

## Chapitre 5

# Analyse Factorielle des Correspondances Multiples (AFCM ou ACM)

Nos allons procéder par traiter un exemple élémentaire et non réaliste, puis nous nous intéresserons à l'exemple traité en cours et enfin nous étudierons une situation pratique.

1. Lancer ensuite R et modifier le répertoire de travail en allant dans **Fichier -> Changer le Répertoire Courant** et en choisissant le répertoire **Bureau/TP\_M1MIASHS\_AD** qui a été créé.
2. Ouvrir une fenêtre d'éditeur **Fichier -> Nouveau Script**.
3. Sauver le fichier dans le répertoire courant sous le nom **TP4.R** : **Fichier -> Sauver sous**
4. Penser à sauvegarder régulièrement le contenu du fichier TP4.R en appuyant sur les touches **Ctrl et S**.

### 5.1 Mise en oeuvre

L'Analyse des Correspondances Multiples (ACM ou MCA pour multiple correspondence analysis) est une extension de l'analyse factorielle des correspondances pour résumer et visualiser un tableau de données contenant plus de deux variables catégorielles. On peut aussi la considérer comme une généralisation de l'analyse en composantes principales lorsque les variables à analyser sont catégorielles plutôt que quantitatives (Abdi and Williams 2010). L'ACM est généralement utilisée pour analyser des données d'enquête ou de sondage. L'objectif est d'identifier :

- Un groupe de personnes ayant un profil similaire dans leurs réponses aux questions
- Les associations entre les catégories des variables

Nous allons décrire comment calculer et visualiser l'analyse des correspondances multiples avec le logiciel R en utilisant les packages **FactoMineR** (pour l'analyse) et **factoextra** (pour la visualisation des données). De plus, nous montrerons comment révéler les variables les plus importantes qui contribuent le plus à expliquer les variations dans le jeu de données. Nous continuons en expliquant comment prédire les résultats pour les individus et les variables supplémentaires. Enfin, nous allons démontrer comment filtrer les résultats de l'ACM afin de ne conserver que les variables les plus contributives.

### 5.1.1 Calcul

1. *Packages R* : Plusieurs fonctions, de différents packages, sont disponibles dans le logiciel R pour le calcul de l'ACM :
 

```
MCA() [package FactoMineR]
dudi.mca() [package ade4]
epMCA() [package ExPosition]
```

Peu importe la fonction que vous décidez d'utiliser, vous pouvez facilement extraire et visualiser les résultats de l'ACM en utilisant les fonctions R fournies dans le package **factoextra**.

Ici, nous utiliserons les deux packages **FactoMineR** (pour l'analyse) et **factoextra** (pour la visualisation basée sur **ggplot2**).

Installez les deux packages comme suit : `install.packages(c("FactoMineR", "factoextra"))`  
Chargez-les dans R, en tapant ceci :

```
library("FactoMineR")
library("factoextra")
```

2. *Format des données* : Nous utiliserons le jeux de données **poison** disponible dans le package **FactoMineR** :

```
data(poison)
head(poison[, 1 :7], 3)
# Age Time Sick Sex Nausea Vomiting Abdominals
# 1 9 22 Sick_y F Nausea_y Vomit_n Abdo_y
# 2 5 0 Sick_n F Nausea_n Vomit_n Abdo_n
# 3 6 16 Sick_y F Nausea_n Vomit_y Abdo_y
```

Ces données proviennent d'une enquête menée auprès d'enfants de l'école primaire qui ont subi des intoxications alimentaires. Ils ont été interrogés sur leurs symptômes et sur ce qu'ils ont mangé.

Les données contiennent 55 lignes (individus) et 15 colonnes (variables). Nous n'utiliserons que certain des individus (enfants) et variables pour effectuer l'ACM. Les coordonnées des individus et des variables restantes seront prédites.

Nos données contiennent donc des :

- Individus actifs (lignes 1 : 55) : individus qui sont utilisés dans l'ACM.
- Variables actives (colonnes 5 : 15) : variables utilisées dans l'ACM.
- Variables supplémentaires : elles ne participent pas à l'ACM. Les coordonnées de ces variables seront prédites.
  - Variables quantitatives supplémentaires (`quanti.sup`) : Colonnes 1 et 2 correspondant aux colonnes `age` et `time`, respectivement.
  - Variables qualitatives supplémentaires (`quali.sup`) : Colonnes 3 et 4 correspondant aux colonnes `Sick` et `Sex`, respectivement. Ces variables seront utilisées pour colorer les individus par groupes.

Nous commençons par extraire les individus actifs et les variables actives pour l'ACM :

```
poison.active <- poison[1 :55, 5 :15]
head(poison.active[, 1 :6], 3)
# Nausea   Vomiting   Abdominals   Fever   Diarrhae   Potato
# 1 Nausea_y   Vomit_n   Abdo_y   Fever_y   Diarrhea_y   Potato_y
# 2 Nausea_n   Vomit_n   Abdo_n   Fever_n   Diarrhea_n   Potato_y
# 3 Nausea_n   Vomit_y   Abdo_y   Fever_y   Diarrhea_y   Potato_y
```

3. *Résumé des données* : La fonction `summary()` peut être utilisée pour calculer la fréquence des catégories des variables. Comme le tableau de données contient un grand nombre de variables, nous afficherons uniquement les résultats pour les 4 premières variables.

Résumés statistiques :

```
# Résumé des 4 premières variables
summary(poison.active)[, 1 :4]
# Nausea       Vomiting       Abdominals       Fever
# Nausea_n :43   Vomit_n :33   Abdo_n :18   Fever_n :20
# Nausea_y :12   Vomit_y :22   Abdo_y :37   Fever_y :35
```

La fonction `summary()` renvoie la taille des catégories des variables.

Il est également possible de visualiser la fréquence des catégories des variables. Le code R ci-dessous, montre les 4 premières colonnes :

```
for (i in 1 :4) {
  plot(poison.active[, i], main = colnames(poison.active)[i],
       ylab = "Count", col="steelblue", las = 2 )
```

Remarquer que ces quelques graphiques peuvent être utilisés pour identifier les catégories à très faible fréquence. Ce type de variables peut fausser l'analyse et doit être

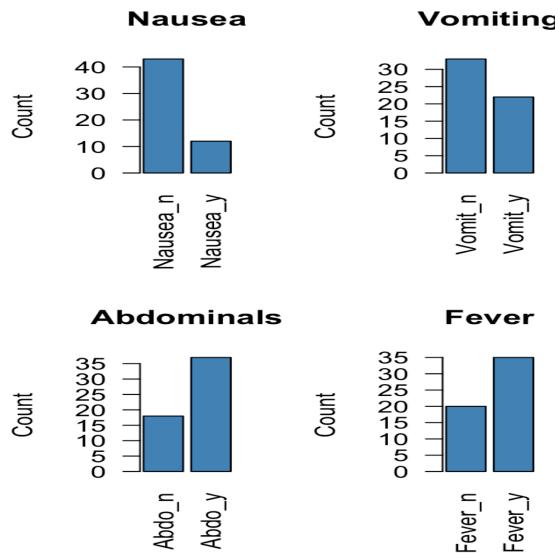


FIGURE 5.1 – Fréquence des catégories des variables

supprimé.

4. *Code R* : La fonction **MCA()** [FactoMineR] peut etre utilisée. Un format simplifié est :

**MCA(X, ncp = 5, graphique = TRUE)**

- **X** : tableau de données avec *n* lignes (individus) et *p* colonnes (variables catégorielles)
- **ncp** : nombre de dimensions conservées dans les résultats finaux.
- **graph** : valeur logique. Si TRUE le graphique est affiché.

Dans le code R ci-dessous, l'ACM est effectuée uniquement sur les individus/variables actifs :

```
res.mca <- MCA(poison.active, graph = FALSE)
```

Le résultat de l'ACM est une liste comprenant :

```
print(res.mca)
# **Results of the Multiple Correspondence Analysis (MCA)**
# The analysis was performed on 55 individuals, described by 11 variables
# *The results are available in the following objects :
#
```

```

#   name           description
#  1 "$eig"         "eigenvalues"
#  2 "$var"          "results for the variables"
#  3 "$var$coord"    "coord. of the categories"
#  4 "$var$cos2"     "cos2 for the categories"
#  5 "$var$contrib"  "contributions of the categories"
#  6 "$var$v.test"   "v-test for the categories"
#  7 "$ind"          "results for the individuals"
#  8 "$ind$coord"    "coord. for the individuals"
#  9 "$ind$cos2"     "cos2 for the individuals"
# 10 "$ind$contrib"  "contributions of the individuals"
# 11 "$call"          "intermediate results"
# 12 "$call$marge.col" "weights of columns"
# 13 "$call$marge.li"  "weights of rows"

```

L'objet créé avec la fonction `MCA()` contient de nombreuses informations trouvées dans de nombreuses listes et matrices différentes. Ces valeurs sont décrites dans la section suivante.

### 5.1.2 Visualisation et interprétation

Nous utiliserons le package R `factoextra` pour aider à l'interprétation et à la visualisation de l'analyse des correspondances multiples. Peu importe la fonction que vous décidez d'utiliser [`FactoMiner : MCA()`], [`ade4 : dudi.mca()`], vous pouvez facilement extraire et visualiser les résultats de l'ACM en utilisant les fonctions R fournies dans le package `factoextra`.

Ces fonctions de `factoextra` incluent :

- `get_eigenvalue(res.mca)` : Extraction des valeurs propres / variances des composantes principales
- `fviz_eig(res.mca)` : Visualisation des valeurs propres
- `get_mca_ind(res.mca)`, `get_mca_var(res.mca)` : Extraction des résultats pour les individus et les variables, respectivement.
- `fviz_mca_ind(res.mca)`, `fviz_mca_var(res.mca)` : visualisation des résultats des individus et des variables, respectivement.
- `fviz_mca_biplot(res.mca)` : Crédit d'un biplot des individus et des variables.

Dans les sections suivantes, nous allons illustrer chacune de ces fonctions. Notez que les résultats de l'ACM sont interprétés comme les résultats de l'AFC. Par conséquent, il est fortement recommandé de lire l'interprétation de l'AFC.

1. *Valeurs propres/Variances* : La proportion des variances retenues par les différentes dimensions (axes) peut être extraite à l'aide de la fonction `get_eigenvalue()` [factoextra package] comme suit :

```
library("factoextra")
eig.val <- get_eigenvalue(res.mca)
# head(eig.val)
```

Pour visualiser les pourcentages de variances expliquées par chaque dimension de l'ACM, utilisez la fonction `fviz_eig()` ou `fviz_screenplot()` dans [package factoextra] : `fviz_screenplot (res.mca, addlabels = TRUE, ylim = c (0, 45))`

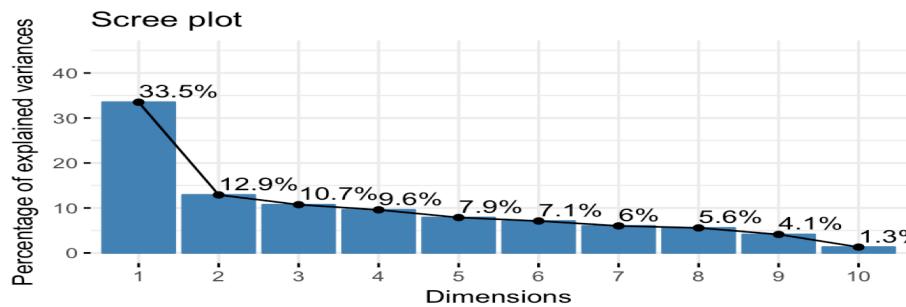


FIGURE 5.2 – Pourcentage de variance expliquée

2. *Biplot* : La fonction `fviz_mca_biplot()` [factoextra] permet de visualiser le biplot des individus et des variables :

```
fviz_mca_biplot (res.mca, repel = TRUE, ggtheme = theme_minimal())
```

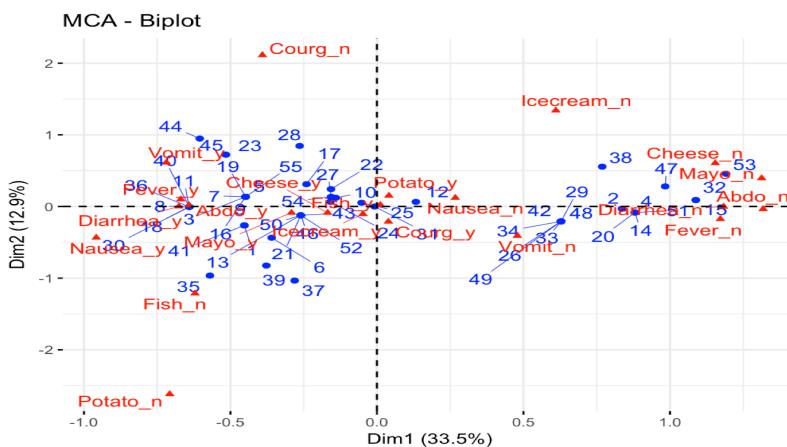


FIGURE 5.3 – Biplot des individus et des variables

Il faut remarquer que dans le graphique obtenu, par cette commande, les lignes (individus) sont représentées par des points bleus et des colonnes (variables) par des triangles rouges.

La distance entre les individus donne une mesure de leur similitude (ou dissemblance). Les individus avec un profil similaire sont proches sur le graphique. Il en va de même pour les variables.

### 3. Graphique des variables :

- (a) *Résultats* : La fonction `get_mca_var()` [factoextra] sert à extraire les résultats pour les catégories des variables. Cette fonction renvoie une liste contenant les coordonnées, les `cos2` et les contributions des catégories :

```
var <- get_mca_var(res.mca)
var
```

```
# Multiple Correspondence Analysis Results for variables #
#=====
# Name
# 1 "$coord"
# 2 "$cos2"
# 3 "$contrib" "contributions of categories"
```

Les composants de `get_mca_var()` peuvent être utilisés dans le graphique des variables comme suit :

- `var$coord` : coordonnées des variables pour créer un nuage de points
- `var$cos2` : qualité de représentation des variables.
- `var$contrib` : contributions (en pourcentage) des variables à la définition des dimensions.

Noter qu'il est possible de visualiser les catégories des variables et de les colorer en fonction de soit leurs qualités de représentation (`cosinus carré`, `cos2`), soit de leurs contributions à la définition des dimensions (`contrib`).

Les différents composants peuvent être consultés comme suit :

```
# Coordonnées
head(var$coord)
# Cos2 : qualité de représentation
head(var$cos2)
# Contributions aux axes
head(var$contrib)
```

Dans cette section, nous décrirons comment visualiser uniquement les catégories des variables. Ensuite, nous mettrons en évidence les catégories en fonction soit de leurs qualités de représentation, soit de leurs contributions aux dimensions.

- (b) *Corrélation entre les variables et les axes principaux* : Pour visualiser la corrélation entre les variables et les axes principaux de l'ACM, tapez ceci :

```
fviz_mca_var (res.mca, choice = "mca.cor", repel = TRUE,
ggtheme = theme_minimal ())
```

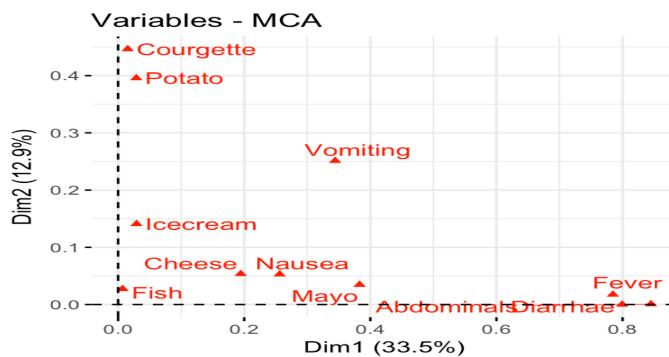


FIGURE 5.4 – Corrélation entre les variables et les axes principaux

- Le graphique ci-dessus permet d'identifier les variables les plus corrélées avec chaque axe. Les corrélations au carré entre les variables et les axes sont utilisées comme coordonnées.
- On constate que les variables Diarrhae, Abdominals et Fever sont les plus corrélées avec la dimension 1. De même, les variables Courgette et Potato sont les plus corrélées avec la dimension 2.

- (c) *Coordonnées des catégories des variables* : Le code R ci-dessous affiche les coordonnées de chacune des catégories des variables dans chaque dimension (1, 2 et 3) :

```
head(round(var$coord, 2), 4)
```

Utiliser la fonction `fviz_mca_var()` [factoextra] pour visualiser uniquement les catégories des variables :

```
fviz_mca_var (res.mca, repel = TRUE, ggtheme = theme_minimal ())
```

Il est possible de modifier la couleur et la forme des points à l'aide des arguments `col.var` et `shape.var` comme suit :

```
fviz_mca_var(res.mca, col.var="black", shape.var = 15, repel = TRUE)
```

Le graphique obtenu montre les relations entre les catégories des variables. Il peut être interprété comme suit :

- Les catégories avec un profil similaire sont regroupées.
- Les catégories corrélées négativement sont positionnées sur les cotés opposés de l'origine du graphique (quadrants opposés).

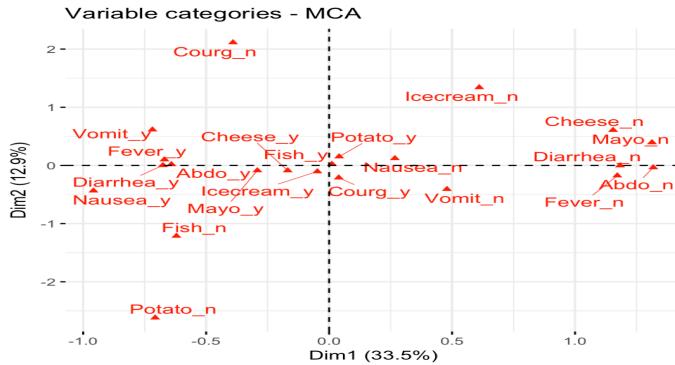


FIGURE 5.5 – Fréquence des catégories des variables

- La distance entre les catégories et l'origine mesure la qualité des catégories. Les points qui sont loin de l'origine sont bien représentés par l'ACM.

- (d) *Qualité de représentation des catégories des variables* : Les deux dimensions 1 et 2 capturent 46% de l'inertie totale (variation) contenue dans les données. Tous les points ne sont pas aussi bien représentés par les deux dimensions.

La qualité de représentation, appelée **cosinus carré** (**cos2**), mesure le degré d'association entre les catégories des variables et les dimensions. Le **cos2** peut être extrait comme suit :

```
head(var$cos2, 4)
```

Si une catégorie d'une variable donnée est bien représentée par deux dimensions, la somme des **cos2** est proche de 1. Pour certains éléments, plus de 2 dimensions sont nécessaires pour représenter parfaitement les données. Il est possible de colorer les variables en fonction de la valeur de leur **cos2** à l'aide de l'argument **col.var = "cos2"**. Cela produit un gradient de couleurs. Dans ce cas, l'argument **gradient.cols** peut être utilisé pour spécifier une palette de couleur personnalisée. Par exemple, **gradient.cols = c("white", "blue", "red")** signifie que :

- les variables à faible valeur de **cos2** seront colorées en **white** (blanc)
- les variables avec des valeurs moyennes de **cos2** seront colorées en **blue** (bleu)
- les variables avec des valeurs élevées de **cos2** seront colorées en **red** (rouge)

```
# Colorer en fonction du cos2
fviz_mca_var(res.mca, col.var = "cos2",
gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"), repel = TRUE,
ggtheme = theme_minimal())
```

Noter qu'il est également possible de modifier la transparence des variables en fonction de leurs **cos2** à l'aide de l'option **alpha.var = "cos2"**. Par exemple, taper ceci :

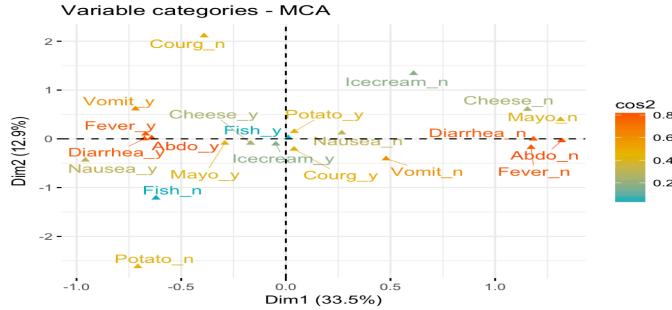


FIGURE 5.6 – Fréquence des catégories des variables

```
# Changer la transparence en fonction du cos2
fviz_mca_var(res.mca, alpha.var = "cos2", repel = TRUE,
ggtheme = theme_minimal())
```

On peut visualiser le **cos2** des catégories sur toutes les dimensions en utilisant le package **corrplot** :

```
library("corrplot")
corrplot(var$cos2, is.corr=FALSE)
```

Il est également possible de créer un barplot du **cos2** des variables avec la fonction **fviz\_cos2()** [factoextra] :

```
# Cos2 des variable sur Dim.1 et Dim.2
fviz_cos2(res.mca, choice = "var", axes = 1 :2)
```

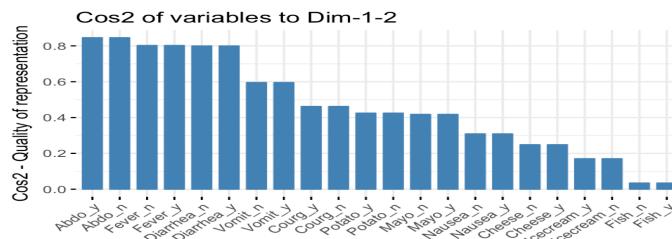


FIGURE 5.7 – cos2 de toutes les catégories sur toutes les dimensions

- (e) *Contribution des variables aux dimensions* : La contribution des variables (en %) à la définition des dimensions peut être extraite comme suit :

```
head(round(var$contrib,2), 4)
```

Les variables avec les plus grandes valeurs, contribuent le mieux à la définition

des dimensions. Les catégories qui contribuent le plus à Dim.1 et Dim.2 sont les plus importantes pour expliquer la variabilité dans le jeu de données.

La fonction `fviz_contrib()` [factoextra] peut être utilisée pour faire un barplot de la contribution des catégories des variables. Le code R ci-dessous montre le top 15 des catégories contribuant aux dimensions :

```
# Contributions des variables à la dimension 1
fviz_contrib(res.mca, choice = "var", axes = 1, top = 15)
# Contributions des variables à la dimension 2
fviz_contrib(res.mca, choice = "var", axes = 2, top = 15)
```

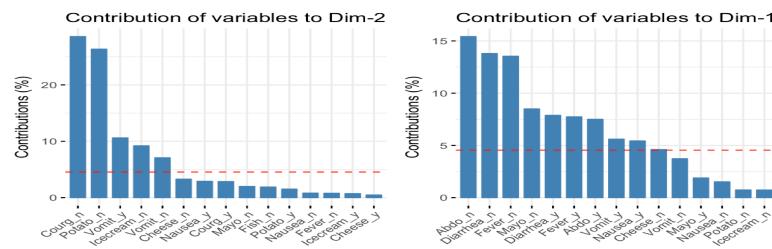


FIGURE 5.8 – Barplot des contributions

Les contributions totales aux dimensions 1 et 2 sont obtenues comme suit :

```
# Contribution totale aux dimensions 1 et 2
fviz_contrib(res.mca, choice = "var", axes = 1 :2, top = 15)
```

La ligne en pointillé rouge, sur le graphique ci-dessus, indique la valeur moyenne attendue sous l'hypothèse nulle.

On peut voir que :

- les catégories `Abdo_n`, `Diarrhea_n`, `Fever_n` et `Mayo_n` sont les plus importantes dans la définition de la première dimension.
- Les catégories `Courg_n`, `Potato_n`, `Vomit_y` et `Icecream_n` contribuent le plus à la dimension 2

Les catégories les plus importantes peuvent être mises en évidence sur le graphique comme suit :

```
fviz_mca_var(res.mca, col.var = "contrib",
gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"), repel = TRUE,
ggtheme = theme_minimal() )
```

Le graphique ci-dessus donne une idée du pôle des dimensions auquel les catégories contribuent réellement.

Il est évident que les catégories `Abdo_n`, `Diarrhea_n`, `Fever_n` et `Mayo_n` ont une contribution importante au pôle positif de la première dimension, tandis que les catégories `Fever_y` et `Diarrhea_y` ont une contribution majeure au pôle négatif de la première dimension ; etc, ....

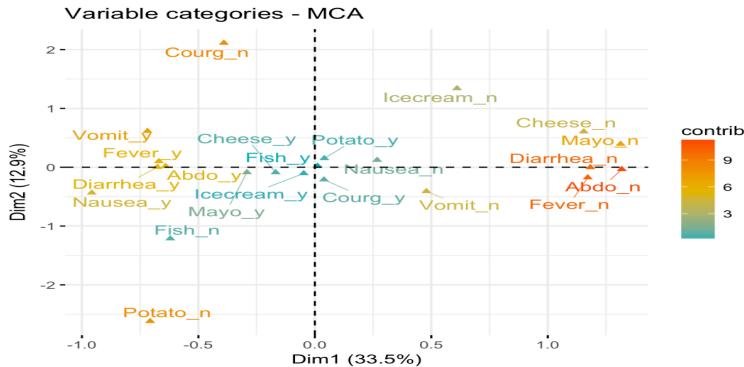


FIGURE 5.9 – Mise en évidence des catégories les plus importantes

Noter qu'il est également possible de contrôler la transparence des catégories en fonction de leurs contributions en utilisant l'option `alpha.var = "contrib"`. Par exemple, tapez ceci :

```
# Changez la transparence en fonction de la contrib
fviz_mca_var (res.mca, alpha.var = "contrib", repel = TRUE,
ggtheme = theme_minimal ())
```

#### 4. Graphique des individus :

- (a) *Résultats* : La fonction `get_mca_ind()` [factoextra] sert à extraire les résultats pour les individus. Cette fonction renvoie une liste contenant les coordonnées, la `cos2` et les contributions des individus :

```
ind <- get_mca_ind (res.mca)
ind
# Multiple Correspondence Analysis Results for individuals
# =====#
# Name
# 1 "$coord"
# 2 "$cos2"
# 3 "$contrib" "contributions of the individuals"
```

Pour accéder aux différents composants, utilisez ceci :

```
# Coordonnées
head(ind$coord)
# Qualité de représentation
head(ind$cos2)
# Contributions
head(ind$contrib)
```

- (b) *Graphique : qualité et contribution* : La fonction `fviz_mca_ind()` [factoextra]

sert à visualiser uniquement des individus. Comme les variables, il est également possible de colorer les individus en fonction de leurs `cos2` :

```
fviz_mca_ind(res.mca, col.ind = "cos2", "#FC4E07"),
gradient.cols = c("#00AFBB", "#E7B800", repel = TRUE,
ggtheme = theme_minimal())
```

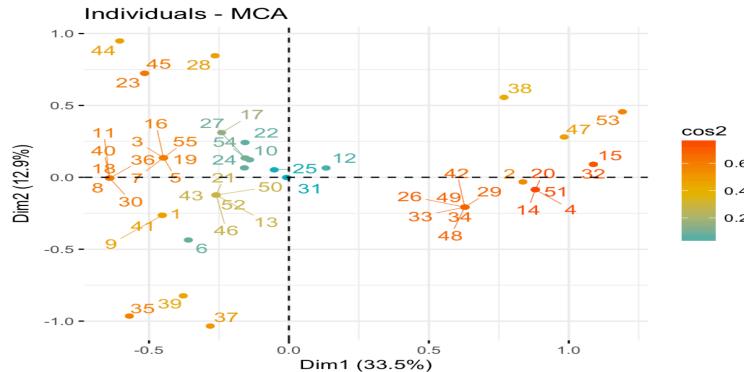


FIGURE 5.10 – Visualisation des individus

Le code R ci-dessous crée un barplot du `cos2` et de la contribution des individus :

```
# Cos2 des individus
fviz_cos2(res.mca, choice = "ind", axes = 1 :2, top = 20)
# Contribution des individus aux dimensions
fviz_contrib(res.mca, choice = "ind", axes = 1 :2, top = 20)
```

5. *Colorer les individus par groupes* : Noter qu'il est possible de colorer les individus en utilisant l'une des variables qualitatives dans le tableau de données initial (`poison`). Le code R ci-dessous colore les individus par groupes en utilisant la variable `Vomiting`. L'argument `habillage` sert à spécifier la variable à utiliser pour colorer les individus par groupes. Une ellipse de concentration peut également être ajoutée autour de chaque groupe en utilisant l'argument `addEllipses = TRUE`. Si vous voulez une ellipse de confiance autour du point moyen (centre de gravité) des groupes, utilisez `ellipse.type = "confidence"`. L'argument `palette` permet de modifier les couleurs du groupe.

```
fviz_mca_ind (res.mca, label = "none", # masquer le texte des individus
"confidence",
habillage = "Vomiting", # colorer par groupes
palette = c ("#00AFBB", "#E7B800"),
addEllipses = TRUE, ellipse.type = ggtheme = theme_minimal ())
```

Noter que, pour spécifier l'argument `habillage`, il est également possible d'utiliser

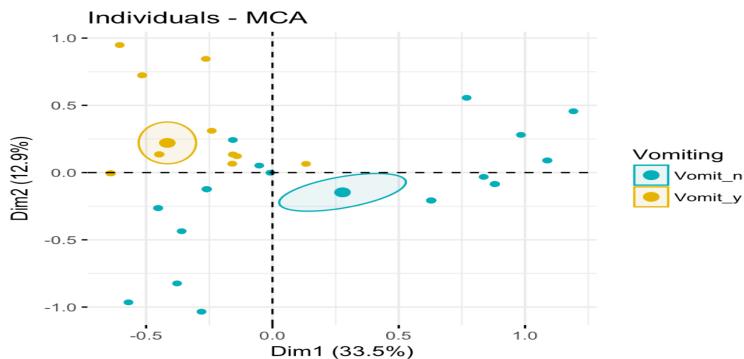


FIGURE 5.11 – coloration des individus par groupes en utilisant la variable Vomiting

l'index de la colonne comme suit (`habillage = 2`). Vous pouvez aussi fournir une variable de regroupement externe comme suit : `habillage = poison$Vomiting`. Par exemple :

```
# habillage = indice de la colonne à utiliser comme variable de regroupement
fviz_mca_ind(res.mca, habillage = 2, addEllipses = TRUE)
# habillage = variable de regroupement externe
fviz_mca_ind(res.mca, habillage = poison$Vomiting, addEllipses = TRUE)
```

Si vous souhaitez colorer les individus à l'aide de plusieurs variables catégorielles en même temps, utilisez la fonction `fviz_ellipses()` [factoextra] comme suit :

```
fviz_ellipses(res.mca, c("Vomiting", "Fever"), geom = "point")
```

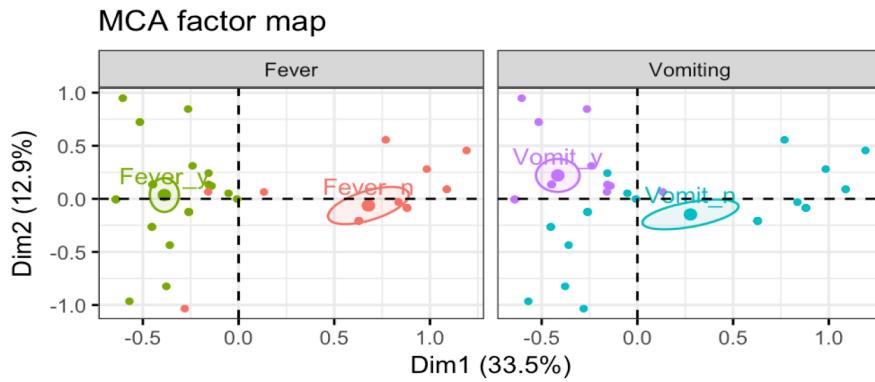


FIGURE 5.12 – coloration des individus à laide de plusieurs variables catégorielles en même temps

Alternativement, vous pouvez spécifier les indices des variables catégorielles :

```
fviz_ellipses (res.mca, 1 :4, geom = "point")
```

6. *Description des dimensions* : La fonction `dimdesc()` [FactoMineR] peut etre utilisée pour identifier les variables les plus corrélées avec une dimension donnée :

```

res.desc <- dimdesc (res.mca, axes = c(1,2))
# Description de la dimension 1
res.desc[[1]]
# Description de la dimension 2
res.desc[[2]]

```

### 5.1.3 Eléments supplémentaires

1. *Définition et types* : Comme décrit ci-dessus, le jeu de données `poison` contient :
  - des variables continues supplémentaires (`quanti.sup = 1 :2`, colonnes 1 et 2 correspondant aux colonnes `age` et `time`, respectivement)
  - des variables qualitatives supplémentaires (`quali.sup = 3 :4`, correspondant aux colonnes `Sick` et `Sex`, respectivement). Ces variables sont utilisées pour colorer les individus par groupes
 Les données ne contiennent pas des `individus supplémentaires`. Cependant, pour la démonstration, nous utiliserons les individus 53 : 55 en tant qu'individus supplémentaires.

Les variables et les individus supplémentaires ne sont pas utilisés pour déterminer les dimensions principales. Leurs coordonnées sont prédites en utilisant uniquement les informations fournies par l'ACM effectuée sur des variables/individus actifs.

2. *Spécification dans l'ACM* : Pour spécifier des individus et des variables supplémentaires, la fonction `MCA()` peut être utilisée comme suit :
 

```
MCA (X, ind = NULL, quanti = NULL, quali.sup = NULL, graph = TRUE, axes = c (1,2))
```

  - `X` : un data frame. Les lignes sont des individus et les colonnes sont des variables numériques.
  - `ind.sup` : un vecteur numérique spécifiant les positions des individus supplémentaires
  - `quanti.sup, quali.sup` : un vecteur numérique spécifiant, respectivement, les positions des variables quantitatives et qualitatives
  - `graph` : une valeur logique. Si `TRUE`, un graphique est affiché.
 Par exemple, on peut taper ceci :
 

```
res.mca <- MCA (poison, ind = 53 :55, quanti.sup = 1 :2,
quali.sup = 3 :4, graph =FALSE)
```
3. *Résultats* : Les résultats prédites pour les individus/variables supplémentaires peuvent être extraits comme suit :

```

# Variables qualitatives supplémentaires
res.mca$quali.sup
# Variables quantitatives supplémentaires

```

```
res.mca$quanti
# Individus supplémentaires
res.mca$ind.sup
```

4. *Graphique* : Pour créer un biplot des individus et des variables, tapez ceci :

```
# Biplot des individus et des variables
fviz_mca_biplot (res.mca, repel = TRUE, ggtheme = theme_minimal ())
```

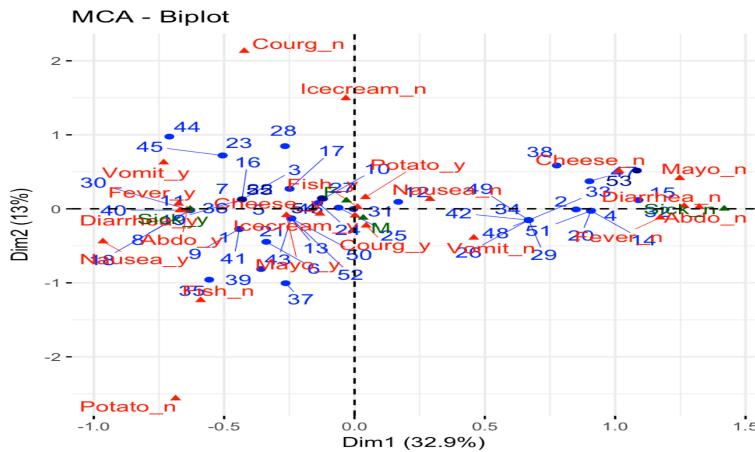


FIGURE 5.13 – Biplot des individus et des variables

- Les individus actifs sont en bleu
- Les individus supplémentaires sont en bleu foncé
- Les catégories des variables actives sont en rouge
- Les catégories des variables supplémentaires sont en vert foncé

Si vous souhaitez mettre en évidence la corrélation entre les variables (actifs et supplémentaires) et les dimensions, utilisez la fonction `fviz_mca_var()` avec l'argument `choice= "mca.cor"` :

```
fviz_mca_var (res.mca, choice = "mca.cor", repel = TRUE)
```

Le code R ci-dessous visualise les catégories des variables qualitatives (variables actives et supplémentaires) :

```
fviz_mca_var(res.mca, repel = TRUE, ggtheme= theme_minimal())
```

Pour les variables quantitatives supplémentaires, tapez ceci :

```
fviz_mca_var(res.mca, choice = "quanti.sup", ggtheme = theme_minimal())
```

Pour visualiser les individus supplémentaires, tapez ceci :

```
fviz_mca_ind(res.mca, label = "ind.sup", ggtheme = theme_minimal())
```

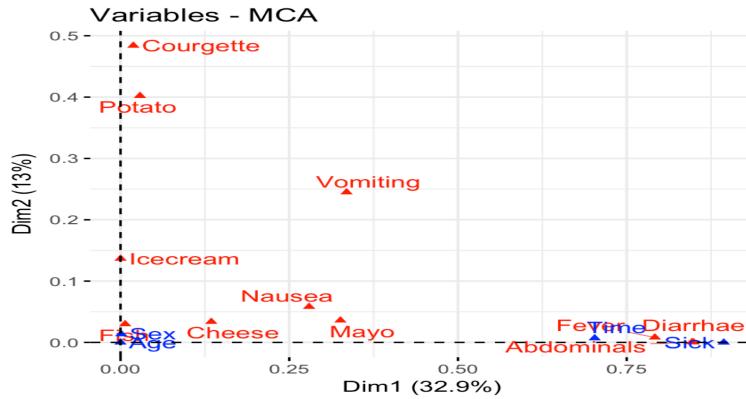


FIGURE 5.14 – Mise en évidence de la corrélation entre les variables (actifs et supplémentaires) et les dimensions

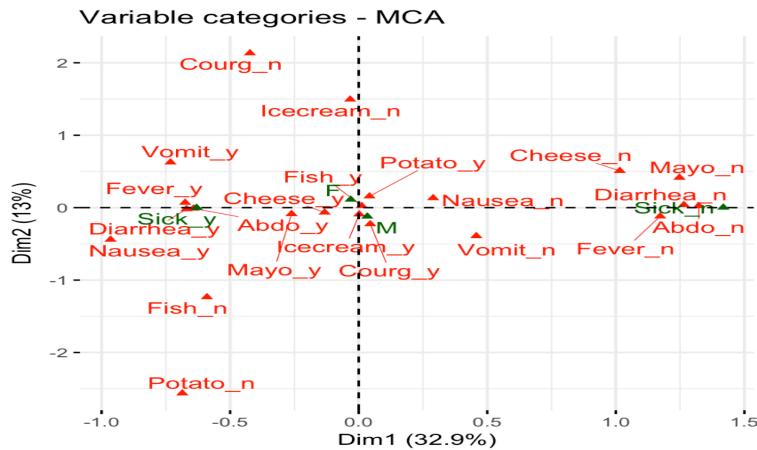


FIGURE 5.15 – Visualisation des catégories des variables qualitatives (variables actives et supplémentaires)

#### 5.1.4 Filtrer des résultats

Si vous avez plusieurs individus/variables, il est possible de visualiser seulement certains d'entre eux en utilisant les arguments `select.ind` et `select.var`.

`select.ind`, `select.var` : une sélection d'individus/variables à visualiser. Les valeurs autorisées sont `NULL` ou une liste contenant le nom des arguments, `cos2` ou `contrib`) :

- `name` : est un vecteur de caractères contenant le nom des individus/variables à visualiser
- `cos2` : si `cos2` est dans  $[0, 1]$ , exemple : 0.6, alors les individus/variables avec un  $\text{cos2} > 0.6$  sont montrés. Si  $\text{cos2} > 1$ , exemple : 5, le top 5 des individus/variables actifs ainsi que le top 5 des individus/variables supplémentaires avec le `cos2` le plus élevé sont montrés

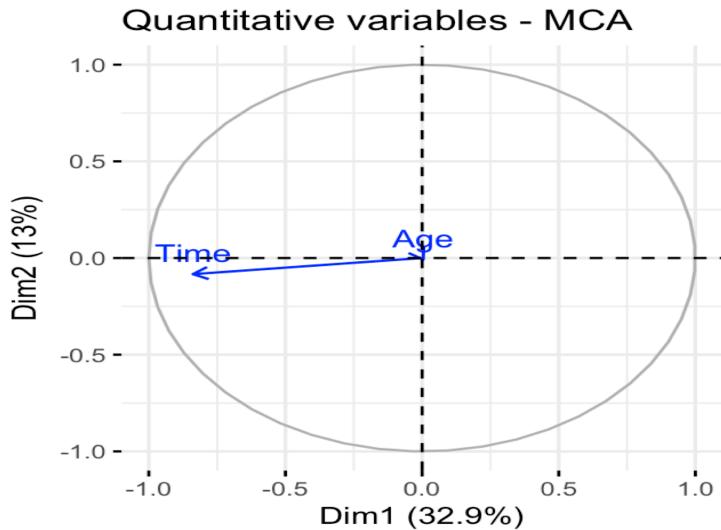


FIGURE 5.16 – Visualisation des variables quantitatives supplémentaires

- `contrib` : si `contrib > 1`, exemple : 5, alors les top 5 individus/variables avec les contributions les plus importantes sont montrés

```
# Visualiser les catégories de variables avec cos2 > = 0.4
fviz_mca_var(res.mca, select.var = list(cos2 = 0.4))
# Top 10 des variables actives avec le cos2 le plus élevé
fviz_mca_var(res.mca, select.var = list(cos2 = 10))
# Sélectionner par noms
name <- list(name = c("Fever_n", "Abdo_y", "Diarrhea_n", "Fever_Y", "Vomit_y",
"Vomit_n"))
fviz_mca_var(res.mca, select.var = name)
# Top 5 des categories de variables les plus contributifs
fviz_mca_biplot(res.mca, select.ind = list(contrib = 5), select.var =
list(contrib = 5),
ggtheme = theme_minimal ())
```

Lorsque la sélection se fait selon les valeurs de contribution, les individus/variables supplémentaires ne sont pas représentés parce qu'ils ne contribuent pas à la construction des axes.

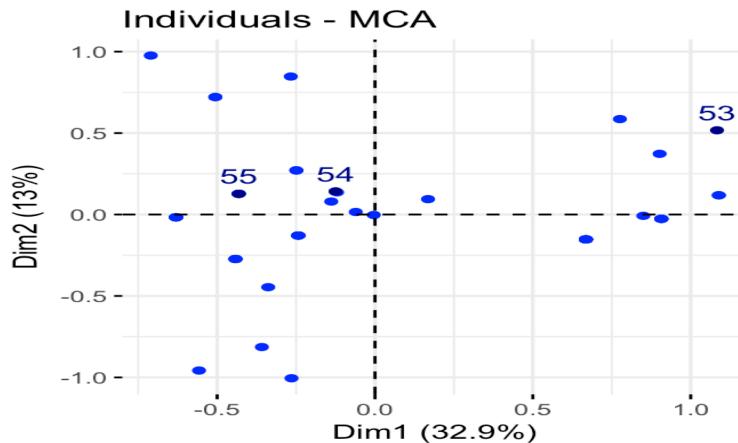


FIGURE 5.17 – Visualisation des individus supplémentaires

### 5.1.5 Exportation des résultats

1. *Exporter les graphiques en PDF/PNG* : Cela se fait en deux étapes :

(a) Créer le graphique d'intérêt en tant qu'objets R :

```
# Scree plot
scree.plot <- fviz_eig (res.mca)
# Biplot des variables de ligne et de colonne
biplot.mca <- fviz_mca_biplot (res.mca)
```

(b) Exporter les graphiques dans un seul fichier pdf comme suit (un graphique par page) :

```
library(ggpubr)
ggeexport(plotlist = list(scree.plot, biplot.mca), filename = "MCA.pdf")
```

2. *Exporter les résultats vers les fichiers txt/csv* : Fonction R facile à utiliser : `write.infile()` [FactoMineR] package.

```
# Exporter vers un fichier TXT
write.infile (res.mca, "mca.txt", sep = "")
```

# Exporter vers un fichier CSV

```
write.infile (res.mca, "mca.csv", sep = ";")
```

### 5.1.6 Résumé

En conclusion, nous avons décrit comment calculer et interpréter l'analyse des correspondances multiples (ACM). Nous avons calculé l'ACM en utilisant la fonction `MCA()` [FactoMineR]. Ensuite, nous avons utilisé le package `factoextra` R pour produire une visualisation `ggplot2` des résultats de l'ACM. Il existe d'autres fonctions [packages] pour calculer l'ACM dans R :

```
dudi.acm() [ade4]
library("ade4")
```

```
res.mca <- dudi.acm(poison.active, scannf = FALSE, nf = 5)

epMCA() [ExPosition]
library("ExPosition")
res.mca <- epMCA(poison.active, graph = FALSE, correction = "bg")
```

Peu importe les fonctions que vous décidez d'utiliser, dans la liste ci-dessus, le package `factoextra` peut gérer le résultat.

```
fviz_eig (res.mca) # Scree plot
fviz_mca_biplot (res.mca) # Biplot
```

### 5.1.7 Autres lectures

Pour les bases mathématiques de l'ACM, referez vous aux cours, articles et livres suivants :

- Exploratory Multivariate Analysis by Example Using R (book) (Husson, Le, and Pages 2017).
- Principal component analysis (article) (Abdi and Williams 2010).  
<https://goo.gl/1Vtwq1>.

## 5.2 Devoir

**Exercice 27.** Récupérer les jeux de données `chiens.rda`. Il s'agit de données fictives où 27 races de chiens sont décrites avec 7 variables qualitatives.

1. Charger le jeu de données chiens dans R avec la commande `load`. Afficher les données. Quelle est la classe de cet objet ?
2. Créez une matrice H contenant la description des  $n = 27$  races canines sur uniquement les  $p = 6$  premières variables.
3. On veut effectuer l'ACM de cette matrice H.
  - (a) Quelle décomposition en valeurs singulières généralisée (GSVD) faut-il faire ? Réaliser cette DSVG avec R.
  - (b) Montrer qu'en ACM, l'inertie totale des données vaut toujours  $m - 1$  où  $m$  est le nombre total  $p$  de modalités et  $p$  le nombre de variables qualitatives. Vérifiez ensuite avec R que la somme des valeurs singulières trouvées à la question précédente vaut bien  $m - 1$ .
  - (c) Vérifiez également que le nombre maximum de dimension de cette ACM vaut bien  $\min(n - 1, m - p)$ .
  - (d) Représenter dans un diagramme en barre les pourcentages d'inertie expliquée par les dimensions de l'ACM.

- (e) Déterminer les matrices  $X$  et  $Y$  des coordonnées factorielles des races de chiens et des modalités des variables qualitatives sur les  $k = 3$  premières dimensions. Modifier les noms des lignes et des colonnes dans X et Y afin qu'ils soient parlants.
  - (f) Faire un plot des individus et des modalités dans le premier plan factoriel.
  - (g) Utiliser la relation quasi-barycentrique pour retrouver les coordonnées factorielles de la modalité T++ à partir des coordonnées factorielles des races de chiens.
  - (h) Quels est le rapport de corrélation entre la variable taille avec la première composante principale ? Entre la variable taille et la seconde composante principale ?
4. On veut maintenant utiliser la fonction MCA du package FactoMineR.
- (a) Faire l'ACM des données sur les races canines en mettant la variable fonction en illustratif.
  - (b) Retrouvez les résultats numériques et les graphiques de la question 2.
  - (c) Retrouver les rapports de corrélations entre les variables qualitatives et les deux premières composantes principales. Faire le plot des variables en fonction de ces rapports de corrélation en utilisant la fonction plot.MCA.
  - (d) Mettre des données manquantes dans les données avec le code suivant :
  - (e) Faire l'ACM de `chiensNA`. Comment les données manquantes sont-elles prises en compte dans la fonction MCA du package FactoMineR ?
5. On veut maintenant comparer l'ACM et l'AFC dans le cas particulier de deux variables qualitatives.
- (a) Avec la fonction CA de FactoMineR, effectuer l'AFC du tableau de contingence croisant les variables taille et poids.
  - (b) Avec la fonction MCA, effectuer l'ACM des deux premières colonnes des données chiens.
  - (c) Comparez les valeurs propres des deux analyses et vérifiez que vous retrouvez les relations du cours.

**Exercice 28.** (ACM avec données manquantes et choix du nombre de composantes)  
Le package R `missMDA` permet de gérer les données manquantes en ACP et en ACM, et de choisir le nombre de composantes par validation croisée.

1. Regarder les vidéos concernant ce package : <https://www.youtube.com/user/HussonFrancois>
2. Préparer un document avec Rmdown qui décrit les principales fonctionnalités de ce package, avec à chaque fois une explication de la méthode, des exemples et du code.



# Bibliographie

- [1] Abdi, Hervé, and Lynne J. Williams. 2010. Principal Component Analysis. John Wiley and Sons, Inc. WIREs Comp Stat 2 : 43359.  
<http://staff.ustc.edu.cn/zwp/teach/MVA/abdi-awPCA2010.pdf>.
- [2] Husson, Francois, Sébastien Le, and Jérôme Pages. 2017. Exploratory Multivariate Analysis by Example Using R. 2nd ed. Boca Raton, Florida : Chapman ; Hall/CRC.  
<http://factominer.free.fr/bookV2/index.html>.



## Chapitre 6

# Classification Automatique CAH et *K*-Means

Nous allons nous intéresser à la classification automatique de fromages à partir des données confinées dans le fichier formage.txt

1. Lancer ensuite R et modifier le répertoire de travail en allant dans **Fichier -> Changer le Répertoire Courant** et en choisissant le répertoire **Bureau/TP\_M1MIASHS\_AD** qui a été créé.
2. Ouvrir une fenêtre d'éditeur **Fichier -> Nouveau Script**.
3. Sauver le fichier dans le répertoire courant sous le nom **TP5.R** : **Fichier -> Sauver sous**
4. Penser à sauvegarder régulièrement le contenu du fichier **TP5.R** en appuyant sur les touches **Ctrl et S**.

### 6.1 Objectifs de l'étude

Ce document retranscrit une démarche de classification automatique d'un ensemble de fromages (29 observations) décrits par leurs propriétés nutritives (ex. **protéines**, **lipides**, etc. ; 9 variables). L'objectif est d'identifier des groupes de fromages homogènes, partageant des caractéristiques similaires.

Nous utiliserons essentiellement deux approches en nous appuyant sur deux procédures du logiciel R : la classification ascendante hiérarchique (CAH) avec **hclust()** ; la méthode des centres mobiles (k-means) avec **kmeans()**.

### 6.1.1 Traitements réalisés

- Chargement et description des données
- Classification automatique avec `hclust()` et `kmeans()`
- Pistes pour la détection du nombre adéquat de classes
- Description – interprétation des groupes

### 6.1.2 Fichier de données : Importation, statistiques descriptives et graphiques

```
— # Modifier le répertoire par défaut
  setwd("... mon dossier...")
— # Charger les données - attention aux options
  fromage <- read.table(file="fromage.txt",header=T,row.names=1,sep="\t",dec=".")
— # Afficher les 6 premières lignes
  print(head(fromage))
— # Stat. descriptives
  print(summary(fromage))
— # Graphique-croisement deux à deux
  pairs(fromage)
```

**Remarque 6.1.1.** Ce type de graphique n'est jamais anodin. Nous constatons par exemple que :

- `lipides` est fortement corrélé avec `calories` et `cholestérol` (sans trop de surprises).
- Dans certaines configurations, des groupes semblent apparaître naturellement (ex. croisement de `protéines` et `cholestérol`, avec une corrélation inter-groupes assez marquée).

## 6.2 Classification Ascendante Hiérarchique

### 6.2.1 La procédure `hclust()` de R (package stats - toujours chargée)

```
— # Centrage réduction des données pour éviter que les variables à forte variance pèsent
  # indûment sur les résultats
  fromage.cr <- scale(fromage,center=T,scale=T)
— # Matrice des distances entre individus
  d.fromage <- dist(fromage.cr)
— # CAH - Critère de Ward
  #method = "ward.D2" correspond au vrai critère de Ward utilisant le carré de la
  #distance
  cah.ward <- hclust(d.fromage,method="ward.D2")
```

```
— # Affichage du dendrogramme
plot(cah.ward)
```

**Remarque 6.2.1.** Le dendrogramme “suggère” un découpage en 4 groupes. On note qu’une classe de fromages, les “fromages frais” (tout à gauche), se démarque fortement des autres au point qu’on aurait pu envisager aussi un découpage en 2 groupes seulement. Nous y reviendrons plus longuement lorsque nous mixerons l’analyse avec une analyse en composantes principales (ACP).

### 6.2.2 Découpage en classes – Matérialisation des groupes

```
— #dendrogramme avec matérialisation des groupes
rect.hclust(cah.ward,k=4)
— #découpage en 4 groupes
groupes.cah <- cutree(cah.ward,k=4)
— #liste des groupes
print(sort(groupes.cah))
```

**Remarque 6.2.2.**

- Le 4ème groupe est constitué de fromages frais.
- Le 3ème de fromages à pâte molle.
- Le 2nd de fromages “durs”.
- Le 1er est un peu fourre-tout !

Pour une caractérisation à l’aide des variables de l’étude, il faut passer par des techniques statistiques univariées (simples à lire) ou multivariées (tenant compte des relations entre les variables).

## 6.3 Méthode des centres mobiles

### 6.3.1 La procédure kmeans() de R (package stats également)

```
— # $K$ -means avec les données centrées et réduites
#center = 4 : nombre de groupes demandés
#nstart = 5 : nombre d’essais avec différents individus de départ (parce que les résultats sont dépendants de l’initialisation)
groupes.kmeans <- kmeans(fromage.cr,centers=4,nstart=5)
— #Affichage des résultats
print(groupes.kmeans)
```

**Remarque 6.3.1.** On peut regarder les correspondances des groupes entre CAH et  $K$ -Means :

```
print(table(groupes.cah,groupes.kmeans$cluster))
```

Le groupe 4 de la CAH coincide avec le groupe 1 des *K*-Means. Après, il y a certes des correspondances, mais elles ne sont pas exactes.

Il faut bien noter qu'il se peut qu'on n'a pas exactement les mêmes résultats avec les *K*-Means.

### 6.3.2 Aide à la détection du nombre adéquat de groupes

*K*-MEANS, à la différence de la CAH, ne fournit pas d'outil d'aide à la détection du nombre de classes.

Nous devons les programmer sous R ou utiliser des procédures proposées par des packages dédiés. Le schéma est souvent le même : on fait varier le nombre de groupes et on surveille l'évolution d'un indicateur de qualité de la solution c.-à -d. l'aptitude des individus à être plus proches de ses congénères du même groupe que des individus des autres groupes.

Deux pistes ici : (1) surveiller l'évolution de la proportion d'inertie expliquée par la partition, on cherche le "coude" dans le graphique (nous programmons la procédure) ; (2) utiliser l'indice de Calinski Harabasz, on recherche alors à maximiser ce second critère (nous utilisons la fonction `kmeansruns()` du package `fpc`, on peut aussi choisir l'indice `silhouette moyenne`)<sup>1</sup>

- # (1) Evaluer la proportion d'inertie expliquée

```
for (k in 2 :10){
  clus <- kmeans(fromage.cr, centers=k, nstart=5)
  inertie.expl[k] <- clus$betweenss/clus$totss
}
```
- # Graphique

```
plot(1 :10,inertie.expl,type="b",xlab="Nb. de groupes",ylab="% inertie expliquée")
```
- # (2) Indice de Calinski Harabasz - utilisation du package `fpc`

```
library(fpc)
```
- # Evaluation des solutions

```
sol.kmeans <- kmeansruns(fromage.cr,krange=2 :10,criterion="ch")
```
- # Graphique

```
plot(1 :10,sol.kmeans$crit,type="b",xlab="Nb. de groupes",ylab="Silhouette")
```

## 6.4 Analyses univariées et multivariées

### 6.4.1 Interprétation des classes : Statistiques comparatives

L'idée est de comparer les moyennes des variables actives conditionnellement aux groupes. Il est possible de quantifier globalement l'amplitude des écarts avec la proportion de variance expliquée. La démarche peut être étendue aux variables illustratives. Pour les catégorielles,

---

1. [https://en.wikipedia.org/wiki/Determining\\_the\\_number\\_of\\_clusters\\_in\\_a\\_data\\_set](https://en.wikipedia.org/wiki/Determining_the_number_of_clusters_in_a_data_set)

nous confronterions les distributions conditionnelles.

L'approche est simple et les résultats faciles à lire. Rappelons cependant que nous ne tenons pas compte des liaisons entre les variables dans ce cas.

```

— # Fonction de calcul des stats
stat.comp <- function(x,y){
  K <- length(unique(y))# Nombre de groupes
  n <- length(x)# Nombre d'observations
  m <- mean(x)# Moyenne globale
  TSS <- sum((x-m)^ 2) # Variabilité totale
  nk <- table(y) # Effectifs conditionnels
  mk <- tapply(x,y,mean) # Moyennes conditionnelles
  BSS <- sum(nk * (mk - m)^ 2) # Variabilité expliquée
  result <- c(mk,100.0*BSS/TSS) # Moyennes + prop. variance expliquée
  names(result) <- c(paste("G",1 :K),"% epl.")# Nommer les éléments du vecteur
  return(result) # Renvoyer le vecteur résultat
}
— # Appliquer stat.comp aux variables de la base originelle fromage et non pas aux
variables centrées et réduites
print(sapply(fromage,stat.comp,y=groupes.cah))

```

**Remarque 6.4.1.** La définition des groupes est – avant tout – dominée par les teneurs en graisses (**lipides**, **cholestérol** et **calories** relèvent de la même idée) et en **protéines**. Le groupe 4 est fortement déterminé par ces variables, les moyennes conditionnelles sont très différentes.

#### 6.4.2 Analyse en composantes principales (ACP)

Avec l'ACP, nous tenons compte des liaisons entre les variables. L'analyse est plus riche. Mais il faut savoir lire correctement les sorties de l'ACP.

```

— #ACP normée
acp <- princomp(fromage,cor=T,scores=T)
— #screeplot - 2 axes retenus
plot(1 :9,acp$sdev^ 2,type="b",xlab="Nb. de facteurs",ylab="Val. Propres")
— #biplot
biplot(acp,cex=0.65)

```

**Remarque 6.4.2.** Il y a un problème. Le groupe des fromages frais écrase l'information disponible et tasse les autres fromages dans un bloc qui s'oriente différemment.

De fait, si l'on comprend bien la nature du groupe 4 des fromages frais, les autres sont plus compliqués à comprendre lorsqu'ils sont replacés dans le premier plan factoriel.

```

— # Positionnement des groupes dans le plan factoriel avec étiquettes des points
  plot(acp$scores[,1],acp$scores[,2],type="n",xlim=c(-5,5),ylim=c(-5,5))
— text(acp$scores[,1],acp$scores[,2],col=c("red","green","blue","black")
  [groupes.cah], cex =0.65, labels=rownames(fromage),xlim=c(-5,5),ylim=c(-5,5))

```

**Remarque 6.4.3.** Pour les groupes 1, 2 et 3 (vert, rouge, bleu), on perçoit à partir du graphique biplot de la page précédente qu'il y a quelque chose autour de l'opposition entre nutriments (lipides, calories, cholestérol, protéines, magnésium, calcium) et vitamines (rétilinol, folates). Mais, dans quel sens exactement ?

La lecture n'est pas facile du fait de l'effet perturbateur du groupe 4.

### 6.4.3 Compléter l'analyse à la lumière des résultats de l'ACP

#### Approfondir l'analyse : Retirer les fromages frais du jeu de données

Les fromages frais sont tellement particuliers – éloignés de l'ensemble des autres observations – qu'ils masquent des relations intéressantes qui peuvent exister entre ces produits. Nous reprenons l'analyse en les excluant des traitements.

```

— # Retirer les 4 obs. du groupe 4
  fromage.subset <- fromage[groupes.cah !=4,]
— # Centrage et réduction
  fromage.subset.cr <- scale(fromage.subset,center=T,scale=T)
— # Matrice des distances
  d.subset <- dist(fromage.subset.cr)
— # cah 2
  cah.subset <- hclust(d.subset,method="ward.D2")
— # Affichage
  plot(cah.subset)
— # Groupes
  groupes.subset <- cutree(cah.subset,k=3)
— # Affichage des groupes
  print(sort(groupes.subset))
— # acp
  acp.subset <- princomp(fromage.subset,cor=T,scores=T)
— # Screeplot — 2 axes retenus
  plot(1 :9,acp.subset$sdev^ 2,type="b")
— # Biplot
  biplot(acp.subset,cex=0.65)
— #Positionnement des groupes dans le plan factoriel
  plot(acp.subset$scores[,1],acp.subset$scores[,2],
    type="n",xlim=c(-6,6),ylim=c(-6,6))
— #Etiquettes des points
  text(acp.subset$scores[,1],acp.subset$scores[,2],

```

```
col=c("red","green","blue")[groupes.subset],  
cex=0.65,labels=rownames(fromage.subset),xlim=c(-6,6),ylim=c(-6,6))
```

**Remarque 6.4.4.** Les résultats ne contredisent pas l'analyse précédente. Mais les concordancesses et oppositions apparaissent plus clairement, notamment sur le 1er facteur. Le positionnement de **folates** est plus explicite.

On peut aussi s'interroger sur l'intérêt de conserver 3 variables qui portent la même information dans l'analyse (**lipides**, **cholestérol** et **calories**).

Les groupes sont constitués essentiellement sur le 1er facteur. Quelques fromages ont changé de camp par rapport à l'analyse précédente : **carré de l'est** et **coulommiers** d'une part ; **cheddar** d'autre part.

## 6.5 Devoir

**Exercice 29.** Appliquer CAH et  $K$ -means aux données fromages.txt après avoir supprimé les fromages frais.