 3,00       | 24,56             | 25,41                        | 0    |
| 4,00       | 25,20             | 26,11                        | 1    |
| ...        |                   |                              |      |
| 62,00      | 17,10             | 16,99                        | 1    |

ggplot(DATA) + ...

Initialisation du graphique  
Set de données



Type de graphique



Echelles / Légendes /  
Thème

# Type de graphique

## `geom_boxplot` :

- `aes (X,Y)`
- `outlier.alpha >>>` mettre à 0 pour retirer les points extrêmes
- `notch >>>` TRUE pour créer un boxplot type notch (affiche un intervalle de confiance autour de la médiane)
- `fill >>>` couleur des boîtes (peut prendre une liste en argument pour colorer différemment les boîtes)
- `color >>>` couleur des bordures des boîtes + points extrêmes
- `size >>>` taille des bordures des boîtes
- `outlier.size >>>` taille des points extrêmes

Initialisation du graphique  
Set de données



Type de graphique



Echelles / Légendes /  
Thème

# Type de graphique

## `geom_density`:

- `aes(X, fill)`
  - `alpha >>>` Permet de gérer l'intensité du remplissage des couleurs (utile en cas de superposition)
  - `color >>>` couleur des contours des distributions
  - `size >>>` taille des contours
- + `scale_fill_manual(values=liste des couleurs pour les densités)`

## `geom_histogram` :

- `aes(X)`
- `bins >>>` Sélectionner le nombre d'intervalle à représenter
- `binwidth >>>` Sélectionner une taille d'intervalle pour la représentation

Initialisation du graphique  
Set de données



Type de graphique



Echelles / Légendes /  
Thème

# Type de graphique

`geom_point` : tracer un nuage de points

`geom_line` : tracer la droite  $y = f(x)$

Ces deux fonctions prennent comme argument :

- `aes(X, Y)`
- `color >>>` couleur points / courbe
- `size >>>` taille points / courbe

`geom_abline` : tracer une droite linéaire  $Ax+B$  (peut être superposée à un autre graphique)

- `aes(intercept = B, slope = A)`
- `color >>>` couleur de la droite

Initialisation du graphique  
Set de données



Type de graphique



Echelles / Légendes /  
Thème

# Echelles / Légendes / Thèmes

Echelles { `xlim(min,max) / ylim` : avec retrait de points hors des bornes  
`coord_cartesian(xlim=c(min,max), ylim)` : sans retrait de points

Thème { `theme_bw()` : fond blanc avec grille en gris  
`theme_dark()` : fond sombre  
`theme_linedraw()` : fond blanc avec grille en noir  
`theme_void()` : fond blanc sans grille (retire aussi les légendes)

Légendes { `ggtitle(« titre du graphique »)`  
`xlab(« légende axe x ») / ylab(« légende axe y »)`



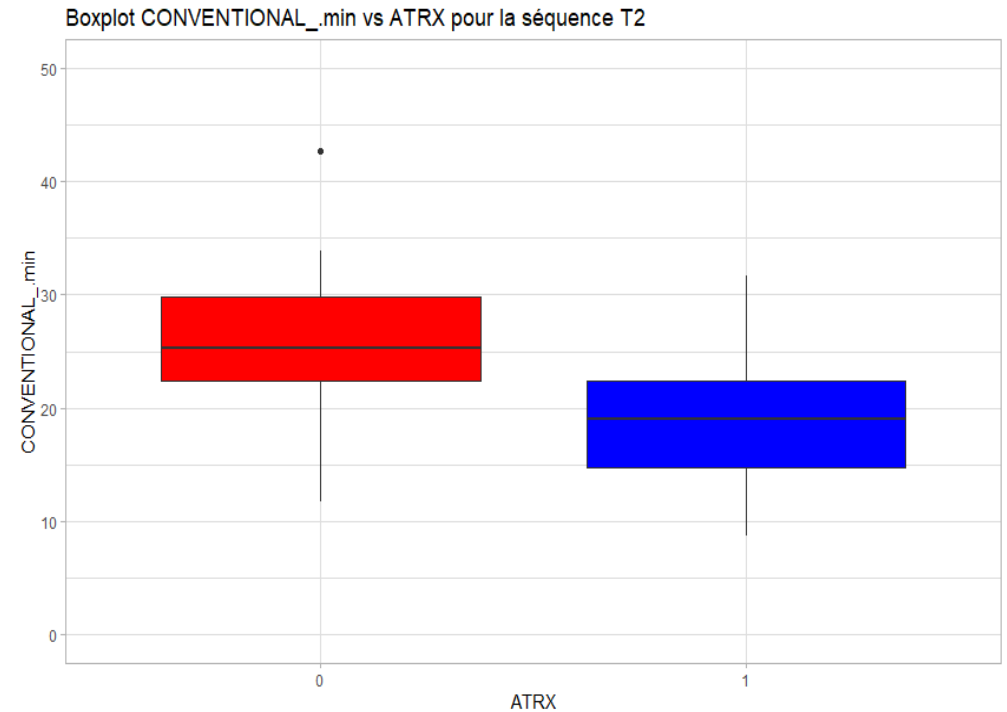


# Exemple boxplot

```
ggplot(DATA)
+ geom_boxplot(aes(x=ATR, y = CONVENTIONAL_.min), fill = c("red", "blue")) #tracer boxplot de
Conventional.min (y) en fonction des patients ATR (x) «0»/«1» qui seront en rouge/bleu (fill)
+ ggtitle("Boxplot CONVENTIONAL_.min vs ATR pour la séquence T2") #titre du graphique
+ ylim(0,50) #échelle en y
+ theme_light() #fond blanc
```

DATA =

| n° patient | CONVENTIONAL_.min | CONVENTIONAL_.min harmonized | ATR |
|------------|-------------------|------------------------------|-----|
| 1,00       | 11,70             | 11,47                        | 0   |
| 2,00       | 42,70             | 40,09                        | 0   |
| 3,00       | 24,56             | 25,41                        | 0   |
| 4,00       | 25,20             | 26,11                        | 1   |
| ...        |                   |                              |     |
| 62,00      | 17,10             | 16,99                        | 1   |

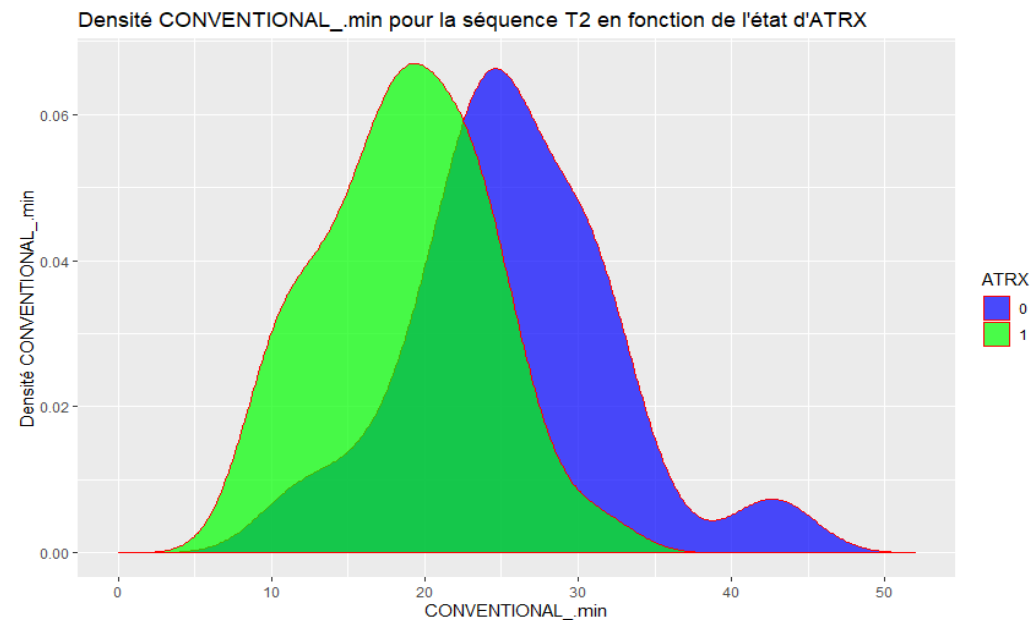


# Exemple densité

```
ggplot(DATA)  
+ geom_density(aes(x=CONVENTIONAL_.min,fill = ATRX),alpha = 0.7,color = c("red"))#tracer  
densité de Conventional.min (x) en fonction des patients ATRX (fill)«0»/«1», contour des  
densités en rouge et alpha permet de gérer la transparence du remplissage  
+ xlim(0,52) #échelle en x  
+ ggtitle("Densité CONVENTIONAL_.min pour la séquence T2 en fonction de l'état d'ATRX")  
+ ylab("Densité CONVENTIONAL_.min") #légende axe y  
+ scale_fill_manual(values=c("blue", "green")) #couleur de remplissage des densités
```

**DATA =**

| n° patient | CONVENTIONAL_.min | CONVENTIONAL_.min hamonized | ATRX |
|------------|-------------------|-----------------------------|------|
| 1,00       | 11,70             | 11,47                       | 0    |
| 2,00       | 42,70             | 40,09                       | 0    |
| 3,00       | 24,56             | 25,41                       | 0    |
| 4,00       | 25,20             | 26,11                       | 1    |
| ...        |                   |                             |      |
| 62,00      | 17,10             | 16,99                       | 1    |

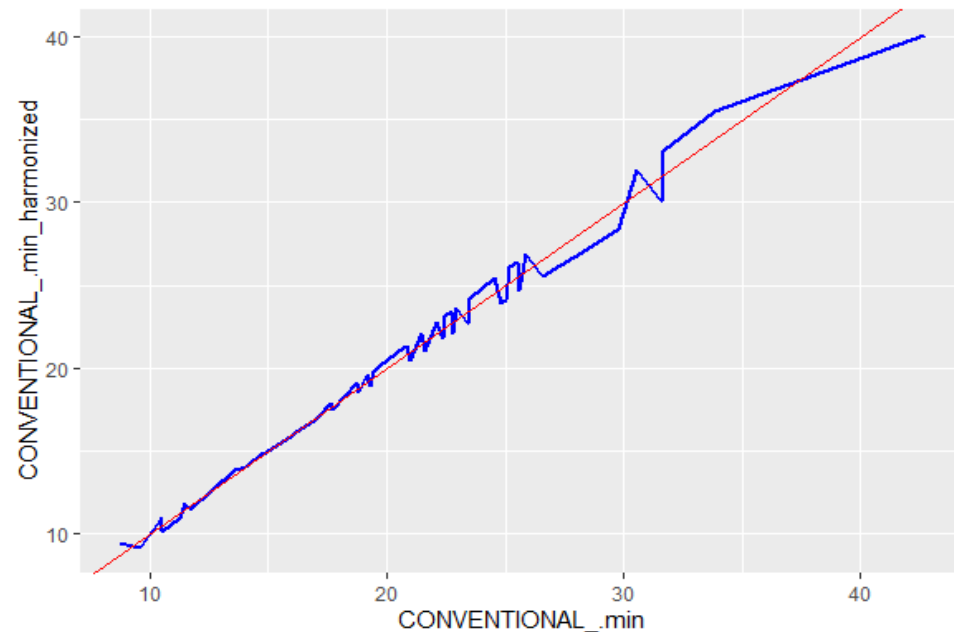


# Exemple courbe

```
ggplot(DATA)
+ geom_line(mapping = aes(x=CONVENTIONAL_.min, y = CONVENTIONAL_.min_harmonized),
color="blue",size = 1) #tracer d'une courbe reliant tous les points correspondant au patient
+ geom_abline(aes(intercept = 0,slope = 1),color="red") #ajout d'une droite linéaire y =
slope*x + intercept >>> ici y = x.
```

DATA =

| n° patient | CONVENTIONAL_.min | CONVENTIONAL_.min_harmonized | ATRX |
|------------|-------------------|------------------------------|------|
| 1.00       | 11,70             | 11,47                        | 0    |
| 2.00       | 42,70             | 40,09                        | 0    |
| 3.00       | 24,56             | 25,41                        | 0    |
| 4.00       | 25,20             | 26,11                        | 1    |
| ...        |                   |                              |      |
| 62.00      | 17,10             | 16,99                        | 1    |

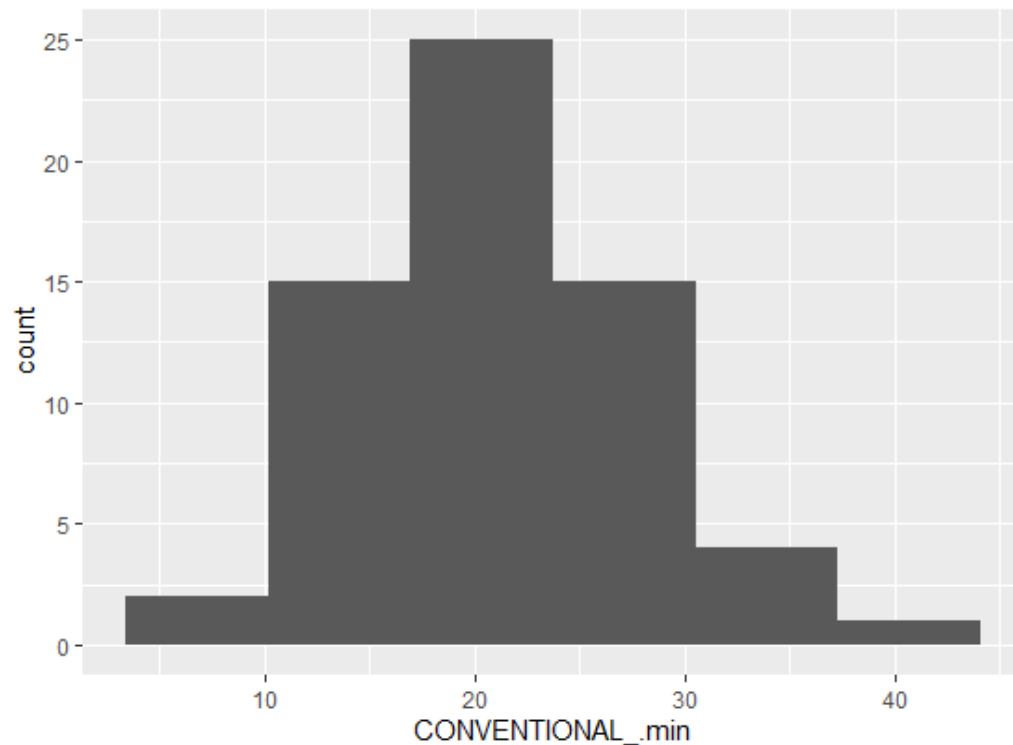


# Exemple histogramme

```
ggplot(DATA)  
+ geom_histogram(mapping = aes(x=CONVENTIONAL_.min),bins=6)#tracer histogramme de  
Conventional.min avec 6 intervalles
```

DATA =

| n° patient | CONVENTIONAL_.min | CONVENTIONAL_.min harmonized | ATRX |
|------------|-------------------|------------------------------|------|
| 1,00       | 11,70             | 11,47                        | 0    |
| 2,00       | 42,70             | 40,09                        | 0    |
| 3,00       | 24,56             | 25,41                        | 0    |
| 4,00       | 25,20             | 26,11                        | 1    |
| ...        |                   |                              |      |
| 62,00      | 17,10             | 16,99                        | 1    |



# Subplot avec librairie gridExtra

- La fonction `grid.arrange` de la librairie `gridExtra` permet de tracer plusieurs graphiques en même temps.

```
grid.arrange(graphique 1, graphique 2, graphique 3, graphique 4, nrow = 2, ncol = 2)
```

>>> permet d'afficher 4 graphiques avec 2 graphiques par ligne.

- Les graphiques s'affichent selon l'ordre spécifié dans la fonction :

graphique 1 = 1<sup>er</sup> ligne/1<sup>er</sup> colonne puis la fonction remplit la 1<sup>er</sup> ligne de gauche à droite avant de passer à la ligne suivante et recommence ce processus.

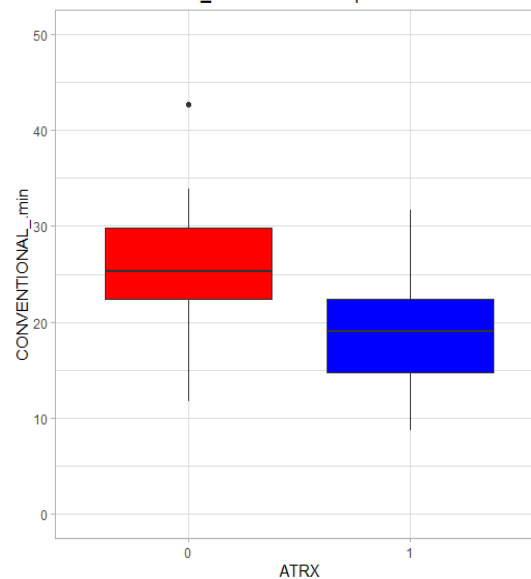
# Exemple subplot

```
grid.arrange(Boxplot_ggplot, Boxplot_ggplot2, nrow=1, ncol=2)  
#tracer deux subplots sur 1 ligne et 2 colonnes
```

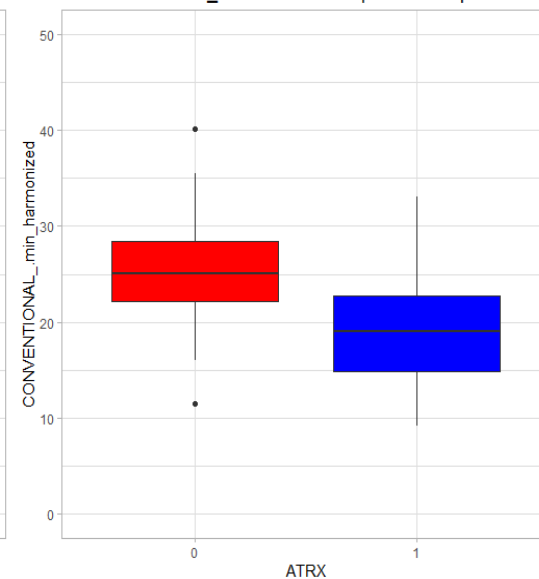
DATA =

| n° patient | CONVENTIONAL .min | CONVENTIONAL .min harmonized | ATRX |
|------------|-------------------|------------------------------|------|
| 1,00       | 11,70             | 11,47                        | 0    |
| 2,00       | 42,70             | 40,09                        | 0    |
| 3,00       | 24,56             | 25,41                        | 0    |
| 4,00       | 25,20             | 26,11                        | 1    |
| • • •      |                   |                              |      |
| 62,00      | 17,10             | 16,99                        | 1    |

CONVENTIONAL\_.min vs ATRX séquence T2 avH



CONVENTIONAL\_.min vs ATRX séquence T2 apH



# Graphique ACP

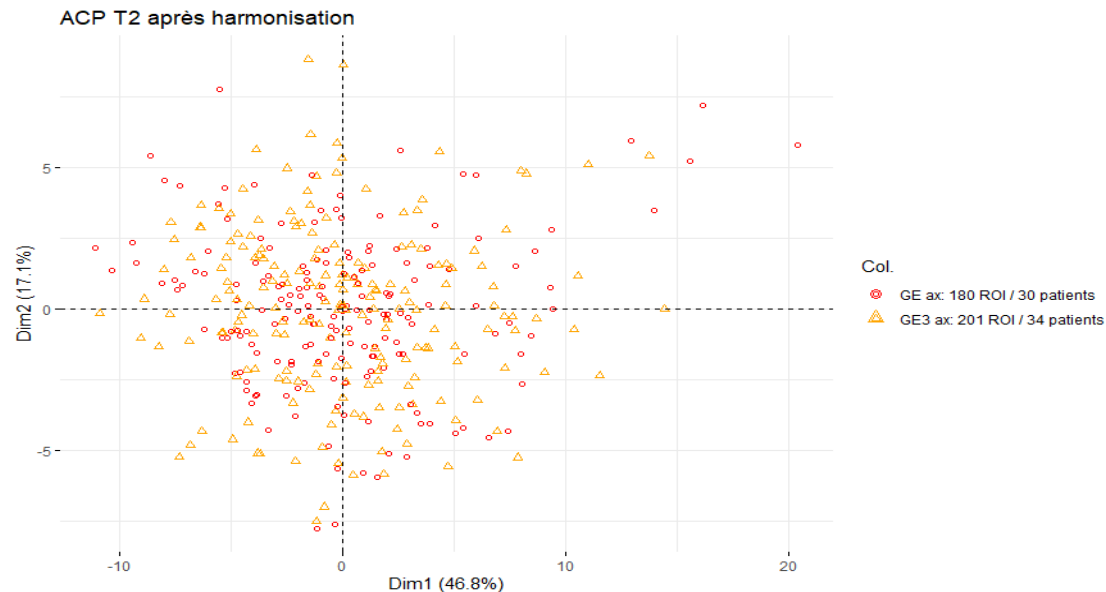
- Librairie FactoMineR pour faire le calcul des ACP
- Librairie factoextra pour les représentations graphiques :
  - `fviz_pca_ind >>>` Graphique selon les individus
  - `fviz_pca_var >>>` Graphique selon les variables
  - `fviz_pca_biplot >>>` Graphique selon les individus et les variables

Variable entrée :

- `X` : données ACP
- `axes` : choisir les dimensions à représenter
- `geom` : « point » / « text » / `c`(« point », « text »)
- `AddEllipse >>>` TRUE pour avoir des ellipses regroupant les points proches
- `col.ind >>>` définir les levels
- `labels >>>` légendes des levels
- `palette >>>` définir des couleurs en fonction des levels
- `title >>>` titre du graphique

# Exemple graphique ACP

```
res_pca = PCA(T2_harmonized, graph = FALSE) #réalisation de l'ACP
fviz_pca_ind(res_pca, geom.ind = "point" #individu représenté par des points,
             col.ind = factor(ListingMachineT2$T2M, levels = c("GE ax", "GE3 ax") #individu coloré
                               selon le type de machine, labels = c("GE ax: 180 ROI / 30 patients", "GE3 ax: 201 ROI
                               / 34 patients")) #définition de la légende des couleurs,
             palette = c("red", "orange") #couleur utilisée,
             title = "ACP T2 après harmonisation") #titre
+scale_shape_manual(values=c(1,2)) #modification des symboles utilisés
```





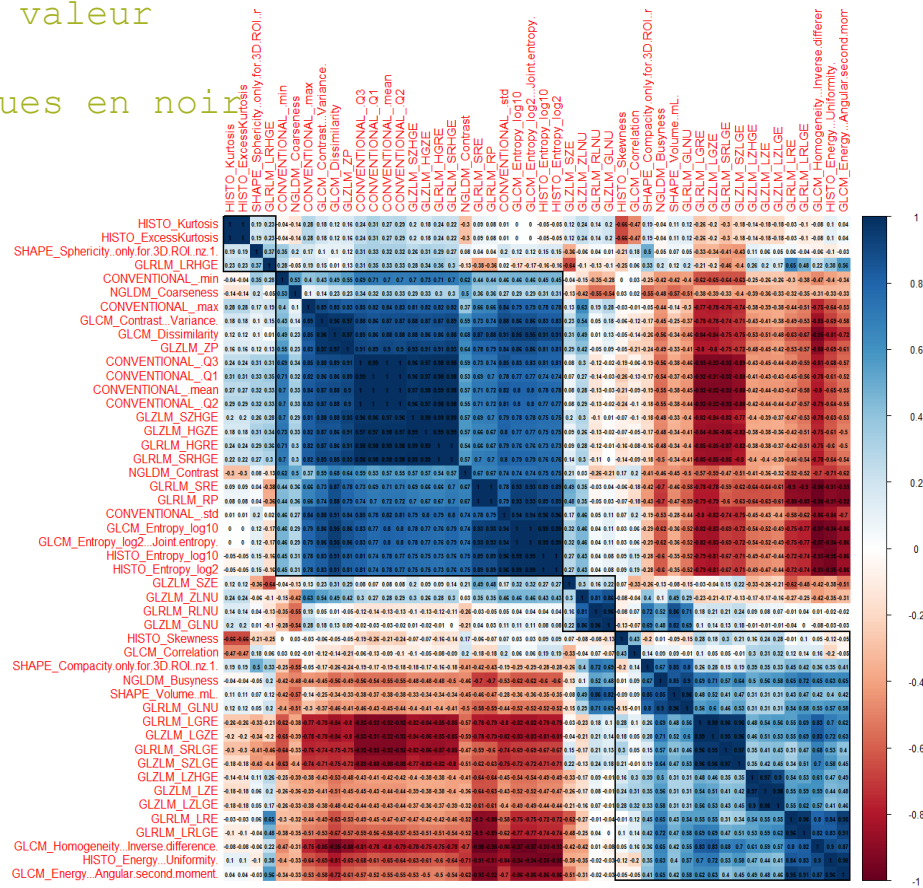
# Graphique Heatmap (corrplot)

- **Librairie corrplot** → `corrplot` :
  - `X` : matrice de corrélation (ou matrice → mettre `is.corr = FALSE` dans la fonction)
  - `method` : représentation des valeurs : "circle", "square", "ellipse", "number", "shade", "color", "pie".
  - `type` : «upper/«lower»/«full» >>> représenter le triangle supérieur de la matrice
  - `title` : « titre du graphique »
  - `cl.lim` : gérer l'étendue de l'échelle de couleurs
  - `cl.cex` : taille pour l'échelle de couleur
  - `tl.cex` : taille des légendes
  - `addCoef.col` : « couleur » place les valeurs de la matrice sur la heatmap selon la couleur définie
  - `number.cex` : taille des nombres dans le corrélogramme
  - `order` : «hclust»/«AOE»/«FPC»/«Original»/«alphabet»
  - `addrect` : trace des rectangles (nécessite `order = hclust`)

# Exemple corrpplot (avec matrice de corrélation)

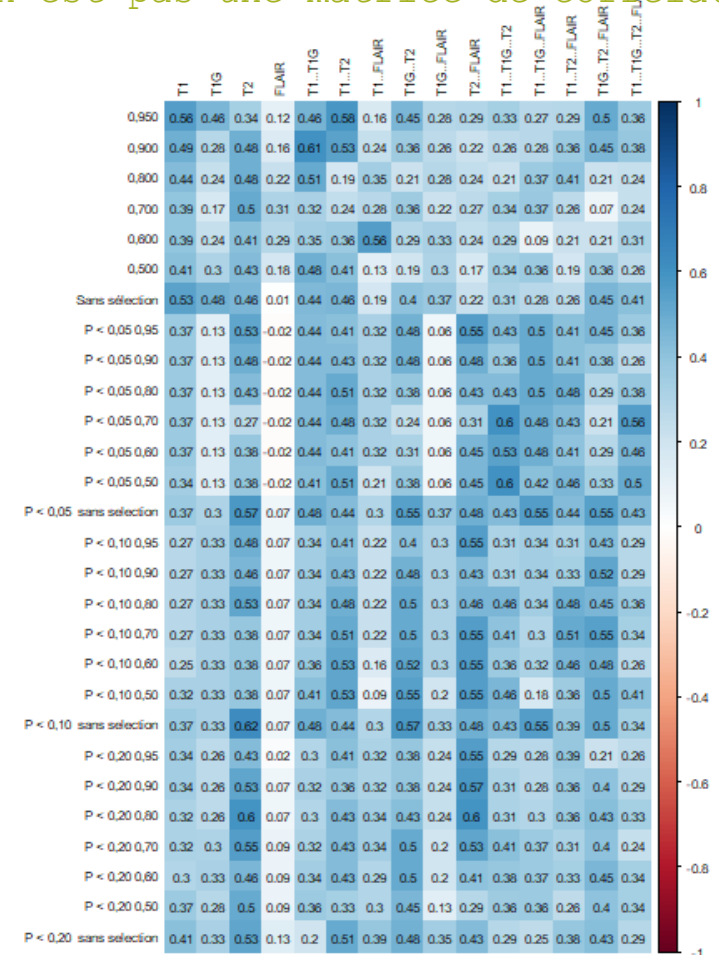
```
Mat_cor = cor(T1,T1) #calcul de la matrice de correlation  
corrplot(Mat_cor,method = "color",  
#colore les cases du corrélogramme en fonction de la valeur  
numérique associée
```

```
addCoef.col = "black", #ajout des valeurs numériques en noir  
number.cex = 0.5, #réglage taille des nombres  
order = "hclust", #méthode de clusterisation  
address = 4) #tracer 4 rectangles selon hclust
```



# Exemple corrplot (avec une matrice normal)

```
corrplot(Youden, is.corr = FALSE, #FALSE car la matrice n'est pas une matrice de corrélation
method = "color",
addCoef.col = "black",
number.cex = 0.5)
```



# Graphique Heatmap (aheatmap)

- Librairie NMF → aheatmap :
  - `X` : matrice/dataframe
  - `main` : « titre »
  - `scale` : «row» : z-scores selon les lignes  
«column» : z-scores selon les colonnes  
«none» : données brutes
  - `border` : TRUE pour tracer les bordures et la grille entre les lignes et colonnes
  - `fontsize` : taille de la légende
  - `annCol` : liste des arguments à représenter >>> ajout de barres de couleur au dessus de la heatmap
  - `annRow` : Idem AnnCol mais barre de couleur à gauche de la heatmap
  - `annColors` : liste définissant les légendes et couleurs si nécessaires

# Exemple aheatmap

```
ann = data.frame(  
  Machine = ifelse(IRM==1, "GE3", "GE"),  
  ATRX = ifelse(ATRX==0, "0", "1")  
)  
#création des annotations en fonction du type de  
machine et de l'état ATRX  
  
color_IRM = c("blue", "red") #def des couleurs  
names(color_IRM) = c("GE3", "GE") #def des légendes  
color_ATRX = c("purple", "yellow")  
names(color_ATRX) = c("0", "1")  
  
ann_color = list(color_IRM, color_ATRX)  
  
aheatmap(t(FLAIR), #matrice FLAIR transposée  
  main = "Heatmap FLAIR", #titre  
  fontsize = 8, #taille de police  
  border = FALSE, #pas de grille  
  annCol = ann, #ajout des barres de couleur  
  annColors = ann_color, #definition des  
barres de couleur  
  Rowv = FALSE, #classement selon les lignes  
  Colv = NA) #pas de classement
```

