

# Analyse des données européennes

FALL

2024-12-31

## Contents

Les données analysées dans ce projet concernent différents indicateurs socio-économiques et environnementaux pour un ensemble de pays européens. Ces indicateurs offrent une vision globale de divers aspects comme la démographie, l'économie, l'énergie, l'emploi et l'environnement. Les analyses visent à explorer les structures sous-jacentes et les relations entre ces variables, ainsi qu'à regrouper les pays selon des caractéristiques communes.

### *Description des variables*

Les données contiennent 16 variables descriptives de 30 pays européens, reflétant des dimensions clés :

Population au 1er janvier : Nombre absolu d'habitants.

Population jeune (15-29 ans) : Pourcentage de jeunes dans la population totale.

Premières demandes d'asile : Nombre absolu de demandes.

Écart de rémunération entre les sexes : Pourcentage de différence de salaire horaire brut moyen entre hommes et femmes.

Salaire minimum : Montant en euros par mois.

Décrocheurs scolaires précoces : Pourcentage de la population âgée de 18 à 24 ans quittant prématurément le système scolaire.

Taux d'inflation : Variation en pourcentage par rapport à l'année précédente.

Taux de chômage : Pourcentage de la population active âgée de 15 à 74 ans.

Taux de chômage des jeunes : Pourcentage de la population active de moins de 25 ans.

PIB par habitant : Produit intérieur brut en euros par habitant.

Dette brute du gouvernement : Pourcentage de la dette brute par rapport au PIB.

Émissions de gaz à effet de serre : Quantité moyenne en tonnes par habitant.

Énergies renouvelables : Pourcentage dans la consommation finale brute d'énergie.

Prix de l'électricité : Montant en euros par MWh, incluant les taxes.

Dépendance aux importations d'énergie : Pourcentage de dépendance à l'énergie importée.

Taux de risque de pauvreté ou d'exclusion sociale : Pourcentage de la population à risque de pauvreté ou d'exclusion sociale.

L'objectif de cette analyse est d'explorer et de réduire la dimensionnalité des données grâce à une analyse en composantes principales (ACP), puis de grouper les pays selon leurs caractéristiques à l'aide de méthodes de classification. L'ACP permet de visualiser les similitudes entre pays et d'identifier les variables les plus importantes. Les méthodes de classification, notamment la classification ascendante hiérarchique (CAH)

et l'algorithme des centres mobiles (k-means), permettent d'interpréter les regroupements obtenus. Les résultats des classifications seront comparés afin de comprendre les proximités entre pays et leur cohérence.

Pour la préparation des données, une normalisation a été appliquée pour rendre les variables comparables. L'analyse inclut la création de matrices de dissimilarité, l'utilisation de la décomposition en valeurs propres pour l'ACP, et l'application des méthodes de classification sur les données normalisées. Les regroupements obtenus seront interprétés à travers l'étude des centres de gravité, des inerties et des plans factoriels. Enfin, une attention particulière sera portée à l'analyse des proximités des pays dans des zones spécifiques de l'espace factoriel, afin de mieux comprendre les similarités entre pays.

```
# Vérification et chargement des bibliothèques nécessaires
if (!require(ggplot2)) install.packages("ggplot2", dependencies = TRUE)
library(ggplot2)

# Chargement de TinyTeX si nécessaire (une seule fois)
if (!tinytex::is_tinytex()) {
  tinytex::install_tinytex()
}

euro_data <- read.csv("data/euro.csv", header = TRUE, sep = ";")

# Normalisation des données (Min-Max Scaling)
euro_data_normalized <- as.data.frame(lapply(euro_data[, -1], function(x) {
  (x - min(x, na.rm = TRUE)) / (max(x, na.rm = TRUE) - min(x, na.rm = TRUE))
})))
```

### Diagrammes à mmoustache

La distribution de la variable Population montre une grande variabilité entre les pays européens. Certains pays, comme l'Allemagne (84,3 millions), la France (68,1 millions) et l'Italie (58,9 millions), présentent des populations nettement supérieures à la majorité. Ces valeurs extrêmes contrastent avec des pays de petite taille comme Malte (0,54 million) et Chypre (0,92 million). La médiane, relativement basse, indique que la plupart des pays européens ont des populations bien inférieures à ces valeurs élevées.

Quant à la variable Youth Population (pourcentage de jeunes âgés de 15 à 29 ans), elle présente une répartition beaucoup plus homogène. Les pourcentages varient de 13,2 % (Bulgarie) à 21,0 % (Islande), sans aucune valeur aberrante. La médiane est bien centrée, reflétant une répartition équilibrée entre les pays. On observe que les pays nordiques, comme l'Islande (21,0 %) et la Norvège (18,7 %), possèdent des proportions relativement élevées de jeunes, tandis que des pays comme la Bulgarie se situent à l'extrémité inférieure de cette répartition.

```
# Supprimer la première colonne si elle contient des noms (optionnel)
euro_data <- euro_data[, -1]

selected_columns <- colnames(euro_data)[1:2]

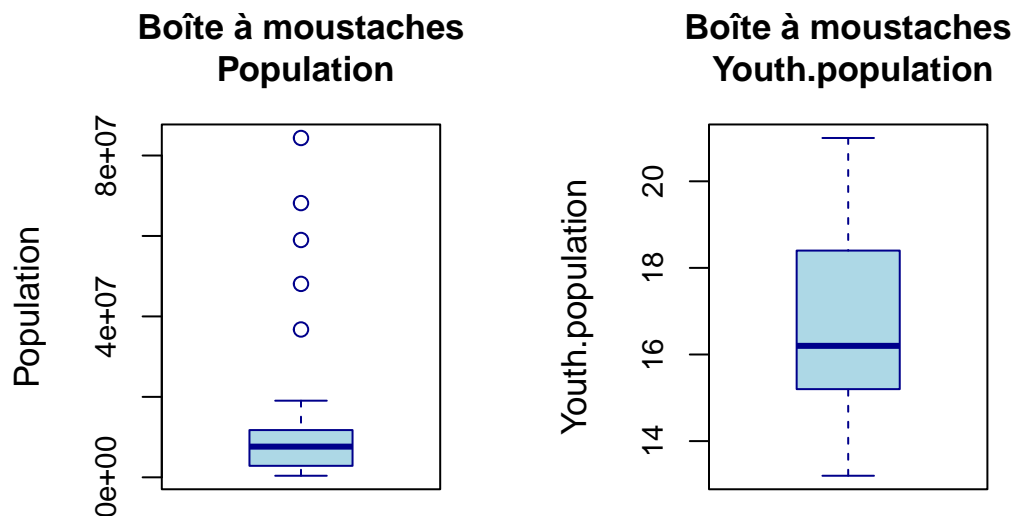
# Ajuster les marges et l'espacement
par(mfrow = c(1, 2), # Deux graphiques côte à côte
    oma = c(2, 2, 2, 2), # Marges extérieures : bas, gauche, haut, droite
    mar = c(5, 5, 4, 2)) # Marges intérieures : bas, gauche, haut, droite

# Boucle pour tracer les boîtes à moustaches
for (col_name in selected_columns) {
  boxplot(euro_data[[col_name]],
```

```

main = paste("Boîte à moustaches\n", col_name), # Titre sur deux lignes
ylab = col_name,
col = "lightblue",
border = "darkblue",
cex.main = 1.1, # Taille du titre
cex.lab = 1.1, # Taille des étiquettes
cex.axis = 0.9) # Taille des axes
}

```



La distribution de la variable First Time Asylum Applicants (premières demandes d'asile) montre une forte asymétrie, avec plusieurs valeurs extrêmes très élevées. Ces valeurs, représentées par des points au-dessus des moustaches, indiquent que quelques pays reçoivent un nombre disproportionné de premières demandes d'asile par rapport à la majorité. La médiane est très basse, ce qui reflète que la plupart des pays ont un nombre relativement faible de demandes.

La variable Gender Pay Gap (écart de rémunération entre les sexes) présente une distribution symétrique et homogène. La médiane est bien centrée, et la boîte à moustaches indique que les valeurs sont concentrées dans une plage relativement étroite. Aucun outlier n'est visible, ce qui suggère que les écarts salariaux entre les sexes sont globalement similaires parmi les pays analysés. Interprétation

Les données brutes confirment les observations issues des diagrammes à moustaches. Concernant les First Time Asylum Applicants, on observe de fortes valeurs pour des pays comme l'Allemagne (0,32 million) et la France (0,14 million), qui reçoivent un grand nombre de demandes. À l'inverse, des pays comme la Slovaquie (0,0003 million) et la Hongrie (0,00003 million) présentent des valeurs bien en dessous de la moyenne, reflétant une disparité importante.

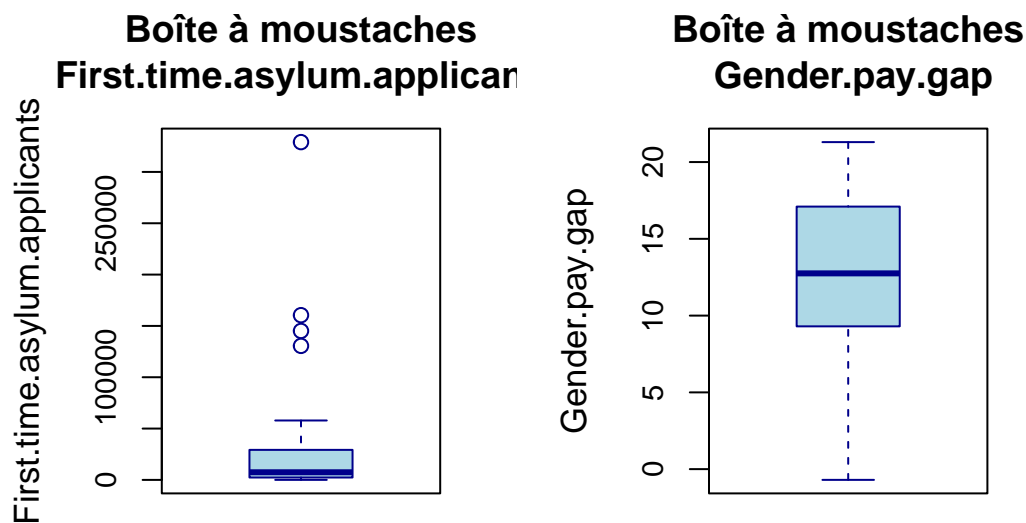
Pour la variable Gender Pay Gap, les données confirment l'homogénéité observée. Toutefois, on peut noter des différences significatives entre certains pays. Par exemple, l'Autriche (18,4) et la Suisse (17,9) affichent

des valeurs élevées, tandis que le Luxembourg présente une valeur négative (-0,7), indiquant une situation inverse inhabituelle.

```
selected_columns <- colnames(euro_data)[3:4]

par(mfrow = c(1, 2),
    oma = c(2, 2, 2, 2),
    mar = c(5, 5, 4, 2))

for (col_name in selected_columns) {
  boxplot(euro_data[[col_name]],
    main = paste("Boîte à moustaches\n", col_name),
    ylab = col_name,
    col = "lightblue",
    border = "darkblue",
    cex.main = 1.2,
    cex.lab = 1.1,
    cex.axis = 0.9)
}
```



La distribution de la variable Minimum Wage (salaire minimum) montre une dispersion modérée avec quelques valeurs basses qui se démarquent, visibles sous forme de moustaches allongées. La médiane est située dans la moitié inférieure de la boîte, indiquant qu'une majorité des pays ont un salaire minimum inférieur à la moyenne globale. Aucune valeur aberrante significative n'est visible.

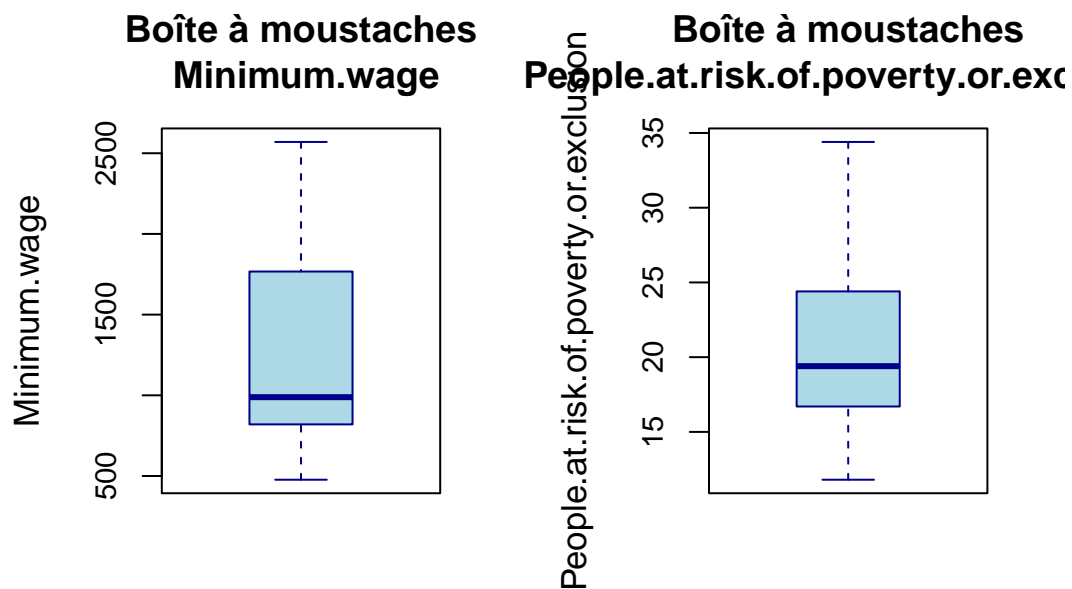
La variable People at Risk of Poverty or Exclusion (personnes à risque de pauvreté ou d'exclusion sociale)

présente une distribution assez homogène. La médiane est centrée, ce qui indique une répartition symétrique des données entre les différents pays. La boîte montre que la majorité des pays ont des valeurs proches les unes des autres, reflétant une cohérence entre les nations sur cet indicateur.

```
selected_columns <- colnames(euro_data)[5:6]

par(mfrow = c(1, 2),
    oma = c(2, 2, 2, 2),
    mar = c(5, 5, 4, 2))

for (col_name in selected_columns) {
  boxplot(euro_data[[col_name]],
    main = paste("Boîte à moustaches\n", col_name),
    ylab = col_name,
    col = "lightblue",
    border = "darkblue",
    cex.main = 1.2,
    cex.lab = 1.1,
    cex.axis = 0.9)
}
```



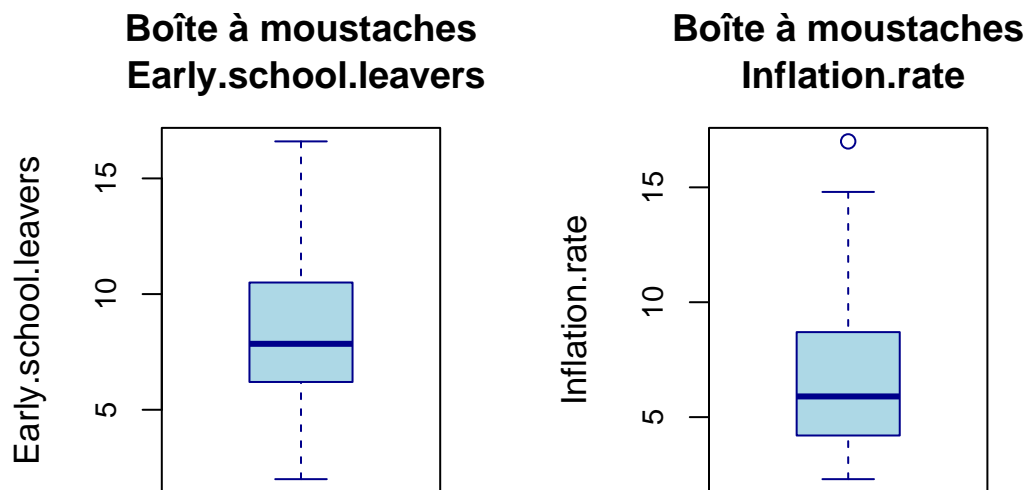
La distribution de la variable Early School Leavers (décrocheurs scolaires précoces) montre une répartition assez homogène, sans valeurs aberrantes significatives. La médiane est centrée dans la boîte, indiquant une répartition équilibrée entre les pays. La plupart des pays ont des taux de décrochage scolaire compris entre les moustaches, avec peu de variabilité.

La variable Inflation Rate (taux d'inflation) présente une valeur aberrante, visible sous forme de point au-dessus des moustaches. Cela indique qu'un ou quelques pays ont un taux d'inflation nettement supérieur à la majorité. La boîte est asymétrique, suggérant une distribution légèrement biaisée vers les valeurs inférieures.

```
selected_columns <- colnames(euro_data)[7:8]

par(mfrow = c(1, 2),
    oma = c(2, 2, 2, 2),
    mar = c(5, 5, 4, 2))

for (col_name in selected_columns) {
  boxplot(euro_data[[col_name]],
    main = paste("Boîte à moustaches\n", col_name),
    ylab = col_name,
    col = "lightblue",
    border = "darkblue",
    cex.main = 1.2,
    cex.lab = 1.1,
    cex.axis = 0.9)
}
```



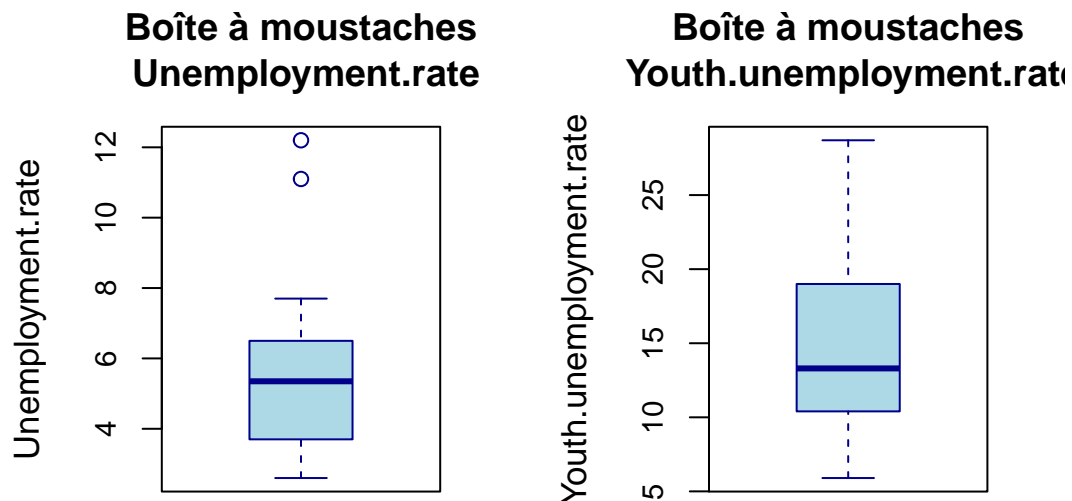
La variable Unemployment Rate (taux de chômage) montre une distribution globalement homogène, bien que deux valeurs aberrantes soient visibles au-dessus des moustaches. Ces valeurs indiquent que certains pays connaissent un taux de chômage nettement plus élevé que la moyenne. La médiane, légèrement au-dessus du centre de la boîte, suggère une légère asymétrie vers des valeurs inférieures.

Pour la variable Youth Unemployment Rate (taux de chômage des jeunes), la distribution est plus dispersée, sans valeurs aberrantes. La boîte et les moustaches reflètent une variabilité importante entre les pays, traduisant des disparités régionales marquées. La médiane, située un peu en dessous du centre, indique une distribution légèrement asymétrique.

```
selected_columns <- colnames(euro_data)[9:10]

par(mfrow = c(1, 2),
    oma = c(2, 2, 2, 2),
    mar = c(5, 5, 4, 2))

for (col_name in selected_columns) {
  boxplot(euro_data[[col_name]],
    main = paste("Boîte à moustaches\n", col_name),
    ylab = col_name,
    col = "lightblue",
    border = "darkblue",
    cex.main = 1.2,
    cex.lab = 1.1,
    cex.axis = 0.9)
}
```



La variable GDP per Capita (PIB par habitant) présente une distribution avec une valeur aberrante notable correspondant à un pays dont le PIB par habitant est extrêmement élevé par rapport aux autres. La majorité des pays se situent dans une plage relativement restreinte, comme le montre la boîte. La médiane

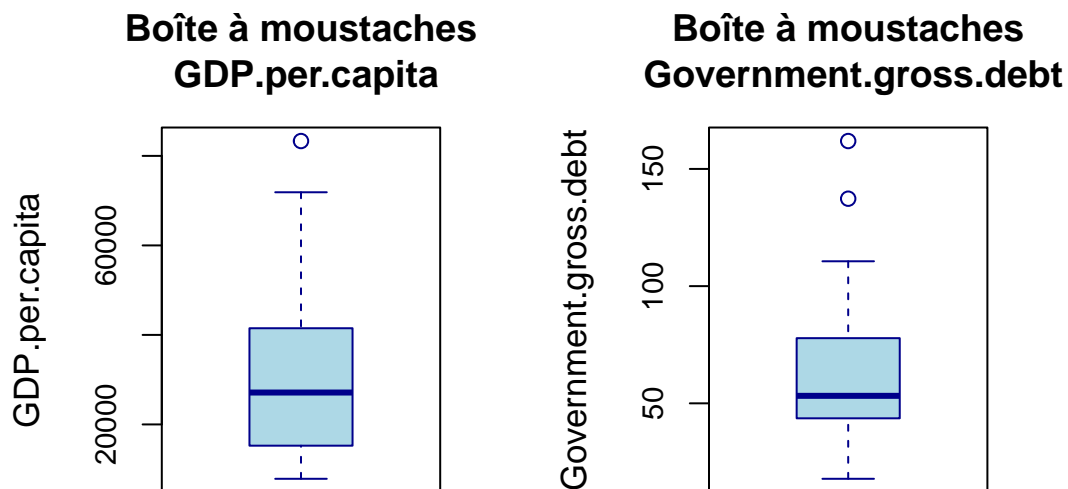
est bien centrée, indiquant une répartition relativement symétrique parmi les pays exclus des valeurs aberrantes.

La variable Government Gross Debt (dette brute du gouvernement en pourcentage du PIB) montre deux valeurs aberrantes au-dessus des moustaches, indiquant que certains pays ont des niveaux de dette exceptionnellement élevés. La boîte indique une distribution concentrée pour la majorité des pays avec une médiane légèrement en dessous du centre, suggérant une légère asymétrie vers les valeurs plus faibles.

```
selected_columns <- colnames(euro_data)[11:12]

par(mfrow = c(1, 2),
    oma = c(2, 2, 2, 2),
    mar = c(5, 5, 4, 2))

for (col_name in selected_columns) {
  boxplot(euro_data[[col_name]],
    main = paste("Boîte à moustaches\n", col_name),
    ylab = col_name,
    col = "lightblue",
    border = "darkblue",
    cex.main = 1.2,
    cex.lab = 1.1,
    cex.axis = 0.9)
}
```



La variable Greenhouse Gas Emissions (émissions de gaz à effet de serre en tonnes par habitant) présente



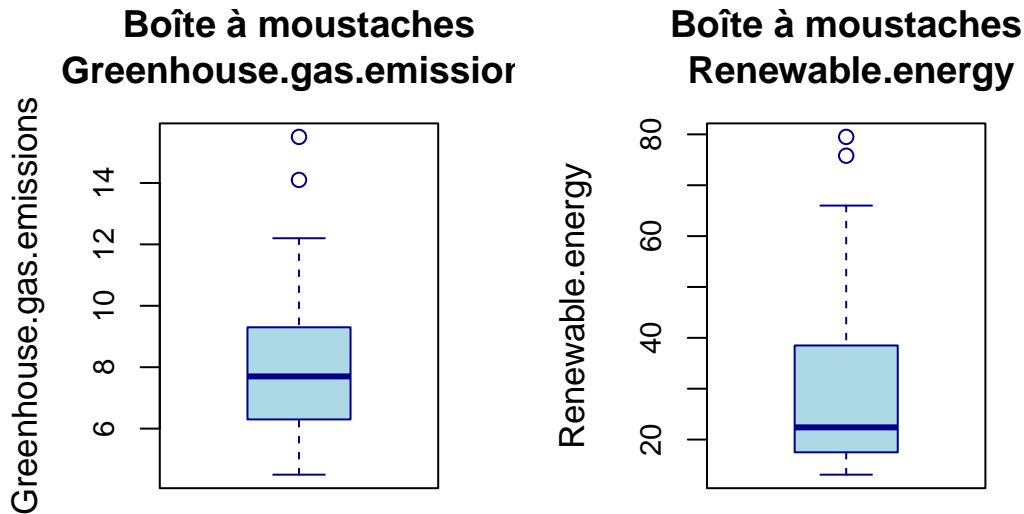
deux valeurs aberrantes correspondant à des pays avec des émissions particulièrement élevées. La médiane est légèrement inférieure au centre de la boîte, indiquant une légère asymétrie vers les valeurs plus faibles. Cela suggère que la majorité des pays ont des émissions modérées, tandis qu'un petit nombre a des émissions bien au-dessus de la moyenne.

La variable Renewable Energy (part des énergies renouvelables dans la consommation énergétique) montre également deux valeurs aberrantes pour des pays ayant une part particulièrement élevée d'énergies renouvelables. La distribution est asymétrique, avec une majorité des pays ayant des parts plus faibles en énergies renouvelables. La médiane est proche du bas de la boîte, indiquant une concentration de valeurs relativement faibles pour la plupart des

```
selected_columns <- colnames(euro_data)[13:14]

par(mfrow = c(1, 2),
    oma = c(2, 2, 2, 2),
    mar = c(5, 5, 4, 2))

for (col_name in selected_columns) {
  boxplot(euro_data[[col_name]],
    main = paste("Boîte à moustaches\n", col_name),
    ylab = col_name,
    col = "lightblue",
    border = "darkblue",
    cex.main = 1.2,
    cex.lab = 1.1,
    cex.axis = 0.9)
}
```



La variable Electricity Prices (prix de l'électricité en euros par MWh, taxes incluses) montre une distribution relativement homogène, sans valeurs aberrantes visibles. La boîte à moustaches illustre une concentration modérée autour de la médiane, indiquant que les prix sont globalement similaires entre les pays européens. Cependant, la largeur de la boîte et des moustaches traduit une certaine variabilité, reflétant des écarts notables pour certains pays. Par exemple, des pays comme l'Allemagne et le Danemark ont historiquement des prix de l'électricité plus élevés.

La variable Energy Imports Dependency (dépendance aux importations d'énergie, exprimée en pourcentage) présente une dispersion importante entre les pays. La médiane, située dans le haut de la boîte, indique que la majorité des pays ont une dépendance modérée à élevée. La variabilité marquée, visible à travers les moustaches longues, reflète des écarts significatifs : certains pays, comme Malte (99,0%) et Chypre (92,0%), affichent une forte dépendance énergétique, tandis que d'autres, comme l'Islande (15,2%) et la Norvège (15,2%), montrent une indépendance énergétique notable. Cette disparité est probablement liée à la disponibilité des ressources énergétiques, à leur gestion efficace et à l'intégration des énergies renouvelables.

```
selected_columns <- colnames(euro_data)[15:16]

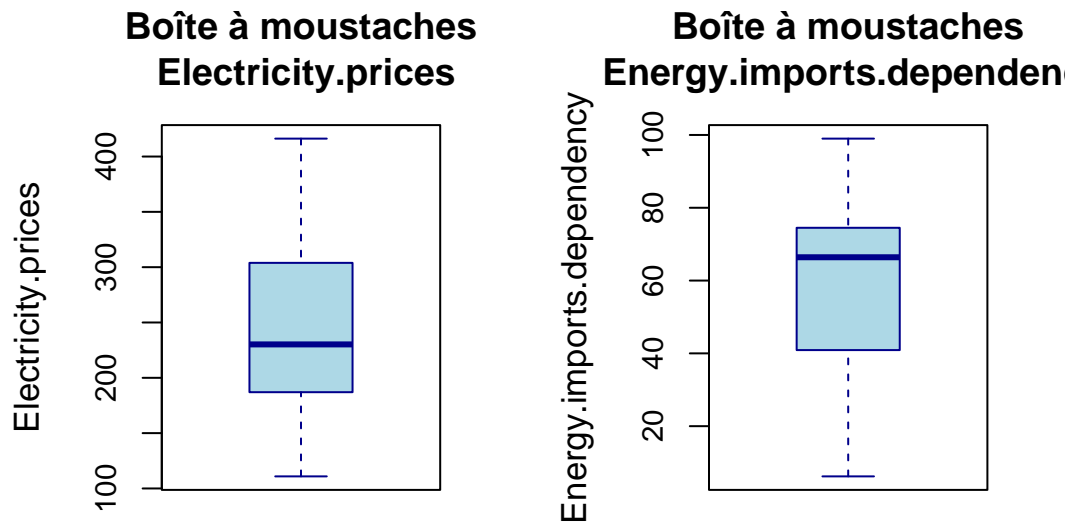
par(mfrow = c(1, 2),
    oma = c(2, 2, 2, 2),
    mar = c(5, 5, 4, 2))

for (col_name in selected_columns) {
  boxplot(euro_data[[col_name]],
    main = paste("Boîte à moustaches\n", col_name),
    ylab = col_name,
```

```

col = "lightblue",
border = "darkblue",
cex.main = 1.2,
cex.lab = 1.1,
cex.axis = 0.9)
}

```



Pour appoter plus de visibilité nous allons remplacer les colones pour des colones numérique en se servant de cette légense Légende des colones (à remplir manuellement si besoin) :

Légende des colones :

- [1]: Population
- [2]: Youth.population
- [3]: First.time.asylum.applicants
- [4]: Gender.pay.gap
- [5]: Minimum.wage
- [6]: People.at.risk.of.poverty.or.exclusion
- [7]: Early.school.leavers
- [8]: Inflation.rate
- [9]: Unemployment.rate
- [10]: Youth.unemployment.rate
- [11]: GDP.per.capita
- [12]: Government.gross.debt
- [13]: Greenhouse.gas.emissions
- [14]: Renewable.energy
- [15]: Electricity.prices
- [16]: Energy.imports.dependency

La matrice de variance-covariance VV calcule la dispersion des variables ainsi que les relations linéaires entre elles. Les valeurs sur la diagonale principale représentent la variance de chaque variable. Par exemple, la variance de Population est très élevée (4.653257e+14), reflétant une grande dispersion entre les pays, comme cela a été observé dans les diagrammes à moustaches. En revanche, la variance de Gender Pay Gap est beaucoup plus faible (25.6190), confirmant l'homogénéité relative notée précédemment, bien qu'elle reste significative.

Une covariance négative entre Renewable Energy et Energy Imports Dependency (−290.3213) reflète une relation inverse plus marquée : les pays avec une part plus élevée d'énergies renouvelables, comme l'Islande et la Norvège, sont généralement moins dépendants des importations d'énergie.

La matrice V enrichit l'analyse exploratoire initiale en fournissant une quantification précise des dispersions et interactions entre variables. Ces résultats constituent une base solide pour des analyses plus avancées, comme l'analyse en composantes principales (ACP), tout en préparant la transition vers l'étude des relations normalisées via la matrice de corrélation.

```
#3
euro_data_centered <- scale(euro_data, center = TRUE, scale = FALSE)

n <- nrow(euro_data)
V <- (t(euro_data_centered) %*% euro_data_centered) / (n - 1)

colnames(V) <- paste0("[", seq_len(ncol(V)), "]")
rownames(V) <- colnames(euro_data)

V <- round(V, 4)

print("Matrice de variance-covariance (indices entre crochets pour les colonnes) :")
```

```
## [1] "Matrice de variance-covariance (indices entre crochets pour les colonnes) :"
```

```
cat("\nPartie 1 : Colonnes [1] à [8]\n")
```

```
##
## Partie 1 : Colonnes [1] à [8]
```

```
print(V[, 1:8])
```

```
##                                [1]                [2]
## Population                    4.653257e+14 -6839267.0638
## Youth.population              -6.839267e+06      3.2609
## First.time.asylum.applicants 1.365717e+12 -15924.8017
## Gender.pay.gap                 -6.195038e+06     -2.2307
## Minimum.wage                   NA                NA
## People.at.risk.of.poverty.or.exclusion NA                NA
## Early.school.leavers          1.621818e+07      1.5759
## Inflation.rate                 -6.534739e+06     -2.6463
## Unemployment.rate              8.017485e+06     -0.8741
## Youth.unemployment.rate        1.445936e+07     -2.2734
```

```
## GDP.per.capita -2.857962e+10 23946.7345
## Government.gross.debt 2.994623e+08 -2.1453
## Greenhouse.gas.emissions -8.092486e+06 1.8637
## Renewable.energy -1.066077e+08 10.1029
## Electricity.prices 7.486738e+08 -0.8086
## Energy.imports.dependency 6.282654e+07 4.0745
## [3] [4] [5] [6]
## Population 1.365717e+12 -6195038.4282 NA NA
## Youth.population -1.592480e+04 -2.2307 NA NA
## First.time.asylum.applicants 4.950971e+09 23416.4569 NA NA
## Gender.pay.gap 2.341646e+04 25.6190 NA NA
## Minimum.wage NA NA NA NA
## People.at.risk.of.poverty.or.exclusion NA NA NA NA
## Early.school.leavers 6.578850e+04 -1.1510 NA NA
## Inflation.rate -5.499202e+04 5.9963 NA NA
## Unemployment.rate 3.117897e+04 -0.8288 NA NA
## Youth.unemployment.rate 1.909011e+04 -9.0915 NA NA
## GDP.per.capita 3.054725e+07 -23296.9276 NA NA
## Government.gross.debt 9.319273e+05 -30.4644 NA NA
## Greenhouse.gas.emissions -1.295407e+04 -3.5834 NA NA
## Renewable.energy -2.707843e+05 23.8361 NA NA
## Electricity.prices 2.853515e+06 -0.5714 NA NA
## Energy.imports.dependency 3.598142e+05 -34.9379 NA NA
## [7] [8]
## Population 16218178.4517 -6534739.2138
## Youth.population 1.5759 -2.6463
## First.time.asylum.applicants 65788.5000 -54992.0172
## Gender.pay.gap -1.1510 5.9963
## Minimum.wage NA NA
## People.at.risk.of.poverty.or.exclusion NA NA
## Early.school.leavers 12.5517 0.7159
## Inflation.rate 0.7159 12.1161
## Unemployment.rate 0.0483 -2.5167
## Youth.unemployment.rate 0.6197 -3.7521
## GDP.per.capita -1667.7931 -40170.7103
## Government.gross.debt 6.2831 -21.8946
## Greenhouse.gas.emissions 0.1055 -0.4309
## Renewable.energy 19.0807 -9.6309
## Electricity.prices -44.9041 -99.8614
## Energy.imports.dependency -27.2752 -28.8008
```

```
cat("\nPartie 2 : Colonnes [9] à [16]\n")
```

```
##
## Partie 2 : Colonnes [9] à [16]
```

```
print(V[, 9:16])
```

```
## [9] [10] [11]
## Population 8017484.9552 14459359.1397 -2.857962e+10
## Youth.population -0.8741 -2.2734 2.394673e+04
## First.time.asylum.applicants 31178.9655 19090.1121 3.054725e+07
## Gender.pay.gap -0.8288 -9.0915 -2.329693e+04
```

## Minimum.wage	NA	NA	NA
## People.at.risk.of.poverty.or.exclusion	NA	NA	NA
## Early.school.leavers	0.0483	0.6197	-1.667793e+03
## Inflation.rate	-2.5167	-3.7521	-4.017071e+04
## Unemployment.rate	5.0494	11.5647	-8.682110e+03
## Youth.unemployment.rate	11.5647	34.5511	-2.704445e+04
## GDP.per.capita	-8682.1103	-27044.4517	3.983704e+08
## Government.gross.debt	43.5801	103.2097	-1.581849e+05
## Greenhouse.gas.emissions	-1.9539	-3.9530	2.158401e+04
## Renewable.energy	-0.8711	-14.7666	1.228200e+05
## Electricity.prices	18.5154	-30.1509	2.644578e+05
## Energy.imports.dependency	7.1974	15.1315	5.860834e+04
##	[12]	[13]	
## Population	2.994623e+08	-8092485.6885	
## Youth.population	-2.145300e+00	1.8637	
## First.time.asylum.applicants	9.319273e+05	-12954.0690	
## Gender.pay.gap	-3.046440e+01	-3.5834	
## Minimum.wage	NA	NA	
## People.at.risk.of.poverty.or.exclusion	NA	NA	
## Early.school.leavers	6.283100e+00	0.1055	
## Inflation.rate	-2.189460e+01	-0.4309	
## Unemployment.rate	4.358010e+01	-1.9539	
## Youth.unemployment.rate	1.032097e+02	-3.9530	
## GDP.per.capita	-1.581849e+05	21584.0069	
## Government.gross.debt	1.240830e+03	-11.3727	
## Greenhouse.gas.emissions	-1.137270e+01	7.0331	
## Renewable.energy	-1.327777e+02	-2.7331	
## Electricity.prices	6.659872e+02	19.0603	
## Energy.imports.dependency	2.703910e+02	-3.6866	
##	[14]	[15]	
## Population	-1.066077e+08	7.486738e+08	
## Youth.population	1.010290e+01	-8.086000e-01	
## First.time.asylum.applicants	-2.707843e+05	2.853515e+06	
## Gender.pay.gap	2.383610e+01	-5.714000e-01	
## Minimum.wage	NA	NA	
## People.at.risk.of.poverty.or.exclusion	NA	NA	
## Early.school.leavers	1.908070e+01	-4.490410e+01	
## Inflation.rate	-9.630900e+00	-9.986140e+01	
## Unemployment.rate	-8.711000e-01	1.851540e+01	
## Youth.unemployment.rate	-1.476660e+01	-3.015090e+01	
## GDP.per.capita	1.228200e+05	2.644578e+05	
## Government.gross.debt	-1.327777e+02	6.659872e+02	
## Greenhouse.gas.emissions	-2.733100e+00	1.906030e+01	
## Renewable.energy	3.593812e+02	-4.262927e+02	
## Electricity.prices	-4.262927e+02	6.935113e+03	
## Energy.imports.dependency	-2.903213e+02	7.416931e+02	
##	[16]		
## Population	62826536.5937		
## Youth.population	4.0745		
## First.time.asylum.applicants	359814.2500		
## Gender.pay.gap	-34.9379		
## Minimum.wage	NA		
## People.at.risk.of.poverty.or.exclusion	NA		
## Early.school.leavers	-27.2752		

```
## Inflation.rate -28.8008
## Unemployment.rate 7.1974
## Youth.unemployment.rate 15.1315
## GDP.per.capita 58608.3414
## Government.gross.debt 270.3910
## Greenhouse.gas.emissions -3.6866
## Renewable.energy -290.3213
## Electricity.prices 741.6931
## Energy.imports.dependency 592.5486
```

La heatmap de la matrice de variance-covariance illustre les relations entre les variables à travers des intensités de couleurs. Les zones rouges traduisent des covariances fortes, tandis que les zones jaunes indiquent des relations plus faibles. Les zones blanches signalent des données manquantes ou nulles.

Par exemple, des relations marquées apparaissent entre Population et Youth Population, ainsi qu'entre Renewable Energy et Energy Imports Dependency, confirmant les observations sur la dépendance énergétique. Cette visualisation synthétique facilite l'identification des interactions clés et des lacunes dans les données.

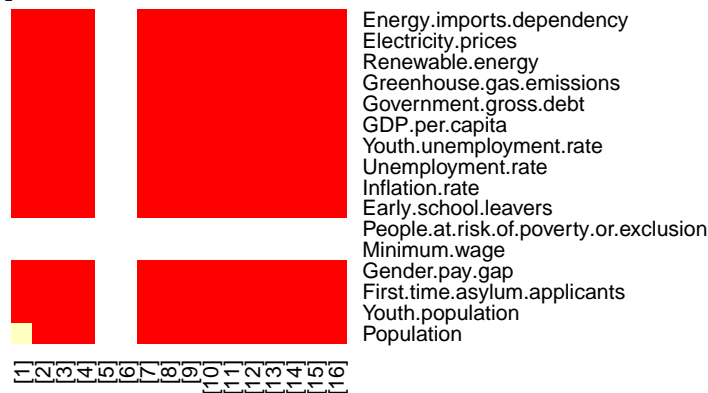
*#3*

```
V_rounded <- round(V, 2)
```

*# Étape 3 : Créer la heatmap avec des noms de variables lisibles*

```
heatmap(as.matrix(V_rounded),
        main = "Heatmap variance-covariance",
        Colv = NA, Rowv = NA, # Désactive le clustering
        scale = "none", # Pas de normalisation supplémentaire
        col = heat.colors(10), # Palette de couleurs
        margins = c(15, 15), # Augmenter les marges pour les étiquettes
        cexRow = 0.8, cexCol = 0.8) # Réduire la taille des étiquettes pour plus de lisibilité
```

## Heatmap variance-covariance



La matrice de corrélation offre une vision normalisée des relations linéaires entre les variables, avec des coefficients compris entre -1 et 1. Une corrélation positive modérée est observée entre Population et Youth Population ( $r=0.87$ ), indiquant que les pays avec une population élevée tendent à avoir une proportion importante de jeunes.

Une relation négative significative est également visible entre Renewable Energy et Energy Imports Dependency ( $r=-0.54$ ), confirmant que les pays intégrant davantage d'énergies renouvelables sont généralement moins dépendants des importations énergétiques.

De plus, une corrélation très forte ( $r=0.87$ ) entre Youth Unemployment Rate et Unemployment Rate montre une relation directe entre ces deux indicateurs. En revanche, certaines variables comme Gender Pay Gap et Minimum Wage ( $r=-0.05$ ) affichent une corrélation quasi nulle, suggérant une absence de relation linéaire significative. Cette matrice met ainsi en évidence les liens les plus marqués tout en soulignant les variables peu ou pas reliées entre elles.

```
# 4

correlation_matrix <- cor(euro_data, use = "complete.obs")

# Arrondir les valeurs à deux décimales pour une meilleure lisibilité
correlation_matrix_rounded <- round(correlation_matrix, 2)

# Ajouter des indices numériques pour les colonnes et les lignes
colnames(correlation_matrix_rounded) <- paste0("[", seq_len(ncol(correlation_matrix_rounded)), "]")
rownames(correlation_matrix_rounded) <- colnames(correlation_matrix)

# Afficher la matrice avec indices numériques
```



```
print("Matrice de corrélation (indices pour les colonnes) :")
```

```
## [1] "Matrice de corrélation (indices pour les colonnes) :"
```

```
print(correlation_matrix_rounded)
```

```
##           [1]  [2]  [3]  [4]  [5]  [6]
## Population      1.00 -0.11  0.90 -0.06  0.25  0.13
## Youth.population -0.11  1.00 -0.07 -0.25  0.63 -0.43
## First.time.asylum.applicants  0.90 -0.07  1.00  0.07  0.34  0.16
## Gender.pay.gap    -0.06 -0.25  0.07  1.00 -0.30 -0.14
## Minimum.wage      0.25  0.63  0.34 -0.30  1.00 -0.35
## People.at.risk.of.poverty.or.exclusion  0.13 -0.43  0.16 -0.14 -0.35  1.00
## Early.school.leavers  0.32  0.03  0.36 -0.01 -0.10  0.43
## Inflation.rate    -0.11 -0.52 -0.25  0.43 -0.63 -0.04
## Unemployment.rate  0.14 -0.15  0.19 -0.05 -0.09  0.43
## Youth.unemployment.rate  0.08 -0.15  0.03 -0.29 -0.19  0.45
## GDP.per.capita     0.01  0.71  0.10 -0.37  0.90 -0.32
## Government.gross.debt  0.41 -0.06  0.38 -0.12  0.03  0.04
## Greenhouse.gas.emissions -0.13  0.30 -0.06 -0.24  0.51 -0.28
## Renewable.energy   -0.22 -0.18 -0.16  0.43 -0.13  0.09
## Electricity.prices  0.40  0.23  0.48 -0.04  0.43 -0.18
## Energy.imports.dependency  0.04  0.49  0.16 -0.40  0.37 -0.24
##           [7]  [8]  [9] [10] [11] [12]
## Population      0.32 -0.11  0.14  0.08  0.01  0.41
## Youth.population  0.03 -0.52 -0.15 -0.15  0.71 -0.06
## First.time.asylum.applicants  0.36 -0.25  0.19  0.03  0.10  0.38
## Gender.pay.gap    -0.01  0.43 -0.05 -0.29 -0.37 -0.12
## Minimum.wage     -0.10 -0.63 -0.09 -0.19  0.90  0.03
## People.at.risk.of.poverty.or.exclusion  0.43 -0.04  0.43  0.45 -0.32  0.04
## Early.school.leavers  1.00  0.01  0.12  0.14 -0.13 -0.02
## Inflation.rate    0.01  1.00 -0.38 -0.25 -0.58 -0.29
## Unemployment.rate  0.12 -0.38  1.00  0.87 -0.12  0.62
## Youth.unemployment.rate  0.14 -0.25  0.87  1.00 -0.15  0.57
## GDP.per.capita    -0.13 -0.58 -0.12 -0.15  1.00 -0.12
## Government.gross.debt -0.02 -0.29  0.62  0.57 -0.12  1.00
## Greenhouse.gas.emissions -0.28 -0.16 -0.28 -0.20  0.61 -0.34
## Renewable.energy   0.13 -0.07  0.28  0.06 -0.17 -0.11
## Electricity.prices  0.02 -0.38  0.05 -0.12  0.36  0.27
## Energy.imports.dependency -0.15 -0.42  0.09  0.09  0.42  0.37
##           [13] [14] [15] [16]
## Population      -0.13 -0.22  0.40  0.04
## Youth.population  0.30 -0.18  0.23  0.49
## First.time.asylum.applicants -0.06 -0.16  0.48  0.16
## Gender.pay.gap    -0.24  0.43 -0.04 -0.40
## Minimum.wage      0.51 -0.13  0.43  0.37
## People.at.risk.of.poverty.or.exclusion -0.28  0.09 -0.18 -0.24
## Early.school.leavers -0.28  0.13  0.02 -0.15
## Inflation.rate    -0.16 -0.07 -0.38 -0.42
## Unemployment.rate -0.28  0.28  0.05  0.09
## Youth.unemployment.rate -0.20  0.06 -0.12  0.09
## GDP.per.capita     0.61 -0.17  0.36  0.42
```

```
## Government.gross.debt          -0.34 -0.11  0.27  0.37
## Greenhouse.gas.emissions       1.00 -0.28  0.25  0.08
## Renewable.energy              -0.28  1.00  0.00 -0.54
## Electricity.prices             0.25  0.00  1.00  0.19
## Energy.imports.dependency      0.08 -0.54  0.19  1.00
```

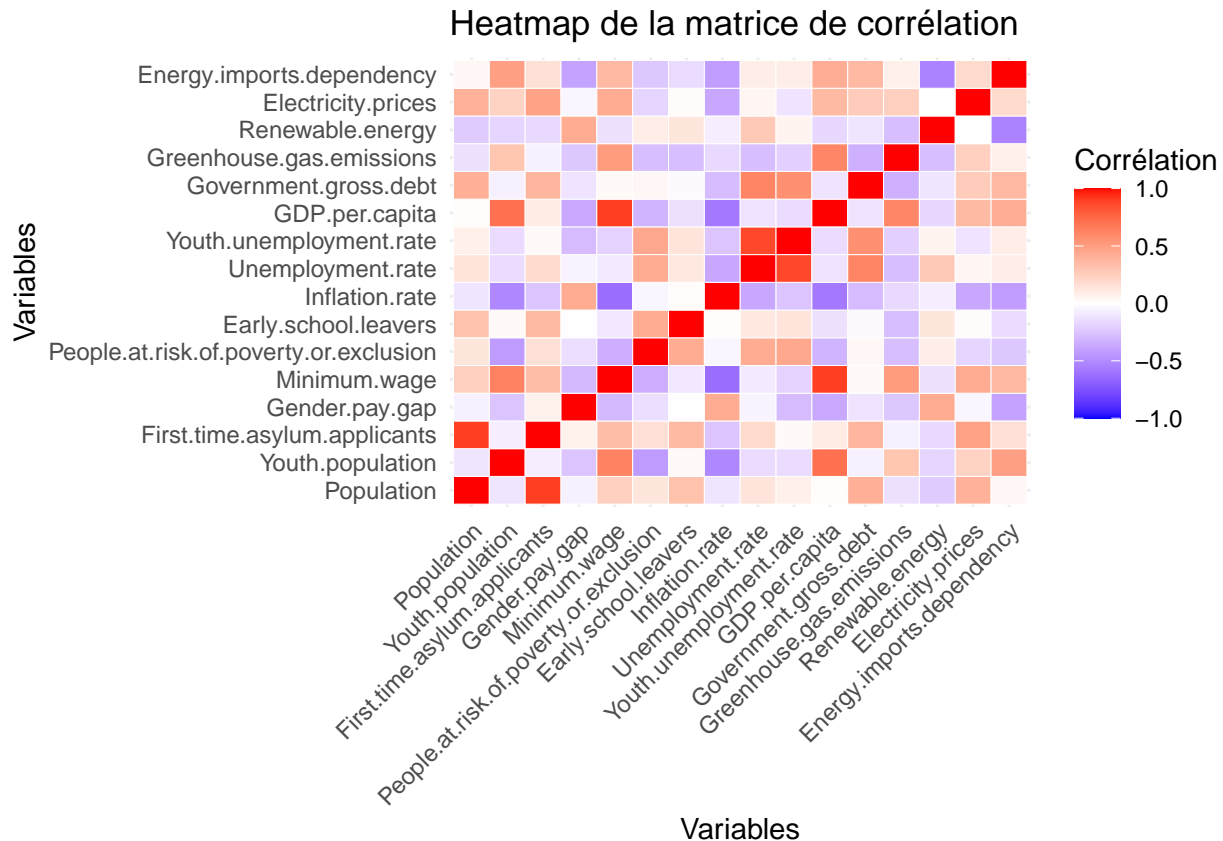
```
# Transformer la matrice de corrélation en format long
correlation_long <- reshape2::melt(round(correlation_matrix, 2))
```

La heatmap de la matrice de corrélation illustre les relations linéaires entre les différentes variables. Les couleurs rouges indiquent des corrélations positives élevées, tandis que les teintes bleues traduisent des corrélations négatives significatives. Les zones blanches ou pâles, quant à elles, reflètent des relations faibles ou inexistantes.

Par exemple, on remarque une forte corrélation positive entre Population et Youth Population, confirmant que les pays avec une population globale importante ont également une proportion significative de jeunes. Une corrélation négative marquée est visible entre Renewable Energy et Energy Imports Dependency, mettant en évidence l'effet des énergies renouvelables sur la réduction de la dépendance énergétique. À l'inverse, des variables comme Gender Pay Gap et Minimum Wage montrent peu ou pas de corrélation, traduisant l'absence de relation linéaire entre elles.

En somme, ce graphique offre une visualisation claire des relations fortes et faibles entre les variables, facilitant l'identification des interactions les plus significatives.

```
# Heatmap de la matrice de corrélation
ggplot(correlation_long, aes(Var1, Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                      midpoint = 0, limit = c(-1, 1),
                      name = "Corrélation") +
  theme_minimal() +
  labs(title = "Heatmap de la matrice de corrélation",
       x = "Variables",
       y = "Variables") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Les composantes principales forment une base orthonormée, ce qui est prouvé numériquement. Le produit scalaire des vecteurs propres (matrice de rotation transposée multipliée par elle-même) donne une matrice diagonale avec des valeurs de 1 sur la diagonale et 0 ailleurs, confirmant leur orthogonalité. Les normes des vecteurs propres, calculées comme la racine carrée de la somme des carrés des coefficients, sont toutes égales à 1, prouvant qu'ils sont normalisés.

Le paramètre `center = TRUE` recentre chaque variable en soustrayant sa moyenne, garantissant que les composantes principales sont calculées par rapport à un centre des données égal à zéro. Le paramètre `scale = TRUE` met chaque variable à l'échelle en divisant par son écart type, standardisant ainsi les variables pour qu'elles aient toutes une variance de 1, ce qui est crucial lorsque les variables ont des échelles différentes.

Sans `center`, les composantes principales seraient biaisées par des variables aux moyennes élevées, faussant leur interprétation. Sans `scale`, les variables avec des échelles ou des variances élevées domineraient les calculs, influençant de manière disproportionnée les composantes principales. En combinant `center` et `scale`, chaque variable contribue de manière équitable à la définition des composantes principales.

```
# 5

euro_data_replace_na <- apply(euro_data, 2, function(x) ifelse(is.na(x), mean(x, na.rm = TRUE), x))

# Calculer les composantes principales
res <- prcomp(euro_data_replace_na, scale = TRUE, center = TRUE)

# Créer un fichier PDF pour les résultats
pdf("resultats_prcomp.pdf", width = 10, height = 8)

# Vérification de l'orthogonalité et des normes
```

```
cat("Produit scalaire des vecteurs propres (orthogonalité) :\n")
```

```
## Produit scalaire des vecteurs propres (orthogonalité) :
```

```
orthogonality_check <- t(res$rotation) %*% res$rotation
print(round(orthogonality_check, 2))
```

```
##      PC1 PC2 PC3 PC4 PC5 PC6 PC7 PC8 PC9 PC10 PC11 PC12 PC13 PC14 PC15 PC16
## PC1    1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## PC2    0  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## PC3    0  0  1  0  0  0  0  0  0  0  0  0  0  0  0  0
## PC4    0  0  0  1  0  0  0  0  0  0  0  0  0  0  0  0
## PC5    0  0  0  0  1  0  0  0  0  0  0  0  0  0  0  0
## PC6    0  0  0  0  0  1  0  0  0  0  0  0  0  0  0  0
## PC7    0  0  0  0  0  0  1  0  0  0  0  0  0  0  0  0
## PC8    0  0  0  0  0  0  0  1  0  0  0  0  0  0  0  0
## PC9    0  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0
## PC10   0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  0
## PC11   0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0
## PC12   0  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0
## PC13   0  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0
## PC14   0  0  0  0  0  0  0  0  0  0  0  0  0  1  0  0
## PC15   0  0  0  0  0  0  0  0  0  0  0  0  0  0  1  0
## PC16   0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1
```

```
cat("Normes des vecteurs propres (1 attendu) :\n")
```

```
## Normes des vecteurs propres (1 attendu) :
```

```
norms <- apply(res$rotation, 2, function(col) sqrt(sum(col^2)))
print(round(norms, 2))
```

```
## PC1 PC2 PC3 PC4 PC5 PC6 PC7 PC8 PC9 PC10 PC11 PC12 PC13 PC14 PC15 PC16
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
```

```
# Afficher les premières coordonnées dans la nouvelle base
```

```
cat("Premières coordonnées des observations :\n")
```

```
## Premières coordonnées des observations :
```

```
print(head(res$x, n = 6))
```

```
##      PC1      PC2      PC3      PC4      PC5      PC6
## [1,] 0.8930784 -0.05981786 -0.97098080 -0.32722747  0.9238143 -0.4927803
## [2,] 1.7565034 -2.16393542 -0.07834658  1.31816585 -0.2325251 -0.1199267
## [3,] -2.6417622  1.77214093 -0.25429763 -0.23821400 -1.6690826  1.9269231
## [4,] -1.4323209  0.71295161 -0.13114739  1.72626176  0.5961859 -0.0925010
## [5,]  1.2302770 -0.94486199  0.27801253  1.08399558 -0.3331520 -1.0172092
## [6,] -0.2887514  2.35926409 -2.92285398  0.04385271 -0.1945308 -0.9634055
```

```
##          PC7          PC8          PC9          PC10          PC11          PC12
## [1,]  0.1009824  0.651241168  0.6247011 -0.2092549  0.2453571  0.5611628
## [2,] -0.3036975  0.514094776 -1.1343872 -0.1550909 -0.6500718  0.4742277
## [3,] -0.1860566  0.006445367  0.4886804 -1.1327358 -0.1528996 -0.3188954
## [4,] -0.1856331 -0.928931619  0.1125664 -0.3373321 -0.2422261 -0.2552262
## [5,]  0.4715290  1.995736250 -0.3374791  0.6185508 -0.1764931 -0.9572414
## [6,] -1.3480963  0.346571689 -0.3554556  0.7335514  0.3869734 -0.1430733
##          PC13          PC14          PC15          PC16
## [1,]  0.4778485  0.29771859 -0.17539637 -0.14631681
## [2,]  0.3592356 -0.10884273 -0.25695322  0.09789929
## [3,]  0.1384509 -0.30770441  0.12484341  0.06258464
## [4,] -0.5752980  0.25874542 -0.35628808  0.18537500
## [5,] -0.4161035 -0.02523226  0.02481071  0.02486392
## [6,]  0.1021369 -0.16528975  0.15868235  0.08522987
```

```
# Ajouter le biplot au PDF
```

```
biplot(res, scale = 0, main = "Biplot des composantes principales")
dev.off()
```

```
## pdf
```

```
## 2
```

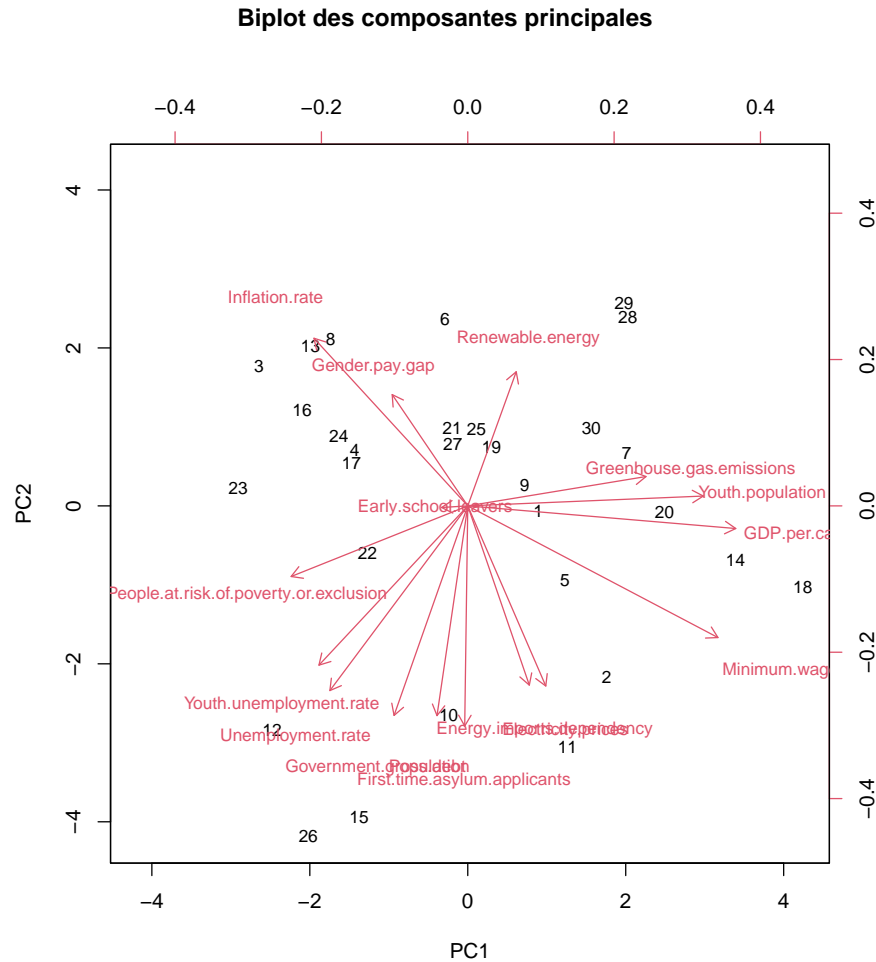
```
# Suite 5
```

```
# Ajuster les dimensions et ajouter un titre pour plus de visibilité
```

```
par(oma = c(0, 0, 2, 0))
```

```
biplot(res, scale = 0, cex = 0.8)
```

```
title("Biplot des composantes principales", outer = TRUE)
```



Les coordonnées des pays dans la nouvelle base sont déterminées par les composantes principales (PC1 et PC2), qui résument la majorité de la variance des données. Ces coordonnées sont affichées sur le premier plan factoriel, permettant une visualisation claire de leur position relative.

Sur le graphique, on observe que certains pays se distinguent par leur éloignement des autres. La Norvège et l'Islande se démarquent nettement dans la partie supérieure droite, ce qui peut s'expliquer par leurs caractéristiques particulières, notamment leur indépendance énergétique et leur forte utilisation des énergies renouvelables. À l'opposé, des pays comme la Grèce, l'Espagne et l'Italie apparaissent isolés dans la partie inférieure gauche, probablement en raison de leurs spécificités économiques ou sociales.

Ce type de visualisation permet d'identifier rapidement les pays qui sortent du lot, comme la Norvège et l'Islande, et, à l'opposé, l'Espagne et l'Italie. Elle aide à mieux comprendre les similarités ou les divergences entre les pays, constituant ainsi une base pour des analyses approfondies.

```
#6
# Étape 1 : Calculer les composantes principales (si non déjà fait)
euro_data_replace_na <- apply(euro_data, 2, function(x) ifelse(is.na(x), mean(x, na.rm = TRUE), x))
res <- prcomp(euro_data_replace_na, scale = TRUE, center = TRUE)
```

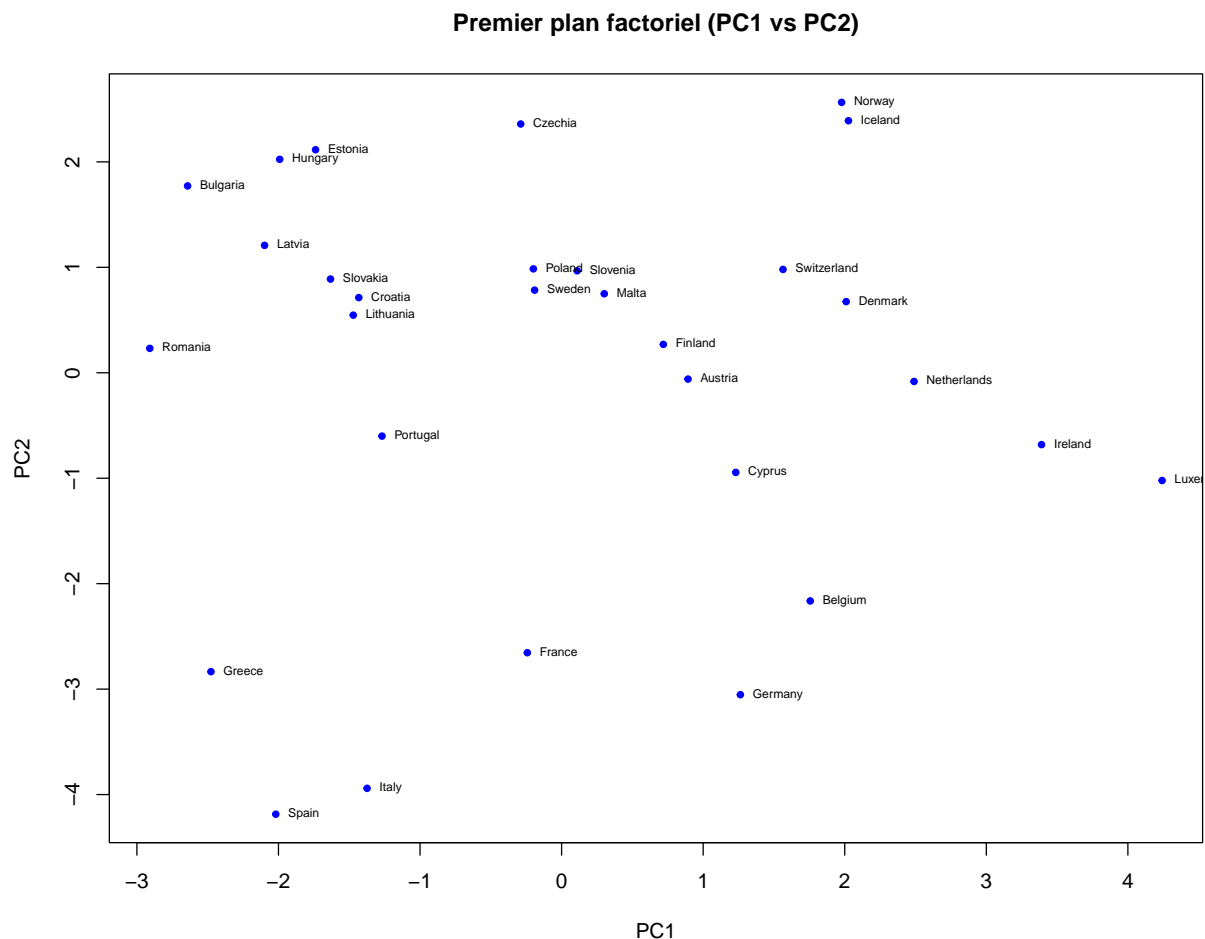
```
#6
coord <- res$x

# Étape 3 : Ajouter les noms des pays comme noms de lignes
```

```
tmp <- read.csv("data/euro.csv", header = TRUE, sep = ";")
rownames(coord) <- tmp[, 1]

# Étape 4 : Créer le graphique du premier plan factoriel
plot(coord[, 1], coord[, 2],
      xlab = "PC1", ylab = "PC2",
      main = "Premier plan factoriel (PC1 vs PC2)",
      pch = 20, col = "blue")

# Étape 5 : Ajouter les noms des pays sur le graphique
text(coord[, 1], coord[, 2], labels = rownames(coord), pos = 4, cex = 0.6)
```



L'ébouli des valeurs propres représente la contribution de chaque composante principale à la variance totale. Dans le graphique, la ligne bleue montre les pourcentages de variances expliquées par chaque composante, tandis que la ligne rouge illustre la variance expliquée cumulée.

En analysant la variance expliquée cumulée, les cinq premières composantes principales expliquent environ 77 % de la variance totale. Cela justifie le choix de retenir les cinq premières composantes pour une analyse approfondie, car elles capturent l'essentiel de l'information tout en réduisant la dimensionnalité des données. Les composantes suivantes apportent une contribution marginale et peuvent être négligées pour simplifier l'interprétation.

Le critère de l'ébouli (coudée visible dans la ligne bleue après la cinquième composante) renforce également

cette décision. Ainsi, la sélection des cinq premières composantes est justifiée à la fois par la proportion de variance expliquée et par la méthode visuelle.

```
#7
variances <- res$sdev^2

# Calculer le pourcentage de variance expliquée
pourcentages <- variances / sum(variances) * 100

# Calculer la variance expliquée cumulée
cumul <- cumsum(pourcentages)

# Afficher les informations pour contrôle
cat("Valeurs propres :\n")

## Valeurs propres :

print(round(variances, 2))

## [1] 3.64 3.52 2.03 1.86 1.28 0.87 0.75 0.56 0.45 0.35 0.21 0.17 0.16 0.09 0.04
## [16] 0.03

cat("\nPourcentages de variances expliquées :\n")

##
## Pourcentages de variances expliquées :

print(round(pourcentages, 2))

## [1] 22.72 22.02 12.72 11.60 8.02 5.43 4.70 3.50 2.80 2.16 1.28 1.05
## [13] 0.99 0.56 0.26 0.19

cat("\nVariance expliquée cumulée :\n")

##
## Variance expliquée cumulée :

print(round(cumul, 2))

## [1] 22.72 44.74 57.46 69.06 77.08 82.51 87.21 90.70 93.51 95.66
## [11] 96.95 98.00 98.99 99.55 99.81 100.00

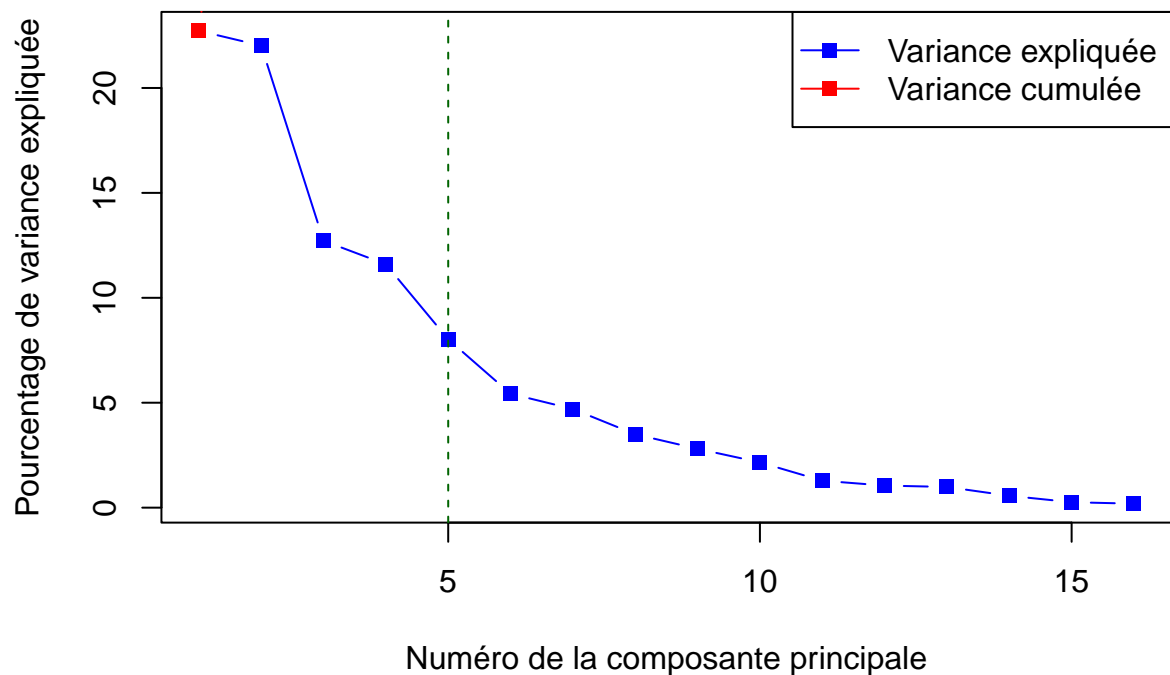
# Créer un graphique combiné
plot(pourcentages, type = "b", pch = 15, col = "blue",
     xlab = "Numéro de la composante principale",
     ylab = "Pourcentage de variance expliquée",
     main = "Ébouli des valeurs propres")
lines(cumul, type = "b", pch = 15, col = "red") # Ajouter la variance cumulée
```



```
# Ajouter une légende
legend("topright", legend = c("Variance expliquée", "Variance cumulée"),
      col = c("blue", "red"), pch = 15, lty = 1)

# Ajouter une ligne verticale pour la sélection des composantes (facultatif)
abline(v = 5, col = "darkgreen", lty = 2) # Par exemple, si 5 composantes sont retenues
```

## Ébouli des valeurs propres



Le biplot permet d'observer simultanément les projections des pays (points bleus) et des variables initiales (flèches rouges) dans le plan défini par les deux premières composantes principales (PC1 et PC2). Les pays proches sur le graphique partagent des similitudes en termes de variables, tandis que les flèches indiquent les contributions des variables aux composantes principales.

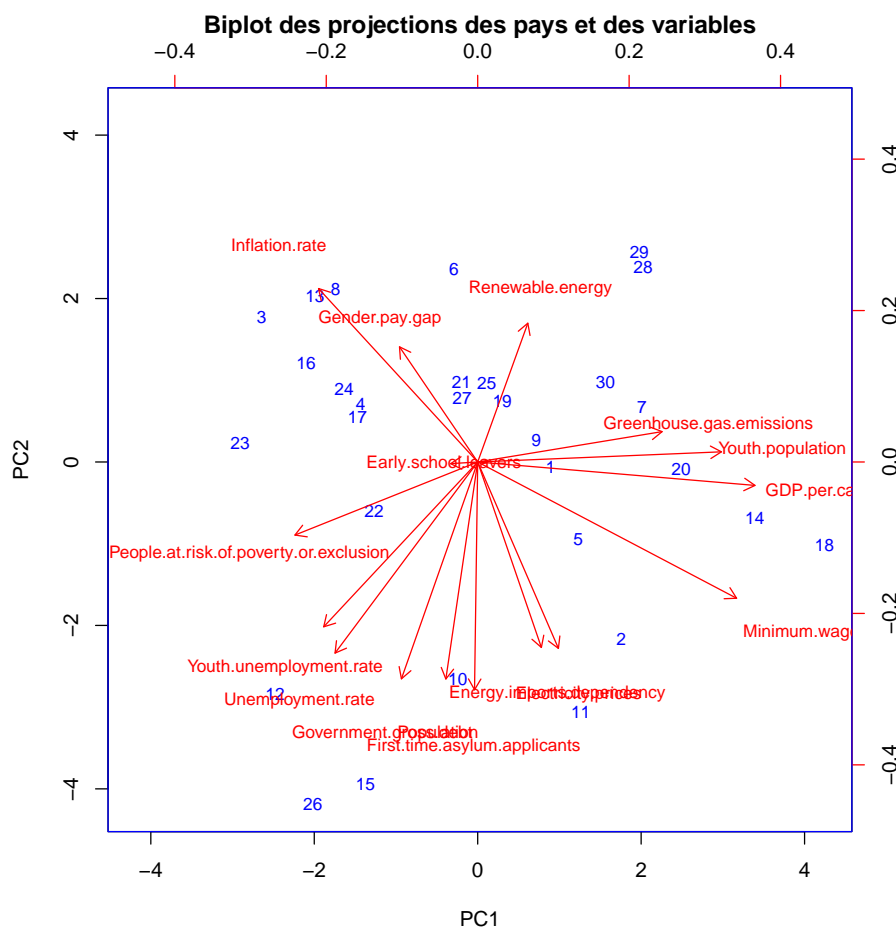
Les flèches rouges montrent la direction et l'intensité des variables. Par exemple, des variables comme Renewable.energy et Greenhouse.gas.emissions influencent fortement le PC1, tandis que People.at.risk.of.poverty.or.exclusion et Youth.unemployment.rate impactent davantage le PC2.

Certains pays, comme la Norvège et l'Islande, se distinguent par leurs projections vers des valeurs élevées de Renewable.energy. À l'opposé, des pays comme la Grèce et l'Italie sont projetés dans des directions qui reflètent d'autres caractéristiques économiques et sociales.

Ce graphique est utile pour identifier les regroupements de pays basés sur des critères communs et pour comprendre les relations entre variables et composantes principales.

```
# 8
# Créer le biplot avec des couleurs et des tailles ajustées
biplot(res, scale = 0,
      main = "Biplot des projections des pays et des variables",
```

```
cex = 0.8, # Taille des points et flèches
col = c("blue", "red"))
```



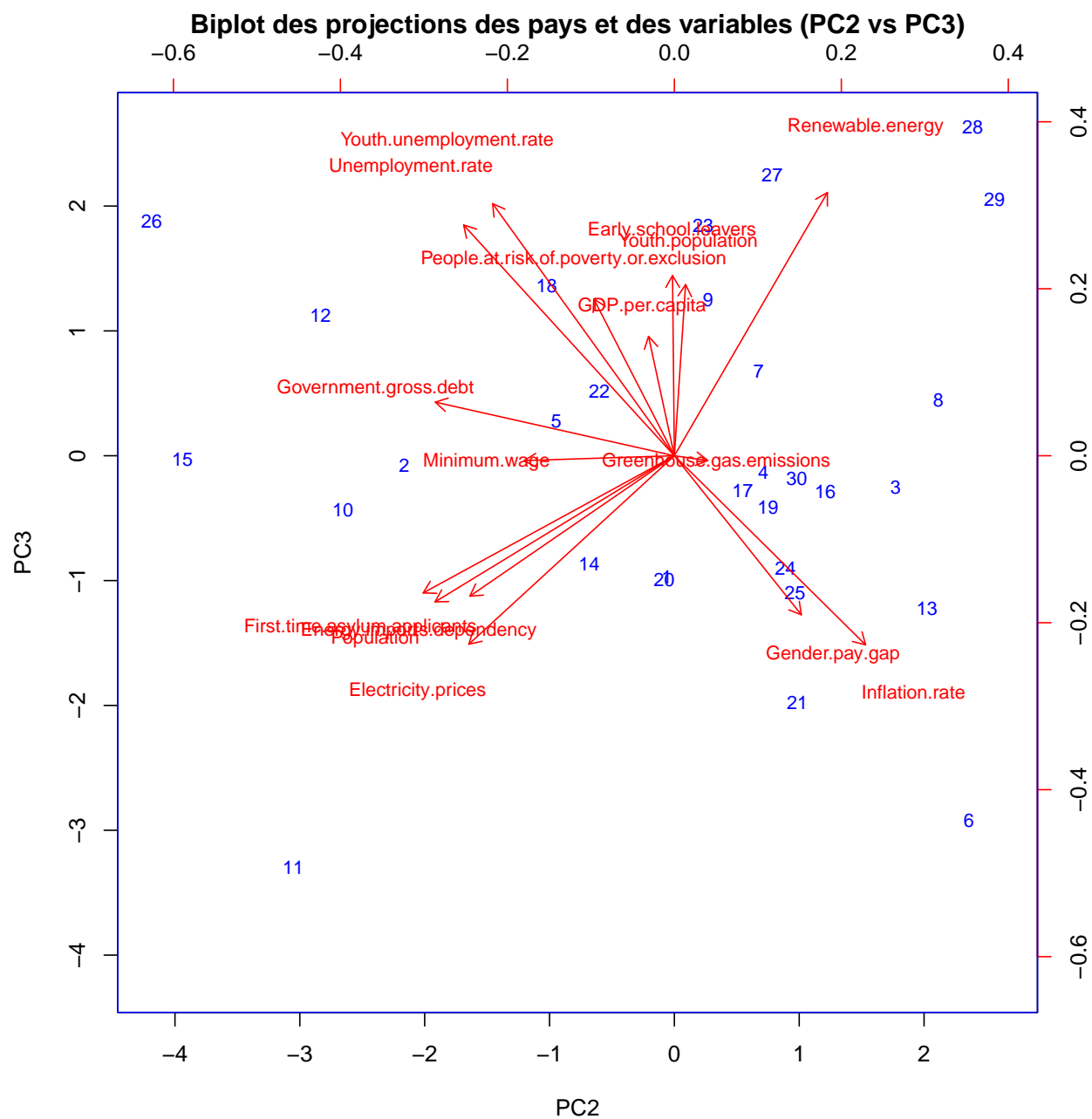
Sur les deux plans factoriels observés, plusieurs éléments significatifs ressortent. Sur le premier plan (PC2 vs PC3), Youth.unemployment.rate et Unemployment.rate se distinguent par leur alignement, reflétant une corrélation positive et leur importance pour la différenciation des pays. Renewable.energy et Inflation.rate se projettent dans des directions opposées, mettant en évidence des dynamiques contrastées entre ces variables. Le pays marqué comme 26 se démarque nettement, tandis que d'autres restent proches du centre, traduisant une certaine homogénéité.

Sur le second plan (PC3 vs PC4), Energy.imports.dependency et Renewable.energy montrent une forte influence, indiquant des relations marquées entre la dépendance énergétique et les efforts en énergies renouvelables. Population et First.time.asylum.applicants se projettent dans des directions opposées, soulignant des contrastes importants. Le pays 11 sort particulièrement du lot sur l'axe PC3, tandis que les autres se regroupent davantage autour du centre.

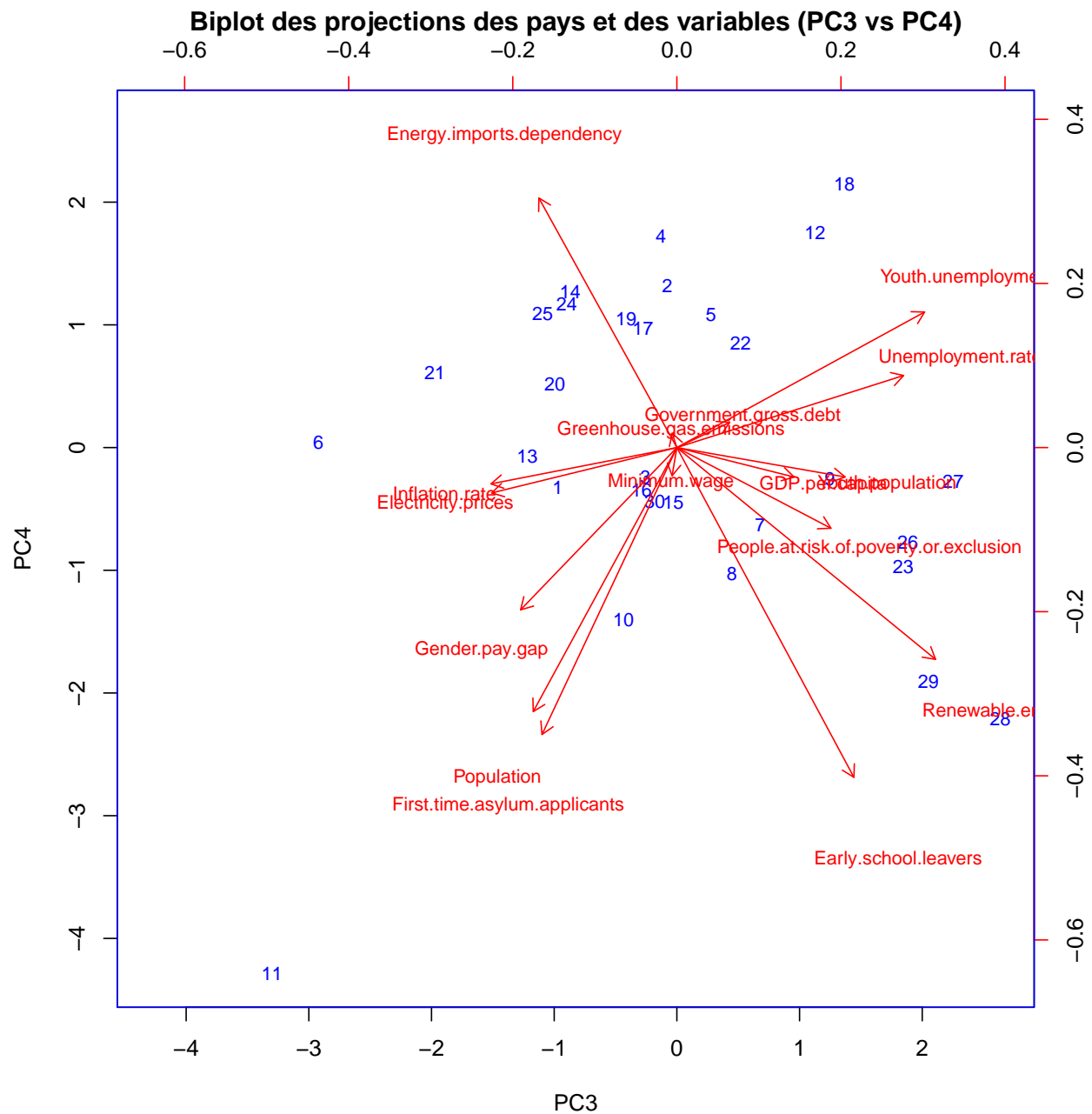
Ces analyses mettent en lumière les relations complexes entre variables et pays, permettant de mieux comprendre les dynamiques sous-jacentes tout en identifiant les facteurs qui différencient certains pays.

```
# 9
# Plan PC2 vs PC3
biplot(res, choices = c(2, 3), scale = 0,
       main = "Biplot des projections des pays et des variables (PC2 vs PC3)",
```

```
cex = 0.8,  
col = c("blue", "red"))
```



```
# Plan PC3 vs PC4
biplot(res, choices = c(3, 4), scale = 0,
       main = "Biplot des projections des pays et des variables (PC3 vs PC4)",
       cex = 0.8, # Taille des points et flèches
       col = c("blue", "red"))
```



Les  $\cos^2$  indiquent la qualité de la représentation des variables sur le premier plan factoriel (PC1-PC2). Plus le  $\cos^2$  est élevé, meilleure est la représentation de la variable sur ce plan. Les variables les mieux représentées incluent Minimum.wage, GDP.per.capita, et Youth.population, avec des  $\cos^2$  respectifs de 0.23, 0.21 et 0.16. Ces variables jouent donc un rôle clé dans la structuration de ce plan factoriel.

En revanche, des variables comme Gender.pay.gap et Early.school.leavers ont des  $\cos^2$  très faibles (0.05 et 0.00), indiquant qu'elles ne sont pas bien représentées sur ce plan. Cela signifie que leur contribution à l'inertie totale du plan PC1-PC2 est limitée, et qu'elles pourraient être mieux représentées sur d'autres plans factoriels.

Ces résultats permettent de se concentrer sur les variables dominantes pour analyser les premières composantes principales et comprendre leur impact sur la structuration des individus et des variables.

```
# 10.
loadings <- res$rotation

# Calculer le cos² pour le plan PC1-PC2
cos2_PC1_PC2 <- rowSums(loadings[, 1:2]^2) # Somme des carrés des charges sur PC1 et PC2

# Afficher les cos²
cat("Cos² des variables sur le plan PC1-PC2 :\n")
```

## Cos² des variables sur le plan PC1-PC2 :

```
print(round(cos2_PC1_PC2, 2))
```

```
##              Population              Youth.population
##              0.13              0.16
## First.time.asylum.applicants      Gender.pay.gap
##              0.14              0.05
## Minimum.wage People.at.risk.of.poverty.or.exclusion
##              0.23              0.11
## Early.school.leavers      Inflation.rate
##              0.00              0.15
## Unemployment.rate      Youth.unemployment.rate
##              0.16              0.14
## GDP.per.capita      Government.gross.debt
##              0.21              0.14
## Greenhouse.gas.emissions      Renewable.energy
##              0.10              0.06
## Electricity.prices      Energy.imports.dependency
##              0.11              0.10
```

```
# Identifier les variables les mieux représentées
best_variables <- names(sort(cos2_PC1_PC2, decreasing = TRUE))
cat("\nVariables les mieux représentées par PC1-PC2 (par ordre de qualité) :\n")
```

```
##
## Variables les mieux représentées par PC1-PC2 (par ordre de qualité) :
```

```
print(best_variables)
```

```
## [1] "Minimum.wage"
## [2] "GDP.per.capita"
## [3] "Youth.population"
## [4] "Unemployment.rate"
## [5] "Inflation.rate"
```

```
## [6] "Government.gross.debt"
## [7] "First.time.asylum.applicants"
## [8] "Youth.unemployment.rate"
## [9] "Population"
## [10] "Electricity.prices"
## [11] "People.at.risk.of.poverty.or.exclusion"
## [12] "Energy.imports.dependency"
## [13] "Greenhouse.gas.emissions"
## [14] "Renewable.energy"
## [15] "Gender.pay.gap"
## [16] "Early.school.leavers"
```

Les individus [1] et [27] ont des contributions relativement élevées sur certaines composantes principales, comme PC12, PC9, ou PC8. Par exemple, pour l'individu [27], la contribution sur PC10PC10 est très marquée avec une valeur de 3.21, ce qui explique une part significative de la variance associée à cette composante. La somme des contributions totales confirme que tous les individus participent de manière significative à la construction des composantes principales, car aucun individu ne tombe en dessous du seuil théorique de contribution ( $1/n$ , où  $n$  est le nombre d'individus).

Dans ce cas, aucun individu n'a une contribution faible, ce qui signifie qu'il n'est pas nécessaire d'éliminer des individus de l'analyse. Tous participent de manière adéquate à la variance expliquée par les composantes principales.

```
# 11

# Étape 1 : Extraire les scores des individus
scores <- res$x # Les coordonnées des individus dans le nouvel espace

# Étape 2 : Extraire les variances des composantes principales
variances <- res$sdev^2 # Valeurs propres

# Étape 3 : Calculer les contributions des individus sur chaque composante
# Contribution = (scores^2) / (variance de la composante)
contributions <- sweep(scores^2, 2, variances, "/")

# Étape 4 : Résumé des contributions
cat("Contributions des individus sur chaque composante principale :\n")
```

## Contributions des individus sur chaque composante principale :

```
print(round(contributions, 2)) # Contributions arrondies
```

```
##      PC1  PC2  PC3  PC4  PC5  PC6  PC7  PC8  PC9  PC10  PC11  PC12  PC13  PC14
## [1,] 0.22 0.00 0.46 0.06 0.66 0.28 0.01 0.76 0.87 0.13 0.29 1.87 1.45 0.98
## [2,] 0.85 1.33 0.00 0.94 0.04 0.02 0.12 0.47 2.87 0.07 2.06 1.34 0.82 0.13
## [3,] 1.92 0.89 0.03 0.03 2.17 4.27 0.05 0.00 0.53 3.72 0.11 0.60 0.12 1.05
## [4,] 0.56 0.14 0.01 1.61 0.28 0.01 0.05 1.54 0.03 0.33 0.29 0.39 2.10 0.74
## [5,] 0.42 0.25 0.04 0.63 0.09 1.19 0.30 7.12 0.25 1.11 0.15 5.45 1.10 0.01
## [6,] 0.02 1.58 4.20 0.00 0.03 1.07 2.42 0.21 0.28 1.56 0.73 0.12 0.07 0.30
## [7,] 1.11 0.13 0.23 0.21 0.66 0.22 0.26 0.70 0.28 1.96 1.71 0.03 0.01 0.72
## [8,] 0.83 1.27 0.10 0.57 0.04 0.93 4.24 1.01 1.45 0.49 1.85 0.45 0.01 0.90
## [9,] 0.14 0.02 0.76 0.04 1.05 0.38 0.33 0.04 0.32 0.00 1.20 0.02 5.32 0.07
## [10,] 0.02 2.00 0.09 1.06 0.29 0.54 0.00 3.02 0.24 0.01 3.33 1.48 2.41 0.43
```

```

## [11,] 0.44 2.65 5.35 9.89 0.06 0.43 0.01 0.01 0.33 0.12 0.06 0.95 0.03 0.47
## [12,] 1.69 2.28 0.62 1.65 1.23 0.62 1.67 0.51 1.02 4.76 0.18 0.16 0.71 0.71
## [13,] 1.09 1.16 0.73 0.00 0.60 3.79 1.43 0.07 2.56 0.79 1.29 3.53 0.00 0.01
## [14,] 3.16 0.13 0.37 0.87 0.08 1.16 0.85 0.67 0.04 0.17 0.27 4.07 1.54 0.23
## [15,] 0.52 4.41 0.00 0.11 0.45 0.15 0.01 0.06 3.52 0.00 3.39 0.12 0.01 0.63
## [16,] 1.21 0.41 0.04 0.07 0.78 1.52 0.09 1.56 0.83 0.00 0.01 1.46 0.00 1.69
## [17,] 0.60 0.08 0.04 0.51 0.00 1.08 0.10 0.15 0.00 0.02 0.23 0.01 0.02 7.92
## [18,] 4.95 0.30 0.92 2.48 3.90 1.30 0.75 1.28 2.35 0.30 1.92 0.15 0.35 0.13
## [19,] 0.02 0.16 0.08 0.60 0.35 0.03 11.17 0.11 0.11 0.57 0.29 0.45 0.11 0.18
## [20,] 1.70 0.00 0.49 0.14 0.00 0.00 0.81 0.23 1.43 0.36 3.32 0.01 0.01 0.02
## [21,] 0.01 0.28 1.91 0.20 0.99 0.10 1.09 2.97 2.49 0.00 0.31 1.05 1.60 0.06
## [22,] 0.44 0.10 0.13 0.39 0.49 0.26 0.25 0.03 0.50 0.19 0.13 0.10 0.64 0.76
## [23,] 2.33 0.02 1.67 0.51 5.34 1.24 1.12 0.07 0.85 1.12 0.21 2.12 0.00 0.42
## [24,] 0.73 0.22 0.40 0.74 0.26 0.57 0.01 0.03 1.18 1.74 0.27 0.52 0.01 0.47
## [25,] 0.00 0.27 0.59 0.64 0.00 0.11 0.01 1.65 2.51 0.76 0.07 0.25 7.63 0.03
## [26,] 1.12 4.97 1.74 0.32 0.01 0.01 0.01 0.04 1.04 1.37 0.12 2.03 0.94 0.65
## [27,] 0.01 0.17 2.49 0.04 2.13 0.21 0.01 3.00 0.49 3.21 0.01 0.00 0.66 1.73
## [28,] 1.13 1.62 3.41 2.64 1.77 4.17 0.50 0.66 0.34 3.40 0.01 0.01 0.81 2.98
## [29,] 1.08 1.87 2.07 1.96 0.30 0.23 0.00 1.02 0.00 0.15 1.78 0.03 0.00 4.37
## [30,] 0.67 0.27 0.02 0.10 4.97 3.10 1.32 0.01 0.30 0.60 3.43 0.23 0.53 0.24
##      PC15 PC16
## [1,] 0.74 0.71
## [2,] 1.59 0.32
## [3,] 0.37 0.13
## [4,] 3.05 1.14
## [5,] 0.01 0.02
## [6,] 0.61 0.24
## [7,] 1.16 0.32
## [8,] 0.25 0.18
## [9,] 0.22 2.69
## [10,] 1.72 0.18
## [11,] 3.44 0.80
## [12,] 0.01 1.17
## [13,] 0.10 0.09
## [14,] 0.30 1.16
## [15,] 0.24 0.81
## [16,] 0.68 0.07
## [17,] 1.73 0.26
## [18,] 0.18 0.25
## [19,] 0.16 0.49
## [20,] 0.10 1.09
## [21,] 0.81 3.21
## [22,] 1.95 4.43
## [23,] 1.82 0.17
## [24,] 2.12 0.14
## [25,] 0.12 4.48
## [26,] 3.52 0.52
## [27,] 1.56 1.31
## [28,] 0.06 0.02
## [29,] 0.36 0.89
## [30,] 0.02 1.67

```

```

# Étape 5 : Calcul des contributions totales par individu (somme sur toutes les composantes)
contributions_totales <- rowSums(contributions)

```



```
cat("\nContributions totales par individu :\n")
```

```
##
```

```
## Contributions totales par individu :
```

```
print(round(contributions_totales, 2))
```

```
## [1]  9.50 12.96 16.01 12.26 18.13 13.44  9.72 14.57 12.61 16.81 25.03 18.96
## [13] 17.24 15.08 14.42 10.42 12.74 21.50 14.88  9.73 17.08 10.81 18.99  9.42
## [25] 19.12 18.42 17.04 23.55 16.11 17.47
```

```
# Étape 6 : Identifier les individus avec des contributions faibles
seuil <- 1 / nrow(scores) # Seuil théorique de contribution moyenne
individus_faibles <- names(contributions_totales[contributions_totales < seuil])
cat("\nIndividus ayant une contribution faible (en dessous du seuil) :\n")
```

```
##
```

```
## Individus ayant une contribution faible (en dessous du seuil) :
```

```
print(individus_faibles)
```

```
## NULL
```

La projection des individus sur les composantes principales (PC1 et PC2) montre des regroupements qui correspondent globalement aux similarités socio-économiques attendues :

Les pays d'Europe du Nord (Norvège, Islande, Danemark) sont situés dans une région distincte, ce qui reflète des caractéristiques socio-économiques communes comme des niveaux élevés de revenus et de bien-être. Les pays d'Europe du Sud (Espagne, Italie, Grèce) apparaissent dans une région différente, marquant des disparités avec les pays du Nord, notamment en termes de chômage ou de performances économiques. Les pays d'Europe de l'Est (Bulgarie, Roumanie, Lettonie) se regroupent dans une zone particulière, suggérant des similarités dues à des caractéristiques économiques ou sociales spécifiques.

Cette distribution confirme que les projections sur les composantes principales reflètent bien certaines similarités et différences entre les pays, conformément aux attentes. Cela valide également l'utilité de l'analyse en composantes principales pour interpréter les relations complexes entre les variables et les individus (pays dans ce cas).

```
# question 12 12. La projection des individus sur les composantes correspond-elle, d'une manière ou d'une autre, aux attendues
```

```
# Étape 1 : Extraire les scores
```

```
coord <- res$x # Scores des individus
```

```
# Étape 2 : Ajouter les noms des pays
```

```
tmp <- read.csv("data/euro.csv", header = TRUE, sep = ";") # Charger les noms
rownames(coord) <- tmp[, 1]
```

```
# Étape 3 : Créer le graphique
```

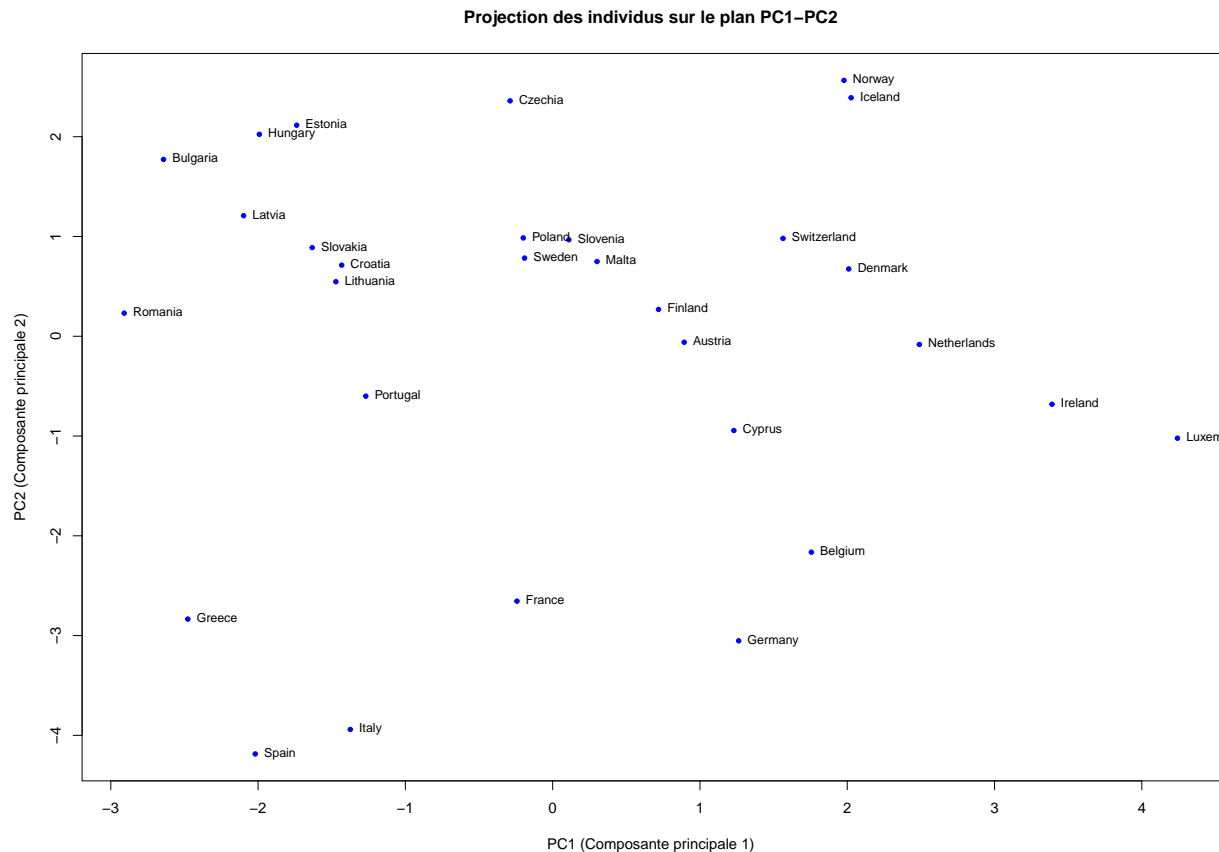
```
plot(coord[, 1], coord[, 2],
      xlab = "PC1 (Composante principale 1)",
      ylab = "PC2 (Composante principale 2)",
```

```

main = "Projection des individus sur le plan PC1-PC2",
pch = 20, col = "blue")

# Ajouter les noms des individus
text(coord[, 1], coord[, 2], labels = rownames(coord), pos = 4, cex = 0.8)

```



*Conclusion* Cette étude en analyse en composantes principales (ACP) a permis de réduire la dimensionnalité des données tout en conservant l'essentiel de l'information. Les résultats obtenus, tels que les projections des pays sur les composantes principales et l'ébouillissement des valeurs propres, ont révélé des regroupements significatifs et des similarités attendues entre les pays en fonction de leurs caractéristiques socio-économiques. Ces visualisations offrent une vue claire des relations entre les variables et des différences structurelles entre les pays, facilitant ainsi une interprétation plus approfondie. Cette analyse constitue une base solide pour explorer des méthodes complémentaires, telles que le partitionnement, afin de segmenter davantage les pays en groupes homogènes.

```
# TP2 *****
```

```
# Question 1
```

```

# Matrice de dissimilarité Euclidienne (non normalisée)
dissimilarity_euclidean <- as.matrix(dist(euro_data, method = "euclidean"))

# Normaliser les données
euro_data_normalized <- scale(euro_data)

```

```
# Matrice de dissimilarité Euclidienne (normalisée)
dissimilarity_reduced <- as.matrix(dist(euro_data_normalized, method = "euclidean"))

# Afficher les matrices
cat("Matrice de dissimilarité Euclidienne (non normalisée) :\n")
```

```
## Matrice de dissimilarité Euclidienne (non normalisée) :
```

```
print(round(dissimilarity_euclidean, 2))
```

```
##           1           2           3           4           5           6           7
## 1          0.0  2638160.9  2657441.6  5254209.8  8184197.7  1723739.7  3172609.6
## 2  2638160.9          0.0  5295172.6  7891982.7  10822113.6  915893.4  5810224.2
## 3  2657441.6  5295172.6          0.0  2596908.1  5527055.2  4379883.5  517377.9
## 4  5254209.8  7891982.7  2596908.1          0.0  2930238.9  6976636.0  2082102.7
## 5  8184197.7  10822113.6  5527055.2  2930238.9          0.0  9906837.9  5012023.0
## 6  1723739.7  915893.4  4379883.5  6976636.0  9906837.9          0.0  4894993.5
## 7  3172609.6  5810224.2  517377.9  2082102.7  5012023.0  4894993.5          0.0
## 8  7739095.7  10376966.3  5081864.8  2485011.2  445423.9  9461646.0  4566922.4
## 9  3541179.2  6178875.8  884403.3  1713222.9  4643283.9  5263592.7  369016.5
## 10 59068272.1  56430300.0  61725394.4  64322245.8  67252408.7  57345630.8  62240489.5
## 11 75254567.8  72616667.8  77911743.7  80508619.6  83438748.0  73532049.3  78426873.1
## 12 1309339.5  1329246.9  3966447.0  6563330.6  9493397.5  417425.3  4481796.4
## 13 498677.6  2143374.4  3152120.1  5748850.2  8679061.1  1227792.4  3667289.2
## 14 3833770.1  6471512.3  1178083.5  1421690.0  4350916.7  5556402.2  661626.8
## 15 49892485.3  47254514.4  52549606.4  55146459.4  58076621.7  48169847.0  53064707.3
## 16 7222010.5  9859856.2  4564752.4  1967886.6  962468.6  8944522.6  4049836.8
## 17 6247781.6  8885592.0  3590504.5  993615.7  1936652.9  7970250.9  3075606.3
## 18 8444257.2  11082114.6  5787427.3  3190822.5  265931.4  10166927.0  5271935.1
## 19 8562908.8  11200787.2  5905729.2  3308864.2  378816.7  10285481.2  5390666.0
## 20 8706540.1  6068506.0  11363651.2  13960476.9  16890619.4  6983909.5  11878694.2
## 21 27649015.7  25010959.5  30306030.3  32902842.6  35833037.6  25926208.1  30821105.6
## 22 1412975.2  1226591.2  4068976.5  6665728.9  9595927.6  310914.0  4584084.3
## 23 9949920.8  7311827.9  12606844.4  15203656.9  18133855.5  8227027.8  13121964.2
## 24 3676462.9  6314104.5  1019192.6  1577899.5  4508119.1  5398737.4  505152.5
## 25 6987988.3  9625861.3  4330788.3  1733946.6  1196292.5  8710559.9  3815806.0
## 26 38980730.5  36342803.8  41637883.5  44234753.4  47164894.8  37258173.3  42153012.4
## 27 1464078.3  1261488.4  4207644.3  6889503.2  9915736.2  317260.5  4739409.5
## 28 9319046.8  12139076.8  6478490.5  3702359.4  569987.2  11160612.2  5927755.2
## 29 3734910.3  6459056.0  992533.1  1692844.2  4718327.5  5513902.8  458665.5
## 30 301607.7  3023539.1  2446003.6  5127617.3  8153693.3  2078822.6  2977400.6
##           8           9           10           11           12           13           14
## 1  7739095.7  3541179.23  59068272  75254568  1309339.5  498677.6  3833770.1
## 2  10376966.3  6178875.82  56430300  72616668  1329246.9  2143374.4  6471512.3
## 3  5081864.8  884403.26  61725394  77911744  3966447.0  3152120.1  1178083.5
## 4  2485011.2  1713222.94  64322246  80508620  6563330.6  5748850.2  1421690.0
## 5  445423.9  4643283.92  67252409  83438748  9493397.5  8679061.1  4350916.7
## 6  9461646.0  5263592.70  57345631  73532049  417425.3  1227792.4  5556402.2
## 7  4566922.4  369016.54  62240490  78426873  4481796.4  3667289.2  661626.8
## 8          0.0  4198142.43  66807245  82993600  9048259.5  8233861.0  3905930.1
## 9  4198142.4          0.00  62609165  78795544  4850339.3  4035840.0  294757.5
## 10 66807244.6  62609165.06          0  16186913  57759062.7  58573415.9  62901731.7
## 11 82993600.2  78795543.54  16186913          0  73945362.1  74759828.2  79088088.5
```

## 12	9048259.5	4850339.35	57759063	73945362	0.0	816305.6	5143049.7
## 13	8233861.0	4035839.96	58573416	74759828	816305.6	0.0	4328749.0
## 14	3905930.1	294757.49	62901732	79088088	5143049.7	4328749.0	0.0
## 15	57631457.5	53433380.51	9175789	25362422	48583274.2	49397631.5	53725951.5
## 16	517133.4	3681040.00	66290127	82476490	8531161.6	7716736.2	3388911.8
## 17	1491399.1	2706784.64	65315861	81502231	7556922.1	6742465.0	2414819.3
## 18	708356.7	4903380.29	67512337	83698686	9753540.9	8939201.5	4610612.9
## 19	823917.6	5021931.53	67631081	83817438	9872100.7	9057701.2	4729577.0
## 20	16445468.8	12247370.11	50361800	66548190	7397378.3	8211691.5	12539950.7
## 21	35387852.2	31189774.11	31419547	47606198	26339802.2	27153993.1	31482393.0
## 22	9150738.2	4952681.53	57656534	73842947	116587.4	916896.3	5245494.4
## 23	17688665.7	13490605.64	49118621	65305082	8640704.0	9454810.0	13783290.5
## 24	4062909.8	136788.50	62744354	78930740	4985522.6	4170952.5	167299.3
## 25	751126.5	3447031.16	66056150	82242504	8297165.5	7482779.5	3154818.4
## 26	46719740.1	42521678.83	20087624	36273877	37671519.1	38485952.9	42814244.4
## 27	9455987.4	5120180.96	59542299	76259541	124863.7	952600.0	5422417.0
## 28	1046036.9	5533605.49	72465604	89769540	10718665.1	9848070.9	5220936.5
## 29	4258721.0	85416.58	64739923	81457137	5087099.3	4245993.3	224880.3
## 30	7694009.0	3358237.98	61304387	78021579	1651970.6	812140.9	3660253.9
##	15	16	17	18	19	20	21
## 1	49892485	7222010.5	6247781.6	8444257.2	8562908.8	8706540	27649016
## 2	47254514	9859856.2	8885592.0	11082114.6	11200787.2	6068506	25010959
## 3	52549606	4564752.4	3590504.5	5787427.3	5905729.2	11363651	30306030
## 4	55146459	1967886.6	993615.7	3190822.5	3308864.2	13960477	32902843
## 5	58076622	962468.6	1936652.9	265931.4	378816.7	16890619	35833038
## 6	48169847	8944522.6	7970250.9	10166927.0	10285481.2	6983909	25926208
## 7	53064707	4049836.8	3075606.3	5271935.1	5390666.0	11878694	30821106
## 8	57631458	517133.4	1491399.1	708356.7	823917.6	16445469	35387852
## 9	53433381	3681040.0	2706784.6	4903380.3	5021931.5	12247370	31189774
## 10	9175789	66290127.4	65315860.8	67512336.5	67631081.0	50361800	31419547
## 11	25362422	82476490.1	81502231.0	83698685.7	83817438.5	66548190	47606198
## 12	48583274	8531161.6	7556922.1	9753540.9	9872100.7	7397378	26339802
## 13	49397632	7716736.2	6742465.0	8939201.5	9057701.2	8211692	27153993
## 14	53725952	3388911.8	2414819.3	4610612.9	4729577.0	12539951	31482393
## 15	0	57114340.6	56140074.3	58336558.1	58455294.8	41186016	22243808
## 16	57114341	0.0	974273.0	1224209.5	1341023.6	15928356	34870729
## 17	56140074	974273.0	0.0	2197538.9	2315257.5	14954089	33896458
## 18	58336558	1224209.5	2197538.9	0.0	131665.5	17150563	36092993
## 19	58455295	1341023.6	2315257.5	131665.5	0.0	17269291	36211688
## 20	41186016	15928356.0	14954089.2	17150563.2	17269290.8	0	18942493
## 21	22243808	34870728.6	33896457.8	36092992.5	36211687.6	18942493	0
## 22	48480750	8633615.5	7659343.8	9856017.3	9974572.6	7294799	26237116
## 23	39942840	17171542.2	16197272.4	18393885.7	18512506.5	1244054	17699189
## 24	53568569	3545785.7	2571513.5	4768452.2	4886751.3	12382589	31324945
## 25	56880363	234200.3	740373.1	1457455.8	1574941.4	15694366	34636765
## 26	10911881	46202627.6	45228366.0	47424850.3	47543579.1	30274323	11332659
## 27	50065591	8921916.1	7915695.8	10184216.3	10306809.1	7528867	27092497
## 28	62656277	1598777.4	2640185.5	295313.7	165765.1	18626576	38876876
## 29	55263219	3724730.3	2718655.6	4986532.4	5109387.3	12726502	32290151
## 30	51827680	7159963.7	6153770.5	8422071.9	8544791.2	9290960	28854655
##	22	23	24	25	26	27	28
## 1	1412975.22	9949921	3676462.90	6987988.3	38980731	1464078.30	9319046.8
## 2	1226591.23	7311828	6314104.51	9625861.3	36342804	1261488.36	12139076.8
## 3	4068976.47	12606844	1019192.57	4330788.3	41637884	4207644.26	6478490.5

## 4	6665728.92	15203657	1577899.47	1733946.6	44234753	6889503.25	3702359.4
## 5	9595927.61	18133856	4508119.13	1196292.5	47164895	9915736.15	569987.2
## 6	310914.02	8227028	5398737.42	8710559.9	37258173	317260.51	11160612.2
## 7	4584084.33	13121964	505152.48	3815806.0	42153012	4739409.52	5927755.2
## 8	9150738.20	17688666	4062909.79	751126.5	46719740	9455987.39	1046036.9
## 9	4952681.53	13490606	136788.50	3447031.2	42521679	5120180.96	5533605.5
## 10	57656533.80	49118621	62744354.29	66056150.0	20087624	59542298.67	72465604.4
## 11	73842947.40	65305082	78930739.77	82242504.0	36273877	76259541.21	89769540.2
## 12	116587.45	8640704	4985522.60	8297165.5	37671519	124863.68	10718665.1
## 13	916896.26	9454810	4170952.55	7482779.5	38485953	952599.98	9848070.9
## 14	5245494.44	13783290	167299.32	3154818.4	42814244	5422416.95	5220936.5
## 15	48480749.68	39942840	53568568.57	56880363.2	10911881	50065591.20	62656277.5
## 16	8633615.51	17171542	3545785.73	234200.3	46202628	8921916.12	1598777.4
## 17	7659343.84	16197272	2571513.54	740373.1	45228366	7915695.76	2640185.5
## 18	9856017.34	18393886	4768452.21	1457455.8	47424850	10184216.28	295313.7
## 19	9974572.55	18512507	4886751.32	1574941.4	47543579	10306809.06	165765.1
## 20	7294799.49	1244054	12382588.81	15694365.8	30274323	7528867.06	18626576.4
## 21	26237115.97	17699189	31324944.91	34636764.8	11332659	27092496.59	38876876.5
## 22	0.00	8537935	5087830.52	8399650.6	37569072	27013.79	10828235.2
## 23	8537935.36	0	13625760.74	16937580.4	29031207	8812907.74	19955666.8
## 24	5087830.52	13625761	0.00	3311831.9	42656870	5259871.68	5389159.8
## 25	8399650.60	16937580	3311831.85	0.0	45968645	8680248.27	1848726.3
## 26	37569072.06	29031207	42656870.30	45968644.6	0	38796051.75	50991159.3
## 27	27013.79	8812908	5259871.68	8680248.3	38796052	0.00	10833487.5
## 28	10828235.15	19955667	5389159.79	1848726.3	50991159	10833487.50	0.0
## 29	5192801.16	14010600	84618.29	3482978.5	43993668	5197695.75	5453536.8
## 30	1757788.42	10575120	3498101.70	6918252.8	40558111	1762330.42	9009575.6
##	29	30					
## 1	3734910.33	301607.7					
## 2	6459056.01	3023539.1					
## 3	992533.11	2446003.6					
## 4	1692844.17	5127617.3					
## 5	4718327.55	8153693.3					
## 6	5513902.84	2078822.6					
## 7	458665.48	2977400.6					
## 8	4258721.03	7694009.0					
## 9	85416.58	3358238.0					
## 10	64739922.94	61304386.8					
## 11	81457137.37	78021579.4					
## 12	5087099.33	1651970.6					
## 13	4245993.26	812140.9					
## 14	224880.27	3660253.9					
## 15	55263218.86	51827680.0					
## 16	3724730.34	7159963.7					
## 17	2718655.56	6153770.5					
## 18	4986532.40	8422071.9					
## 19	5109387.27	8544791.2					
## 20	12726501.58	9290960.2					
## 21	32290151.02	28854655.5					
## 22	5192801.16	1757788.4					
## 23	14010599.65	10575119.5					
## 24	84618.29	3498101.7					
## 25	3482978.53	6918252.8					
## 26	43993667.87	40558110.9					

```
## 27 5197695.75 1762330.4
## 28 5453536.80 9009575.6
## 29      0.00 3435576.1
## 30 3435576.09      0.0
```

```
cat("\nMatrice de dissimilarité Euclidienne (normalisée) :\n")
```

```
##
## Matrice de dissimilarité Euclidienne (normalisée) :
```

```
print(round(dissimilarity_reduced, 2))
```

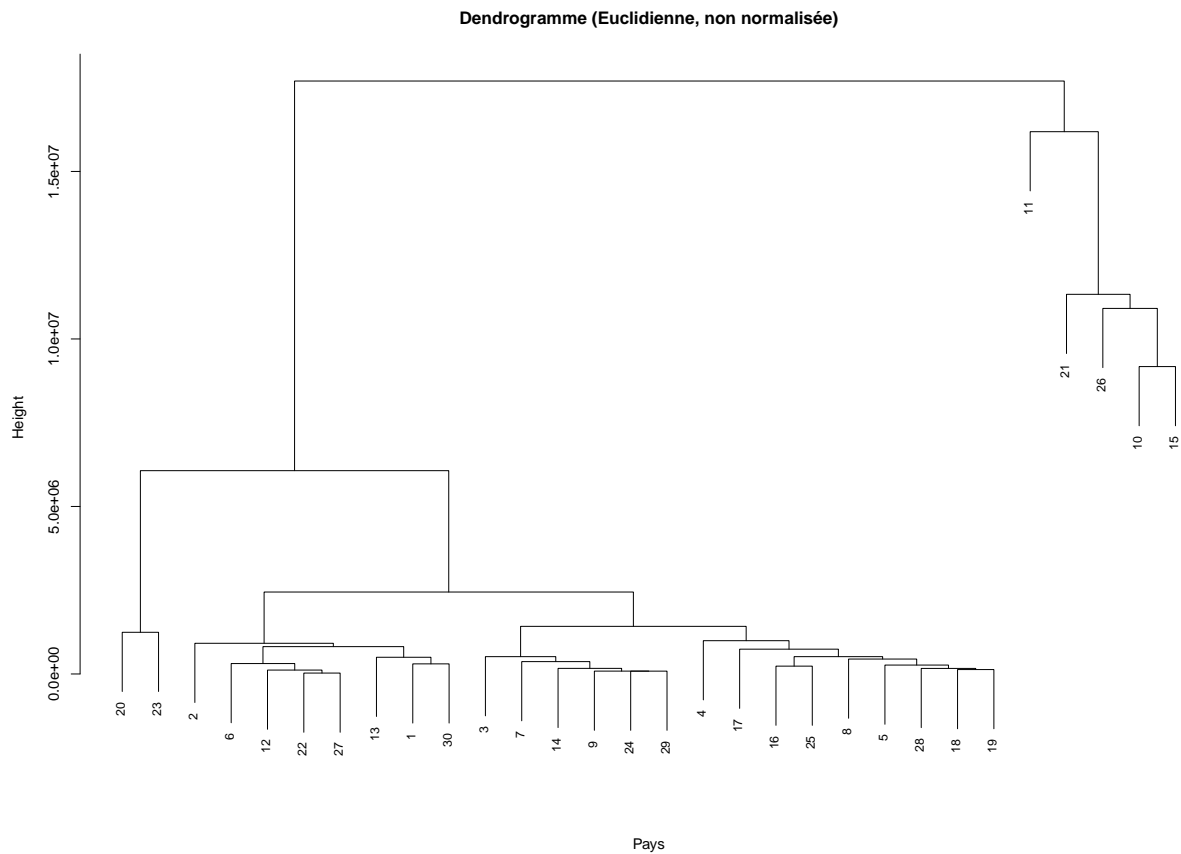
```
##      1      2      3      4      5      6      7      8      9     10     11     12     13     14     15
## 1  0.00  3.80  5.55  3.97  3.61  4.15  3.04  4.70  2.71  4.04  5.87  5.60  4.48  3.95  5.40
## 2  3.80  0.00  6.87  4.91  3.05  6.20  4.09  6.63  4.12  4.43  6.90  5.34  6.83  3.25  4.39
## 3  5.55  6.87  0.00  4.26  6.58  5.39  6.13  3.53  5.61  6.52  8.25  6.66  4.71  7.13  6.77
## 4  3.97  4.91  4.26  0.00  4.79  4.65  4.87  4.29  3.81  4.94  8.36  4.59  4.18  5.66  5.74
## 5  3.61  3.05  6.58  4.79  0.00  5.64  3.65  5.96  4.01  5.21  7.36  5.63  5.87  4.10  5.24
## 6  4.15  6.20  5.39  4.65  5.64  0.00  5.36  4.59  5.30  6.46  7.50  7.65  4.00  5.79  7.31
## 7  3.04  4.09  6.13  4.87  3.65  5.36  0.00  4.84  2.67  5.01  6.95  6.94  5.88  3.89  6.41
## 8  4.70  6.63  3.53  4.29  5.96  4.59  4.84  0.00  4.14  6.11  8.09  6.43  4.87  6.66  7.09
## 9  2.71  4.12  5.61  3.81  4.01  5.30  2.67  4.14  0.00  4.28  7.33  5.32  5.22  4.86  5.73
## 10 4.04  4.43  6.52  4.94  5.21  6.46  5.01  6.11  4.28  0.00  5.10  5.01  5.98  5.82  3.58
## 11 5.87  6.90  8.25  8.36  7.36  7.50  6.95  8.09  7.33  5.10  0.00  8.79  8.22  7.12  6.18
## 12 5.60  5.34  6.66  4.59  5.63  7.65  6.94  6.43  5.32  5.01  8.79  0.00  6.90  7.16  4.51
## 13 4.48  6.83  4.71  4.18  5.87  4.00  5.88  4.87  5.22  5.98  8.22  6.90  0.00  7.13  6.86
## 14 3.95  3.25  7.13  5.66  4.10  5.79  3.89  6.66  4.86  5.82  7.12  7.16  7.13  0.00  6.54
## 15 5.40  4.39  6.77  5.74  5.24  7.31  6.41  7.09  5.73  3.58  6.18  4.51  6.86  6.54  0.00
## 16 3.91  5.73  3.58  3.47  5.39  4.51  4.54  2.95  4.15  5.47  7.62  5.60  4.51  6.25  5.94
## 17 3.49  4.76  3.38  2.26  4.42  4.55  4.40  3.82  3.85  5.04  7.63  4.96  4.08  5.32  5.32
## 18 6.32  4.96  8.16  6.98  5.60  8.09  5.79  8.15  6.06  7.37  9.10  8.26  8.53  3.86  7.64
## 19 3.92  4.87  5.17  4.07  4.00  5.59  4.24  6.04  4.63  5.61  7.80  6.71  4.31  5.34  6.34
## 20 2.79  3.57  6.17  4.55  3.80  5.20  3.15  5.98  3.70  4.62  6.42  6.81  5.62  3.01  6.32
## 21 4.26  4.89  4.65  3.46  5.12  3.06  5.07  4.96  4.83  5.16  7.14  6.74  4.46  5.11  5.86
## 22 3.28  3.90  4.84  2.55  3.73  5.30  4.22  4.64  3.02  4.07  7.53  3.70  4.55  5.57  4.18
## 23 6.37  6.72  3.96  5.49  6.36  7.12  6.26  5.05  5.90  6.23  8.59  6.76  5.37  7.83  5.67
## 24 3.47  5.31  4.47  2.11  4.54  3.71  4.81  4.00  3.91  5.04  7.90  5.09  2.99  5.85  5.81
## 25 3.37  4.17  4.78  2.92  4.59  3.75  4.19  5.03  3.65  5.16  7.51  6.13  4.42  4.90  5.82
## 26 6.17  5.76  7.12  6.23  5.89  8.60  6.76  6.98  5.66  4.18  7.29  4.31  7.44  7.68  3.24
## 27 4.68  5.57  5.89  3.95  5.55  6.49  3.66  4.65  2.93  5.19  8.63  6.26  5.96  6.14  6.52
## 28 6.24  7.31  7.09  7.46  6.48  7.17  5.44  6.42  5.21  7.79  9.45  9.25  7.11  7.27  8.84
## 29 4.96  6.67  6.61  6.09  6.24  6.39  3.65  5.37  3.74  6.63  8.53  8.53  6.30  6.18  8.27
## 30 3.63  5.44  6.33  5.08  5.48  6.00  3.15  5.72  4.06  5.89  7.31  7.40  6.66  4.76  7.31
##      16      17      18      19      20      21      22      23      24      25      26      27      28      29      30
## 1  3.91  3.49  6.32  3.92  2.79  4.26  3.28  6.37  3.47  3.37  6.17  4.68  6.24  4.96  3.63
## 2  5.73  4.76  4.96  4.87  3.57  4.89  3.90  6.72  5.31  4.17  5.76  5.57  7.31  6.67  5.44
## 3  3.58  3.38  8.16  5.17  6.17  4.65  4.84  3.96  4.47  4.78  7.12  5.89  7.09  6.61  6.33
## 4  3.47  2.26  6.98  4.07  4.55  3.46  2.55  5.49  2.11  2.92  6.23  3.95  7.46  6.09  5.08
## 5  5.39  4.42  5.60  4.00  3.80  5.12  3.73  6.36  4.54  4.59  5.89  5.55  6.48  6.24  5.48
## 6  4.51  4.55  8.09  5.59  5.20  3.06  5.30  7.12  3.71  3.75  8.60  6.49  7.17  6.39  6.00
## 7  4.54  4.40  5.79  4.24  3.15  5.07  4.22  6.26  4.81  4.19  6.76  3.66  5.44  3.65  3.15
## 8  2.95  3.82  8.15  6.04  5.98  4.96  4.64  5.05  4.00  5.03  6.98  4.65  6.42  5.37  5.72
## 9  4.15  3.85  6.06  4.63  3.70  4.83  3.02  5.90  3.91  3.65  5.66  2.93  5.21  3.74  4.06
```

```
## 10 5.47 5.04 7.37 5.61 4.62 5.16 4.07 6.23 5.04 5.16 4.18 5.19 7.79 6.63 5.89
## 11 7.62 7.63 9.10 7.80 6.42 7.14 7.53 8.59 7.90 7.51 7.29 8.63 9.45 8.53 7.31
## 12 5.60 4.96 8.26 6.71 6.81 6.74 3.70 6.76 5.09 6.13 4.31 6.26 9.25 8.53 7.40
## 13 4.51 4.08 8.53 4.31 5.62 4.46 4.55 5.37 2.99 4.42 7.44 5.96 7.11 6.30 6.66
## 14 6.25 5.32 3.86 5.34 3.01 5.11 5.57 7.83 5.85 4.90 7.68 6.14 7.27 6.18 4.76
## 15 5.94 5.32 7.64 6.34 6.32 5.86 4.18 5.67 5.81 5.82 3.24 6.52 8.84 8.27 7.31
## 16 0.00 2.24 8.37 4.97 5.58 4.56 3.34 4.75 3.38 4.05 6.44 4.25 7.22 5.92 4.51
## 17 2.24 0.00 6.90 3.66 4.48 3.76 2.59 4.56 2.55 3.18 5.81 4.23 7.28 6.12 4.60
## 18 8.37 6.90 0.00 6.52 4.82 6.73 6.97 8.30 7.45 6.46 8.06 6.83 7.37 6.70 6.86
## 19 4.97 3.66 6.52 0.00 3.42 4.71 3.85 5.61 4.00 3.69 6.98 5.47 7.00 5.88 4.88
## 20 5.58 4.48 4.82 3.42 0.00 4.26 4.53 7.07 4.66 3.66 7.08 5.25 6.42 5.15 3.98
## 21 4.56 3.76 6.73 4.71 4.26 0.00 4.37 6.02 3.67 2.63 7.40 5.64 7.07 6.26 5.81
## 22 3.34 2.59 6.97 3.85 4.53 4.37 0.00 4.90 2.73 3.27 4.73 3.82 7.09 5.86 4.75
## 23 4.75 4.56 8.30 5.61 7.07 6.02 4.90 0.00 5.53 6.04 5.68 5.60 6.66 6.96 7.54
## 24 3.38 2.55 7.45 4.00 4.66 3.67 2.73 5.53 0.00 3.28 6.25 4.52 7.59 6.17 5.25
## 25 4.05 3.18 6.46 3.69 3.66 2.63 3.27 6.04 3.28 0.00 7.02 4.79 6.74 5.53 4.71
## 26 6.44 5.81 8.06 6.98 7.08 7.40 4.73 5.68 6.25 7.02 0.00 6.18 9.12 8.42 7.90
## 27 4.25 4.23 6.83 5.47 5.25 5.64 3.82 5.60 4.52 4.79 6.18 0.00 6.36 4.17 4.34
## 28 7.22 7.28 7.37 7.00 6.42 7.07 7.09 6.66 7.59 6.74 9.12 6.36 0.00 3.62 7.28
## 29 5.92 6.12 6.70 5.88 5.15 6.26 5.86 6.96 6.17 5.53 8.42 4.17 3.62 0.00 4.63
## 30 4.51 4.60 6.86 4.88 3.98 5.81 4.75 7.54 5.25 4.71 7.90 4.34 7.28 4.63 0.00
```

```
# Question 1 suite
# CAH avec la matrice non normalisée
cah_single_euclidean <- hclust(dist(euro_data, method = "euclidean"), method = "single")

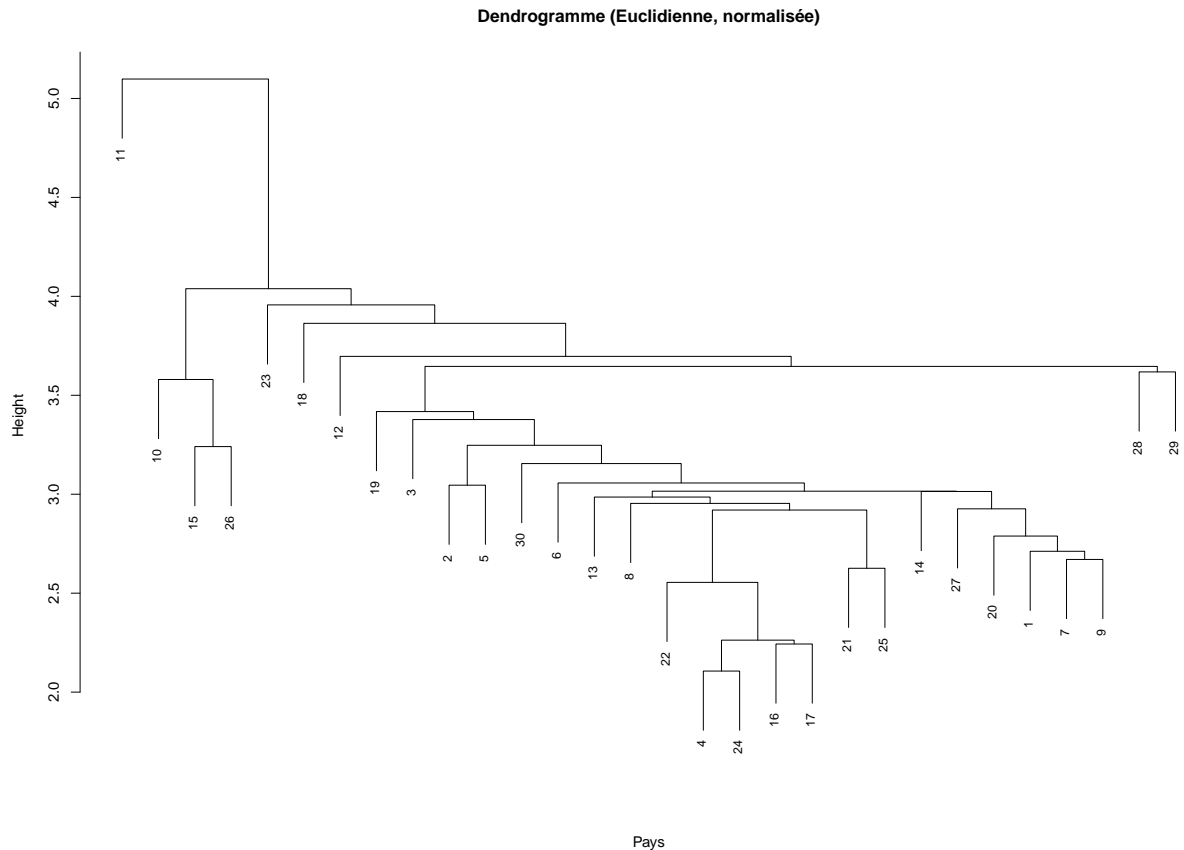
# CAH avec la matrice normalisée
cah_single_reduced <- hclust(dist(euro_data_normalized, method = "euclidean"), method = "single")

# Dendrogramme pour les données non normalisées
plot(cah_single_euclidean,
     main = "Dendrogramme (Euclidienne, non normalisée)",
     xlab = "Pays", sub = "", cex = 0.8)
```



```
# Dendrogramme pour les données normalisées
plot(cah_single_reduced,
     main = "Dendrogramme (Euclidienne, normalisée)",
     xlab = "Pays", sub = "", cex = 0.8)
```





Les dendrogrammes obtenus à partir des matrices de dissimilarité Euclidienne normalisée et non normalisée mettent en évidence des différences dans les regroupements des pays. Avec la matrice normalisée, les échelles des variables sont équilibrées, ce qui permet une contribution équitable de chaque caractéristique aux regroupements.

Cela se traduit par une structure de classification où les similarités relatives entre les pays sont mieux prises en compte. En revanche, la matrice non normalisée amplifie l'influence des variables à grande échelle, ce qui peut biaiser les regroupements et privilégier certaines caractéristiques au détriment d'autres.

Les fusions à des hauteurs plus élevées dans le dendrogramme non normalisé indiquent des dissimilarités globales plus marquées. Cette comparaison souligne l'importance de la normalisation pour éviter des biais et obtenir des regroupements reflétant plus fidèlement les similarités relatives entre les pays.

#### *# Question 2*

*#2. Représentez le dendrogramme ainsi que la "hauteur" (attribut height) en fonction du nombre de classes. #représente la "hauteur" ici ? Du couperiez-vous le dendrogramme ?*

*# Récupérer les hauteurs des fusions*

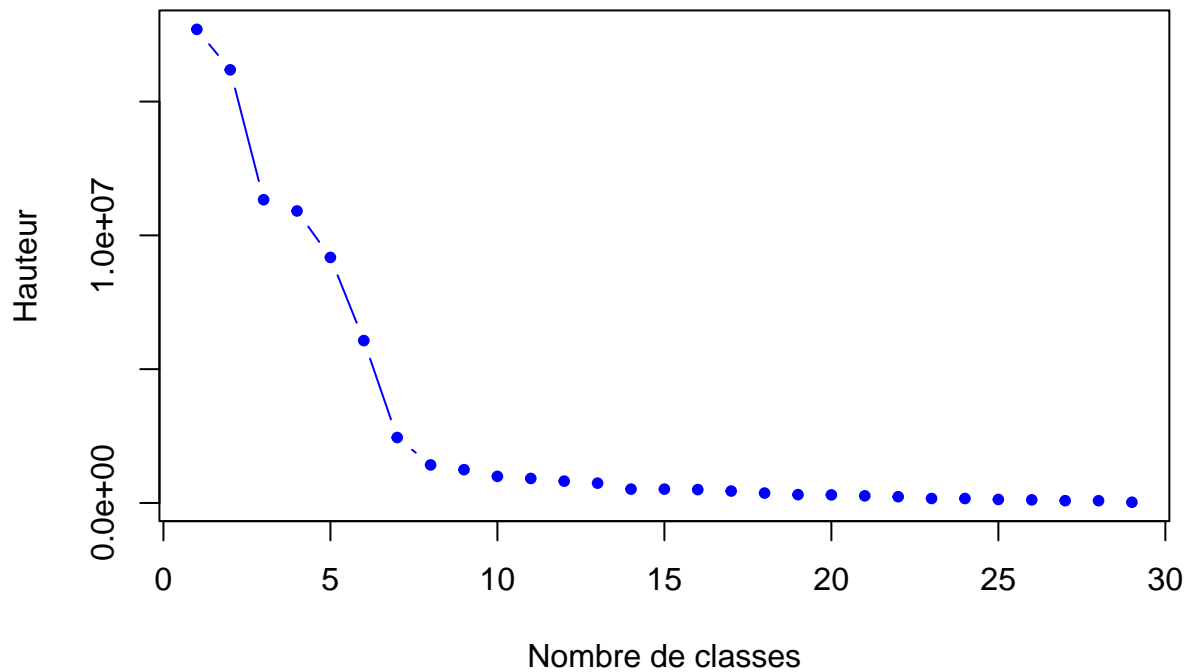
```
heights_euclidean <- sort(cah_single_euclidean$height, decreasing = TRUE)
```

```
heights_reduced <- sort(cah_single_reduced$height, decreasing = TRUE)
```

*# Courbe des hauteurs pour la matrice non normalisée*

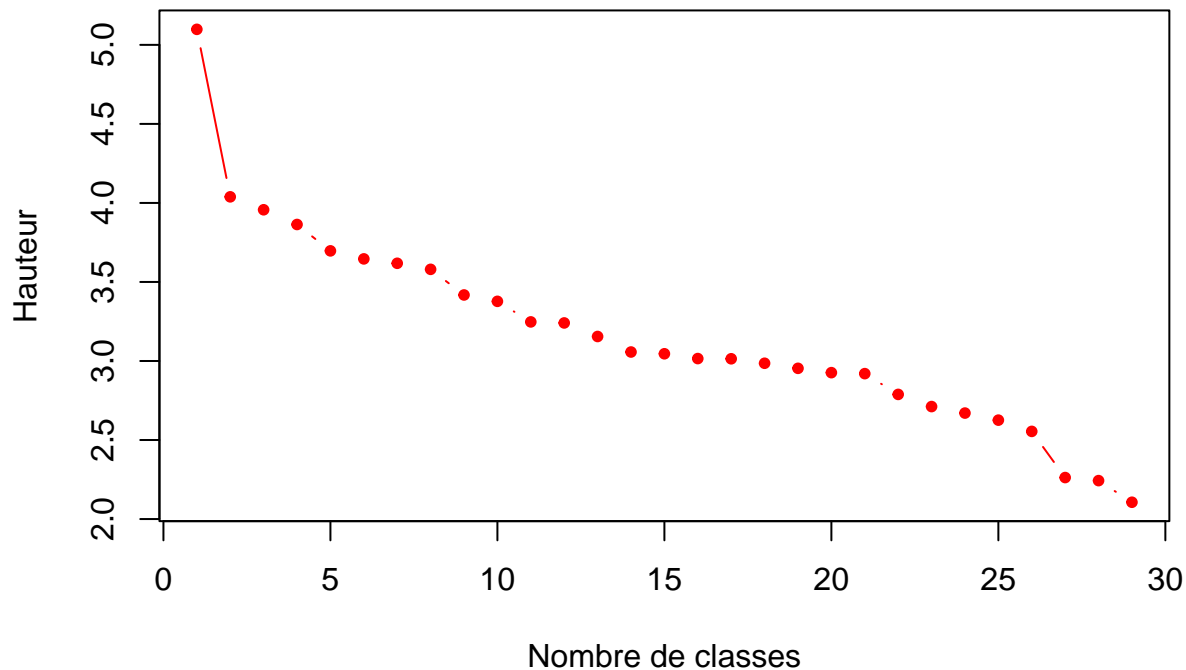
```
plot(1:length(heights_euclidean), heights_euclidean,
     type = "b", col = "blue", pch = 20,
     xlab = "Nombre de classes",
     ylab = "Hauteur",
     main = "Hauteur en fonction du nombre de classes (Euclidienne, non normalisée))")
```

## Hauteur en fonction du nombre de classes (Euclidienne, non normalis)



```
# Courbe des hauteurs pour la matrice normalisée
plot(1:length(heights_reduced), heights_reduced,
     type = "b", col = "red", pch = 20,
     xlab = "Nombre de classes",
     ylab = "Hauteur",
     main = "Hauteur en fonction du nombre de classes (Euclidienne, normalisée)")
```

## Hauteur en fonction du nombre de classes (Euclidienne, normalisée)



Les deux graphiques montrent la relation entre le nombre de classes et la hauteur des fusions dans le dendrogramme pour des matrices de dissimilarité euclidiennes, normalisées et non normalisées.

Représentation des dendrogrammes et des hauteurs : Les dendrogrammes illustrent les regroupements hiérarchiques des observations en fonction des dissimilarités. La hauteur des branches représente la distance ou la dissimilarité entre les clusters fusionnés. Plus la hauteur est élevée, plus les groupes fusionnés sont dissimilaires.

Interprétation de la hauteur : La hauteur dans ces graphiques correspond à la dissimilarité mesurée entre les clusters au moment de leur fusion. Dans le cas de la matrice normalisée, les contributions des différentes variables sont équilibrées, tandis que pour la matrice non normalisée, certaines variables peuvent dominer les regroupements en raison de leurs échelles.

Découpage du dendrogramme : Le point de découpe optimal dépend du contexte de l'analyse, mais il est souvent déterminé en identifiant une hauteur au-delà de laquelle les fusions sont moins significatives (par exemple, un "saut" important dans la hauteur des fusions). Pour le dendrogramme basé sur la matrice normalisée, un découpage à environ 3 ou 4 clusters pourrait être justifié, car les différences de hauteur sont notables à ces niveaux. Pour le dendrogramme non normalisé, un découpage similaire pourrait être envisagé, mais avec prudence, car les fusions pourraient être influencées par des variables dominantes.

### # Question 3 : Caractérisation des classes

# Étape 1 : Couper le dendrogramme pour obtenir les classes

# Utiliser par exemple 4 classes

```
classes <- cutree(cah_single_euclidean, k = 4) # Modifier k selon le nombre souhaité
```

# Étape 2 : Calculer les centres de gravité

```
centers <- aggregate(euro_data, by = list(Classe = classes), FUN = mean)
```

```

# Étape 3 : Calculer l'inertie intra-classe
inertia <- sapply(unique(classes), function(k) {
  members <- euro_data[classes == k, ] # Individus dans la classe k
  center <- colMeans(members)          # Centre de gravité de la classe k
  sum(rowSums((members - center)^2))    # Somme des carrés des distances
})

# Assigner des noms aux inerties pour chaque classe
names(inertia) <- paste("Classe", unique(classes))

# Étape 4 : Afficher les résultats
cat("Centres de gravité des classes :\n")

```

```
## Centres de gravité des classes :
```

```
print(centers)
```

```
## Classe Population Youth.population First.time.asylum.applicants
## 1      1      6685083      16.85600      12510.0
## 2      2      58418513      16.13333      145373.3
## 3      3      84358845      16.00000      329035.0
## 4      4      36753736      15.50000      7720.0
## Gender.pay.gap Minimum.wage People.at.risk.of.poverty.or.exclusion
## 1      12.668000      NA      NA
## 2      8.966667      1413.333      23.7
## 3      17.700000      2054.000      21.1
## 4      7.800000      977.000      15.9
## Early.school.leavers Inflation.rate Unemployment.rate Youth.unemployment.rate
## 1      8.268      6.98      5.348000      14.48000
## 2      10.600      5.00      9.066667      22.86667
## 3      12.800      6.00      3.100000      5.90000
## 4      3.700      10.90      2.800000      11.40000
## GDP.per.capita Government.gross.debt Greenhouse.gas.emissions
## 1      33038      58.1400      8.356000
## 2      29160      118.5333      6.533333
## 3      36290      63.6000      9.300000
## 4      14750      49.6000      10.400000
## Renewable.energy Electricity.prices Energy.imports.dependency
## 1      32.54      227.8080      57.15600
## 2      20.50      289.4667      68.46667
## 3      20.80      416.2000      68.60000
## 4      16.90      229.1000      46.00000

```

```
cat("\nInerties intra-classes :\n")
```

```
##
## Inerties intra-classes :
```

```
print(inertia)
```

```
## Classe 1      Classe 2      Classe 3      Classe 4
##      NA 1.269667e+16 0.000000e+00 0.000000e+00

```

```

# Étape 5 : Visualisation des classes
# Ajouter une colonne "Classe" aux données pour la visualisation
euro_data_with_classes <- euro_data
euro_data_with_classes$Classe <- as.factor(classes)

# Afficher les données avec les classes
cat("\nAperçu des données avec les classes :\n")

```

```

##
## Aperçu des données avec les classes :

```

```

print(head(euro_data_with_classes))

```

```

##      Population Youth.population First.time.asylum.applicants Gender.pay.gap
## 1      9104772              16.9              56135              18.4
## 2     11742796              17.8              29260               5.0
## 3      6447710              13.2              22390              13.0
## 4      3850894              15.9               1635              12.5
## 5       920701              19.8              11660              10.2
## 6     10827529              15.1               1130              17.9
##      Minimum.wage People.at.risk.of.poverty.or.exclusion Early.school.leavers
## 1           1766              17.5              8.6
## 2           1994              18.7              6.2
## 3            477              32.2              9.3
## 4            840              19.9              2.0
## 5           1000              16.7             10.5
## 6            764              11.8              6.4
##      Inflation.rate Unemployment.rate Youth.unemployment.rate GDP.per.capita
## 1              7.7              5.1              10.4             37460
## 2              2.3              5.5              16.1             37300
## 3              8.6              4.3              12.1              7850
## 4              8.4              6.1              19.0             14750
## 5              3.9              6.1              16.9             27720
## 6             14.8              2.6               8.3             18480
##      Government.gross.debt Greenhouse.gas.emissions Renewable.energy
## 1              77.8              8.3              33.8
## 2             105.2              9.3              13.8
## 3              23.1              9.1              19.1
## 4              63.0              6.8              27.9
## 5              77.3             10.5              19.4
## 6              44.0             11.1             18.2
##      Electricity.prices Energy.imports.dependency Classe
## 1              288.5              74.5           1
## 2              377.2              74.0           1
## 3              119.4              37.1           1
## 4              154.3              60.3           1
## 5              351.9              92.0           1
## 6              303.9              41.8           1

```

```

# Calculer l'inertie intra-classe
# Question 3 suite
calculate_inertia <- function(data, centers, classes) {

```

```

inertia <- 0
for (class in unique(classes)) {
  class_data <- data[classes == class, ]
  class_center <- centers[class, -ncol(centers)] # Exclure la colonne des classes
  inertia <- inertia + sum(rowSums((class_data - class_center)^2))
}
return(inertia)
}

```

*# Inertie totale*

```

inertia <- calculate_inertia(euro_data_normalized, centers, classes)
cat("\nInertie intra-classe totale : ", round(inertia, 2), "\n")

```

##

## Inertie intra-classe totale : NA

*# Question 4*

*# Classification ascendante hiérarchique avec le critère de Ward*

```

cah_ward <- hclust(dist(euro_data_normalized, method = "euclidean"), method = "ward.D2")

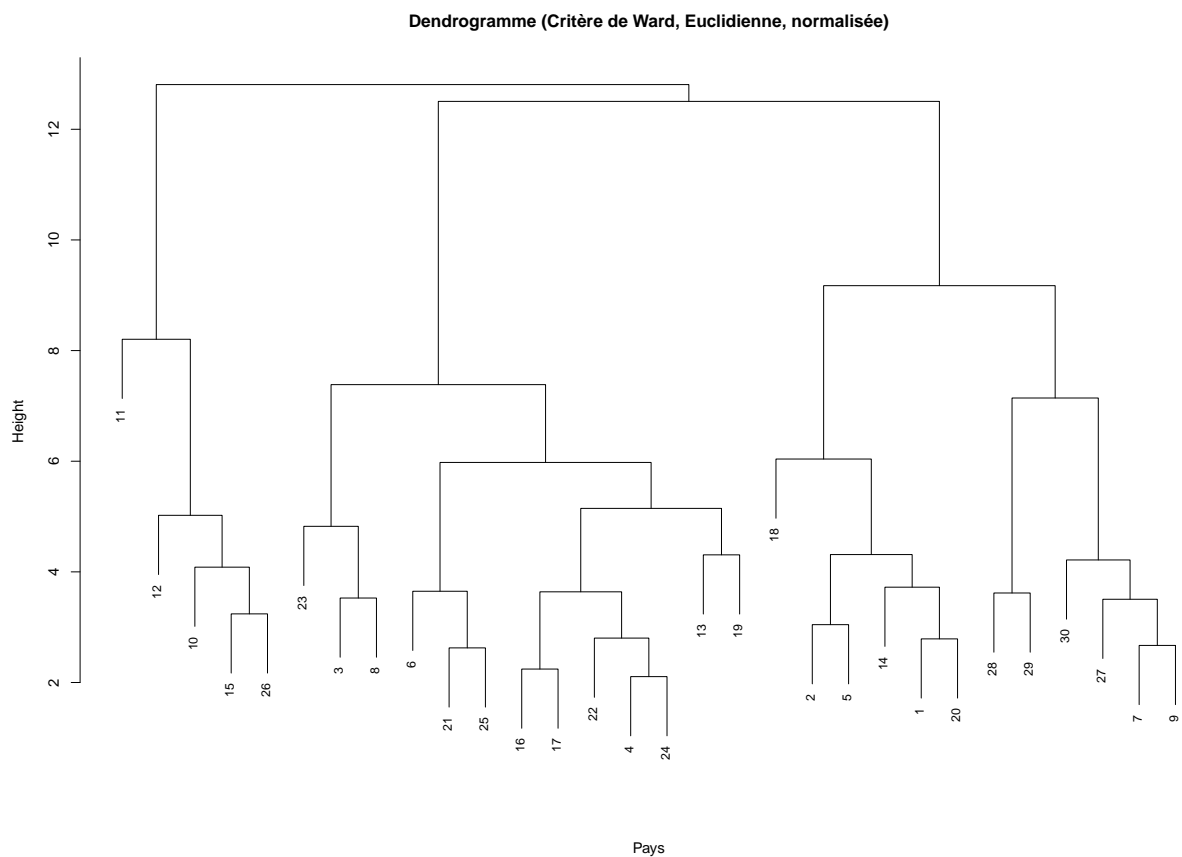
```

*# Représenter le dendrogramme*

```

plot(cah_ward,
     main = "Dendrogramme (Critère de Ward, Euclidienne, normalisée)",
     xlab = "Pays", sub = "", cex = 0.8)

```



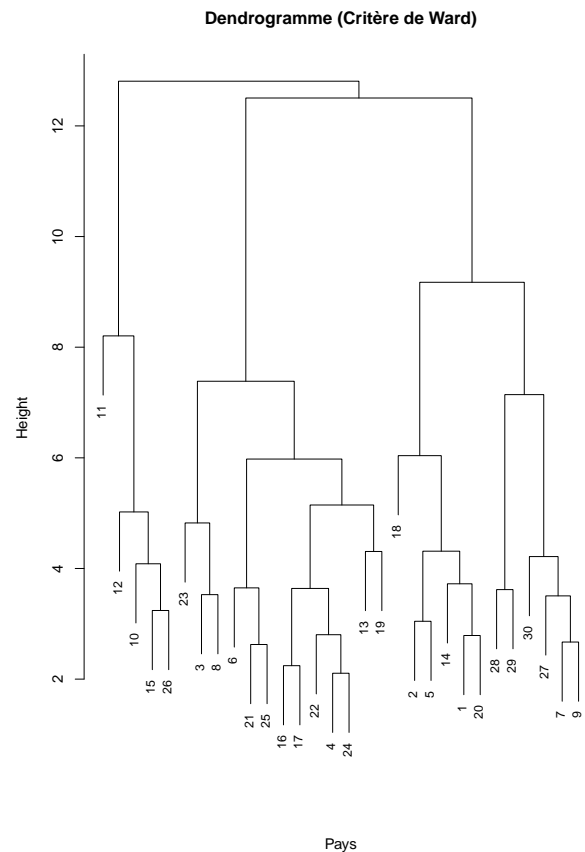
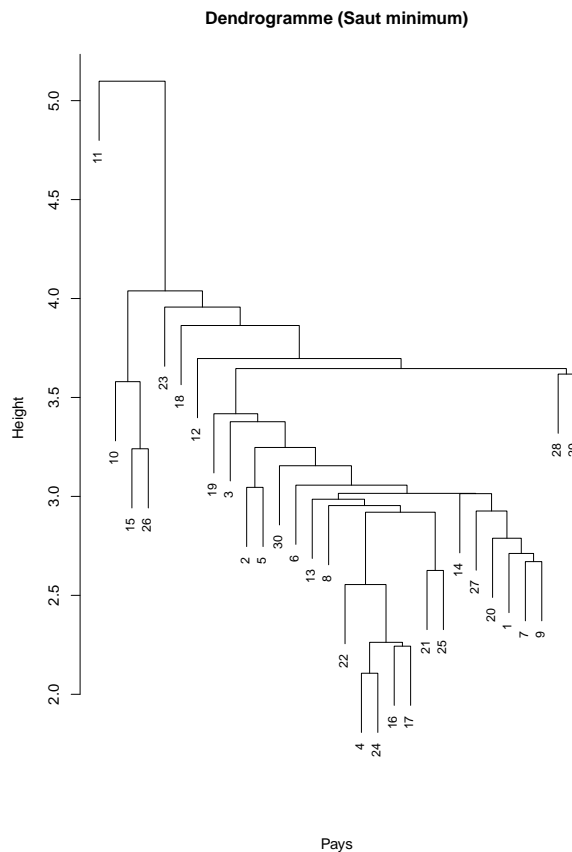
```

par(mfrow = c(1, 2)) # Afficher les deux dendrogrammes côte à côte

# Dendrogramme avec le saut minimum
plot(cah_single_reduced,
     main = "Dendrogramme (Saut minimum)",
     xlab = "Pays", sub = "", cex = 0.8)

# Dendrogramme avec Ward
plot(cah_ward,
     main = "Dendrogramme (Critère de Ward)",
     xlab = "Pays", sub = "", cex = 0.8)

```



```

par(mfrow = c(1, 1)) # Réinitialiser l'affichage

```

```

# Question 4 Effectuez une autre classification en utilisant le critère de Ward. Commentez les différences
# Découpage en classes avec les deux méthodes
classes_single <- cutree(cah_single_reduced, k = 4)
classes_ward <- cutree(cah_ward, k = 4)

# Ajouter les classes aux données
euro_data_with_classes <- data.frame(euro_data_normalized, Classe_Single = classes_single, Classe_Ward = classes_ward)

# Afficher un aperçu
head(euro_data_with_classes)

```

```
## Population Youth.population First.time.asylum.applicants Gender.pay.gap
## 1 -0.2941425 0.1052162 0.28349361 1.20451373
## 2 -0.1718499 0.6036090 -0.09845354 -1.44291393
## 3 -0.4173177 -1.9437317 -0.19608989 0.13763990
## 4 -0.5377000 -0.4485535 -0.49105968 0.03885528
## 5 -0.6735369 1.7111484 -0.34858451 -0.41555394
## 6 -0.2142795 -0.8915692 -0.49823673 1.10572911
## Minimum.wage People.at.risk.of.poverty.or.exclusion Early.school.leavers
## 1 0.8677402 -0.5670191 0.02822593
## 2 1.2628220 -0.3346014 -0.64919646
## 3 -1.3658578 2.2800979 0.22580746
## 4 -0.7368461 -0.1021837 -1.83468565
## 5 -0.4595957 -0.7219642 0.56451866
## 6 -0.8685400 -1.6710032 -0.59274459
## Inflation.rate Unemployment.rate Youth.unemployment.rate GDP.per.capita
## 1 0.2355764 -0.20470989 -0.7706675 0.2660926
## 2 -1.3157803 -0.02670129 0.1990466 0.2580762
## 3 0.4941358 -0.56072710 -0.4814545 -1.2174325
## 4 0.4366782 0.24031161 0.6924098 -0.8717275
## 5 -0.8561190 0.24031161 0.3351468 -0.2219025
## 6 2.2753231 -1.31726367 -1.1279305 -0.6848465
## Government.gross.debt Greenhouse.gas.emissions Renewable.energy
## 1 0.38958609 0.01005534 0.1781191
## 2 1.16743348 0.38713049 -0.8768806
## 3 -1.16326982 0.31171546 -0.5973057
## 4 -0.03056505 -0.55555739 -0.1331058
## 5 0.37539180 0.83962068 -0.5814807
## 6 -0.56994828 1.06586577 -0.6447807
## Electricity.prices Energy.imports.dependency Classe_Single Classe_Ward
## 1 0.5788290 0.66564454 1 1
## 2 1.6439448 0.64510418 1 1
## 3 -1.4517357 -0.87077425 1 2
## 4 -1.0326540 0.08229837 1 2
## 5 1.3401407 1.38455708 1 1
## 6 0.7637533 -0.67769488 1 2
```

```
# Fixer le nombre de classes
# question 5
# Vérifier les NA ou NaN dans les données normalisées
cat("Y a-t-il des NA/NaN dans les données ?\n")
```

```
## Y a-t-il des NA/NaN dans les données ?
```

```
print(any(is.na(euro_data_normalized)))
```

```
## [1] TRUE
```

```
# Si des NA sont présents, afficher leur localisation
if (any(is.na(euro_data_normalized))) {
  cat("Position des NA/NaN :\n")
  print(which(is.na(euro_data_normalized), arr.ind = TRUE))
  euro_data_normalized <- apply(euro_data_normalized, 2, function(x) {
```



```

    ifelse(is.na(x), mean(x, na.rm = TRUE), x)
  })
print(any(is.na(euro_data_normalized)))
}

```

```

## Position des NA/NaN :
##      row col
## [1,]  27   5
## [2,]  28   5
## [3,]  29   5
## [4,]  30   5
## [5,]  28   6
## [1] FALSE

```

```

euro_data_normalized <- apply(euro_data_normalized, 2, function(x) {
  ifelse(is.na(x), mean(x, na.rm = TRUE), x)
})
print(any(is.na(euro_data_normalized)))

```

```
## [1] FALSE
```

```

# Suite question 5
# Fixer le nombre de classes
k <- 4

# Appliquer k-means
set.seed(42) # Fixer la graine pour rendre les résultats reproductibles
kmeans_result <- kmeans(euro_data_normalized, centers = k, nstart = 10)

# Afficher les résultats
cat("Classes affectées par k-means :\n")

```

```
## Classes affectées par k-means :
```

```
print(kmeans_result$cluster)
```

```
## [1] 3 2 4 4 2 4 3 4 3 1 1 1 4 2 1 4 4 2 4 2 4 4 4 4 1 3 3 3 3
```

```
cat("\nCentres des classes :\n")
```

```
##
## Centres des classes :
```

```
print(round(kmeans_result$centers, 2))
```

```

##      Population Youth.population First.time.asylum.applicants Gender.pay.gap
## 1      1.79                -0.44                1.83                -0.08
## 2     -0.38                 1.10               -0.24               -0.98
## 3     -0.41                 0.79               -0.30                0.41

```

```
## 4      -0.32      -0.68      -0.45      0.19
## Minimum.wage People.at.risk.of.poverty.or.exclusion Early.school.leavers
## 1      0.30      0.63      0.33
## 2      1.20      -0.44      -0.50
## 3      0.34      -0.50      0.42
## 4      -0.76      0.19      -0.16
## Inflation.rate Unemployment.rate Youth.unemployment.rate GDP.per.capita
## 1      -0.53      1.21      0.90      -0.18
## 2      -0.92      -0.28      -0.13      1.04
## 3      -0.44      -0.17      -0.40      0.88
## 4      0.79      -0.27      -0.08      -0.81
## Government.gross.debt Greenhouse.gas.emissions Renewable.energy
## 1      1.48      -0.34      -0.50
## 2      -0.12      1.16      -0.81
## 3      -0.34      -0.03      1.47
## 4      -0.34      -0.29      -0.29
## Electricity.prices Energy.imports.dependency
## 1      0.79      0.51
## 2      0.74      1.03
## 3      -0.26      -0.72
## 4      -0.45      -0.20
```

```
cat("\nInertie intra-classe totale :\n")
```

```
##
## Inertie intra-classe totale :
```

```
print(round(kmeans_result$tot.withinss, 2))
```

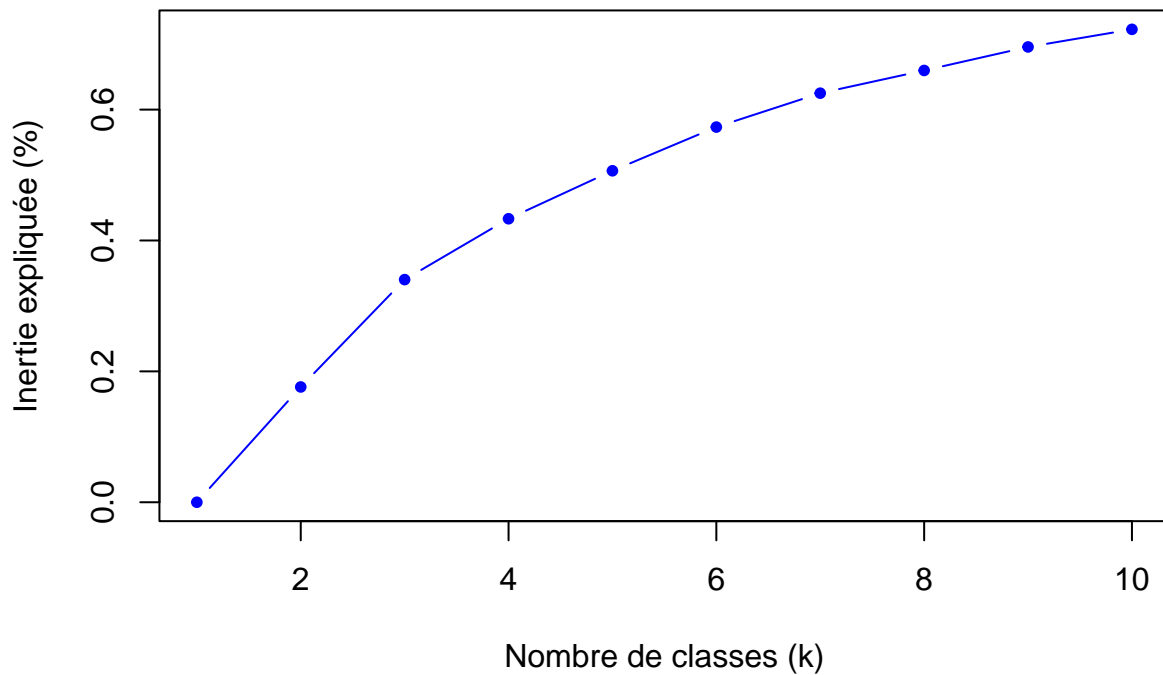
```
## [1] 260.15
```

```
# Initialiser les variables
max_k <- 10 # Tester jusqu'à 10 classes
inertie_totale <- sum(scale(euro_data_normalized, center = TRUE, scale = FALSE)^2)
inertie_intra <- numeric(max_k)
inertie_expliquee <- numeric(max_k)

# Calculer l'inertie intra-classe pour chaque k
for (k in 1:max_k) {
  set.seed(42) # Graine pour reproductibilité
  kmeans_result <- kmeans(euro_data_normalized, centers = k, nstart = 10)
  inertie_intra[k] <- kmeans_result$tot.withinss
  inertie_expliquee[k] <- 1 - (inertie_intra[k] / inertie_totale)
}

# Tracer la courbe
plot(1:max_k, inertie_expliquee, type = "b", pch = 20, col = "blue",
     xlab = "Nombre de classes (k)", ylab = "Inertie expliquée (%)",
     main = "Inertie expliquée en fonction du nombre de classes")
```

## Inertie expliquée en fonction du nombre de classes



```
# Question 7
# Étape 1 : Effectuer l'ACP
# Effectuer l'ACP sur les données normalisées
res_acp <- prcomp(euro_data_normalized, center = TRUE, scale. = TRUE)

# Résumé de l'ACP pour comprendre les variances expliquées
cat("Résumé de l'ACP :\n")
```

```
## Résumé de l'ACP :
```

```
print(summary(res_acp))
```

```
## Importance of components:
##              PC1      PC2      PC3      PC4      PC5      PC6      PC7
## Standard deviation    1.9068 1.8769 1.4265 1.3623 1.13291 0.93197 0.86710
## Proportion of Variance 0.2272 0.2202 0.1272 0.1160 0.08022 0.05428 0.04699
## Cumulative Proportion 0.2272 0.4474 0.5746 0.6906 0.77079 0.82507 0.87207
##              PC8      PC9      PC10     PC11     PC12     PC13     PC14
## Standard deviation    0.74801 0.66980 0.58743 0.45336 0.41016 0.39737 0.30054
## Proportion of Variance 0.03497 0.02804 0.02157 0.01285 0.01051 0.00987 0.00565
## Cumulative Proportion 0.90704 0.93508 0.95664 0.96949 0.98000 0.98987 0.99552
##              PC15     PC16
## Standard deviation    0.2040 0.17357
## Proportion of Variance 0.0026 0.00188
## Cumulative Proportion 0.9981 1.00000
```

```

# Étape 2 : Récupérer les résultats de classification
# Fixer le nombre de classes optimal
k <- 4 # Ajuster selon les résultats précédents

# Classes obtenues par CAH
classes_cah <- cutree(cah_single_reduced, k = k)

# Classes obtenues par k-means
classes_kmeans <- kmeans_result$cluster

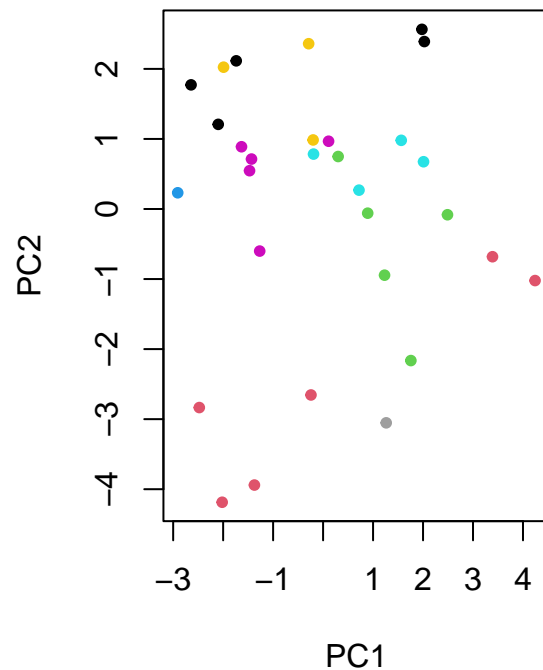
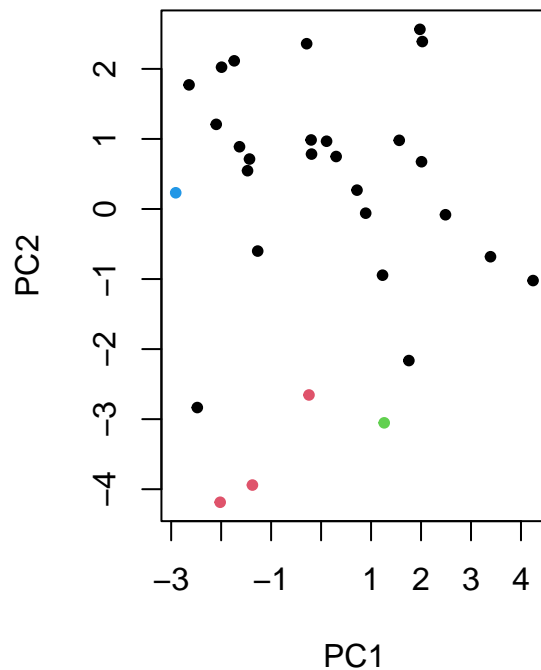
# Étape 3 : Représenter les classes dans le plan factoriel
par(mfrow = c(1, 2)) # Afficher les deux graphiques côte à côte

# Représentation des classes CAH
plot(res_acp$x[, 1], res_acp$x[, 2],
     col = classes_cah,
     pch = 20,
     xlab = "PC1", ylab = "PC2",
     main = "Plan factoriel avec classes CAH")

# Représentation des classes k-means
plot(res_acp$x[, 1], res_acp$x[, 2],
     col = classes_kmeans,
     pch = 20,
     xlab = "PC1", ylab = "PC2",
     main = "Plan factoriel avec classes k-means")

```

## Plan factoriel avec classes CAH Plan factoriel avec classes k-means



```
par(mfrow = c(1, 1)) # Réinitialiser l'affichage
```

```
# Étape 1 : Effectuer l'ACP
```

```
# Effectuer l'ACP sur les données normalisées
```

```
res_acp <- prcomp(euro_data_normalized, center = TRUE, scale. = TRUE)
```

```
# Résumé de l'ACP pour comprendre les variances expliquées
```

```
cat("Résumé de l'ACP :\n")
```

```
## Résumé de l'ACP :
```

```
print(summary(res_acp))
```

```
## Importance of components:
```

```
##          PC1    PC2    PC3    PC4    PC5    PC6    PC7
## Standard deviation  1.9068 1.8769 1.4265 1.3623 1.13291 0.93197 0.86710
## Proportion of Variance 0.2272 0.2202 0.1272 0.1160 0.08022 0.05428 0.04699
## Cumulative Proportion 0.2272 0.4474 0.5746 0.6906 0.77079 0.82507 0.87207
##          PC8    PC9    PC10    PC11    PC12    PC13    PC14
## Standard deviation  0.74801 0.66980 0.58743 0.45336 0.41016 0.39737 0.30054
## Proportion of Variance 0.03497 0.02804 0.02157 0.01285 0.01051 0.00987 0.00565
## Cumulative Proportion 0.90704 0.93508 0.95664 0.96949 0.98000 0.98987 0.99552
##          PC15    PC16
## Standard deviation  0.2040 0.17357
```

```
## Proportion of Variance 0.0026 0.00188
## Cumulative Proportion 0.9981 1.00000

# Étape 2 : Récupérer les classes des deux méthodes
# Fixer le nombre optimal de classes (par exemple, k = 4)
k <- 4

# Classes obtenues par CAH
classes_cah <- cutree(cah_single_reduced, k = k)

# Classes obtenues par k-means
classes_kmeans <- kmeans_result$cluster

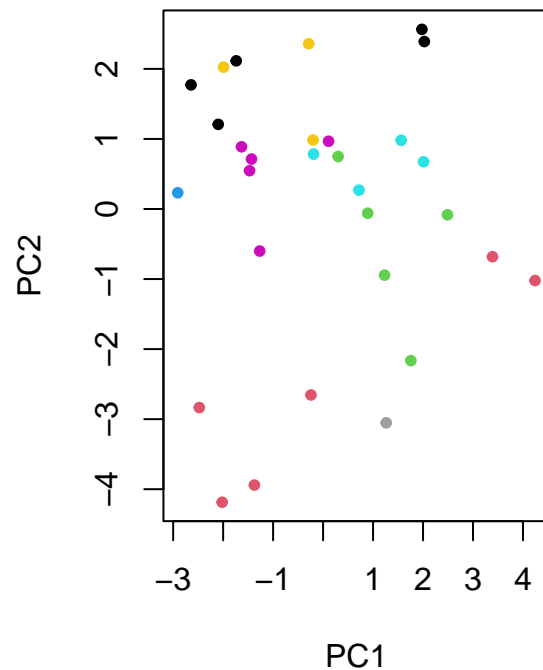
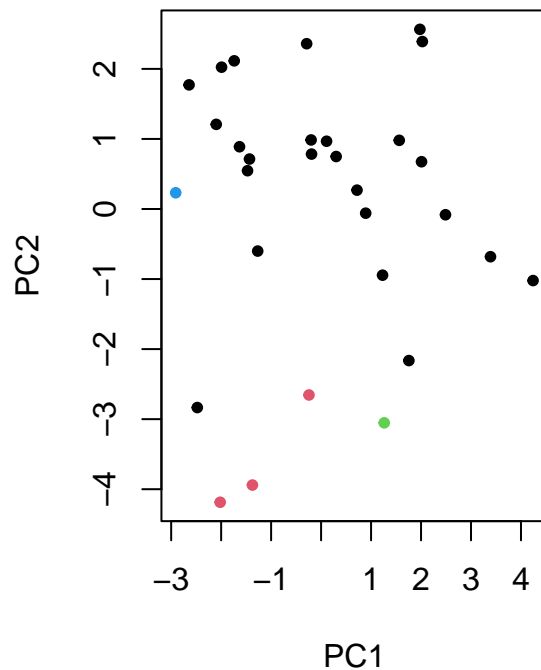
# Étape 3 : Ajouter les classes au tableau des coordonnées de l'ACP
coord_acp <- data.frame(res_acp$x[, 1:2], Classe_CAH = as.factor(classes_cah), Classe_Kmeans = as.factor(classes_kmeans))

# Étape 4 : Représenter les classes dans le plan factoriel
par(mfrow = c(1, 2)) # Afficher les deux graphiques côte à côte

# Représentation des classes CAH
plot(coord_acp$PC1, coord_acp$PC2,
      col = coord_acp$Classe_CAH,
      pch = 20,
      xlab = "PC1", ylab = "PC2",
      main = "Plan factoriel avec classes CAH")

# Représentation des classes k-means
plot(coord_acp$PC1, coord_acp$PC2,
      col = coord_acp$Classe_Kmeans,
      pch = 20,
      xlab = "PC1", ylab = "PC2",
      main = "Plan factoriel avec classes k-means")
```

## Plan factoriel avec classes CAH Plan factoriel avec classes k-means



```
par(mfrow = c(1, 1)) # Réinitialiser l'affichage
```

```
# Étape 1 : Définir une zone restreinte dans le plan factoriel
```

```
zone_restante <- coord_acp[coord_acp$PC1 > -1 & coord_acp$PC1 < 1 & coord_acp$PC2 > -1 & coord_acp$PC2 < 1]
```

```
# Afficher les pays dans la zone restreinte
```

```
cat("Pays dans la zone sélectionnée :\n")
```

```
## Pays dans la zone sélectionnée :
```

```
print(zone_restante)
```

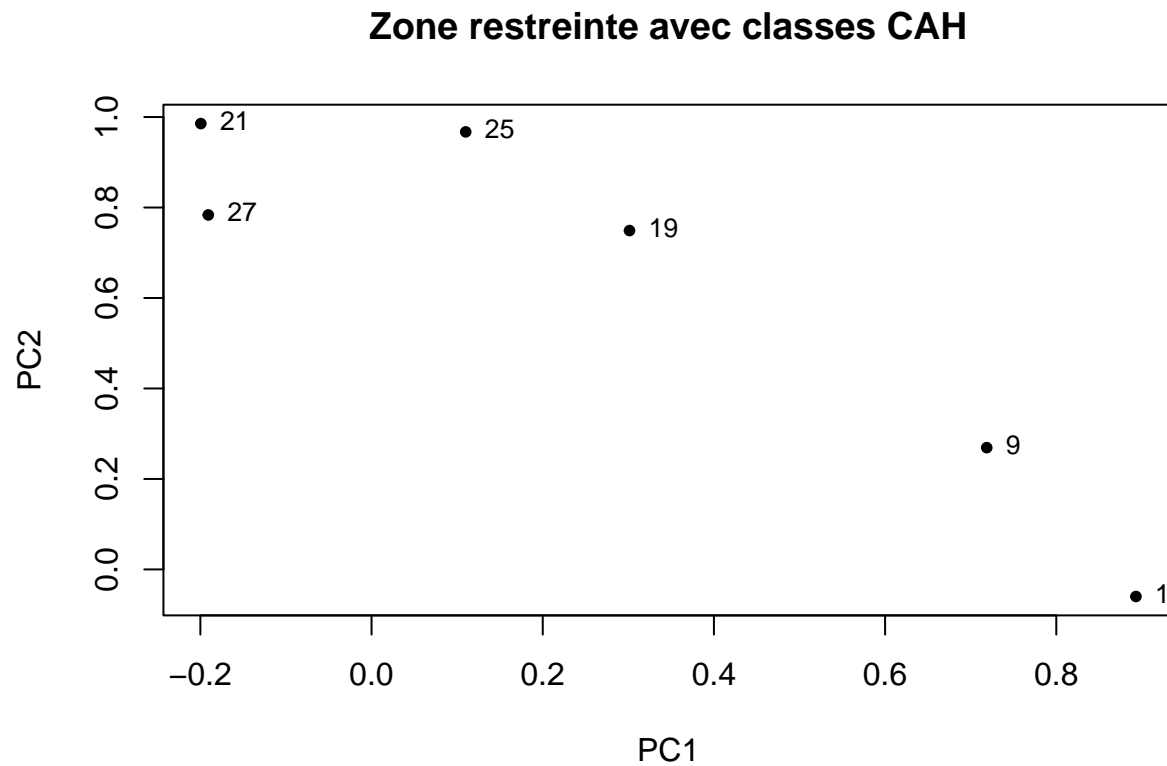
```
##          PC1          PC2 Classe_CAH Classe_Kmeans
## 1  0.8930784 -0.05981786          1           3
## 9  0.7187603  0.26913474          1           5
## 19 0.3013992  0.74908977          1           3
## 21 -0.1994403  0.98555691          1           7
## 25  0.1100162  0.96722107          1           6
## 27 -0.1907443  0.78363267          1           5
```

```
# Étape 2 : Représenter graphiquement la zone
```

```
plot(zone_restante$PC1, zone_restante$PC2,
     col = zone_restante$Classe_CAH,
```

```
pch = 20,
xlab = "PC1", ylab = "PC2",
main = "Zone restreinte avec classes CAH")

# Ajouter les noms des pays
text(zone_restante$PC1, zone_restante$PC2, labels = rownames(zone_restante), pos = 4, cex = 0.8)
```



```
tinytex::is_tinytex()
```

```
## [1] TRUE
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.