Statistiques : TP2

Université de Tours

`Département informatique de Blois`

Analyse en composantes principales et apprentissage

*
* *

Contents

1 Rappels

1.1 Principe général de l'ACP

Considérons un nuage $\mathcal{N}$ de n points dans un espace E de dimension p. Lorsque E est de dimension élevée, on ne peut pas visualiser l'espace de points. Un des buts de l'**analyse en composantes principales** (ACP) est alors de trouver le meilleur sous-espace H de E, de dimension h égale à 2 ou 3 par exemple, dans lequel on aura la meilleure représentation du nuage et que l'on pourra ainsi visualiser dans un plan en 2D ou espace 3D et déterminer alors la topologie des individus (proches vs éloignés).

L'ACP vise à trouver le sous-espace sur lequel le projeté de $\mathcal{N}$ aura la plus grande "dispersion". Aussi, le principe de l'ACP répond simultanément aux deux objectifs suivants:

- Pour les individus

  L'objectif de la méthode ACP est de projeter les individus sur des axes appelés *axes factoriels* en conservant le mieux possible les distances entre individus. Cela revient à déformer le moins possible le nuage de points initial lorsqu'on le projette sur un axe ou un plan.

- Pour les variables

On construit de nouvelles variables, appelées *composantes principales*, par combinaison linéaire des variables initiales et telles que ces nouvelles variables aient la plus grande variance possible. Cette variance est représentée par la *valeur propre* de l'axe. Enfin, on choisit les composantes principales les plus non corrélées de façon à ce que les axes de représentation dans le nouvel espace H soient les plus orthogonaux possibles.

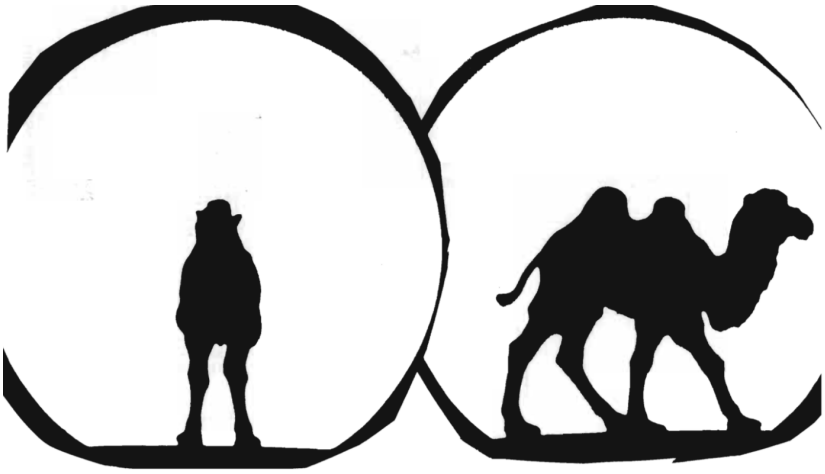


Figure 1: Quel est le meilleur axe ?

1.2 Centrage et réduction

Notons X, la matrice représentant le nuage de points $\mathcal{N}$. Les lignes représentent les individus $I_{1\leq i\leq n}$ et les colonnes les variables $X_{1\leq j\leq p}$. On a:

$$X = \begin{array}{c} \\ I_1 \\ \vdots \\ I_i \\ \vdots \\ I_n \end{array} \begin{array}{c} \begin{array}{cccccc} X_1 & \cdots & X_j & \cdots & X_p \end{array} \\ \begin{pmatrix} x_{1,1} & \cdots & x_{1,j} & \cdots & x_{1,p} \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ x_{i,1} & \cdots & x_{ij} & \cdots & x_{i,p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,j} & \cdots & x_{np} \end{pmatrix} \end{array}$$

La matrice centrée-réduite $\tilde{X}$ est obtenue telle que:

$$\tilde{X} = \bar{X}.S^{-1} = \begin{pmatrix} \frac{x_{11}-\bar{X}_1}{\sigma(X_1)} & \cdots & \frac{x_{1p}-\bar{X}_p}{\sigma(X_p)} \\ \vdots & \ddots & \vdots \\ \frac{x_{1n}-\bar{X}_1}{\sigma(X_1)} & \cdots & \frac{x_{np}-\bar{X}_p}{\sigma(X_p)} \end{pmatrix}$$

Le choix de réduire ou non est un choix de modèle :

- si l'on ne réduit pas le nuage : une variable à forte variance va "tirer" tout l'effet de l'ACP à elle,
- si l'on réduit le nuage : une variable qui n'est qu'un bruit va se retrouver avec une variance apparente égale à une variable informative.

Si les variables aléatoires sont dans des unités différentes, la réduction est obligatoire pour ne pas accorder un poids plus important à certaines variables.

Pour centrer-réduire une matrice X, on utilise la commande `scale`:

```
X_tilde <- scale(X)
```

On peut convertir `X_tilde` en un dataframe à l'aide de la commande `as.data.frame`.

Dans la suite du document, on considère que X est centrée-réduite de sorte que:

$$X \leftarrow \tilde{X}$$

1.3 Sous-espace d'ajustement

Comme énoncé précédemment, l'ACP vise à trouver le sous-espace H qui résume au mieux les données de $\mathcal{N}$.

Une analogie intéressante est celle d'une image pour résumer un objet. La qualité de restitution est jugée optimale si les critères suivants sont satisfaits:

- L'image restitue fidèlement la forme générale.
- On a la meilleure représentation de la diversité.
- Les distances entre individus sont préservées.

Ainsi, comment peut-on trouver la meilleure image, c'est-à-dire le meilleur sous-espace H pour représenter $\mathcal{N}$?

Pour cela, on cherche l'axe qui déforme le moins possible le nuage par projeté.

Par exemple, si l'on considère la figure 2, on cherche à minimiser la distance [au carré] entre l'individu I_i et son projeté $\pi(I_i)$ sur l'axe engendré par le vecteur $\vec{v_1}$. Par le théorème de Pythagore, cela revient à maximiser la distance entre O et $\pi(I_i)$.

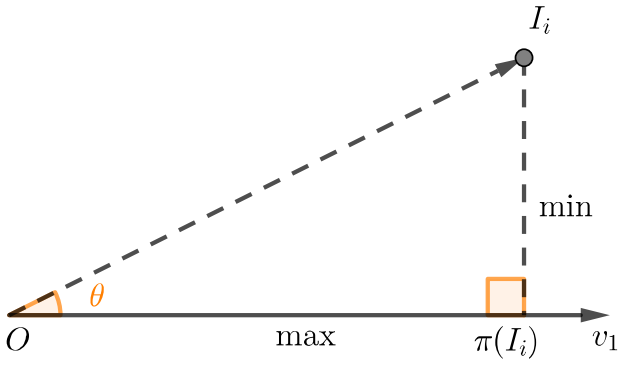


Figure 2: Projeté orthogonal d'un point sur un axe factoriel

Dès lors, l'objectif devient maximiser $\sum_i d(O, \pi(I_i))^2$.

Sans perte de généralités, notre exemple effectué ici sur un individu peut être appliqué sans peine dans le cas d'un ajustement d'une variable. On revient toujours à la recherche d'axes orthogonaux minimisant les distances des projetés sur ces axes. On montre d'ailleurs que l'étude des individus est équivalente à l'étude des variables.

Dans le cours, on démontre alors que $\vec{v_1}$ est le vecteur propre associé à la valeur propre λ_1 de la matrice symétrique $X.X^\top$.

Un critère couramment utilisé est la mesure de dispersion (ou variabilité) d'un nuage de points que l'on appelle aussi l'*inertie* et qui correspond, grosso modo, à la notation variance généralisée à plusieurs dimensions.

Ainsi, soit $(\lambda_1 \geq ... \geq \lambda_p)$ le spectre de la matrice symétrique $X.X^\top$, l'inertie de l'axe associé au vecteur $\vec{v_i}$ est la valeur propre λ_i. On exprime souvent cette quantité relativement en pourcentage: $\frac{\lambda_i}{\sum_i^p \lambda_i}$.

2 Aide à l'interprétation

Afin d'introduire le problème de l'ACP, nous utiliserons le fichier `temperature.csv` constitué de 15 villes de France et 12 températures mensuelles moyennes (sur 30 ans).

2.1 Valeurs propres λ et choix des axes

Pour définir le nombre d'axes à retenir, on étudie les valeurs propres obtenues. Chaque valeur propre correspond à la part d'inertie projetée sur un axe donné.

On ne retient donc que les axes avec les plus fortes valeurs propres. Le choix des axes retenus est un peu délicat. On peut donner quelques règles :

- *Règle de l'inertie minimale* : On sélectionne les premiers axes jusqu'à atteindre un % donné d'inertie expliquée (80% par exemple).
- *Règle du coude* : On observe souvent de fortes valeurs propres au départ puis ensuite de faibles valeurs avec un décrochage dans le graphique des éboulis des valeurs propres (Scree plot). On retient alors les axes jusqu'à ce qu'on observe une chute brutale entre deux valeurs successives.

Sous R, on utilise les packages `FactorMineR` et `factoextra` dédiés à l'exploration et l'analyse en hautes dimensions pour appliquer une ACP[1].

La commande utilisée pour réaliser un modèle d'ACP sur une matrice X avec h dimensions est:

```
PCA(X, ncp = h, graph = FALSE)
```

L'option `graph` sera détaillée dans la suite du document. Par défaut, si `ncp` est négligée, $h = 2$.

```
library("FactoMineR")
temp.pca <- PCA(temp_scale, graph = FALSE)
```

Les valeurs propres (i.e proportion de variances) retenues par les composantes principales peuvent être extraites à l'aide de la fonction `get_eigenvalue`:

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

```
## Loading required package: ggplot2
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
eig.val <- get_eigenvalue(temp.pca)
round(eig.val,2) # Pour restreindre à 2 chiffres après la virgule
```

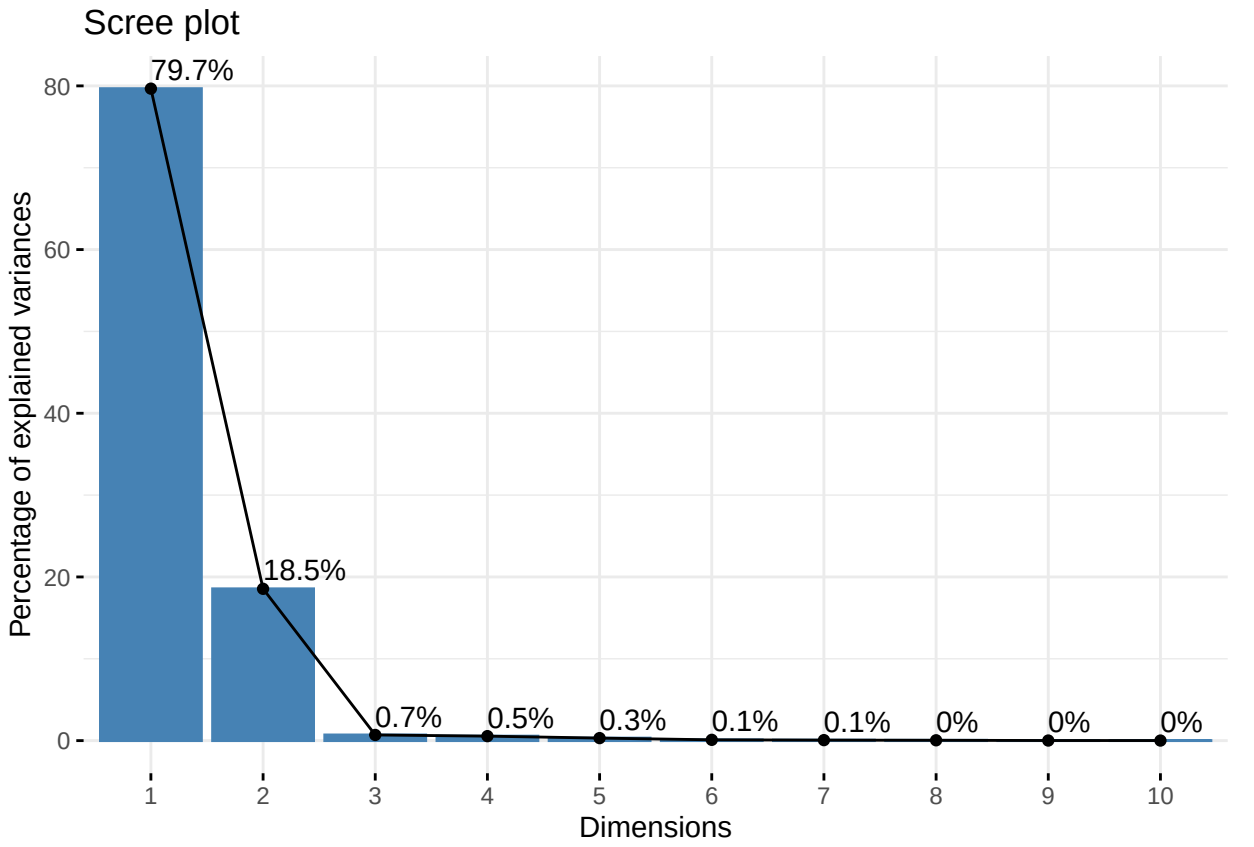
```
##        eigenvalue variance.percent cumulative.variance.percent
## Dim.1        9.56           79.66                       79.66
## Dim.2        2.23           18.55                       98.21
## Dim.3        0.08            0.70                       98.91
## Dim.4        0.07            0.55                       99.46
## Dim.5        0.04            0.31                       99.77
## Dim.6        0.01            0.10                       99.86
```

[1] Il existe d'autres packages sous R comme `ade4` spécialisé pour l'analyse biologique.

```
## Dim.7       0.01            0.06                99.93
## Dim.8       0.01            0.04                99.97
## Dim.9       0.00            0.01                99.98
## Dim.10      0.00            0.01               100.00
## Dim.11      0.00            0.00               100.00
## Dim.12      0.00            0.00               100.00
```

Pour observer le graphique des éboulis des valeurs propres, on utilise la commande:

```
fviz_eig(temp.pca, addlabels = TRUE)
```



Sur ce graphique, on distingue clairement que la *première dimension explique une grande partie de l'information de notre nuage de points* (à hauteur de 79.7%).

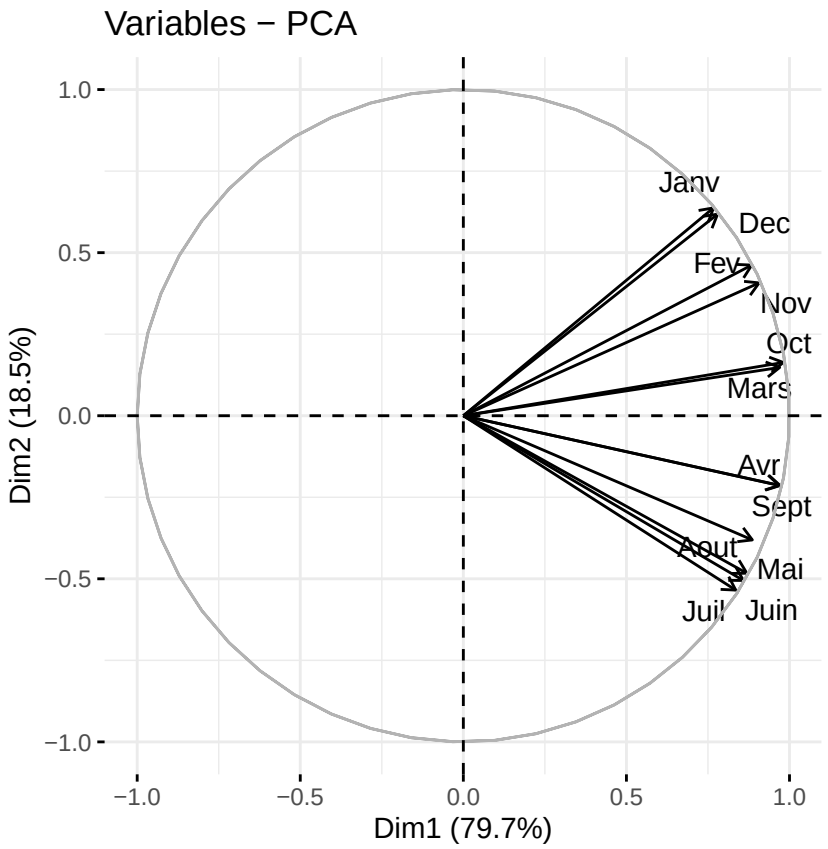
En terme de variance cumulée, les deux premières dimensions expliquent *98.21%* de l'information de notre jeu de données ce qui est excellent.

2.2 Analyse des variables

2.2.1 Cercle des corrélations

Le cercle des corrélations est un cercle, de rayon 1 où l'axe des abscisses représente le premier axe factoriel selon $\vec{v_1}$ et l'axe des ordonnées le second axe.

```
fviz_pca_var(temp.pca, repel = TRUE) # Évite les chevauchements
```



À l'intérieur du cercle, des flèches partent du centre. Elles sont plus ou moins grandes, et peuvent aller jusqu'à toucher le cercle, sans jamais le dépasser.

Le projeté sur l'axe des abscisses de la variable X_j correspond à la corrélation entre X_j et la composante principale $\vec{v_1}$ (Dim1). On a:

$$\cos(\vec{X_j}) = \rho_{X_j, v_1}$$

Le projeté sur l'axe des ordonnées donne la corrélation avec la composante principale $\vec{v_2}$ (Dim2).

$$\sin(\vec{X_j}) = \rho_{X_j, v_2}$$

Par exemple, on voit ici que toutes les variables sont très corrélées à l'axe factoriel engendré par $\vec{v_1}$. Toutes les flèches dépassent une valeur projetée de 0.75 sur `Dim1`.

Les coordonnées précises des composantes principales s'obtiennent à l'aide de la commande:

```
round(get_pca_var(temp.pca)$coord, 2)
```

```
##       Dim.1 Dim.2 Dim.3 Dim.4 Dim.5
## Janv   0.77  0.64  0.01 -0.02  0.06
## Fev    0.88  0.46 -0.01 -0.04 -0.01
## Mars   0.97  0.15 -0.04 -0.16 -0.09
## Avr    0.97 -0.21 -0.02 -0.11  0.06
## Mai    0.87 -0.48 -0.03 -0.02  0.08
## Juin   0.86 -0.51 -0.06  0.00  0.05
## Juil   0.84 -0.53 -0.07  0.09 -0.02
## Aout   0.89 -0.38  0.26  0.00 -0.02
## Sept   0.97 -0.21 -0.04  0.04 -0.07
```

```
## Oct   0.98  0.16 -0.01  0.08 -0.05
## Nov   0.91  0.41 -0.01  0.09 -0.01
## Dec   0.78  0.62  0.02  0.06  0.06
```

Par exemple, On voit que $\vec{v_1}$ est formé par une combinaison linéaire des différentes variables initiales telle que:

$$\vec{v_1} = 0.77\vec{Janv} + 0.89\vec{Fev} + 0.97\vec{Mars} + ... + 0.91\vec{Nov} + 0.78\vec{Dec}$$

2.2.2 Qualité de représentation

Il n'est pas toujours simple d'interpréter le cercle des corrélations. Plusieurs choses à savoir pour ne pas faire d'erreurs:

- Il est préférable de n'interpréter *uniquement que les variables avec une norme proche de 1*.

  Une flèche qui est petite sur le premier plan factoriel signifie qu'elle est faiblement corrélée à la première composante principale $\vec{v_1}$, et faiblement corrélée aussi à $\vec{v_2}$. Mais elle peut très bien l'être à $\vec{v_3}$, $\vec{v_4}$, etc.

  Pour s'assurer de la qualité de représentation d'une variable au sein d'un axe (ou plan) factoriel, on utilise la mesure du $\cos^2$.

  Soit une variable X_j, la valeur $\cos^2$ pour l'axe factoriel engendré par $\vec{v_k}$ correspond à la valeur $\rho^2_{X_j,v_k}$.

  $$qlt(X_j, v_k) = \cos^2(\vec{X_j}, \vec{v_k}) = \rho^2_{X_j,v_k}$$

  Cette valeur peut aussi être interprêtée géométriquement[2]. Elle est obtenue par la commande:

```
round(get_pca_var(temp.pca)$cos2, 2)
```

```
##       Dim.1 Dim.2 Dim.3 Dim.4 Dim.5
## Janv   0.59  0.41  0.00  0.00  0.00
## Fev    0.78  0.21  0.00  0.00  0.00
## Mars   0.94  0.02  0.00  0.03  0.01
## Avr    0.94  0.05  0.00  0.01  0.00
## Mai    0.76  0.23  0.00  0.00  0.01
## Juin   0.74  0.26  0.00  0.00  0.00
## Juil   0.70  0.29  0.01  0.01  0.00
## Aout   0.79  0.14  0.07  0.00  0.00
## Sept   0.94  0.05  0.00  0.00  0.01
## Oct    0.96  0.03  0.00  0.01  0.00
## Nov    0.82  0.17  0.00  0.01  0.00
## Dec    0.61  0.38  0.00  0.00  0.00
```

- Concernant l'angle formé entre deux variables X_i et X_j, le cosinus de celui-ci peut être interprété intuitivement comme la corrélation entre ces deux variables. Autrement dit, $\cos(\vec{X_i}, \vec{X_j}) \approx \rho_{X_i,X_j}$,

  Mais *attention* ! Le cercle des corrélations est une projection de l'espace des variables $\mathbb{R}^p$ dans le plan $\mathbb{R}^2$. En conséquence les angles sont déformés et cette interprétation n'est valable que si les variables X_i et X_j sont correctement représentées dans le plan factoriel (i.e. que la norme des variables est proche de 1). Pour connaitre la corrélation entre les variables, on fera plutôt *un corrélogramme* comme vu dans le TP1.

[2] On parle de $\cos^2$ car on analyse prioritairement l'axe factoriel engendré par $\vec{v_1}$, ce qui revient à calculer la valeur $\cos^2(\vec{X_k})$.

2.2.3 Contribution aux axes

La contribution d'une variable est la valeur relative d'implication de la variable dans la composante principale. Plus la valeur de contribution est forte, plus la variable contribue à la composante principale en question. La contribution ctr de la variable X_j pour la composante principale v_k est calculée telle que:

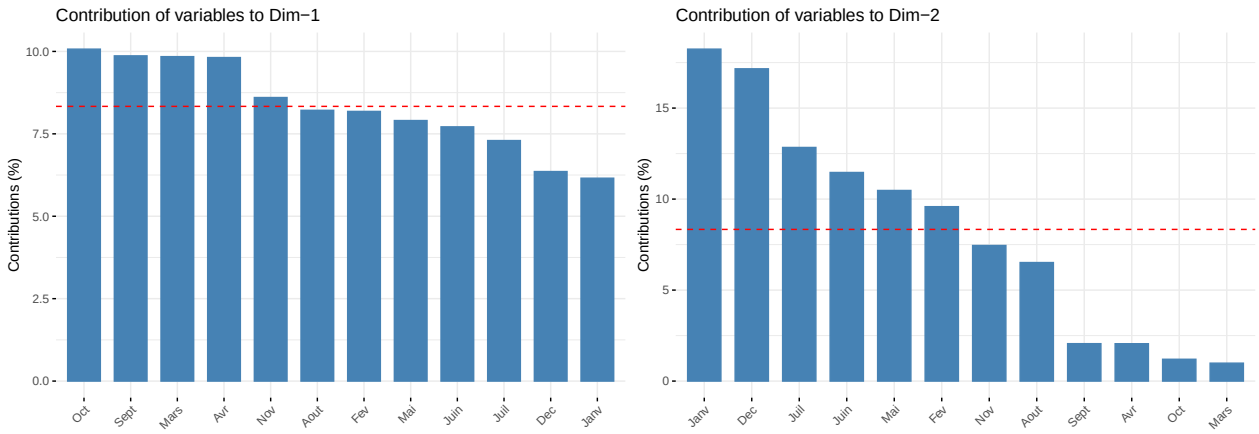
$$ctr(X_j, v_k) = \frac{\rho^2_{X_j, v_k}}{\lambda_k}$$

Sous hypothèse d'une loi uniforme, la contribution moyenne d'une variable est de $\frac{1}{p}$. On peut visualiser le graphique des contributions des variables à l'axe i grace à la commande:

```
fviz_contrib(X.pca, choice = "var", axes = i)
```

Pour les deux premiers axes, on a:

```
fviz_contrib(temp.pca, choice = "var", axes = 1)
fviz_contrib(temp.pca, choice = "var", axes = 2)
```



On voit grâce à ces graphiques que presque toutes les variables contribuent aux axes du plan factoriel. On note néanmoins que la variable *Aout* est la seule à contribuer légérement moins que la moyenne aux deux axes.

2.2.4 Interprétation des axes

Dans notre cas, on peut assumer que toutes les variables sont bien représentées par notre plan factoriel. Grâce à la contribution des variables sur les axes, on peut tenter d'interpréter chacun des deux axes:

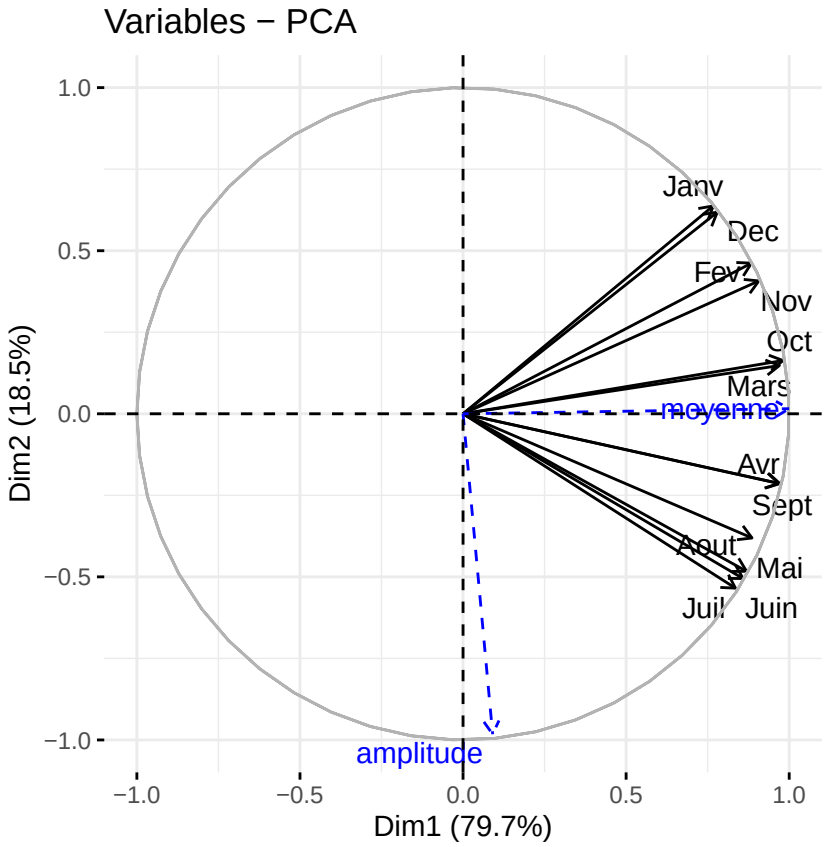
- Pour l'axe factoriel engendré par $\vec{v_1}$ (Dim1). Toutes les variables semblent corrélées à celui-ci. On remarque néanmoins que les variables les plus fortement corrélées et contributives sont globalement celles des mois de Printemps (Mars, Avril) et Automne (Septembre, Octobre, Novembre). Cet axe peut-être interprété comme celui des **villes chaudes et froides**.
- Pour l'axe factoriel engendré par $\vec{v_2}$ (Dim2), les variables sont plus faiblement corrélées. Néanmoins, on remarque un effet de symétrie entre les mois "froids" de l'année (Octobre $\rightarrow$ Mars) et les mois "chauds" (Avril $\rightarrow$ Septembre). De même, les variables les plus contributives sont celles qui apportent les pics de températures: en hiver (Decembre, Janvier) et été (Juillet). Cet axe peut-être interprété comme l'**amplitude thermique** d'une ville durant l'année.

On peut confirmer nos hypothèses en créant les deux nouvelles variables suivantes:

```
temp_scale$moyenne <- scale(apply(temp[, 2:13], 1, mean))
temp_scale$amplitude <- scale(apply(temp[, 2:13], 1, max) - apply(temp[, 2:13], 1, min))
```

Lorsque l'on effectue une ACP, il est possible d'indiquer certaines variables descriptives avec l'option `quanti.sup` et `quali.sup` (en fonction de si les variables sont quantitatives ou qualitatives) Elles ne sont alors pas prises en compte lors de l'évaluation de l'ACP mais viennent éclairer certaines analyses:

```
# On indique les numéros de colonnes
temp.pca <- PCA(temp_scale, graph = FALSE, quanti.sup = c(13,14))
fviz_pca_var(temp.pca, repel = TRUE)
```

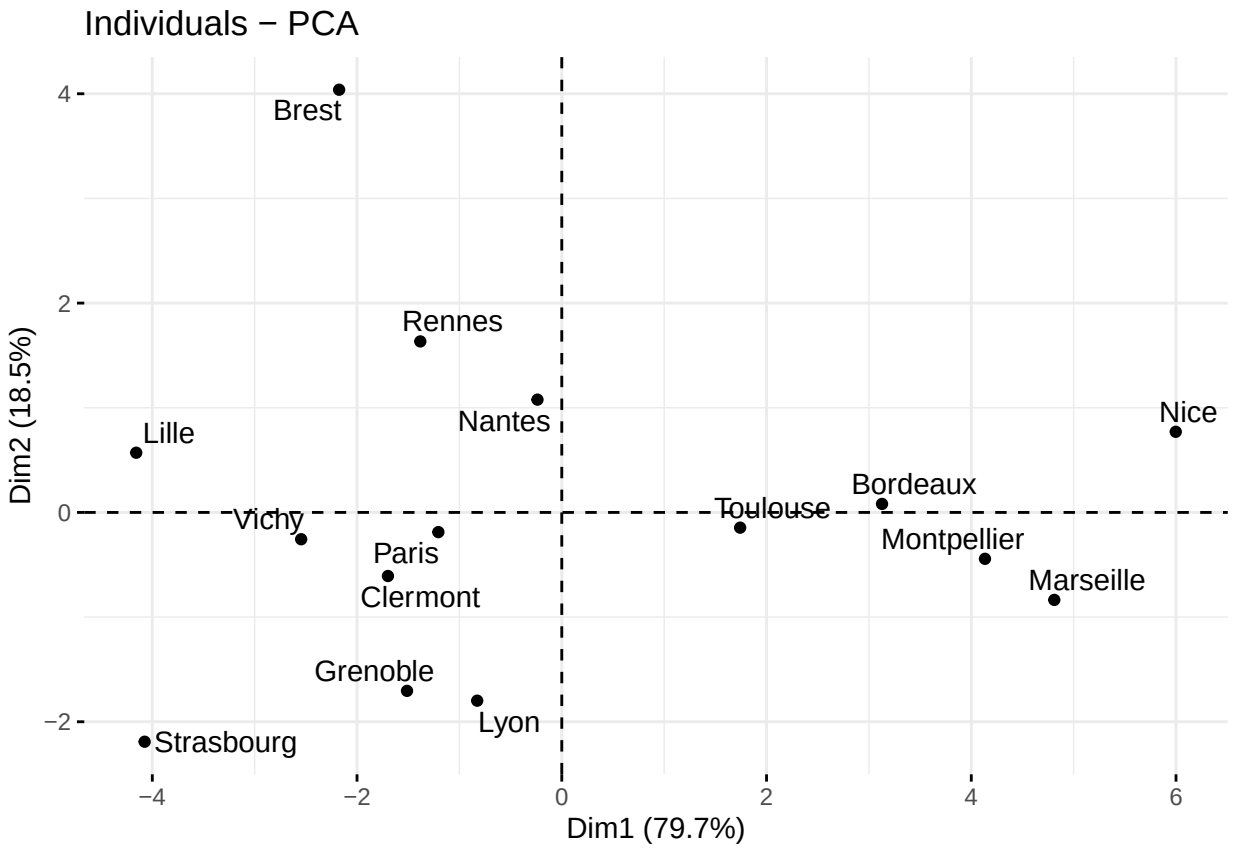


On voit sur ce nouveau cercle que la variable `moyenne` est extrêmement corrélée à la Dim1. De même, la variable `amplitude` est fortement corrélée à Dim2 ce qui vient confirmer nos précédentes hypothèses.

2.3 Analyse des individus

Lorsque l'on retient uniquement les deux premières composantes principales, il est possible de représenter nos individus dans le plan factoriel suivant $\vec{v_1}$ et $\vec{v_2}$. La commande utilisée est:

```
fviz_pca_ind (temp.pca,
              repel = TRUE)
```



Cette projection vient confirmer notre analyse précédente du cercle des corrélations.

- On retrouve bien un clivage des villes selon l'axe Dim1, entre la gauche et la droite. Les villes à gauche peuvent être interprétées comme des villes à *climat frais* (Lilles, Strasbourg, Vichy). Alors que les villes à droite de l'axe sont des villes à *climat chaud* (Toulouse, Nice, Montpellier).
- De même, l'axe Dim2 semble effectivement représenter l'amplitude thermique des villes. Les villes en haut de l'axe (Brest) semblent adopter un *climat stable* entre les saisons. L'hiver est doux et l'été est frais. Les villes en bas de l'axe (Strasbourg, Lyon) sont caractérisées par un *climat à amplitude forte*, les hivers sont froids et les étés chauds.

De la même façon que lors de l'étude des variables, des mesures de qualité et de contribution peuvent être appliquées aux individus.

Soit un individu I_i et un axe engendré par la composante principale $\vec{v_k}$, la qualité de représentation de I_i correspond au $\cos^2$ entre le vecteur (O, I_i) et la $\vec{v_k}$:

$$qlt(I_i, v_k) = \cos^2(\vec{I_i}, \vec{v_k}) = \frac{\pi(I_i, v_k)^2}{\sum_k \pi(I_i, v_k)^2}$$

Où $\pi(I_i, v_k)$ représente la valeur de la coordonnée projetée de l'individu I_i sur l'axe engendré par $\vec{v_k}$. Ces valeurs sont obtenues par la commande:

```
round(get_pca_ind(temp.pca)$coord, 2)
```

```
##             Dim.1 Dim.2 Dim.3 Dim.4 Dim.5
## Bordeaux     3.13  0.08 -0.10 -0.70  0.05
## Brest       -2.18  4.04  0.08  0.11 -0.06
## Clermont    -1.70 -0.61  0.19 -0.06 -0.15
```

```
## Grenoble    -1.51 -1.70  0.04 -0.14 -0.43
## Lille       -4.16  0.57  0.30  0.33 -0.04
## Lyon        -0.83 -1.80  0.11 -0.02 -0.03
## Marseille    4.81 -0.84  0.02  0.37  0.11
## Montpellier  4.13 -0.44  0.07  0.17  0.01
## Nantes      -0.24  1.08 -0.02 -0.20  0.13
## Nice         6.00  0.77 -0.05  0.22  0.03
## Paris       -1.21 -0.19  0.01 -0.16  0.34
## Rennes      -1.38  1.63  0.07 -0.16  0.05
## Strasbourg  -4.07 -2.19  0.15  0.09  0.34
## Toulouse     1.74 -0.15  0.16 -0.02 -0.28
## Vichy       -2.55 -0.26 -1.02  0.17 -0.06
```

La qualité de représentation de l'individu est obtenue par la commande:

```
round(get_pca_ind(temp.pca)$cos2, 2)
```

```
##             Dim.1 Dim.2 Dim.3 Dim.4 Dim.5
## Bordeaux     0.95  0.00  0.00  0.05  0.00
## Brest        0.22  0.77  0.00  0.00  0.00
## Clermont     0.86  0.11  0.01  0.00  0.01
## Grenoble     0.42  0.53  0.00  0.00  0.03
## Lille        0.97  0.02  0.01  0.01  0.00
## Lyon         0.17  0.82  0.00  0.00  0.00
## Marseille    0.96  0.03  0.00  0.01  0.00
## Montpellier  0.99  0.01  0.00  0.00  0.00
## Nantes       0.04  0.90  0.00  0.03  0.01
## Nice         0.98  0.02  0.00  0.00  0.00
## Paris        0.89  0.02  0.00  0.02  0.07
## Rennes       0.41  0.57  0.00  0.01  0.00
## Strasbourg   0.77  0.22  0.00  0.00  0.01
## Toulouse     0.95  0.01  0.01  0.00  0.02
## Vichy        0.85  0.01  0.14  0.00  0.00
```

Tous les individus sont très bien représentés (> 90%) par le plan factoriel. Si l'on constate des individus mal représentés, ils ne doivent pas être pris en compte dans l'analyse.

La contribution de l'individu I_i à l'axe engendré par $\vec{v_k}$ est quant à elle calculée telle:

$$ctr(I_i, v_k) = \frac{\pi(I_i, v_k)^2}{n \times \lambda_k}$$

La contribution des individus pour l'axe k peut être visualisée à l'aide de la commande:

```
fviz_contrib(X.pca, choice = "ind", axes = k)
```

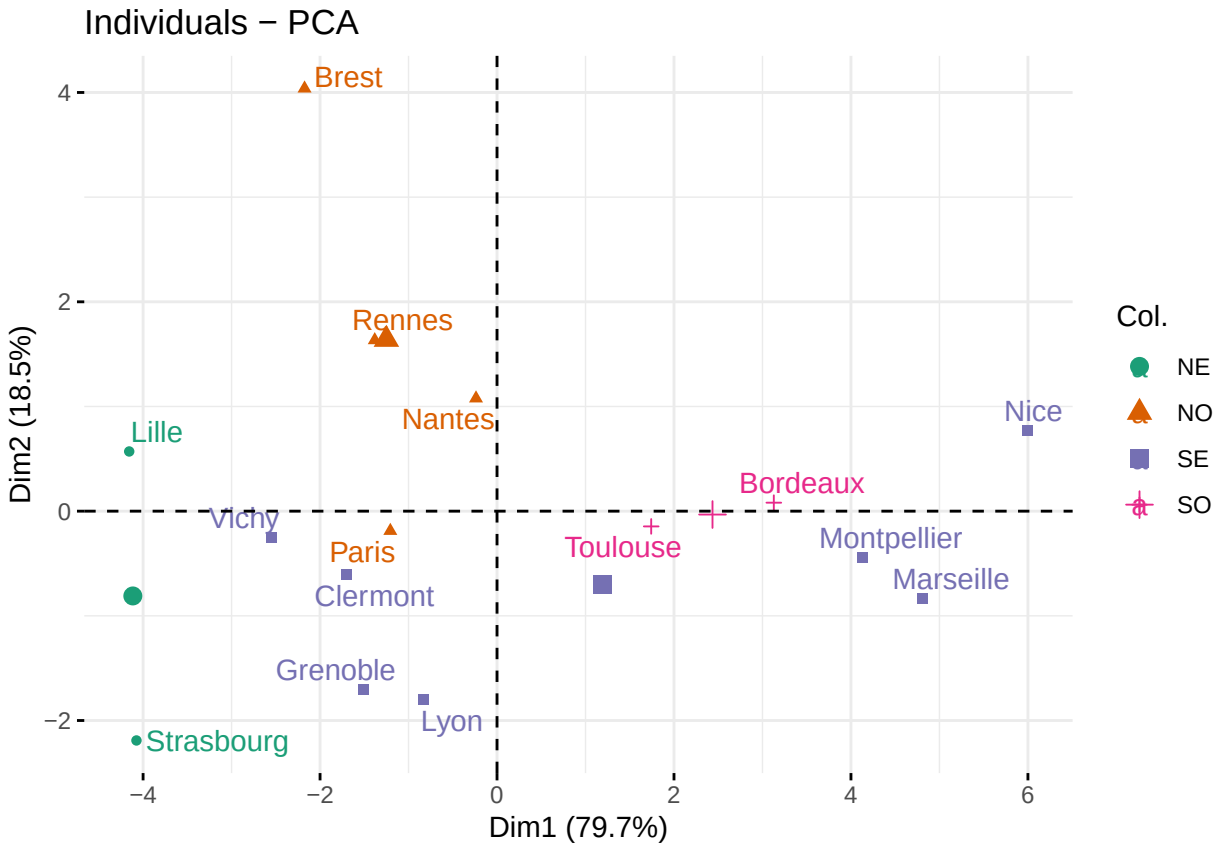
On peut également affecter des variables explicatives aux individus. Par exemple, l'orientation géographique des villes.

```
temp_scale$orientation <- c("SO", "NO", "SE", "SE", "NE", "SE", "SE", "SE", "NO", "SE",
                            "NO", "NO", "NE", "SO", "SE")
```

On ajoute l'orientation en tant que variable qualitative à l'ACP.

```
library(RColorBrewer)
temp.pca <- PCA(temp_scale, graph = FALSE, quanti.sup = c(13,14), quali.sup = c(15))
fviz_pca_ind(temp.pca,
```

```
            repel = TRUE,
            col.ind = temp_scale$orientation,
            palette = brewer.pal(n = 4, name = "Dark2"))
```



3 Exercices

3.1 Les iris de Fischer

On considère le fichier `iris.csv` sur Celene répertoriant 150 individus fleurs d'iris. On donne la description suivante des colonnes:

Colonne	Description	Value
`sepal_length`	Longueur des sépales	Int
`sepal_width`	Largeur des sépales	Int
`petal_length`	Longueur des pétales	Int
`petal_width`	Largeur des pétales	Int
`species`	Espèce d'iris	{Versicolor, Virginica, Setosa}

1. Statistiques descriptives

   (a) Proposer une analyse préliminaire par statistiques descriptives du jeu de données `iris`. Votre analyse[3] devra contenir notamment:

[3] Vous pourrez vous aider la fonction `chart.Correlation` de la librairie `PerformanceAnalytics`.

Figure 3: Les iris de Fischer

- Distribution de chaque variable puis analyses synthétiques agrégées par espèce.
- Corrélation entre les variables.

(b) Sur la base de ces analyses, quelles variables vous semblent pertinentes pour l'ACP ?

2. Calculer les valeurs propres de la matrice des données `iris`. Combien d'axes proposez vous de retenir pour l'ACP ? Détaillez votre réponse.
3. Analyse des variables
   (a) Dresser le cercle des corrélations de l'ACP. Commentez la qualité de représentation et la contribution de chaque variable quant aux axes retenus.
   (b) Interpréter la signification des axes retenus. Vous pourrez vous aider de la contribution des variables aux axes factoriels.
4. Analyse des individus
   (a) Présenter la projection des indivus dans le plan factoriel. Vous colorerez dans un premier temps les points en fonction de l'espèce d'iris.
   (b) Colorer les individus en fonction de leur contribution aux axes factoriels. Que remarquez-vous ? Pouvez l'expliquer ?
   (c) Commenter la qualité de représentation des individus.
5. Apprentissage statistique

   L'option `addEllipses=TRUE` de la fonction `fviz_pca_ind` permet de dessiner l'ellipse de confiance (covariance ellipse error) à 95%.
   (a) Sous quelle condition la définition d'ellipses de confiance est-elle valable ? Est-ce le cas selon vous-ici ? Pourquoi ?
   (b) Proposer un algorithme permettant de classifier automatiquement une nouvelle iris inconnue et ainsi déterminer son espèce. Vous évoquerez les limites de votre approche et possibilités pour pallier à ces effets.
5. Reprendre l'analyse du jeu de données `iris` mais en effectuant ici une ACP **non réduite**. On appliquera pour ça l'option `scale = FALSE` lors de l'exécution de la fonction `PCA`.

   Que remarquez vous ? Quelle méthode semble finalement donner les meilleurs résultats ici ? Expliquer ces résultats.

3.2 Sommeil des mammifères

On considère le fichier `sleep.csv` sur Celene répertoriant les données de 70 espèces de mammifères concernant leur sommeil et quelques autres caractéristiques. On donne la description suivante des colonnes:

Colonne	Description	Value
`name`	Nom français vernaculaire de l'animal	String
`genus`	Genre, subdivion de la classification biologique	String
`vore`	Régime alimentaire de l'animal	String
`order`	Ordre, subdivion de la classification biologique	String
`sleep_total`	Durée (en h) de sommeil sur une journée	Double
`sleep_rem`	Durée (en h) de sommeil paradoxal	Double
`awake`	Durée (en h) où l'animal est éveillé	Double
`brain_wt`	Masse (en kg) moyenne du cerveau de l'animal	Double
`body_wt`	Masse (en kg) totale moyenne de l'animal	Double
`brain_body_ratio`	Ratio masse cerveau, masse totale $\frac{\texttt{brain\_wt}}{\texttt{body\_wt}}$	Double
`gest_day`	Période de gestation moyenne de l'animal	Int

1. Statistiques descriptives
   (a) Proposer une analyse préliminaire par statistiques descriptives du jeu de données `sleep`. Votre analyse devra contenir notamment:
      - Distribution de chaque variable puis analyses synthétiques agrégées selon différentes variables qualitatives.
      - Corrélation entre les variables.

   (b) Sur la base de ces analyses, quelles variables vous semblent pertinentes pour l'ACP ? Quelles variables explicatives proposez-vous ?
2. On propose de compléter les données manquantes de la colonne `sleep_rem` en utilisant une technique de regression par *la méthode des moindres carrés*. Quelle valeur est estimée pour l'individu *Lamantin* ? Compléter les valeurs manquantes.
3. Calculer les valeurs propres de la matrice des données `sleep`. Combien d'axes proposez vous de retenir pour l'ACP ? Détaillez votre réponse.
4. Analyse des variables
   (a) Commentez la qualité de représentation et la contribution de chaque variable quant aux axes retenus.

   (b) Interpréter la signification des axes retenus. Vous pourrez vous aider de la contribution des variables aux axes factoriels.
5. Analyse des individus
   (a) Présenter la projection des indivus dans l'espace factoriel retenu. Vous colorerez dans un premier temps les points en fonction de la variable explicative retenue.

   Pour une projection 3D, on utilisera la commande `plot_ly(df, x = ~Dim.1, y = ~Dim.2, z = ~Dim.3)` de la librairie `plotly` où `df` est votre dataframe des coordonnées des individus et `Dim.k`, la colonne des coordonnées sur l'axe k.

   (b) Colorer les individus en fonction de leur qualité de réprésentation aux axes factoriels puis en fonction de la contribution. Commentez ces résultats.

3.3 Classiffication de caractères manuscrits

On considère le fichier `mnist.csv` sur Celene. Ces données proviennent de la base MNIST[4] sur laquelle des milliers de chercheurs ont travaillé. Elle est constituée initialement de 70.000 chiffres manuscrits au format 28 pixels par 28 pixels où chaque pixel est représenté par un niveau de gris allant de 0 à 255. Un chiffre manuscrit est vu comme un vecteur de $\{0, ..., 255\}^{28\times 28}$.

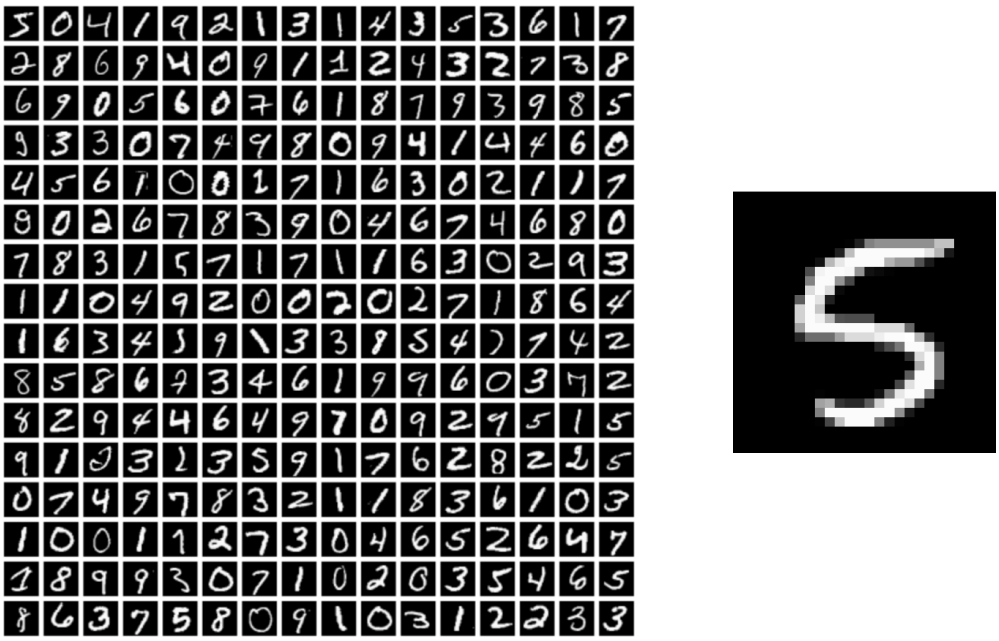


Figure 4: Exemple de caractères manuscrits. Le caractère manuscrit à droite fait partie de la classe '5'

Pour limiter le temps de calcul et la mémoire nécessaire, nous ne considérons que les 20.000 premiers chiffres manuscrits de la base originale. On donne la description des colonnes suivantes:

- chaque ligne correspond à un chiffre manuscrit.
- la première colonne contient la *classe* (ou label) du caractère, c'est-à-dire le chiffre qu'il représente.
- les colonnes suivantes, contiennent les valeurs des $28 \times 28 = 784$ pixels de l'image en commençant par le coin supérieur gauche et parcourant l'image ligne par ligne.

On donne la fonction de visualisation suivante:

```
img <- function(data, row_index){

    r <- as.numeric(data[row_index, 2:785])
    im <- matrix(nrow = 28, ncol = 28)
    j <- 1
    for(i in 28:1){
        im[,i] <- r[j:(j+27)]
        j <- j+28
    }
    png(file = "out.png", width = 210, height = 300)
    image(x = 1:28,
          y = 1:28,
```

[4] http://yann.lecun.com/exdb/mnist/

```
          z = im,
          col=gray((0:255)/255),
          main = paste("Number:", data[row_index, 1]))
    dev.off()
}
```

L'appel `img(mnist, i)` retourne la figure correspondant au caractère manuscrit ligne i.

1. Statistiques descriptives

   (a) Proposer une analyse préliminaire par statistiques descriptives du jeu de données `mnist`. Votre analyse devra contenir notamment:

   - Nombre de caractères de chaque classe.
   - Des premiers indicateurs sur la proportion de gris par pixel, puis agrégé par classe de caractère.

   (b) Sur la base de ces analyses, certaines zones de l'image vous semblent t-elles plus pertinentes pour l'analyse ? Lesquelles ? Pourquoi ?

2. Classification par l'algorithme des k plus proches voisins (kNN).

   L'algorithme des k proches voisins (k-Nearest Neightbors) est une méthode de prédiction qui, pour une base de données d'apprentissage, cherche à déterminer la classe d'une donnée inconnue. On donne l'algorithme en pseudo-code suivant:

Algorithm 1: k-Nearest Neightbors

Result: $Classify(\mathbf{X}, \mathbf{Y}, x)$
Input : $\mathbf{X}$, l'ensemble d'apprentissage,
$\mathbf{Y}$, classe des données de $\mathbf{X}$,
x, une nouvelle donnée à classifier,
k, nombre de voisins considérés
Output: Classe prédite pour x

for $i \in [\![1, |\mathbf{X}|]\!]$ **do**
  Calcul de la distance euclidienne $\|\mathbf{X}_i - x\|^2$
end
Calcul de l'ensemble I contenant les indices des k plus petites distances $\|\mathbf{X}_i - x\|^2$.
return Classe majoritaire pour $\{\mathbf{Y}_i | i \in I\}$

L'idée générale de cet algorithme est très simple. Pour une nouvelle donnée d'entrée x, on évalue sa distance à toutes les autres données connues de notre base d'apprentissage $\mathbf{X}$.

On rappelle que la distance euclidienne entre deux éléments $x, y \in \mathbb{R}^p$ est définie telle que:

$$\|x - y\| = \sqrt{\sum_{i=1}^{p} (x_i - y_i)^2}$$

On retient ensuite uniquement les k voisins $\mathbf{X}_i$ les plus proches de x. On regarde alors les classes $\mathbf{Y}_i$ de ces données $\mathbf{X}_i$, puis on prédit la classe la plus présente. Par défaut on utilisera $k = 1$.

(a) En assumant que $\mathbf{X}$ est doté de n individus définis dans un espace de dimension p. Quelle est la complexité de l'algorithme des k-Nearest Neightbors pour $k = 1$.

(b) Diviser le jeu de données `mnist` en deux ensembles :

- Un ensemble d'apprentissage (train set) qui contiendra 80% du jeu initial.

- Un ensemble test (test set) qui contiendra le reste des données.

On veillera à conserver les labels des deux ensembles dans un vecteur à part.

(c) La commande `knn` du package `class` permet de réaliser une classification à l'aide de l'algorithme des k-Nearest Neightbors:

```
library(class)

knn(X_train, X_test, cl = Y_train_label, k = nb_neightbors)
```

Appliquer l'algorithme kNN (avec $k = 1$) sur votre ensemble d'apprentissage et de test. On veillera à sauvegarder le résultat de la fonction dans une variable `prediction`:

```
prediction <- knn(...)
```

Donner le temps d'exécution de l'algorithme.

(d) La commande `table(Y_test_label, prediction)` permet de dresser la *matrice de confusion* C de la classification effectuée. Le nombre c_{ij} représente le nombre d'éléments de la classe i classifiés en tant que j.

Quel est le pourcentage de caractères manuscrits de l'ensemble de test qui ont été mal classés ? Cet algorithme vous semble t-il efficace ? Quel critique peut-on lui faire ?

(e) Pour chaque classe, identifier un exemple de caractère mal classé par l'algorithme. Vous illustrerez ces caractères à l'aide de la fonction `img` donnée plus haut et ferez figurer la classe prédite et réelle des caractères.

3. Prétraitement-compression des données par ACP

(a) Effectuer une ACP du jeu `mnist` et analyser la série des valeurs propres. Combien de composantes doivent être conservées pour avoir plus de 95% de l'inertie.

(b) Appliquer à nouveau l'algorithme kNN mais ici vous utiliserez comme jeu initial la projection via ACP réalisée à la question précédente. Que constatez-vous ?

(c) Dresser la nouvelle matrice de confusion à l'issu de la classification précédente. Comparer ces résultats avec la matrice de la question 2. (d). Que peut-on dire ?