Projet STATIS: Réponse feuille de route

ATTOUMANI Ibrahim et MANNEQUIN Jeanne

Contents

Introduction

L'ACP que nous avons l'habitude de voir est une méthode d'analyse de données qui permet de résumer les informations contenues dans un tableau de données. Dans le cas de plusieurs tableaux, il nous faut trouver une méthode permettant d'analyser les données dans leur ensemble.
On se propose d'étendre l'ACP à l'analyse d'un multi-tableau. L'objectif est de comparer les individus et les variables de plusieurs tableaux. Il existe plusieurs façons de la faire, STATIS en est une. STATIS est une méthode qui permet de synthétiser les informations contenues dans les différents tableaux en une seule représentation. Il existe aussi plusieurs types de multi-tableaux. Nous en distinguons deux : les multi-tableaux à trois entrées (situation 1) et les multi-tableaux avec partition thématique (situation 2).
- Le tableau à trois entréees est un tableau de taille $n \times p \times T$, où n est le nombre d'individus, p le nombre de variables et T les différentes dates. Les indices seront notés $1 \leq i \leq n$, $1 \leq j \leq p$ et $1 \leq t \leq T$
- Le tableau avec partition thématique est composé de q tableaux décrivants les n individus à l'aide de groupes de variables différents, chaque groupe appartenant conceptuellement à un thème précis. Chacun de ces tableaux X_m, avec $1 \leq m \leq q$, possède p_m variables (colonnes).

1 Situation 1

Dans cette première partie nous utilisons les tableaux à trois entrées. Un tel tableau peut être décomposé en T tableaux juxtaposés de dimensions identiques (n, p). Cette identité de dimension entre les T tableaux homologues permet une extension de l'ACP dans laquelle on considère chaque tableau comme une "variable" décrivant $n \times p$ "individus" (i, j). Nous formalisons d'abord cette extension de l'ACP, STATIS 1.

1.1 Préliminaires

Il est possible de fabriquer ou de trouver un tableau à trois entrées (INSEE, ...). R propose justement un jeu de données (simulated) du package "multiblock", composé de 4 tableaux (A,B, C et D), de chacun 200 individus et 10 varaibles. Ces tableaux, variables et individus ne portent pas de nom mais des lettres ou numéros.

Puisque nos valeurs sont des indicateurs numériques, on les centre-réduit.

1.1.1 Produit scalaire

La matrice diagonale des poids des n individus est W (par défaut, $W = \frac{1}{n} I_n$). On considérera également une matrice diagonale des poids des p colonnes : $C = \frac{1}{p} I_p$ par défaut.

1. Le produit scalaire entre deux matrices A et B de taille (n, p) est :

$$[A|B] = \mathrm{tr}(CA'WB)$$

La norme d'une matrice A correspondant à ce produit scalaire sera notée $[|A|]$.

a) Ecrivons le produit scalaire sous forme $tr(\tilde{A}'\tilde{B})$ (produit scalaire de Frobénius) en explicitant la transformation Z vers $\tilde{Z}$, $\forall Z$ (n, p).

Soit $A, B \in M_{n,p}(\mathbb{R})$

$$\begin{aligned}
[A \mid B] &= \mathrm{tr}(CA'WB) \\
&= \mathrm{tr}(C^{\frac{1}{2}} A' W^{\frac{1}{2}} W^{\frac{1}{2}} B C^{\frac{1}{2}}) \\
&= \mathrm{tr}((W^{\frac{1}{2}} A C^{\frac{1}{2}})' W^{\frac{1}{2}} B C^{\frac{1}{2}}) \\
&= \mathrm{tr}(\tilde{A}'\tilde{B})
\end{aligned}$$

où $\forall (n,p)\ \tilde{Z} = W^{\frac{1}{2}} Z C^{\frac{1}{2}}$

b) Pour montrer que $[A \mid B] = \mathrm{tr}(\tilde{A}'\tilde{B})$ est un produit scalaire, nous devons démontrer les propriétés suivantes :

(i) Symétrie : $\forall\ A, B \in M_{n,p}(\mathbb{R})$

$$[A \mid B] = [B \mid A]$$

(ii) Linéarité : $\forall\ A, B_1, B_2 \in M_{n,p}(\mathbb{R})$ et $\alpha, \beta \in \mathbb{R}$

$$[A \mid (\alpha B_1 + \beta B_2)] = \alpha[A \mid B_1] + \beta[A \mid B_2]$$

(iii) Positivité définie :$\forall\ A \in M_{n,p}(\mathbb{R})$

$$[A \mid A] \geq 0 \quad \text{et} \quad [A \mid A] = 0 \iff A = 0.$$

Vérifions ces propriétés :

(i) Soit $A, B \in M_{n,p}(\mathbb{R})$

$$\begin{aligned}
[A \mid B] &= \mathrm{tr}(\tilde{A}'\tilde{B}) \\
&= \mathrm{tr}((W^{\frac{1}{2}} A C^{\frac{1}{2}})' W^{\frac{1}{2}} B C^{\frac{1}{2}}) \\
&= \mathrm{tr}(C^{\frac{1}{2}} A' W^{\frac{1}{2}} W^{\frac{1}{2}} B C^{\frac{1}{2}})
\end{aligned}$$

Or, par la propriété de la trace, $\mathrm{tr}(AB) = \mathrm{tr}(BA)$ et l'invariance par transposition des matrices de poids $W' = W$ et $C' = C$, on a alors:

$$\begin{aligned}
\mathrm{tr}(C^{\frac{1}{2}} A' W^{\frac{1}{2}} W^{\frac{1}{2}} B C^{\frac{1}{2}}) &= \mathrm{tr}(C^{\frac{1}{2}} B' W^{\frac{1}{2}} W^{\frac{1}{2}} A C^{\frac{1}{2}}) \\
&= [B \mid A]
\end{aligned}$$

(ii) Soit $A, B_1, B_2 \in M_{n,p}(\mathbb{R})$ et $\alpha, \beta \in \mathbb{R}$

$$\begin{aligned}
[A \mid (\alpha B_1 + \beta B_2)] &= \mathrm{tr}(C^{\frac{1}{2}} A' W^{\frac{1}{2}} W^{\frac{1}{2}} (\alpha B_1 + \beta B_2) C^{\frac{1}{2}}) \\
&= \alpha \mathrm{tr}(C^{\frac{1}{2}} A' W^{\frac{1}{2}} W^{\frac{1}{2}} B_1 C^{\frac{1}{2}}) + \beta \mathrm{tr}(C^{\frac{1}{2}} A' W^{\frac{1}{2}} W^{\frac{1}{2}} B_2 C^{\frac{1}{2}}) \\
&= \alpha \mathrm{tr}(\tilde{A}'\tilde{B}_1) + \beta \mathrm{tr}(\tilde{A}'\tilde{B}_2) \\
&= \alpha[A \mid B_1] + \beta[A \mid B_2]
\end{aligned}$$

La linéarité de la trace assure que cette propriété est respectée.

(iii) Soit $A \in M_{n,p}(\mathbb{R})$, on note $(\tilde{a}_{ij})$ le terme générique de la matrice $\tilde{A}$. On pose $\Delta = \tilde{A}'\tilde{A}$ et on note (δ_{ij}) le terme générique de la matrice Δ.

$$\begin{aligned}
[A \mid A] &= \mathrm{tr}(C^{\frac{1}{2}} A' W^{\frac{1}{2}} W^{\frac{1}{2}} A C^{\frac{1}{2}}) \\
&= tr(\tilde{A}'\tilde{A}) \\
&= tr(\Delta) \\
&= \sum_{i=1}^{p} \delta_{ii}
\end{aligned}$$

Or, $\delta_{ij} = \sum_{k=1}^{p} \tilde{a}_{ki}\tilde{a}_{kj}$ donc:

$$\begin{aligned} tr(\tilde{A}'\tilde{A}) &= tr(\Delta) \\ &= \sum_{i=1}^{p}\sum_{k=1}^{n} \tilde{a}_{ki}\tilde{a}_{ki} \\ &= \sum_{i=1}^{p}\sum_{k=1}^{n} \tilde{a}_{ki}^2 \geq 0 \end{aligned}$$

On suppose que $[A \mid A] = 0$ montrons que $A = 0_{\mathbb{R}^{n\times p}}$:

$$\begin{aligned} & [A \mid A] = 0 \\ \iff & tr(\tilde{A}'\tilde{A}) = 0 \\ \iff & tr(\Delta) = 0 \\ \iff & \sum_{i=1}^{p} \delta_{ii} = 0 \\ \iff & \sum_{i=1}^{p}\sum_{k=1}^{n} \tilde{a}_{ki}^2 = 0 \end{aligned}$$

Cela implique que

$$\forall (i,k) \in [|1,p|] \times [|1,n|], \quad \tilde{a}_{ki}^2 = 0$$

donc

$$\forall (i,k) \in [|1,p|] \times [|1,n|], \quad \tilde{a}_{ki} = 0$$

donc

$$\tilde{A} = W^{\frac{1}{2}} A C^{\frac{1}{2}} = 0_{\mathbb{R}^{n\times p}}$$

On en déduit que $A = 0_{\mathbb{R}^{n\times p}}$ car W et C sont régulières donc inversibles.

Par conséquent, la quantité $[A \mid B] = \mathrm{tr}(\tilde{A}'\tilde{B})$ est un produit scalaire, car elle satisfait toutes les propriétés requises.

c) Écrivons le produit scalaire précédent sous forme de double somme. Soit $A, B \in M_{n,p}(\mathbb{R})$ on note $(\tilde{a}_{ij})$ respectivement $(\tilde{b}_{ij})$ le terme générique de la matrice $\tilde{A}$ respectivement $\tilde{B}$. On pose $\Omega = \tilde{A}'\tilde{B}$ et on note (ω_{ij}) le terme générique de la matrice de Ω. Ainsi:

$$\begin{aligned} [A \mid B] &= tr(\tilde{A}'\tilde{B}) \\ &= tr(\Omega) \\ &= \sum_{i=1}^{p} \omega_{ii} \\ &= \sum_{i=1}^{p}\sum_{k=1}^{n} \tilde{a}_{ki}\tilde{b}_{ki} \end{aligned}$$

Or, $\forall A, \tilde{A}$ a pour élément générique $\tilde{a}_{ki} = \sqrt{w_k}\sqrt{c_i}a_{ki}$ donc :

$$[A \mid B] = \sum_{k=1}^{n}\sum_{i=1}^{p} w_k c_i a_k i b_{ki}$$

d) La fonction "prd_scalaire" est définie en Annexe. Toutefois, pour mieux comprendre son utilisation, nous présentons ci-dessous un exemple concret de son application. Le programme du précédent produit scalaire est affiché, accompagné de la norme associée à ce produit scalaire.

La fonction calculant le produit scalaire de Frobénius

Considérons à présent un exemple d'application de ma fonction **prd_scalaire** (le produit scalaire de Frobénius) ci-dessous:

Nous commençons par initialiser les matrices A et B avec les valeurs suivantes :

$$A = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \\ 0 & 2 & -1 \end{pmatrix}$$

$$B = \begin{pmatrix} -1 & 1 & 2 \\ -1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & -1 \end{pmatrix}$$

Maintenant que les matrices A et B sont initialisées, nous pouvons appliquer la fonction prd_scalaire pour calculer le produit scalaire de Frobenius entre elles.

```
# Initialisation de A et B
a <- c(0,1,1,0,-1,0,1,2,0,-1,0,-1)
b <- c(-1,-1,1,0,1,0,0,1,2,1,0,-1)
A  = matrix(a, nrow =4)
B = matrix(b, nrow =4)

# On applique la fonction prd_scalaire
resultat = prd_scalaire(A,B)
```

Suite à l'application de la fonction prd_scalaire sur les matrices A et B, le résultat obtenu est 0.08333333, donc $[A \mid B] = 0.08333333$.

Après avoir calculé le produit scalaire de Frobenius entre les matrices A et B, nous allons maintenant calculer la norme associée à ce produit scalaire ($[|\cdot|] = \sqrt{[\cdot \mid \cdot]}$) pour chaque matrice, A et B. La fonction pour calculer la norme associée à ce produit est définie en Annexe.

```
# On utilise les matrices précédente afin de calculer leurs normes
rn1 = norme(A)
rn2 = norme(B)
```

Nous obtenons après calcul les résultats suivants: $[|A|] = 0.9128709$ et $[|B|] = 0.9574271$

1.1.2 Coefficient RV

On définit le coefficient de RV d'Escoufier entre deux matrices A et B de taille (n, p) par :

$$R(A, B) = \frac{[A \mid B]}{[|A|]\,[|B|]}$$

a) Géométriquement, le coefficient de RV représente le cosinus entre les matrices A et B.

b) Les fonctions pour calculer le coefficient de RV et fournir la matrice des coefficients de RV en (n,p) tableaux $X_{(n,p)}$ seront présentées en Annexe. En revanche, ici, nous expliciterons un exemple d'application de celles-ci.

Supposons que nos matrices X_1, X_2, X_3 et X_4 sont toutes de dimension 3×2:

$$X_1 = \begin{pmatrix} 2 & -1 \\ 0 & -2 \\ 1 & 1 \end{pmatrix}, \quad X_2 = \begin{pmatrix} 1 & -2 \\ -1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X_3 = \begin{pmatrix} -1 & 0 \\ 2 & 1 \\ 0 & -1 \end{pmatrix}, \quad X_4 = \begin{pmatrix} 0 & -1 \\ 1 & 2 \\ -2 & 0 \end{pmatrix}$$

$$[X_1|X_2] = \mathrm{tr}(CX_1'WX_2)$$

$$\frac{1}{6}\begin{pmatrix} 2 & 0 & 1 \\ -1 & -2 & 1 \end{pmatrix}\begin{pmatrix} 1 & -2 \\ -1 & 0 \\ 0 & 1 \end{pmatrix} = \frac{1}{6}\begin{pmatrix} 2 & -3 \\ 1 & 3 \end{pmatrix}$$

donc $[X_1|X_2] = \frac{5}{6}$. Or, $\|X_1\| = \frac{11}{6}$ et $\|X_2\| = \frac{7}{6}$, donc $R(X_1, X_2) = \frac{[X_1|X_2]}{\|X_1\|\|X_2\|} = \frac{\frac{5}{6}}{\frac{11}{6}\frac{7}{6}} = \frac{30}{77}$

De manière analogue, on obtient les résultats suivants:

- $[X_1|X_3] = \frac{-5}{6}$
- $[X_1|X_4] = \frac{-5}{6}$
- $[X_2|X_3] = \frac{-4}{6}$
- $[X_2|X_4] = \frac{1}{6}$
- $[X_3|X_4] = \frac{11}{6}$
- $\|X_3\| = \frac{7}{6}$
- $\|X_4\| = \frac{10}{6}$

En notant R_{ij} le terme général de la matrice R des coefficients RV, on a:

$$R_{ij} = \frac{[X_i \mid X_j]}{\|X_i\| \, \|X_j\|}$$

On en déduit alors que:

$$R = \begin{pmatrix} 1 & \frac{30}{77} & \frac{-30}{77} & \frac{-30}{110} \\ \frac{30}{77} & 1 & \frac{-24}{49} & \frac{6}{70} \\ \frac{-30}{77} & \frac{-24}{49} & 1 & \frac{24}{70} \\ \frac{-30}{110} & \frac{6}{70} & \frac{24}{70} & 1 \end{pmatrix}$$

Pour appliquer la fonction coeff_RV, nous utiliserons les données citées en introduction.

```
X1 = as.matrix(read.csv("2021_filtre.csv", row.names = 1, header = T))
X2 = as.matrix(read.csv("2022_filtre.csv", row.names = 1, header = T))
X3 = as.matrix(read.csv("2023_filtre.csv", row.names = 1, header = T))

filtre <- list(X1 = X1, X2 = X2, X3 = X3)

resultats_n = matcoef_RV(filtre)
```

Le résultat obtenu est la matrice suivante arrondie à 10^{-4} :

$$\begin{pmatrix} 1.0000 & 0.9992 & 0.9989 \\ 0.9992 & 1.0000 & 0.9991 \\ 0.9989 & 0.9991 & 1.0000 \end{pmatrix}$$

1.2 Programme de STATIS 1

1.2.1 Programme

a) Dans le contexte de l'ACP des tableaux juxtaposés, les "variables" sont les différents tableaux $\mathbf{X_t}$, et les "individus" sont les observations à chaque période de temps. L'expression $\left[\left|\sum_{t=1}^{T} \frac{u_t}{[|X_t|]} X_t\right|\right]^2$ représente l'inertie le long d'un axe $< u >$, où u est un vecteur I-unitaire: $||u||^2 = 1$. On cherche à projeter I-orthogonalement le nuage direct sur un espace $E_k =< u_1, ..., u_k >$ de dimension $k < p$. cela revient à maximiser l'inertie du nuage direct projeté sur $< u >$.

$$\begin{aligned}\max_{||u||^2=1} \left[\left|\sum_{t=1}^{T} \frac{X_t}{[|X_t|]} u_t\right|\right]^2 &= \max_{||u||^2=1} \sum_{t=1}^{T}\sum_{\tau=1}^{T} u_\tau \frac{[X_\tau \mid X_t]}{[|X_t|]\,[|X_\tau|]} u_t \\ &= \max_{||u||^2=1} u'Ru\end{aligned}$$

où:

1. X_t: $n \times p$ (la matrice X_t a des dimensions $n \times p$).
2. $\frac{u_t}{[|X_t|]} X_t$: $n \times p$ (chaque colonne de X_t est pondérée par $\frac{u_t}{[|X_t|]}$).
3. $\sum_{t=1}^{T} \frac{u_t}{[|X_t|]} X_t$: $n \times p$ (somme des termes précédents sur t).
4. $\left(\sum_{t=1}^{T} \frac{u_t}{[|X_t|]} X_t\right)'$: $p \times n$ (transposée de la matrice résultante).
5. $\left[\left|\sum_{t=1}^{T} \frac{u_t}{[|X_t|]} X_t\right|\right]^2$: 1×1 (la norme euclidienne au carré est un scalaire).
6. R est la matrice des coefficients RV

b) Résolvons le programme ci-dessus, le langrangien associé s'écrit:

$$L(u, \lambda) = u'Ru - \lambda(\|u\|^2 - 1)$$

On a alors :

$$\begin{cases} \frac{\partial L}{\partial u}(u, \lambda) = 2Ru - 2\lambda u \\ \frac{\partial L}{\partial \lambda}(u, \lambda) = \|u\|^2 - 1 \end{cases}$$

Les conditions de premier ordre donnent:

$$\iff (S) \begin{cases} Ru = \lambda u & (*) \\ \|u\|^2 = 1 & (**) \end{cases}$$

On a par suite: $u' \quad (*)$ et $(**) \Rightarrow u'Ru = \lambda$.

D'après, (S) on en déduit que les vecteurs u solution de premier ordre sont les vecteurs propres de la matrice $R = ((R(X_t, X_\tau)))_{t,\tau}$ des coefficients RV d'Escoufier entre T tableaux $X_t(n, p)$. Or, pour tout vecteur propre u de R (tel que $\|u\| = 1$) de valeur propre λ, on a $u'Ru = \lambda$.

La valeur maximale de $u'Ru$ est obtenue pour les vecteurs propres associés à la plus grande valeur propre de R.

c) Nous allons écrire le programme R fournissant les vecteurs u solutions des équations du premier ordre.

Considérons un exemple d'application de la fonction vecval_prop:

```
# on applique vecval_prop aux données simulated
vecval = vecval_prop(filtre)

val_prop <- vecval$val_prop
vec_prop <- vecval$vec_prop
```

Aprés exécution de la fonction vecval_prop les résultats suivants:

$$U = \begin{pmatrix} -0.5773 & -0.6214 & 0.5296 \\ -0.5773 & -0.1479 & -0.8029 \\ -0.5773 & 0.7693 & 0.2734 \end{pmatrix}$$

$$\lambda = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} = \begin{pmatrix} 2.9982 \\ 0.0010 \\ 0.0006 \end{pmatrix}$$

où, U est la matrice des vecteurs propres et λ les valeurs propres associées arrondie à 10^{-4}.

Montrons à présent que les vecteurs u obtenus forment une base I-orthonormée.

Reprenons la matrice R des coefficients RV précédente. Soient u_i et u_j des vecteurs propres de R associés aux valeurs propres λ_i et λ_j distinctes. Comme u_i et u_j sont des solutions du problème suivant :

$$\max_{||u||^2=1} u'Ru$$

on a alors $\|u_i\|^2 = 1$ et $\|u_j\|^2 = 1$. Il suffit de montrer que u_i et u_j sont orthogonaux et ainsi de conclure que les vecteurs u forment une base I-orthonormée. En d'autres termes, nous allons montrer que $u_i'u_j = 0$.

Sachant que u_i et u_j sont des vecteurs propres de R associés aux valeurs propres λ_i et λ_j, nous avons :

$$\begin{cases} Ru_i = \lambda_i u_i \\ Ru_j = \lambda_j u_j \end{cases}$$

Par la suite, on a:

$$\begin{aligned} u_i'Ru_j &= \lambda_j u_i'u_j \\ \iff u_i'R'u_j &= \lambda_j u_i'u_j \quad \text{(car } R \text{ est une matrice symétrique)} \\ \iff (Ru_i)'u_j &= \lambda_j u_i'u_j \\ \iff (\lambda_i u_i)'u_j &= \lambda_j u_i'u_j \\ \iff \lambda_i u_i'u_j &= \lambda_j u_i'u_j \\ \iff (\lambda_i - \lambda_j)u_i'u_j &= 0 \\ \iff u_i'u_j &= 0 \quad (\text{car } \lambda_i \neq \lambda_j) \end{aligned}$$

On en déduit donc que les vecteurs u forment une base I-orthonormée.

1.2.2 Équivalence avec l'ACP d'un tableau juxtaposé "dépliant" le tableau cubique

a) Il suffit de décomposer X_t comme une juxtaposition de colonnes:

$$X_t = \left[x_t^1 \dots x_t^p\right]_{\text{Profils Colonnes}} = \begin{bmatrix} x_{t,1} \\ \vdots \\ x_{t,n} \end{bmatrix}_{\text{Profils Lignes}}$$

et de le réécrire sous forme "verticalisée":

$$y^t = \begin{pmatrix} x_t^1 \\ \vdots \\ x_t^p \end{pmatrix} \in \mathbb{R}^{np} \quad \text{où} \quad \forall j \in [|1,p|], x_t^j = \begin{pmatrix} x_{t,1}^j \\ \vdots \\ x_{t,n}^j \end{pmatrix}$$

Avant de démontrer l'égalité des produits scalaires, définissons d'abord le Produit Kronecker:

Soient A une matrice de taille $m \times n$ et B une matrice de taille $p \times q$. Leur produit tensoriel est la matrice $A \otimes B$ de taille mp par nq, définie par blocs successifs de taille $p \times q$, le bloc d'indice i, j valant $a_{ij}B$.

En d'autres termes,

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix}$$

Ou encore, en détaillant les coefficients,

$$A \otimes B = \begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & \cdots & a_{11}b_{1q} & \cdots & \cdots & a_{1n}b_{11} & a_{1n}b_{12} & \cdots & a_{1n}b_{1q} \\ a_{11}b_{21} & a_{11}b_{22} & \cdots & a_{11}b_{2q} & \cdots & \cdots & a_{1n}b_{21} & a_{1n}b_{22} & \cdots & a_{1n}b_{2q} \\ \vdots & \vdots & \ddots & \vdots & & & \vdots & \vdots & \ddots & \vdots \\ a_{11}b_{p1} & a_{11}b_{p2} & \cdots & a_{11}b_{pq} & \cdots & \cdots & a_{1n}b_{p1} & a_{1n}b_{p2} & \cdots & a_{1n}b_{pq} \\ \vdots & \vdots & & \vdots & \ddots & & \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots & & \ddots & \vdots & \vdots & & \vdots \\ a_{m1}b_{11} & a_{m1}b_{12} & \cdots & a_{m1}b_{1q} & \cdots & \cdots & a_{mn}b_{11} & a_{mn}b_{12} & \cdots & a_{mn}b_{1q} \\ a_{m1}b_{21} & a_{m1}b_{22} & \cdots & a_{m1}b_{2q} & \cdots & \cdots & a_{mn}b_{21} & a_{mn}b_{22} & \cdots & a_{mn}b_{2q} \\ \vdots & \vdots & \ddots & \vdots & & & \vdots & \vdots & \ddots & \vdots \\ a_{m1}b_{p1} & a_{m1}b_{p2} & \cdots & a_{m1}b_{pq} & \cdots & \cdots & a_{mn}b_{p1} & a_{mn}b_{p2} & \cdots & a_{mn}b_{pq} \end{pmatrix}$$

Maintenant, montrons que :

$$[X^s|X^t] = \langle y^s|y^t\rangle_L$$

où $L = C \otimes W(ProduitdeKronecker)$.

Sachant que, $W = \frac{1}{n}I_n$ et $C = \frac{1}{p}I_p$ on en déduit que $L = \frac{1}{np}I_{np}$.

D'une part, soit $(s,t) \in [|1,T|]$:

$$
\begin{aligned}
[X_s \mid X_t] &= tr(\tilde{X_s}' \tilde{X_t}) \\
&= \sum_{k=1}^{n} \sum_{i=1}^{p} w_k c_i x_{s,k}^i x_{t,k}^i \\
&= \sum_{k=1}^{n} \sum_{i=1}^{p} \frac{1}{np} x_{s,k}^i x_{t,k}^i
\end{aligned}
$$

D'autre part:

$$
\begin{aligned}
\langle y^s | y^t \rangle_L &= y^{s\prime} L y^t \\
&= \sum_{i=1}^{p} \frac{1}{np} x_s^{i\prime} x_t^i \\
&= \sum_{i=1}^{p} \frac{1}{np} \sum_{k=1}^{n} x_{s,k}^i x_{t,k}^i \\
&= \sum_{k=1}^{n} \sum_{i=1}^{p} \frac{1}{np} x_{s,k}^i x_{t,k}^i
\end{aligned}
$$

Donc $[X^s | X^t] = \langle y^s | y^t \rangle_L$.

Il en découle que X^t est "verticalisé" en $y^{t*} = y^t$ L-normé.

Par suite, on note $Y = [y^1, \ldots, y^T]$. Un individu correspond à une ligne de ce tableau, correspondant donc à l'indice (j, i) dans le cube initial. Un individu est un vecteur de taille T, dont les coordonnées sont les T valeurs de la case (i, j) du tableau X_t sur la période. C'est la trajectoire de cette case. Son poids est $L_(j, i) = w_i c_j = \frac{1}{np}$.

Par exemple, pour un tableau cubique ventilant la valeur ajoutée d'une économie par région (i), secteur (j) et année (t), l'individu du tableau dépliant ainsi le cube serait la trajectoire dans le temps de la V.A. réalisée par le secteur j dans la région i.

Et on a :

$$
\left[\left| \sum_{t=1}^{T} u_t \frac{X_t}{[|X_t|]} \right| \right]^2 = \|Y^* u\|_W^2 = \|F\|_W^2
$$

, où $Y^* = [y^{1*}, \ldots, y^{T*}]$ et $F = Y^* u$ est la composante associée à u.

On remarque que:

Métrique usuelle : Si $M = I_T$ alors $\langle y^{s*}, y^{t*} \rangle = \langle y^s, y^t \rangle_L$.

Métrique réduite : Diviser les variables y^t par $\sigma_t = \sqrt{\|y^t\|}$ est équivalent à prendre $m_t = \frac{1}{{\sigma_t}^2}$. On a $M_{\frac{1}{\sigma^2}} = M_{\frac{1}{\sigma}} M_{\frac{1}{\sigma}}$ et donc $\langle y^s m_{\frac{1}{\sigma_s}}, y^t m_{\frac{1}{\sigma_t}} \rangle_L = m_{\frac{1}{\sigma_s}} y^{s\prime} L y^t m_{\frac{1}{\sigma}\,t}$.

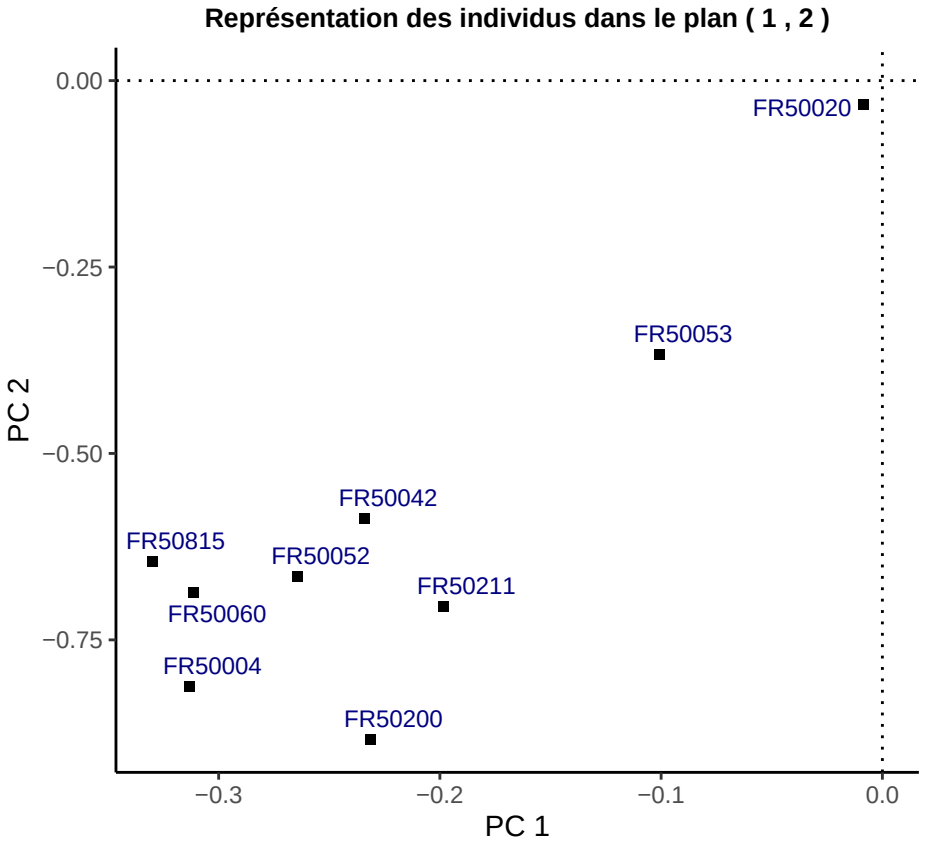
Travailler avec la métrique $M_{\frac{1}{\sigma}^2}$ revient à utiliser la métrique L sur des variables $M_{\frac{1}{\sigma}^2}$-normées.

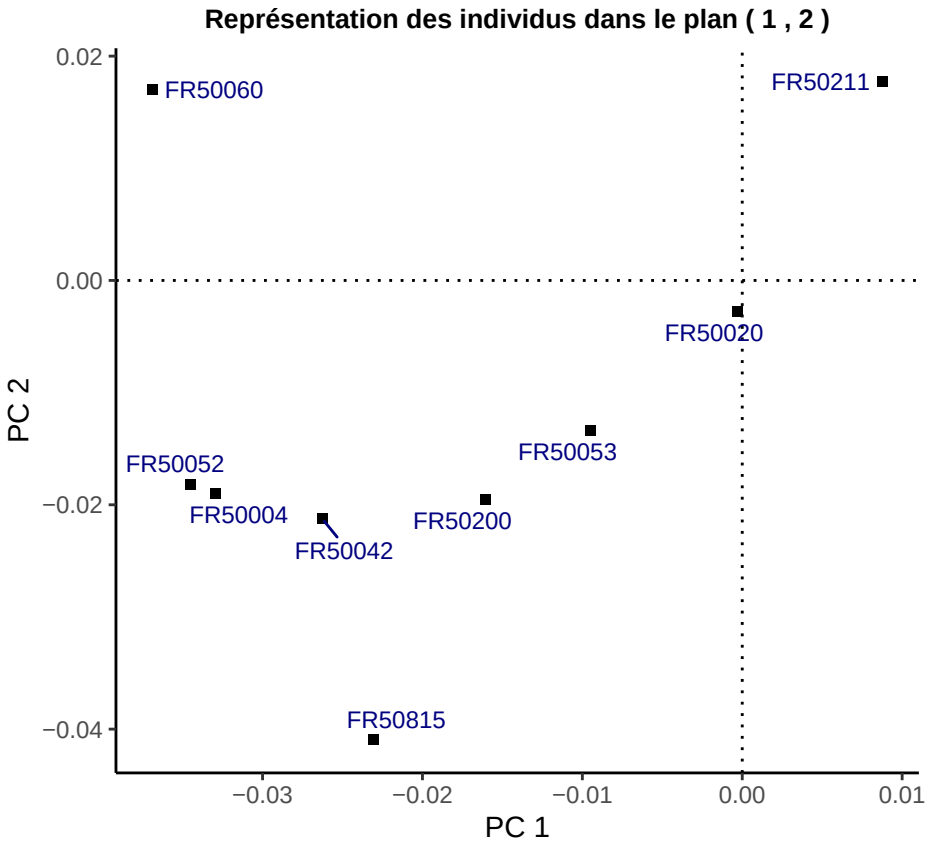
b) Pour déduire que les composantes principales $F^1, \ldots, F^k, \ldots$ sont orthogonales au sens du produit scalaire convenable, nous devons considérer la façon dont ces composantes sont construites à partir des vecteurs propres obtenus.

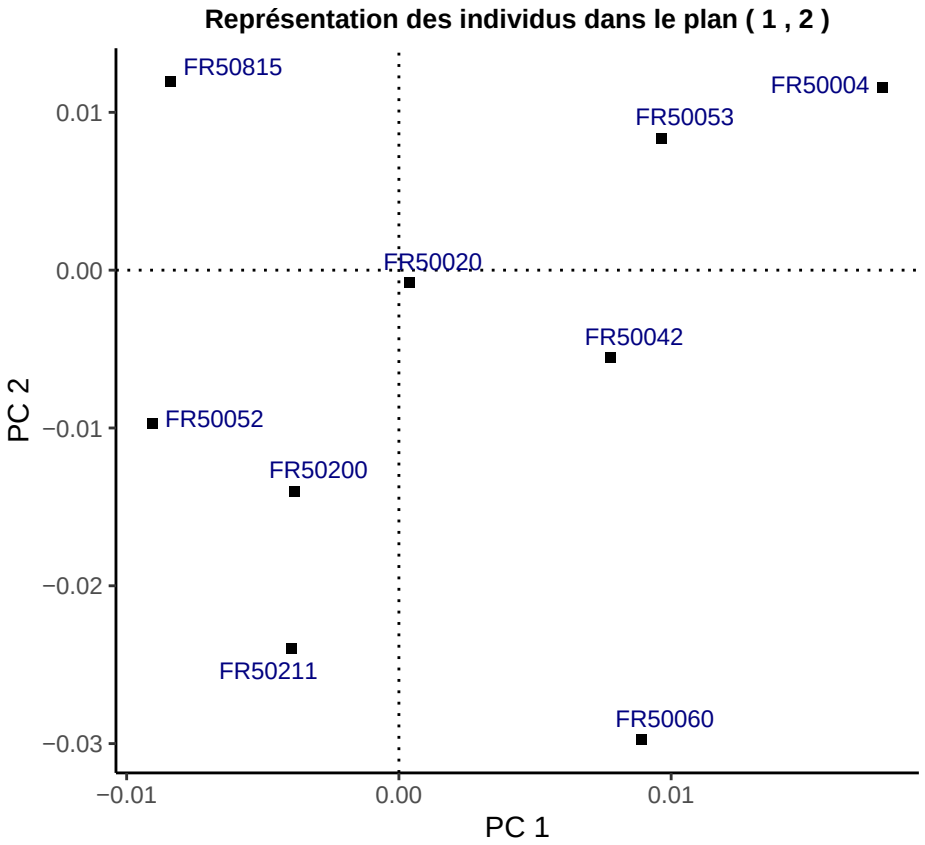
$$
\begin{aligned}
\langle F_k | F_l \rangle_L &= F_k' L F_l \\
&= u_k' Y^{*\prime} L Y^* u_l \\
&= u_k' M_{\frac{1}{\sigma}} Y' L Y M_{\frac{1}{\sigma}} u_l \\
&= 1 \quad \text{si} \quad k = l \\
&= 0 \quad \text{si} \quad k \neq l
\end{aligned}
$$

c) La fonction calculant les composantes, la représentation des individus et celle des tableaux setrouvent en annexe. Nous verrons ci-dessous un exemple d'application de ces fonctions sur les données simulated.

```
# Afficher les graphiques pour les composantes principales de 1 à
for (i in 1:3) {
  Ft <- cmp(filtre,i)$F_k
  representation_graph_ind(Ft, 1, 2, aff_noms = TRUE,cex_titre = 0.80, repel = T)
}
```

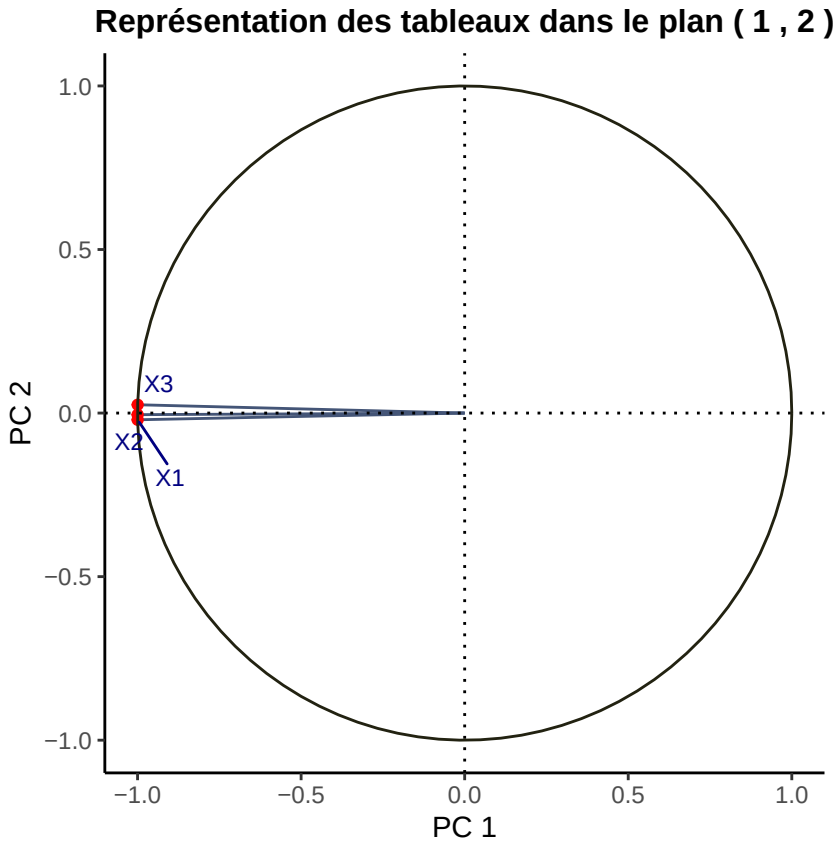



Représentation des individus dans le plan (1 , 2)
0.02
0.00
−0.02
−0.04
PC 2
FR50060
FR50211
FR50020
FR50053
FR50052
FR50004
FR50042
FR50200
FR50815
−0.03
−0.02
−0.01
0.00
0.01
PC 1



d) Cosinus entre les tableaux et les composantes principales : La fonction calculant le cosinus entre le tableau X_t et la composante F_k se trouve en annexe.

```
# Exemple d'utilisation avec les tableaux n_tableaux et les composantes principales 3 et 4
representation_graph_tab(filtre, 1, 2, aff_noms = T, repel = TRUE)
```



1.2.3 Illustration d'un ACP d'un tableau juxtaposé dépliant le tableau cubique

Soit A, B et C les matrices données comme suit:

$$\mathrm{A} = \begin{pmatrix} 1 & 0 \\ -1 & 1 \\ 2 & 2 \end{pmatrix}, \quad \mathrm{B} = \begin{pmatrix} 2 & -1 \\ -1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \mathrm{C} = \begin{pmatrix} 3 & 1 \\ 1 & -2 \\ 0 & -3 \end{pmatrix}$$

En utilisant le produit scalaire défini par:

$$\begin{aligned} [A \mid B] &= \mathrm{tr}(CA'WB) \\ &= \mathrm{tr}(C^{\frac{1}{2}}A'W^{\frac{1}{2}}W^{\frac{1}{2}}BC^{\frac{1}{2}}) \\ &= \mathrm{tr}((W^{\frac{1}{2}}AC^{\frac{1}{2}})'W^{\frac{1}{2}}BC^{\frac{1}{2}}) \\ &= \mathrm{tr}(\tilde{A}'\tilde{B}) \end{aligned}$$

avec $W := \frac{1}{n}I_n$ la matrice de poids des individus et $C := \frac{1}{p}I_p$ la matrice de poids des variables.

Nous allons calculer la matrice des coefficients RV d'Escoufier que l'on notera Γ. Ci-dessous la formule permettant de calculer le coefficient RV entre deux matrices A et B de taille n, p ici $n = 3$ et $p = 2$:

$$R(A, B) = \frac{[A \mid B]}{[|A|]\,[|B|]}$$

Calculons les coefficients $R(A, B)$, $R(A, C)$ et $R(B, C)$.

Procédons étape par étape:

$$[A \mid B] = \mathrm{tr}(CA^tWB) = \frac{1}{6}\mathrm{tr}(A^tB)$$

Or,

$$\frac{1}{6}\begin{pmatrix}1 & -1 & 2\\ 0 & 1 & 2\end{pmatrix}\begin{pmatrix}2 & -1\\ -1 & 0\\ 0 & 1\end{pmatrix} = \frac{1}{6}\begin{pmatrix}3 & 1\\ -1 & 2\end{pmatrix}$$

donc $[A \mid B] = \frac{5}{6}$

Pour $[A \mid C]$:

$$[A \mid C] = \frac{1}{6}\mathrm{tr}(A^TC)$$

Or,

$$\frac{1}{6}\begin{pmatrix}1 & -1 & 2\\ 0 & 1 & 2\end{pmatrix}\begin{pmatrix}3 & 1\\ 1 & -2\\ 0 & -3\end{pmatrix} = \frac{1}{6}\begin{pmatrix}2 & -3\\ 1 & -8\end{pmatrix}$$

donc $[A \mid C] = \frac{-6}{6} = -1$

Pour $[B \mid C]$:

$$[B \mid C] = \frac{1}{6}\mathrm{tr}(B^TC)$$

Or,

$$\frac{1}{6}\begin{pmatrix}2 & -1 & 0\\ -1 & 0 & 1\end{pmatrix}\begin{pmatrix}3 & 1\\ 1 & -2\\ 0 & -3\end{pmatrix} = \frac{1}{6}\begin{pmatrix}5 & 4\\ -3 & -4\end{pmatrix}$$

donc $[B \mid C] = \frac{1}{6}$. En utilisant la norme associée à ce produit scalaire on a:

$[|A|] = \sqrt{\frac{11}{6}}$, $[|B|] = \sqrt{\frac{7}{6}}$ et $[|C|] = \sqrt{\frac{24}{6}} = 2$.

D'où

$$R(A,B) = \frac{\frac{5}{6}}{\sqrt{\frac{11}{6}}\sqrt{\frac{7}{6}}}$$

$$R(A,C) = \frac{-1}{2\sqrt{\frac{11}{6}}}$$

et

$$R(B,C) = \frac{\frac{1}{6}}{2\sqrt{\frac{7}{6}}}$$

Par symétrie du produit scalaire, la matrice Γ des coefficients RV est donc égale à:

$$\Gamma = \begin{pmatrix}1.0000 & 0.5698 & -0.3692\\ 0.5698 & 1.0000 & 0.0771\\ -0.3692 & 0.0771 & 1.0000\end{pmatrix}$$

Maintenant, considérons le dépliement des matrices juxtaposées :

$$\mathrm{X} = \begin{pmatrix} 1 & 2 & 3 \\ -1 & -1 & 1 \\ 2 & 0 & 0 \\ 0 & -1 & 1 \\ 1 & 0 & -2 \\ 2 & 1 & -3 \end{pmatrix}$$

où la première variable est la matrice A dépliée, la deuxième variable est la matrice B dépliée et la troisième variable est la matrice C dépliée.

On pose M la matrice diagonale p, p $(ici \quad p = 3, \quad n = 6)$ dont le coefficient $M_{ii} = \frac{1}{||X_i||_W}$ avec:

- $W := \frac{1}{n} I_n$
- X_i la ième variable
- $\|X_i\|_W = \sqrt{(X_i^t W X_i)}$

Ici M sera donc égale à:

$$M = \begin{pmatrix} \frac{1}{\sqrt{\frac{11}{6}}} & 0 & 0 \\ 0 & \frac{1}{\sqrt{\frac{7}{6}}} & 0 \\ 0 & 0 & \frac{1}{2} \end{pmatrix}$$

On a alors:

$$\Delta := MX'WXM = X*'WX* = \Gamma$$

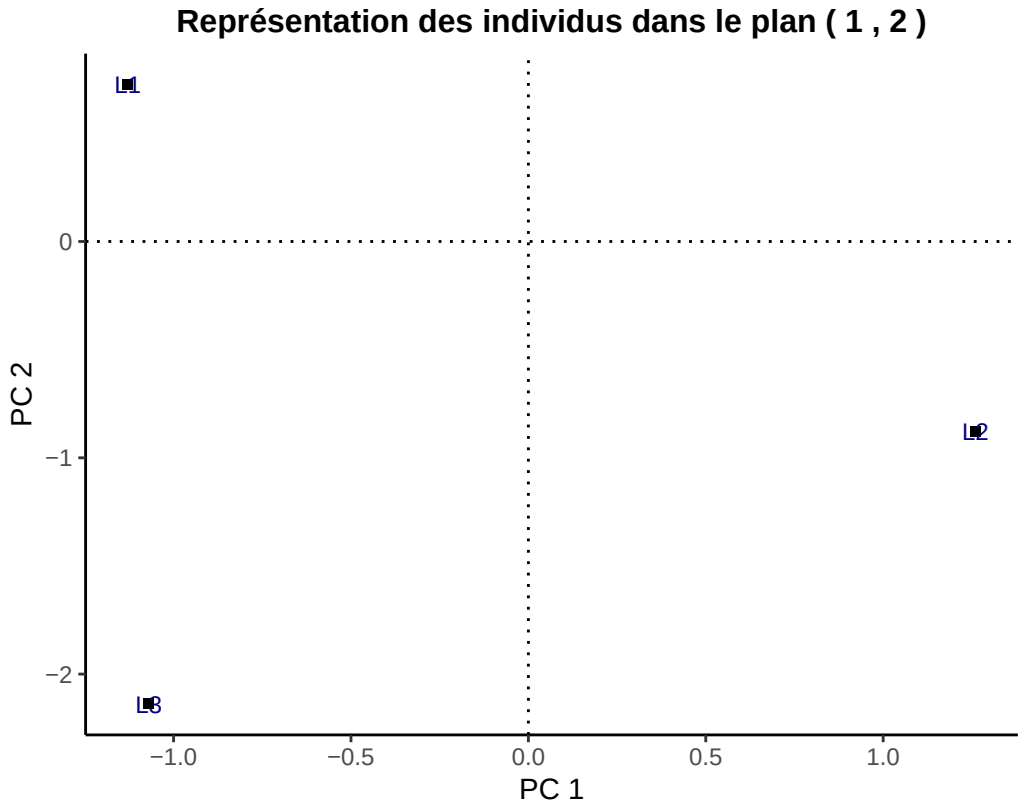
où, $X*$ est la matrice dont les variable sont normée.

La fonction ci-dessous fait les calculs matriciels puis nous donne la matrice obtenue delta qui est égale à la matrice des coefficients RV calculée précédemment.

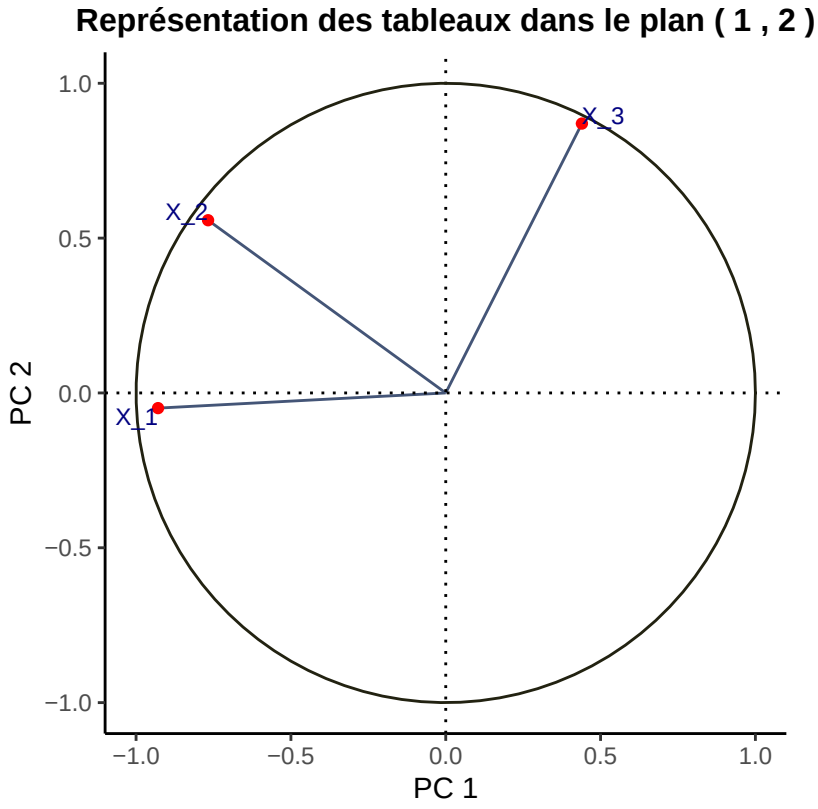
```
# Données
X <- matrix(
  c( 1, 2, 3,
    -1,-1, 1,
     2, 0, 0,
     0,-1, 1,
     1, 0,-2,
     2, 1,-3
  ), nrow = 6, byrow = TRUE)

L <- list()
for (i in 1:3) {
  L[[i]] <- matrix(X[,i], nrow = 3, ncol = 2)
  # Ajout des noms de lignes
  rownames(L[[i]]) <- paste0("L", 1:3)
}
```

Finalement, on en déduit qu'il y a équivalence entre les matrices juxtaposées A, B et C avec la matrice X dont les variables sont les matrices A, B et C dépliées.

```
Ft <- cmp(L,1)$F_k
representation_graph_ind(Ft, 1, 2, aff_noms = TRUE)
```




```
representation_graph_tab(L, 1, 2)
```



1.2.4 Quelles ACP d'autres "dépliages" du tableau cubique ?

a) Disons que l'on dispose d'un tableau cubique $X(i, j, t)$. On peut le déplier de trois manières différentes : $Y((i, j), t)$ de façon à avoir les individus et caractéristiques en fonction du temps. Pour avoir les caractéristiques dans le temps pour chaque individu $Y((j, t), i)$. Ou pour avoir les individus dans le temps pour chaque caractéristique $Y((i, t), j)$.
Chacun de ces dépliages donnera une matrice différente et donc une ACP différente. Mais chacun reste intéressant.

b) Selon le dépliage choisi, les individus et variables ne seront pas les mêmes. En individus on peut avoir :

- Les individus par caractéristiques (ou caractéristiques par individus) décrit dans le temps,
- Les individus dans le temps (à un instant donné), décrit par les caractéristiques,
- Les caractéristiques dans le temps (à un instant donné), décrivant les individus.
  Les ACP à envisager dépendent du jeu de données que l'on souhaite étudier. Ici les individus et variables ne sont pas clairement définit car ne porte que des numéros. Ce qui est sûr c'est que les individus regrouperont les tableaux dépliés. Les éléments à projeter en supplémentaire dépendent, encore une fois, de ce que l'on souhaite étudier.

c) Un inconvénient est qu'on perde une grosse partie de l'information qui sera noyée dans le tableau déplié. Cependant, cela permet de mieux visualiser les données et de les traiter plus facilement (utilisation du produit matriciel possible). Le plus gros inconvénient reste la perte de la symétrie, par exemple lors d'un dépliage de $X(i, j, t)$ en $Y((i, j), t)$. En effet, les individus et variables ne sont plus équivalents. Cela peut poser problème si l'on souhaite comparer les individus et variables entre eux.

L'idéal serait de réaliser les trois dépliages possibles et de les comparer pour voir lequel est le plus pertinent ! Tout dépend du tableau initial.

2 Situation 2

2.1 Les données traitées

Les données qui illustreront ce premier emploi de la méthode STATIS sont publiées par Friday. Ces données correspondent à des observations faunistiques effectuées dans plusieurs étangs, où différents taxons ont été identifiés.

Définition des taxons:

Un taxon est une unité taxonomique utilisée pour classer les organismes vivants. Il peut représenter un groupe d'espèces, de genres, de familles, etc. Voici quelques exemples de taxons courants :

- **Ordre** : une catégorie de rang élevé dans la classification biologique, regroupant des familles apparentées. Par exemple, les libellules et les cigales appartiennent à l'ordre des Hemiptera.
- **Famille** : un niveau de classification plus spécifique que l'ordre, regroupant des genres similaires. Par exemple, les coléoptères, ou scarabées, appartiennent à la famille des Coleoptera.
- **Genre** : une catégorie encore plus spécifique, regroupant des espèces étroitement liées. Par exemple, le genre *Apis* regroupe les abeilles.
- **Espèce** : la plus petite unité taxonomique, représentant un groupe d'organismes qui peuvent se reproduire entre eux et produire une descendance fertile. Par exemple, l'abeille domestique (*Apis mellifera*) est une espèce d'abeille.

Répartition des taxons par groupe:

- **Groupe 1 (Hemiptera)** : 11 taxons (ordre d'insectes comprenant des espèces telles que les punaises et les cigales)
- **Groupe 2 (Odonata)** : 7 taxons (ordre d'insectes comprenant les libellules et les demoiselles)
- **Groupe 3 (Trichoptera)** : 13 taxons (ordre d'insectes comprenant les trichoptères, ou phryganes, souvent appelés "mouches à caddis")
- **Groupe 4 (Ephemeroptera)** : 4 taxons (ordre d'insectes comprenant les éphémères)
- **Groupe 5 (Coleoptera)** : 13 taxons (ordre d'insectes comprenant les coléoptères, ou scarabées)
- **Groupe 6 (Diptera)** : 22 taxons (ordre d'insectes comprenant les diptères, ou mouches)
- **Groupe 7 (Hydracarina)** : 4 taxons (sous-classe d'acariens aquatiques, également connue sous le nom de "tiques d'eau" ou "tiques de mare")
- **Groupe 8 (Malacostraca)** : 3 taxons (classe d'arthropodes comprenant des crustacés comme les crabes, les crevettes et les homards)
- **Groupe 9 (Mollusca)** : 8 taxons (embranchement d'animaux incluant les mollusques, comme les escargots, les moules et les calmars)
- **Groupe 10 (Oligochaeta)** : 6 taxons (classe de vers annélides comprenant les oligochètes, comme les vers de terre)

Identification des tableaux:

Chaque tableau de données est identifié par le groupe de taxons qu'il contient. Par exemple, le tableau 1 contient des données sur le groupe de taxons Hemiptera, le tableau 2 sur le groupe de taxons Odonata, et ainsi de suite jusqu'au tableau 10 pour le groupe de taxons Oligochaeta.

Identification des étangs:

Les 16 étangs sont étiquetés par les lettres Q, P, R, J, E, C, D, K, B, A, G, M, L, F, H, N.

Le groupe t représente l'indexation des différents tableaux de données, allant de 1 à 10 dans cet exemple. Chaque tableau t correspond à un groupe taxonomique spécifique.
Les observations (étangs) sont représentées par l'index i, allant de 1 à 16.
Les variables (taxons) sont représentées par l'index j, dont le nombre de variables peut varier d'un groupe taxonomique à l'autre. Les valeurs de j vont de 1 à p_t, où p_t est le nombre de variables dans le groupe taxonomique t.

2.2 De nouvelles matrices dans un nouvel espace

1.1.a) On souhaite exprimer la matrice P_m en fonction des matrices X_m et M_m. La formule est donnée par :

$$P_m = X_m M_m X'_m$$

où X_m est une matrice des données de dimension (n, p_m) , M_m est une matrice de pondération de dimension (p_m, p_m), et X'_m est la transposée de X_m.

b) L'espace P est l'ensemble des matrices $n \times n$. Cet espace est noté :

$$P = \mathbb{R}^{n \times n}$$

La dimension de cet espace est n^2, car une matrice $n \times n$ possède n^2 éléments indépendants. 1.2.a) Les lignes et les colonnes de P_m représentent les individus. Le poids naturel des lignes et des colonnes est donc le même, noté W de dimension (n, n).

b) On peut munir **P** du produit scalaire de Frobénius qui est défini par:

$$(P_m, P_k) = tr(P_m W P_k W) \quad (p)$$

c) Les matrices X_m peuvent avoir des dimensions différentes, et les matrices M_m peuvent varier. Cela signifie que les individus des tableaux X_m résident dans des espaces euclidiens différents selon le tableau. Leurs produits scalaires ne sont donc pas directement comparables d'un tableau à l'autre. Pour rendre les matrices P_m comparables, il est nécessaire de les normer.

d) Une fois les matrices P_m normées, elles se situent toutes sur la sphère unité de P. La proximité entre deux matrices peut être mesurée par le cosinus de l'angle entre elles, noté RV. Si le coefficient est proche de 1 on pourra dire que la matrice X_m et X_k sont similaire (correlées fortement) et donc une similarité plus grande entre les ensembles d'individus représentés par X_m et X_k. En revanche, si le coefficient RV est proche de -1 on dira que les matrices X_m et X_k sont anticorrelées. Enfin, si le coefficient RV est proche de 0, on dira que X_m et X_k sont décorrelées.

2.3 STATIS 2

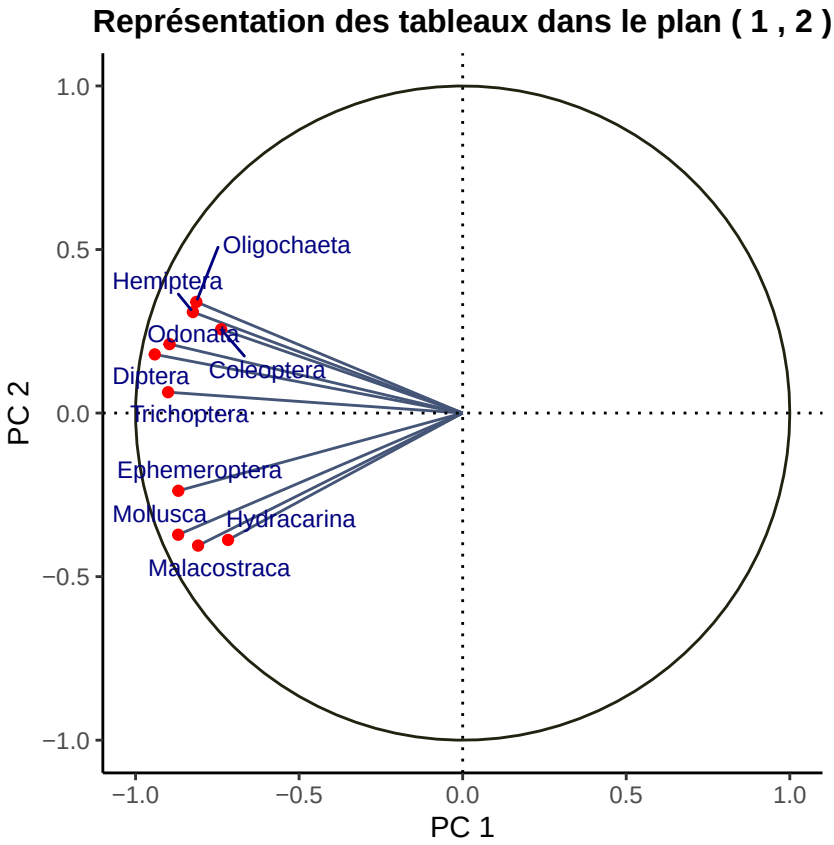
On applique le programme de STATIS 1 à un ensemble de matrices P_m issues de tableaux thématiques décrivant les mêmes individus. On note F_1,..., F_k les composantes principales obtenues.

2.3.1 Graphiques directs de STATIS

a) Pour projeter chaque P_m sur un graphe dont les directions sont un couple de composantes principales (F_k, F_l), on utilise leurs cosinus (RV) avec ces composantes.

b)

```
representation_graph_tab(lst, 1, 2,aff_noms = TRUE, repel = TRUE)
```



2.3.2 Graphiques projetant les individus et les variables initiales

a) Pour montrer que les composantes principales sont des matrices symétriques, nous devons rappeler que les matrices de produits scalaires P_m sont symétriques. Les composantes principales sont calculées à partir de ces matrices de produits scalaires, ce qui garantit que les composantes principales seront symétriques. En effet, Les composantes principales F_k sont des combinaisons linéaires des matrices P_m qui sont elles-mêmes symétriques.

b) En interprétant chaque composante principale comme une matrice de produits scalaires entre individus, on peut obtenir une image du nuage des individus dans un plan de dimension réduite. Pour cela, on diagonalise chaque matrice F_k :

$$F_k = U_k \Lambda_k U_k' = (U_k \Lambda_k^{1/2})(U_k \Lambda_k^{1/2})'$$

Les coordonnées des individus sur les axes principaux sont données par $U_k \Lambda_k^{1/2}$, fournissant une représentation euclidienne hiérarchisée.

c) On applique cette méthode de représentation au tableau de données d'application pour obtenir une visualisation des individus.

2.3.3 Aides à l'interprétation

a) La formule pour calculer le cosinus carré $\cos^2(P_m, F_k)$ entre P_m et F_k est :

$$\cos^2(P_m, F_k) = \frac{[P_m \mid F_k]^2}{[|P_m|]^2 \, [|F_k|]^2}$$

b) On en déduit le calcul du $QLT_k(P_m)$, qui est la somme des cosinus carrés jusqu'à la composante k :

$$QLT_k(P_m) = \cos^2(P_m, \langle F_1, \ldots, F_k \rangle) = \sum_{j=1}^{k} \cos^2(P_m, F_j)$$

c) Le calcul de la contribution $CTR_k(P_m)$ est donné par :

$$CTR_k(P_m) = \frac{\cos^2(P_m, F_k)}{\lambda_k}$$

d) Programmation et application.

Il est nécessaire de programmer ces fonctions et de les appliquer au tableau de données d'application pour obtenir des mesures quantitatives de la proximité et de la contribution des matrices P_m aux composantes principales F_k.

```
CTR_k <- function(Pm, Fk, lambda_k) {
  c <- cos_square(Pm,Fk)/lambda_k
  return(c)
}

# Appliquer les fonctions au tableau de données d'application
# Supposons que P_m représente les données projetées sur une composante principale
set.seed(123)
# Exemple de données projetées (100 individus, 2 dimensions)
P_m <- matrix(runif(100), ncol=10)
# Exemple de 5 composantes principales
Fs <- replicate(5, matrix(runif(100), ncol=10), simplify = FALSE)
# Exemple de valeurs propres
lambda <- c(6.47, 3.98, 1.42, 0.88, 0.08)
for (k in 1:5) {
  l <- lambda[k]
  cat("CTR_", k, "(P_m): ", CTR_k(P_m, Fs[[k]],lambda_k = l), "\n")
}
```

```
## CTR_ 1 (P_m):  0.08801209
## CTR_ 2 (P_m):  0.132837
## CTR_ 3 (P_m):  0.372453
## CTR_ 4 (P_m):  0.5777055
## CTR_ 5 (P_m):  6.727484
```

3 Annexe

3.1 Situation 1

3.1.1 Programme des fonctions prd_scalaire et norme

La fonction calculant le produit scalaire de Frobénius:

```
prd_scalaire = function(A,B){
  # les dimension des matrices
  na = length(A[,1]); nb = length(B[,1])
  pa = length(A[1,]); pb = length(B[1,])

  # Les matrices de ponderation
  Wa = (1/na)*diag(na); Ca = (1/pa)*diag(pa)
```

```
  Wb = (1/nb)*diag(na); Cb = (1/pb)*diag(pb)

  # Calcul des matrices A_tild et B_tild
  A_tild = sqrt(Wa) %*% A %*% sqrt(Ca)
  B_tild = sqrt(Wb) %*% B %*% sqrt(Cb)

  # Produit scalaire
  ps = sum(diag(t(A_tild)%*%B_tild))

  return(ps)
 }
```

La fonction calculant la norme d'une matrice A associée au produit scalaire de Frobénius

```
norme = function(A){
  return(sqrt(prd_scalaire(A,A)))
}
```

3.1.2 Coefficient RV

La fonction calculant le coefficient RV d'Escoufier:

```
coef_RV <- function(Xi,Xj){
      prd_sclr = prd_scalaire(Xi, Xj)
      norm_i = norme(Xi)
      norm_j = norme(Xj)
      rv = prd_sclr / (norm_j * norm_i)

      return(rv)
}

matcoef_RV = function(T_tableaux) {
  t = length(T_tableaux)
  mat_rv = matrix(rep(NA, t * t), nrow = t, ncol = t)

  for (i in seq_along(T_tableaux)) {
    for (j in seq_along(T_tableaux)) {
      # Stocker le résultat dans une matrice
      mat_rv[i, j] = coef_RV(T_tableaux[[i]],T_tableaux[[j]])
    }
  }

  return(mat_rv)
}
```

Fonction donnant les vecteurs u solutions et les valeurs propres associées:

```
vecval_prop = function(T_tableau) {
  matrice = matcoef_RV(T_tableau)# on calcule la matrice contenant les coefs d'Escoufier

  resultat_propre = eigen(matrice)# list ayant les valeurs et vecteurs propres
  val_prop = resultat_propre$values #valeurs propres
  vec_prop = resultat_propre$vectors #vecteurs propres associées

  return(list(val_prop = val_prop, vec_prop = vec_prop))
}
```

3.1.3 Dépliage du tableau cubique en un tableau juxtaposé et Composantes principales:

```
oper_inert = function(X){

  p = length(X[1,]) # la dimension des individus
  n = length(X[,1]) # la dimension des variables
  W = diag(n)/n
  M = matrix(0, nrow = p, ncol = p)

  for (i in 1:p) {
    M[i,i] = 1/sqrt(t(X[,i])%*%W%*%X[,i])
  }
  mat_d = M %*% t(X) %*% W %*% X %*% M
  return(mat_d)
}
```

3.1.4 La représentation des individus et des tableaux

Représentation des individus:

```
representation_graph_ind <- function(F_t, k, l, aff_noms = FALSE, col = "navy", cex_titre = 1, repel = F
  if (l <= k) {
    print("Le 1er axe doit être strictement plus petit que le 2nd")
    return(NULL)
  }

  # Créer un data frame avec les composantes principales
  df <- data.frame(PC_k = F_t[, k], PC_l = F_t[, l])
  rownames(df) <- rownames(F_t)

  # Trouver les limites des axes x et y pour centrer l'origine
  x_limits <- range(df$PC_k)
  y_limits <- range(df$PC_l)
  limits <- range(c(-x_limits, x_limits, -y_limits, y_limits))

  # Plot les composantes principales dans un plan avec l'origine au centre
  plot(df$PC_k, df$PC_l, xlab = paste("PC", k), ylab = paste("PC", l), xlim = limits,
       ylim = limits, pch = 22, cex = 0.75, col = "black", bg = "#222211")

  # Ajouter des lignes pour les axes horizontal et vertical se coupant à l'origine
  abline(h = 0, col = "black", lty = "dotted", lwd = 0.5)
  abline(v = 0, col = "black", lty = "dotted", lwd = 0.5)

  if (aff_noms == TRUE) {
    if (repel) {
      # Calculer les positions pour les étiquettes en repoussant légèrement les points
      offset <- 0.05 * max(diff(range(df$PC_k)), diff(range(df$PC_l)))
      text_positions <- data.frame(
        x = df$PC_k + ifelse(df$PC_k >= 0, offset, -offset),
        y = df$PC_l + ifelse(df$PC_l >= 0, offset, -offset)
      )

      # Ajuster les positions pour que les étiquettes ne sortent pas des limites du graphique
      text_positions$x <- pmax(pmin(text_positions$x, max(limits) - offset), min(limits) + offset)
```

```
      text_positions$y <- pmax(pmin(text_positions$y, max(limits) - offset), min(limits) + offset)

      for (i in 1:nrow(df)) {
        text(text_positions$x[i], text_positions$y[i], labels = rownames(df)[i], cex = 0.85, col = col)
      }
    } else {
      text(df$PC_k, df$PC_l, labels = rownames(df), cex = 0.85, col = col, pos = ifelse(df$PC_k >= 0, 4
    }
  }

  # Réglage de la taille du titre
  title(main = paste("Représentation des individus dans le plan (", k, ",", l, ")"), cex.main = cex_titi
}
```

```
representation_graph_ind <- function(F_t, k, l, aff_noms = FALSE, col = "navy",
                                     cex_titre = 1, repel = FALSE) {

  if (l <= k) {
    print("Le 1er axe doit être strictement plus petit que le 2nd")
    return(NULL)
  }

  # Créer un data frame avec les composantes principales
  df <- data.frame(PC_k = F_t[, k], PC_l = F_t[, l])
  rownames(df) <- rownames(F_t)

  # Plot les composantes principales dans un plan avec l'origine au centre
  p <- ggplot(df, aes(x = PC_k, y = PC_l)) +
    geom_point(shape = 15, color = "black", size = 1.5) + # Utilisation de petits carrés noirs
    theme_classic() +
    labs(x = paste("PC", k), y = paste("PC", l)) +
    ggtitle(paste("Représentation des individus dans le plan (", k, ",", l, ")")) +
    theme(plot.title = element_text(size = cex_titre * 12, face = "bold", hjust = 0.5))

  # Ajouter des lignes pour les axes horizontal et vertical se coupant à l'origine
  p <- p + geom_hline(yintercept = 0, linetype = "dotted", lwd = 0.5) +
    geom_vline(xintercept = 0, linetype = "dotted", lwd = 0.5)

  if (aff_noms == TRUE) {
    if (repel) {
      # Ajouter les étiquettes avec repulsion
      p <- p + geom_text_repel(aes(label = rownames(df)), size = 3, color = col)
    } else {
      # Ajouter les étiquettes sans repulsion
      p <- p + geom_text(aes(label = rownames(df)), size = 3, color = col)
    }
  }

  print(p)
}
```

Cosinus entre les tableaux et les composantes principales :

```r
# Calcul des cosinus entre Xt (resultats_n) et chaque composante principale F_k
cosinus = function(X_t, F_k){
  return(coef_RV(X_t,F_k))
}
```

Représentation des tableaux:

```r
representation_graph_tab <- function(T_tab, k, l, aff_noms = TRUE, col = "navy", cex_titre = 1, repel =
  if (l <= k) {
    print("Le 1er axe doit être strictement plus petit que le 2nd")
    return(NULL)
  }

  t <- length(T_tab)
  mat_cos <- matrix(NA, nrow = t, ncol = t)
  for (i in 1:t) {
    for (j in 1:t) {
      mat_cos[i, j] <- cosinus(T_tab[[i]], cmp(T_tab, j)$F_k)
    }
  }

  # Récupérer les noms des tableaux ou les nommer par défaut
  row_names <- if (is.null(names(T_tab))) paste("X", 1:t, sep = "_") else names(T_tab)
  mat_cos <- as.data.frame(mat_cos)
  rownames(mat_cos) <- row_names

  F_k <- mat_cos[, k]  # Récupération de la kième composante principale
  F_l <- mat_cos[, l]  # Récupération de la lième composante principale

  # Création des données pour le cercle unité
  df <- data.frame(
    varabs = cos(seq(0, 2 * pi, length.out = 100)),
    varord = sin(seq(0, 2 * pi, length.out = 100))
  )

  p <- ggplot(mat_cos, aes(x = F_k, y = F_l)) +
    geom_segment(aes(xend = 0, yend = 0), color = "#445577") +
    geom_point(color = "red", size = 1.5) +
    theme_classic() +
    coord_fixed(ratio = 1) +
    geom_hline(yintercept = 0, linetype = "dotted", lwd = 0.5) +
    geom_vline(xintercept = 0, linetype = "dotted", lwd = 0.5) +
    labs(x = paste("PC", k),
         y = paste("PC", l)) +
    ggtitle(paste("Représentation des tableaux dans le plan (", k, ",", l, ")")) +
    geom_path(data = df, aes(varabs, varord), color = "#222211", linewidth = 0.5) +  # Ajout du cercle
    xlim(-1, 1) +
    ylim(-1, 1) +
    theme(plot.title = element_text(size = cex_titre * 12, face = "bold", hjust = 0.5))

  if (aff_noms) {
    if (repel) {
      p <- p + geom_text_repel(aes(label = rownames(mat_cos)), color = col, size = 3)
    } else {
```

```
      p <- p + geom_text(aes(label = rownames(mat_cos)), color = col, size = 3,
                         hjust = ifelse(F_k >= 0, 0, 1), vjust = ifelse(F_l >= 0, 0, 1))
    }
  }

  print(p)
}
```

3.2 Situation 2

3.2.1 Les données partitionné en thèmes

```
# Tableau à trois dimension
data(chickenk)

M = as.matrix(chickenk$Mortality)
rownames(M) = seq(1,351,1)

FS = as.matrix(chickenk$FarmStructure)
rownames(FS) = seq(1,351,1)

OFH = as.matrix(chickenk$OnFarmHistory)
rownames(OFH) = seq(1,351,1)

FC = as.matrix(chickenk$FlockCharacteristics)
rownames(FC) = seq(1,351,1)

CTS = as.matrix(chickenk$CatchingTranspSlaught)
rownames(CTS) = seq(1,351,1)

data = list(M, FS, OFH, FC, CTS)
```

```
data("friday87")
Tab_friday <- function() {

  t_tab <- friday87$fau
  v <- c(0,11, 18, 31, 35, 48, 70, 74, 77, 85, 91)
  noms <- friday87$tab.names
  n_result <- list()

  for (i in 2:11) {
    nom_tableau <- noms[i-1]  # Nom du tableau
    tableau <- as.matrix.data.frame(t_tab[, (v[i-1]+1):v[i]])  # Extraction des colonnes
    n_result[[nom_tableau]] <- tableau  # Stocker le tableau nommé dans n_result
  }

  return(n_result)
}
```

```
# Créer une liste vide
lst <- list()

# Boucle sur les éléments de Tab_friday
for (i in 1:length(Tab_friday())) {
```

```
  nom <- friday87$tab.names  # Créer un nom unique pour chaque élément de la liste
  tableau <- mat_coldiff(Tab_friday()[[i]])  # Calculer le tableau
  lst[[nom[i]]] <- tableau  # Ajouter le tableau à la liste
}
```

3.2.2 Aides à l'interprétation