Feuille de Travaux Dirigés no 6
Analyse factorielle des correspondances multiples

Exercice VI.1. Manipulations de tableaux pouvant servir comme support à une AFCM

Nous nous intéressons à un jeu de données fictif. Il comporte les réponses de 10 personnes aux trois questions suivantes :

a) Êtes-vous un homme ou une femme ?

b) Quel est votre niveau de revenus : moyen ou élevé ?

c) Choisissez le dessert que vous préférez parmi les trois suivants : un fruit (A), une crème glacée (B), du chocolat (C) ?

1. Créer les données dans R en exécutant les instructions suivantes.

```
> Chemin <- "C:\\..."
> Sexe <- rep(c("F", "M"), c(5, 5))
> Revenu <- rep(c("M", "E", "M"), c(2, 5, 3))
> Préf. <- c("A", "A", "B", "C", "C", "C", "B", "B", "B",
+     "A")
> Résultats <- data.frame(cbind(Sexe, Revenu, Préf.))
> print(Résultats)
```

	Sexe	Revenu	Préf.
1	F	M	A
2	F	M	A
3	F	E	B
4	F	E	C
5	F	E	C
6	M	E	C
7	M	E	B
8	M	M	B
9	M	M	B
10	M	M	A

2. Sous quelle forme les données sont-elles stockées dans l'objet `Résultats` ? Obtenir la table de contingence `Résultats.cont`, puis le tableau de Burt `Résultats.burt` et enfin le tableau disjonctif complet `Résultats.disj`.

```
> (Résultats.cont <- as.data.frame(table(Résultats)))
```

	Sexe	Revenu	Préf.	Freq
1	F	E	A	0
2	M	E	A	0
3	F	M	A	2
4	M	M	A	1
5	F	E	B	1
6	M	E	B	1
7	F	M	B	0
8	M	M	B	2
9	F	E	C	2
10	M	E	C	1
11	F	M	C	0
12	M	M	C	0

```
> burt <- function(table) {
+     disj <- data.frame(lapply(1:ncol(table), function(i) {
+         col <- table[, i]
+         lev <- names(table)[i]
+         n <- length(col)
+         col <- as.factor(col)
+         x <- matrix(0, n, length(levels(col)))
+         x[(1:n) + n * (unclass(col) - 1)] <- 1
+         dimnames(x) <- list(row.names(table), paste(lev,
+             levels(col), sep = "."))
+         return(x)
+     }))
+     burt <- as.matrix(t(disj)) %*% as.matrix(disj)
+     burt <- data.frame(burt)
+     names(burt) <- names(disj)
+     row.names(burt) <- names(disj)
+     return(burt)
+ }
> (Résultats.burt <- burt(Résultats))
```

	Sexe.F	Sexe.M	Revenu.E	Revenu.M	Préf..A	Préf..B	Préf..C
Sexe.F	5	0	3	2	2	1	2
Sexe.M	0	5	2	3	1	3	1
Revenu.E	3	2	5	0	0	2	3
Revenu.M	2	3	0	5	3	2	0
Préf..A	2	1	0	3	3	0	0
Préf..B	1	3	2	2	0	4	0
Préf..C	2	1	3	0	0	0	3

```
> disj <- function(table) {
+     return(data.frame(lapply(1:ncol(table), function(i) {
+         col <- table[, i]
```

```
+         lev <- names(table)[i]
+         n <- length(col)
+         col <- as.factor(col)
+         x <- matrix(0, n, length(levels(col)))
+         x[(1:n) + n * (unclass(col) - 1)] <- 1
+         dimnames(x) <- list(row.names(table), paste(lev,
+             levels(col), sep = "."))
+         return(x)
+     }))
+ }
> (Résultats.disj <- disj(Résultats))
```

	Sexe.F	Sexe.M	Revenu.E	Revenu.M	Préf..A	Préf..B	Préf..C
1	1	0	0	1	1	0	0
2	1	0	0	1	1	0	0
3	1	0	1	0	0	1	0
4	1	0	1	0	0	0	1
5	1	0	1	0	0	0	1
6	0	1	1	0	0	0	1
7	0	1	1	0	0	1	0
8	0	1	0	1	0	1	0
9	0	1	0	1	0	1	0
10	0	1	0	1	1	0	0

Les fonctions `acm.burt` et `acm.disjonctif` du package `ade4` permettent de créer les tableaux de Burt et disjonctif complet.

```
> library(ade4)
> acm.burt(Résultats, Résultats)
> acm.disjonctif(Résultats)
```

La fonction `tab.disjonctif` du package `FactoMineR` permet de créer le disjonctif complet.

```
> library(FactoMineR)
> tab.disjonctif(Résultats)
```

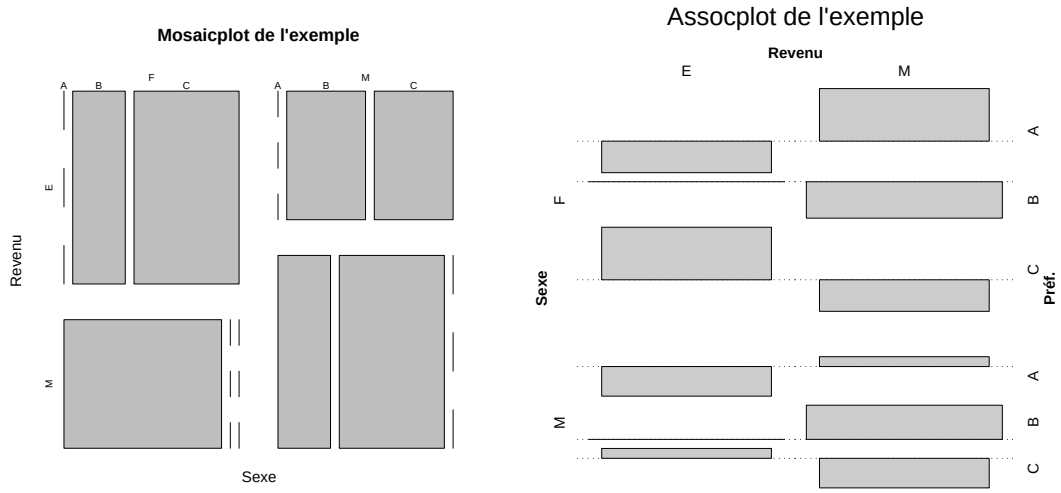
La fonction `expand.dft` reproduite ci-dessous permet de retrouver le tableau des observations à partir du tableau de contingence. Nous nous en servirons à l'exercice **2**.

```
> expand.dft <- function(x, na.strings = "NA", as.is = FALSE,
+     dec = ".") {
+     DF <- sapply(1:nrow(x), function(i) x[rep(i, each = x$Freq[i]),
+         ], simplify = FALSE)
+     DF <- subset(do.call("rbind", DF), select = -Freq)
+     for (i in 1:ncol(DF)) {
+         DF[[i]] <- type.convert(as.character(DF[[i]]),
+             na.strings = na.strings, as.is = as.is,
+             dec = dec)
+     }
```

```
+     DF
+ }
> expand.dft(Résultats.cont)
```

	Sexe	Revenu	Préf.
3	F	M	A
3.1	F	M	A
4	M	M	A
5	F	E	B
6	M	E	B
8	M	M	B
8.1	M	M	B
9	F	E	C
9.1	F	E	C
10	M	E	C

```
> mosaicplot(table(Résultats), main = "Mosaicplot de l'exemple")
> library(vcd)
> assoc(table(Résultats), main = "Assocplot de l'exemple")
```



2. Réaliser l'AFCM du tableau de données `Résultats` des deux manières élémentaires suivantes :

a. L'analyse factorielle des correspondances du tableau de Burt.

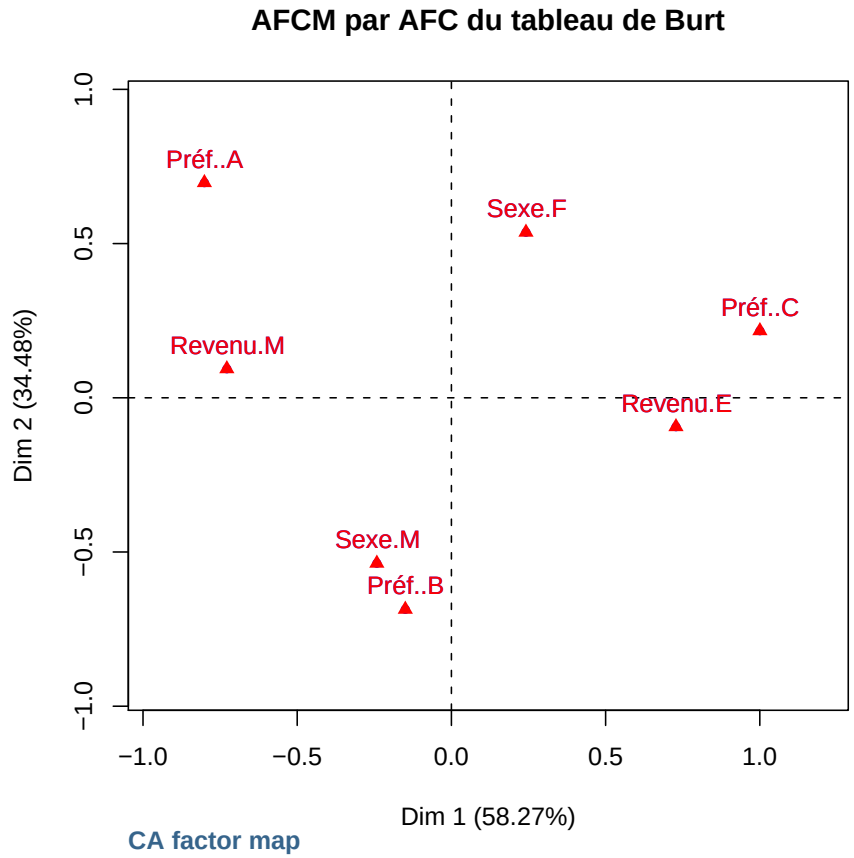
b. L'analyse factorielle des correspondances du tableau de disjonctif complet.

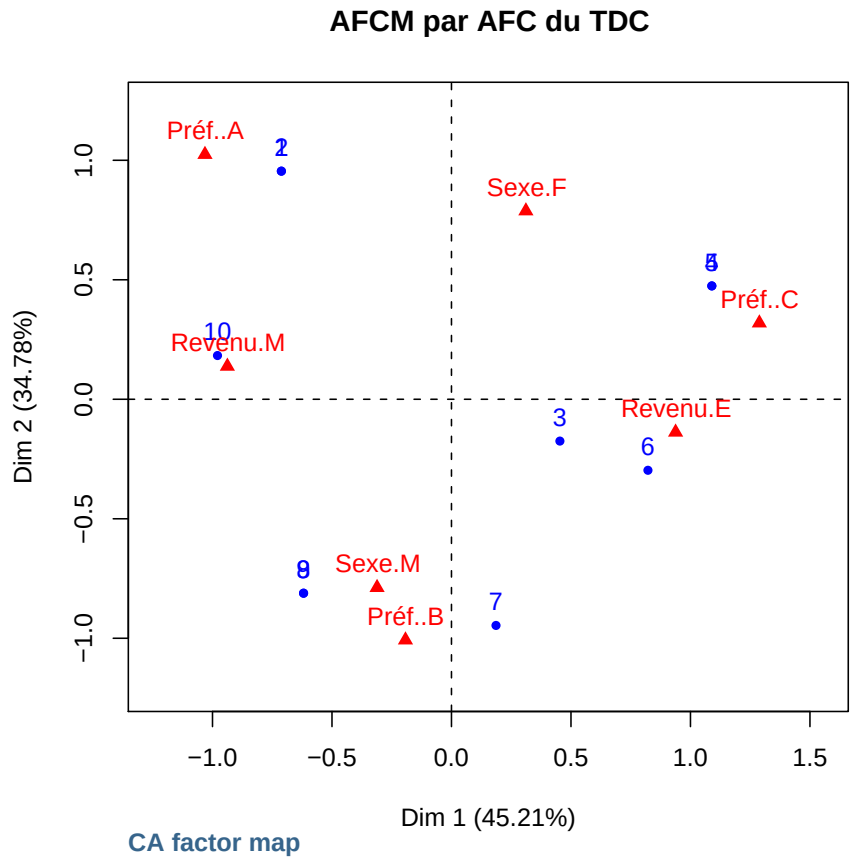
```
> Résultats.burt.acm <- CA(Résultats.burt, graph = F,
+     ncp = 4)
> Résultats.disj.acm <- CA(Résultats.disj, graph = F,
+     ncp = 4)
> Résultats.burt.acm$eig
> Résultats.disj.acm$eig
```

	eigenvalue	percentage of variance	cumulative percentage of variance
dim 1	0.36	58.27	58.27
dim 2	0.22	34.48	92.74
dim 3	0.04	6.60	99.35
dim 4	0.00	0.65	100.00
dim 5	0.00	0.00	100.00
dim 6	0.00	0.00	100.00

	eigenvalue	percentage of variance	cumulative percentage of variance
dim 1	0.60	45.21	45.21
dim 2	0.46	34.78	79.99
dim 3	0.20	15.22	95.21
dim 4	0.06	4.79	100.00
dim 5	0.00	0.00	100.00
dim 6	0.00	0.00	100.00
dim 7	0.00	0.00	100.00

```
> plot(Résultats.burt.acm, title = "AFCM par AFC du tableau de Burt")
> plot(Résultats.disj.acm, title = "AFCM par AFC du TDC")
> plot(Résultats.disj.acm, title = "AFCM par AFC du TDC",
+     invisible = "row")
```





2. Quels commentaires pouvez-vous formuler sur les liens entre les résultats des deux analyses précédentes. Il y a 7 modalités pour 3 variables. Vous vous intéresserez donc, en premier lieu, au lien entre les (7-3=4) quatre premières valeurs propres, les seules qui puissent être non-nulles. Puis aux liens entre les deux représentations graphiques obtenues.

```
> multtest <- mapply(all.equal, (Résultats.burt.acm$eig$eig)[1:4],
+     ((Résultats.disj.acm$eig$eig)[1:4])^2)
> multiden <- sapply(multtest, identical, TRUE)
> all(multiden)
[1] TRUE
```

Remarquer que l'égalité n'a pas été testée avec la fonction `==` mais plutôt avec la fonction `all.equal` qui tolère une legère différence entre les valeurs. Elle est donc utile pour éviter de tenir compte de fluctuations dues aux approximations du calcul numérique et est associée à la fonction `identical`.
L'exemple ci-dessous est édifiant. Le test `x1 == x2` est `FALSE` pour la plupart des utilisateurs tandis que le test `identical(all.equal(x1, x2), TRUE)` est toujours vrai.

```
> x1 <- 0.5 - 0.3
> x2 <- 0.3 - 0.1
> x1 == x2
[1] FALSE
```

```
> identical(all.equal(x1, x2), TRUE)
[1] TRUE
```

Ainsi les valeurs propres non-nulles de l'analyse factorielle des correspondances du tableau de Burt sont égales au carré des valeurs propres non-nulles de l'analyse factorielle des correspondances du TDC.

```
> (rapcol <- (Résultats.burt.acm$col$coord/Résultats.disj.acm$col$coord))
> (sqvptdc <- sqrt((Résultats.disj.acm$eig$eig)[1:4]))
> rapcol %*% diag(1/sqvptdc)
```

	Dim 1	Dim 2	Dim 3	Dim 4
Sexe.F	0.78	0.68	0.45	0.25
Sexe.M	0.78	0.68	0.45	0.25
Revenu.E	0.78	0.68	0.45	0.25
Revenu.M	0.78	0.68	0.45	0.25
Préf..A	0.78	0.68	0.45	0.25
Préf..B	0.78	0.68	0.45	0.25
Préf..C	0.78	0.68	0.45	0.25

	1	2	3	4
1	0.78	0.68	0.45	0.25

	1	2	3	4
Sexe.F	1.00	1.00	1.00	1.00
Sexe.M	1.00	1.00	1.00	1.00
Revenu.E	1.00	1.00	1.00	1.00
Revenu.M	1.00	1.00	1.00	1.00
Préf..A	1.00	1.00	1.00	1.00
Préf..B	1.00	1.00	1.00	1.00
Préf..C	1.00	1.00	1.00	1.00

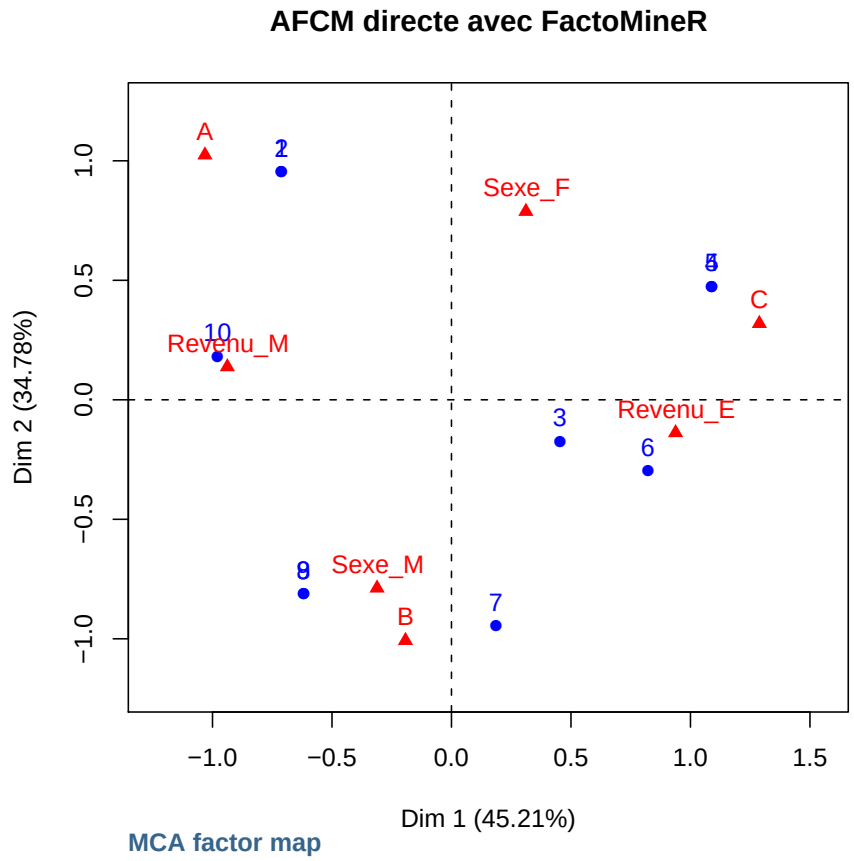
Les coordonnées des colonnes, donc celles des modalités des variables, obtenues par AFC du tableau de Burt sont donc égales l'image de celles obtenues par AFC du TDC par l'application linéaire diagonale égale aux racines carrées des valeurs propres obtenues lors l'AFC du TDC.

Les réprésentations graphiques des modalités des variables sur les axes factoriels ou sur les plans factoriels sont donc image l'une de l'autre par une affinité. Les graphiques obtenus ne seront donc pas la plupart du temps identiques.

3. Réaliser l'analyse factorielle des correspondances multiples du tableau initial avec la fonction `MCA` du package `FactoMineR`. Quelle est l'approche utilisée par cette fonction : AFC du tableau de Burt ou AFC du TDC ?

```
> Résultats.init.acm <- MCA(Résultats, graph = F, ncp = 4)
> Résultats.init.acm$eig
```

```
> equaltest <- function(xx1, xx2) {
+     return(all(sapply(mapply(all.equal, xx1, xx2), identical,
+         TRUE)))
+ }
> vpburt <- (Résultats.burt.acm$eig$eig)[1:4]
> vpdisj <- (Résultats.disj.acm$eig$eig)[1:4]
> vpinit <- (Résultats.init.acm$eig$eig)[1:4]
> equaltest(vpinit, vpburt)
[1] FALSE
> equaltest(vpinit, vpdisj)
[1] TRUE
> plot(Résultats.init.acm, title = "AFCM directe avec FactoMineR")
> plot(Résultats.init.acm, title = "AFCM directe avec FactoMineR",
+     invisible = "ind")
> plot(Résultats.init.acm, title = "AFCM directe avec FactoMineR",
+     new.plot = FALSE)
```



Exercice VI.2. Admissions d'étudiants à l'Université de Californie, Berkeley

Le jeu de données `UCBAdmissions` est un tableau à tridimensionnel comportant 4526 observations de 3 variables sur les étudiants ayant postulés à l'admission à l'Université de Californie, Berkeley, en 1973. Les modalités des trois variables qualitatives qui ont été observées sur chacun des étudiants ont été reportées dans le tableau ci-dessous.

N°	Nom_en	Modalités_en	Nom_fr	Modalités_fr
1	Admit	Admitted, Rejected	Résultat d'admission	Admis, Rejeté
2	Gender	Male, Female	Sexe	Homme, Femme
3	Dept	A, B, C, D, E, F	Departement	A, B, C, D, E, F

1. Le jeu de données `UCBAdmissions` est directement disponible dans **R**. Représenter les données avec la fonction `assoc` et la fonction `mosaicplot`.
2. Procéder à l'analyse des correspondances multiples avec la fonction `MCA` du package `FactoMineR`. Quel est le problème ? Afficher le code de la fonction `MCA`. La première instruction de cette fonction est `X <- as.data.frame(X)`. La fonction `MCA` réalise donc l'analyse des correspondances multiples du tableau `data.frame(UCBAdmissions)`.
   Afficher le tableau `data.frame(UCBAdmissions)` et identifier la cause de l'échec de la première analyse factorielle des correspondances multiples.
3. Transformer la `table UCBAdmissions` en un tableau disjonctif complet à l'aide de la fonction `expand.dft` introduite à l'exercice **1**. Procéder alors à l'analyse factorielle des données multiples du jeu de données.

```
> UCBA.df <- expand.dft(data.frame(UCBAdmissions))
> head(UCBA.df)
> tail(UCBA.df)
> str(UCBA.df)
> table(UCBA.df)
```