Multi-Method Clustering for Socio-Economic Segmentation

Samuele Aglieco

2024-12-22

Prof: Antonio Punzo

Introduction

This project applies a range of clustering techniques to segment countries based on socio-economic indicators, recognizing that these variables do not form perfectly distinct groups. Given the overlapping nature of the data, soft clustering methods are particularly appropriate, as they allow for probabilistic membership rather than rigid assignments.

To capture the underlying structure, model-based clustering is employed through **Gaussian Mixture Models (GMM)** and finite mixture models, leveraging the Expectation-Maximization algorithm for parameter estimation. **Cluster-Weighted Models (CWM)** further refine the analysis by incorporating dependencies between socio-economic variables, enhancing interpretability. In parallel, fuzzy clustering techniques provide an alternative soft classification, offering a flexible approach to partitioning.

Among hard clustering methods, **k-means** has produced a well-defined partition, effectively capturing key patterns in the data, while **hierarchical** approaches proved less effective. By integrating these diverse methodologies, the project ensures a robust and nuanced segmentation, reflecting both the structure and uncertainty inherent in socio-economic data.

Libraries and data import

The data are from kaggle Rohan (2021).

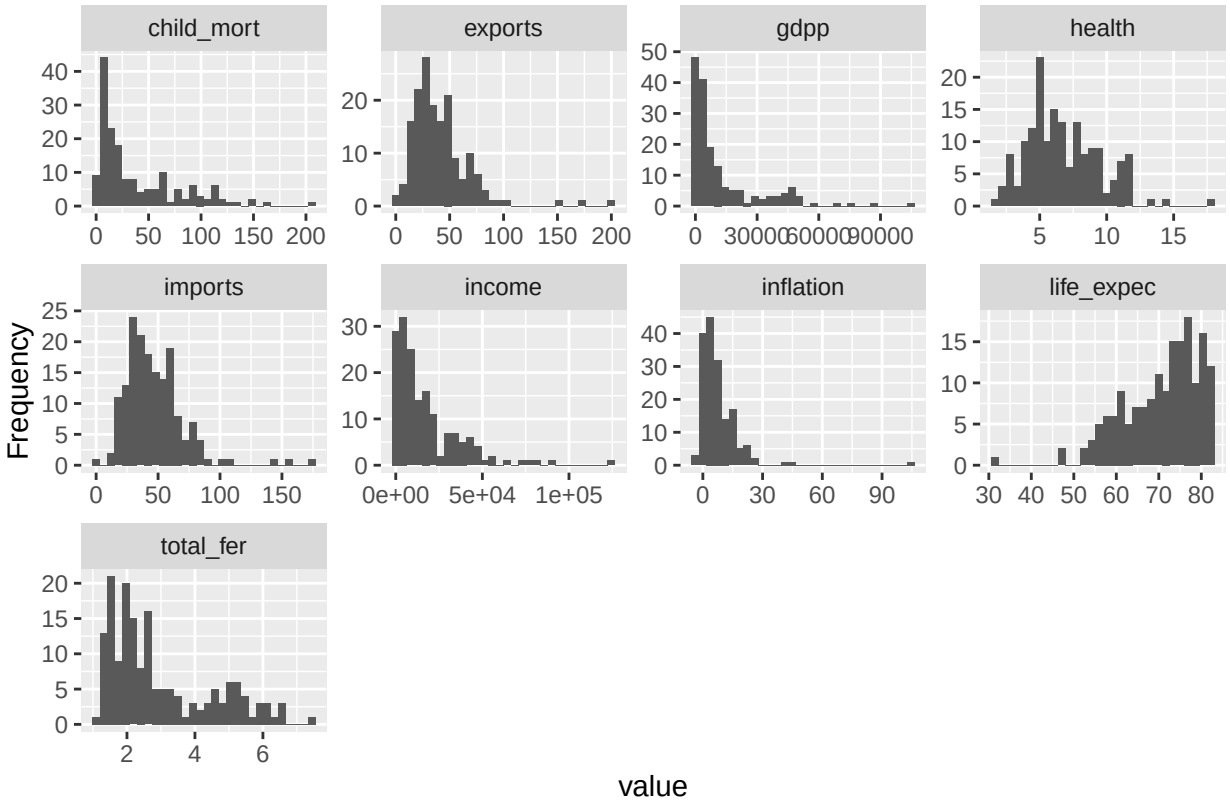
Univariate Analysis

For the univariate analysis, I performed some initial modeling using the `gamlss` and `mixR` packages for Gamma mixture models. However, before diving into the modeling, I first conducted preliminary data visualizations.

Preliminary Data Visualization

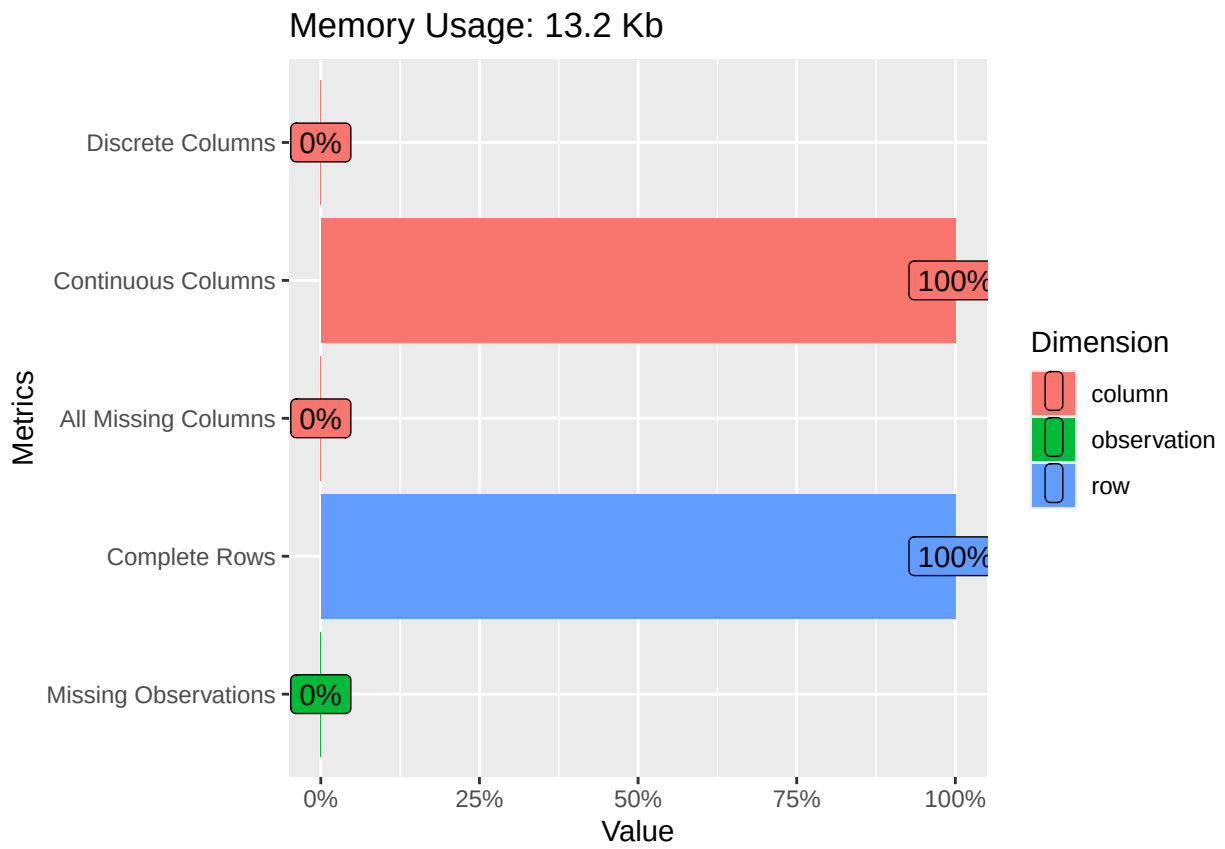
For the initial visualization, I utilized the `DataExplorer` and `visdat` packages. I began with histogram plots to explore the distribution of the data.

```
plot_histogram(data)
```



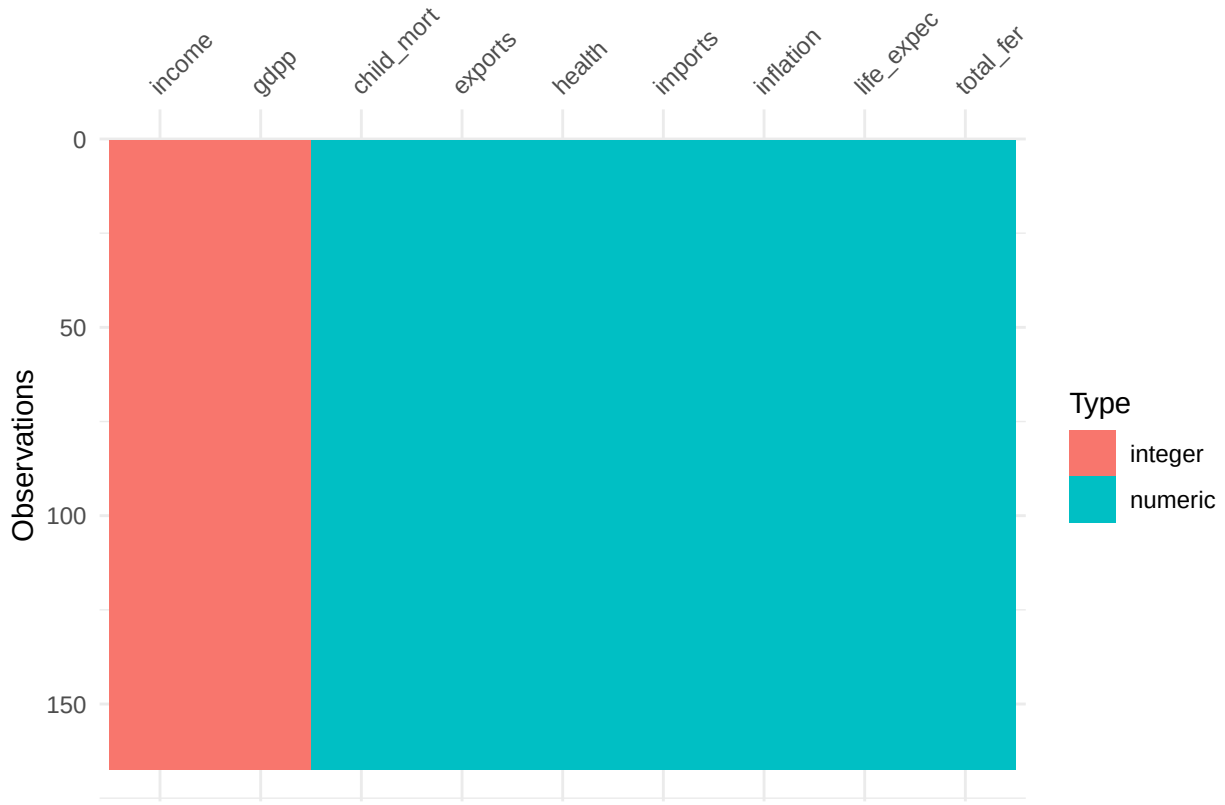
Through this plot, we can visualize a summary of the univariate distribution of the data. Most of the variables appear to be skewed to the left, with the exception of the variable `life_expectancy`, which seems to be skewed to the right. In general, most of the variables are continuously distributed in $\mathbb{R}^+$. In further analysis, I will examine all the models in more detail.

```
plot_intro(data)
```



This plot provides a useful preliminary visualization of the data structure. It reveals that all variables are continuous and there are no missing observations.

```
visdat::vis_dat(data)
```



This plot displays all the data types, which are predominantly numeric. However, the variables `income` and `gdpp` are integers.

I have now analyzed all the univariate distributions, attempting to fit a parametric model for each using the `gamlss` and `mclust` libraries.

Quick Introduction to Generalized Additive Models for Location, Scale, and Shape (GAMLSS)

Before starting the implementation with the `gamlss` package, I found it helpful to first gain a general understanding of these types of models. This allowed me to better interpret and make sense of the output from the functions.

As described in Stasinopoulos et al. (2017), the GAMLSS is an extension of Generalized Linear Models (GLMs) and Generalized Additive Models (GAMs). GAMLSS allows for flexible modeling of not only the location (*mean*) but also the scale (*variance*), *skewness*, and *kurtosis* of a distribution, making it particularly useful for complex datasets with non-normal features such as skewed or heavy-tailed distributions.

The definition of GAMs is:

$$Y \sim \mathcal{E}(\mu, \phi)$$

$$\eta = g(\mu) = X\beta + s_1(x_1) + ... + s_J(x_J)$$

Where s_j is a nonparametric smoothing function applied to covariate x_j , for $j = 1, ..., J$ and ϕ represent others **fixed** paramethers connected with the exponential family distribution ($\mathcal{E}$)

The main limitation of GAMs is that **only** the location parameter μ is modeled as a function of explanatory variables. This is problematic when:

- The **variance** (scale parameter σ) changes with covariates (**heteroscedasticity**).
- The data exhibits **skewness** or **heavy tails** that cannot be captured by a two-parameter distribution.

The GAMLSS Framework:

GAMLSS extends the GAM framework by allowing all four parameters of a flexible distribution to be modeled as functions of covariates:

$$Y \sim D(\mu, \sigma, \nu, \tau)$$

where:

- μ = location (mean or median)
- σ = scale (spread or standard deviation)
- ν = skewness (asymmetry of the distribution)
- τ = kurtosis (tail behavior, peakedness)

Each parameter is linked to covariates through a link function and linear or smooth terms:

$\eta_1 = g_1(\mu) = X_1\beta_1 + s_{11}(x_{11}) + ... + s_{J1}(x_{J1})$

$\eta_2 = g_2(\sigma) = X_2\beta_2 + s_{12}(x_{12}) + ... + s_{J2}(x_{J2})$

$\eta_3 = g_3(\nu) = X_3\beta_3 + s_{13}(x_{13}) + ... + s_{J3}(x_{J3})$

$\eta_4 = g_4(\tau) = X_4\beta_4 + s_{14}(x_{14}) + ... + s_{J4}(x_{J4})$

Where:

- η is a transformed parameter through a **link function**
- $g(\cdot)$ is a link function ensuring parameter constraints (e.g., positivity for variance)
- β is a linear predictor
- sj is a nonparametric smoothing function applied to covariate xj $(j = 1, 2, , ..., J)$

In our specific implementation, I fit a model without any covariates. As a result, the model estimates the moments of the distribution purely based on the data itself, without the influence of explanatory variables. The model proceeds by estimating the distribution parameters in this manner:

$\eta_1 = g_1(\mu) = \beta_{0,1}$

$\eta_2 = g_2(\sigma) = \beta_{0,2}$

$\eta_3 = g_3(\nu) = \beta_{0,3}$

$\eta_4 = g_4(\tau) = \beta_{0,4}$

Where:

- η_1 represents the estimated location (central tendency) of the distribution, which is constant across the dataset since no covariates are included.
- η_2 represents the estimated scale (dispersion) of the distribution, which is also treated as a constant value.
- η_3 and η_4 represent the estimated skewness and kurtosis, respectively, which are similarly treated as fixed parameters.

The model thus estimates these distributional moments based solely on the observed data, without adjusting for any external explanatory variables.

Child mortality

child_mort is a continuous numerical variable defined on $\mathbb{R}^+$ with the following distribution:

```
summary(child_mort)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2.60    8.25   19.30   38.27   62.10  208.00
```

```
sd(child_mort)
```

```
## [1] 40.32893
```

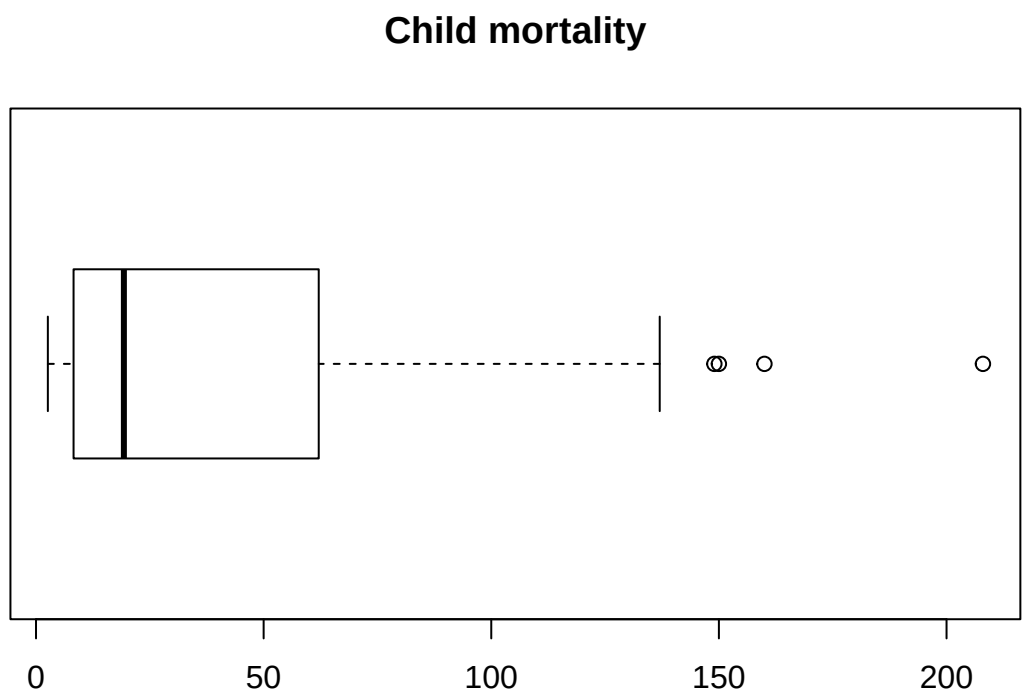
```
kurtosis(child_mort, excess = T) # Normalized -> 0 = std normal kurtosis
```

```
## [1] 1.766882
```

```
skewness(child_mort)
```

```
## [1] 1.450774
```

```
# boxplot
boxplot(data$child_mort, horizontal = T, col = "white", main = "Child mortality")
```




```
# outliers
rownames(data[which(data$child_mort %in% boxplot(data$child_mort, plot = F)$out), ])
```

```
## [1] "Central African Republic" "Chad"
## [3] "Haiti"                    "Sierra Leone"
```

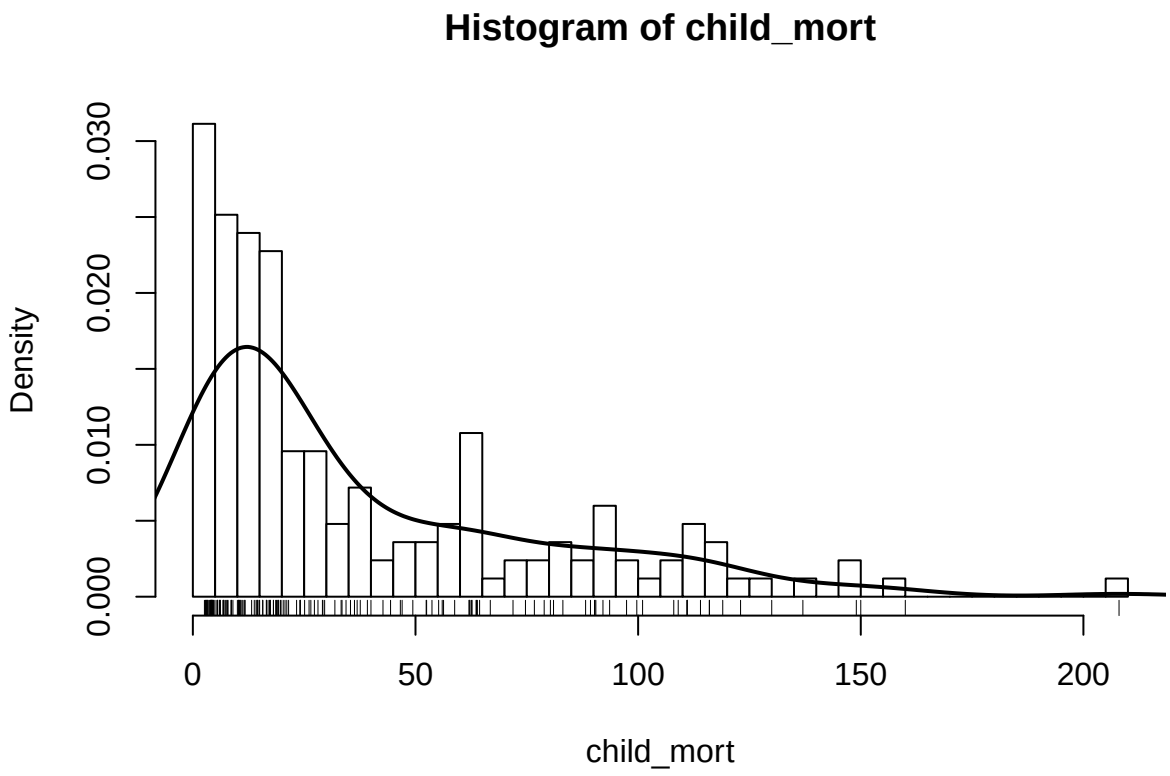
The child_mort variable spans the range from 2.60 to 208, with a relatively high excess kurtosis of 1.77,

indicating heavy tails in the distribution. The central tendency of the data is focused within the interquartile range, roughly between 8.25 and 62.10.

Although the distribution's tails are not extremely long, there are several outliers on the right side, representing countries with significantly higher child mortality rates, such as the Central African Republic, Chad, Haiti, and Sierra Leone.

The skewness of 1.45 suggests that the distribution is not symmetrical, with a tendency toward higher values.

```
hist(child_mort, col = "white", freq = F, breaks = 50)
lines(density(child_mort), col = "black", lwd = 2)
rug(child_mort)
```



In this plot, I created a histogram of the observations, with a rug plot displayed along the x-axis. Additionally, I applied the `density()` function to generate a Kernel Density Estimate (KDE), which provides a smooth, non-parametric estimate of the data distribution. This visualization helps us form a preliminary idea of the data's underlying distribution, guiding the selection of an appropriate parametric model for further analysis.

Parametric Modelling For this data, I use the `fitDist` function in order to find the best distribution for the model. The package employs Maximum Likelihood Estimation (MLE) for parameter estimation.

Based on the AIC and BIC values, the Box-Cox power exponential distribution appears to be the best model for the data, as it has the lowest values for both AIC and BIC.

```
# Modeling with AIC & BIC computations
mod1 <- fitDist(child_mort, data=data, k=2, type = "realplus")
```

```
## Warning in MLE(ll3, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma, :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
## Warning in MLE(ll4, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma, :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
mod2 <- fitDist(child_mort, data=data, k=log(dim(data)[1]), type = "realplus")
```

```
## Warning in MLE(ll3, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma, :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll3, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma, :
## possible convergence problem: optim gave code=1 false convergence (8)
```

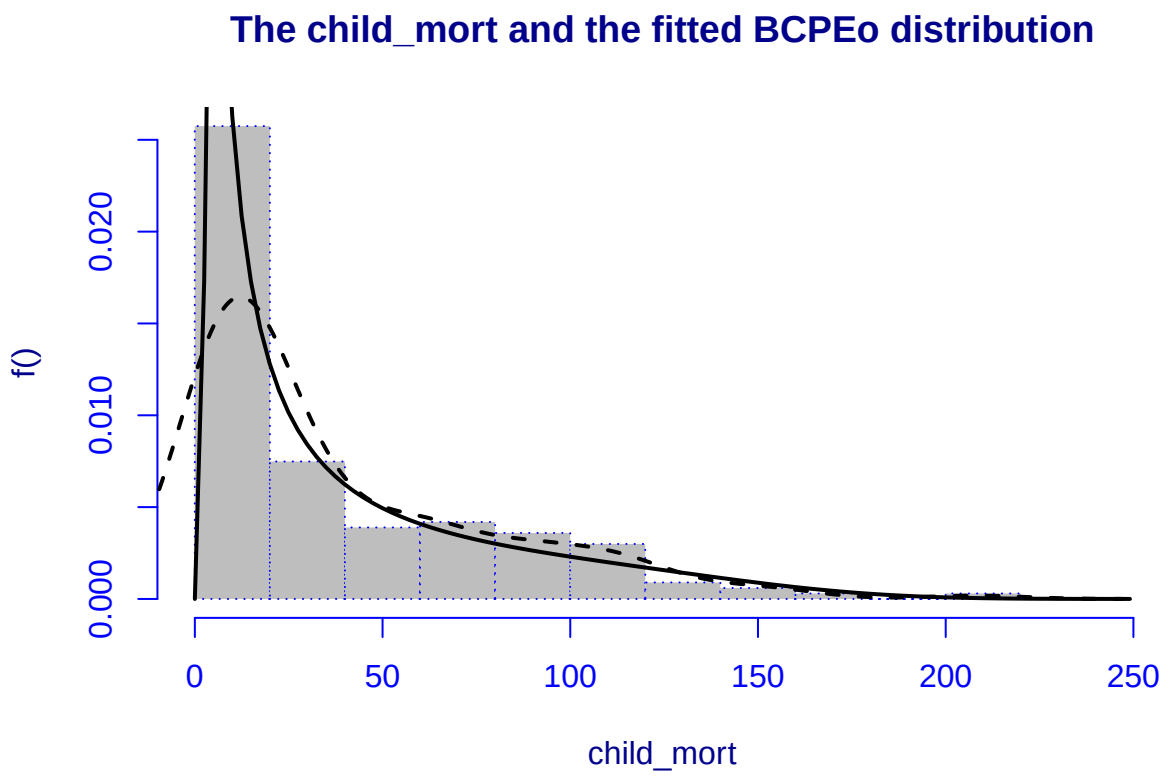
```
mod1$fits[1:6]
```

```
##    BCPEo     BCPE      GIG       IG    LOGNO   LOGNO2
## 1510.835 1523.699 1533.477 1536.893 1544.504 1544.504
```

```
mod2$fits[1:6]
```

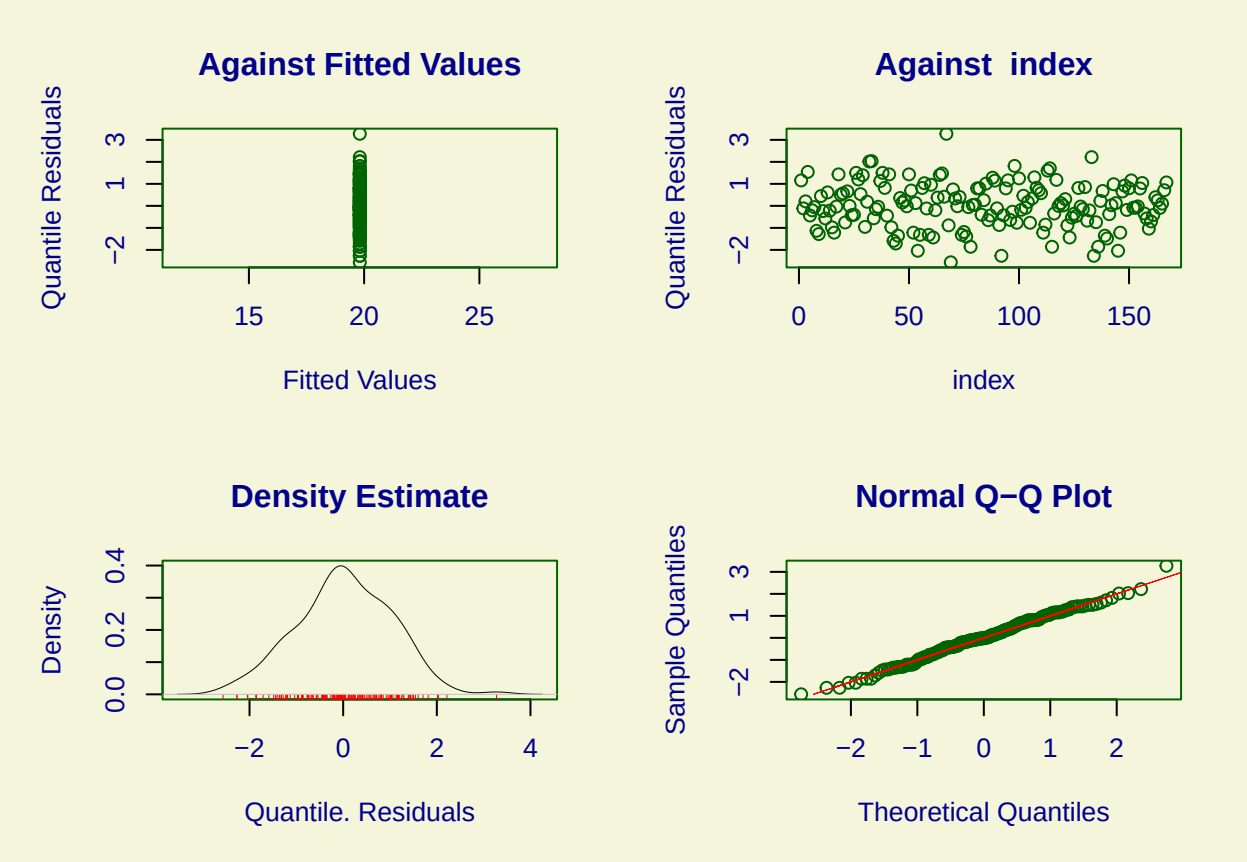
```
##    BCPEo     BCPE      GIG       IG    LOGNO   LOGNO2
## 1523.307 1536.171 1542.831 1543.129 1550.740 1550.740
```

```
m1 <-histDist(child_mort, "BCPEo" , density=TRUE, line.col=c(1,1), line.ty=c(1,2), breaks = 50)
```



The *Box-Cox Power Exponential* (BCPEo) model is a flexible distribution that generalizes the normal distribution by modeling not only the first and second moments (mean and variance), but also the third and fourth moments (skewness and kurtosis). To assess the model's fit, I conducted a quantile residual analysis to verify whether the residuals follow a standard normal distribution, $N(0,1)$.

```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  0.02426947
##                         variance   =  1.022605
##                 coef. of skewness  =  -0.03552723
##                 coef. of kurtosis  =  2.934708
## Filliben correlation coefficient   =  0.9969355
## ****************************************************************
```

In the summary, we observe that the mean is close to zero, the variance is near 1, the distribution is nearly symmetrical, and the kurtosis is approximately 3, which is the kurtosis value for a normal distribution. We have also a near to one value of FCC near to one witch suggest that the residual are distributed as a $N(0,1)$

```
m1_g <- gamlssML(child_mort~1, data=data, family= BCPEo, trace=FALSE)
summary(m1_g)
```

```
## ****************************************************************
## Family:  c("BCPEo", "Box-Cox Power Exponential-orig.")
##
## Call:  gamlssML(formula = child_mort ~ 1, family = BCPEo, data = data,
##     trace = FALSE)
##
## Fitting method: "nlminb"
##
##
## Coefficient(s):
##             Estimate  Std. Error  t value   Pr(>|t|)
```

```
## eta.mu     2.9862961   0.1198079 24.92571 < 2.22e-16 ***
## eta.sigma  0.1372435   0.0285409  4.80867 1.5194e-06 ***
## eta.nu    -0.0363728   0.0600052 -0.60616    0.54441
## eta.tau    2.4647901   0.3145267  7.83650 4.6629e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   163
## Global Deviance:     1502.83
##             AIC:     1510.83
##             SBC:     1523.31
```

The **Box-Cox Power Exponential (BCPE)** distribution is a flexible probabilistic model defined by four parameters:

- μ: location (central tendency)
- σ: scale (dispersion)
- ν: skewness (asymmetry)
- τ: kurtosis (tail behavior)

This model is particularly useful for capturing complex data distributions that deviate from normality by adjusting for asymmetry and heavy tails.

Model Summary The model was fitted using **Maximum Likelihood Estimation (MLE)**. The parameter estimates are as follows:

- $\eta_\mu = 2.99$ ($p < 2.22 \times 10^{-16}$): Indicates the central value of the distribution.
- $\eta_\sigma = 0.14$ ($p = 1.52 \times 10^{-6}$): Reflects a relatively small dispersion.
- $\eta_\nu = -0.04$ ($p = 0.544$): Suggests no significant skewness in the data. According with the p-value this estimate is not significant
- $\eta_\tau = 2.46$ ($p = 4.66 \times 10^{-15}$)

The model achieved:
- **Global Deviance**: 1502.83,
- **AIC**: 1510.83,
- **BIC**: 1523.31.

These metrics suggest a good overall fit to the data.

Pearson's chi-square test Pearson's chi-square goodness of fit test involves comparing the observed frequencies in categorized (binned) data with the expected frequencies from a theoretical model that we assume fits the data well.

The hypothesis are:

$$H_0 : (p_1 = \pi_1) \ \cap \ ... \cap \ (p_s = \pi_s) \qquad H_1 : (p_1 \neq \pi_1) \ \cap \ ... \cap \ (p_s \neq \pi_s)$$

The test statistics is :

$$\mathcal{X}^2 = \sum_{j=1}^{s} \frac{(n_j - \hat{n_j})^2}{\hat{n_j}} \overset{n\to\infty}{\sim} \chi^2_{(s-1)}$$

```
##############################
## Perason's chi square test ##
##############################

quantiles <- quantile(child_mort, probs = c(0, .25, .5, .75, 1))
quantiles <- as.numeric(quantiles)

br0 <- pBCPEo(quantiles[1], mu = m1$mu, sigma = m1$sigma,
              nu = m1$nu, tau = m1$tau)

# I have to exclude the lowest quantile [0, 2.6]

br1 <- pBCPEo(quantiles[2], mu = m1$mu, sigma = m1$sigma,
              nu = m1$nu, tau = m1$tau) - br0

br2 <- pBCPEo(quantiles[3], mu = m1$mu, sigma = m1$sigma,
              nu = m1$nu, tau = m1$tau) - (br1 + br0)

br3 <- pBCPEo(quantiles[4], mu = m1$mu, sigma = m1$sigma,
              nu = m1$nu, tau = m1$tau) - (br1 + br2 + br0)

br4 <- 1 - (br1 + br2 + br3 + br0)

brks <- c(br1, br2, br3, br4)

theor_counts <- c()

for(i in 1:4){
  theor_counts[i] <- brks[i] * nrow(data)

}

sum(theor_counts) # not exactly = to 167 because I removed the first cut
```

```
## [1] 166.1371
```

```
empiric_count <- cut(child_mort, breaks = quantiles, include.lowest = T)
emp <- as.numeric(table(empiric_count))

chisqstat <- sum((emp - theor_counts)^2 / theor_counts)

pvalue <- 1 - pchisq(chisqstat, df = length(theor_counts) - 1)

pvalue
```

```
## [1] 0.3572299
```

For a value of $p = 0.35$, we cannot reject H_0 because there is not enough evidence to do so.

Exports

`exports` is a continuous numerical variable defined on $\mathbb{R}^+$ with the following distribution:

```
summary(exports) # defined in R+
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##   0.109  23.800  35.000  41.109  51.350 200.000
```

```
sd(exports)
```

```
## [1] 27.41201
```

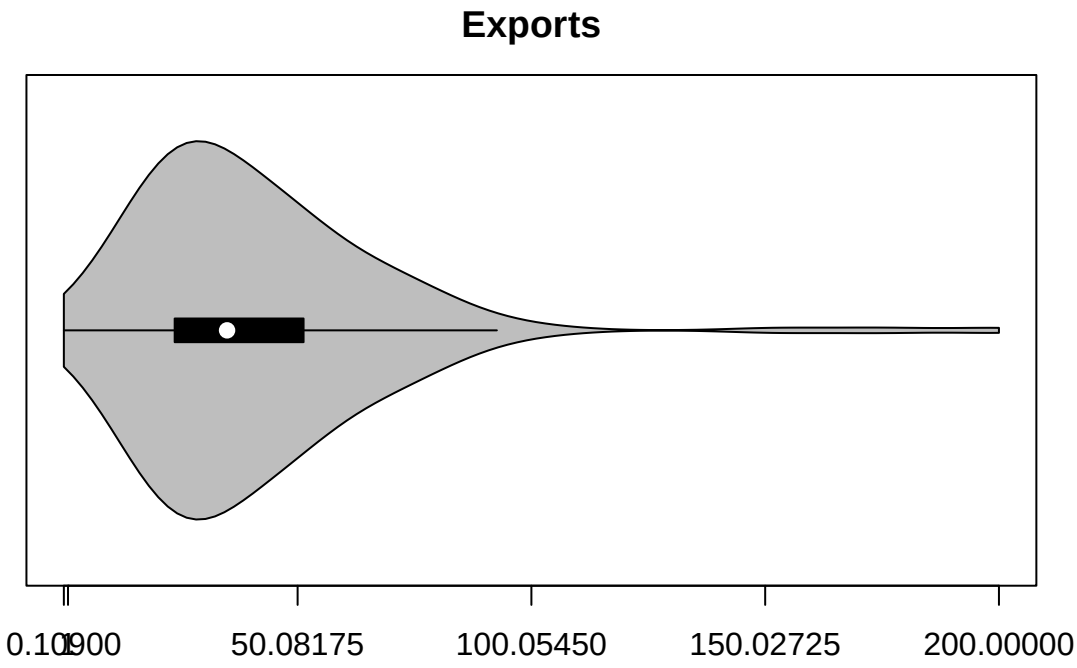
```
kurtosis(exports, excess = T)
```

```
## [1] 10.13867
```

```
skewness(exports)
```

```
## [1] 2.445824
```

```
vioplot(exports, horizontal = T, col = "gray", main = "Exports", xaxt = "n")
axis(1, at = seq(min(exports), max(exports), length.out = 5))
```



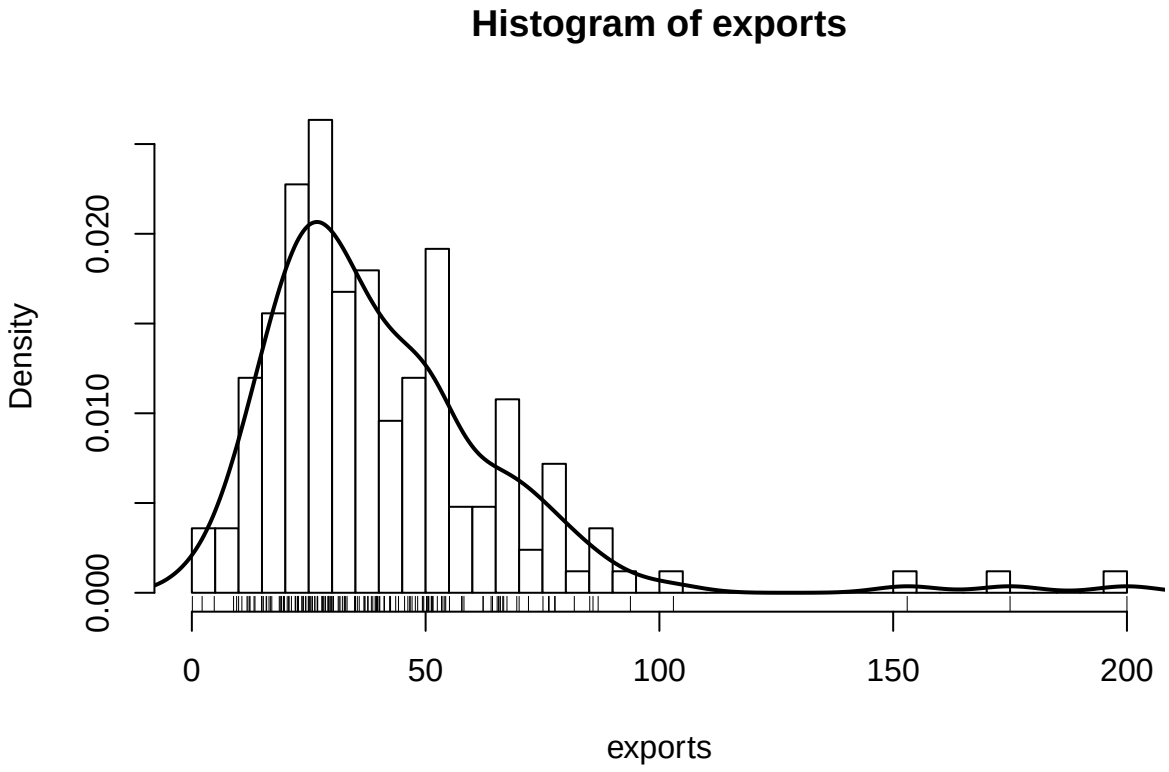

```
# wich countries are outliers?
rownames(data[which(data$exports %in% boxplot(data$exports, plot = F)$out), ])
```

```
## [1] "Ireland"    "Luxembourg" "Malta"      "Seychelles" "Singapore"
```

The `exports` variable is distributed over the 0-200 range, with a relatively high excess kurtosis of 10.13. The majority of the data is concentrated within the interquartile range, approximately between 23 and 51. The distribution's tails are not particularly long; however, there are a few outliers on the right-hand side, which correspond to wealthier countries, reflecting their significantly higher values compared to the rest of the data. The skewness is 2.44, which indicates that the data are not symmetrical

```
hist(exports, col = "white", freq = F, breaks = 60)
lines(density(exports), col = "black", lwd = 2)
```

```
rug(exports)
```




```
mod1 <- fitDist(exports, data=data, k=2, type = "realplus")
```

Parametric Modelling

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
mod2 <- fitDist(exports, data=data, k=log(dim(data)[1]), type = "realplus")
```

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

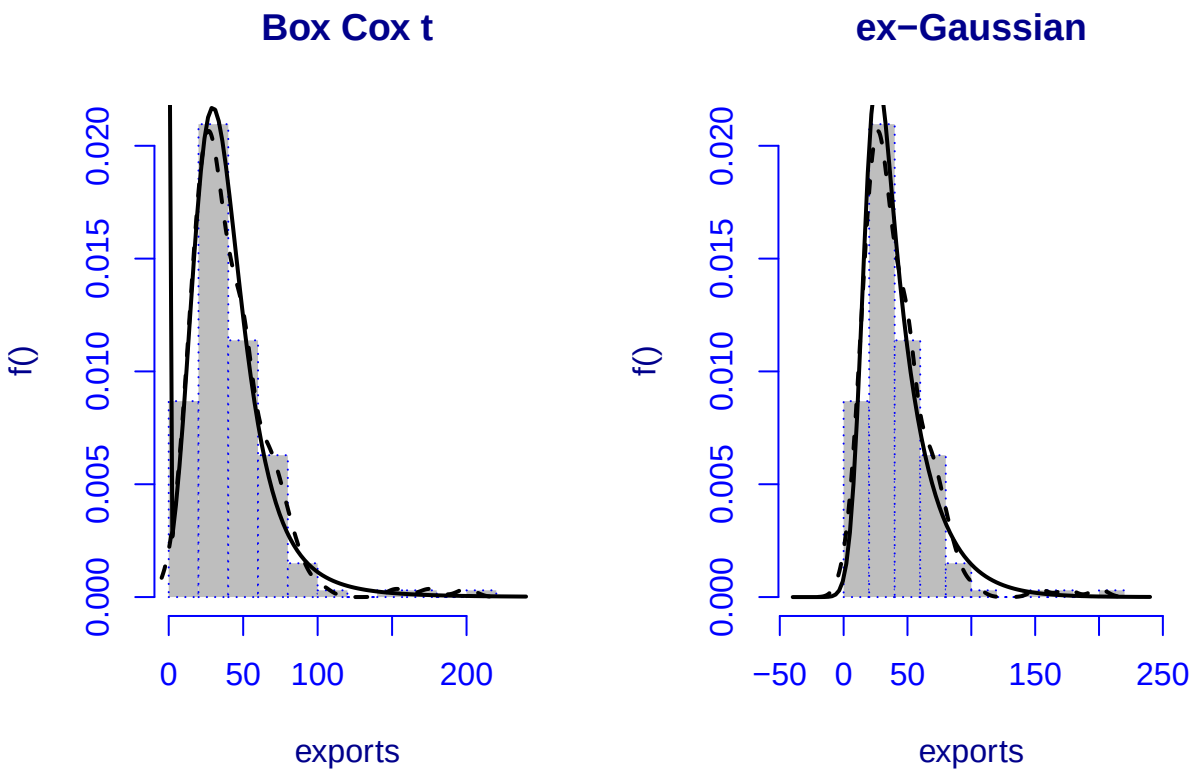
```
mod1$fits[1:6]
```

```
##     BCTo      BCT   exGAUS     BCPE    BCPEo      GB2
## 1503.001 1503.001 1504.471 1507.339 1507.339 1509.643
```

```
mod2$fits[1:6]
```

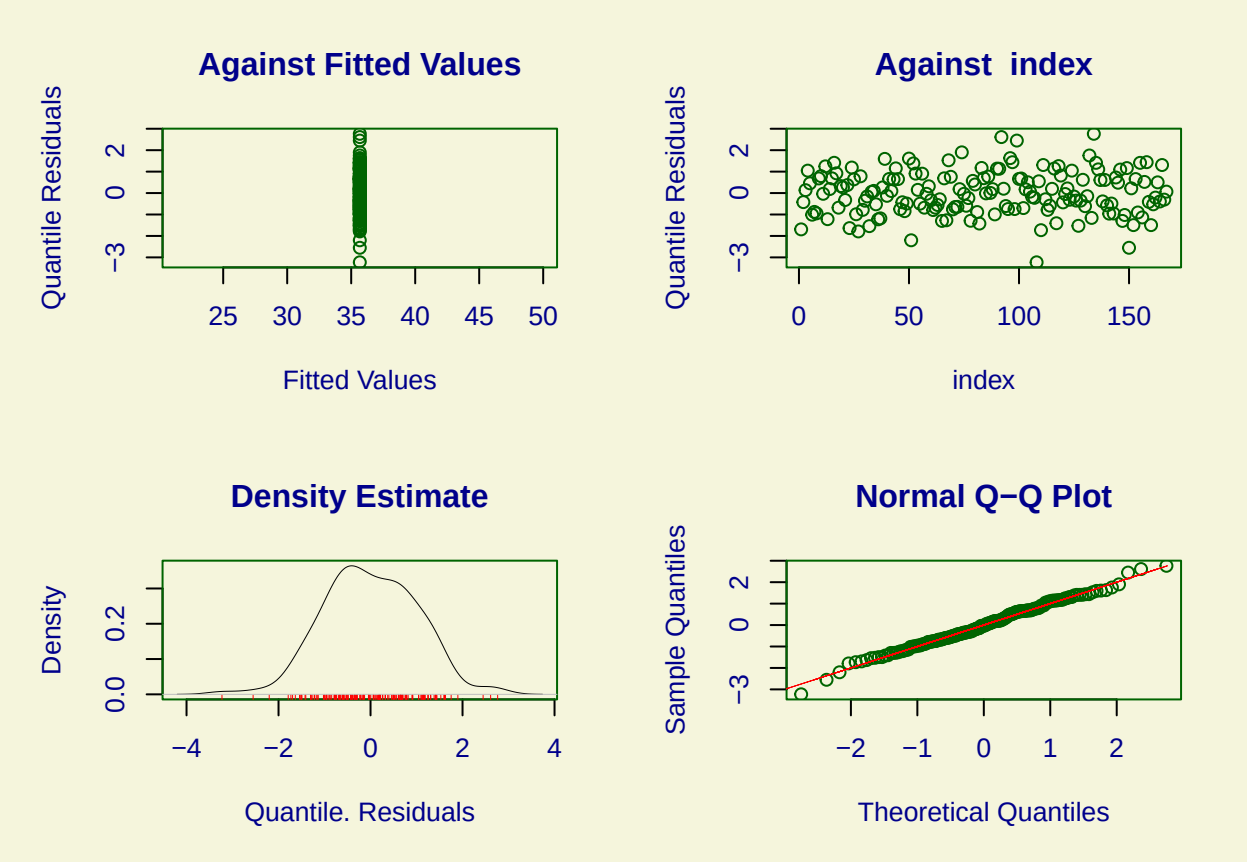
```
##   exGAUS     BCTo      BCT     BCPE    BCPEo     BCCG
## 1513.825 1515.473 1515.473 1519.811 1519.811 1521.025
```

```
par(mfrow = c(1, 2))
m1 <- histDist(exports, "BCTo" , density=TRUE, line.col=c(1,1), line.ty=c(1,2),
               breaks = 50,main = "Box Cox t")
m2 <- histDist(exports, "exGAUS" , density=TRUE, line.col=c(1,1), line.ty=c(1,2),
               breaks = 50, main = "ex-Gaussian")
```



Despite the good fit of the *Exponential Gaussian distribution* based on the scoring, it is defined on the entire real line, $\mathbb{R}$, and is theoretically inappropriate for this data, which consists solely of positive values. Therefore, for the analysis, I will consider only the *Box-Cox t distribution*.

```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  0.002816074
##                         variance   =  1.008446
##                 coef. of skewness  =  -0.01615781
##                 coef. of kurtosis  =  3.074646
## Filliben correlation coefficient  =  0.9964711
## ****************************************************************
```

The residuals between the data and the model are empirically distributed as $N(0, 1)$, which indicates a good fit throughout the model and the data.

Gamma Mixture model For this model, I attempted to fit a *Gamma Mixture Model* because the kernel density estimate (KDE) suggests that the data could be a combination of multiple Gamma distributions.

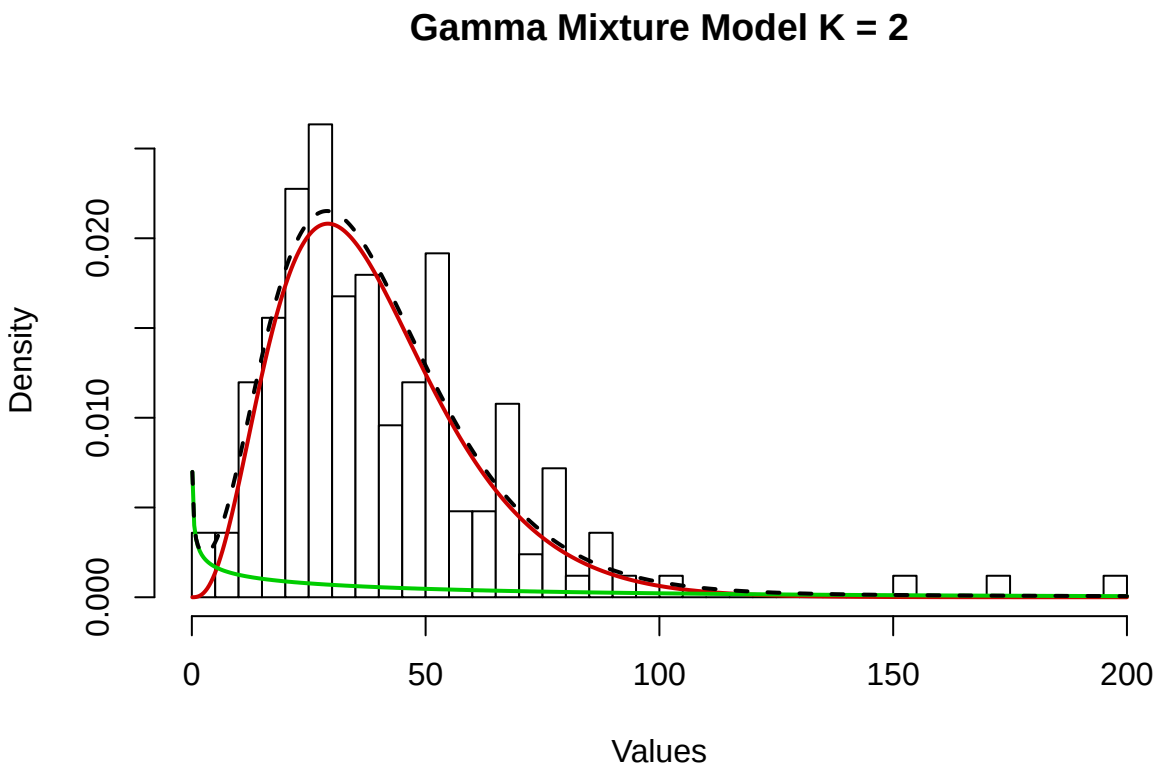
```
fit_mix <- mixfit(exports,
             family = "gamma",
             ncomp = 2,    # from the KDE 2 seems to be enough
             tol = 1e-06)

fit_mix
```

```
## Gamma mixture model with 2 components
##            comp1      comp2
## pi     0.9159365  0.0840635
## mu    39.1038869 62.9559721
## sd    19.7782545 78.5772522
```

```
## shape  3.9089844  0.6419189
## rate   0.0999641  0.0101963
##
## EM iterations: 54 AIC: 1501.17 BIC: 1516.76 log-likelihood: -745.59
```

Based on this output, the model appears to perform better according to the AIC. This suggests that, despite its higher complexity, the model provides a better fit for these data.

```
############
##  Plot  ##
############

colors <- c("red3", "green3")

weights <- fit_mix$pi
mu <- fit_mix$mu
shape <- fit_mix$alpha
rate <- fit_mix$lambda

hist(exports, breaks = 30, probability = T, col = "white", border = "black",
     main = "Gamma Mixture Model K = 2",
     xlab = "Values", ylab = "Density")

x_vals <- seq(min(exports), max(exports), length.out = 500)

for (i in 1:length(weights)) {
  lines(x_vals, weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i]), col = colors[i], lwd = 2
}

overall_density <- rowSums(sapply(1:length(weights), function(i) {
  weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i])
}))

lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)
```



Additionally, the Weibull distribution could be a potential candidate for a mixture model, based on the available distribution families in the package. However, due to computational constraints, it was not applied.

```
m1_g <- gamlssML(exports~1, data=data, family= BCTo, trace=FALSE)
summary(m1_g)
```

Model Summary (BCTo)

```
## ******************************************************************
## Family:  c("BCTo", "Box-Cox-t-orig.")
##
## Call:
## gamlssML(formula = exports ~ 1, family = BCTo, data = data, trace = FALSE)
##
## Fitting method: "nlminb"
##
##
## Coefficient(s):
##             Estimate  Std. Error  t value   Pr(>|t|)
## eta.mu     3.5743781   0.0471061 75.87932 < 2.22e-16 ***
## eta.sigma -0.6413611   0.0858206 -7.47327 7.8160e-14 ***
## eta.nu     0.2885993   0.1279083  2.25630   0.024052 *
## eta.tau    1.7682690   0.4076145  4.33809 1.4373e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   163
## Global Deviance:     1495
##             AIC:     1503
##             SBC:     1515.47
```

The **Box-Cox t (BCTo)** distribution is a flexible probabilistic model defined by four parameters:

- μ: location (central tendency)
- σ: scale (dispersion)
- ν: skewness (asymmetry)
- τ: kurtosis (tail behavior)

This model is particularly useful for capturing complex data distributions that deviate from normality by adjusting for asymmetry and heavy tails.

The model was fitted using **Maximum Likelihood Estimation (MLE)**. The parameter estimates are as follows:

- $\mu = 3.57$ ($p < 2.22 \times 10^{-16}$): Indicates the central value of the distribution.
- $\sigma = -0.64$ ($p = 7.82 \times 10^{-14}$): Reflects a relatively small dispersion.
- $\nu = 0.29$ ($p = 0.024$): Suggests a slight right skew in the data. This parameter is the less signficant.
- $\tau = 1.77$ ($p = 1.44 \times 10^{-5}$): Implies heavy tails compared to the normal distribution.

The model achieved:
- **Global Deviance**: 1495,
- **AIC**: 1503,
- **BIC**: 1515.47.

These metrics suggest a good overall fit to the data.

Health

`health` is a continuous numerical variable defined on $\mathbb{R}^+$ with the following distribution:

```
summary(health)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.810   4.920   6.320   6.816   8.600  17.900
```

```
sd(health)
```

```
## [1] 2.746837
```

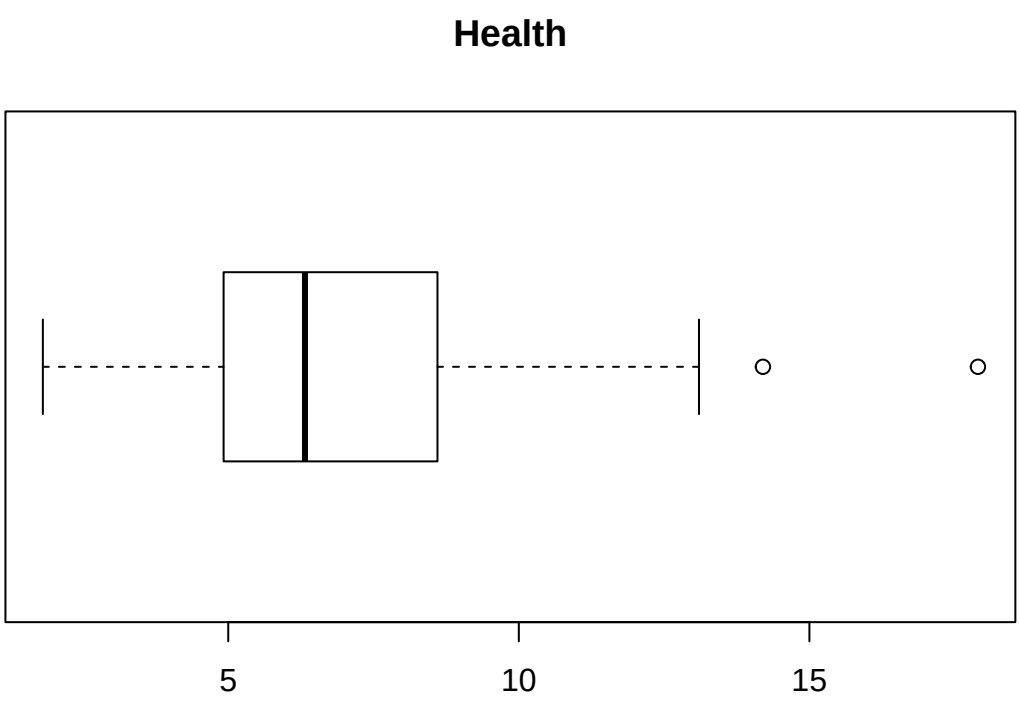
```
kurtosis(health, excess = T)
```

```
## [1] 0.6941956
```

```
skewness(health)
```

```
## [1] 0.7057461
```

```
boxplot(health, horizontal = T, col = "white", main = "Health")
```

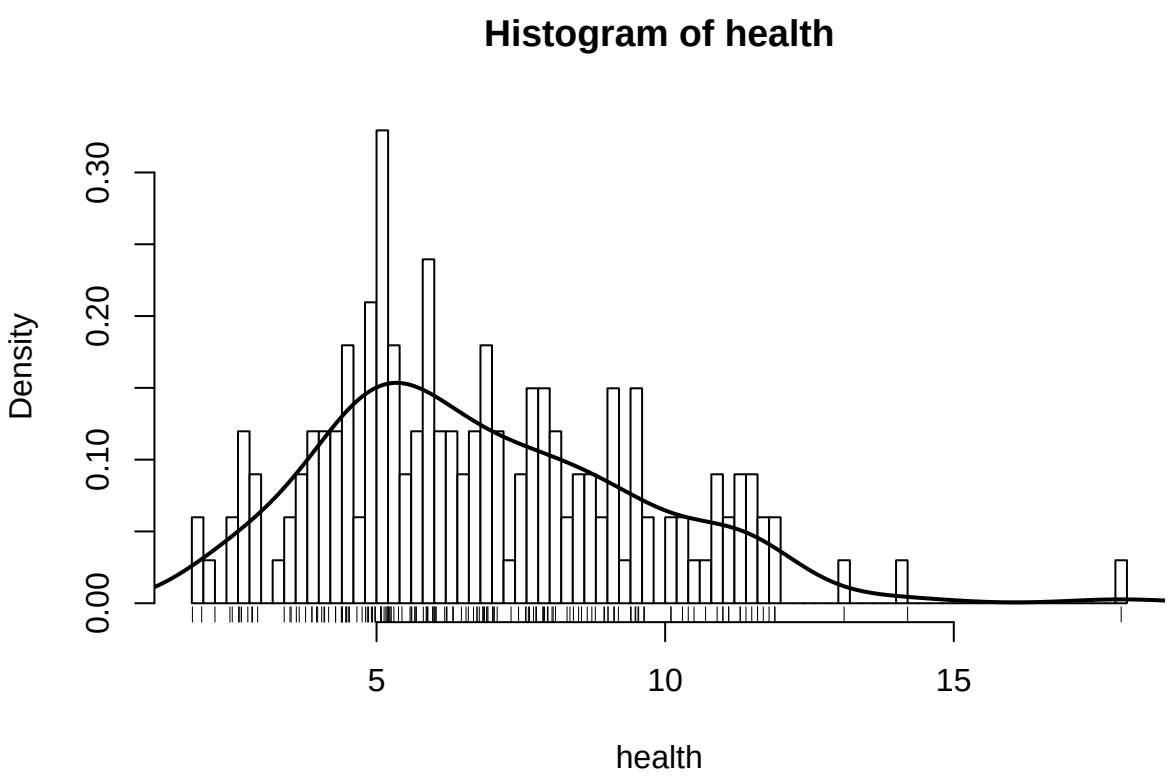



```
# outliers
rownames(data[which(health %in% boxplot(health, plot = F)$out), ])
```

```
## [1] "Micronesia, Fed. Sts." "United States"
```

The `health` variable, representing the percentage of GDP spent per person, ranges from 1.81 to 17.90. The distribution has a relatively low kurtosis and a skewness of 0.69, indicating a slight asymmetry towards the right. Most of the data falls within the interquartile range, between 4.92 and 8.60. The distribution does not exhibit long tails, but there are a few outliers on the right, notably corresponding to countries like Micronesia and the United States, with significantly higher values compared to the rest of the data.

```
hist(health, col = "white", freq = F, breaks = 60)
lines(density(health), col = "black", lwd = 2)
rug(health)
```




```
mod1 <- fitDist(health, data=data, k=2, type = "realplus")
```

Parametric Modelling

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
mod2 <- fitDist(health, data=data, k=log(dim(data)[1]), type = "realplus")
```

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
mod1$fits[1:6]
```

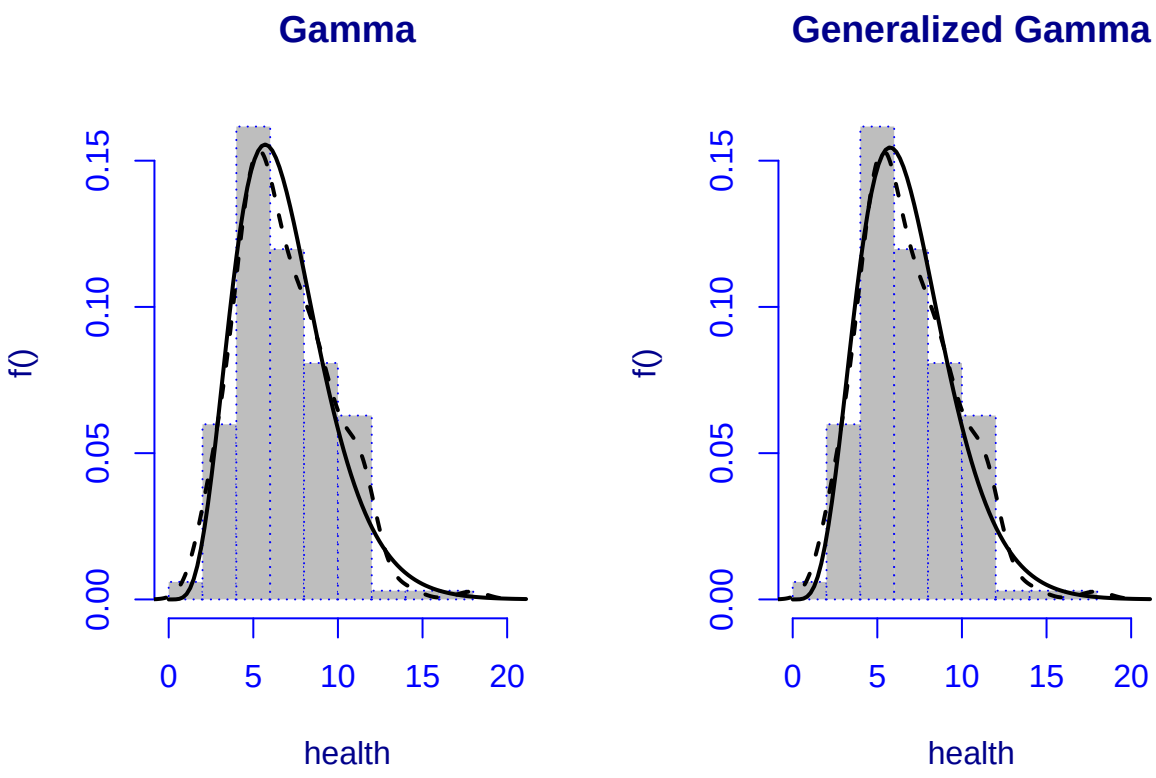
```
##       GA       GG      GIG    BCCGo     BCCG     BCPE
## 797.9751 799.9259 799.9662 800.0988 800.0988 801.3592
```

```
mod2$fits[1:6]
```

```
##       GA   LOGNO2    LOGNO       GG      GIG    BCCGo
```

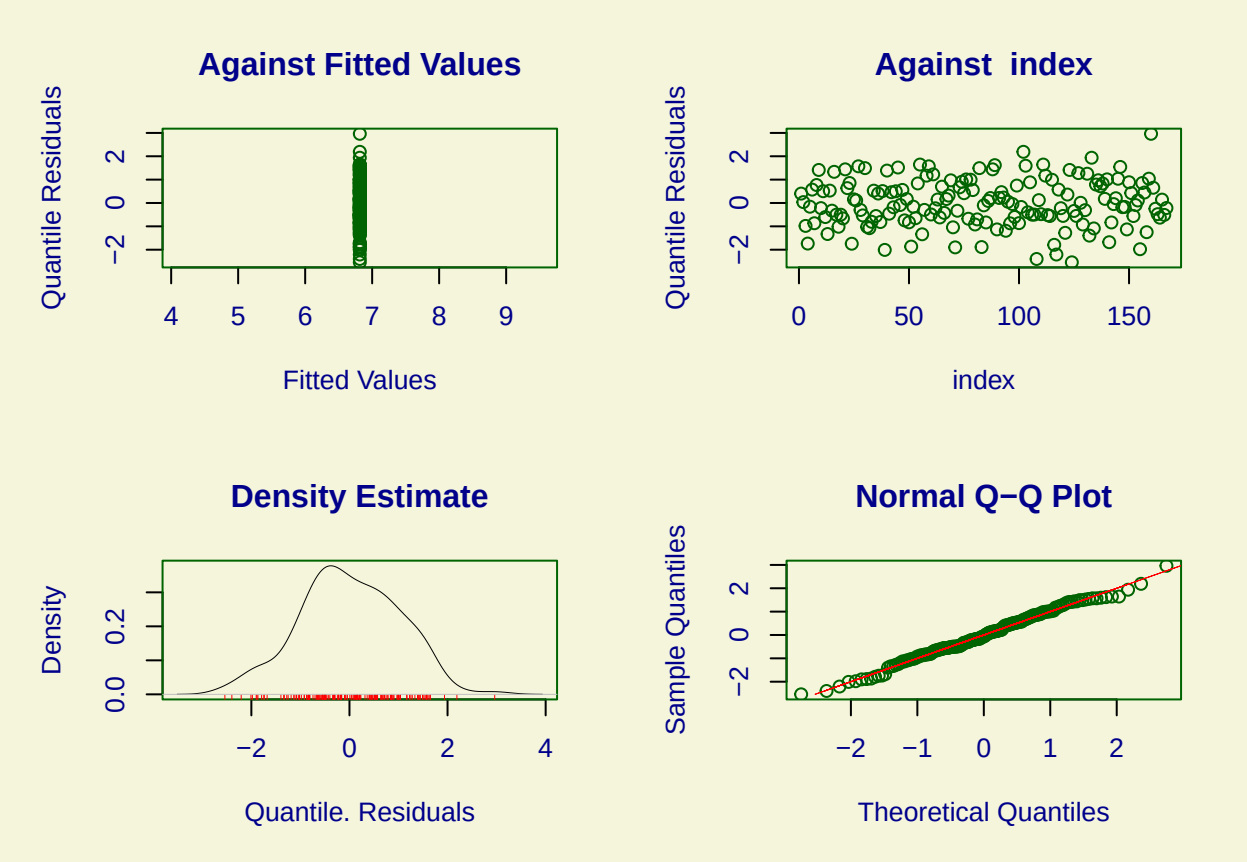
```
## 804.2111 809.2740 809.2740 809.2799 809.3202 809.4527
```

```
par(mfrow = c(1, 2))
m1 <- histDist(health, "GA" , density=TRUE, line.col=c(1,1), line.ty=c(1,2), breaks = 50, main = "Gamma
m2 <- histDist(health, "GG" , density=TRUE, line.col=c(1,1), line.ty=c(1,2), breaks = 50, main = "Genera
```



The health variable fits the *Gamma distribution* less well compared to the other models, as it yields lower values for both AIC (797.98) and BIC (804.21) compared to the other distributions tested.

```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  0.00012377
##                         variance   =  1.00605
##                 coef. of skewness  =  -0.03694263
##                 coef. of kurtosis  =  2.754093
## Filliben correlation coefficient   =  0.9964949
## ****************************************************************
```

In the summary of the quantile residuals, we observe that the mean is very close to zero (0.0001), the variance is near 1 (1.006), and the kurtosis is approximately 2.75, which is slightly lower than the kurtosis value of 3 for a normal distribution. The skewness is close to zero (−0.036), indicating a near-symmetrical distribution. Additionally, the Filliben correlation coefficient is very close to one (0.9964), suggesting that the residuals are distributed approximately as $N(0, 1)$.

```
m1_g <- gamlssML(health~1, data=data, family= GA, trace=FALSE)
summary(m1_g)
```

Model Summary

```
## ****************************************************************
## Family:  c("GA", "Gamma")
##
## Call:  gamlssML(formula = health ~ 1, family = GA, data = data, trace = FALSE)
##
##
```

```
## Fitting method: "nlminb"
##
##
## Coefficient(s):
##             Estimate  Std. Error  t value   Pr(>|t|)
## eta.mu     1.9192271   0.0313339  61.2508 < 2.22e-16 ***
## eta.sigma -0.9040577   0.0532882 -16.9655 < 2.22e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Degrees of Freedom for the fit: 2 Residual Deg. of Freedom   165
## Global Deviance:     793.975
##             AIC:     797.975
##             SBC:     804.211
```

The **Gamma (GA)** distribution is a flexible probabilistic model characterized by two key parameters:

- μ: location (central tendency)
- σ: scale (dispersion)

This model is particularly suitable for modeling skewed data with positive values, such as the `health` variable, representing the percentage of GDP spent per person. It can effectively capture the asymmetry and dispersion of data that does not follow a normal distribution.

The model was fitted using **Maximum Likelihood Estimation (MLE)**. The parameter estimates are as follows:

- $\eta_\mu = 1.92$ ($p < 2.22 \times 10^{-16}$): Indicates the central value of the distribution.
- $\eta_\sigma = -0.90$ ($p < 2.22 \times 10^{-16}$): Reflects a moderate dispersion of the data.

The model achieved:

- **Global Deviance**: 793.98
- **AIC**: 797.98
- **SBC**: 804.21

These metrics suggest that the Gamma distribution provides a good fit for the `health` data, capturing its skewness and dispersion effectively.

Likelihood ratio test I compute the model `m2` to perform a likelihood ratio test between two models:

- $X \sim Gamma(\alpha, \lambda)$
- $X \sim Generalized\ Gamma(\alpha, d, p)$

In the Generalized Gamma model, we introduce an additional shape parameter p. The goal is to assess whether implementing this more flexible model improves the fit to the observed data.

The underlying idea of the likelihood ratio test is to compare these two models by testing the following hypotheses:

Given a set of observations $X = \{x_1, x_2, ..., x_n\}$ sampled from $X \sim f(x; \theta)$, we whant to chose between those hypothesis:

$$H_0 : \theta \in \Theta_0 \qquad H_1 : \theta \in \Theta$$

where $\Theta_0 \subset \Theta$, meaning that the simpler model (Gamma) is nested within the more flexible model (Generalized Gamma).

To compute the test statistic, we evaluate the likelihood ratio:

$$LR = \frac{\max_{\theta \in \Theta_0} L(\theta, X)}{\max_{\theta \in \Theta} L(\theta, X)}$$

The test statistic is then defined as:

$$D = -2log(LR) \overset{n \to \infty}{\sim} \chi^2_m$$

Here, m is the difference in the number of parameters between the two models ($m = |\Theta| - |\Theta_0|$) . In our case, $m = 1$, because the Generalized Gamma model adds one additional parameter (p) compared to the Gamma model.

Thanks to this asymptotic property, we can compute the p-value associated with D. If the p-value is below a chosen significance level (e.g., 0.05), we reject the null hypothesis H_0, concluding that the additional parameter in the Generalized Gamma model significantly improves the fit to the data.

We have no a direct value for log-likelihood in the list `m1` and `m2`.

But we can find those value from the AIC:

$log - likelihood = -\frac{AIC - 2k}{2}$

```
lglik1 <- -(m1$aic - 2*m1$df.fit)/2
lglik2 <- -(m2$aic - 2*m2$df.fit)/2

lik1 <- exp(lglik1)
lik2 <- exp(lglik2)
```

```
####################
##  LR statistic  ##
####################

LR <- lik1 / lik2

D <- -2 * log(LR)

pvalue <- 1 - pchisq(D, df = 1)
pvalue
```

```
## [1] 0.8245396
```

The p-value of **0.8245** is quite high, indicating that there is **no significant evidence** to reject the null hypothesis. Therefore, the simpler model (Gamma with 2 parameters) is a sufficient fit for the data, and the more complex model (Generalized Gamma with 3 parameters) does not provide a significantly better fit.

Imports

`imports` is a continuous numerical variable defined on $\mathbb{R}^+$ with the following distribution:

```
summary(imports)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##   0.0659  30.2000  43.3000  46.8902  58.7500 174.0000
```

```
sd(imports)
```

```
## [1] 24.20959
```

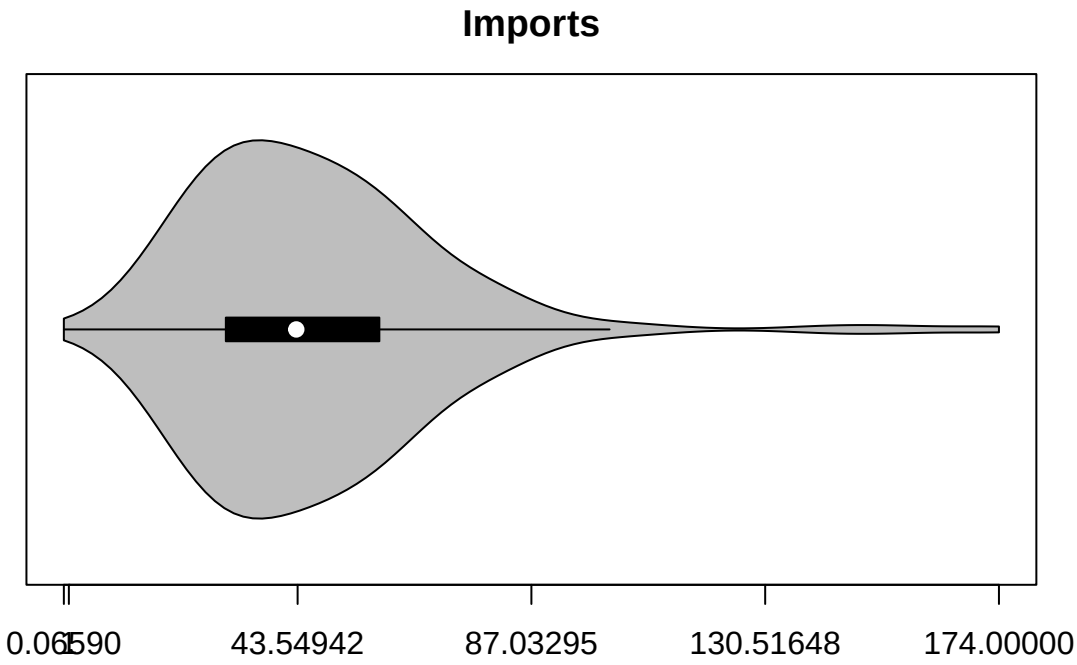
```r
kurtosis(imports, excess = T)
```

```
## [1] 6.755854
```

```r
skewness(imports)
```

```
## [1] 1.905276
```

```r
vioplot(imports, horizontal = T, col = "gray", main = "Imports", xaxt = "n")
axis(1, at = seq(min(imports), max(imports), length.out = 5))
```

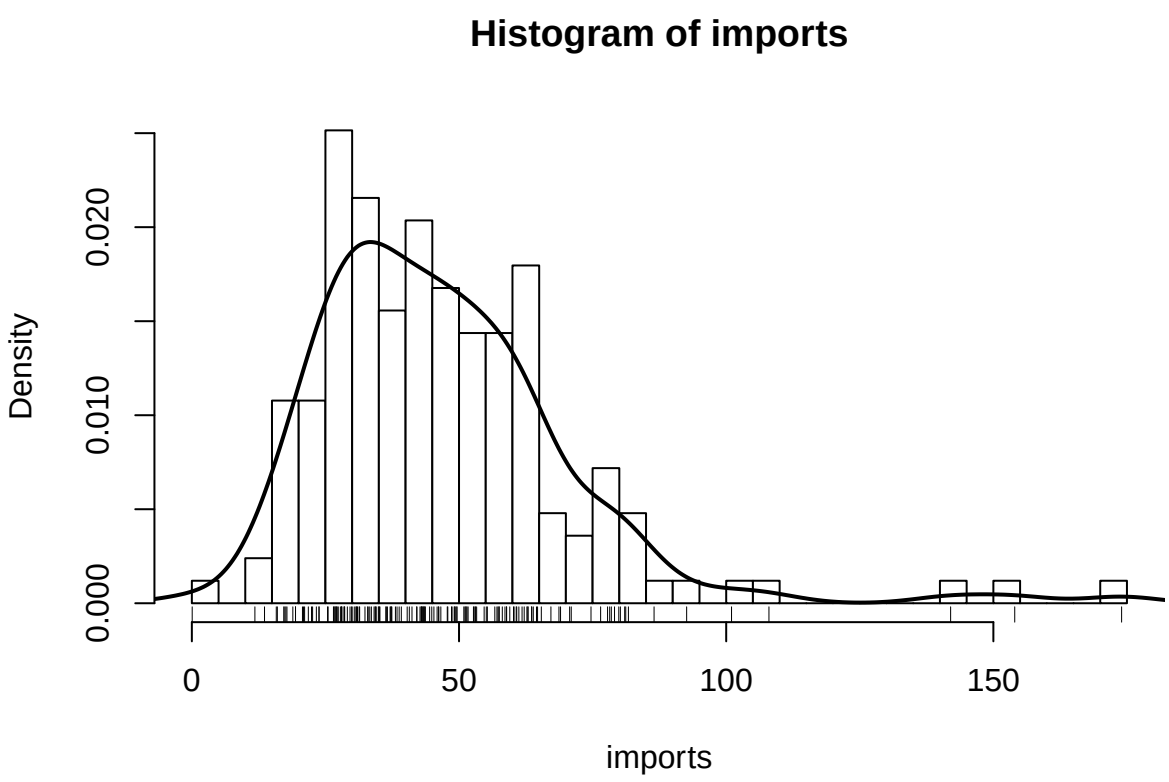



```r
# outliers
rownames(data[which(imports %in% boxplot(imports, plot = F)$out), ])
```

```
## [1] "Luxembourg" "Malta"      "Seychelles" "Singapore"
```

The `import` variable spans a range from 0.0659 to 174. The distribution exhibits a moderately high excess kurtosis of 6.76, indicating a sharper peak and heavier tails compared to a normal distribution. Most of the data points are concentrated within the interquartile range, between approximately 30.2 and 58.75, suggesting that the majority of countries have relatively similar values for `import`. The distribution is positively skewed, with a skewness of 1.91 reflecting a lack of symmetry and a longer tail on the right-hand side. A few outliers, including Luxembourg, Malta, Seychelles, and Singapore, represent wealthier nations with significantly higher `import` values than the rest of the dataset.

```r
hist(imports, col = "white", freq = F, breaks = 60)
lines(density(imports), col = "black", lwd = 2)
rug(imports)
```




```
mod1 <- fitDist(imports, data=data, k=2, type = "realplus")
```

Parametric Modeling

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
mod2 <- fitDist(imports, data=data, k=log(dim(data)[1]), type = "realplus")
```

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

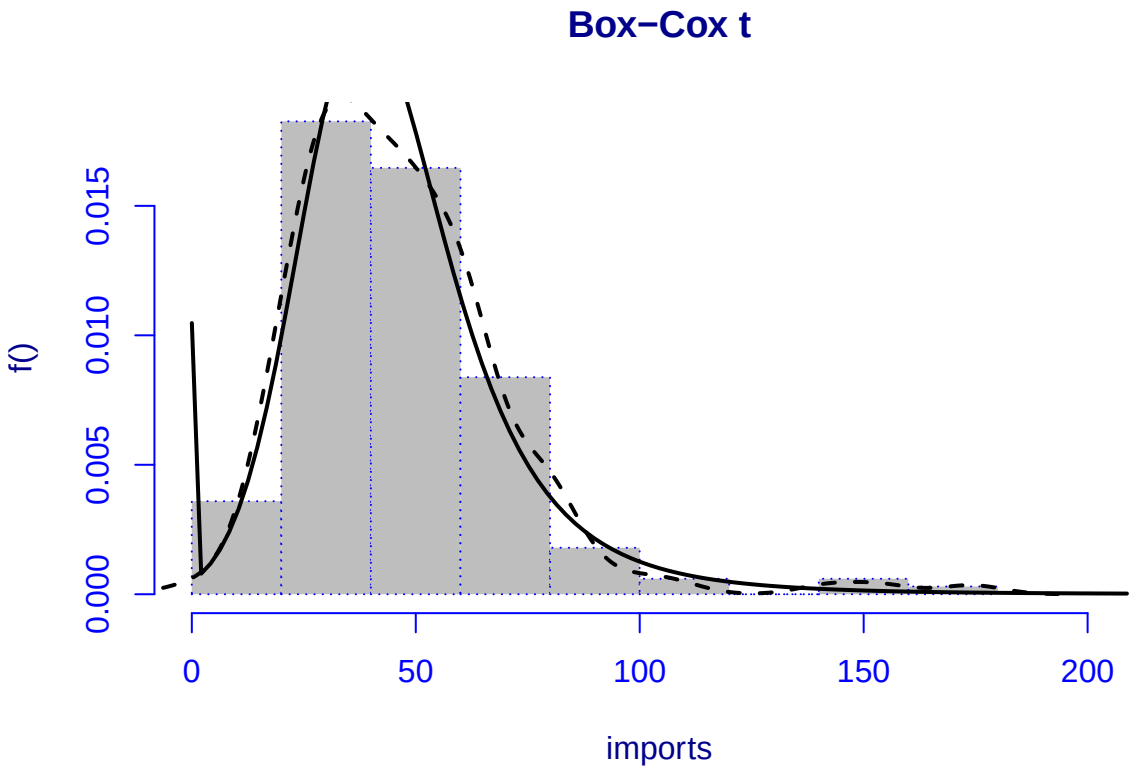
```
mod1$fits[1:6]
```

```
##     BCTo      BCT   exGAUS     BCPE    BCPEo    BCCGo
## 1489.329 1489.329 1490.446 1495.157 1495.157 1500.288
```

```
mod2$fits[1:6]
```

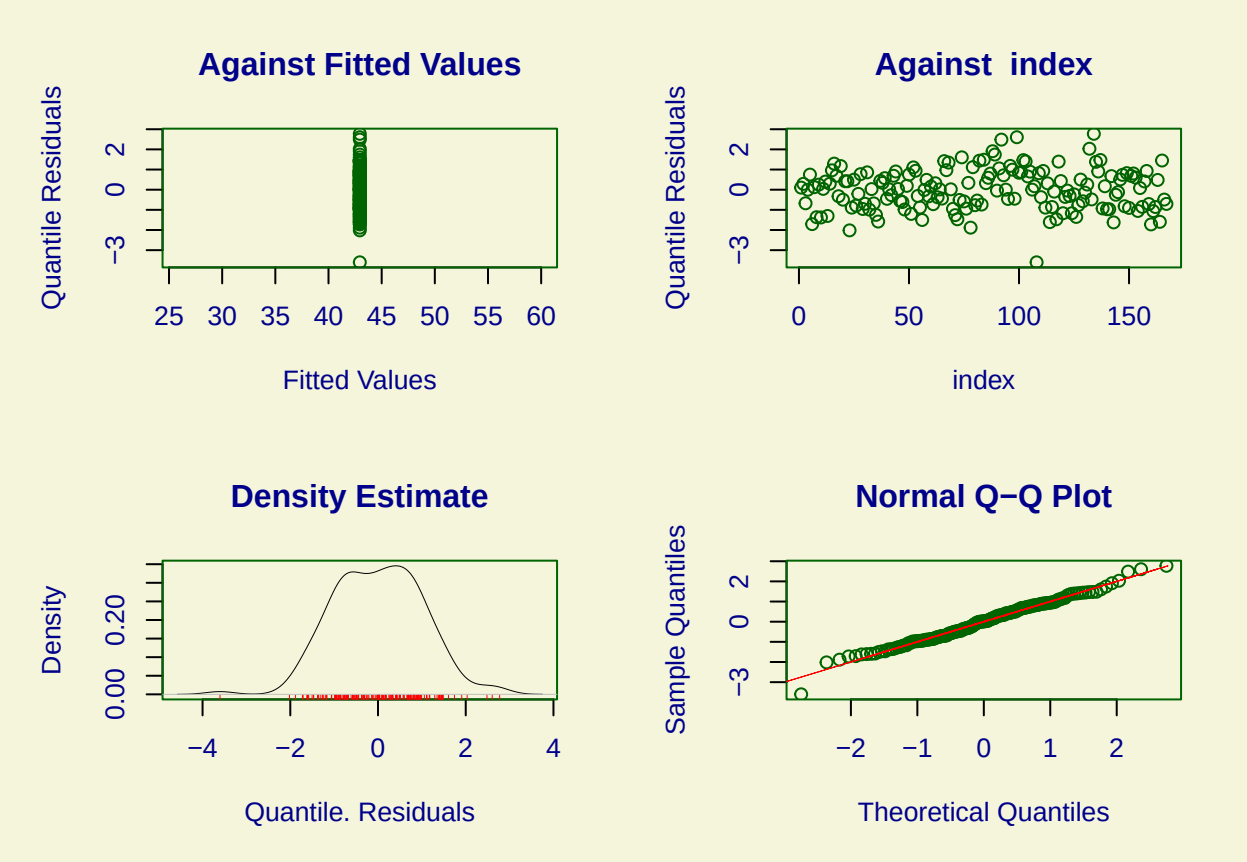
```
##   exGAUS     BCTo      BCT     BCPE    BCPEo    BCCGo
## 1499.800 1501.801 1501.801 1507.629 1507.629 1509.642
```

```
m1 <- histDist(imports, "BCTo" , density=TRUE, line.col=c(1,1), line.ty=c(1,2),
               breaks = 50, main = "Box-Cox t")
```



As before we exclude the *Exponential Gaussian* model in favour of *Box-Cox t* distribution.

```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  0.003848311
##                         variance   =  1.012017
##                 coef. of skewness  =  -0.02824001
##                 coef. of kurtosis  =  3.237237
## Filliben correlation coefficient   =  0.9941813
## ****************************************************************
```

The summary of the quantile residuals shows that the mean is very close to zero (0.0038), the variance is nearly 1 (1.012), and the kurtosis is approximately 3.24, which is slightly higher than the normal distribution kurtosis of 3. The skewness is very close to zero (-0.028), indicating near symmetry. The Filliben correlation coefficient is also very close to one (0.994), suggesting that the residuals are nearly distributed as $N(0, 1)$.

```
m1_g <- gamlssML(imports~1, data=data, family= BCTo, trace=FALSE)
summary(m1_g)
```

Model Summary

```
## ******************************************************************
## Family:  c("BCTo", "Box-Cox-t-orig.")
##
## Call:
## gamlssML(formula = imports ~ 1, family = BCTo, data = data, trace = FALSE)
##
## Fitting method: "nlminb"
```

```
##
##
## Coefficient(s):
##             Estimate  Std. Error   t value   Pr(>|t|)
## eta.mu     3.7598909   0.0370812 101.39603 < 2.22e-16 ***
## eta.sigma -0.8791842   0.0819742 -10.72514 < 2.22e-16 ***
## eta.nu     0.3180417   0.1457615   2.18193   0.029115 *
## eta.tau    1.7805235   0.3811292   4.67171 2.9871e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   163
## Global Deviance:     1481.33
##             AIC:     1489.33
##             SBC:     1501.8
```

The **Box-Cox t-Original (BCTo)** distribution is a flexible probabilistic model suitable for modeling positive data with skewness and heavy tails. This model is characterized by four key parameters:

- η_μ: location (central tendency)
- η_σ: scale (dispersion)
- η_ν: shape (asymmetry)
- η_τ: tail heaviness (kurtosis)

This distribution is particularly useful when modeling data like the `imports` variable, which often exhibits skewness and does not conform to a normal distribution. The model was fitted using **Maximum Likelihood Estimation (MLE)**.

The parameter estimates are as follows:

- $\eta_\mu = 3.76$ ($p < 2.22 \times 10^{-16}$): Indicates the central tendency of the distribution, showing a strong and highly significant result.
- $\eta_\sigma = -0.88$ ($p < 2.22 \times 10^{-16}$): Reflects a moderate dispersion in the data, with a very significant p-value.
- $\eta_\nu = 0.32$ ($p = 0.029$): Indicates the asymmetry of the distribution. However, its significance is lower compared to the other parameters, with a p-value of 0.029, suggesting that it may not be as impactful in capturing the skewness of the data.
- $\eta_\tau = 1.78$ ($p < 2.22 \times 10^{-16}$): Represents the tail heaviness, showing a strong and highly significant result.

The model achieved the following goodness-of-fit metrics:

- **Global Deviance**: 1481.33
- **AIC**: 1489.33
- **SBC**: 1501.8

These metrics suggest that the Box-Cox t-Original distribution provides a good fit for the `imports` data, capturing its skewness, dispersion, and tail behavior effectively. However, the relatively low significance of the η_ν parameter indicates that the impact of the skewness in this particular dataset is less pronounced compared to the other factors.

Kolmogorov-Smirnov test The Kolmogorov-Smirnov test is a goodness-of-fit test used to compare two cumulative distribution functions, which can either be theoretical or empirical.

In this case, I compared the empirical cumulative distribution function (ECDF $\phi(x)$) of my data with the theoretical cumulative distribution function of the Box-Cox t-distribution $F(x)$. The hypotheses for the test are as follows:$$

$$H_0 : X \sim BCTo(\mu, \sigma, \nu, \tau) \qquad H_1 : not\ H_0$$

The test statistics is computed as follow:

$$\Delta_{\text{one-sample}} = \max\{|F(x) - \phi(x)|\}$$

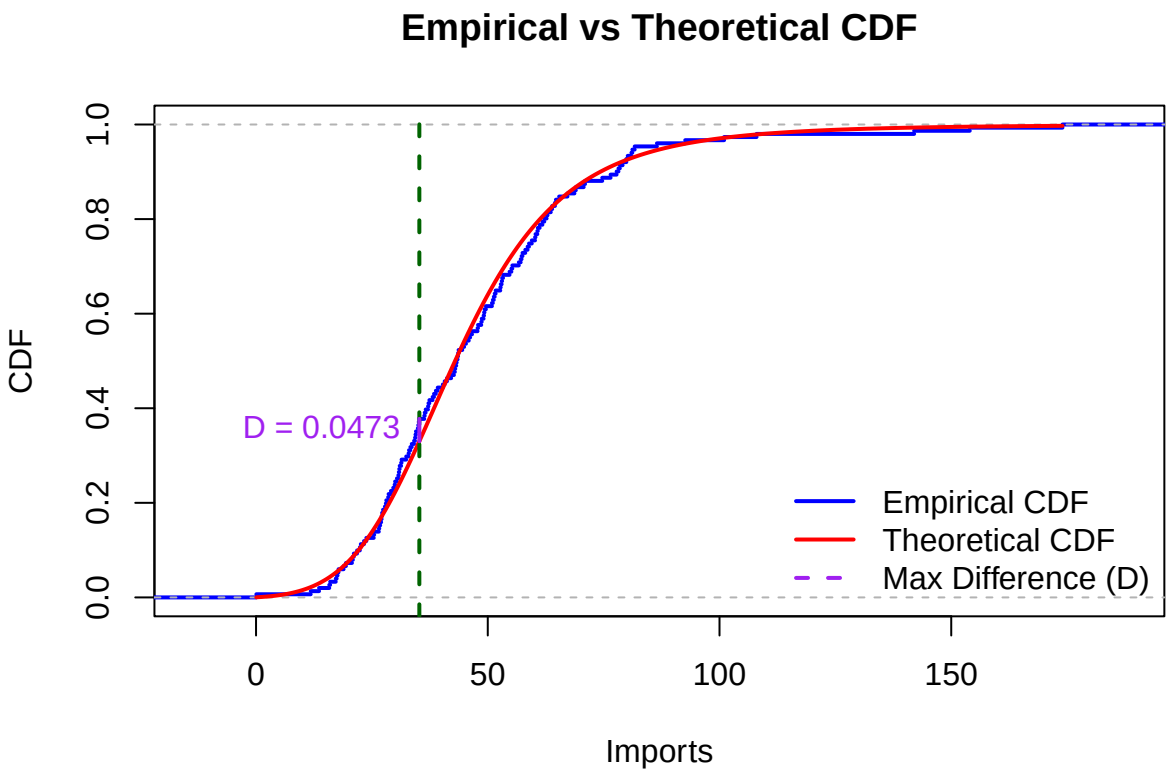
```
theoretical_cdf <- function(x) {
  pBCT(x, mu = m1$mu, sigma = m1$sigma, nu = m1$nu, tau = m1$tau)
}

ks_test_res <- ks.test(unique(imports), theoretical_cdf)

ks_test_res
```

```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  unique(imports)
## D = 0.048047, p-value = 0.8767
## alternative hypothesis: two-sided
```

In this case, the Kolmogorov-Smirnov test statistic ($\Delta_{\text{one-sample}}$) represents the maximum vertical distance between the empirical and the theoretical cumulative distribution functions. Given that this distance is very small (with a p-value of 0.8767), we can conclude that there is no significant difference between the two distributions.

Since the test statistic is so small and the p-value is large, we fail to reject the null hypothesis. The data appear to follow the Box-Cox t-distribution, and we do not have enough evidence to suggest otherwise. The similarity between the two CDFs further supports the idea that the Box-Cox t-distribution is an appropriate model for the data.

```
### Graphic ##############################################################################
ecdf_data <- ecdf(unique(imports))
x_vals <- seq(min(imports), max(imports), length.out = 1000)
cdf_theoretical <- theoretical_cdf(x_vals)

differences <- abs(ecdf_data(x_vals) - cdf_theoretical)
max_D <- max(differences)
x_at_max_D <- x_vals[which.max(differences)]



plot(ecdf_data, col = "blue", lwd = 2, main = "Empirical vs Theoretical CDF",
     xlab = "Imports", ylab = "CDF", do.points = FALSE)
lines(x_vals, cdf_theoretical, col = "red", lwd = 2)

abline(v = x_at_max_D, col = "darkgreen", lwd = 2, lty = 2)
segments(x_at_max_D, ecdf_data(x_at_max_D), x_at_max_D,
         cdf_theoretical[which.max(differences)], col = "purple", lwd = 2)

text(x_at_max_D, mean(c(ecdf_data(x_at_max_D), cdf_theoretical[which.max(differences)])),
     labels = paste0("D = ", round(max_D, 4)), pos = 2, col = "purple")

legend("bottomright", legend = c("Empirical CDF", "Theoretical CDF", "Max Difference (D)"),
       col = c("blue", "red", "purple"), lwd = 2, lty = c(1, 1, 2), bty = "n")
```



Income

`income` is a continuous numerical variable defined on $\mathbb{R}^+$ with the following distribution:

```
summary(income)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     609    3355    9960   17145   22800  125000
```

```
sd(income)
```

```
## [1] 19278.07
```

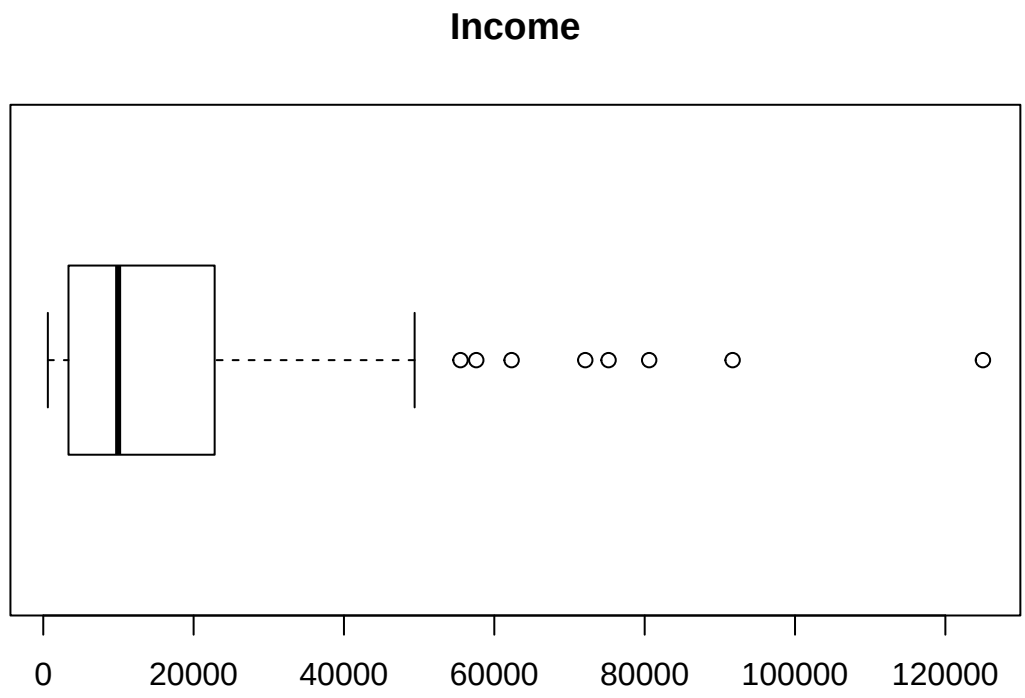
```
kurtosis(income, excess = T)
```

```
## [1] 7.028657
```

```
skewness(income)
```

```
## [1] 2.23148
```

```r
boxplot(income, horizontal = T, col = "white", main = "Income")
```

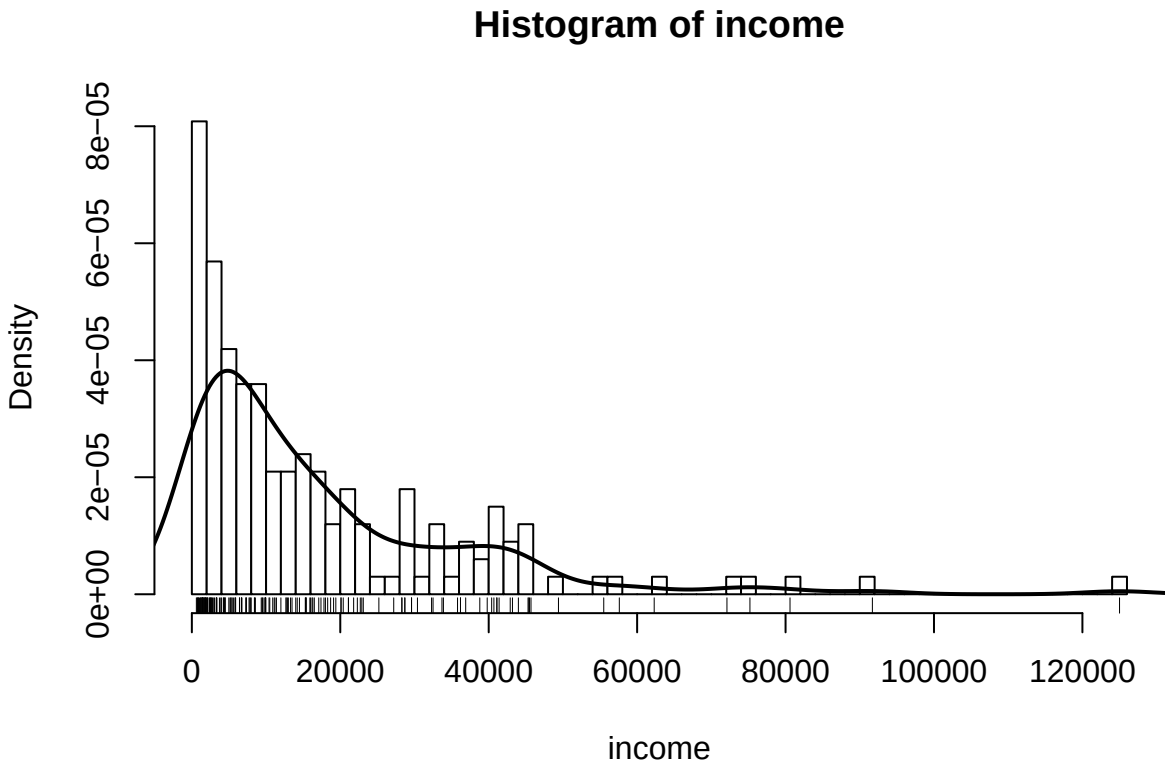



```r
# outliers
rownames(data[which(income %in% boxplot(income, plot = F)$out), ])
```

```
## [1] "Brunei"               "Kuwait"               "Luxembourg"
## [4] "Norway"               "Qatar"                "Singapore"
## [7] "Switzerland"          "United Arab Emirates"
```

The `income` variable ranges from 609 to 125,000, with a high excess kurtosis of 7.03, indicating a distribution that is more peaked and has heavier tails than a normal distribution. Most data points fall within the interquartile range of 3,355 to 22,800, suggesting that the majority of countries have similar income levels. The distribution is positively skewed (skewness of 2.23), meaning there is a longer tail on the higher-income side. Outliers, including wealthy nations like Brunei, Kuwait, Luxembourg, and others, significantly surpass the income levels of the rest of the countries in the dataset.

```r
hist(income, col = "white", freq = F, breaks = 60)
lines(density(income), col = "black", lwd = 2)
rug(income)
```



The shape of the Kernel Density Estimate (KDE) suggests the possibility of fitting a Gaussian Mixture Model (GMM), as the distribution clearly exhibits a multimodal structure. This indicates the presence of multiple subpopulations or groups within the data, which the GMM can capture effectively.

```
mod1 <- fitDist(income, data=data, k=2, type = "realplus")
```

Parametric Modeling

```
## Warning in MLE(ll4, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma, :
## possible convergence problem: optim gave code=1 function evaluation limit
## reached without convergence (9)
```

```
mod2 <- fitDist(income, data=data, k=log(dim(data)[1]), type = "realplus")
```

```
## Warning in MLE(ll4, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma, :
## possible convergence problem: optim gave code=1 function evaluation limit
## reached without convergence (9)
```

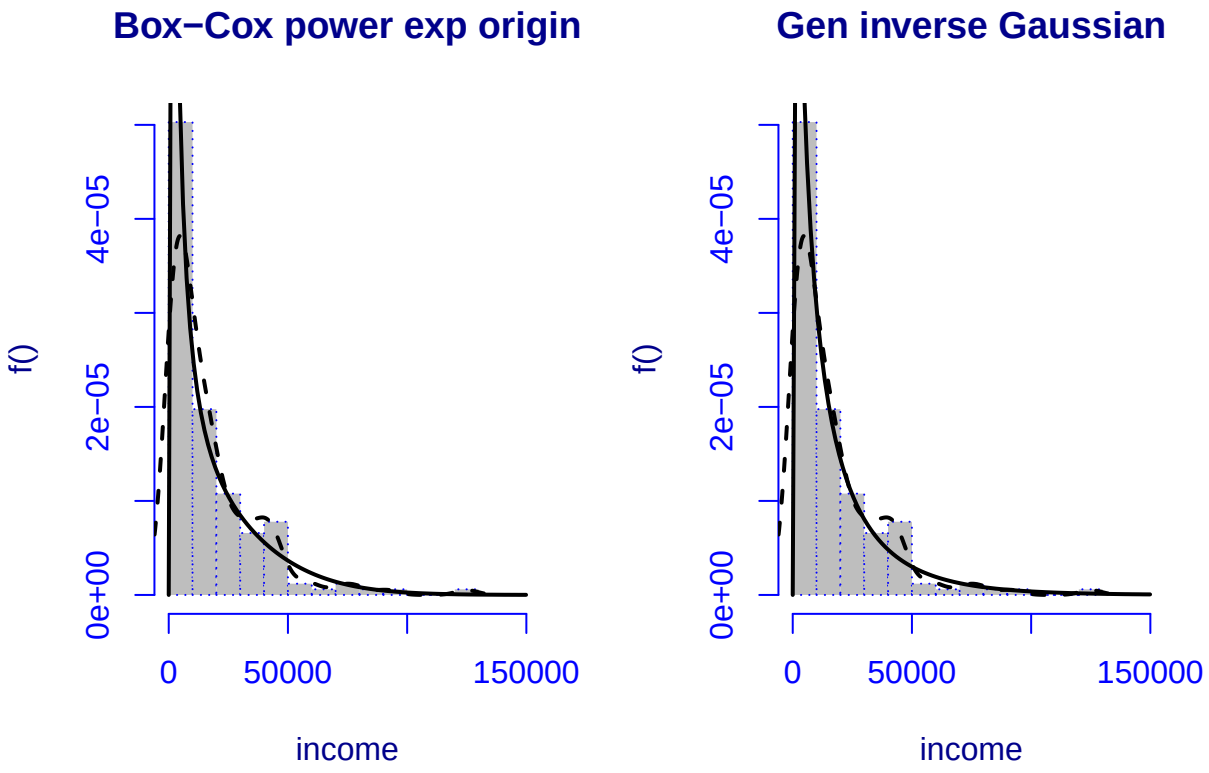
```
mod1$fits[1:6]
```

```
##    BCPEo     BCPE      GIG       GG     BCCG    BCCGo
## 3578.413 3578.413 3580.287 3589.531 3589.985 3589.985
```

```
mod2$fits[1:6]
```

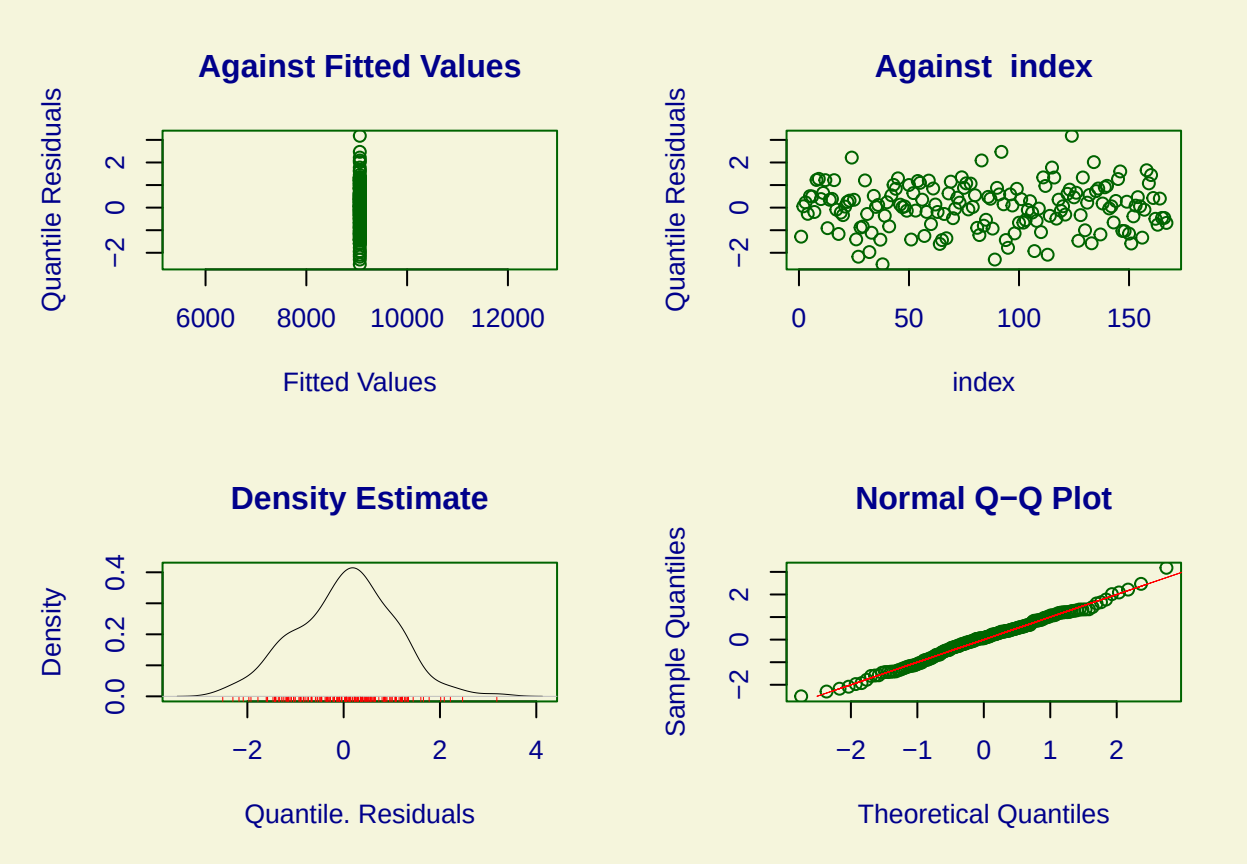
```
##      GIG    BCPEo     BCPE      EXP   LOGNO2    LOGNO
## 3589.641 3590.885 3590.885 3595.432 3596.568 3596.568
```

```
par(mfrow = c(1, 2))
m1 <- histDist(income, "BCPEo" , density=TRUE, line.col=c(1,1), line.ty=c(1,2),
               breaks = 50, main = "Box-Cox power exp origin")

m2 <- histDist(income, "GIG" , density=TRUE, line.col=c(1,1), line.ty=c(1,2),
               breaks = 50, main = "Gen inverse Gaussian")
```



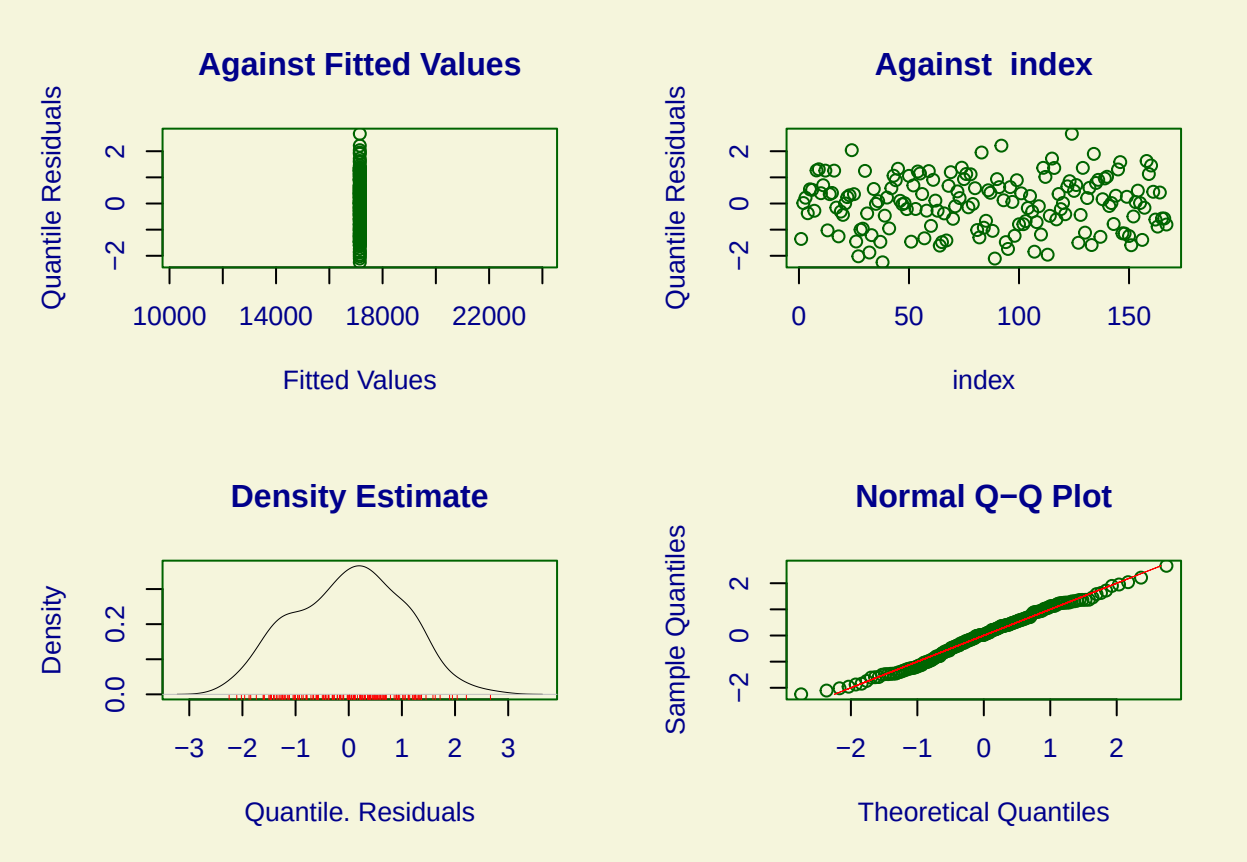
According to AIC the best model is the *Box-Cox power exp origin* instad cosidering BIC *Gen inverse Gaussian*.

```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  0.02990454
##                         variance   =  1.002757
##                coef. of skewness   =  -0.0289727
##                coef. of kurtosis   =  3.002118
## Filliben correlation coefficient   =  0.9967539
## ****************************************************************
```

```
plot(m2)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  0.0008216531
##                         variance   =  1.016742
##                 coef. of skewness  =  -0.04205974
##                 coef. of kurtosis  =  2.373241
## Filliben correlation coefficient   =  0.995173
## ****************************************************************
```

Residual distribution of models comparaison

1. **First Model (Box-Cox Power Exponential Origin - BCTo)**
   - **Mean**: 0.0299 (near zero), **Variance**: 1.0028 (close to 1).
   - **Skewness**: −0.029, **Kurtosis**: 3.0021 (close to normal).
   - **Filliben**: 0.9968 indicating near-normal residuals.
2. **Second Model (Generalized Inverse Gaussian - GIG)**
   - **Mean**: 0.0008 (close to zero), **Variance**: 1.0167 (slightly higher).
   - **Skewness**: −0.042, **Kurtosis**: 2.3732 (slightly platykurtic, wich means thinner tails).
   - **Filliben**: 0.9952 also approximates normality.

Despite the second model having a better BIC, the **first model (BCPEo)** is preferred due to its better AIC, closer match to a normal distribution (looking at Filliben statistic) in residuals, and overall better fit.

```
fit_mix <- mixfit(income,
             family = "gamma",
             ncomp = 2,
```

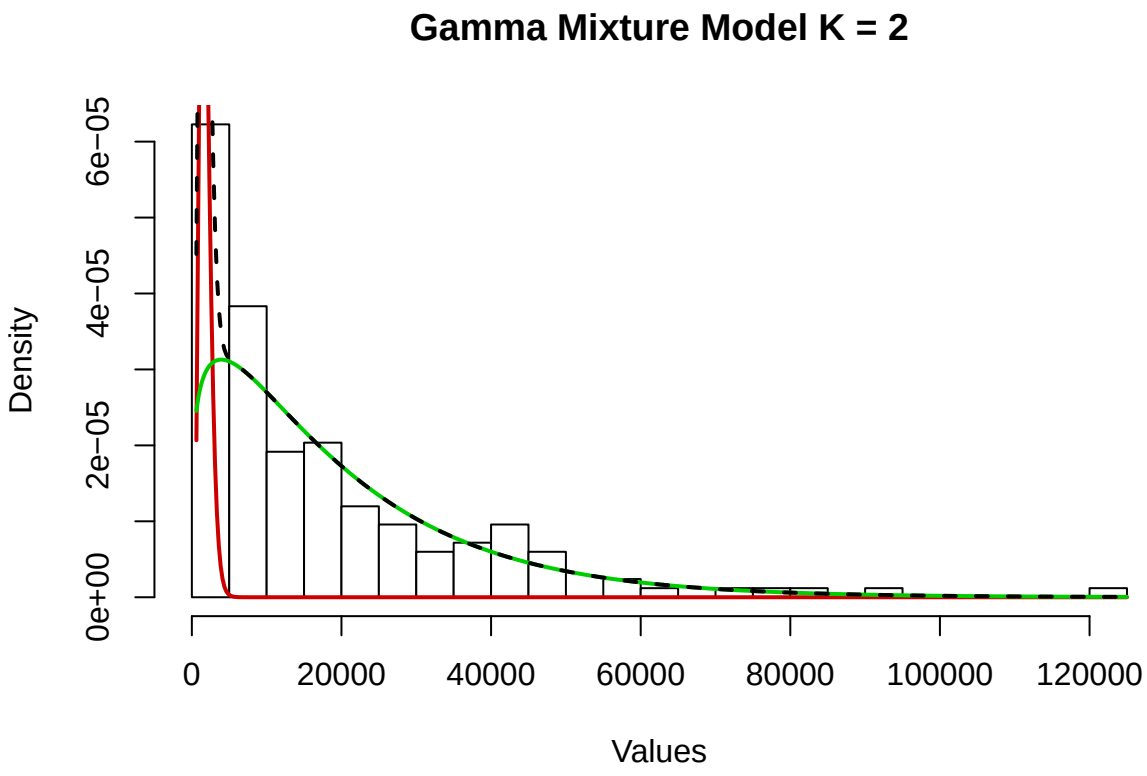
```
             tol = 1e-06)

fit_mix
```

Gamma Mixture Model

```
## Gamma mixture model with 2 components
##              comp1        comp2
## pi       0.1685635 8.314365e-01
## mu    1794.2163939 2.025681e+04
## sd     725.6674318 1.819507e+04
## shape    6.1132813 1.239465e+00
## rate     0.0034072 6.120000e-05
##
## EM iterations: 162 AIC: 3580.83 BIC: 3596.42 log-likelihood: -1785.42
```

Based on the AIC and BIC values, this model performs worse compared to the two previously considered models.

```
#############
##  Plot  ##
#############

colors <- c("red3", "green3")

weights <- fit_mix$pi
mu <- fit_mix$mu
shape <- fit_mix$alpha
rate <- fit_mix$lambda

hist(income, breaks = 30, probability = T, col = "white", border = "black",
     main = "Gamma Mixture Model K = 2",
     xlab = "Values", ylab = "Density")

x_vals <- seq(min(income), max(income), length.out = 500)

for (i in 1:length(weights)) {
  lines(x_vals, weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i]), col = colors[i], lwd = 2)
}

overall_density <- rowSums(sapply(1:length(weights), function(i) {
  weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i])
}))

lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)
```




```
m1_g <- gamlssML(income~1, data= data, family= BCPE, trace=T)
summary(m1_g)
```

Model Summary

```
## ******************************************************************
## Family:  c("BCPE", "Box-Cox Power Exponential")
##
## Call:  gamlssML(formula = income ~ 1, family = BCPE, data = data, trace = T)
##
## Fitting method: "nlminb"
##
##
## Coefficient(s):
##             Estimate  Std. Error  t value   Pr(>|t|)
## eta.mu    9.06091e+03 1.09258e+03  8.29315 < 2.22e-16 ***
## eta.sigma 2.00312e-01 4.08369e-02  4.90516 9.3350e-07 ***
## eta.nu    5.79194e-02 5.27577e-02  1.09784    0.27228
## eta.tau   1.56072e+00 2.38687e-01  6.53877 6.2026e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   163
## Global Deviance:     3570.41
##             AIC:     3578.41
```

```
##             SBC:     3590.89
```

Box-Cox Power Exponential (BCPE) Distribution

The **Box-Cox Power Exponential (BCPE)** distribution was fitted to the income data using the **Maximum Likelihood Estimation (MLE)** method with **"nlminb"** optimization.

Parameter Estimates:

- $\eta_\mu = 9060.91$ (p < 2.22e-16): Highly significant location parameter.
- $\eta_\sigma = 0.2003$ (p = 9.34e-07): Highly significant scale parameter.
- $\eta_\nu = 0.0579$ (p = 0.272): Not statistically significant.
- $\eta_\tau = 1.5607$ (p = 6.20e-11): Highly significant shape parameter.

Goodness-of-Fit:

- **Global Deviance**: 3570.41
- **AIC**: 3578.41
- **SBC**: 3590.89

The **BCPE** distribution fits the income data well, with significant parameters for location, scale, and shape, while the skewness parameter is less relevant.

Inflation

`inflation` is a continuous numerical variable defined on $\mathbb{R}$ with the following distribution:

```
summary(inflation)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -4.210   1.810   5.390   7.782  10.750 104.000
```

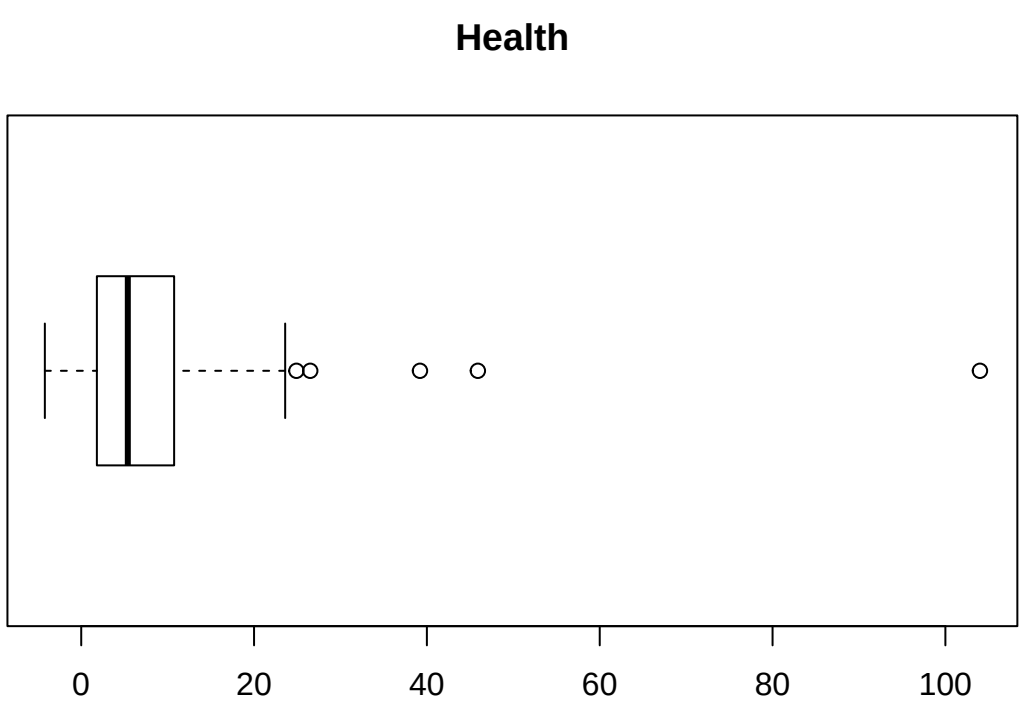
```
kurtosis(inflation, excess = T)
```

```
## [1] 41.7425
```

```
skewness(inflation)
```

```
## [1] 5.154049
```

```
boxplot(inflation, horizontal = T, col = "white", main = "Health")
```

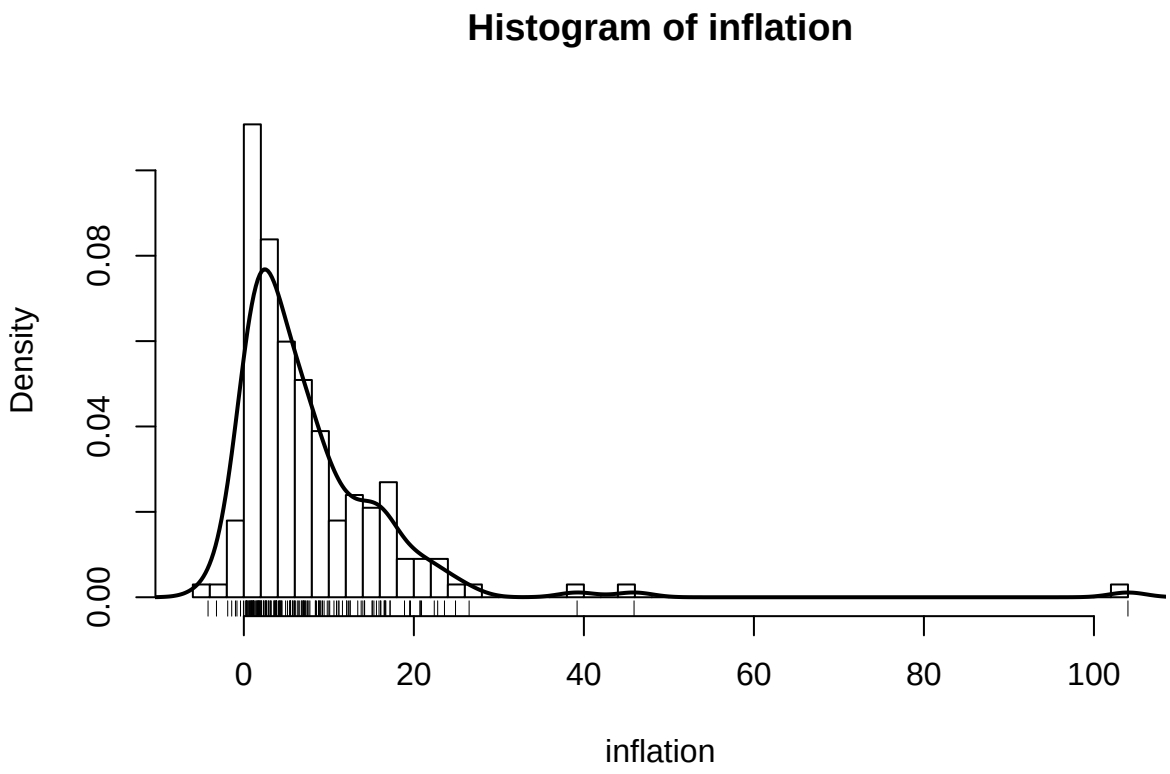



```
# outliers
rownames(data[which(inflation %in% boxplot(inflation, plot = F)$out), ])
```

```
## [1] "Equatorial Guinea" "Mongolia"          "Nigeria"
## [4] "Timor-Leste"       "Venezuela"
```

The `inflation` variable ranges from -4.21 to 104, with a mean value of 7.78. The distribution shows a substantial positive skew, with a skewness of 41.74, indicating a significant imbalance and a longer tail on the right. The interquartile range spans from approximately 1.81 to 10.75, suggesting that most countries have moderate inflation rates. Outliers include nations like Equatorial Guinea, Mongolia, Nigeria, Timor-Leste, and Venezuela, which have much higher inflation rates compared to the rest of the dataset. The kurtosis of the distribution is 5.15, reflecting a distribution with a peak more pronounced than a normal distribution.

```
hist(inflation, col = "white", freq = F, breaks = 60)
lines(density(inflation), col = "black", lwd = 2)
rug(inflation)
```



The `inflation` data likely comes from two distinct populations. A Gaussian Mixture Model (GMM) could help capture this by modeling the data as a combination of two Gaussian distributions, accounting for both moderate and extreme inflation values.

```
transformed_inf <- inflation[inflation > 0]
```

```
mod1$fits[1:6]
```

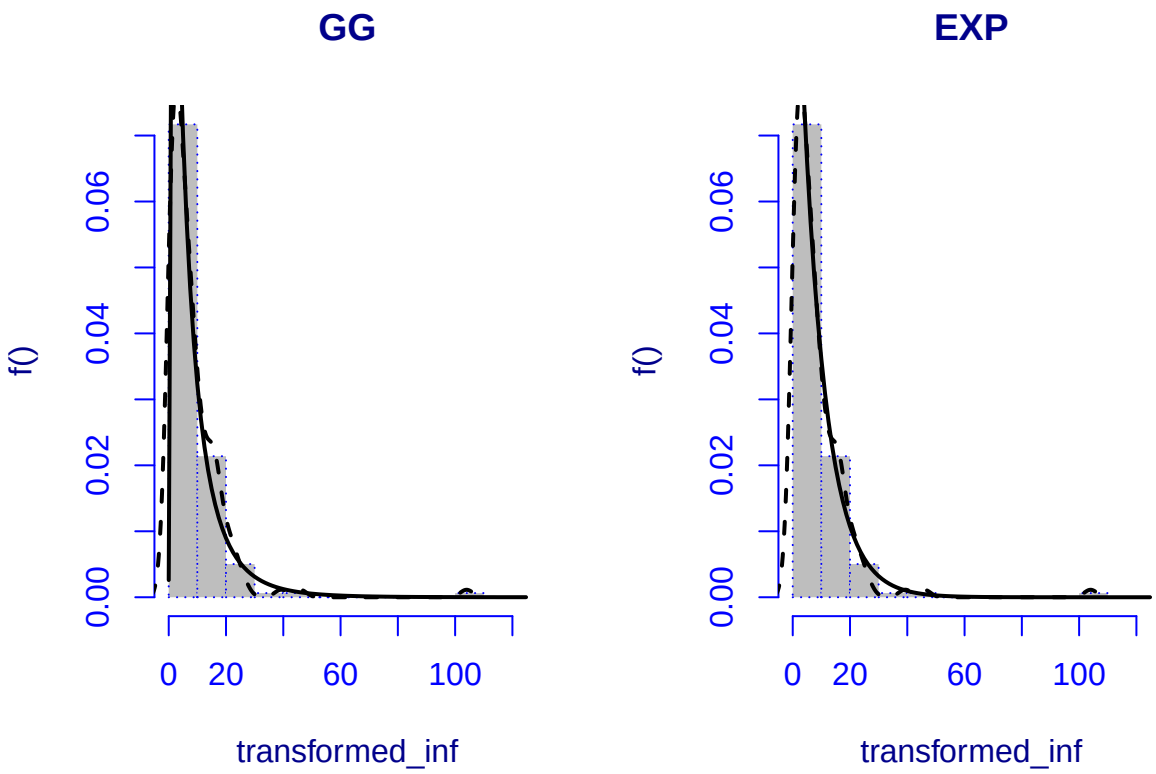
Parametric Modeling

```
##       GG     BCCG    BCCGo      GIG  PARETO2 PARETO2o
## 988.1921 988.3887 988.3887 988.5799 988.6934 988.6934
```

```
mod2$fits[1:6]
```

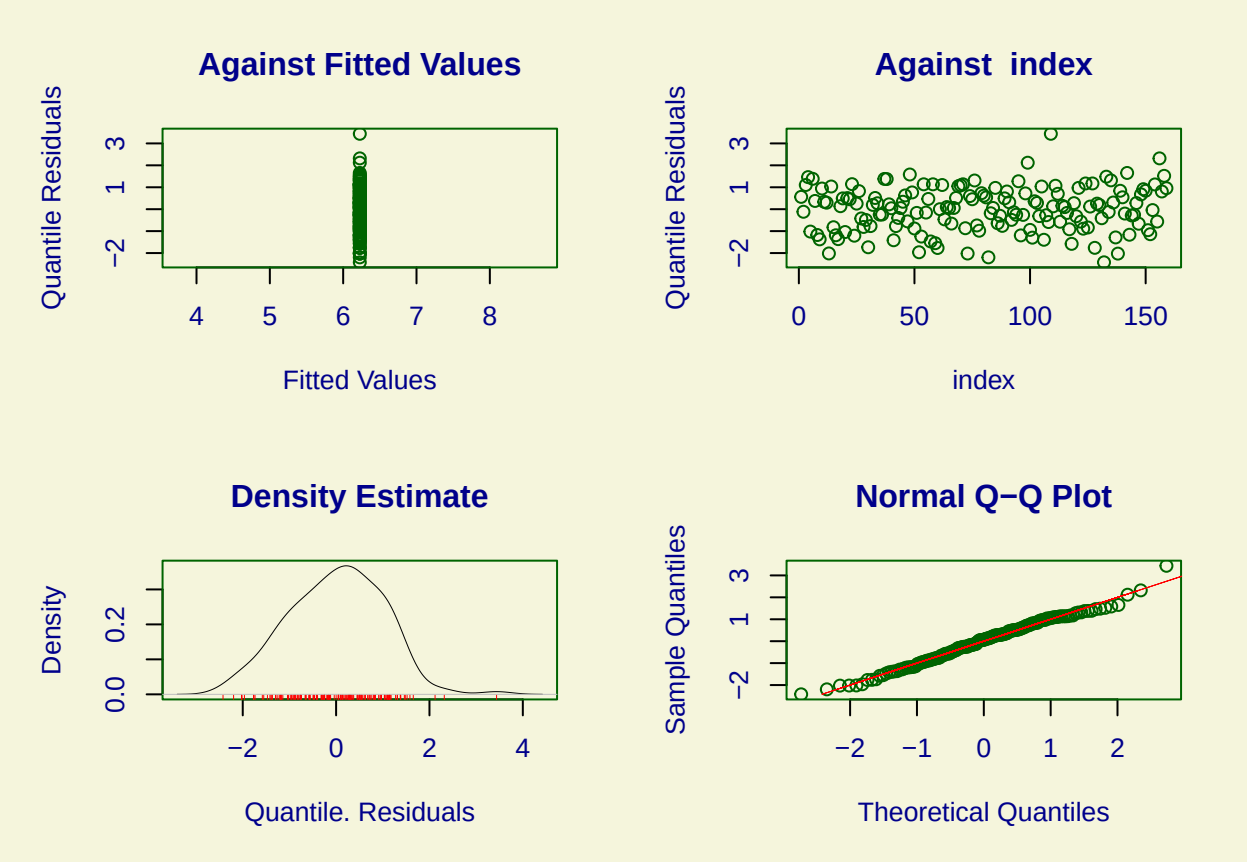
```
##      EXP  PARETO2 PARETO2o       GP       GG     BCCG
## 994.3631 994.9294 994.9294 994.9294 997.5460 997.7426
```

```
par(mfrow = c(1, 2))
m1 <- histDist(transformed_inf, "GG" , density=TRUE, line.col=c(1,1), line.ty=c(1,2), breaks = 50,
               main = "GG")
m2 <- histDist(transformed_inf, "EXP" , density=TRUE, line.col=c(1,1), line.ty=c(1,2), breaks = 50,
               main = "EXP")
```



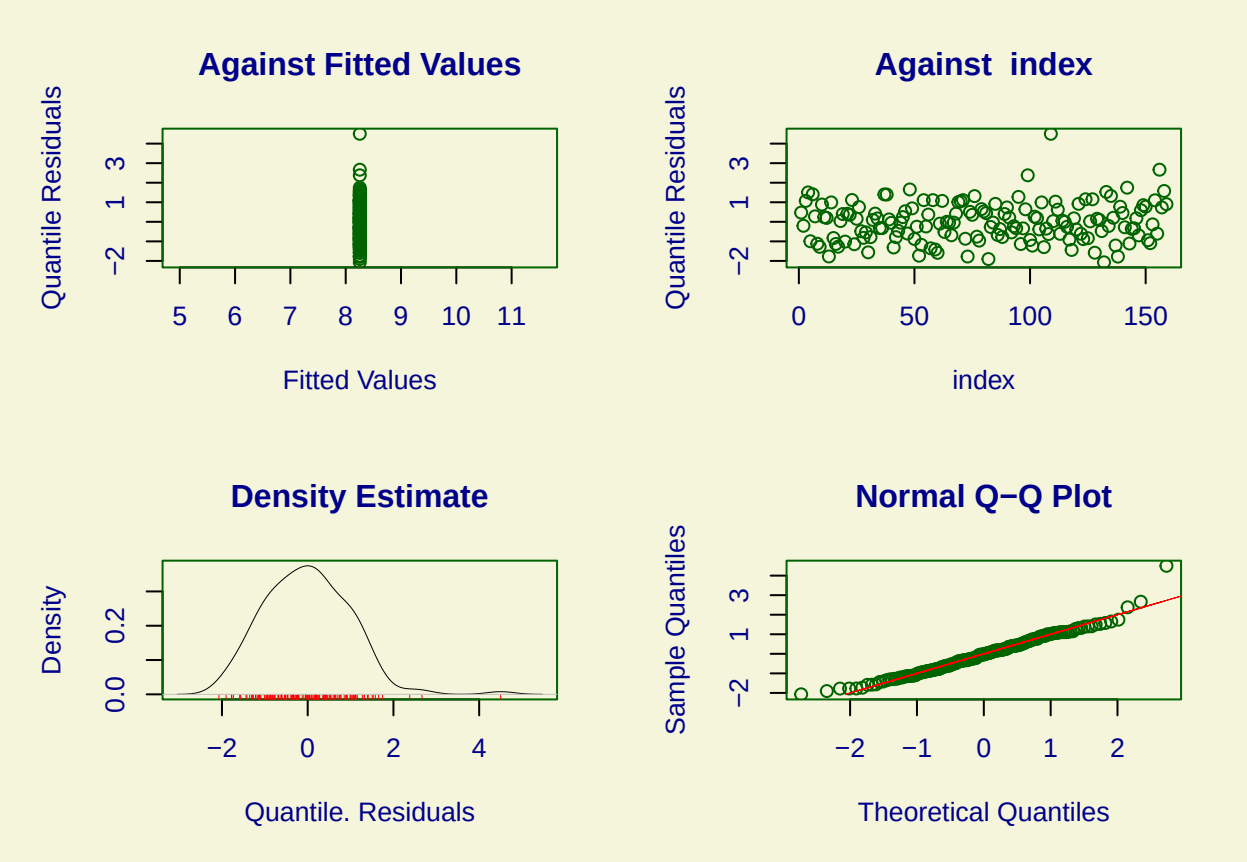
According to the AIC criterion, the best-fitting model is the *Generalized Gamma*, while the BIC suggests the *Exponential* distribution as the optimal choice. Negative inflation values were excluded from the analysis due to issues encountered during the optimization process. As a result, the models presented consider only inflationary scenarios, excluding cases of deflation.

```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  -6.620257e-05
##                         variance   =  1.006276
##                coef. of skewness   =  0.008641209
##                coef. of kurtosis   =  2.986467
## Filliben correlation coefficient   =  0.9945117
## ****************************************************************
```

```
plot(m2)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  -0.01267458
##                         variance   =  0.9901408
##                coef. of skewness   =  0.6176818
##                coef. of kurtosis   =  4.593667
## Filliben correlation coefficient   =  0.9841783
## ****************************************************************
```

Residual Distribution Comparison of Models

1. **First Model (Generalized Gamma - GG)**

- **Mean**: -6.62e-05 (close to zero), **Variance**: 1.0063 (approximately 1).
- **Skewness**: 0.0086, **Kurtosis**: 2.9865 (almost normal, slightly platykurtic).
- **Filliben**: 0.9945, indicating that the residuals are nearly normal.

2. **Second Model (Exponential - EXP)**

- **Mean**: -0.0127 (close to zero), **Variance**: 0.9901 (slightly lower).
- **Skewness**: 0.6177 (moderately skewed), **Kurtosis**: 4.5937 (leptokurtic, indicating heavier tails).
- **Filliben**: 0.9842, suggesting a departure from normality in the residuals.

While the second model (*EXP*) demonstrates a slightly lower variance and a better BIC, the **first model (GG)** is preferred due to its more favorable AIC, better alignment with normal distribution (as indicated by the Filliben statistic), and overall better fit to the data.

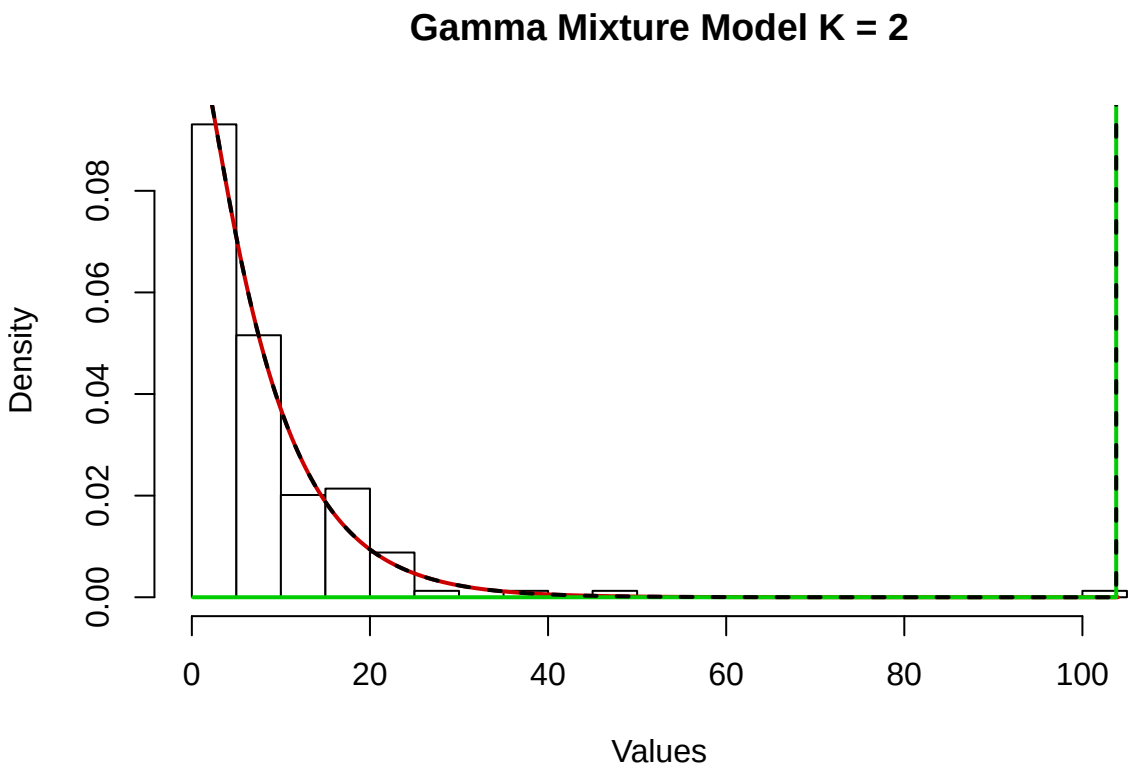
```r
fit_mix <- mixfit(transformed_inf,
             family = "gamma",
             ncomp = 2,
             tol = 1e-06)

fit_mix
```

Gamma Mixture model

```
## Gamma mixture model with 2 components
##           comp1        comp2
## pi    0.9937107 6.289300e-03
## mu    7.6491392 1.040000e+02
## sd    7.2888824 1.570000e-05
## shape 1.1012939 4.398047e+13
## rate  0.1439762 4.228891e+11
##
## EM iterations: 116 AIC: 959.86 BIC: 975.21 log-likelihood: -474.93
```

In this application, the Gamma mixture with K=2 provides a better fit for the data, as indicated by both the AIC and BIC.

```r
#############
##  Plot  ##
#############

colors <- c("red3", "green3")

weights <- fit_mix$pi
mu <- fit_mix$mu
shape <- fit_mix$alpha
rate <- fit_mix$lambda

hist(transformed_inf, breaks = 30, probability = T, col = "white", border = "black",
     main = "Gamma Mixture Model K = 2",
     xlab = "Values", ylab = "Density")

x_vals <- seq(min(transformed_inf), max(transformed_inf), length.out = 500)

for (i in 1:length(weights)) {
  lines(x_vals, weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i]), col = colors[i], lwd = 2
}

overall_density <- rowSums(sapply(1:length(weights), function(i) {
  weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i])
}))

lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)
```



This reveals an interesting situation: the EM algorithm appears to create a model primarily for outliers, which seem to be significant in terms of the data distribution. In my opinion, such a context would typically warrant the application of Extreme Value Theory, but this goes beyond the scope of this analysis.

Despite this, we can still obtain a model that fits the data better compared to simpler, single models.

```
m1_g <- gamlssML(transformed_inf~1, data=data, family= GG, trace=FALSE)
summary(m1_g)
```

Model Summary

```
## ******************************************************************
## Family:  c("GG", "generalised Gamma Lopatatsidis-Green")
##
## Call:  gamlssML(formula = transformed_inf ~ 1, family = GG, data = data,
##     trace = FALSE)
##
## Fitting method: "nlminb"
##
##
## Coefficient(s):
##             Estimate  Std. Error  t value Pr(>|t|)
## eta.mu    1.8288727   0.1386354 13.19196  < 2e-16 ***
## eta.sigma 0.0900904   0.0626383  1.43827 0.150359
## eta.nu    0.4688121   0.1888027  2.48308 0.013025 *
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Degrees of Freedom for the fit: 3 Residual Deg. of Freedom   156
## Global Deviance:     982.192
##             AIC:     988.192
##             SBC:     997.399
```

Generalized Gamma (GG) Distribution

The **Generalized Gamma (GG)** distribution is a flexible probabilistic model suitable for data with skewness and heavy tails. This model is characterized by three key parameters:

- η_μ: Location (central tendency)
- η_σ: Scale (dispersion)
- η_ν: Shape (asymmetry)

For the inflation data, which may exhibit skewness and deviate from a normal distribution, this model proved useful. It was fitted using **Maximum Likelihood Estimation (MLE)**.

Parameter Estimates:

- $\eta_\mu = 1.83$ (p < 2e-16): The central tendency of the distribution, with a highly significant result.
- $\eta_\sigma = 0.09$ (p = 0.150): Dispersion, with a p-value suggesting that it is not statistically significant.
- $\eta_\nu = 0.47$ (p = 0.013): The shape of the distribution, capturing the skewness of the data, with moderate significance.

Goodness-of-Fit Metrics:

- **Global Deviance**: 982.192
- **AIC**: 988.192
- **SBC**: 997.546

These results suggest that the **Generalized Gamma (GG)** distribution provides a good fit for the inflation data, effectively capturing its skewness and dispersion. However, the relatively low significance of the η_σ parameter indicates that the dispersion may not be as impactful in this particular dataset.

Life Expectation

The variable `life_expec` is numerical distributed on the $\mathbb{R}^+$

```
summary(life_expec)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   32.10   65.30   73.10   70.56   76.80   82.80
```

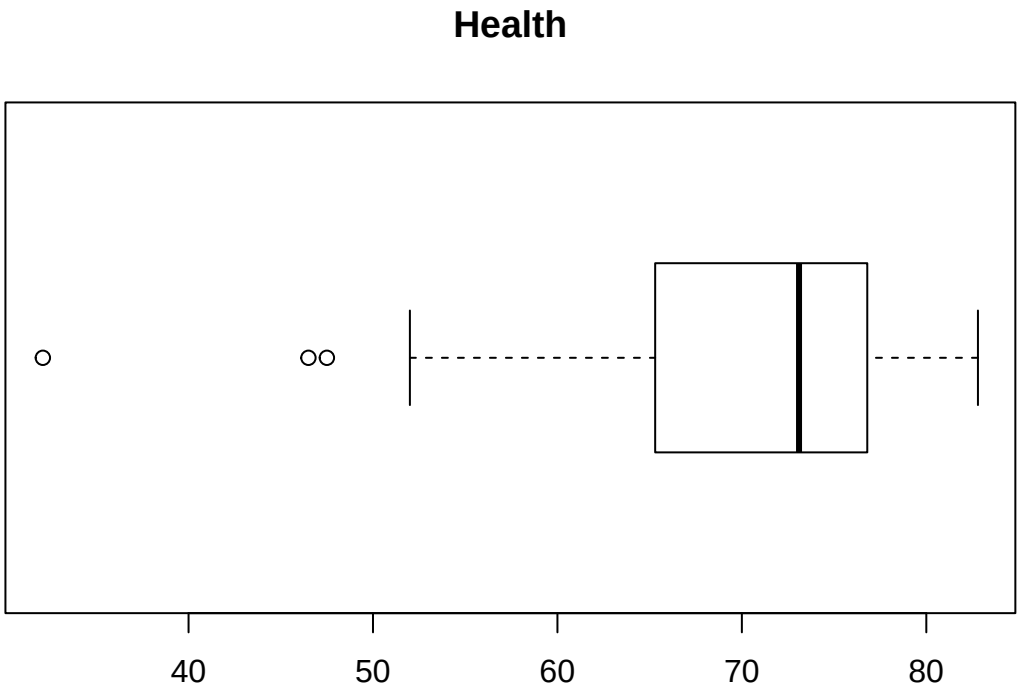
```
kurtosis(life_expec, excess = T)
```

```
## [1] 1.151591
```

```
skewness(life_expec)
```

```
## [1] -0.9709956
```

```
boxplot(life_expec, horizontal = T, col = "white", main = "Health")
```




```
# outliers
rownames(data[which(life_expec %in% boxplot(life_expec, plot = F)$out), ])
```

```
## [1] "Central African Republic" "Haiti"
## [3] "Lesotho"
```

The `life_expect` variable ranges from 32.10 to 82.80 years, with a mean of 70.56. The distribution has a slight negative skew (-0.97), indicating a longer left tail. The interquartile range is between 65.30 and 76.80, with most countries having moderate life expectancy. Outliers include countries like the Central African Republic, Haiti, and Lesotho. The kurtosis is 1.15, indicating a less peaked distribution than normal.

```
hist(life_expec, col = "white", freq = F, breaks = 60)
lines(density(life_expec), col = "black", lwd = 2)
rug(life_expec)
```

The observed multimodality in the distribution pattern clearly indicates the presence of multiple distinct populations in the dataset

```
mod1 <- fitDist(life_expec, data=data, k=2, type = "realplus")
```

Parametric Modelling

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
## Warning in nlminb(start = start, objective = f, control = optim.control):
## NA/NaN function evaluation
```

```
mod2 <- fitDist(life_expec, data=data, k=log(dim(data)[1]), type = "realplus")
```

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
## Warning in nlminb(start = start, objective = f, control = optim.control):
```

```
## NA/NaN function evaluation
```

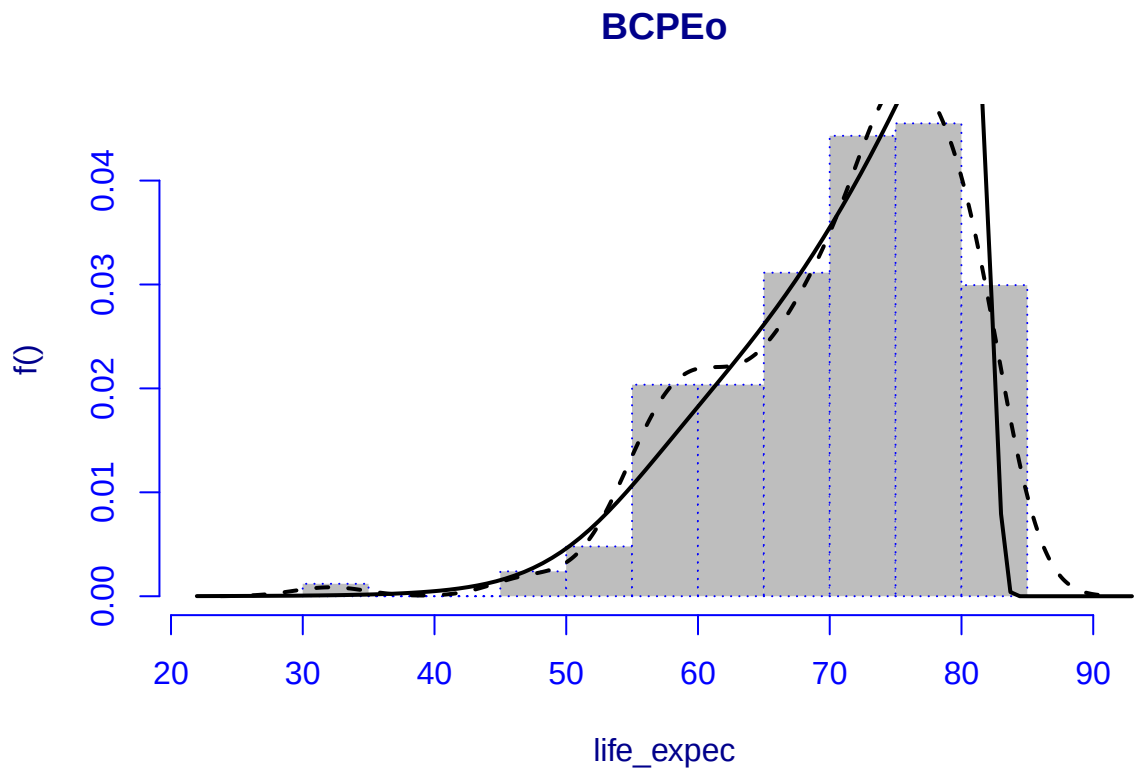
```
mod1$fits[1:6]
```

```
##    BCPEo     BCPE       GG      GB2    BCCGo     BCCG
## 1157.300 1157.300 1161.773 1163.773 1172.547 1172.547
```

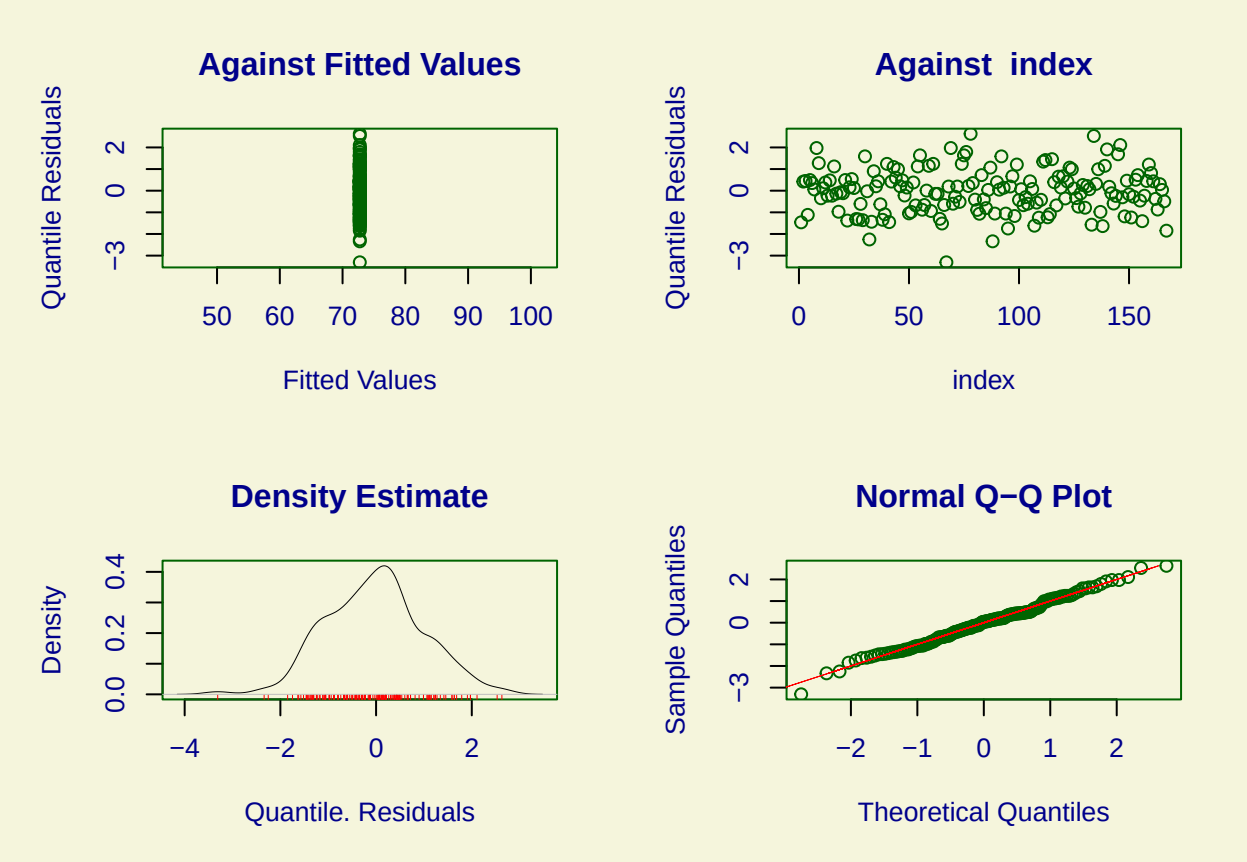
```
mod2$fits[1:6]
```

```
##    BCPEo     BCPE       GG      GB2    BCCGo     BCCG
## 1169.772 1169.772 1171.127 1176.245 1181.901 1181.901
```

```
m1 <- histDist(life_expec, "BCPEo" , density=TRUE, line.col=c(1,1), line.ty=c(1,2), breaks = 50,
               main = "BCPEo")
```




```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  -0.02877363
##                         variance   =  1.015257
##                coef. of skewness   =  0.01228459
##                coef. of kurtosis   =  3.058769
## Filliben correlation coefficient   =  0.9965929
## ****************************************************************
```

Residual Distribution of the Model

Model (Box-Cox Power Exponential Origin) - **Mean**: -0.029, **Variance**: 1.015 (approximately 1). - **Skewness**: 0.012, **Kurtosis**: 3.05. - **Filliben**: 0.996, indicating that the residuals are nearly normal.

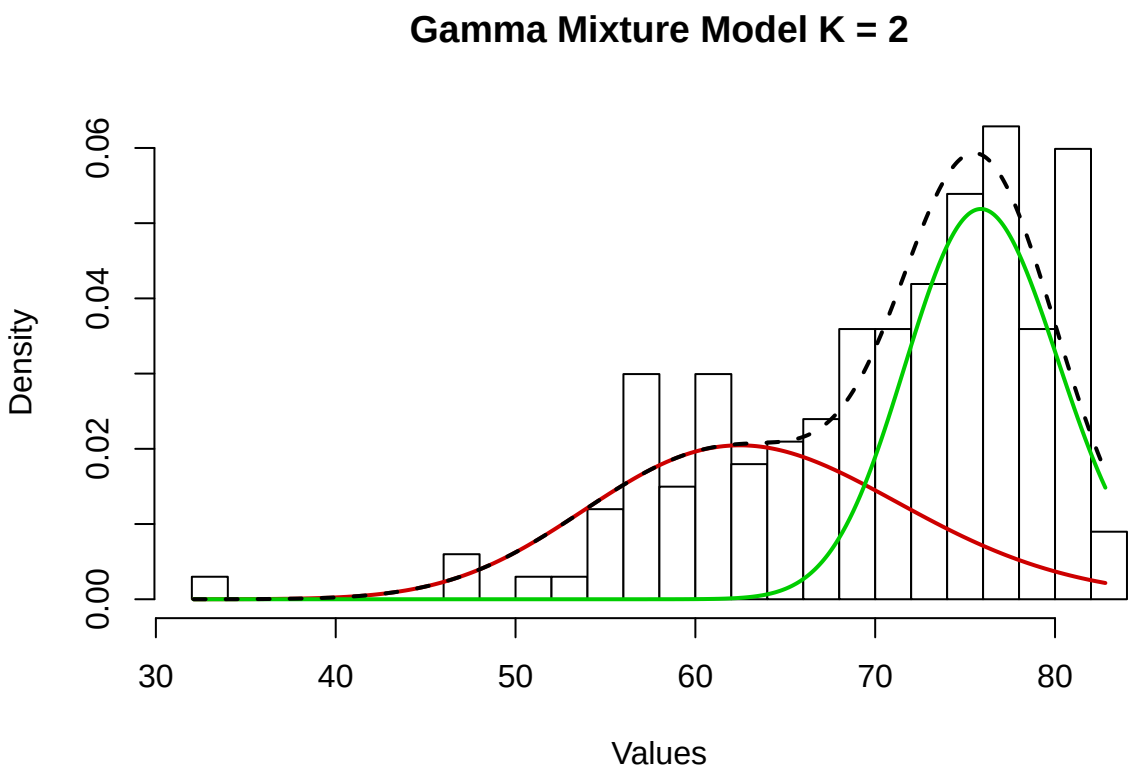
The **BCPEo** model is the best fit for our data, as indicated by its superior AIC and BIC values. Diagnostic plots of the residuals reveal that they follow a standard normal distribution, further supporting the model's suitability for the data.

```
fit_mix <- mixfit(life_expec,
             family = "gamma",
             ncomp = 2,
             tol = 1e-06)
fit_mix
```

Gamma Mixture model

```
## Gamma mixture model with 2 components
##            comp1       comp2
## pi     0.4472917   0.5527083
## mu    63.6865526  76.1146925
## sd     8.7893995   4.2549381
## shape 52.5021484 320.0003906
## rate   0.8243836   4.2041869
##
## EM iterations: 36 AIC: 1183.54 BIC: 1199.13 log-likelihood: -586.77
```

```
#############
##  Plot  ##
#############

colors <- c("red3", "green3")

weights <- fit_mix$pi
mu <- fit_mix$mu
shape <- fit_mix$alpha
rate <- fit_mix$lambda

hist(life_expec, breaks = 30, probability = T, col = "white", border = "black",
     main = "Gamma Mixture Model K = 2",
     xlab = "Values", ylab = "Density")

x_vals <- seq(min(life_expec), max(life_expec), length.out = 500)

for (i in 1:length(weights)) {
  lines(x_vals, weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i]), col = colors[i], lwd = 2)
}

overall_density <- rowSums(sapply(1:length(weights), function(i) {
  weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i])
}))

lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)
```




```
m1_g <- gamlssML(life_expec~1, data=data, family= BCPEo, trace=FALSE)
summary(m1_g)
```

Model Summary

```
## ******************************************************************
## Family:  c("BCPEo", "Box-Cox Power Exponential-orig.")
##
## Call:  gamlssML(formula = life_expec ~ 1, family = BCPEo, data = data,
##     trace = FALSE)
##
## Fitting method: "nlminb"
##
##
## Coefficient(s):
##              Estimate  Std. Error   t value   Pr(>|t|)
## eta.mu     4.28697291  0.00926891 462.51096 < 2.22e-16 ***
## eta.sigma -2.32754151  0.06337904 -36.72415 < 2.22e-16 ***
## eta.nu     5.11349191  0.53050001   9.63900 < 2.22e-16 ***
## eta.tau    2.29500513  0.33998650   6.75028 1.4756e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   163
## Global Deviance:     1149.3
```

```
##             AIC:     1157.3
##             SBC:     1169.77
```

Box-Cox Power Exponential-orig (BCPEo) Distribution

The **Box-Cox Power Exponential-orig (BCPEo)** distribution was applied to the life expectancy data using **Maximum Likelihood Estimation (MLE)** with the **"nlminb"** method.

Parameter Estimates: -

- $\eta_\mu = 4.287$ (p < 2.22e-16): Highly significant location parameter, central tendency of the distribution.
- $\eta_\sigma = -2.328$ (p < 2.22e-16): Highly significant scale parameter, indicating strong dispersion.
- $\eta_\nu = 5.113$ (p < 2.22e-16): Highly significant shape parameter, capturing the asymmetry of the distribution.
- $\eta_\tau = 2.295$ (p = 1.48e-11): Highly significant shape parameter, influencing the heavy tails.

Goodness-of-Fit:

- **Global Deviance**: 1149.3
- **AIC**: 1157.3
- **SBC**: 1169.77

The **BCPEo** distribution provides a strong fit for the life expectancy data, with all parameters being highly significant, capturing the data's central tendency, dispersion, and asymmetry effectively.

Total Fertility

The variable `total_fer` is numerical distribuited on the $\mathbb{R}^+$

```
summary(total_fer)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.150   1.795   2.410   2.948   3.880   7.490
```

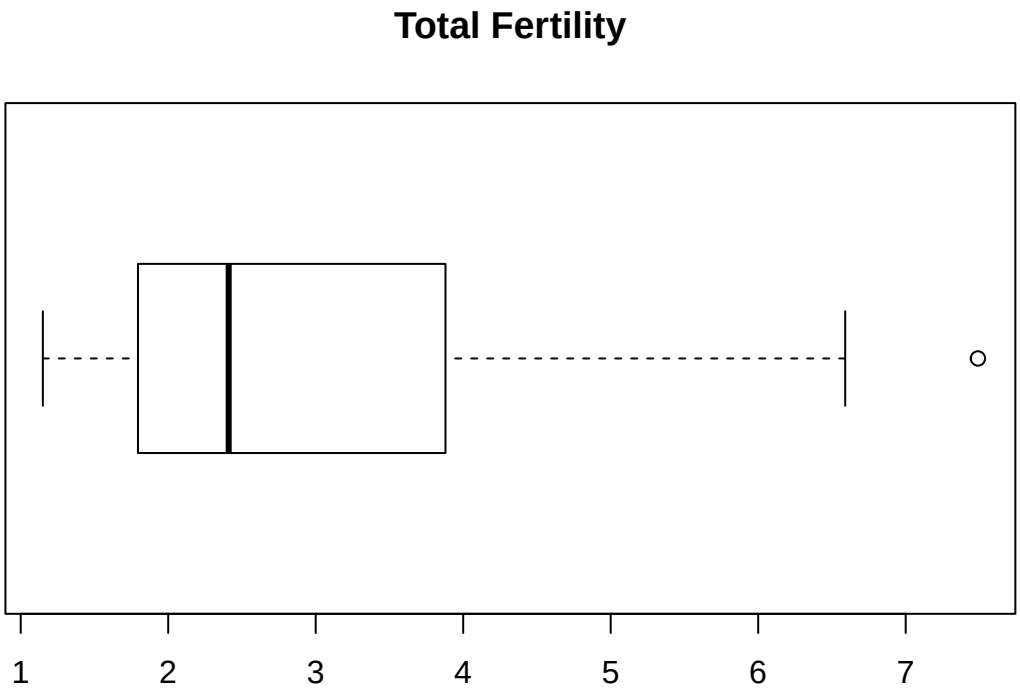
```
kurtosis(total_fer, excess = T)
```

```
## [1] -0.186779
```

```
skewness(total_fer)
```

```
## [1] 0.9670917
```

```r
boxplot(total_fer, horizontal = T, col = "white", main = "Total Fertility")
```

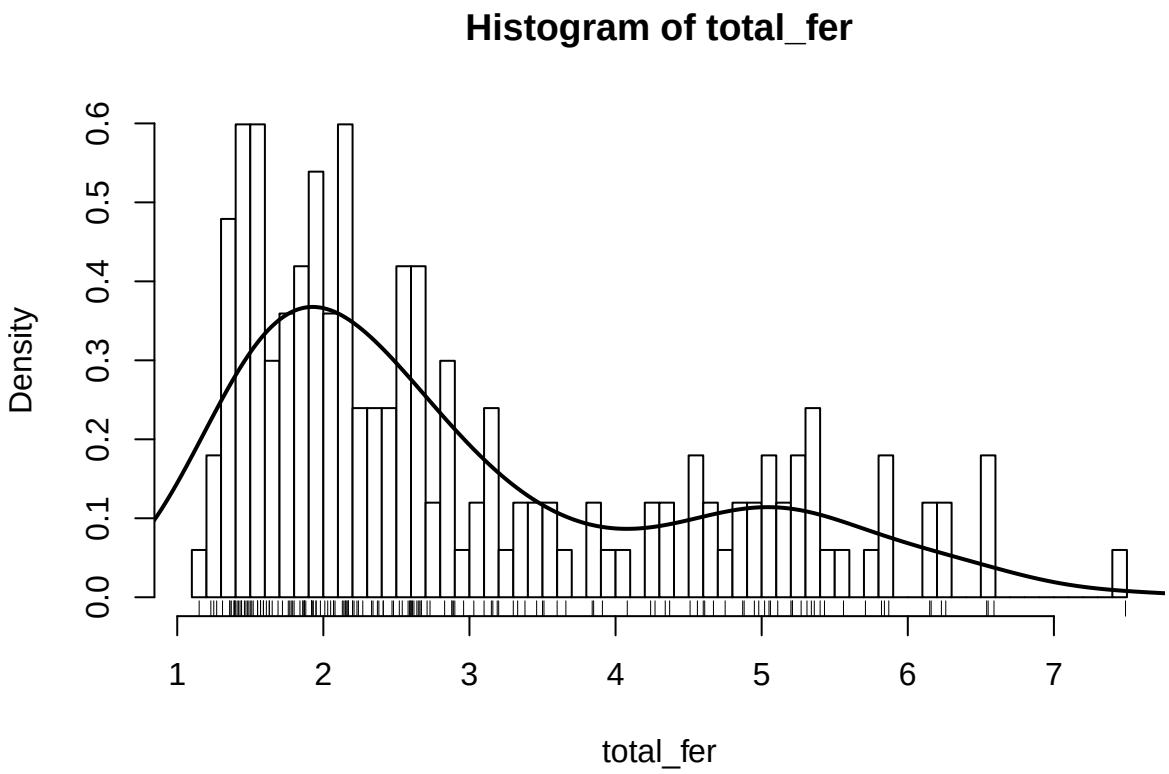



```r
# outliers
rownames(data[which(total_fer %in% boxplot(total_fer, plot = F)$out), ])
```

```
## [1] "Niger"
```

The `total_fer` variable ranges from 0.11 to 23.01, with a mean of 5.96. The distribution shows a slight positive skew (0.92), indicating a longer right tail. The interquartile range is between 1.99 and 9.80, suggesting that most countries have moderate fertility rates. Outliers include countries like St. Vincent and the Grenadines. The kurtosis is 0.24, indicating a relatively flat distribution compared to a normal distribution.

```r
hist(total_fer, col = "white", freq = F, breaks = 60)
lines(density(total_fer), col = "black", lwd = 2)
rug(total_fer)
```



The evident multimodal nature of the distribution suggests the presence of several distinct groups within the dataset.

```
mod1 <- fitDist(total_fer, data=data, k=2, type = "realplus")
```

Parametric Modelling

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
## Warning in MLE(ll4, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma, :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
mod2 <- fitDist(total_fer, data=data, k=log(dim(data)[1]), type = "realplus")
```

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
## Warning in MLE(ll2, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma), :
## possible convergence problem: optim gave code=1 false convergence (8)
```

```
## Warning in MLE(ll4, start = list(eta.mu = eta.mu, eta.sigma = eta.sigma, :
```

```
## possible convergence problem: optim gave code=1 false convergence (8)
```

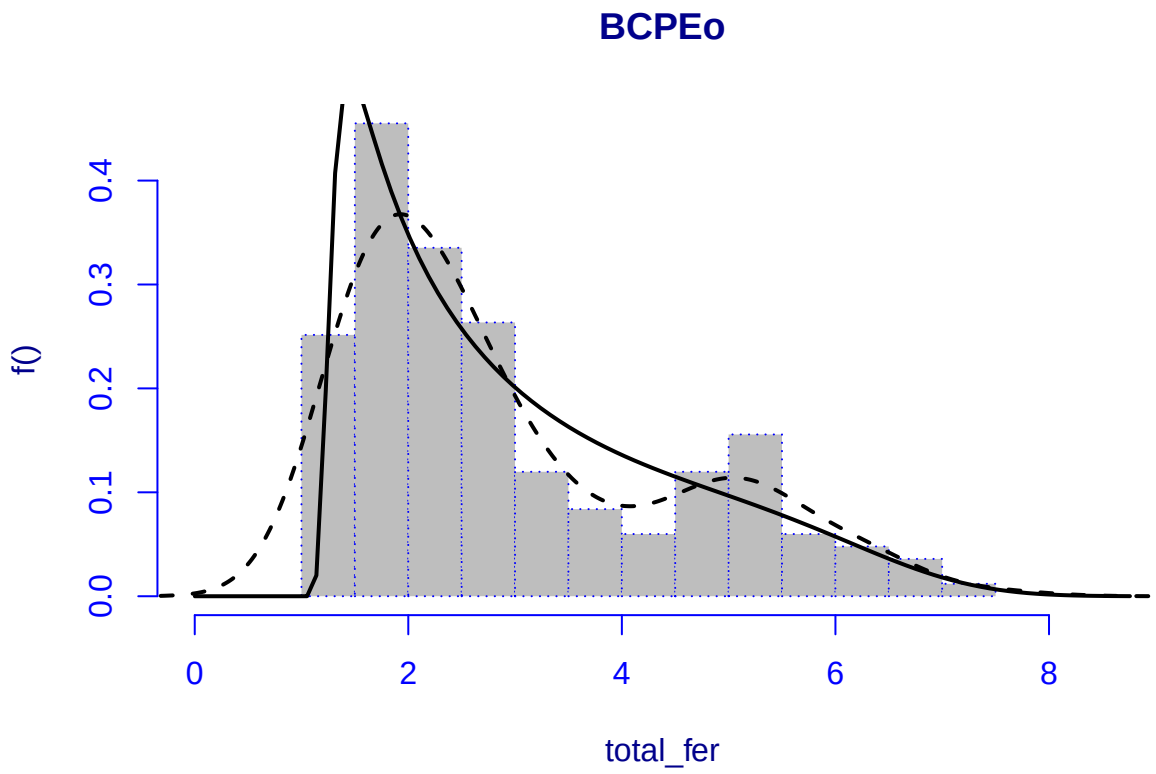
```
mod1$fits[1:6]
```

```
##     BCPE    BCPEo   exGAUS       GG   IGAMMA      GB2
## 515.9790 515.9790 526.1702 545.0110 546.1735 547.0118
```

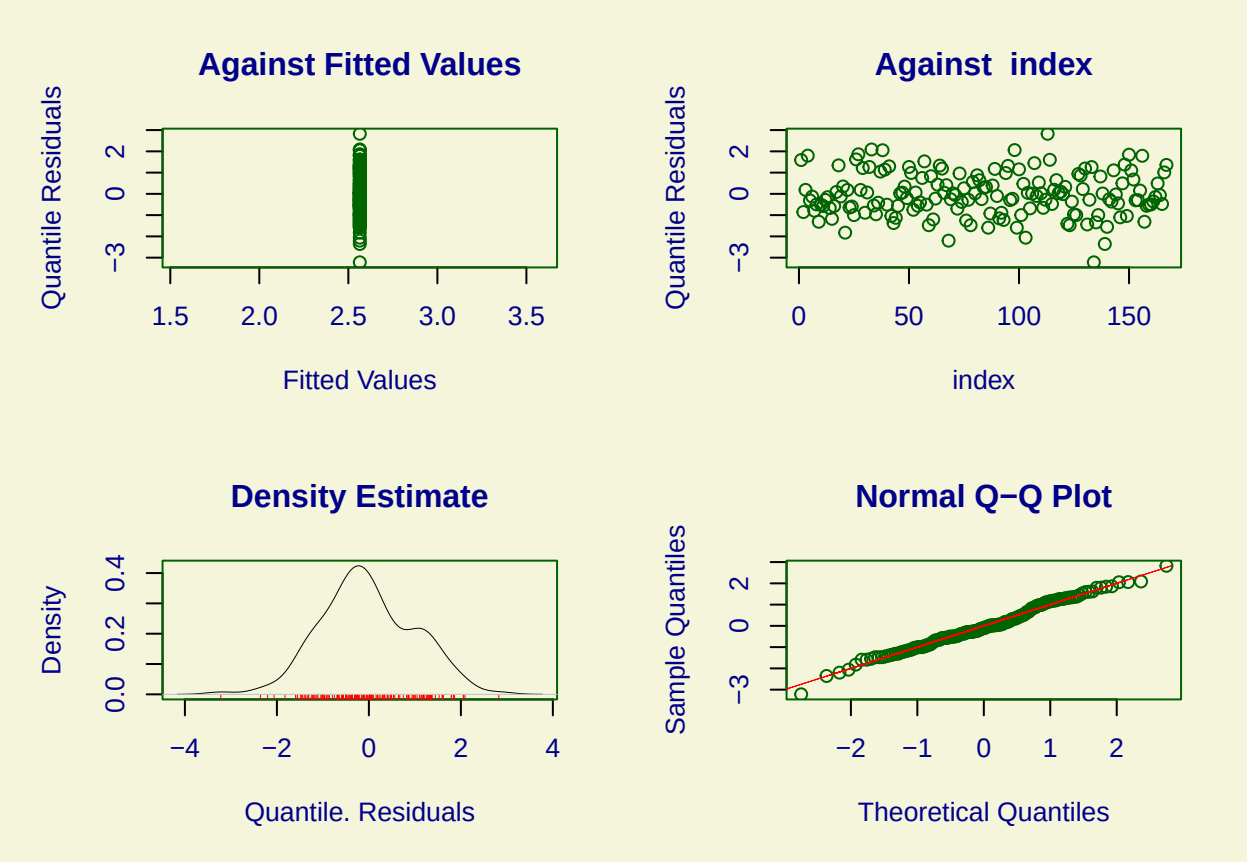
```
mod2$fits[1:6]
```

```
##     BCPE    BCPEo   exGAUS   IGAMMA       GG       IG
## 528.4510 528.4510 535.5242 552.4094 554.3650 555.8598
```

```
m1 <- histDist(total_fer, "BCPE" , density=TRUE, line.col=c(1,1), line.ty=c(1,2), breaks = 50,
               main = "BCPEo")
```




```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  -0.0263004
##                         variance   =  1.023432
##                coef. of skewness   =  0.07378096
##                coef. of kurtosis   =  2.947071
## Filliben correlation coefficient   =  0.9950264
## ****************************************************************
```

Residual Distribution of the Model

Model (Box-Cox Power Exponential)
- **Mean**: -0.013, **Variance**: 1.025
- **Skewness**: 0.020, **Kurtosis**: 3.02
- **Filliben**: 0.996, indicating that the residuals are nearly normal.

The **BCPE** model is the best fit for our data, as indicated by its superior AIC and BIC values. Diagnostic plots of the residuals reveal that they follow a standard normal distribution, further supporting the model's suitability for the data.

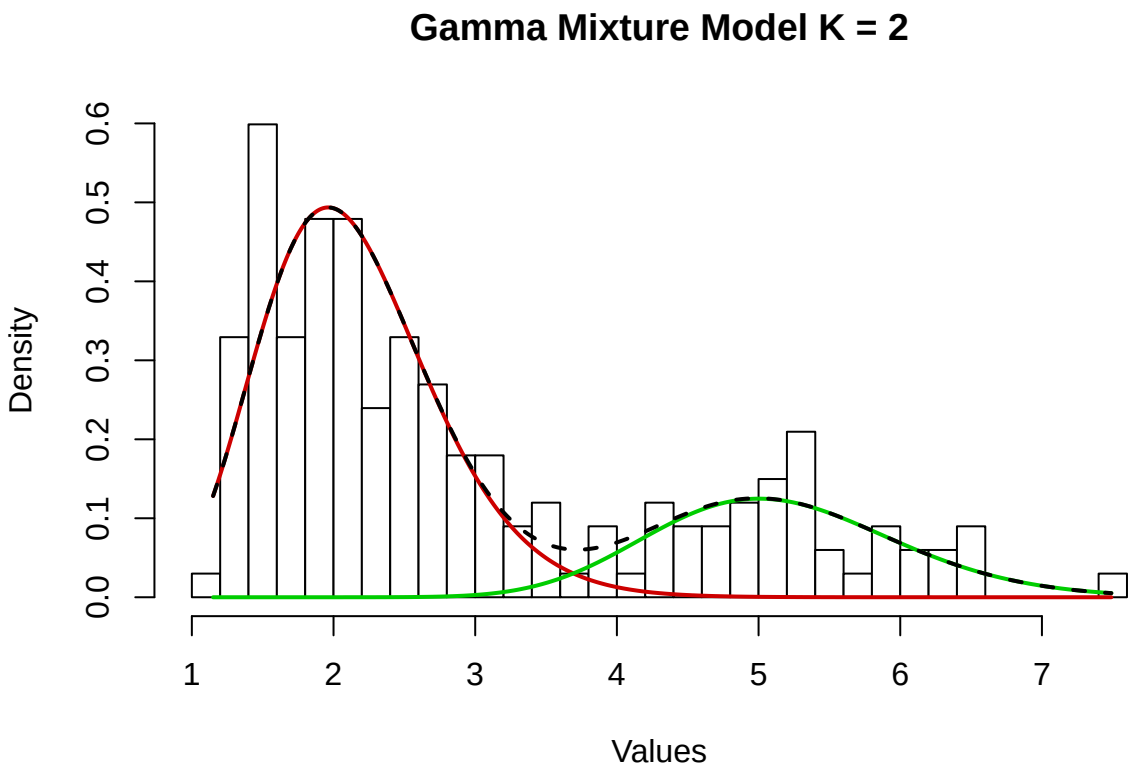
```
fit_mix <- mixfit(total_fer,
             family = "gamma",
             ncomp = 2,
             tol = 1e-06)
```

```
fit_mix
```

Gamma Mixture model

```
## Gamma mixture model with 2 components
##            comp1      comp2
## pi     0.7309393  0.2690607
## mu     2.1374353  5.1498739
## sd     0.6120145  0.8704681
## shape 12.1972656 35.0015625
## rate   5.7064959  6.7965863
##
## EM iterations: 15 AIC: 519.59 BIC: 535.18 log-likelihood: -254.79
```

```
############
##  Plot  ##
############

colors <- c("red3", "green3")

weights <- fit_mix$pi
mu <- fit_mix$mu
shape <- fit_mix$alpha
rate <- fit_mix$lambda

hist(total_fer, breaks = 30, probability = T, col = "white", border = "black",
     main = "Gamma Mixture Model K = 2",
     xlab = "Values", ylab = "Density")

x_vals <- seq(min(total_fer), max(total_fer), length.out = 500)

for (i in 1:length(weights)) {
  lines(x_vals, weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i]), col = colors[i], lwd = 2
}

overall_density <- rowSums(sapply(1:length(weights), function(i) {
  weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i])
}))

lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)
```




```
m1_g <- gamlssML(total_fer~1, data=data, family= BCPEo, trace=FALSE)
summary(m1_g)
```

Model Summary

```
## ******************************************************************
## Family:  c("BCPEo", "Box-Cox Power Exponential-orig.")
##
## Call:  gamlssML(formula = total_fer ~ 1, family = BCPEo, data = data,
##     trace = FALSE)
##
## Fitting method: "nlminb"
##
##
## Coefficient(s):
##             Estimate  Std. Error   t value Pr(>|t|)
## eta.mu     0.9417139   0.0463064  20.33657  < 2e-16 ***
## eta.sigma -0.7707523   0.0352989 -21.83502  < 2e-16 ***
## eta.nu    -0.3546753   0.1379053  -2.57188 0.010115 *
## eta.tau    2.2884978   0.2680580   8.53732  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   163
## Global Deviance:     507.979
```

```
##             AIC:     515.979
##             SBC:     528.451
```

Box-Cox Power Exponential-orig (BCPEo) Distribution

The **Box-Cox Power Exponential-orig (BCPEo)** distribution was applied to the total fertility data using **Maximum Likelihood Estimation (MLE)** with the **"nlminb"** method.

Parameter Estimates:

- $\eta_\mu = 0.942$ (p < 2e-16): Highly significant location parameter, indicating the central tendency of the distribution.
- $\eta_\sigma = -0.771$ (p < 2e-16): Highly significant scale parameter, showing strong dispersion.
- $\eta_\nu = -0.355$ (p = 0.010): Significant shape parameter, capturing the asymmetry of the distribution.
- $\eta_\tau = 2.288$ (p < 2e-16): Highly significant shape parameter, influencing the heavy tails.

Goodness-of-Fit:

- **Global Deviance**: 507.979
- **AIC**: 515.979
- **SBC**: 528.451

The **BCPEo** distribution provides a strong fit for the total fertility data, with all parameters being highly significant, effectively capturing the data's central tendency, dispersion, and asymmetry.

GDPP

The variable `GDPP` is numerical distribuited on the $\mathbb{R}^+$

```r
summary(gdpp)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     231    1330    4660   12964   14050  105000
```

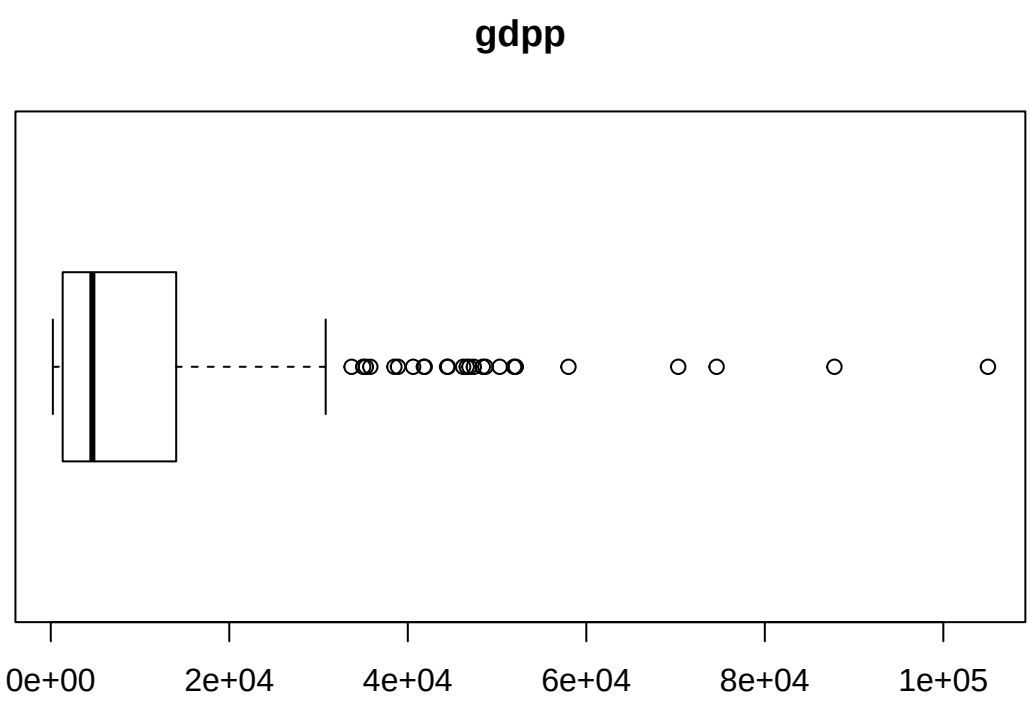
```r
kurtosis(gdpp, excess = T)
```

```
## [1] 5.527891
```

```r
skewness(gdpp)
```

```
## [1] 2.218051
```

```r
boxplot(gdpp, horizontal = T, col = "white", main = "gdpp")
```

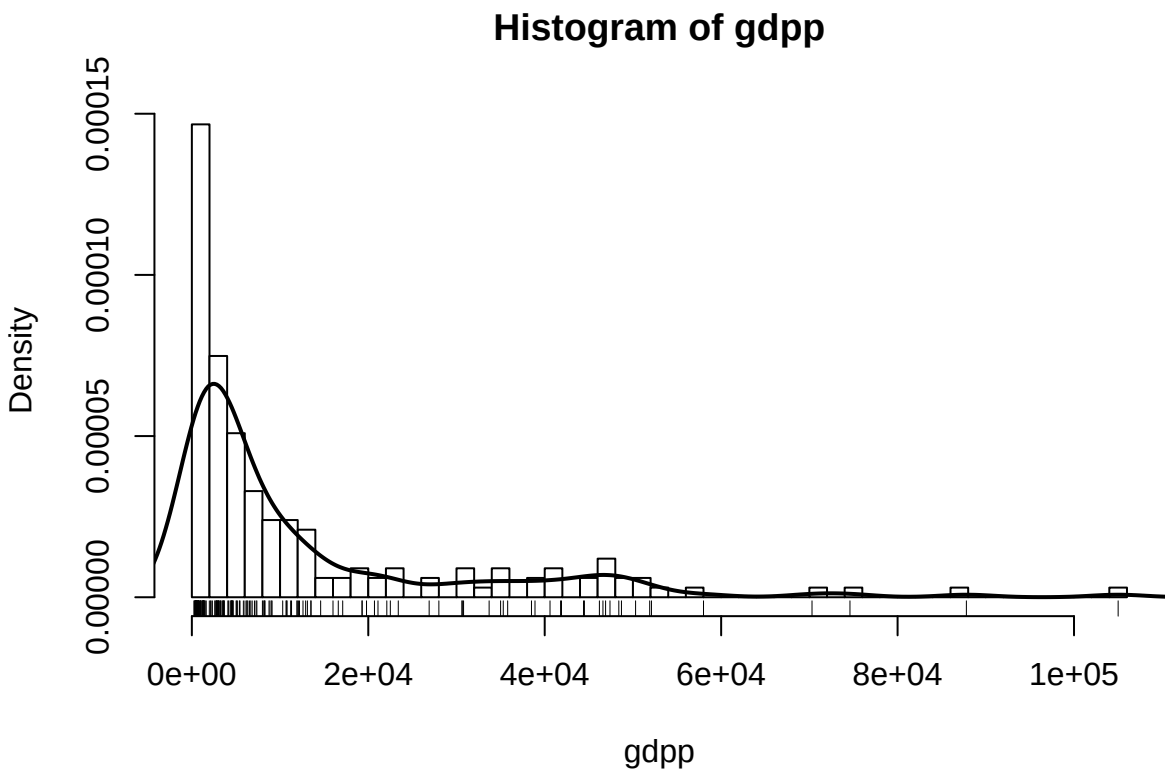



```
# outliers
rownames(data[which(gdpp %in% boxplot(gdpp, plot = F)$out), ])
```

```
##  [1] "Australia"            "Austria"              "Belgium"
##  [4] "Brunei"               "Canada"               "Denmark"
##  [7] "Finland"              "France"               "Germany"
## [10] "Iceland"              "Ireland"              "Italy"
## [13] "Japan"                "Kuwait"               "Luxembourg"
## [16] "Netherlands"          "New Zealand"          "Norway"
## [19] "Qatar"                "Singapore"            "Sweden"
## [22] "Switzerland"          "United Arab Emirates" "United Kingdom"
## [25] "United States"
```

We can observe that this variable contains several outliers, which correspond to the wealthier countries.

```
hist(gdpp, col = "white", freq = F, breaks = 60)
lines(density(gdpp), col = "black", lwd = 2)
rug(gdpp)
```




```
mod1 <- fitDist(gdpp, data=data, k=2, type = "realplus")
mod2 <- fitDist(gdpp, data=data, k=log(dim(data)[1]), type = "realplus")
```

```
mod1$fits[1:6]
```

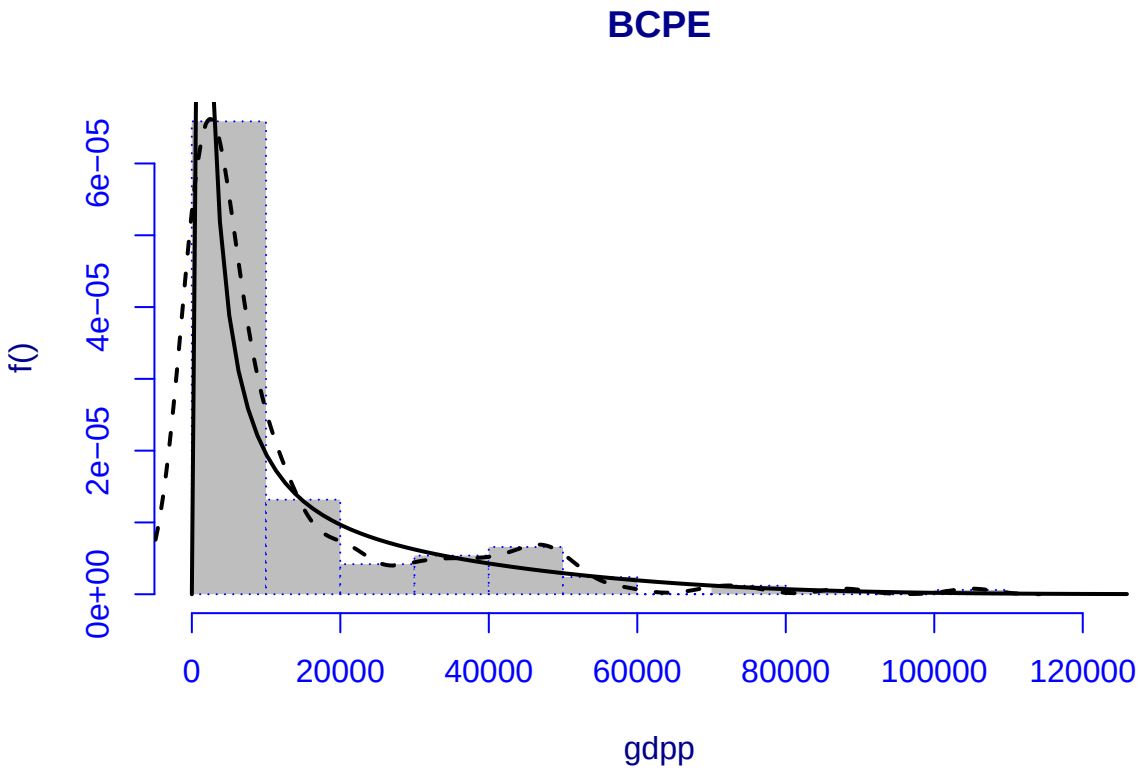
Parametric Modelling

```
##     BCPE    BCPEo      GIG    LOGNO   LOGNO2       IG
## 3432.675 3432.675 3439.412 3451.228 3451.228 3451.931
```

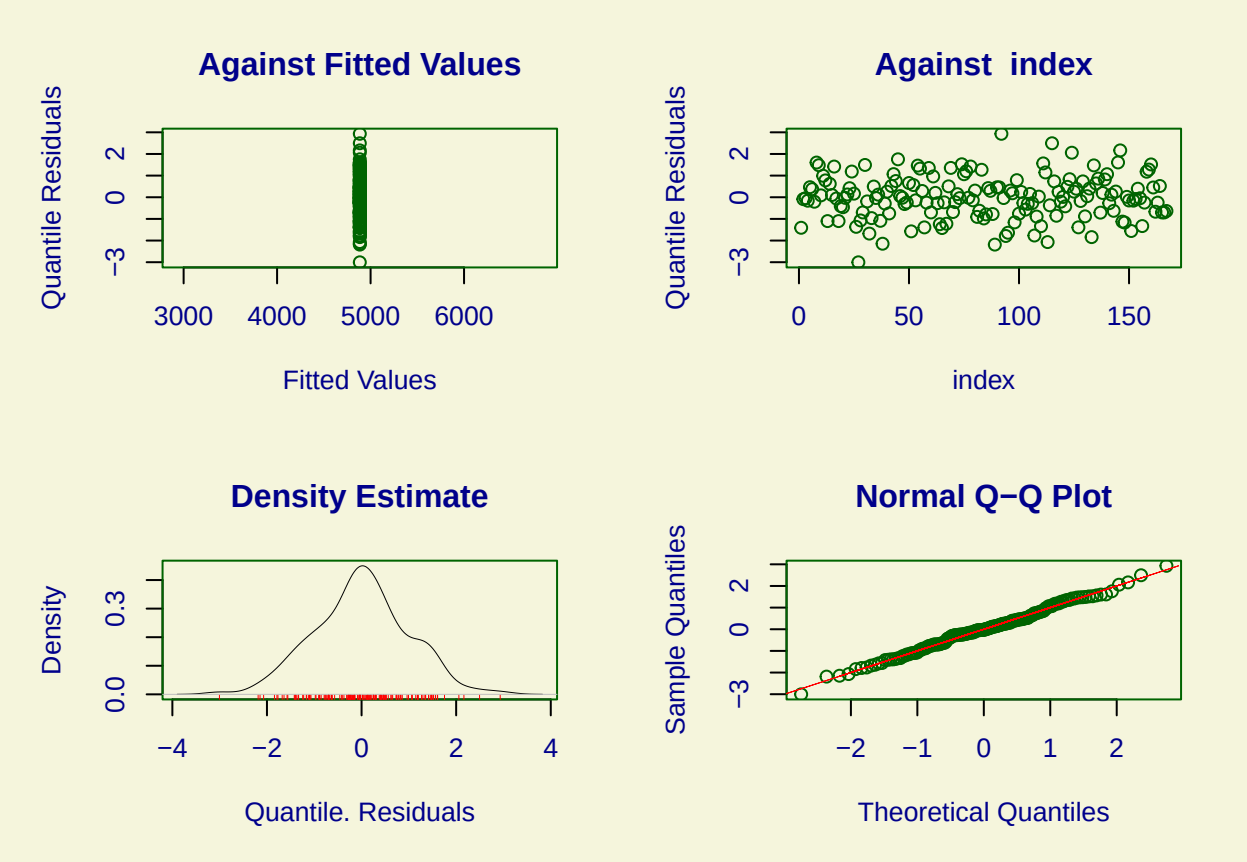
```
mod2$fits[1:6]
```

```
##     BCPE    BCPEo      GIG    LOGNO   LOGNO2       IG
## 3445.147 3445.147 3448.766 3457.464 3457.464 3458.167
```

```
m1 <- histDist(gdpp, "BCPE" , density=TRUE, line.col=c(1,1), line.ty=c(1,2), breaks = 50,
               main = "BCPE")
```


BCPE
f()
0e+00
2e−05
4e−05
6e−05
0
20000
40000
60000
80000
100000
120000
gdpp

```
plot(m1)
```




```
## ****************************************************************
##        Summary of the Quantile Residuals
##                             mean   =  -6.92245e-06
##                         variance   =  0.9875769
##                 coef. of skewness  =  -0.001276188
##                 coef. of kurtosis  =  3.091782
## Filliben correlation coefficient   =  0.9970613
## ****************************************************************
```

Residual Distribution of the Model

Model (Box-Cox Power Exponential)
- **Mean**: -6.92245e-06, **Variance**: 0.98
- **Skewness**: -0.001, **Kurtosis**: 3.09
- **Filliben**: 0.997, indicating that the residuals are nearly normal.

The **BCPE** model is the best fit for our data, as indicated by its superior AIC and BIC values. Diagnostic plots of the residuals reveal that they follow a standard normal distribution, further supporting the model's suitability for the data.

Gamma Mixture model $\mathbf{K = 2}$

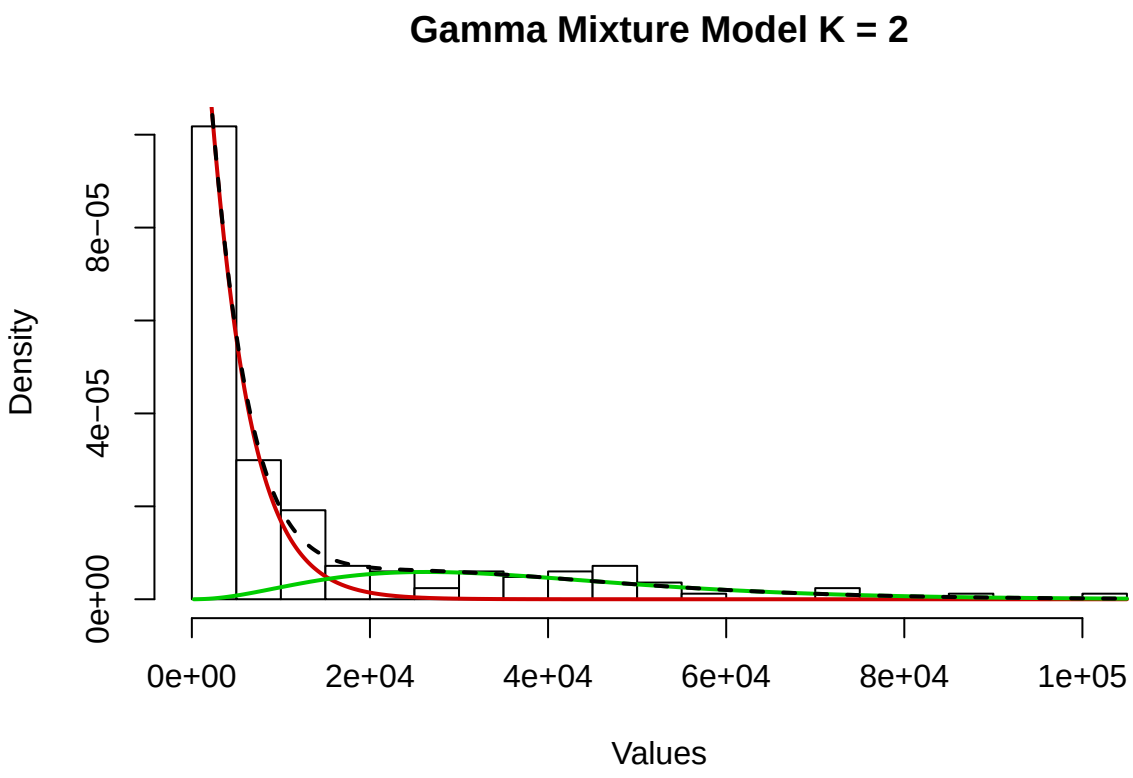
```
fit_mix <- mixfit(gdpp,
             family = "gamma",
             ncomp = 2,
             tol = 1e-06)
```

```
fit_mix
```

```
## Gamma mixture model with 2 components
##              comp1        comp2
## pi       0.7387337 2.612663e-01
## mu    4291.4494931 3.748635e+04
## sd    4123.0603345 2.046905e+04
## shape    1.0833496 3.353906e+00
## rate     0.0002524 8.950000e-05
##
## EM iterations: 96 AIC: 3450.57 BIC: 3466.16 log-likelihood: -1720.28
```

fit_mix <- mixfit(gdpp, family = "gamma", ncomp = 2, tol = 1e-06) fit_mix

```
############
##  Plot  ##
############

colors <- c("red3", "green3")

weights <- fit_mix$pi
mu <- fit_mix$mu
shape <- fit_mix$alpha
rate <- fit_mix$lambda

hist(gdpp, breaks = 30, probability = T, col = "white", border = "black",
     main = "Gamma Mixture Model K = 2",
     xlab = "Values", ylab = "Density")

x_vals <- seq(min(gdpp), max(gdpp), length.out = 500)

for (i in 1:length(weights)) {
  lines(x_vals, weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i]), col = colors[i], lwd = 2
}

overall_density <- rowSums(sapply(1:length(weights), function(i) {
  weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i])
}))

lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)
```



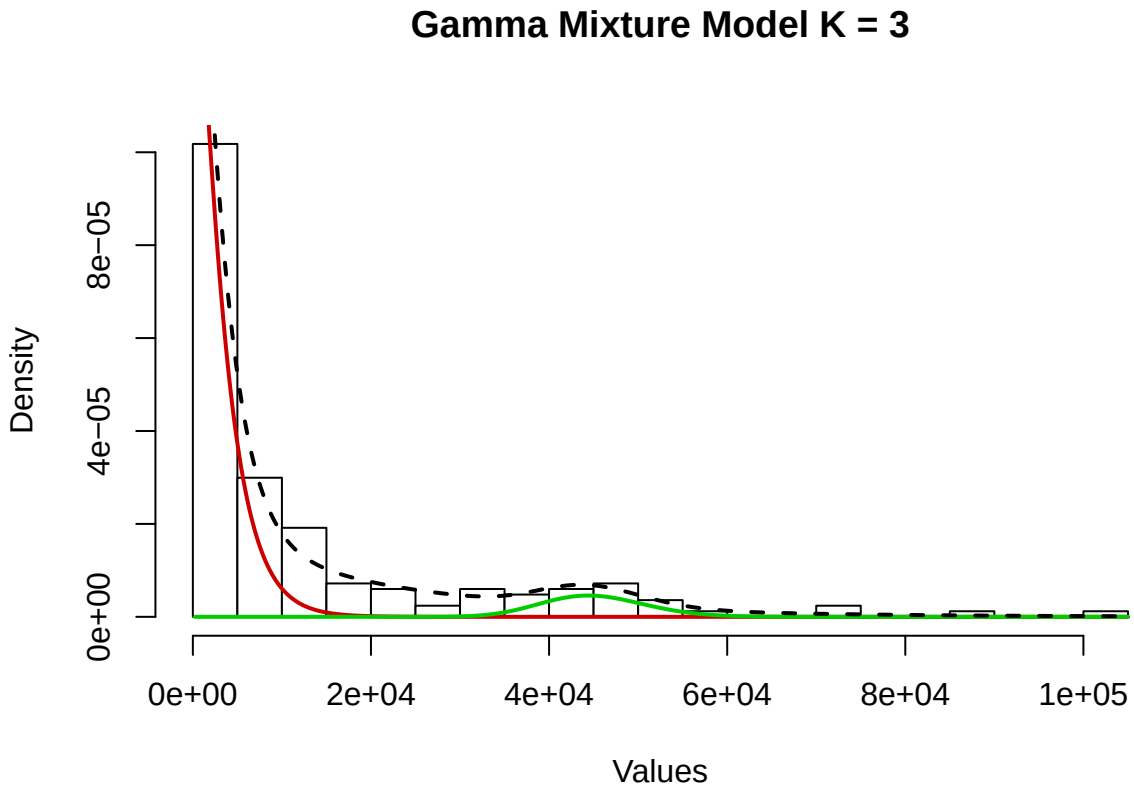
K = 3

```
fit_mix <- mixfit(gdpp,
             family = "gamma",
             ncomp = 3,
             tol = 1e-06)
fit_mix
```

```
## Gamma mixture model with 3 components
##               comp1        comp2        comp3
## pi        0.5356947 6.641860e-02 3.978868e-01
## mu     3093.9163195 4.515298e+04 2.087972e+04
## sd     2804.1658114 5.829149e+03 2.073544e+04
## shape     1.2173340 6.000156e+01 1.013965e+00
## rate      0.0003935 1.328900e-03 4.860000e-05
##
## EM iterations: 300 AIC: 3450.05 BIC: 3474.99 log-likelihood: -1717.02
```

```
#############
##  Plot  ##
#############

colors <- c("red3", "green3")

weights <- fit_mix$pi
mu <- fit_mix$mu
shape <- fit_mix$alpha
```

```r
rate <- fit_mix$lambda

hist(gdpp, breaks = 30, probability = T, col = "white", border = "black",
     main = "Gamma Mixture Model K = 3
     ",
     xlab = "Values", ylab = "Density")

x_vals <- seq(min(gdpp), max(gdpp), length.out = 500)

for (i in 1:length(weights)) {
  lines(x_vals, weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i]), col = colors[i], lwd = 2
}

overall_density <- rowSums(sapply(1:length(weights), function(i) {
  weights[i] * dgamma(x_vals, shape = shape[i], rate = rate[i])
}))

lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)
```



The Gamma mixture with K = 3 is not optimal, likely due to its increased complexity.

```r
m1_g <- gamlssML(gdpp~1, data=data, family= BCPE, trace=FALSE)
summary(m1_g)
```

Model Summary

```
## ******************************************************************
## Family:  c("BCPE", "Box-Cox Power Exponential")
##
## Call:  gamlssML(formula = gdpp ~ 1, family = BCPE, data = data, trace = FALSE)
##
##
## Fitting method: "nlminb"
##
##
## Coefficient(s):
##               Estimate   Std. Error  t value   Pr(>|t|)
## eta.mu     4.88477e+03  7.20840e+02  6.77650 1.2313e-11 ***
## eta.sigma  4.18475e-01  3.31639e-02 12.61841 < 2.22e-16 ***
## eta.nu    -4.63372e-03  4.10988e-02 -0.11275    0.91023
## eta.tau    2.09287e+00  2.78657e-01  7.51056 5.8842e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Degrees of Freedom for the fit: 4 Residual Deg. of Freedom   163
## Global Deviance:     3424.67
##             AIC:     3432.67
##             SBC:     3445.15
```

Box-Cox Power Exponential (BCPE) Distribution

The Box-Cox Power Exponential (BCPE) distribution was applied to the GDP data using Maximum Likelihood Estimation (MLE).

Parameter Estimates:

- $\eta_\mu = 4884.77$ (p < 1.23e-11): Highly significant location parameter, indicating the central tendency of the distribution.
- $\eta_\sigma = 0.418$ (p < 2.22e-16): Highly significant scale parameter, reflecting strong dispersion in the data.
- $\eta_\nu = -0.0046$ (p = 0.910): Insignificant shape parameter, indicating no clear asymmetry in the distribution.
- $\eta_\tau = 2.092$ (p < 5.88e-14): Highly significant shape parameter, influencing the heavy tails of the distribution.

Goodness-of-Fit:

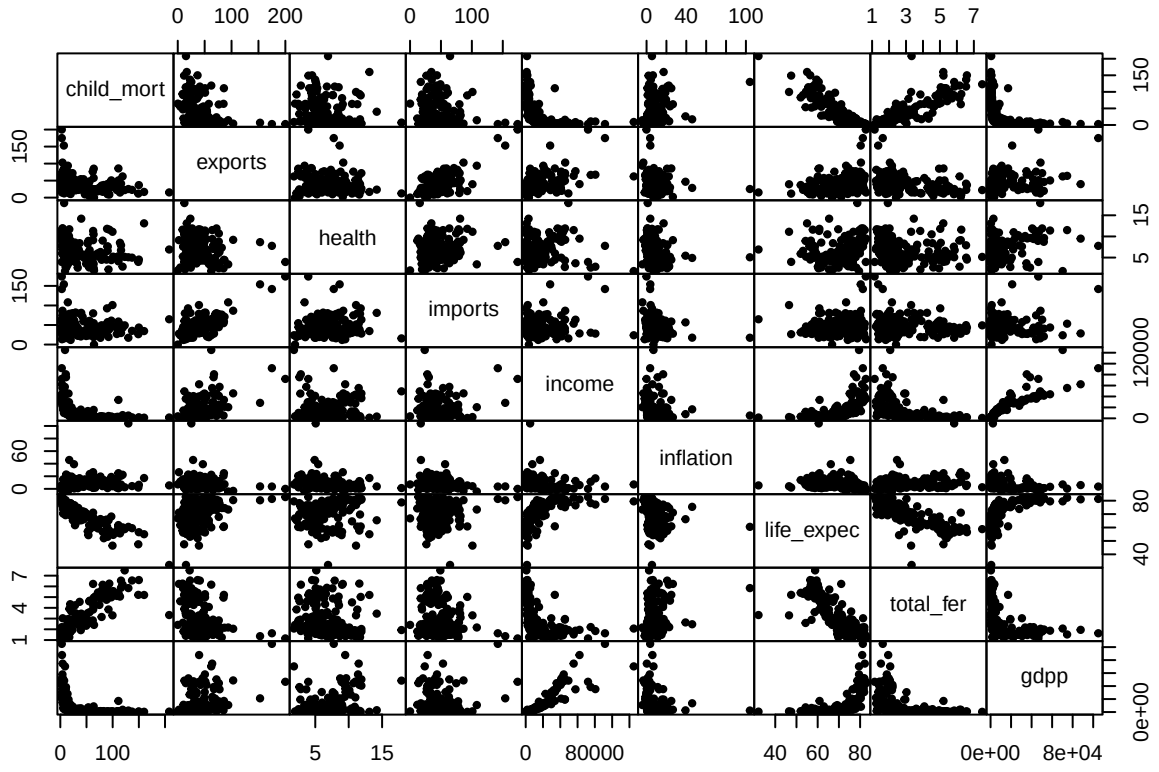
- **Global Deviance**: 3424.67
- **AIC**: 3432.67
- **SBC**: 3445.15

The BCPE distribution provides a good fit for the GDP data, with all parameters except for the shape parameter η_ν being highly significant. This model effectively captures the central tendency, dispersion, and the heavy tails of the data.

Multivariate Analysis

Pairplot

```r
pairs(data, gap = 0, pch = 20)
```



The pairplot highlights potential relationships between the different variables, showing negative correlations, such as between child mortality and GDP or life expectancy, and positive trends, such as between GDP and income or imports and exports. Overall, the plot suggests the presence of patterns among these variables that could be used to group observations based on socio-economic indicators.

Principal Component Analysis

In general, when dealing with dimensionality reduction, we have two primary approaches:

- **Feature Selection**: In this approach, we select only the most significant variables in terms of the total variance explained by the data. The advantage of this technique is that it preserves the interpretability of the data. However, it may result in the loss of more information compared to feature extraction.
- **Feature Extraction**: In this case, we aim to extract information from the data by creating a new feature space. The trade-off here is that, while interpretability is reduced, more information is preserved through the variability in the data.

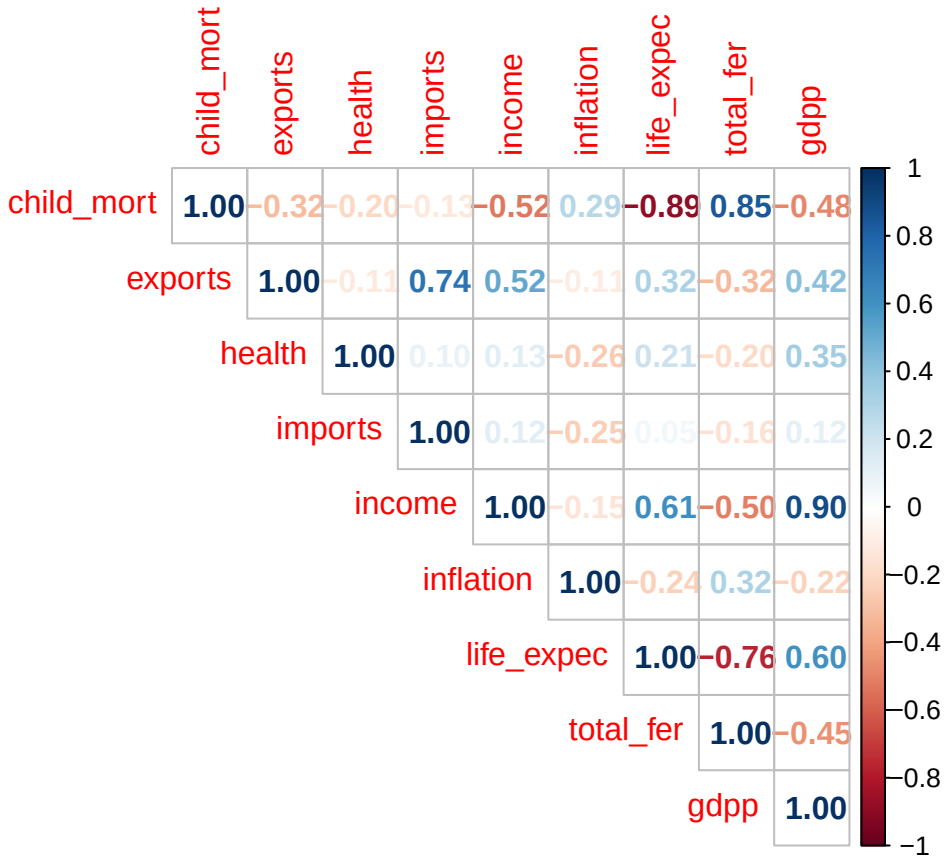
Principal Component Analysis (PCA) is an Feature Extraction technique used for dimensionality reduction. This technique involves constructing a new feature space in which each dimension is a linear combination of the original dimensions. Specifically, if X_i represents the original data and Y_i represents the first principal component (PC1), we have:

$$Y_1 = \Phi_{11}\tilde{X_1} + \Phi_{21}\tilde{X_2} + \cdots + \Phi_{d1}\tilde{X_d} = \sum_{j=1}^{d} \Phi_{j1}\tilde{X_j}$$

In whitch Φ represent the loadings. $\sum_{j=1}^{d} \Phi_{j1}^2 = 1$

Due to its construction properties, PCA works particularly well in situations where the data needs to be "summarized." In cases of high correlation, the variance explained by the first principal component is typically very high, allowing us to apply our model to this new dimension with minimal loss of information. To assess whether applying PCA is appropriate for the dataset, I calculated the correlation matrix between the variables. This helps evaluate the underlying structure of the data and the potential relationships among variables.

```
cor_mat <- cor(scaled_data)
corrplot(cor_mat, method = "number", type = "upper")
```



The data appears to exhibit notable correlations; for instance, GDP per capita is highly positively correlated with income, while life expectancy shows a strong negative correlation with child mortality. Given this correlation structure, it is reasonable to proceed with applying PCA.

For the computation I used the package `factoextra`.

```
res_pca <- PCA(data, scale.unit = T, graph = F)
res_pca$eig
```

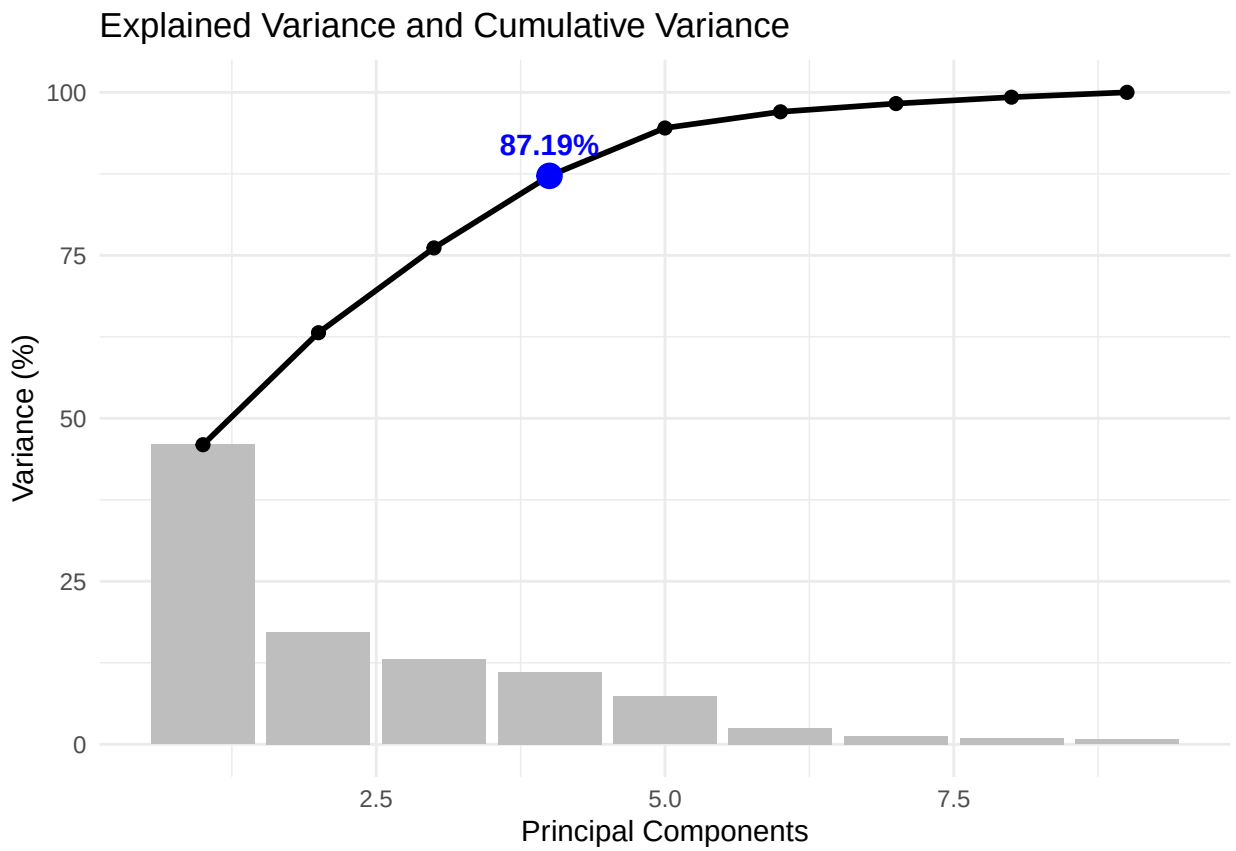
```
##        eigenvalue percentage of variance cumulative percentage of variance
## comp 1 4.13565658             45.9517398                           45.95174
## comp 2 1.54634631             17.1816257                           63.13337
## comp 3 1.17038330             13.0042589                           76.13762
## comp 4 0.99478456             11.0531618                           87.19079
## comp 5 0.66061903              7.3402114                           94.53100
## comp 6 0.22358112              2.4842347                           97.01523
## comp 7 0.11343874              1.2604304                           98.27566
## comp 8 0.08831536              0.9812817                           99.25694
```

```
## comp 9 0.06687501              0.7430556                       100.00000
```

From this computation, I obtained the following table as output. The first column represents the eigenvalues, which indicate the amount of variance explained by each Principal Component (PC).

```
explained_variance <- res_pca$eig[,1] / sum(res_pca$eig[,1]) * 100

variance_table <- data.frame(
  PC = 1:length(explained_variance),
  ExplainedVariance = explained_variance,
  CumulativeVariance = res_pca$eig[, 3]
)

ggplot(variance_table, aes(x = PC)) +
  geom_bar(aes(y = ExplainedVariance), stat = "identity", fill = "grey") +
  geom_line(aes(y = CumulativeVariance), group = 1, color = "black", size = 1) +
  geom_point(aes(y = CumulativeVariance), color = "black", size = 2) +
  geom_point(data = variance_table[4, ], aes(x = PC, y = CumulativeVariance), color = "blue", size = 4)
  geom_text(data = variance_table[4, ], aes(x = PC, y = CumulativeVariance, label = paste0("87.19%")),
            vjust = -1, color = "blue", fontface = "bold") +
  labs(title = "Explained Variance and Cumulative Variance",
       x = "Principal Components",
       y = "Variance (%)") +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



In this plot, we observe that the threshold of 80% explained variance is reached by including the first 4 principal components (PCs).

PCA var

```
var <- get_pca_var(res_pca)

# Cos2: quality on the factor map
var$cos2
```

```
##                Dim.1        Dim.2       Dim.3        Dim.4       Dim.5
## child_mort 0.72786136 0.0575305975 0.001021534 1.366673e-01 0.018861171
## exports    0.33332356 0.5813790019 0.024526152 9.504568e-06 0.002192981
## health     0.09409466 0.0913754315 0.416621553 2.122366e-01 0.177260193
## imports    0.10784378 0.6979325480 0.105283061 5.143716e-03 0.043083667
## income     0.65655748 0.0007853122 0.106424002 1.529866e-01 0.040352543
## inflation  0.15432525 0.0001092264 0.483170891 2.251468e-02 0.337601319
## life_expec 0.74995658 0.0766961382 0.015188572 4.131670e-02 0.007736856
## total_fer  0.67409990 0.0372627974 0.000447289 1.423672e-01 0.012086597
## gdpp       0.63759400 0.0032752557 0.017700248 2.815422e-01 0.021443701
```

```
# Contributions to the principal components
var$contrib
```

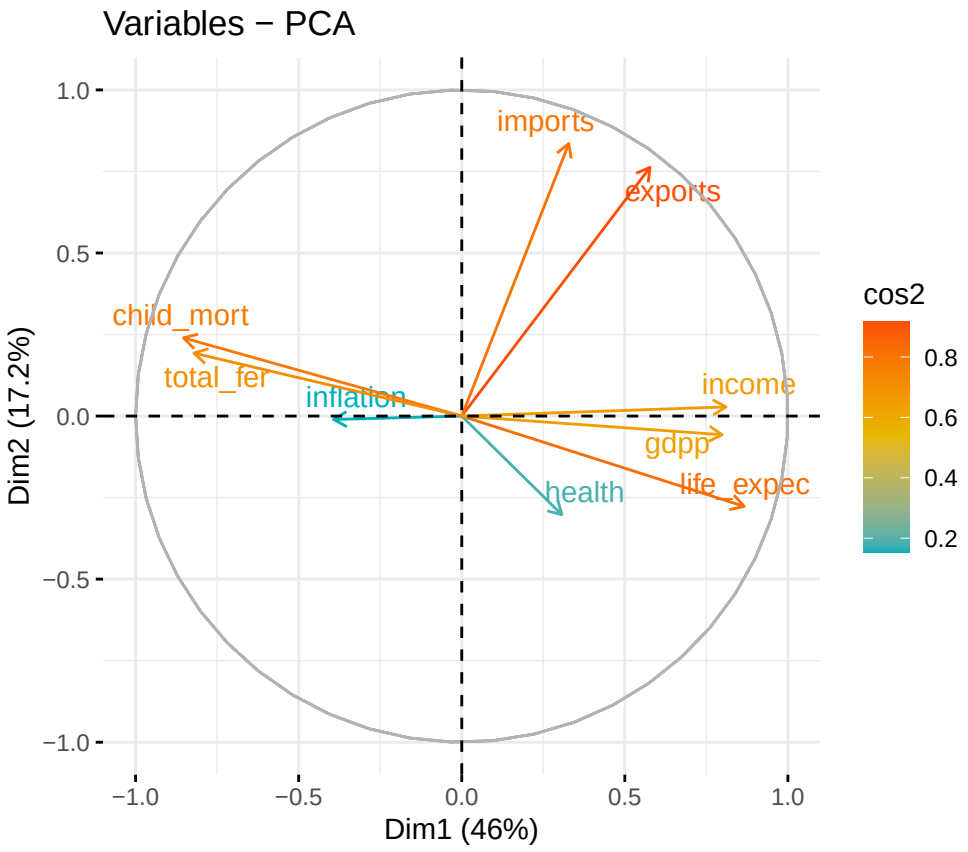
```
##                Dim.1        Dim.2       Dim.3        Dim.4      Dim.5
## child_mort 17.599657  3.720421304  0.08728199 1.373838e+01  2.8550754
## exports     8.059750 37.596946987  2.09556575 9.554399e-04  0.3319585
```

```
## health      2.275205  5.909118219 35.59701792 2.133493e+01 26.8324382
## imports     2.607658 45.134297800  8.99560518 5.170683e-01  6.5217114
## income     15.875532  0.050785012  9.09308954 1.537887e+01  6.1082926
## inflation   3.731578  0.007063517 41.28313265 2.263272e+00 51.1037837
## life_expec 18.133918  4.959829356  1.29774340 4.153331e+00  1.1711525
## total_fer  16.299707  2.409731711  0.03821731 1.431136e+01  1.8295866
## gdpp       15.416996  0.211806093  1.51234626 2.830182e+01  3.2460012
```

The `cos2` value represents the quality of representation of a variable on a specific principal component. It is calculated as the squared cosine of the angle between the original variable's vector and the principal component's vector, indicating how much of the variable's variance is explained by that component.

In contrast, the `contrib` measures the extent to which each variable contributes to the construction of a principal component. This contribution is determined by the variable's loading on the component and its relative importance in explaining the total variance of the component.

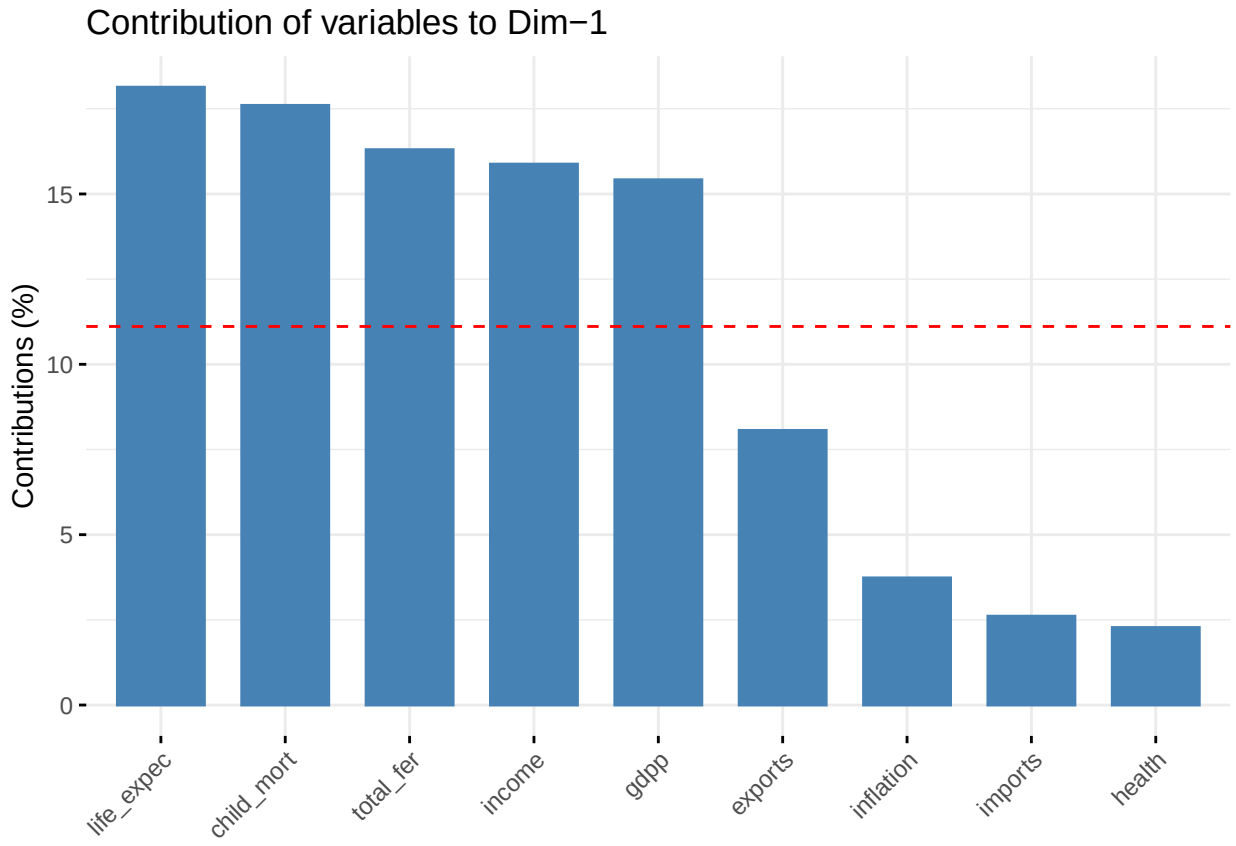
```
fviz_pca_var(res_pca, col.var = "cos2",
gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
repel = TRUE
)
```



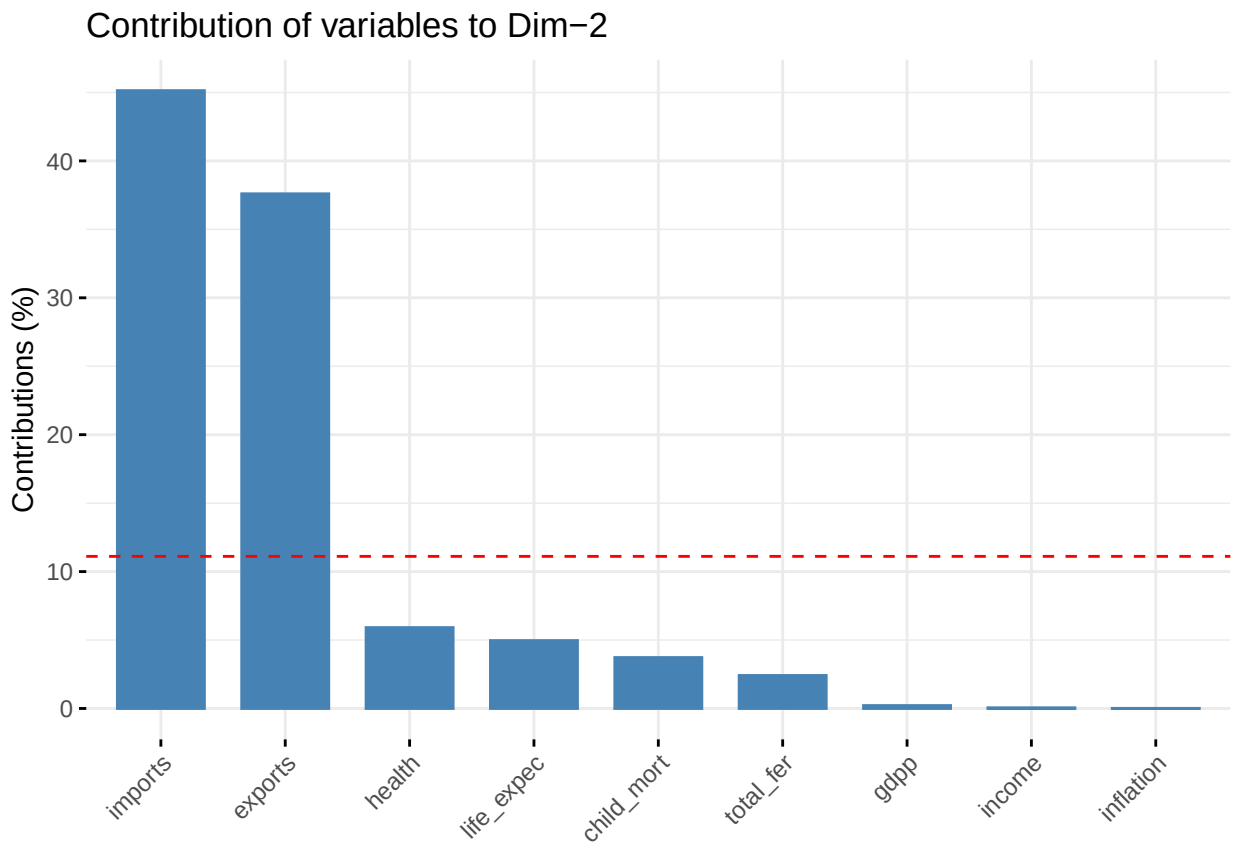
Through this plot, we can see that most variables contribute significantly to PC1, particularly economic indicators like income, GDP, and life expectancy. In contrast, variables like imports and exports have a stronger representation on PC2. The plot also highlights strong correlations, such as the positive link between GDP and income and the negative association between life expectancy and child mortality. This could suggests that PC1 captures *economic factors*, while PC2 reflects *health and demographic patterns*.

Variables contributions to PC1 & PC2

```
fviz_contrib(res_pca, choice = "var", axes = 1, top = 10)
```



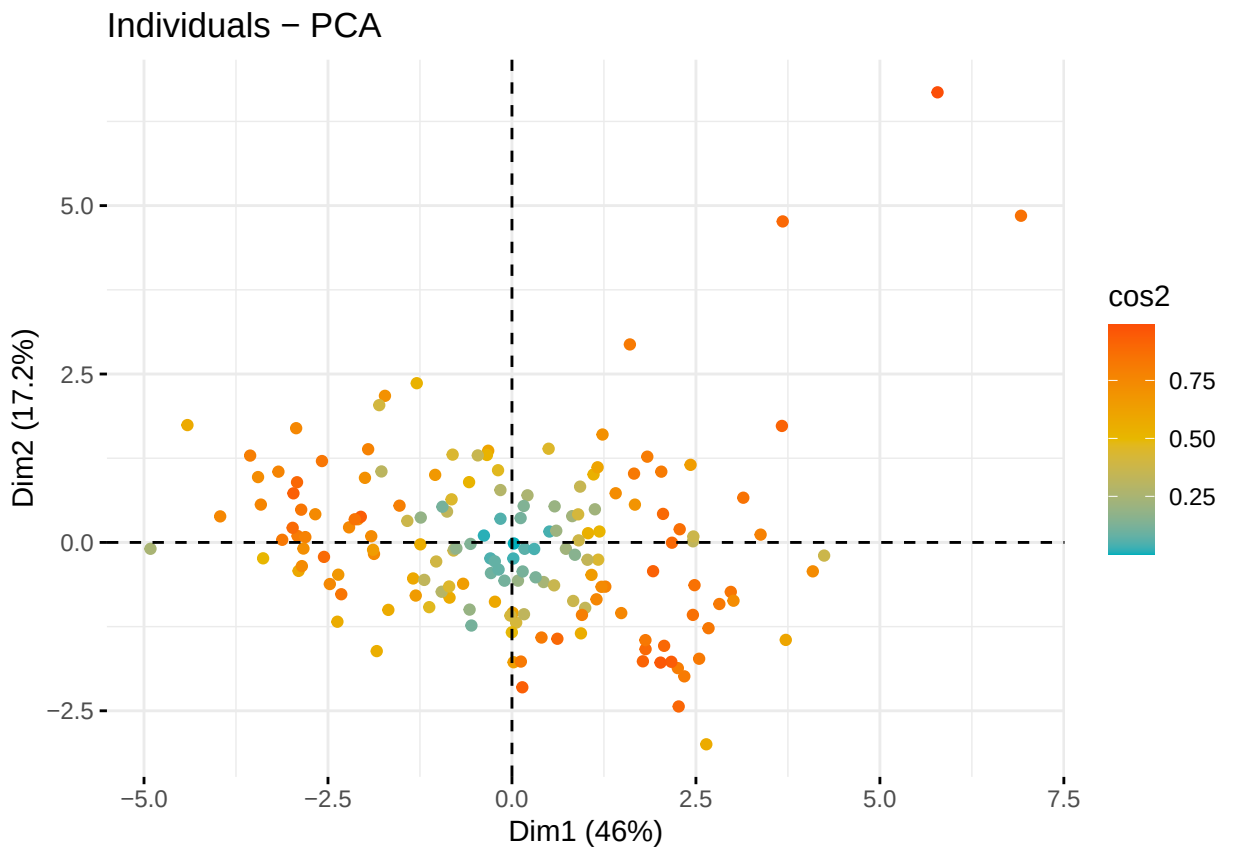

```
fviz_contrib(res_pca, choice = "var", axes = 2, top = 10)
```



This plot provides a clearer illustration that the second principal component is primarily influenced by the contributions of `imports` and `exports`.

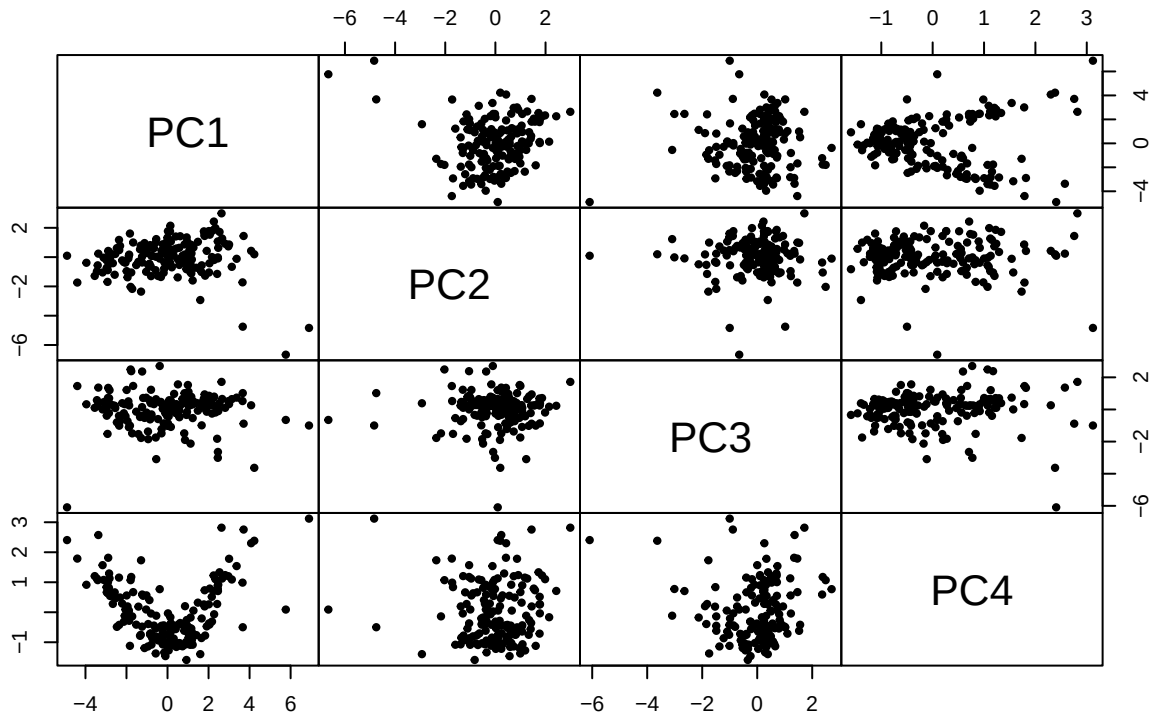
Plot quality and contribution

```
fviz_pca_ind(res_pca, col.ind = "cos2",
gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
repel = TRUE,
label = "none")
```



Dim1 (46% variance) primarily reflects socio-economic and demographic factors, driven by variables like GDP, income, life expectancy, and child mortality. Observations with high Dim1 values are likely more developed economies, while lower values indicate less developed ones. Dim2 (17.2% variance), on the other hand, is mainly influenced by import and export variables, capturing variations related to international trade. The color gradient shows how well each observation is represented by these two dimensions, with red/orange points being better represented than blue/green ones (witch are better explained by other PCs). This plot highlights clear patterns based on development and trade characteristics.

```
pca_result <- prcomp(data, scale = TRUE)
pca_data <- pca_result$x[, 1:4]
pairs(pca_data, pch = 20, gap = 0,
      main = "Pair Plot of the First 4 Principal Components")
```



Clustering analysis

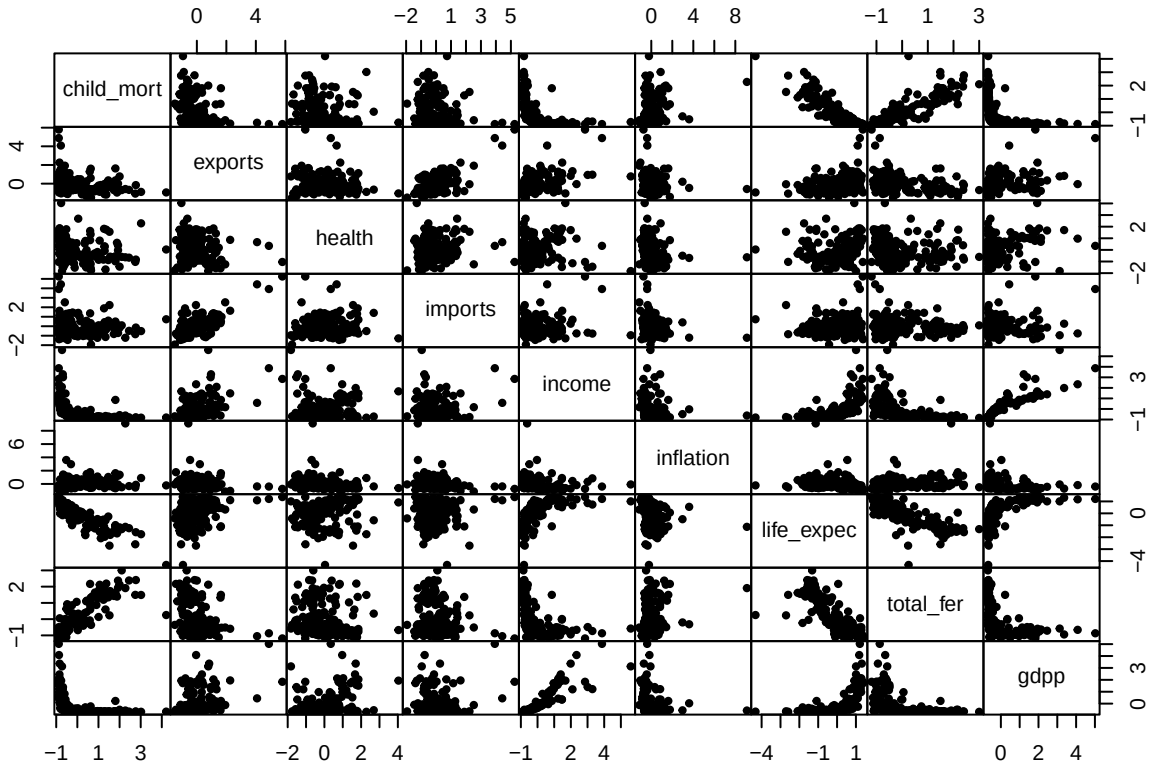
First, we need to determine whether the data should be clustered. To do this, we must assess the cluster tendency, which involves evaluating whether meaningful clusters are present in the data.

```
random_df <- apply(data, 2,
                   function(x){runif(length(x), min(x), max(x))})

random_df <- as.data.frame(random_df)
```

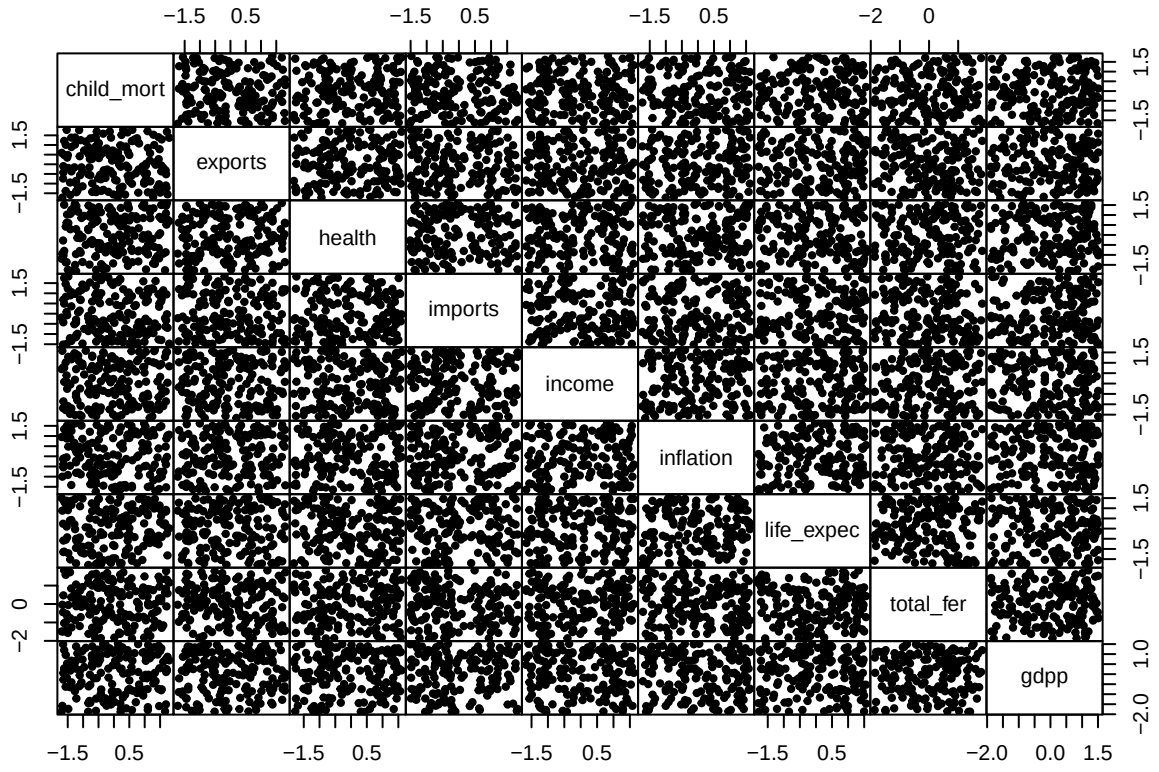
Cluster Validation

Preliminary Visualization

```
pairs(scale(data), gap = 0, pch = 20)
```



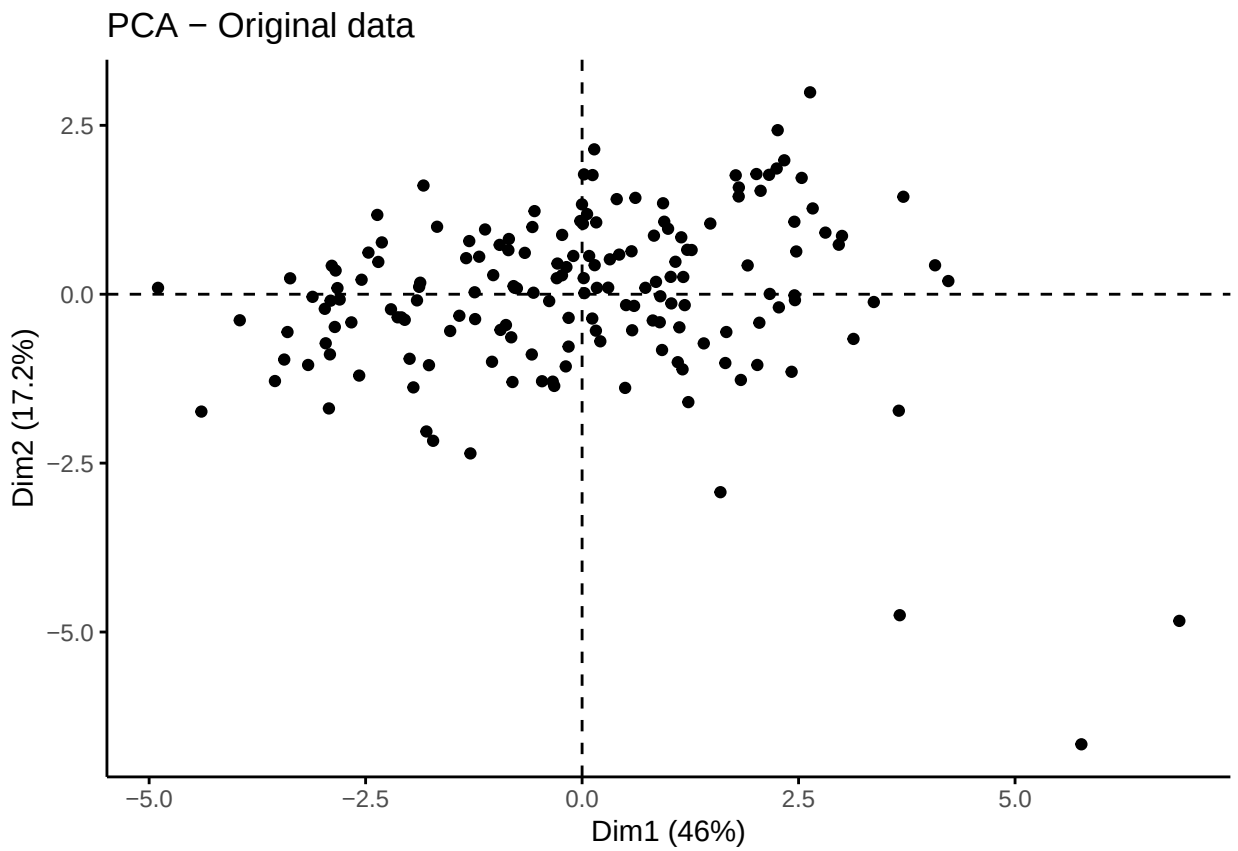

```
pairs(scale(random_df), gap = 0, pch = 20)
```



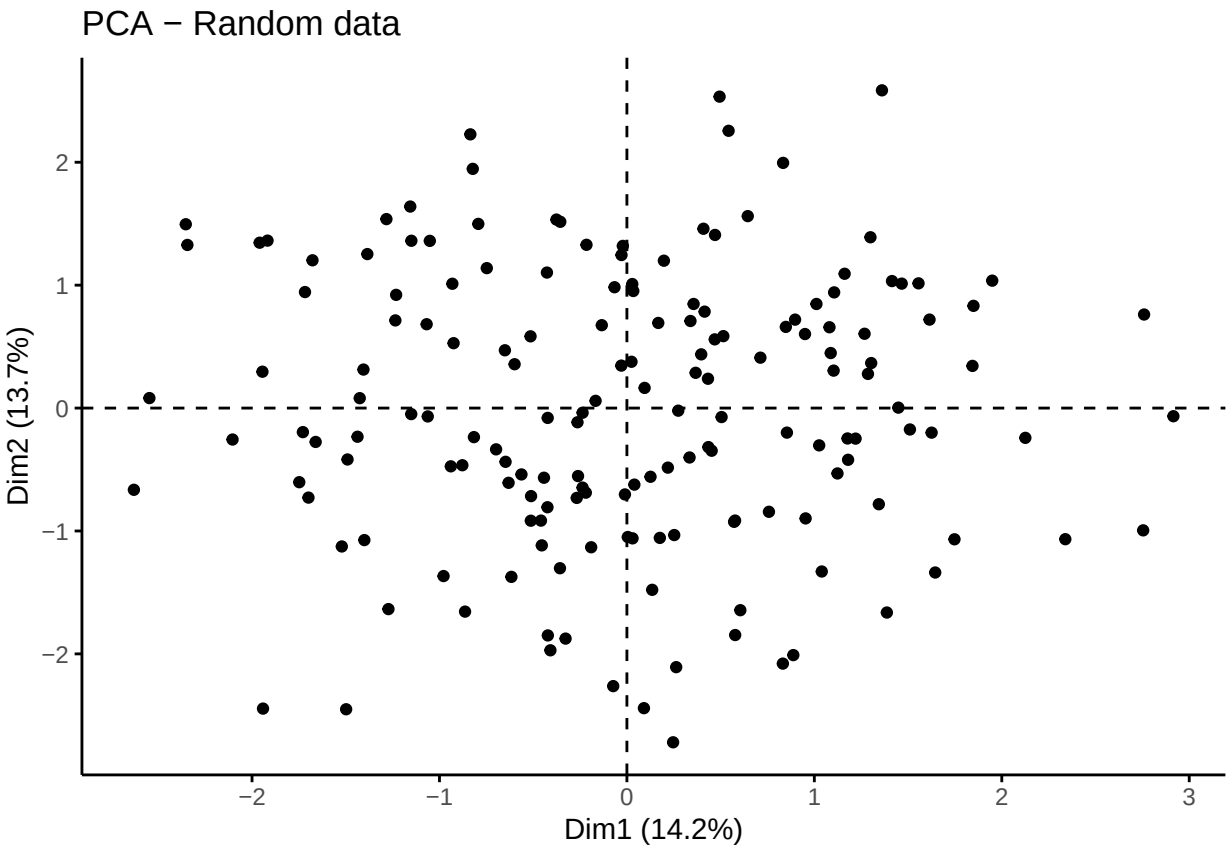
These two pairplots appear quite different. In the first plot, there seem to be visible patterns and correlations among variables, suggesting potential groupings within the data. This could indicate that dividing the data into distinct clusters might be meaningful.

We can do the same in two dimension in the PC space.

```
fviz_pca_ind(prcomp(scale(data)), title = "PCA - Original data",
             palette = "jco",
             geom = "point", ggtheme = theme_classic(),
             legend = "bottom")
```




```
fviz_pca_ind(prcomp(scale(random_df)), title = "PCA - Random data",
             geom = "point", ggtheme = theme_classic())
```



Hopkins statistic (statistical method)

One way to measure the cluster tendency is the Hopkins statistic:

$$H = \frac{\sum_{i=1}^{m} o_i}{\sum_{i=1}^{m} r_i + \sum_{i=1}^{m} o_i}$$

This statistic essentially measures the difference between the distances of a sample of points from the dataset to their nearest neighbors within the dataset, and the distances of uniformly generated random points to their nearest neighbors in the dataset. If these distances are similar, the data does not exhibit a clustering tendency, and the Hopkins statistic will have a value close to $H = 0.5$

```
set.seed(333)

clustertend::hopkins(data)
```

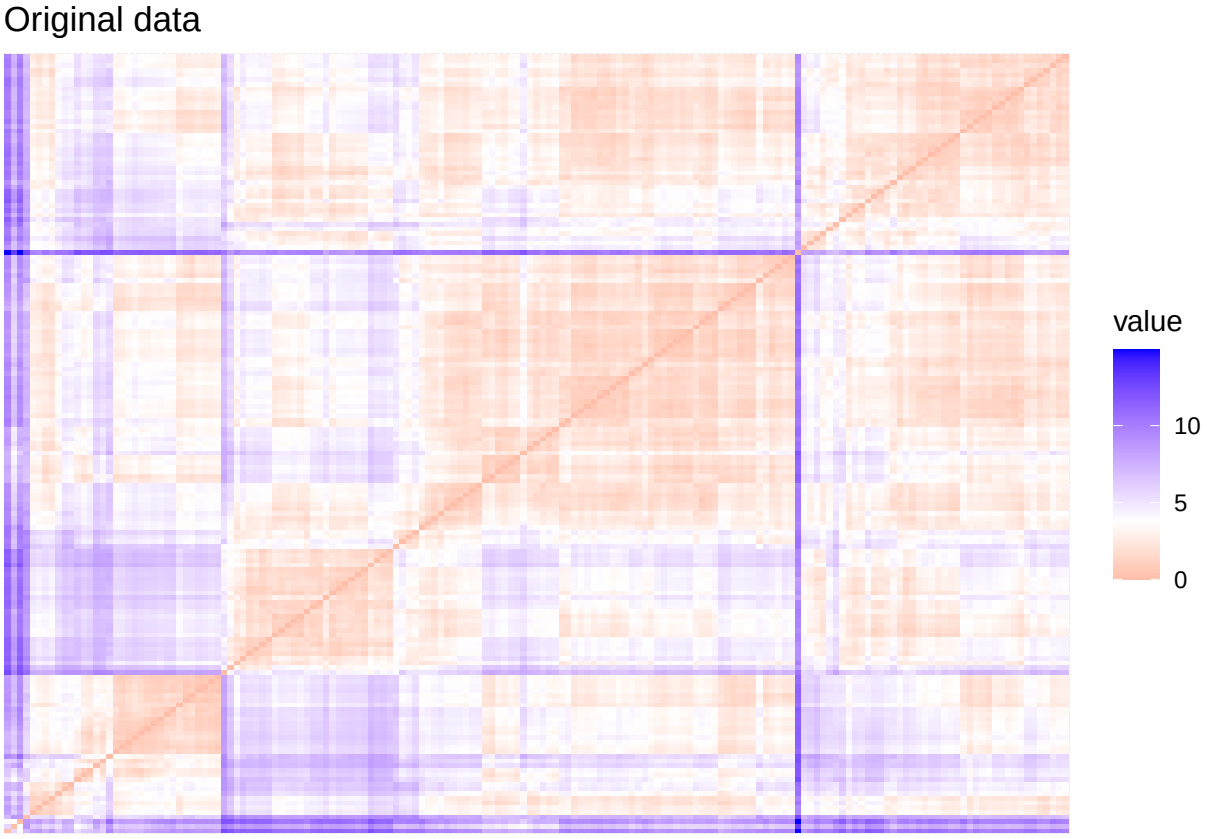
```
## Warning in clustertend::hopkins(data): Package `clustertend` is deprecated.
## Use package `hopkins` instead.
```

```
## $H
## [1] 0.07042849
```

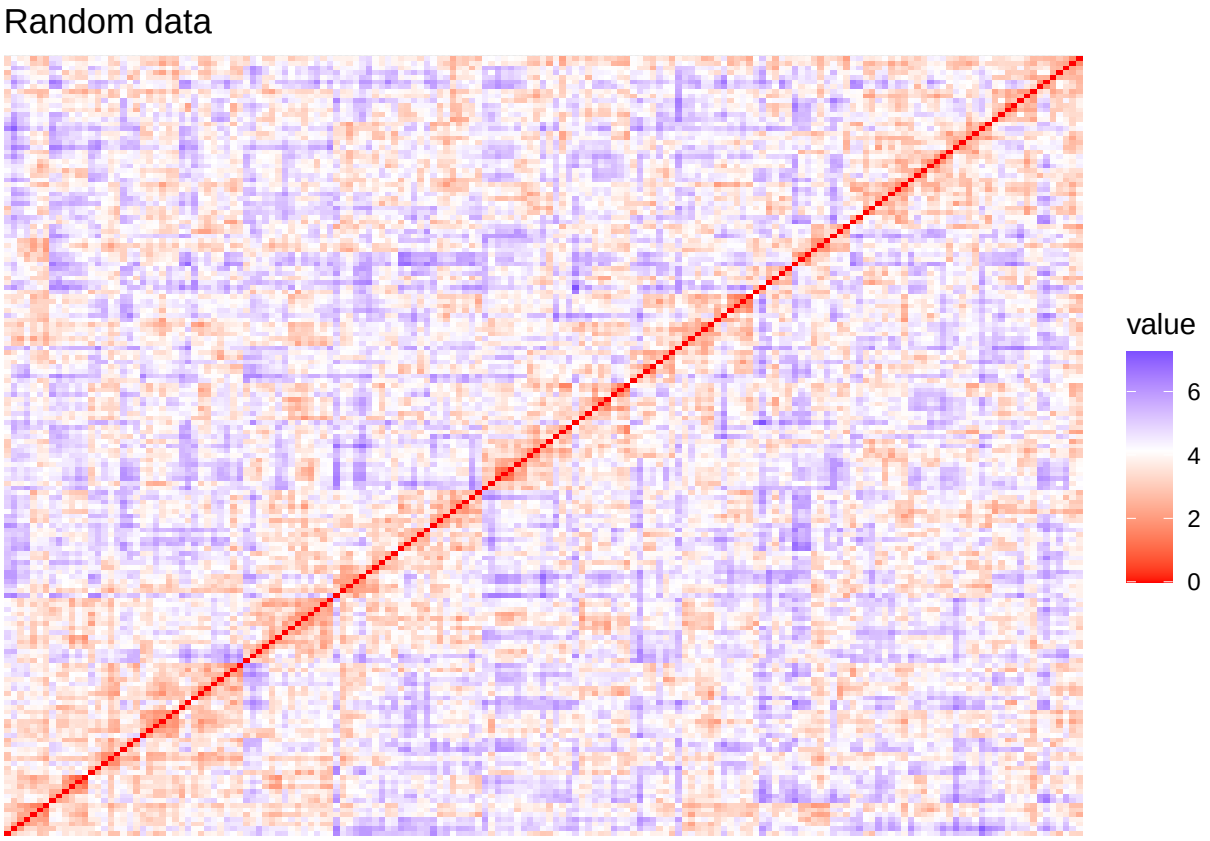
A value of $H = 0.0704$ indicate that the data have a strong tendency to be clusterized.

VAT method

```
fviz_dist(dist(scale(data)), show_labels = FALSE)+
  labs(title = "Original data")
```




```
fviz_dist(dist(scale(random_df)), show_labels = FALSE)+
  labs(title = "Random data")
```



According to the VAT (Visual Assessment of Tendency) algorithm, the data shows a clear cluster tendency. The presence of distinct square-shaped patterns in the VAT plot suggests that the dataset contains separable groups, reinforcing the need to apply clustering methods to identify these clusters.

Hierarchical (Agglomerative)

For hierarchical clustering, different distance metrics and linkage criteria will be tested to determine the best approach for this dataset.

I will compute the algorithm using various combinations of distances and linkage methods, evaluating their performance based on how well they preserve the structure of the data and align with the original distance matrix. This systematic approach ensures the most suitable hierarchical method is selected.

1. Euclidian distance:

$$d(x_i, x_j) = \sqrt{\sum_{h=1}^{d} (x_{ih} - x_{jh})^2}$$

2. Manhattan distance:

$$d(x_i, x_j) = \sum_{h=1}^{d} |x_{ih} - x_{jh}|$$

In with i will apply the following linkage criterion:

- Single Linkage

- Average Linkage
- Complete Linkage
- Ward Distance

The best clustering method will be selected by evaluating the connections made by the algorithm, comparing the cophenetic distances to the original distance matrix. This approach ensures that the chosen method accurately reflects the data structure and preserves the relationships between observations.

```
dist_measures <- list(euclidean = dist(scaled_data, method = "euclidean"),
                      manhattan = dist(scaled_data, method = "manhattan"))

linkage_methods <- c("average", "single", "complete", "ward.D2")

results <- list()

for(i in names(dist_measures)){
  for(link in linkage_methods){

    hc <- hclust(dist_measures[[i]], method = link)

    coph_cor <- cor(dist_measures[[i]], cophenetic(hc))

    results[[paste(i, link, sep = '_')]] <- list(hier_clust = hc,
                                                 coph_value = coph_cor)
  }
}

data.frame(row.names = names(results),
           cophenetic_correlation = sapply(results, function(x)  x$coph_value))
```

```
##                    cophenetic_correlation
## euclidean_average               0.8394248
## euclidean_single                0.7604288
## euclidean_complete              0.4908965
## euclidean_ward.D2               0.5290291
## manhattan_average               0.7650241
## manhattan_single                0.6616520
## manhattan_complete              0.5736511
## manhattan_ward.D2               0.4987501
```

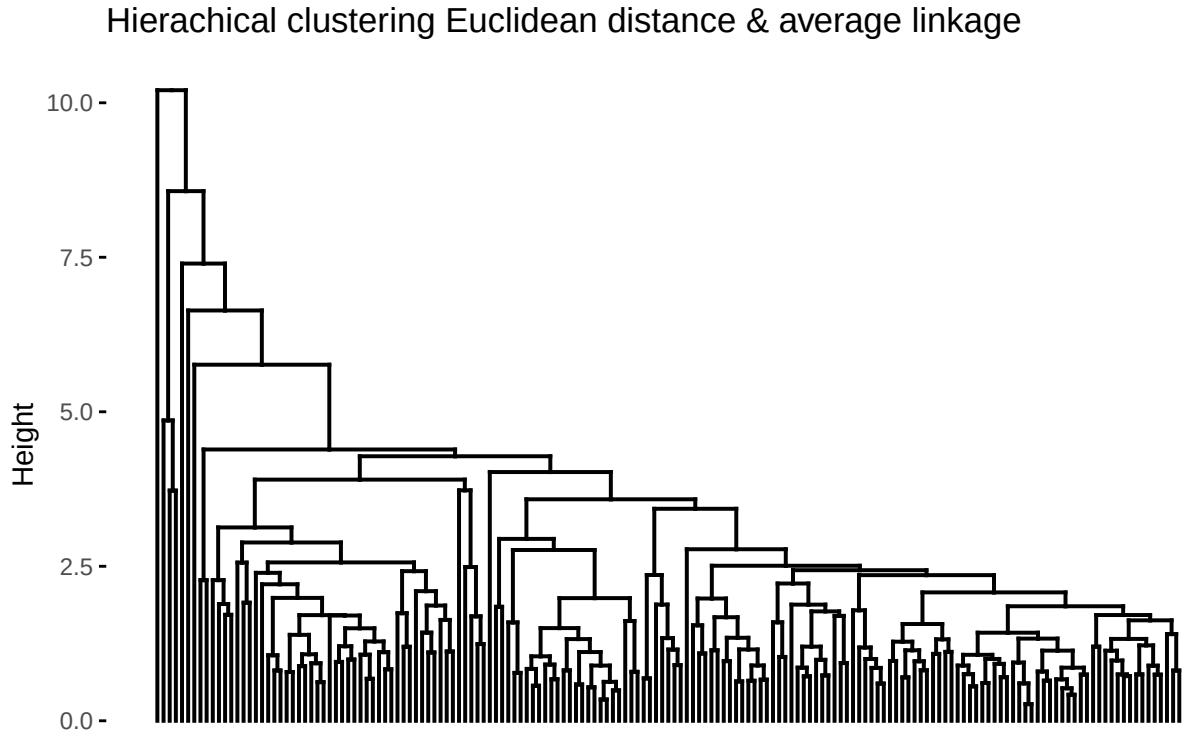
Based on the cophenetic distances, which measures the correlation between the dissimilarity matrix and the clustering structure generated by the hierarchical clustering method, the combination of the Euclidean distance and the average linkage method achieves the highest correlation, making it the best-performing approach in this context.

The best partition, according to the cophenetic distance, is achieved using Euclidean distance and average linkage, as it best preserves the pairwise dissimilarities between observations.

```
fviz_dend(results$euclidean_average$hier_clust, cex = .3,
          main = "Hierachical clustering Euclidean distance & average linkage",
          show_labels = F)
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## i The deprecated feature was likely used in the factoextra package.
##   Please report the issue at <https://github.com/kassambara/factoextra/issues>.
```

```
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



The interpretation of this dendrogram is straightforward. On the left side, there are a few observations that are noticeably distant from both the other data points and from one another. The algorithm is likely to identify these as a distinct cluster. If the number of clusters K is increased, this cluster may be further split due to its relatively small within-cluster sum of squares (WSS).

The second-best partition in terms of cophenetic distance is the one created using the Manhattan distance and complete linkage. This method seems to provide a better fit compared to other distance metrics and linkage methods, though it's not the best overall.

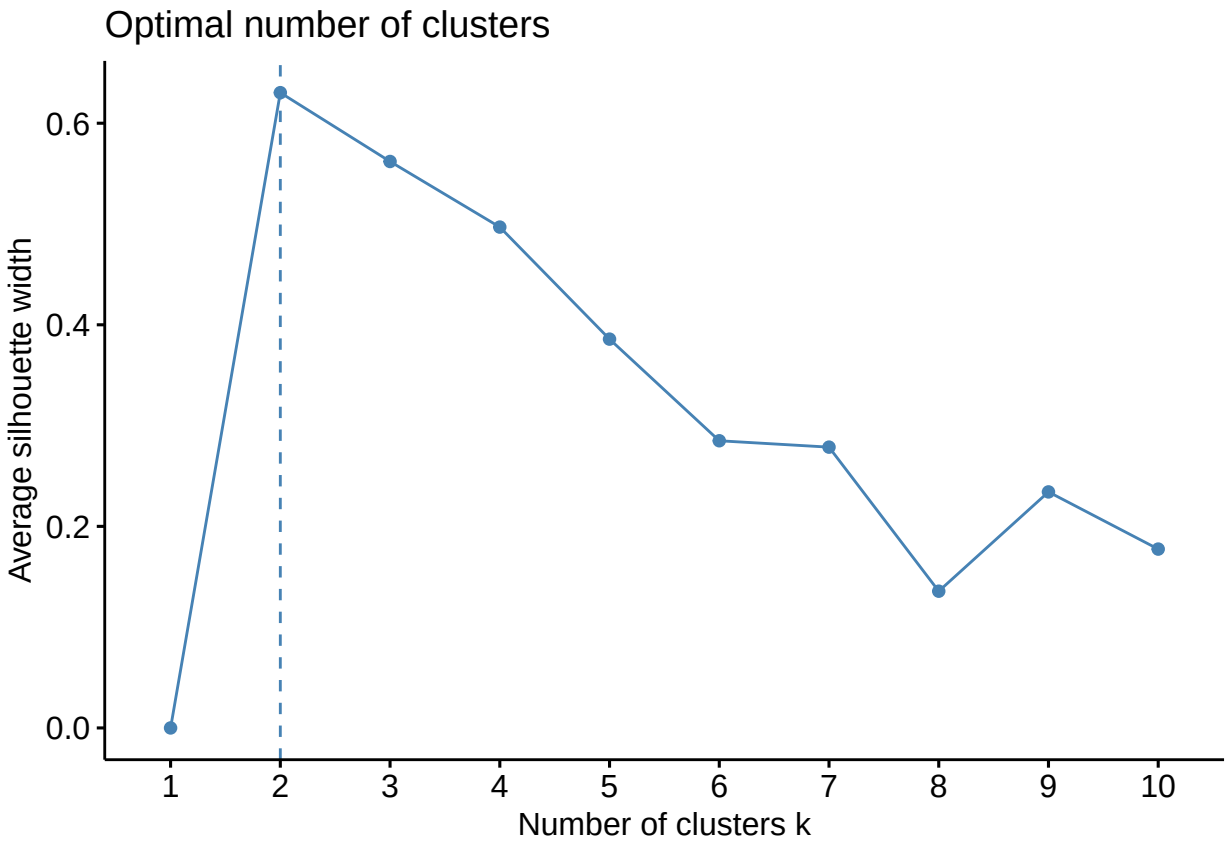
Manhattan distance tends to be more robust to outliers than Euclidean distance, and complete linkage often leads to more compact and well-separated clusters. The cophenetic distance, which measures how well the hierarchical clustering preserves the pairwise dissimilarities of the observations, suggests that this partitioning method offers a good balance between the internal cohesion of clusters and their separation.

Partition euclidean distance and average linkage

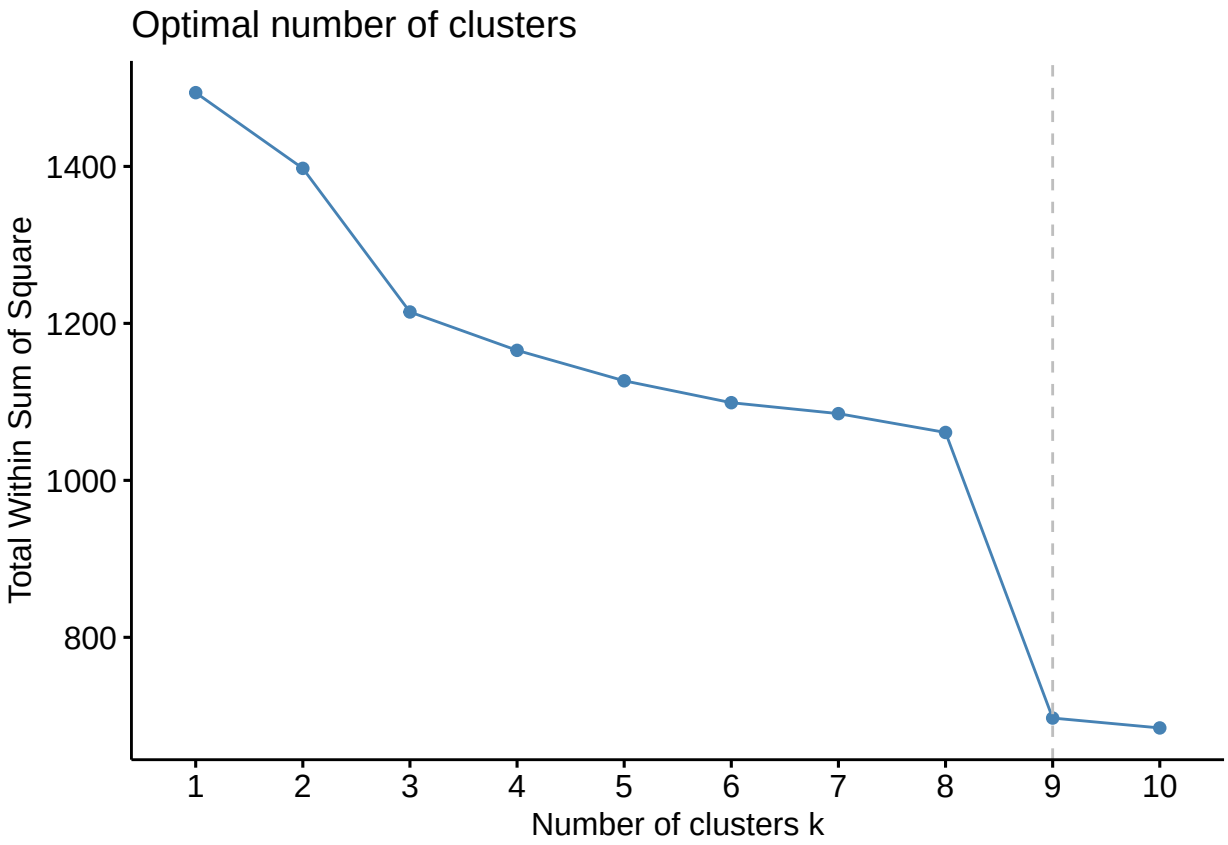
In order to find the optimal number of cluster for this data I applied the following techniques:

- Average silhouette width
- Total WSS
- GAP statistic

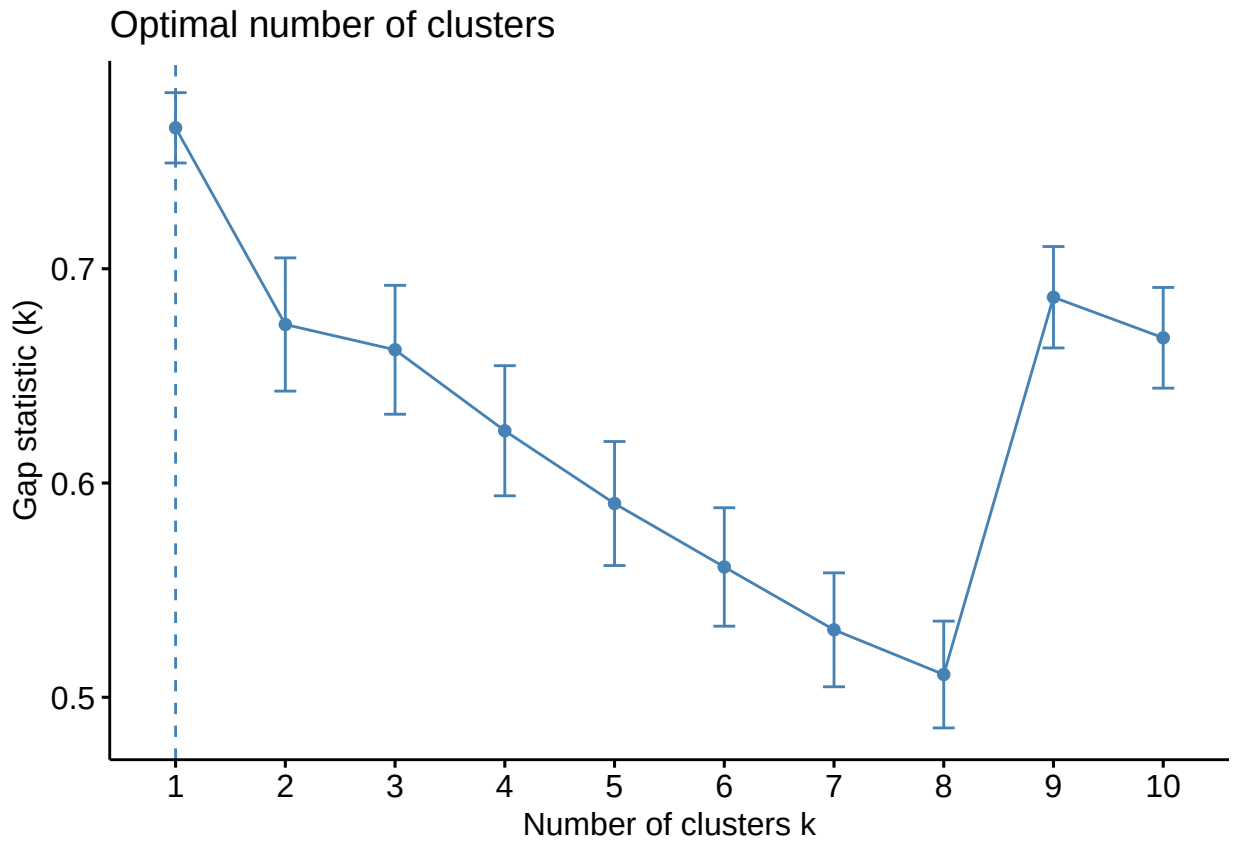
```
fviz_nbclust(scaled_data, hcut, method = "silhouette", hc_metric = "euclidean", hc_method = "average")
```




```
fviz_nbclust(scaled_data, hcut, method = "wss", hc_metric = "euclidean", hc_method = "average") +
  geom_vline(xintercept = 9, linetype = 2, col = "grey")
```




```
fviz_nbclust(scaled_data, hcut, method = "gap_stat", hc_metric = "euclidean", hc_method = "average", nb
```

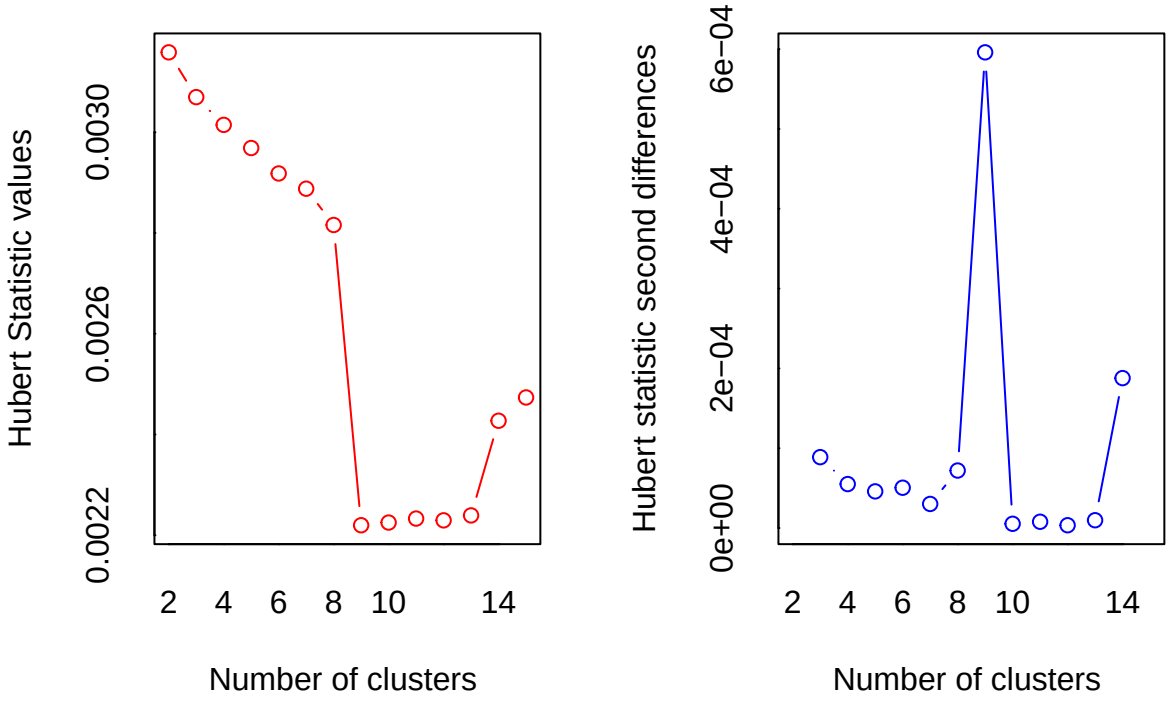


the result are:

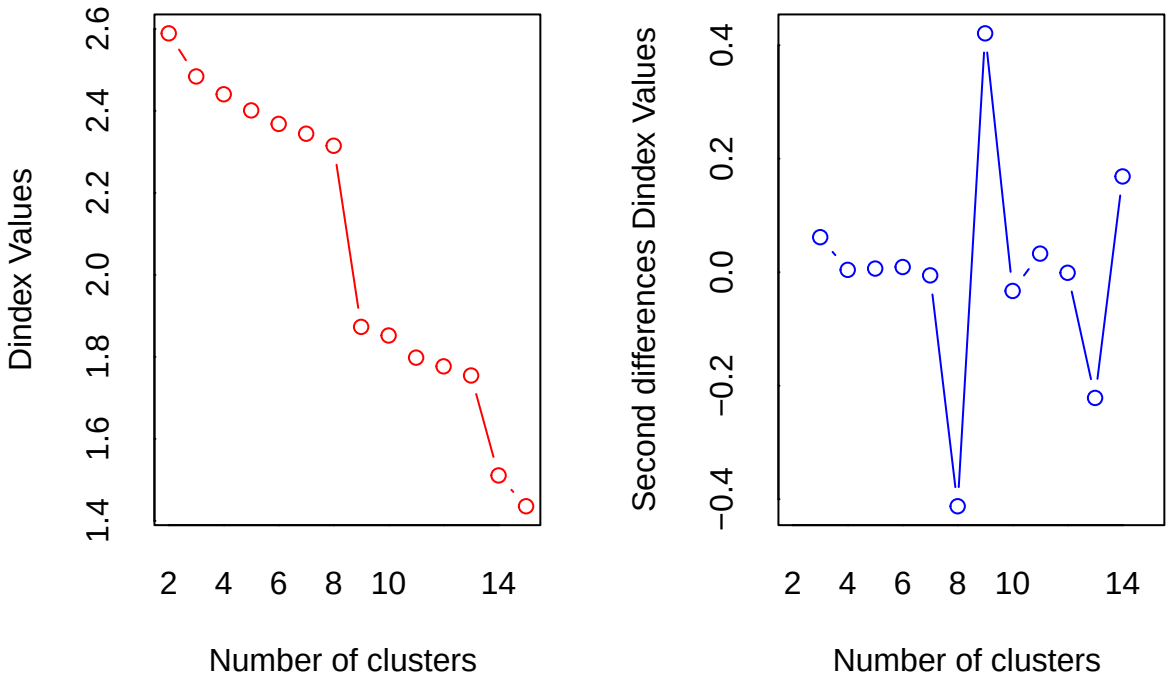
- Average silhouette width suggest k = 2
- Total WSS suggest k = 9
- Gap statistic suggest k = 1

```
nb_res <- NbClust(scaled_data, distance = 'euclidean', method = 'average')
```

```
## Warning in pf(beale, pp, df2): NaNs produced
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

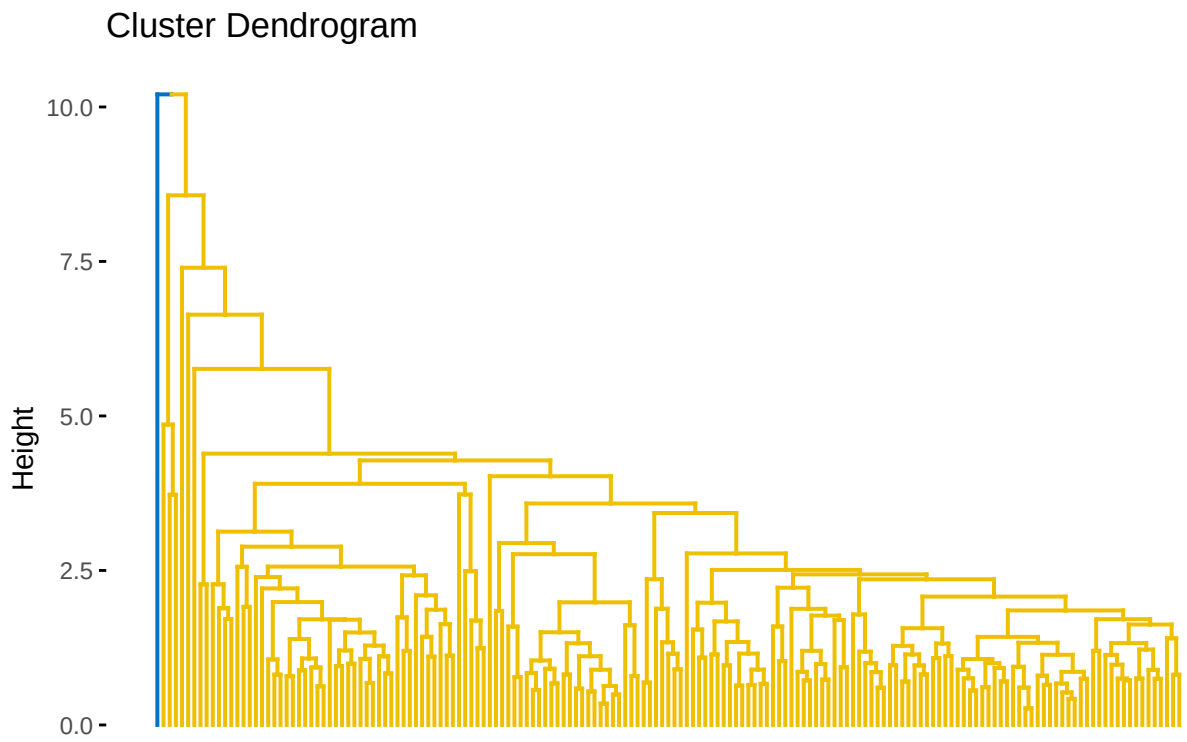



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 8 proposed 2 as the best number of clusters
## * 1 proposed 3 as the best number of clusters
## * 1 proposed 5 as the best number of clusters
## * 1 proposed 7 as the best number of clusters
## * 1 proposed 8 as the best number of clusters
## * 8 proposed 9 as the best number of clusters
## * 1 proposed 13 as the best number of clusters
## * 3 proposed 15 as the best number of clusters
##
##                   ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  2
##
##
## *******************************************************************
```

By running the `NbClust` function, we can gain an overview of the optimal partitioning for our dataset. In this case, the most frequently selected options are 2 and 9 clusters.

Given these results, the algorithm will likely partition all the outliers separately.

```
hc_res <- eclust(scaled_data, "hclust", k = 2, hc_metric = "euclidean",
                 hc_method = "average", graph = F) # chose k = 4 based on TSS plot

fviz_dend(hc_res, show_labels = F,
          palette = "jco")
```



As expected, the algorithm is significantly influenced by the presence of outliers, resulting in the creation of a single cluster containing one single outliers. This suggests that an agglomerative clustering algorithm may not be the most suitable choice for this dataset.

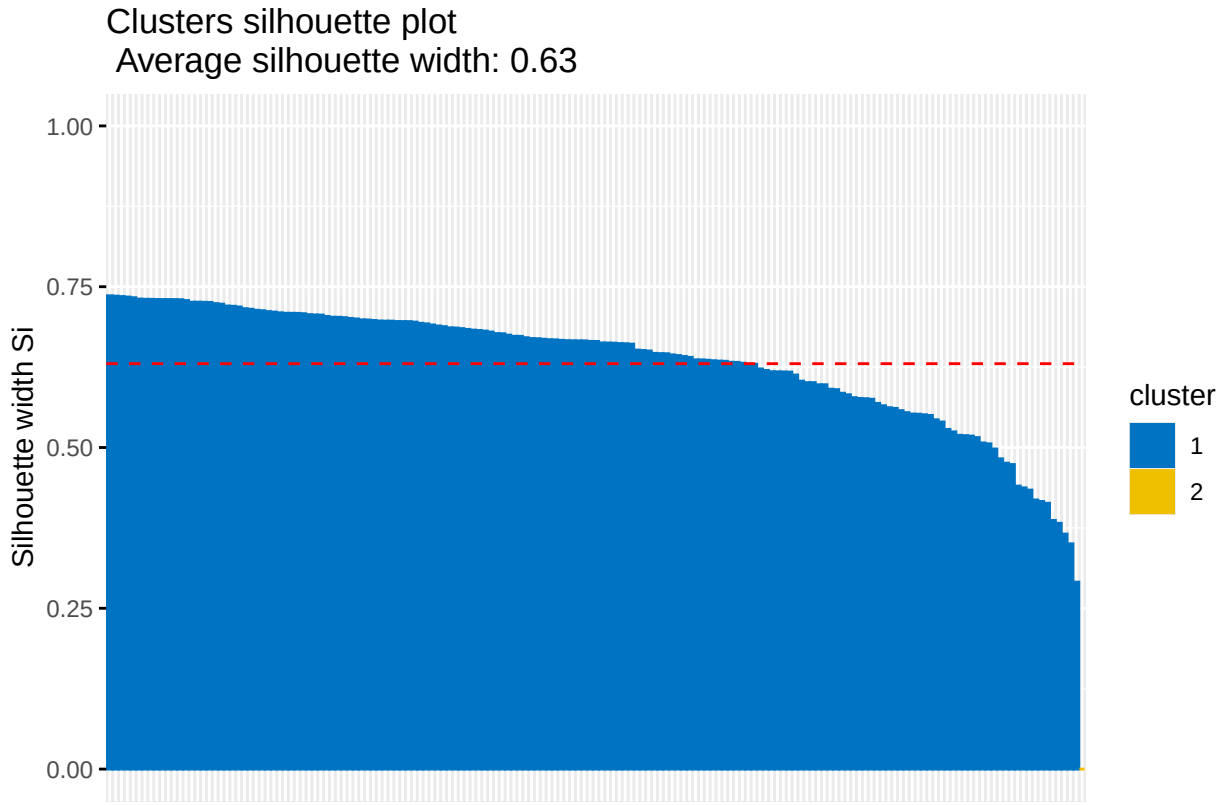
```
idx_cl <- which(hc_res$cluster== 2)
rownames(data)[idx_cl]
```

```
## [1] "Nigeria"
```

The second cluster is composed only by Nigeria

```
fviz_silhouette(hc_res, palette = "jco")
```

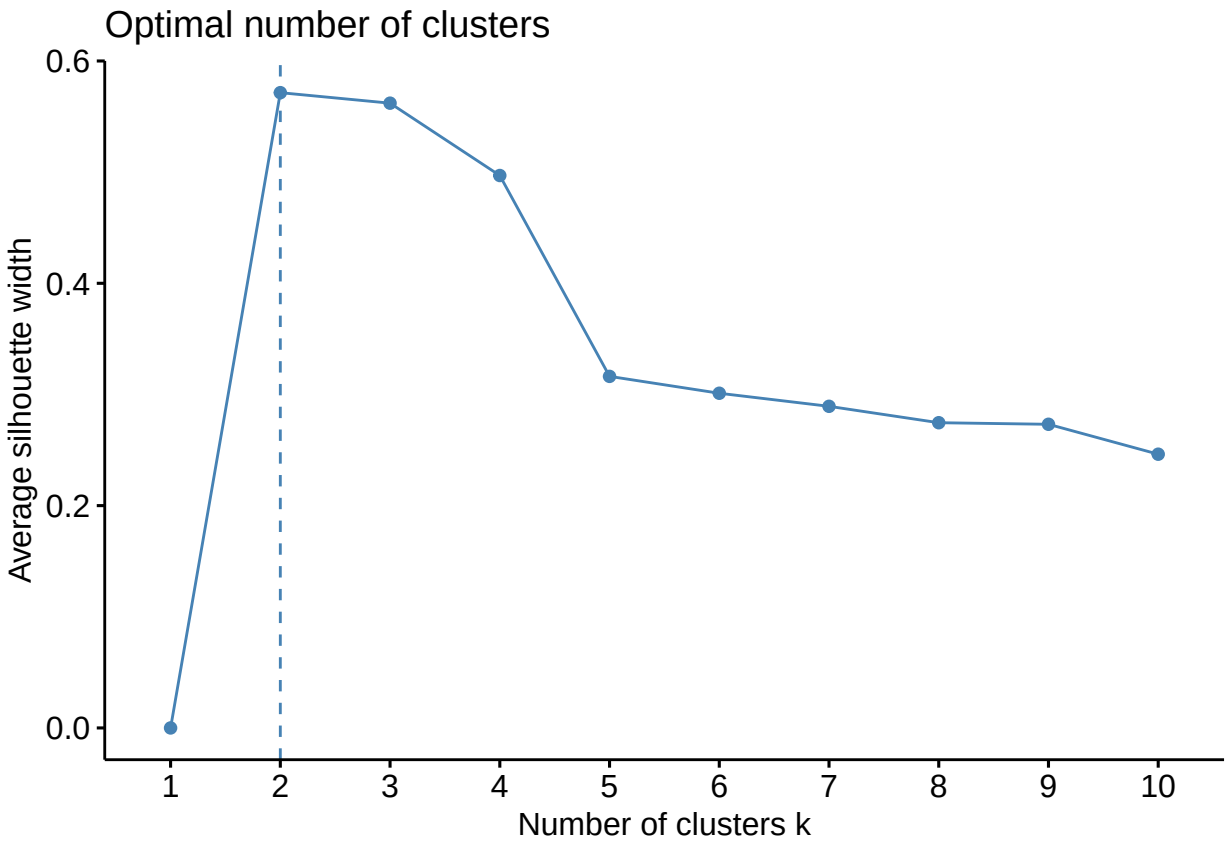
```
##   cluster size ave.sil.width
## 1       1  166          0.63
## 2       2    1          0.00
```



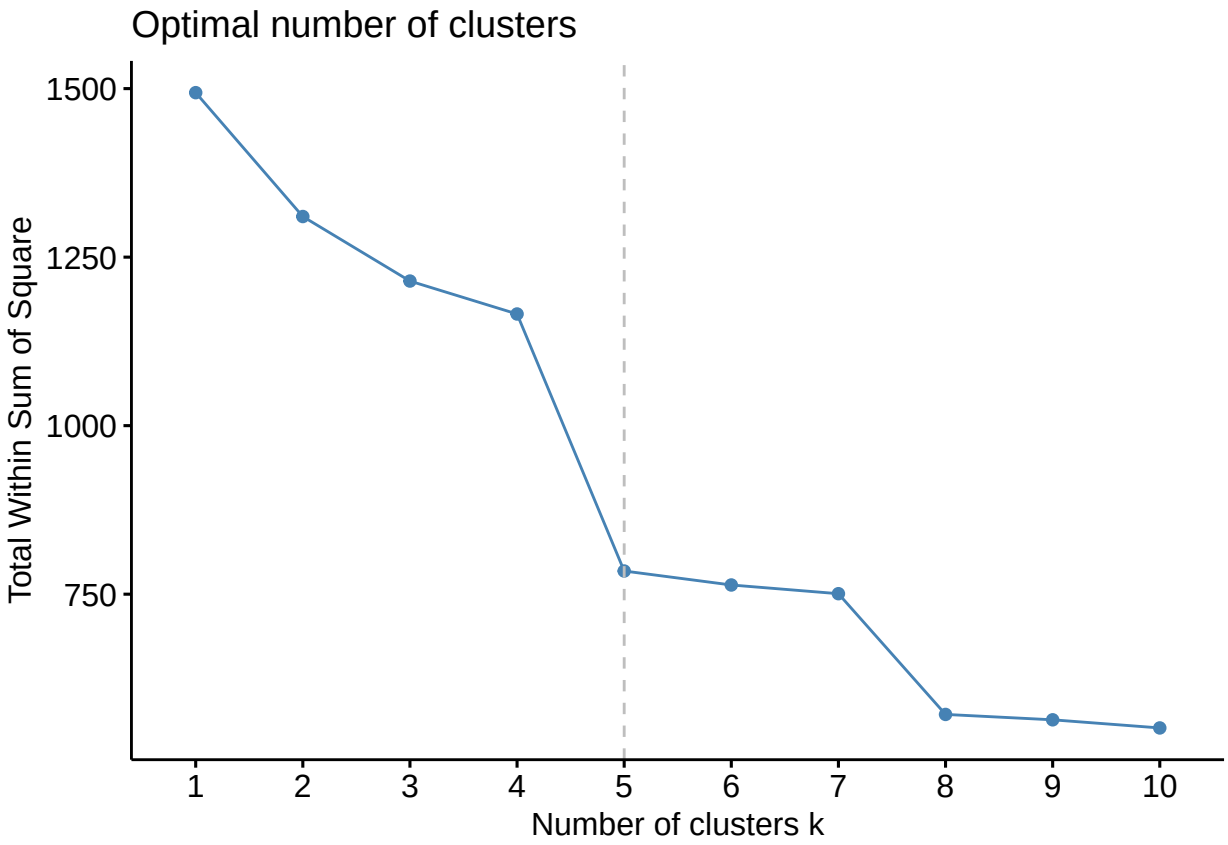
Although the average silhouette score is not low (due to the second cluster consisting of only one element that is distant from the other observations), the partition does not appear to be meaningful. Before drawing any conclusions or making further observations, it is essential to test alternative algorithms on this dataset.

Partition Manhattan distance and Average linkage

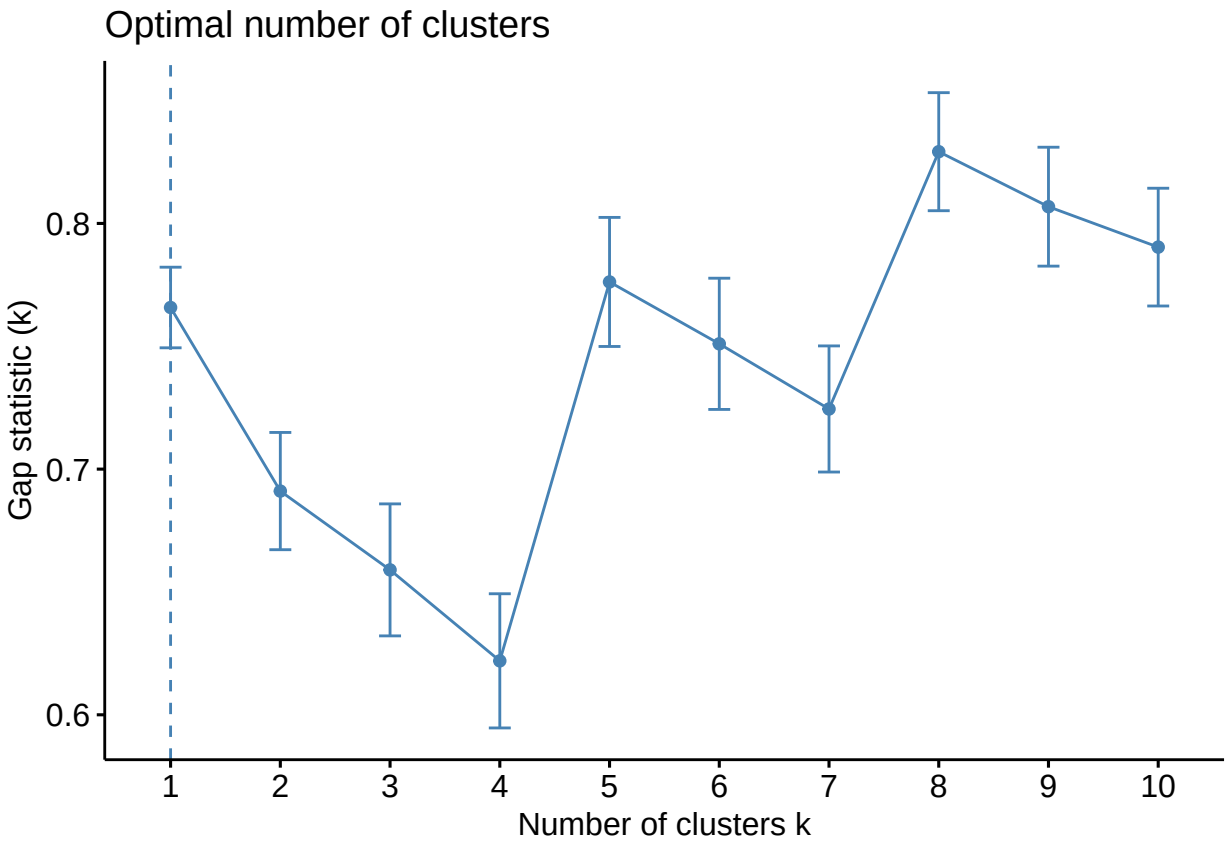
```
fviz_nbclust(scaled_data, hcut, method = "silhouette", hc_metric = "manhattan", hc_method = "average")
```




```
fviz_nbclust(scaled_data, hcut, method = "wss", hc_metric = "manhattan", hc_method = "average") +
  geom_vline(xintercept = 5, linetype = 2, col = "grey")
```



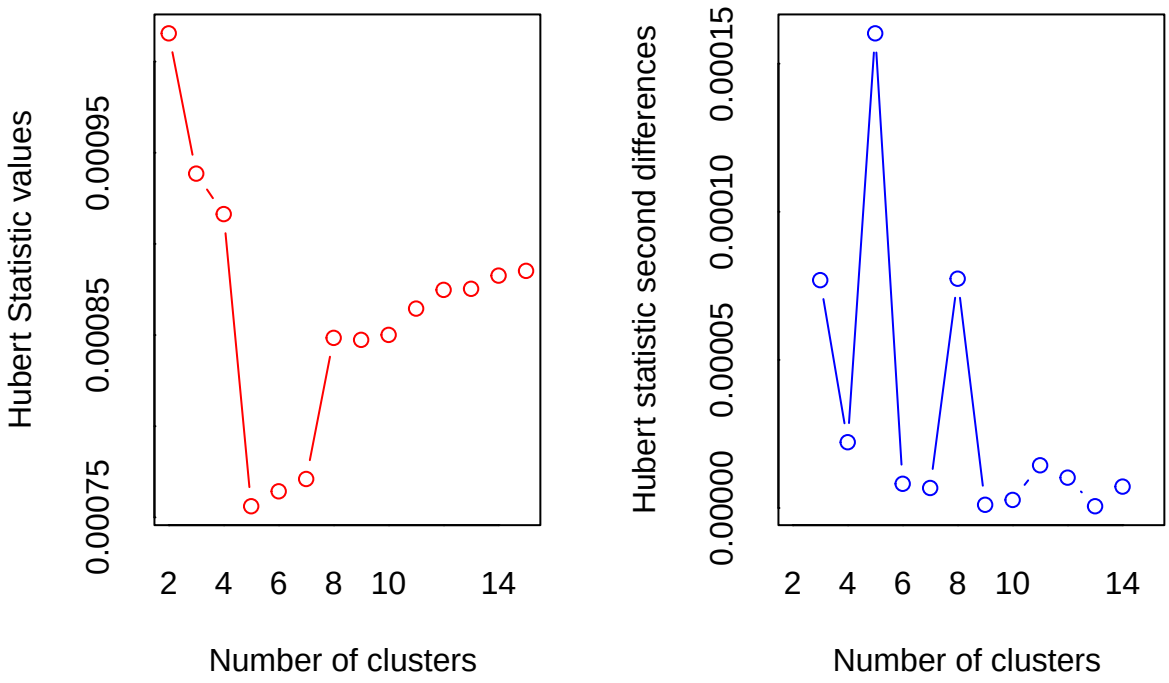

```
fviz_nbclust(scaled_data, hcut, method = "gap_stat", hc_metric = "manhattan", hc_method = "average", nb
```



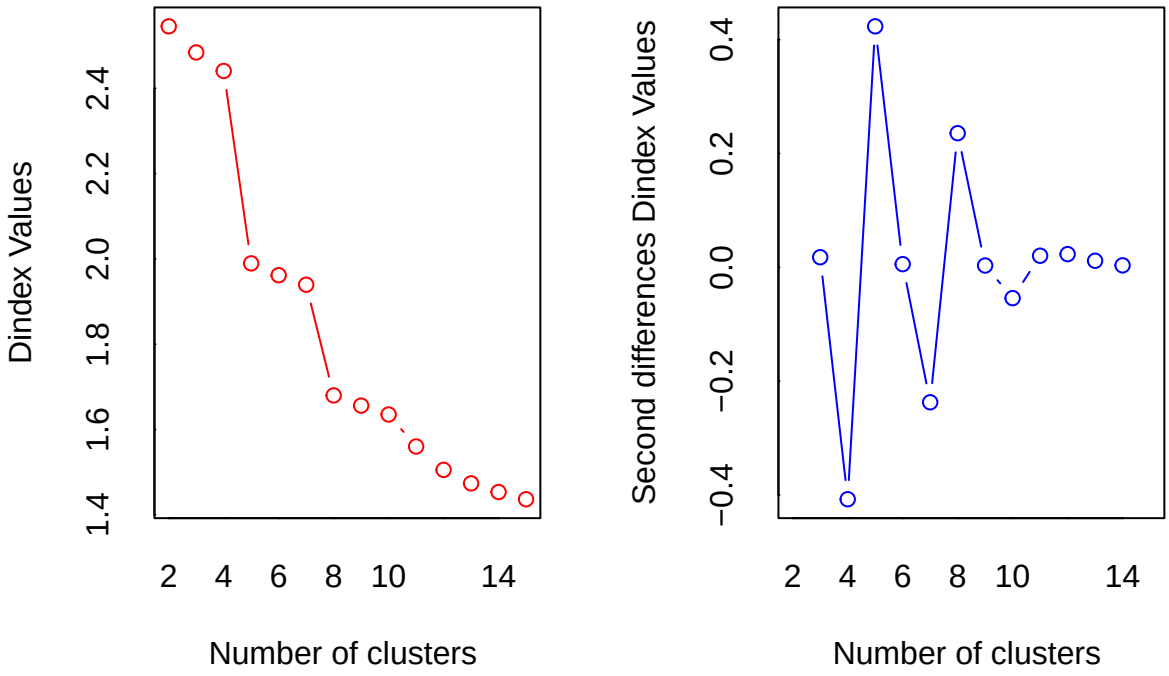
For similar reasons as in the previous case, the optimal value for k in this instance appears to be 3.

```
nb <- NbClust(scaled_data, distance = 'manhattan', method = 'average')
```

```
## Warning in pf(beale, pp, df2): NaNs produced
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

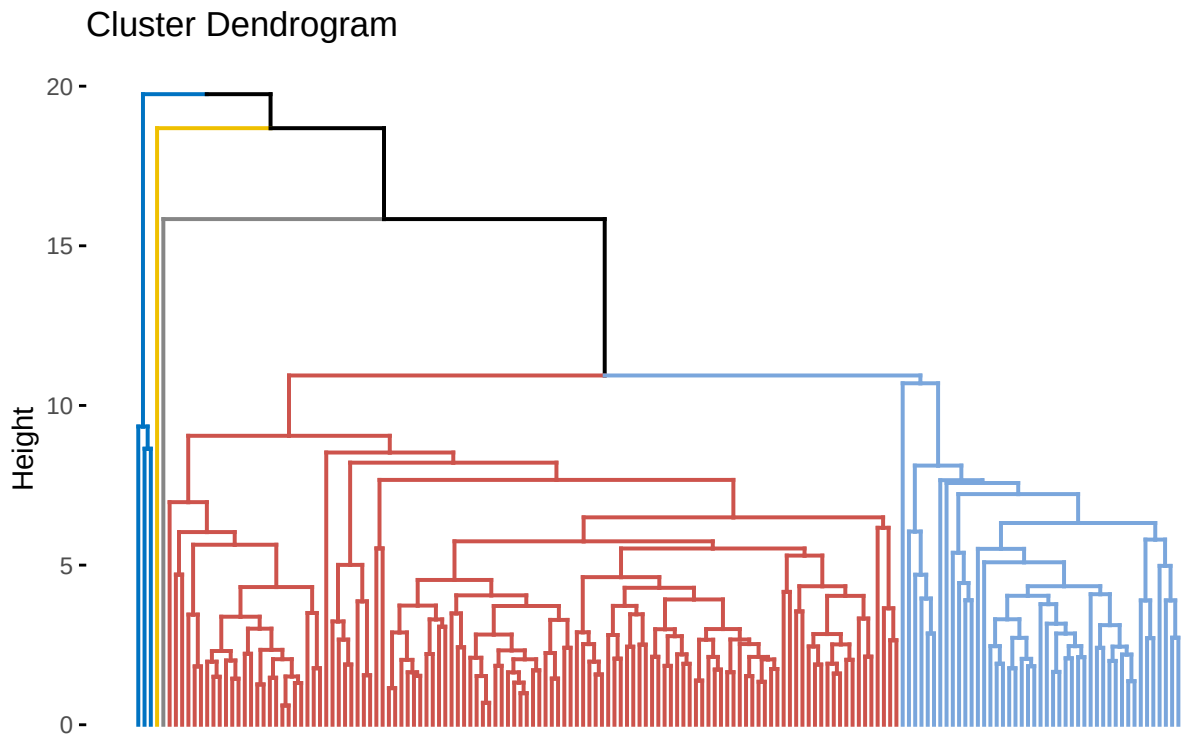



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 6 proposed 2 as the best number of clusters
## * 2 proposed 3 as the best number of clusters
## * 3 proposed 4 as the best number of clusters
## * 7 proposed 5 as the best number of clusters
## * 2 proposed 8 as the best number of clusters
## * 2 proposed 10 as the best number of clusters
## * 1 proposed 11 as the best number of clusters
## * 1 proposed 13 as the best number of clusters
##
##                     ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  5
##
##
## *******************************************************************
```

The majority rule here suggest us an optimal number of clusters equals to 4

```
hc_res <- eclust(scaled_data, "hclust", k = 5, hc_metric = "manhattan",
                 hc_method = "average", graph = F)
```

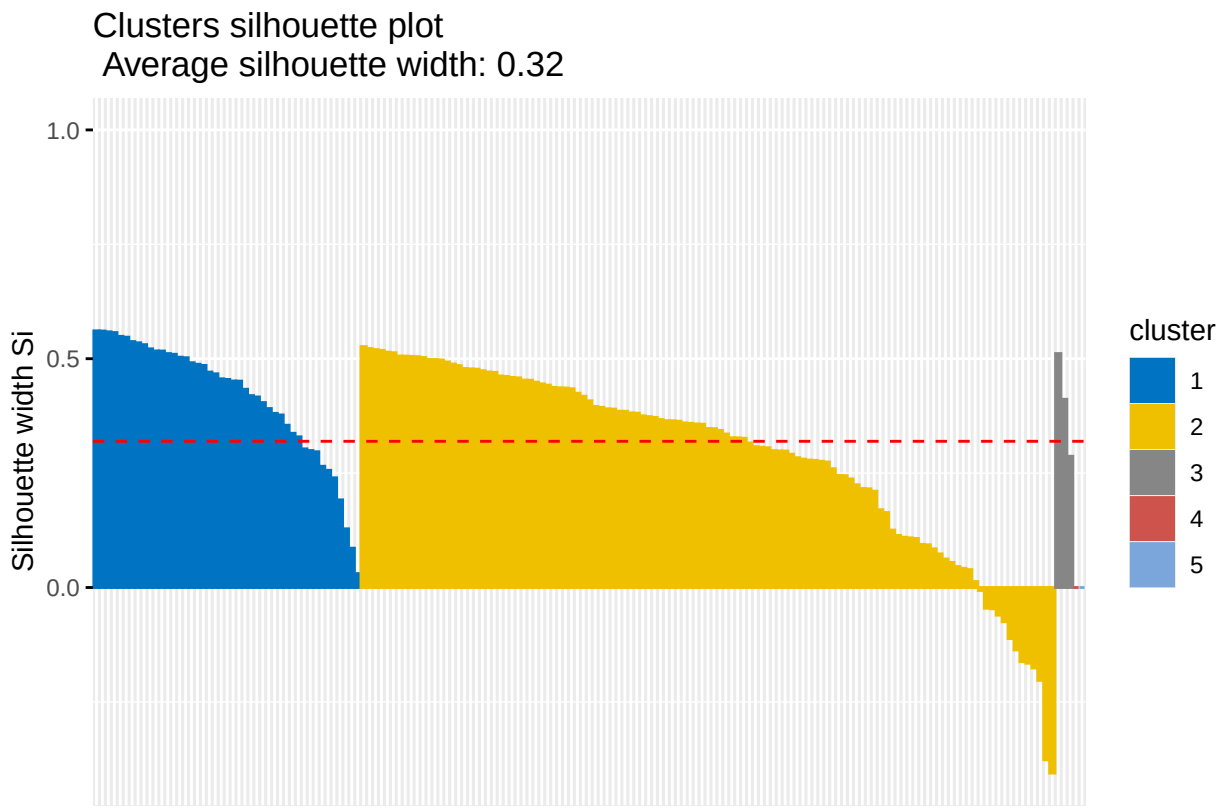
```
fviz_dend(hc_res, show_labels = F,
          palette = "jco")
```



We can observe two meaningful clusters and three clusters containing only a few outlier observations.

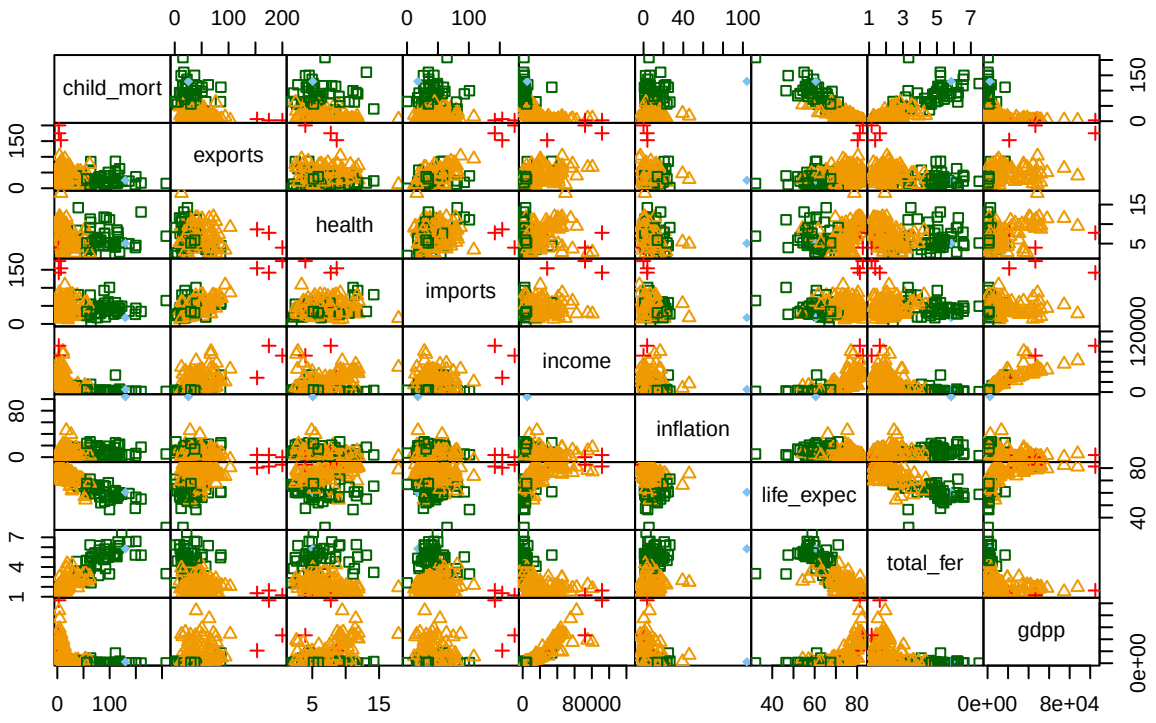
```
fviz_silhouette(hc_res, palette = "jco")
```

```
##   cluster size ave.sil.width
## 1       1   45          0.41
## 2       2  117          0.29
## 3       3    3          0.40
## 4       4    1          0.00
## 5       5    1          0.00
```



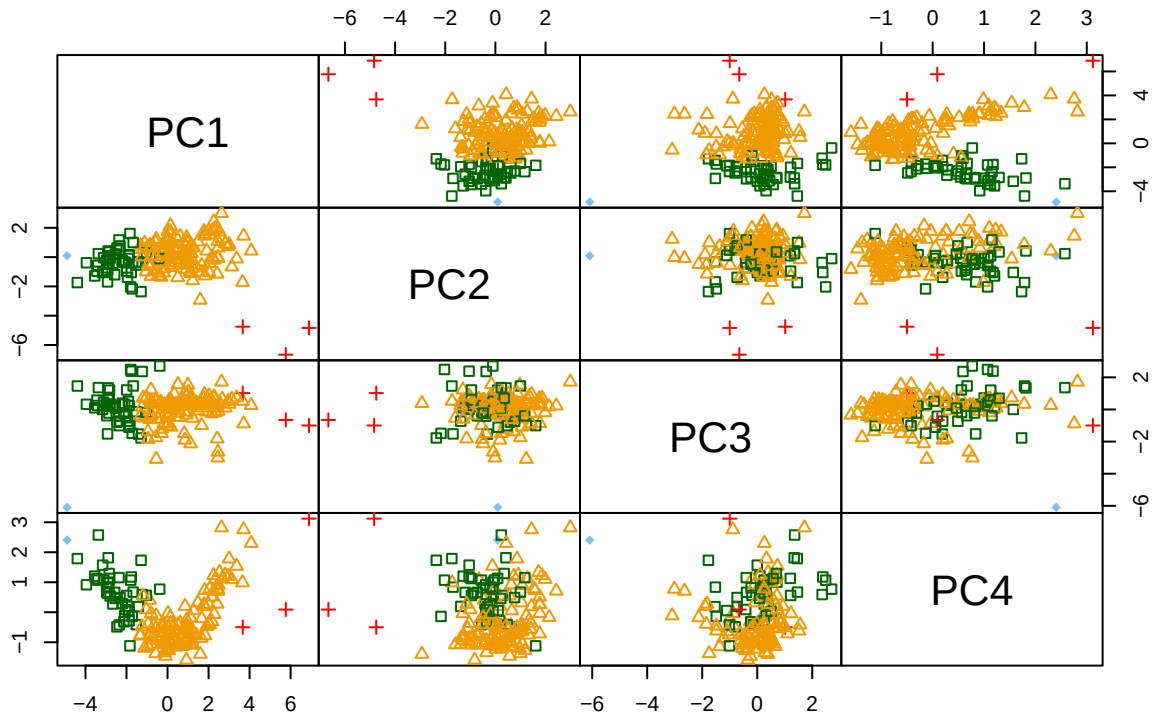
According to the silhouette plot, there are a lot of misclassifications in the second cluster, and overall, the average silhouette width is not high. Below is a representation of this partition:

```
pairs(data, gap = 0, pch = c(0, 2, 3, 18) [hc_res$cluster],
      col = c("darkgreen", "orange2", "red", "skyblue2") [hc_res$cluster],
      main = "Pairplot of HC Manhattan & Complete Linkage")
```



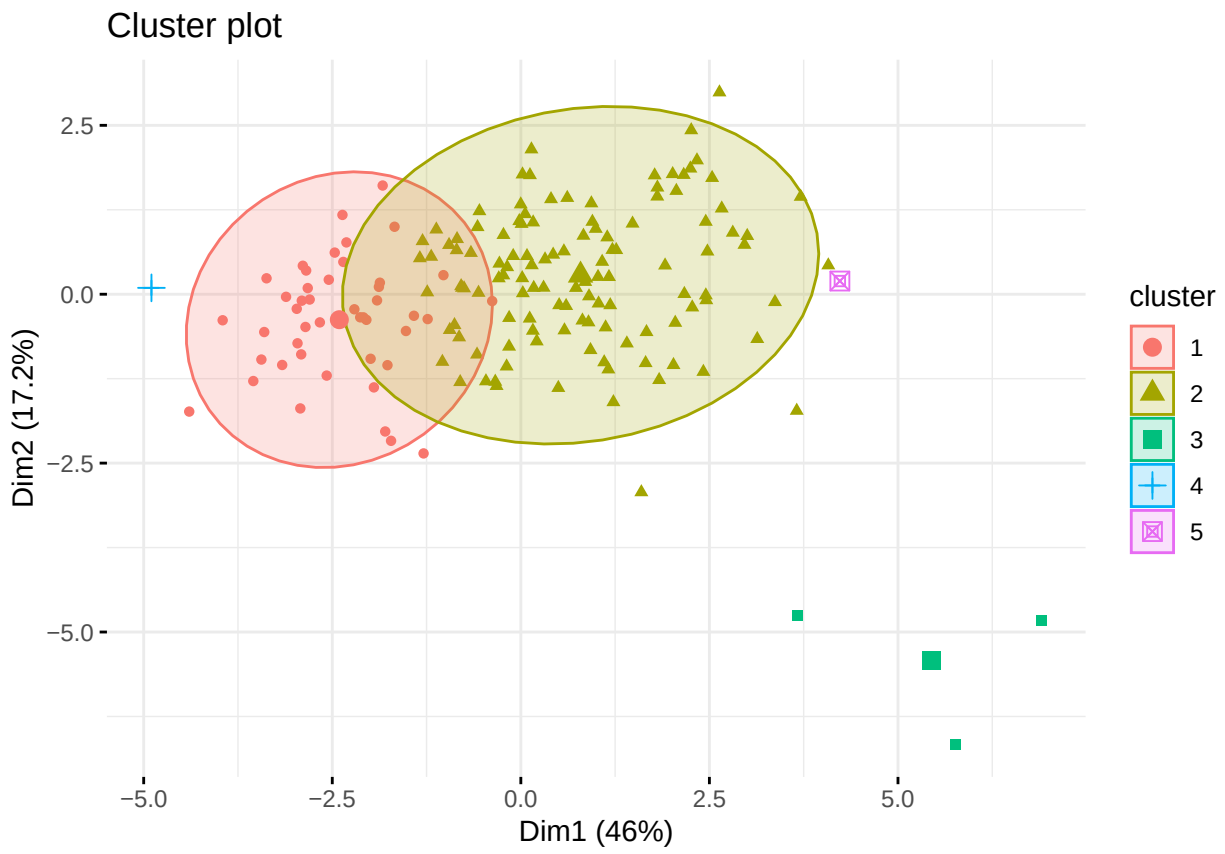
We note that some variables, such as `gdpp` and `child_mort` are highly significant in terms of classification. Now, let's look at the pair plot in the principal component space, even though this partition was not generated in this space.

```
pairs(pca_data, gap = 0, pch = c(0, 2, 3, 18) [hc_res$cluster],
      col = c("darkgreen", "orange2", "red","skyblue2") [hc_res$cluster],
      main = "Pairplot of HC Manhattan & Complete Linkage in PC space")
```




```
fviz_cluster(hc_res, scaled_data, ellipse.type = "norm", ggtheme = theme_minimal(), labelsize = 0)
```

```
## Too few points to calculate an ellipse
## Too few points to calculate an ellipse
## Too few points to calculate an ellipse
```



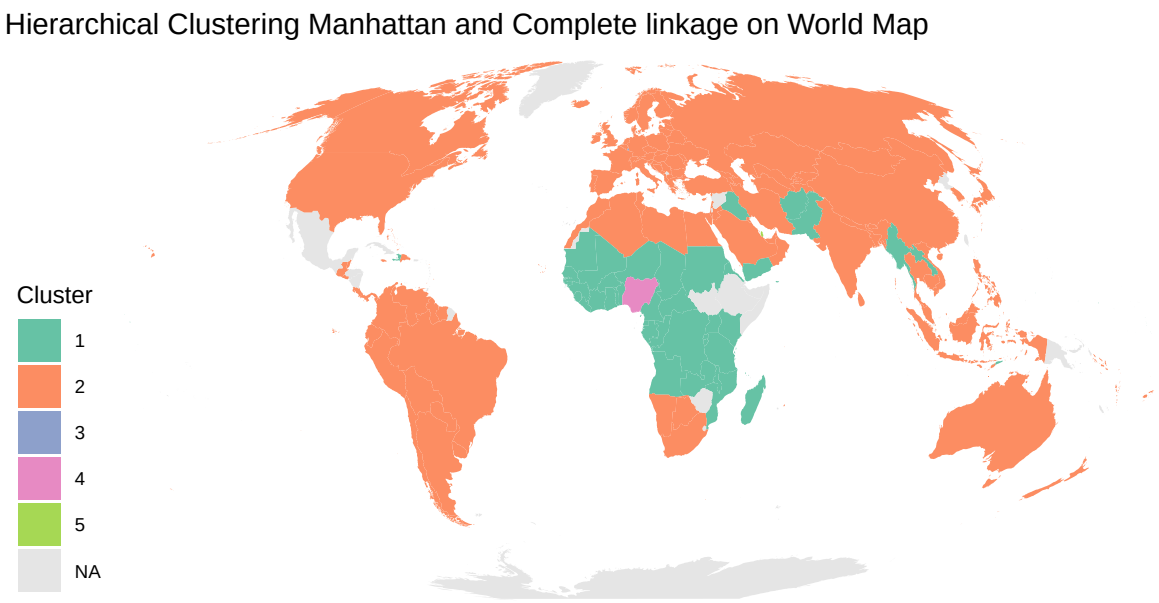
This clustering plot is provided for visualization purposes only, as the actual partitioning was performed in the **original variable space** rather than in the PCA-transformed space. Consequently, the cluster separations visible here may not fully reflect the criteria used to create the clusters.

The clusters 1 and 2 displayed in the PCA space exhibit **significant overlap**, indicating that the underlying features contributing to cluster separation in the original space are less distinct when reduced to two dimensions.

This partition reveals a classification of countries primarily based on socio-economic indicators (such as those forming PC1), grouped mainly into two distinct clusters. Additionally, three other clusters consist solely of outliers, characterized by extreme values (both positive and negative) for both PC1 and PC2.

```
cluster_data <- tibble(
  region  = hc_res$labels,
  cluster = as.factor(unname(hc_res$cluster))
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
```

```
                          "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                          "United Kingdom" = "UK",
                          "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(!long > 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")


ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Hierarchical Clustering Manhattan and Complete linkage on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```




```
# Test for the discrepancies
anti_join(cluster_data, world_map, by = "region") %>% pull(region)
```

```
## [1] "Saint Vincent and the Grenadines"
```

```r
head(hc_res$cluster[hc_res$cluster == 2], 15)
```

```
##             Albania             Algeria Antigua and Barbuda           Argentina
##                   2                   2                   2                   2
##             Armenia           Australia             Austria          Azerbaijan
##                   2                   2                   2                   2
##             Bahamas             Bahrain          Bangladesh            Barbados
##                   2                   2                   2                   2
##             Belarus             Belgium              Belize
##                   2                   2                   2
```

The classification here is not particularly significant. Most observations fall into the second cluster, which includes the majority of wealthier countries and some developing nations. The other three clusters consist of outliers, mainly small countries, and are not clearly visible in the plot due to their size.

Partitioning Clustering method

```r
set.seed(333)
```

Regarding partitioning clustering methods, I have explored two algorithms: K-means and K-Medoids. Both algorithms rely on an iterative process where the allocation of each observation to a cluster is based on the distances between the observations and a representative point (real or artificial) for the cluster. I set a seed because the partitioning is highly sensitive to initialization, and I wanted to ensure the reproducibility of this analysis.

For those method is also crucial the initial choice of the numbers of clusters. I based the choice on elbow method on TSS, gap statistics and average silhouette width.s

K-Means

K-Means is a popular clustering algorithm that partitions data into `k` groups based on similarity. Each cluster is represented by its **centroid**, calculated as the mean of all points in the cluster. The algorithm iteratively assigns points to the nearest centroid and updates the centroids until convergence.

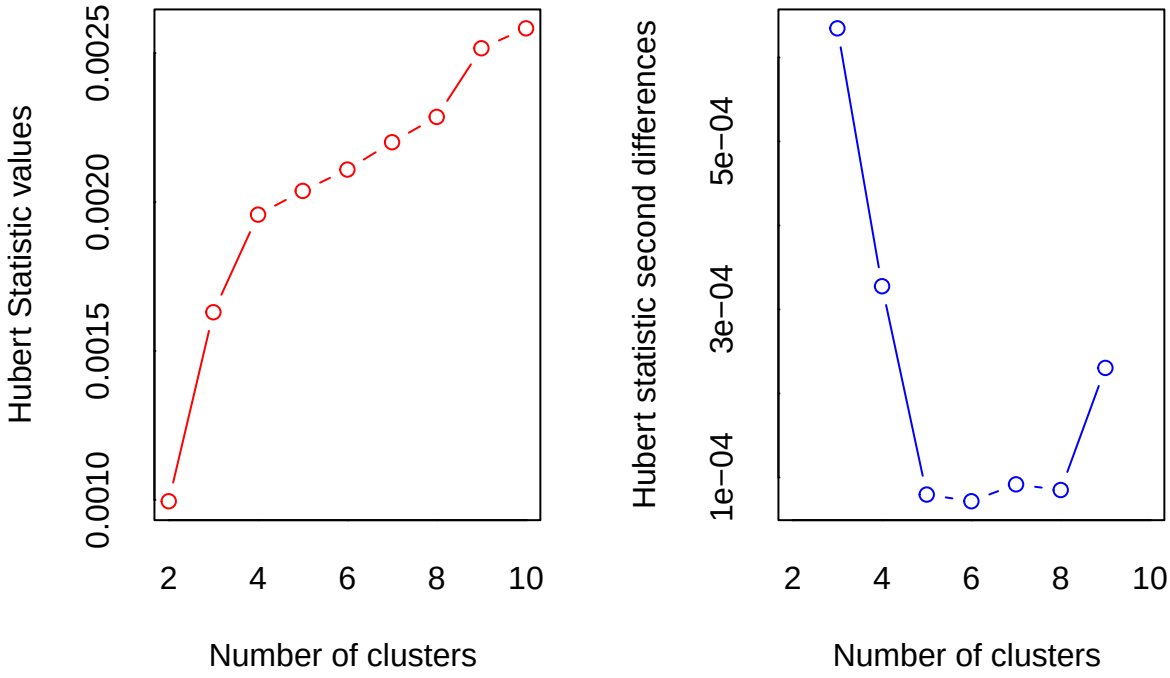
- **Centroid Calculation**: The centroid is the average position of all points in a cluster.
- **Efficiency**: Simple and computationally fast, suitable for large datasets.
- **Limitations**: Sensitive to outliers, assumes clusters are spherical and can handle only numerical data.

K-Means is best for well-separated, compact clusters but may struggle with irregular or noisy data.

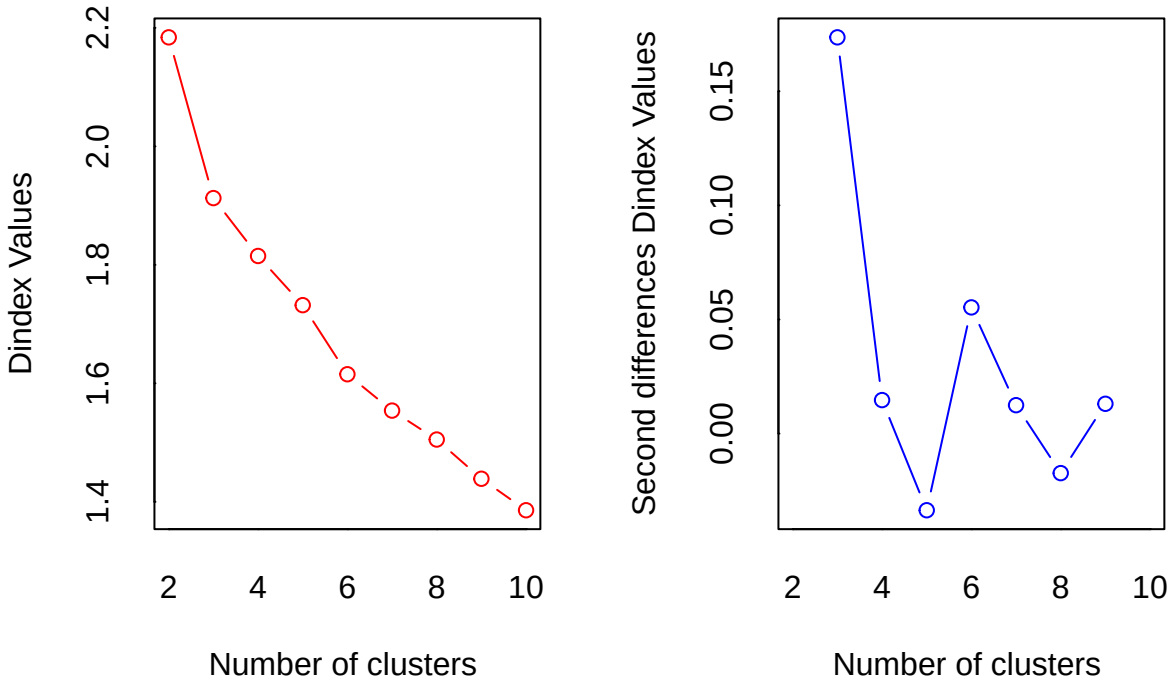
$$WSS = \sum_{k=1}^{K} \sum_{i=1}^{n} u_{ik} d^2_{Euc}(\mathbf{x_i}, \bar{\mathbf{x}}_\mathbf{i})$$

Optimal k for K-means

```r
nb <- NbClust(scaled_data, distance = "euclidean", min.nc = 2, max.nc = 10,
              method = "kmeans")
```




```
## *** : The Hubert index is a graphical method of determining the number of clusters.
##                 In the plot of Hubert index, we seek a significant knee that corresponds to a
##                 significant increase of the value of the measure i.e the significant peak in Hubert
##                 index second differences plot.
##
```

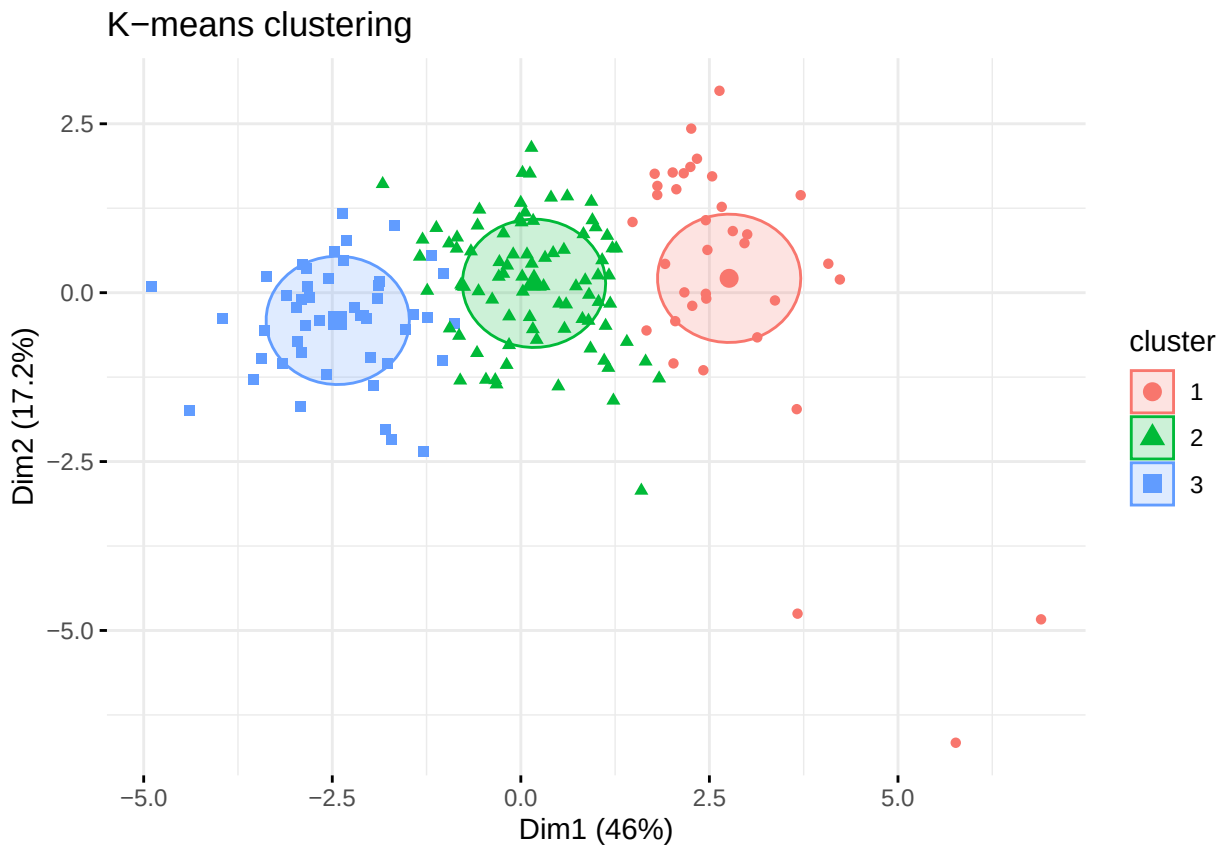



```
## *** : The D index is a graphical method of determining the number of clusters.
##                 In the plot of D index, we seek a significant knee (the significant peak in Dindex
##                 second differences plot) that corresponds to a significant increase of the value of
##                 the measure.
##
## *******************************************************************
## * Among all indices:
## * 5 proposed 2 as the best number of clusters
## * 7 proposed 3 as the best number of clusters
## * 1 proposed 4 as the best number of clusters
## * 4 proposed 5 as the best number of clusters
## * 1 proposed 7 as the best number of clusters
## * 2 proposed 9 as the best number of clusters
## * 3 proposed 10 as the best number of clusters
##
##                   ***** Conclusion *****
##
## * According to the majority rule, the best number of clusters is  3
##
##
## *******************************************************************
```

The **NbClust** function combines multiple indices to determine the ideal number of clusters. The majority of indices (7 out of the total) suggested that **3 clusters** are optimal. This conclusion is further supported by the **D Index**, which identifies a clear improvement in cluster separation at this point.

Thus, the analysis concludes that the most meaningful partitioning is achieved with **3 clusters**.

```r
km_res <- kmeans(scaled_data, iter.max = 10, centers = 3)
fviz_cluster(km_res, scaled_data, ellipse.type = "euclid", ggtheme = theme_minimal(), labelsize = 0,
             main = "K-means clustering")
```



The clustering analysis, performed in the original variable space, has been projected into the PCA space to facilitate interpretation. The results highlight three distinct clusters (red, green, and blue), with some degree of overlap between them, reflecting shared characteristics across countries.

Cluster 1 (Red) This cluster likely represents **wealthier countries**. These countries score high on PC1, which is strongly associated with high GDP per capita (`gdpp`), income, and life expectancy (`life_expec`), while exhibiting low child mortality (`child_mort`) and total fertility rates (`total_fer`). On PC2, these countries also show moderate-to-high values, reflecting significant participation in international trade (`imports` and `exports`). Collectively, these traits suggest socio-economically advanced nations with robust global economic integration.

Cluster 2 (Green) The second cluster appears to represent **transitional countries**. These countries are positioned between Clusters 1 and 3, with medium scores on PC1 and PC2. They likely exhibit intermediate socio-economic characteristics, such as moderate GDP, income, and life expectancy, along with average trade activity. This group serves as a bridge, sharing some features of both wealthier and poorer countries.

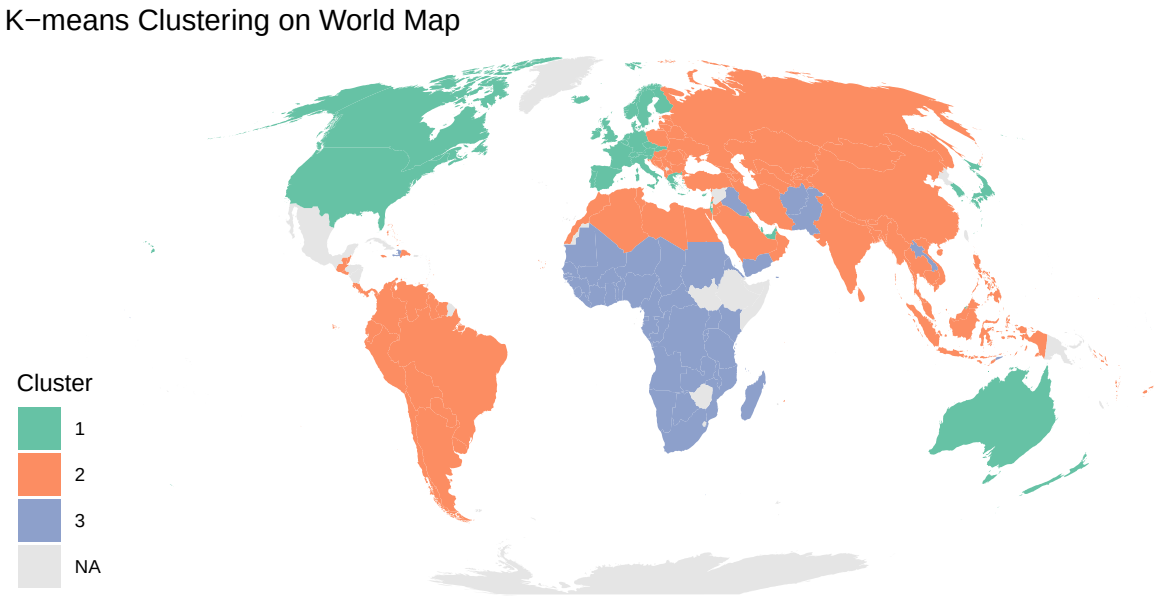
Cluster 3 (Blue) This cluster is indicative of **poorer countries**. These countries have negative scores on PC1, reflecting low GDP, income, and life expectancy, paired with high levels of child mortality and fertility. Their low scores on PC2 further suggest limited involvement in international trade, with relatively low values for imports and exports. Overall, this group comprises nations facing significant socio-economic challenges.

The visual overlap between the clusters demonstrates that while the groups are distinct, they are not entirely separate. This suggests a continuum of socio-economic and trade-related characteristics across countries, reflecting the complexity and gradual transitions in global development patterns.

By analyzing these clusters in the context of the PCA, we gain a nuanced understanding of how countries differ in terms of socio-economic status and global economic engagement.

Also in this case this representation is made only for visualization purpose, because the partition was not produced on the PCs space

```
cluster_data <- tibble(
  region  = names(km_res$cluster),
  cluster = as.factor(km_res$cluster)
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
                         "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                         "United Kingdom" = "UK",
                         "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(!long > 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")


ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "K-means Clustering on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



From this plot, we can clearly observe the classification of countries based on their attributes. The observations are divided into three main groups:

1. **Richer Countries (Green)**: This cluster includes nations such as the USA, Australia, and most European countries.
2. **Developing Countries (Orange)**: Countries in this category are in an intermediate stage of development, such as Russia, Brazil, and Egypt.
3. **Poorer Countries (Purple)**: This cluster predominantly consists of nations from Africa, along with some from the Middle East.

This classification highlights the socio-economic disparities between different regions of the world.

K-Medoids (PAM)

An alternative approach to clustering is the **K-Medoids method**, which differs from methods like K-Means in how cluster centroids are determined. Instead of using the mean of the points in each cluster as the centroid, K-Medoids selects the **most representative point** (also known as the medoid) within the cluster. This point is the one with the smallest average dissimilarity to all other points in the cluster.

By using an actual data point as the cluster representative, K-Medoids is **less sensitive to outliers** and can provide more robust clustering results, especially in cases where the data contains noise or non-numeric features.

$$WSS = \sum_{k=1}^{K} \sum_{i=1}^{n} u_{ik} d^2(\mathbf{x_i}, \mathbf{m_k})$$

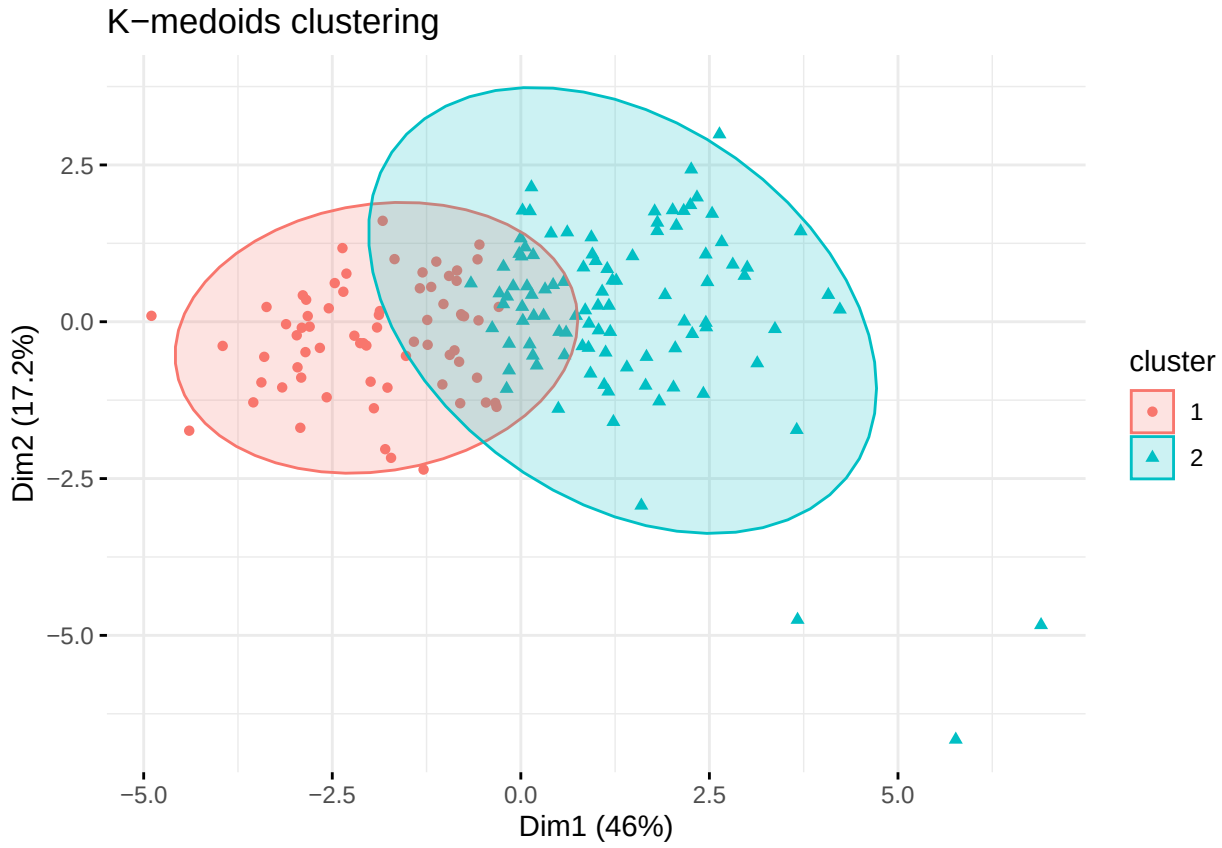
Optimal k for K-medoids

```
res <- fpc::pamk(data, krange = 2:10, criterion = "asw", scaling = T)
res$nc
```

```
## [1] 2
```

The **pamk** function find the ideal number of clusters. The function suggested that **2 clusters** are optimal.

```
kmed_res <- pam(scaled_data, k = 2, metric = "euclidean")
fviz_cluster(kmed_res, scaled_data, ellipse.type = "norm", ggtheme = theme_minimal(), labelsize = 0,
             main = "K-medoids clustering")
```

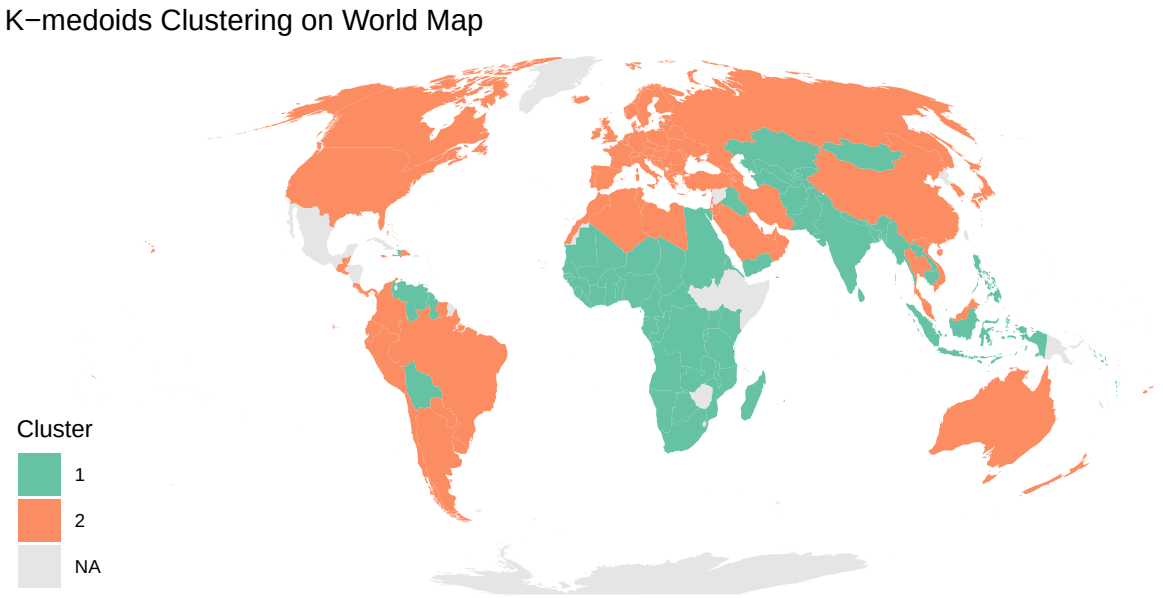


This clustering analysis, visualized in PCA space for interpretative clarity, highlights two clusters with differing assignments compared to K-Means.

K-Medoids provides a partition into two distinct clusters: the first cluster (red) appears to group mostly poorer countries along with some moderately developed ones, while the second cluster (light blue) encompasses the remaining countries, including wealthier nations and outliers.

```
cluster_data <- tibble(
  region  = names(kmed_res$cluster),
  cluster = as.factor(kmed_res$cluster)
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
```

```
                          "Congo, Rep." = "Republic of Congo",
                          "Cote d'Ivoire" = "Ivory Coast",
                          "Kyrgyz Republic" = "Kyrgyzstan",
                          "Lao" = "Laos",
                          "Macedonia, FYR" = "North Macedonia",
                          "Micronesia, Fed. Sts." = "Micronesia",
                          "Slovak Republic" = "Slovakia",
                          "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                          "United Kingdom" = "UK",
                          "United States" = "USA"))
world_map <- map_data("world") %>%
  filter(!long > 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")


ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "K-medoids Clustering on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



The classification using only two clusters seems somewhat imprecise. For instance, some very wealthy countries, such as Switzerland, are grouped in the same cluster as less wealthy nations, like Russia or countries in North Africa, despite having clearly different attribute values. In my opinion, two clusters might not be an optimal choice for interpretation purposes.

Comparing clustering algorithms

To compare the different methods used so far, I employed internal validation measures (compactness, separation, connectivity, Dunn index) and stability measures (average proportion of non-overlap, average distance, average distance between means, and figure of merit).

Internal:

```
intern <- clValid(scaled_data, nClust = 2:6,
                  clMethods = c("hierarchical", "kmeans", "pam"),
                  validation = "internal")

summary(intern)
```

```
##
## Clustering Methods:
##  hierarchical kmeans pam
##
## Cluster sizes:
##  2 3 4 5 6
##
## Validation Measures:
##                                 2       3       4       5       6
```

```
##
## hierarchical Connectivity   2.9290  7.8825 11.0794 14.0083 16.9373
##              Dunn           0.5457  0.3431  0.3284  0.3771  0.2914
##              Silhouette     0.6303  0.5620  0.4970  0.3858  0.2850
## kmeans       Connectivity   2.9290 39.2329 44.1865 42.5603 44.4325
##              Dunn           0.5457  0.0666  0.0838  0.0861  0.0972
##              Silhouette     0.6303  0.2859  0.3005  0.3044  0.3107
## pam          Connectivity  37.8583 48.0448 80.9861 82.9302 77.9345
##              Dunn           0.0666  0.0600  0.0633  0.0633  0.0633
##              Silhouette     0.2846  0.2810  0.2054  0.2203  0.2412
##
## Optimal Scores:
##
##              Score  Method       Clusters
## Connectivity 2.9290 hierarchical 2
## Dunn         0.5457 hierarchical 2
## Silhouette   0.6303 hierarchical 2
```

Stability:

```
stability <- clValid(scaled_data, nClust = 2:6,
                  clMethods = c("hierarchical", "kmeans", "pam"),
                  validation = "stability")

summary(stability)
```

```
##
## Clustering Methods:
##  hierarchical kmeans pam
##
## Cluster sizes:
##  2 3 4 5 6
##
## Validation Measures:
##                         2      3      4      5      6
##
## hierarchical APN  0.0039 0.0075 0.0088 0.0097 0.0615
##              AD   3.6786 3.5207 3.4670 3.4064 3.3719
##              ADM  0.0402 0.0869 0.1072 0.1019 0.2769
##              FOM  0.9983 0.9553 0.9519 0.9375 0.9200
## kmeans       APN  0.0532 0.1457 0.2549 0.0804 0.2572
##              AD   3.6779 3.3239 3.0551 2.5760 2.5219
##              ADM  0.1834 1.0672 0.9868 0.3352 0.6675
##              FOM  0.9947 0.9275 0.8903 0.7574 0.7390
## pam          APN  0.0685 0.1172 0.1507 0.1835 0.2991
##              AD   3.1650 2.8395 2.6782 2.5528 2.5126
##              ADM  0.2476 0.4276 0.4409 0.4950 0.7425
##              FOM  0.8550 0.8084 0.7822 0.7606 0.7457
##
## Optimal Scores:
##
##     Score  Method       Clusters
## APN 0.0039 hierarchical 2
## AD  2.5126 pam          6
## ADM 0.0402 hierarchical 2
```

```
## FOM 0.7390 kmeans       6
```

Hierarchical, **kmeans**, and **pam** clustering methods were evaluated across cluster sizes (2 to 6). **Hierarchical** clustering with 2 clusters consistently performed best, showing superior internal validation with the lowest Connectivity, highest Dunn Index, and best Silhouette scores, as well as the strongest stability with the lowest APN and ADM scores.

Kmeans excelled in stability with the best FOM score at 6 clusters, while pam showed moderate stability, particularly in AD scores for 6 clusters.

Overall, **hierarchical** clustering with 2 clusters was the most robust, though **kmeans** with 6 clusters proved a viable alternative in specific cases.

Overall, the indices largely suggest that the optimal number of clusters is 2. However, in my opinion, this may not be the best choice in terms of interpretability and overall partitioning. I observed that two clusters are insufficient to adequately distinguish countries with varying socio-economic indices.

For this dataset, a soft clustering technique (like *Fuzzy clustering* or *Model based clustering*) might be more appropriate, as it allows for more flexible assignments by not rigidly grouping observations into a single cluster.

Fuzzy Clustering

Fuzzy clustering is a technique similar to crisp clustering, with the key difference lying in how observations are assigned to clusters. In fuzzy clustering, the allocation matrix can take values $u \in [0, 1]$, meaning that observations are not assigned exclusively to a single cluster. Instead, each observation is associated with a *degree of membership* to one or more clusters.

This approach is particularly advantageous in scenarios where clusters overlap significantly, and there is inherent uncertainty in classification. By allowing for partial memberships, fuzzy clustering provides a more nuanced representation of the data structure, making it especially useful in complex datasets where rigid boundaries between clusters are unrealistic or less meaningful.

Fuzzy K-Means Clustering

This algorithm functions like classical K-Means but differs in the allocation matrix, which represents the degree of membership of each observation to the clusters, enabling more flexible classifications in overlapping or uncertain cases.

```
fkm_res <- FKM(X = scaled_data, stand = 1, RS = 10, seed = 333)
```

```
## The default value k=2:6 has been set
## The default index SIL.F has been set
## The default value alpha=1 has been set for computing SIL.F
```

```
fkm_res$k
```

```
## Number of clusters
##                  2
```

```
fkm_res$criterion
```

```
## SIL.F k=2 SIL.F k=3 SIL.F k=4 SIL.F k=5 SIL.F k=6
## 0.5108947 0.4523286 0.3538631 0.1952534 0.1395022
```

```
fkm2_res <- FKM(X = scaled_data, index = "XB", stand = 1, RS = 10, seed = 333)
```

```
## The default value k=2:6 has been set
```

```
fkm2_res$k
```

```
## Number of clusters
##                   3
```

```
fkm2_res$criterion
```

```
##    XB k=2    XB k=3    XB k=4    XB k=5    XB k=6
## 0.5647240 0.5317398 1.4549003 3.6655895 5.1084981
```

Based on the chosen criterion, k = 3 could represent a better compromise, even though the suggested optimal value is k = 2.

In my opinion, selecting k = 2 does not provide a well-representative partition of the data in terms of interpretability, as observed in the k-medoids clustering results.

```
fkm3_res <- FKM(X = data, stand = 1, k = 3, RS = 10, seed = 333)
head(fkm3_res$clus)
```

```
##                      Cluster Membership degree
## Afghanistan                3         0.8457475
## Albania                    2         0.8115113
## Algeria                    2         0.6676411
## Angola                     3         0.6746080
## Antigua and Barbuda        2         0.7356444
## Argentina                  2         0.4864439
```

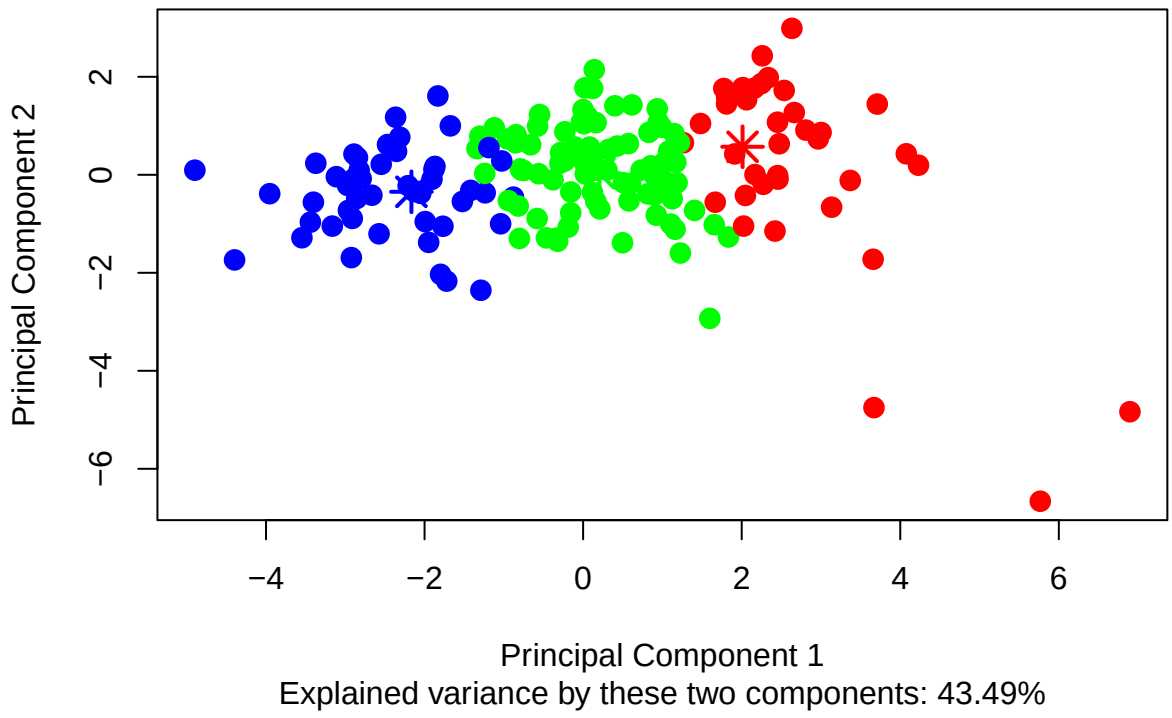
```
Hraw(fkm3_res$X, fkm3_res$H)
```

```
##        child_mort  exports   health  imports    income inflation life_expec
## Clus 1   9.764841 46.90486 8.873436 45.06974 35223.913  3.690840   78.47802
## Clus 2  24.430167 42.11457 6.195869 48.88110 13587.237  7.133917   72.53190
## Clus 3  85.240064 29.76488 6.044918 41.91307  4768.862 10.029266   60.83353
##        total_fer      gdpp
## Clus 1  1.918182 34027.142
## Clus 2  2.379597  7718.508
## Clus 3  4.827478  2672.754
```

This table is particularly insightful for understanding the differences in the centroids of each variable across clusters (data shown on the original scale). For instance, child mortality varies significantly: it is very low in Cluster 1, intermediate in Cluster 2, and exceptionally high in Cluster 3. Similarly, income and GDP per capita display clear distinctions, with Cluster 1 representing wealthier countries, Cluster 3 the poorer, and Cluster 2 in between.

Other variables, such as inflation and total fertility, also highlight stark differences. Inflation is lowest in Cluster 1, moderate in Cluster 2, and highest in Cluster 3. Total fertility follows a similar pattern, being lowest in Cluster 1 and peaking in Cluster 3. These variations provide a clear characterization of the clusters, distinguishing wealthier, more developed countries from less developed ones.

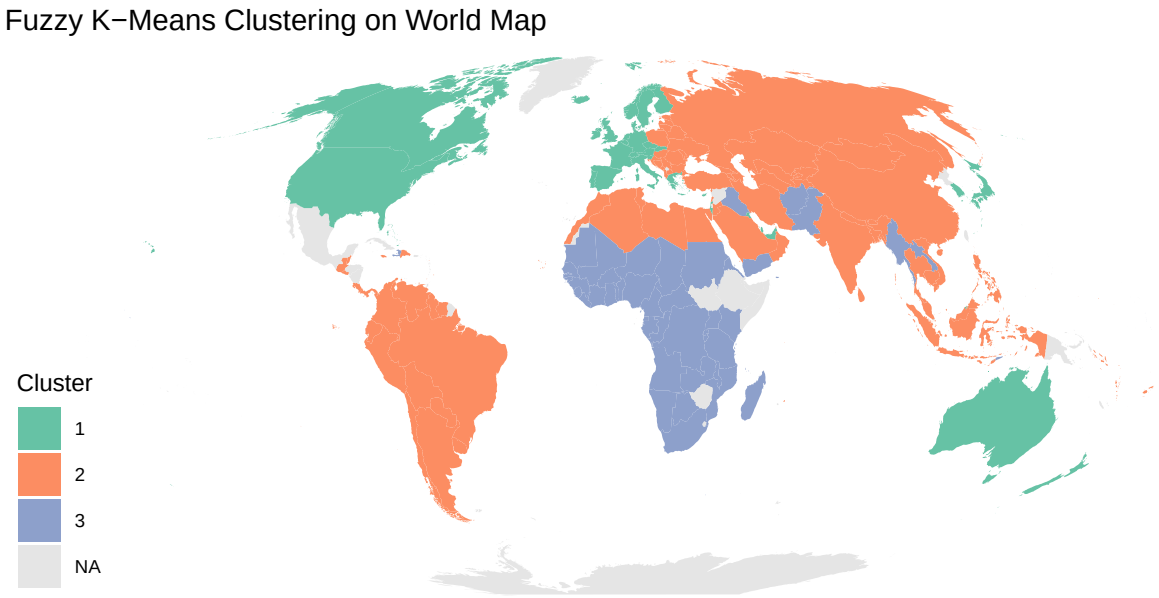
```
plot.fclust(x = fkm3_res, pca = T)
```



There are a representation of the classification made with the fuzzy K-means clustering

```
cluster_data <- tibble(
  region  = rownames(fkm3_res$clus),
  cluster = as.factor(fkm3_res$clus[, "Cluster"])
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
                         "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                         "United Kingdom" = "UK",
                         "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(long <= 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")
```

```
ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Fuzzy K-Means Clustering on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



The classification appears similar to standard k-means; however, it's important to note that, in this case, we account for degrees of membership.

```
head(sort(fkm3_res$clus[, 'Membership degree'], decreasing = F), 10)
```

```
## Micronesia, Fed. Sts.                 Malta             Singapore
##             0.3766012             0.3934596             0.3957082
##               Nigeria             Venezuela       Solomon Islands
##             0.3993999             0.4075664             0.4189087
##               Myanmar            Luxembourg            Costa Rica
##             0.4249462             0.4380420             0.4399162
##              Mongolia
##             0.4399465
```

These classifications exhibit a higher level of uncertainty. We can observe that most of the observations correspond to smaller countries.

Gustafson-Kessel Fuzzy K-Means Clustering

This version differs from the previous one for the choice of the distance metric, which is the *Mahalanobis distance*:

$$d_M^2(\mathbf{x_i}, \bar{\mathbf{x}}_\mathbf{i}) = (\mathbf{x_i} - \bar{\mathbf{x}}_\mathbf{i})' \ \mathbf{M}_k \ (\mathbf{x_i} - \bar{\mathbf{x}}_\mathbf{i})$$

This implementation enables the algorithm to identify more flexible cluster shapes, which are not necessarily spherical.

```
fkm2gk_res <- FKM.gk(X = scaled_data, index = "XB", stand = 1, RS = 10, seed = 333)
```

```
## The default value k=2:6 has been set
```

```
fkm2gk_res$k
```

```
## Number of clusters
##                   2
```

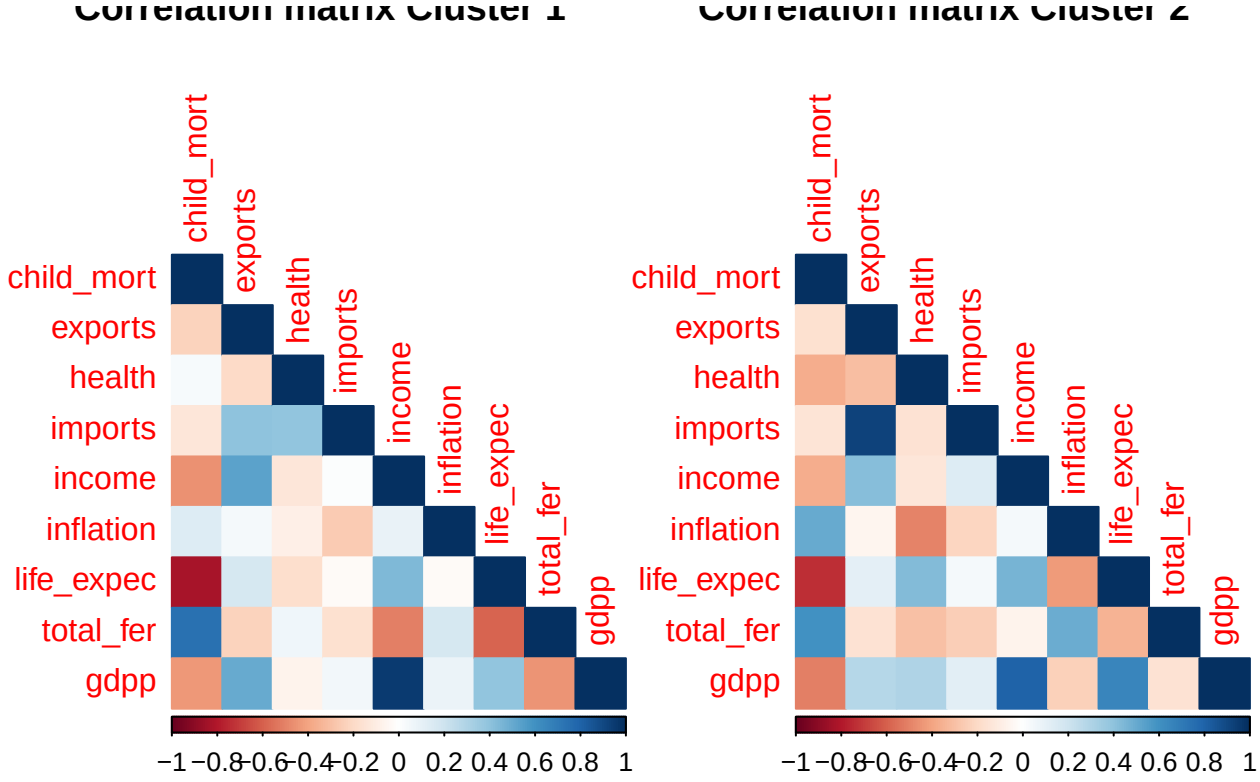
```
fkm2gk_res <- FKM.gk(X = scaled_data, index = "SIL.F", stand = 1, RS = 10, seed = 333)
```

```
## The default value k=2:6 has been set
## The default value alpha=1 has been set for computing SIL.F
```

```
fkm2gk_res$k
```

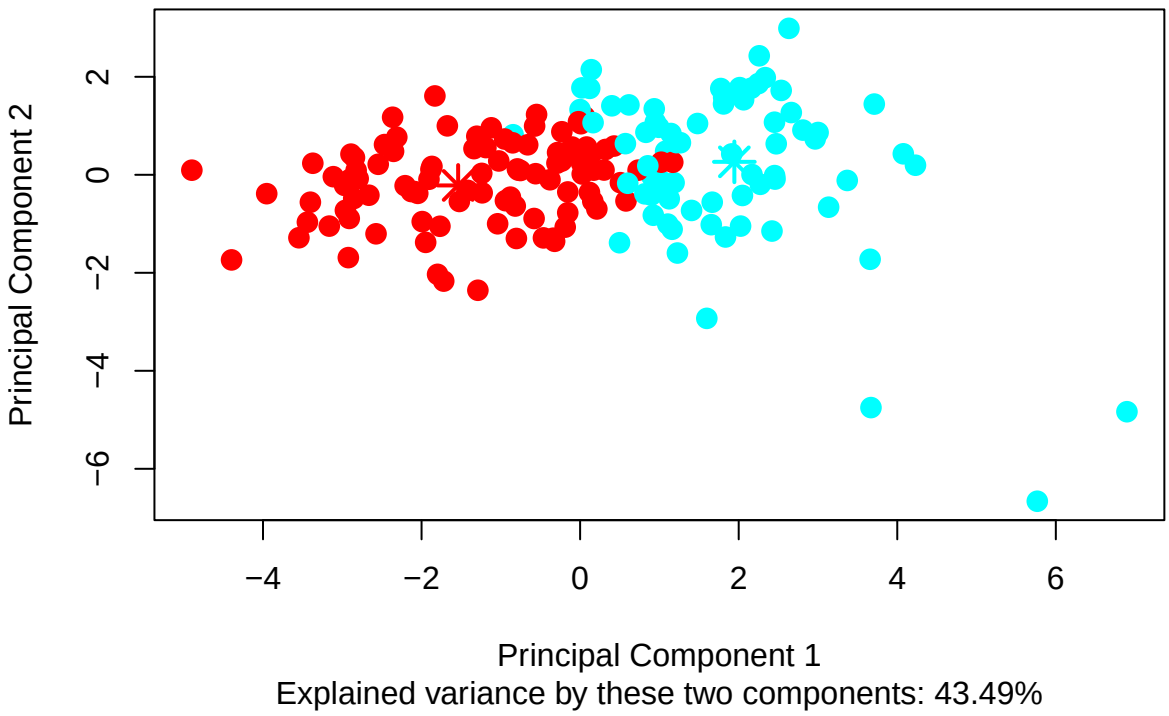
```
## Number of clusters
##                   2
```

```
gk_res <- FKM.gk(data, stand = 1,
                 index = "XB",
                 RS = 10, seed = 333, k = 2)

par(mfrow = c(1, 2))
corrplot(corpcor::decompose.cov(round(gk_res$F[, , 1], 3))$r, type = "lower", method = "color",
         main = "Correlation matrix Cluster 1")
corrplot(corpcor::decompose.cov(round(gk_res$F[, , 2], 3))$r, type = "lower", method = "color",
         main = "Correlation matrix Cluster 2")
```



Based on the criterion I chose a k = 2. The shape of the plot will be this:

```
plot.fclust(x = gk_res, pca = T)
```



The plot illustrates the clustering results from the fuzzy Gustafson-Kessel algorithm, showing significant overlap between clusters. This highlights the fuzzy nature of the algorithm, where data points can belong to multiple clusters with varying degrees of membership, reflecting the complexity and lack of clear separation in the data structure.

```
head(sort(gk_res$clus[, 'Membership degree'], decreasing = F), 10)
```

```
##      Venezuela     Kazakhstan Macedonia, FYR        Georgia        Lebanon
##      0.5093478      0.5105095      0.5186953      0.5199959      0.5273310
##       Thailand          Egypt         Panama      Mauritius         Russia
##      0.5294379      0.5294978      0.5347099      0.5357253      0.5377243
```

Here, the degree of membership appears to start at higher levels compared to the previous partition.

Fuzzy K-Means with Noise Cluster

In this algorithm, an additional cluster is introduced to capture all the outliers. The formula to optimize is:

$$WSS = \sum_{k=1}^{K}\sum_{i=1}^{n} u_{ik}^{m} d_{Euc}^{2}(\mathbf{x_i}, \bar{\mathbf{x}}_\mathbf{i}) + \sum_{i=1}^{n} \delta^2 (1 - \sum_{k=1}^{K} u_{ik})^m$$

Here, the second term represents the allocation values for the noise cluster, weighted by the squared distance of each unit to the noise cluster, δ^2. This component ensures that the noise cluster accounts for outliers effectively.

```
fkmn_res <- FKM.noise(data, stand = 1, RS = 10, seed = 333, index = "XB")
```

```
## The default value k=2:6 has been set
```

```
fkmn_res$criterion
```

```
##    XB k=2    XB k=3    XB k=4    XB k=5    XB k=6
## 0.2504491 0.3112420 0.8459011 2.3733833 5.7597183
```

```
fkmn_res$k
```

```
## Number of clusters
##                   2
```

```
fkmnSIL_res <- FKM.noise(data, stand = 1, RS = 10, seed = 333, index = "SIL.F")
```

```
## The default value k=2:6 has been set
## The default value alpha=1 has been set for computing SIL.F
```

```
fkmnSIL_res$criterion
```

```
## SIL.F k=2 SIL.F k=3 SIL.F k=4 SIL.F k=5 SIL.F k=6
## 0.5027511 0.4419682 0.3442047 0.1578417 0.1328655
```

```
fkmnSIL_res$k
```

```
## Number of clusters
##                   2
```

The optimal number of cluster for this method is 2, in witch one of them is expected to be the noisy cluster

```
Hraw(fkmn_res$X, fkmn_res$H)
```

```
##        child_mort  exports   health  imports    income inflation life_expec
## Clus 1   66.69388 31.08520 5.848128 41.88086  6177.936  9.215969   63.98210
## Clus 2   15.88069 42.66242 7.195112 47.41957 19077.291  4.863195   75.13539
##        total_fer      gdpp
## Clus 1  4.182868  3539.443
## Clus 2  2.050746 13889.209
```

There are significant differences between the centroids of the two clusters escpecially for the variables: child mortality, income, inflation, total_fert and gdpp

```
head(sort(fkmn_res$clus[, 'Membership degree'], decreasing = F), 10)
```

```
##       Nigeria    Luxembourg     Singapore         Qatar         Malta
##     0.1035323     0.1116747     0.1167448     0.1694632     0.1967470
##         Haiti United States        Norway     Venezuela   Switzerland
##     0.2477910     0.2581701     0.2694830     0.2780613     0.3109570
```

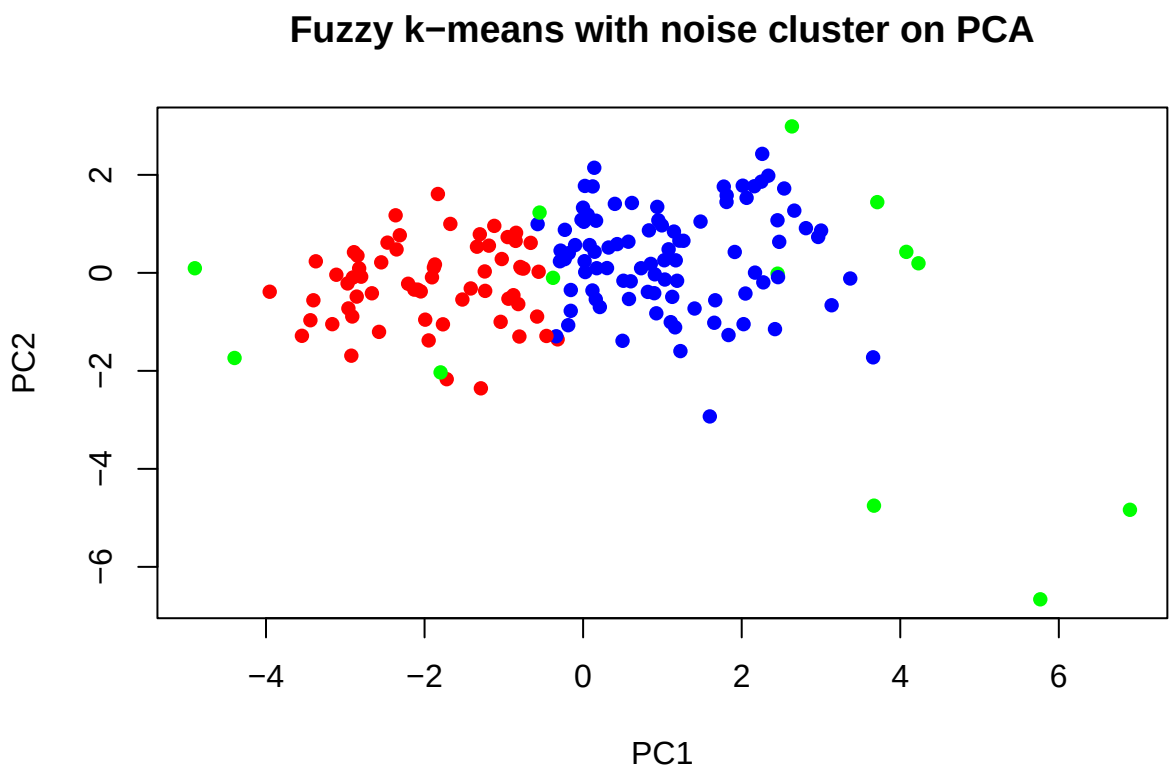
The results from the Fuzzy K-Means with Noise Cluster show that several observations, such as Nigeria, Luxembourg, and Singapore, exhibit low membership degrees in the primary clusters (below 0.5). This indicates a significant association with the noise cluster, suggesting these countries do not fit well into the main partition structure, likely due to unique or outlying characteristics.

```
noise <- as.numeric(fkmn_res$clus[, 2])
idxs <- which(noise <= (1/3))
fkmn_res$clus[idxs] = 3

head(fkmn_res$clus)
```

```
##                     Cluster Membership degree
## Afghanistan               1         0.6637825
## Albania                   2         0.8138319
## Algeria                   2         0.4830960
```

```
## Angola                    1         0.4927589
## Antigua and Barbuda       2         0.8861046
## Argentina                 2         0.4939313
```

```r
pca_data <- as.data.frame(pca_data)

cluster_colors <- c("red", "blue", "green")
colors <- cluster_colors[fkmn_res$clus[, 1]]

plot(pca_data$PC1, pca_data$PC2,
     xlab = "PC1",
     ylab = "PC2",
     main = "Fuzzy k-means with noise cluster on PCA",
     pch = 16, col = colors)
```



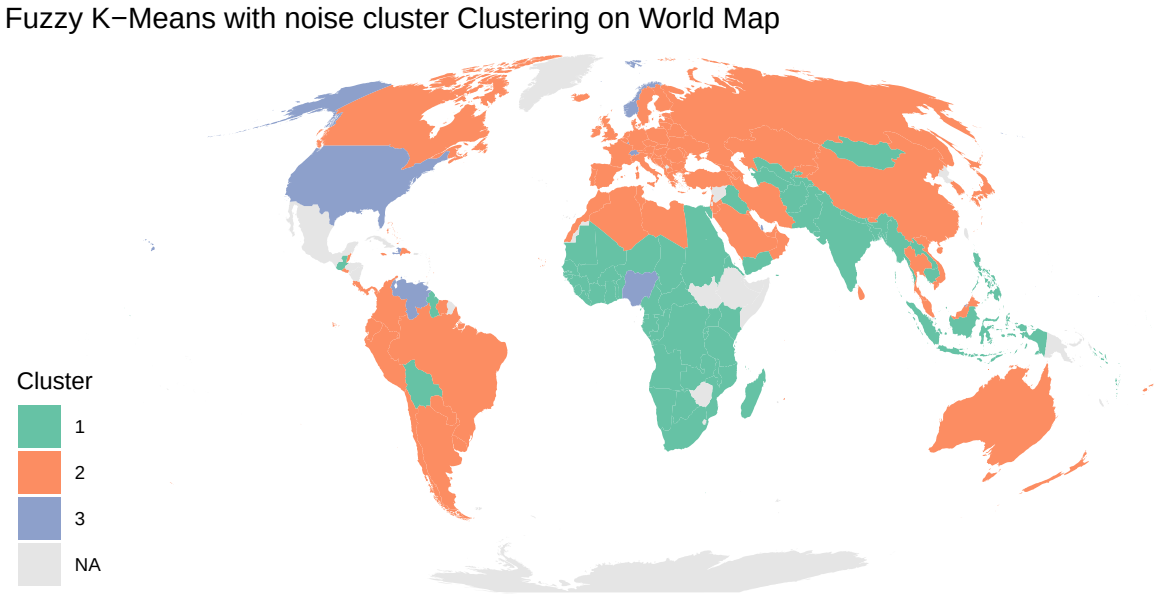
Through this plot, we can clearly observe the purpose of this variant of the k-means algorithm. The Fuzzy K-means with noise cluster algorithm assigns to the noise cluster all observations that are difficult to classify or, more generally, where membership is highly uncertain. In this case, we can see that all outliers are included in the noise cluster (green cluster). Additionally, some points positioned between the red and blue clusters are also classified as part of the noise cluster.

```r
fkmn_res$clus[, 1][fkmn_res$clus[,1] == 3]
```

```
##                Brunei                 Haiti               Lesotho
##                     3                     3                     3
##            Luxembourg                 Malta Micronesia, Fed. Sts.
##                     3                     3                     3
##               Nigeria                Norway                 Qatar
```

```
##                     3                     3                     3
##             Singapore           Switzerland         United States
##                     3                     3                     3
##             Venezuela
##                     3
```

```
cluster_data <- tibble(
  region  = rownames(fkmn_res$clus),
  cluster = as.factor(fkmn_res$clus[, "Cluster"])
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
                         "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                         "United Kingdom" = "UK",
                         "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(long <= 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")

ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Fuzzy K-Means with noise cluster Clustering on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



From this plot, we can observe that certain countries, such as the USA, Nigeria, and Norway, are difficult to classify based on the given data.

Fuzzy K-Medoids Clustering

Also in this case, the algorithm closely resembles the classical Fuzzy K-Medoids. However, it differs in the allocation matrix, where $u_{ik} \in [0,1],\ i \in \{1,...,n\}$ and $k \in \{1,...,m\}$

```
fkmed_res <- FKM.med(X = scaled_data, stand = 1, RS = 10, seed = 333, index = 'SIL.F')
```

```
## The default value k=2:6 has been set
## The default value alpha=1 has been set for computing SIL.F
```

```
fkmed_res$k
```

```
## Number of clusters
##                  2
```

```
fkmed_res$criterion
```

```
## SIL.F k=2 SIL.F k=3 SIL.F k=4 SIL.F k=5 SIL.F k=6
## 0.3900855 0.2810285 0.2278158 0.2132317 0.2359423
```

```
fkmed2_res <- FKM.med(X = scaled_data, index = "XB", stand = 1, RS = 10, seed = 333)
```

```
## The default value k=2:6 has been set
```

```
fkmed2_res$k
```

```
## Number of clusters
```

```
##                  6
```

```
fkmed2_res$criterion
```

```
##   XB k=2   XB k=3   XB k=4   XB k=5   XB k=6
## 4.047208 6.150297 4.583934 3.892761 3.423222
```

In this case the Xi-Bin index suggest an optimal K equale to 6

```
fkmed3_res <- FKM(X = data, stand = 1, k = 6, RS = 10, seed = 333)
head(fkmed3_res$clus)
```

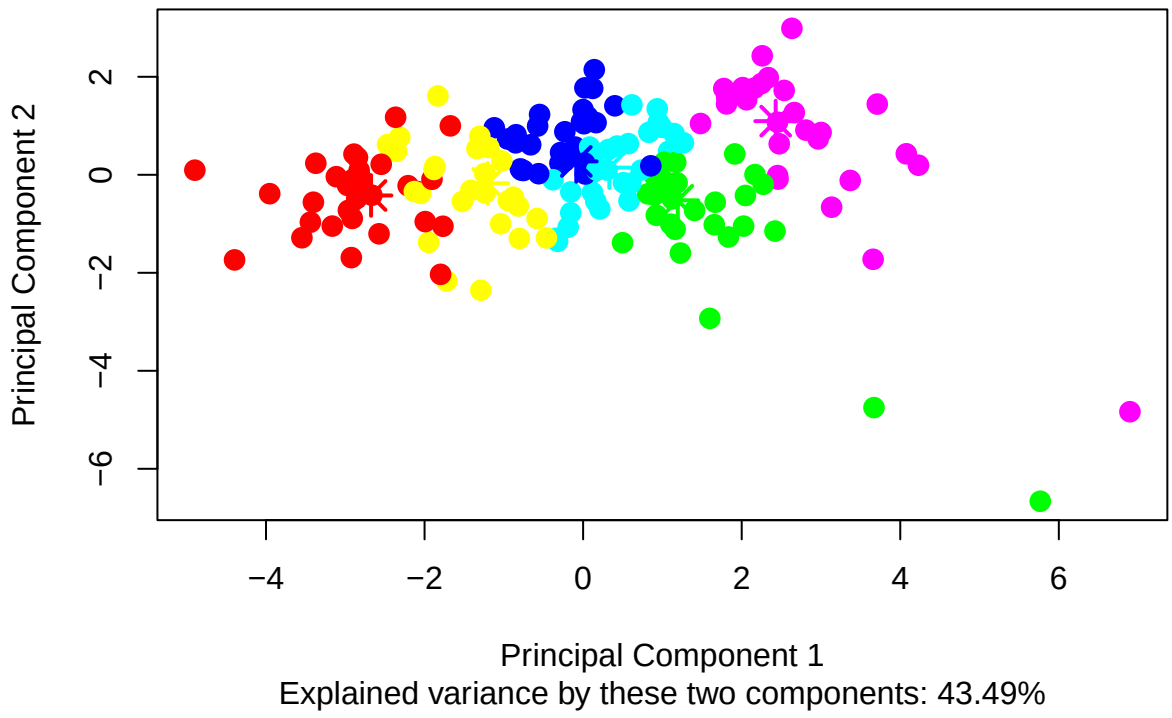
```
##                     Cluster Membership degree
## Afghanistan               1         0.7226798
## Albania                   4         0.4370386
## Algeria                   5         0.3922532
## Angola                    1         0.3884110
## Antigua and Barbuda       3         0.4721616
## Argentina                 5         0.2882233
```

```
Hraw(fkmed3_res$X, fkmed3_res$H)
```

```
##        child_mort  exports   health  imports    income inflation life_expec
## Clus 1 100.039494 27.14485 6.176585 40.00321  2930.847  8.890146   58.09357
## Clus 2  57.403780 34.41867 5.711170 44.24747  8195.439 10.971305   65.73101
## Clus 3  14.517640 60.00650 6.838479 61.84130 21713.987  4.195526   74.74921
## Clus 4  23.148110 40.99956 6.607439 47.92732 15099.472  6.866165   73.04286
## Clus 5  29.357710 36.49577 6.017359 41.98476 13339.019  8.946452   71.86798
## Clus 6   5.926735 40.62321 9.997347 38.13294 39647.890  2.256127   80.31424
##        total_fer      gdpp
## Clus 1  5.392333  1583.186
## Clus 2  3.708577  4582.275
## Clus 3  1.964251 14229.760
## Clus 4  2.325339  9215.363
## Clus 5  2.582250  7617.037
## Clus 6  1.826258 43226.530
```

Since we have already observed that k=2 does not provide meaningful interpretability, I have decided to choose k=6. I expect this partition to result in clusters that are more challenging to interpret, with smaller clusters characterized by higher uncertainty.

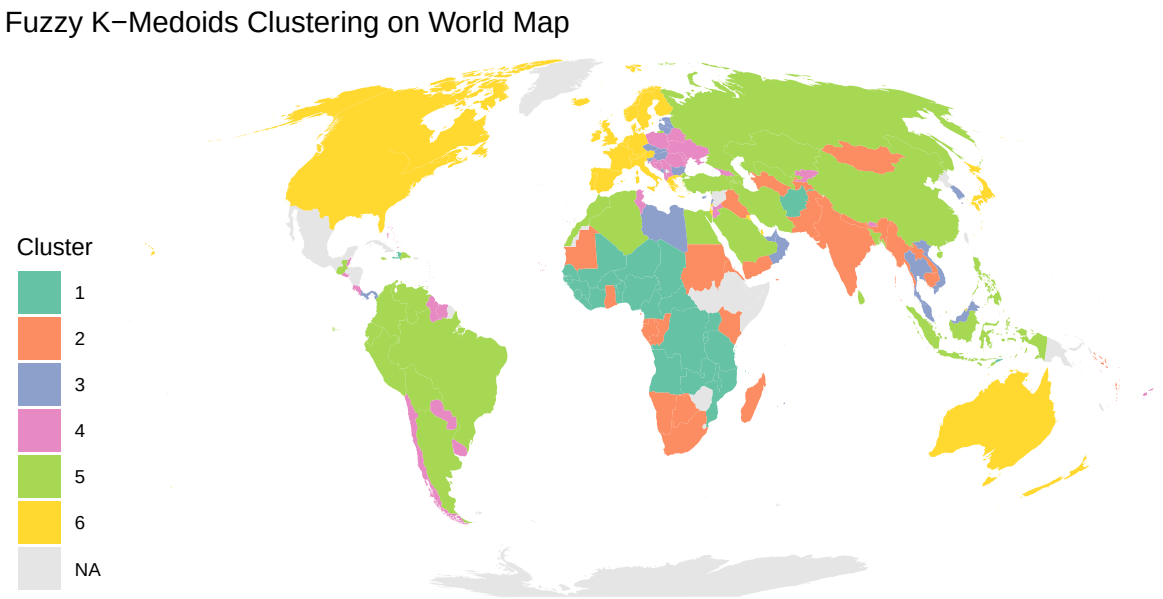
```
plot.fclust(x = fkmed3_res, pca = T)
```



We can observe numerous small clusters positioned very close to each other. This does not appear to be a well-defined partition overall.

```
cluster_data <- tibble(
  region  = rownames(fkmed3_res$clus),
  cluster = as.factor(fkmed3_res$clus[, "Cluster"])
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
                         "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                         "United Kingdom" = "UK",
                         "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(long <= 180)

merged_map <- world_map %>%
```

```
  left_join(cluster_data, by = "region")

ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Fuzzy K-Medoids Clustering on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



Some clusters appear relatively uniform, such as the yellow one (when considering its other members). However, the others don't seem significantly different from each other. In my opinion, two or more clusters might need to be merged to create more meaningful partitions.

```
head(sort(fkmed3_res$clus[, 'Membership degree'], decreasing = F), 10)
```

```
## Micronesia, Fed. Sts.              Nigeria            Singapore
##             0.1956969            0.1979543            0.2218935
##             Venezuela           Luxembourg             Kiribati
##             0.2226430            0.2228281            0.2246311
##                Brunei               Kuwait                 Oman
##             0.2250302            0.2335908            0.2341097
##                Guyana
##             0.2341444
```

We can observe from the output above that, due to the higher number of clusters, the uncertainty in the classification is greater compared to the partitions observed with a lower number of clusters.

Fuzzy K-Medoids with noise Cluster

Lastly, I applied a more robust version of this analysis by using the K-medoids variation of the previous algorithm. The core methodology remains the same, but the key difference lies in the choice of the centroid. In this approach, the most representative point for each cluster is selected from the actual observations, rather than being calculated as an average. This makes the clustering process less sensitive to outliers and provides a more realistic representation of the cluster centers.

```
fkmedn_res <- FKM.med.noise(data, stand = 1, RS = 10, seed = 333, index = "XB")
```

```
## The default value k=2:6 has been set
```

```
fkmedn_res$criterion
```

```
##   XB k=2   XB k=3   XB k=4   XB k=5   XB k=6
## 1.709786 3.219387 2.409855 2.714633 4.374760
```

```
fkmedn_res$k
```

```
## Number of clusters
##                  2
```

```
fkmedn_res2 <- FKM.med.noise(data, stand = 1, RS = 10, seed = 333, index = "SIL.F")
```

```
## The default value k=2:6 has been set
## The default value alpha=1 has been set for computing SIL.F
```

```
fkmedn_res2$criterion
```

```
##   SIL.F k=2   SIL.F k=3   SIL.F k=4   SIL.F k=5   SIL.F k=6
##  0.37460581  0.27052078 -0.07623072  0.14251598  0.08329312
```

```
fkmedn_res2$k
```

```
## Number of clusters
##                  2
```

```
Hraw(fkmedn_res$X, fkmn_res$H)
```

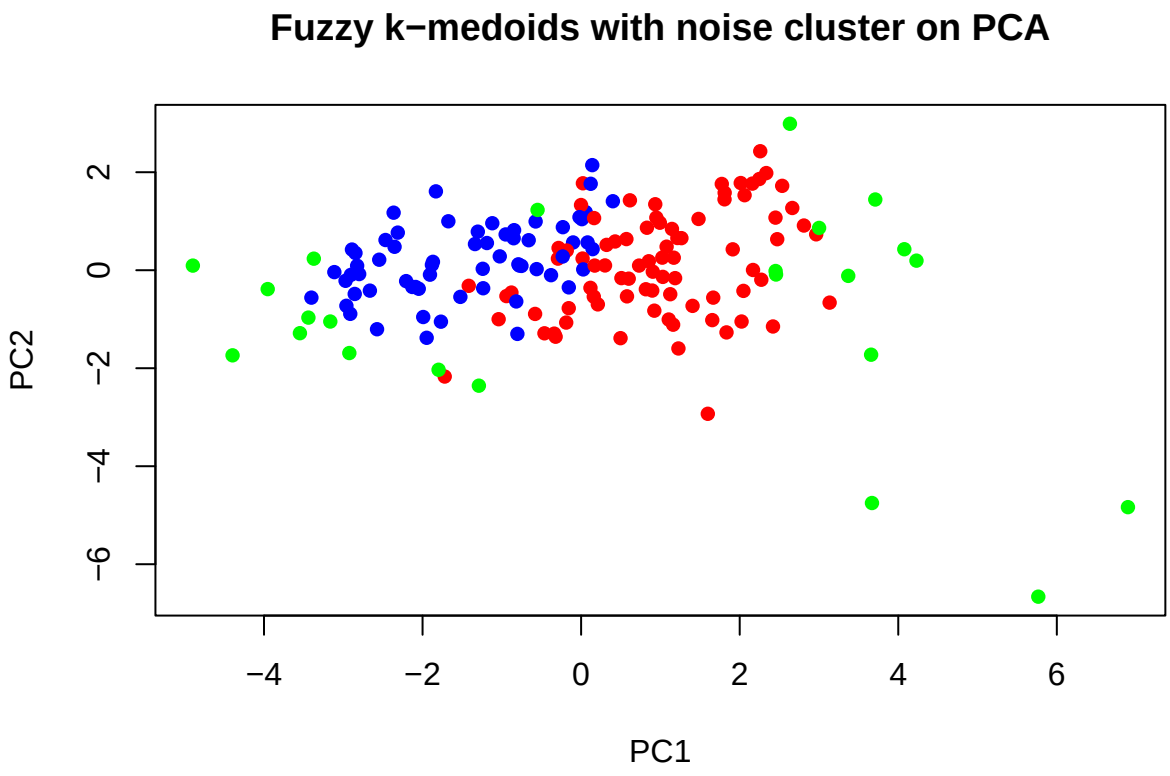
```
##        child_mort  exports   health  imports    income inflation life_expec
## Clus 1   66.69388 31.08520 5.848128 41.88086  6177.936  9.215969   63.98210
## Clus 2   15.88069 42.66242 7.195112 47.41957 19077.291  4.863195   75.13539
##        total_fer      gdpp
## Clus 1  4.182868  3539.443
## Clus 2  2.050746 13889.209
```

```
noise <- as.numeric(fkmedn_res$clus[, 2])
idxs <- which(noise <= (1/3))
fkmedn_res$clus[idxs] = 3

head(fkmedn_res$clus)
```

```
##                     Cluster Membership degree
## Afghanistan               2         0.5302120
## Albania                   1         0.5299334
## Algeria                   1         0.5114922
## Angola                    3         0.2651800
```

```
## Antigua and Barbuda       1         0.7519923
## Argentina                 1         0.4614877
```

```r
cluster_colors <- c("red", "blue", "green")
colors <- cluster_colors[fkmedn_res$clus[, 1]]

plot(pca_data$PC1, pca_data$PC2,
     xlab = "PC1",
     ylab = "PC2",
     main = "Fuzzy k-medoids with noise cluster on PCA",
     pch = 16, col = colors)
```



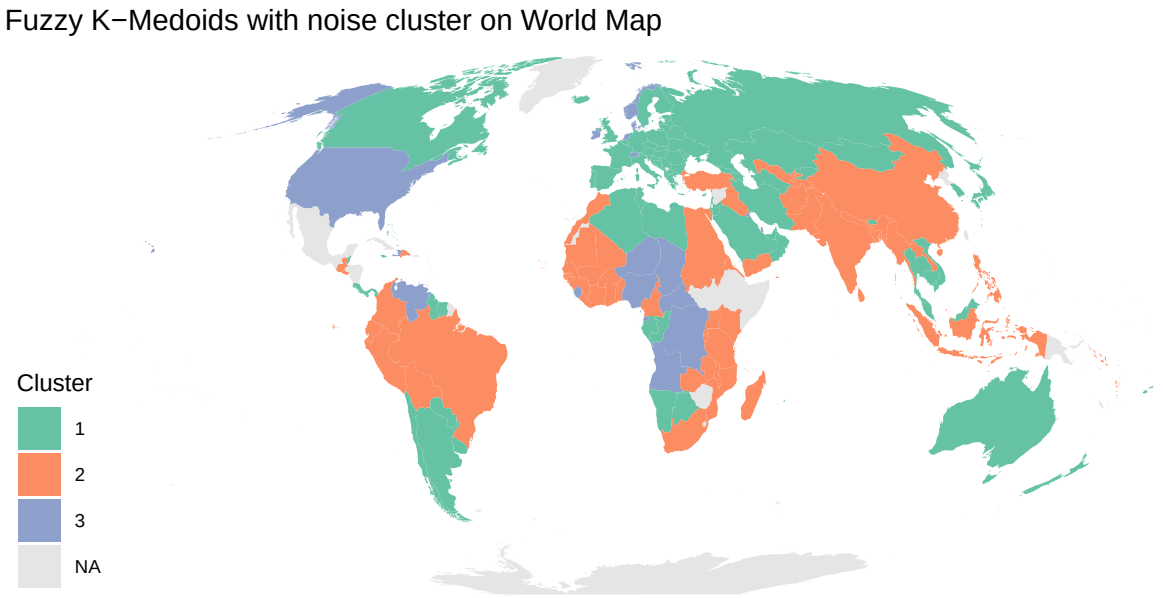
The data show greater overlap, which translates into increased uncertainty in classifying observations that fall between the two clusters. This overlap highlights the fuzzy boundaries and complexity of the dataset, making it more challenging to distinctly separate the groups. Such uncertainty emphasizes the importance of considering partial memberships when interpreting the clustering results.

```r
head(sort(fkmedn_res$clus[, 'Membership degree'], decreasing = F), 10)
```

```
##       Nigeria    Luxembourg     Singapore         Qatar         Malta
##    0.01318574    0.01766033    0.02045058    0.04810494    0.07282620
##         Haiti United States        Norway   Switzerland       Lesotho
##    0.07291074    0.12420403    0.12901442    0.16845737    0.19713348
```

We can observe very low membership values for some observations, such as Nigeria with a degree of membership of 0.013, which could indicate association with the outlier cluster.

```r
cluster_data <- tibble(
  region  = rownames(fkmedn_res$clus),
  cluster = as.factor(fkmedn_res$clus[, "Cluster"])
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
                         "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                         "United Kingdom" = "UK",
                         "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(long <= 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")

ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Fuzzy K-Medoids with noise cluster on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



Model Based Clustering

This approach involves the use of probabilistic models to better understand the assignment of observations to the appropriate clusters. By adopting a probabilistic perspective, we gain more insight into the likelihood of each observation belonging to a particular cluster, which is especially useful in cases of overlapping data or inherent uncertainty in cluster boundaries.

Similar to fuzzy clustering, this method employs a *soft assignment* of observations, meaning that data points are not strictly assigned to a single cluster but instead share affiliations with multiple clusters. However, unlike fuzzy clustering, where we use degrees of membership, here we can interpret these affiliations as **probabilities**. This allows for a more rigorous statistical understanding of the clustering process.

The probabilistic clustering models the overall distribution of the data as a mixture of distributions, where the probability density function $p(\mathbf{x})$ is expressed as (parametric formulation):

$$p(\mathbf{x};\psi) = \sum_{k=1}^{K} \pi_k f_k(\mathbf{x};\theta_k)$$

This formula reflects that the probability density at any point $\mathbf{x}$ is a weighted sum of the individual cluster distributions $f_k(\mathbf{x})$ with weights given by the prior probabilities $P(k)$.

This probabilistic framework not only enhances interpretability, but also provides a better understanding of the uncertainty in the assignment of observations, particularly in datasets with overlapping clusters or noise (like in this case).

Expected Maximization

For the estimation of the parameters, we cannot directly use Maximum Likelihood (ML) because we do not know the responsabilities for each observation. The cluster assignments are latent variables, and this missing information prevents us from straightforwardly applying ML.

To address this, the **Expectation-Maximization (EM)** algorithm is used. This iterative approach alternates between estimating the responsabilities of cluster membership (E-step) based on the current parameters and adjusting the parameters (M-step) based on these probability estimates.

The objective of this algorithm is to estimate the **complete data Likelihood**:

$$L_c(\psi|(\mathbf{x}_1, \mathbf{z}_1), ..., (\mathbf{x}_n, \mathbf{z}_n)) = \prod_{i=1}^{n} \prod_{k=1}^{K} [\pi_k f(\mathbf{x}_i; \theta_k)]^{z_{ik}}$$

- **Initialization**: The algorithm starts with an initial guess for the parameters ψ, $\dot{\psi}$
- **E-step (Expectation)**: Using the current parameters, it calculates the responsibilities for each observation's membership in each cluster:

$$\hat{z}_{ik} = \frac{\dot{\pi}_k f(\mathbf{x}_i; \dot{\theta}_k)}{p(\mathbf{x}; \dot{\psi})}$$

- **M-step (Maximization)**: These responsabilities are then used to update (form $\dot{\psi}$ to $\ddot{\psi}$) the parameters by maximizing the expectation of the complete data likelihood function, ensuring better alignment with the observed data.

This process repeats iteratively until convergence, which occurs when the increase in the expectation of the *Likelihood Function* becomes negligible. The EM algorithm thus efficiently handles the uncertainty in cluster assignments and provides robust parameter estimates.

```
mcl_res <- Mclust(data = scaled_data, G = 1:9, modelNames = NULL)
summary(mcl_res)
```

```
## ----------------------------------------------------
## Gaussian finite mixture model fitted by EM algorithm
## ----------------------------------------------------
##
## Mclust EVE (ellipsoidal, equal volume and orientation) model with 3 components:
##
##  log-likelihood   n df       BIC       ICL
##       -964.6146 167 90 -2389.849 -2400.492
##
## Clustering table:
##  1  2  3
## 50 74 43
```

```
### Paramethers ################################################################################

mcl_res$parameters$pro
```

```
## [1] 0.3016813 0.4316285 0.2666902
```

```
mcl_res$parameters$mean
```

```
##                  [,1]        [,2]       [,3]
## child_mort  1.1865530 -0.34103357 -0.7902842
## exports    -0.6095171  0.07528567  0.5676417
```

```
## health     -0.1385771 -0.24725312  0.5569294
## imports    -0.1223892  0.01962089  0.1066916
## income     -0.7645003 -0.25796704  1.2823169
## inflation   0.2999961  0.03365464 -0.3938259
## life_expec -1.0919392  0.14913860  0.9938317
## total_fer   1.1946290 -0.38000616 -0.7363441
## gdpp       -0.6459555 -0.37174700  1.3323672
```

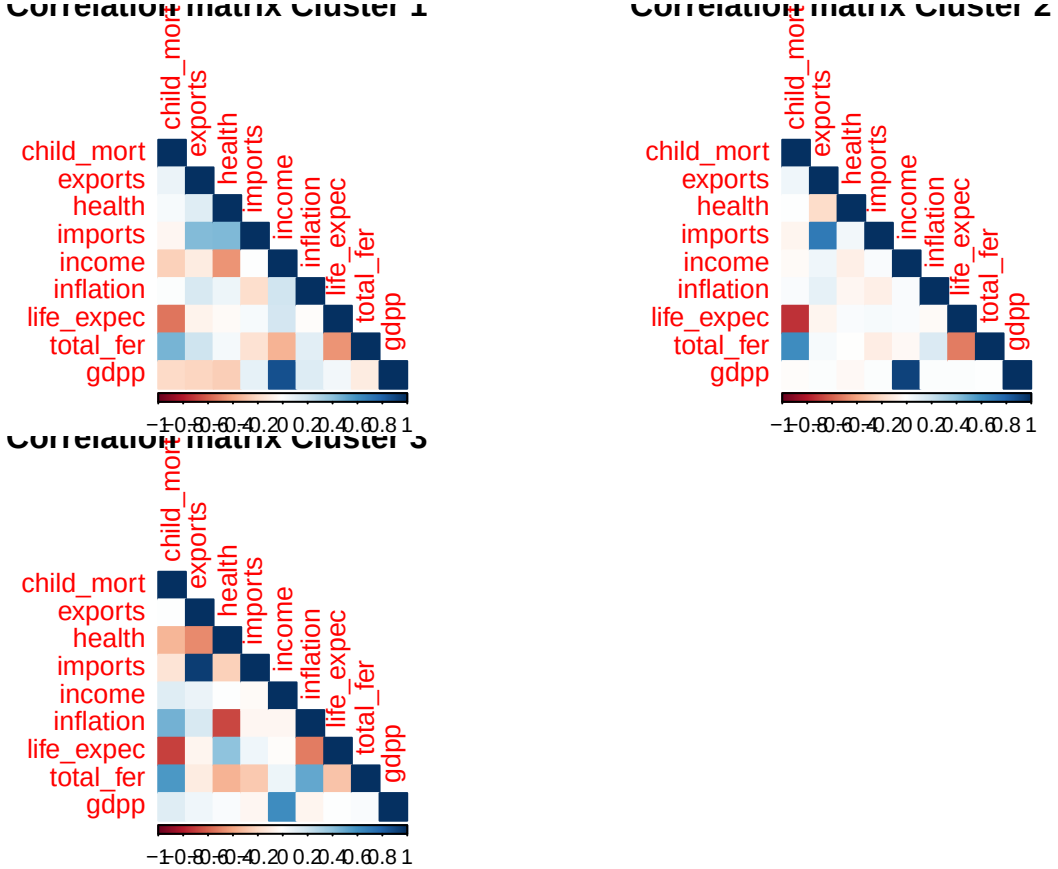
```
mcl_res$parameters$variance
```

```
## $modelName
## [1] "EVE"
##
## $d
## [1] 9
##
## $G
## [1] 3
##
## $sigma
## , , 1
##
##              child_mort      exports      health       imports        income
## child_mort  0.566654370  0.037475310  0.03233771 -0.0241380798 -0.0149044751
## exports     0.037475310  0.352475970  0.09349547  0.1838116913 -0.0047965332
## health      0.032337707  0.093495465  1.43221077  0.3807987095 -0.0447897920
## imports    -0.024138080  0.183811691  0.38079871  0.5015533061  0.0003474916
## income     -0.014904475 -0.004796533 -0.04478979  0.0003474916  0.0073825369
## inflation   0.016409445  0.122764295  0.11662379 -0.1603217599  0.0205308910
## life_expec -0.382715233 -0.035215333 -0.02380078  0.0265751545  0.0145891283
## total_fer   0.316651180  0.107804962  0.04814253 -0.1004029878 -0.0262819825
## gdpp       -0.008056949 -0.006883353 -0.01585643  0.0040312831  0.0040188240
##              inflation  life_expec    total_fer         gdpp
## child_mort  0.01640944 -0.38271523  0.316651180 -0.008056949
## exports     0.12276430 -0.03521533  0.107804962 -0.006883353
## health      0.11662379 -0.02380078  0.048142527 -0.015856430
## imports    -0.16032176  0.02657515 -0.100402988  0.004031283
## income      0.02053089  0.01458913 -0.026281983  0.004018824
## inflation   1.64148968 -0.02017276  0.143159624  0.010008373
## life_expec -0.02017276  0.92008726 -0.386063033  0.003067870
## total_fer   0.14315962 -0.38606303  0.811534697 -0.004867599
## gdpp        0.01000837  0.00306787 -0.004867599  0.003267363
##
## , , 2
##
##               child_mort     exports       health      imports       income
## child_mort  0.1551011945  0.01716283  0.002016293 -0.016405401 -0.003448626
## exports     0.0171628317  0.50906517 -0.116994370  0.374651827  0.015136734
## health      0.0020162931 -0.11699437  0.746084725  0.034477250 -0.022298841
## imports    -0.0164054014  0.37465183  0.034477250  0.550728390  0.006017282
## income     -0.0034486259  0.01513673 -0.022298841  0.006017282  0.098187800
## inflation   0.0077901752  0.06069419 -0.027550614 -0.052452029  0.007058122
## life_expec -0.1581408803 -0.02137306  0.011140353  0.015912038  0.005338666
## total_fer   0.1287309233  0.01437835 -0.004481182 -0.037701066 -0.005963350
## gdpp       -0.0009691469  0.00220763 -0.006233138  0.002265833  0.064232510
```

```
##               inflation   life_expec     total_fer          gdpp
## child_mort  0.007790175 -0.158140880  0.1287309233 -0.0009691469
## exports     0.060694193 -0.021373057  0.0143783484  0.0022076299
## health     -0.027550614  0.011140353 -0.0044811820 -0.0062331375
## imports    -0.052452029  0.015912038 -0.0377010662  0.0022658331
## income      0.007058122  0.005338666 -0.0059633503  0.0642325095
## inflation   0.609141452 -0.012182689  0.0639467227  0.0027583663
## life_expec -0.012182689  0.307930103 -0.1493176045  0.0016645298
## total_fer   0.063946723 -0.149317605  0.2744644273  0.0004622381
## gdpp        0.002758366  0.001664530  0.0004622381  0.0485608522
##
## , , 3
##
##             child_mort     exports      health     imports       income
## child_mort  0.02269211  0.00149000 -0.05985427 -0.02985513  0.023462084
## exports     0.00149000  1.73546205 -0.73427730  1.63855230  0.133722279
## health     -0.05985427 -0.73427730  1.38191977 -0.36687656  0.010580874
## imports    -0.02985513  1.63855230 -0.36687656  1.73955857 -0.043813311
## income      0.02346208  0.13372228  0.01058087 -0.04381331  1.340328677
## inflation   0.03516858  0.10440340 -0.37992013 -0.03137198 -0.027185489
## life_expec -0.02514144 -0.01854298  0.11411466  0.02141511 -0.003098843
## total_fer   0.02290069 -0.03497786 -0.10610565 -0.09347056  0.022660454
## gdpp        0.01910802  0.07529780  0.02694005 -0.05713055  0.659783866
##               inflation    life_expec    total_fer          gdpp
## child_mort  0.03516858 -0.0251414359  0.022900687  0.0191080155
## exports     0.10440340 -0.0185429775 -0.034977863  0.0752977982
## health     -0.37992013  0.1141146615 -0.106105652  0.0269400475
## imports    -0.03137198  0.0214151078 -0.093470562 -0.0571305480
## income     -0.02718549 -0.0030988433  0.022660454  0.6597838656
## inflation   0.23319566 -0.0596984098  0.066223408 -0.0232343909
## life_expec -0.05969841  0.0575100763 -0.017917836  0.0003268637
## total_fer   0.06622341 -0.0179178356  0.069585188  0.0066377240
## gdpp       -0.02323439  0.0003268637  0.006637724  0.8322834481
##
##
## $scale
## [1] 0.1689399
##
## $shape
##               [,1]       [,2]        [,3]
##  [1,]  0.002594791 0.02528363  2.24552799
##  [2,]  9.225009068 4.52094503  7.83231213
##  [3,]  2.890891703 5.47934417 21.65129612
##  [4,]  0.884113808 0.70664786  0.12665771
##  [5,]  8.650670026 3.08699768  0.26352560
##  [6,] 10.555585424 3.77141865  0.74658025
##  [7,]  1.811351347 0.28336639  0.03655823
##  [8,]  2.861668478 0.81985501  0.26927993
##  [9,]  0.034533197 0.83535938 10.70502252
##
## $orientation
##              child_mort      exports       health     imports       income
## child_mort -0.007622142  0.060897971  0.001171175 -0.02639648 -0.419146471
## exports    -0.021557780 -0.076572775  0.683632303 -0.69129595 -0.025466473
```

```
## health       0.006296072 -0.848833404 -0.333171926 -0.15582849  0.012984445
## imports      0.012602805 -0.382155493  0.646451713  0.64394116 -0.003575092
## income       0.567859421  0.025245241  0.027249794  0.05594978  0.025350256
## inflation   -0.001358950  0.313542489  0.051575696  0.11477321  0.479890455
## life_expec -0.002237248 -0.099661895 -0.011060252  0.05110294  0.612513889
## total_fer    0.019666960  0.122081472 -0.012450430  0.24186134 -0.466158159
## gdpp        -0.822448011  0.009191606  0.007887131  0.07112467  0.008493503
##                inflation   life_expec   total_fer         gdpp
## child_mort   0.188761356  0.883584096 -0.05697778  0.015914117
## exports      0.127809013 -0.046190527  0.16376070  0.049938421
## health       0.374992536 -0.018777695  0.02632425  0.048193313
## imports     -0.004030152  0.037208536 -0.12389396 -0.066896197
## income      -0.008948960  0.002078465 -0.02937746  0.819411589
## inflation    0.800099270  0.031456069 -0.11658901 -0.028638214
## life_expec -0.259524696  0.394596777  0.62377444  0.001078231
## total_fer    0.315917750 -0.241998746  0.74178989  0.011589281
## gdpp        -0.001530965 -0.012028717 -0.00951101  0.563971830
```

```
par(mfrow = c(2, 2))
corrplot(corpcor::decompose.cov(round(mcl_res$parameters$variance$sigma[, , 1], 3))$r, type = "lower", m
         main = "Correlation matrix Cluster 1")
corrplot(corpcor::decompose.cov(round(mcl_res$parameters$variance$sigma[, , 2], 3))$r, type = "lower", m
         main = "Correlation matrix Cluster 2")
corrplot(corpcor::decompose.cov(round(mcl_res$parameters$variance$sigma[, , 3], 3))$r, type = "lower", m
         main = "Correlation matrix Cluster 3")
```



For this type of models, in order to maintain the number of parameters to estimate relatively low, *Parsimo-*

nious Models are used. These models allow us to set equal in advance certain characteristics of the clusters, such as **Volume**, **Shape**, and **Orientation**, thereby reducing model complexity while still capturing the essential structure of the data.

In this case, the algorithm identifies the **EVE** model (Equal Volume, Variable Shape, and Equal Orientation) as the optimal choice. This model is flexible in terms of shape for each cluster but assumes that all clusters share the same orientation and volume. This balance between flexibility and simplicity makes it well-suited to the data.

The number of free covariance parameters for the **EVE** model is calculated as:

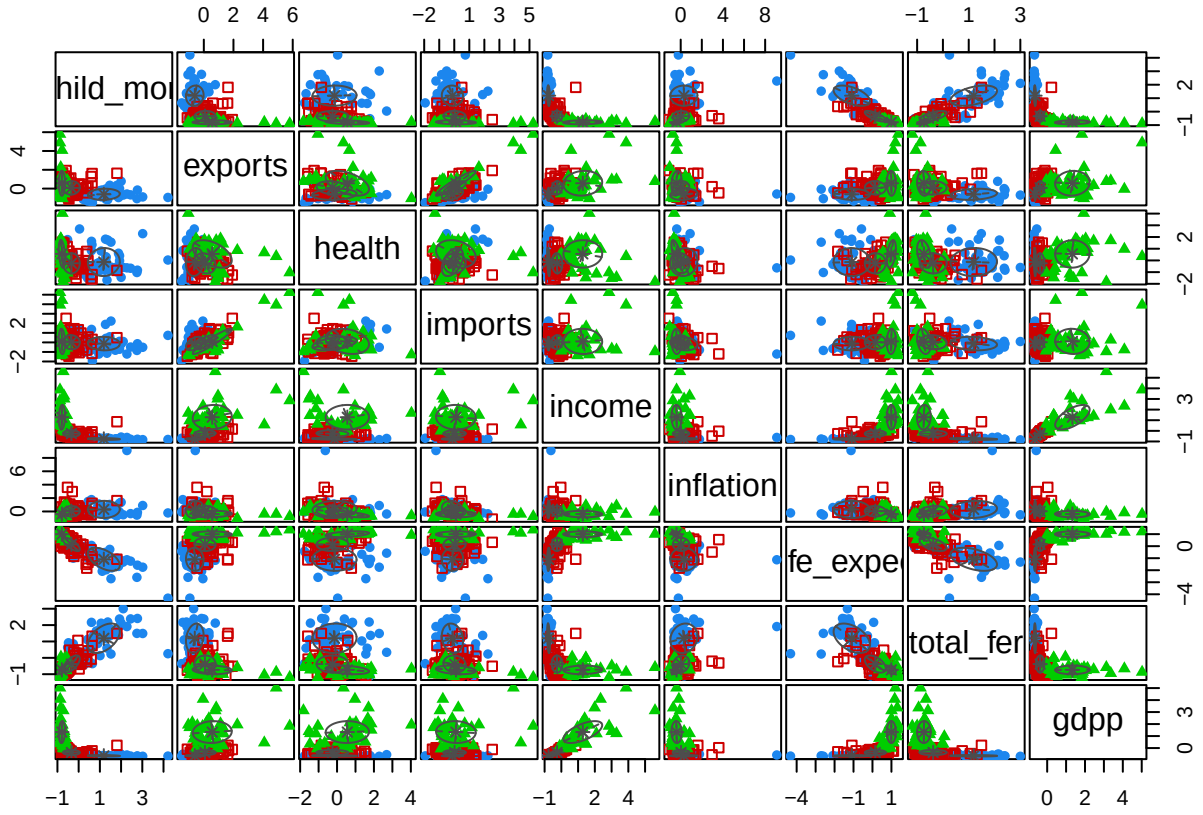
$$1 + K(d-1) + \frac{d(d-1)}{2}$$

Where:

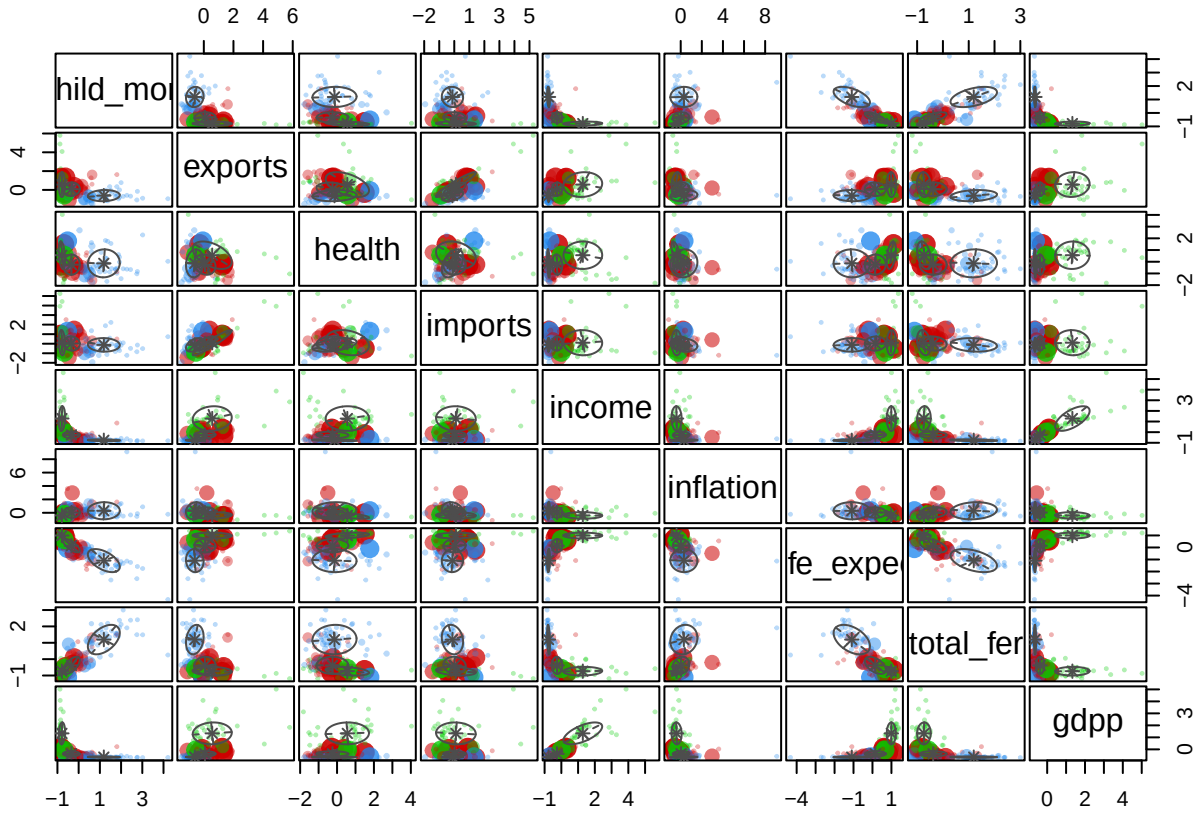
- K: Number of clusters,
- d: Dimensionality of the data.

This formula accounts for $1 + K(d-1)$ parameters for the means of the clusters and an additional $\frac{d(d-1)}{2}$ for the shared orientation matrix. By enforcing this structure, the model reduces the risk of overfitting while maintaining the ability to represent complex data distributions.

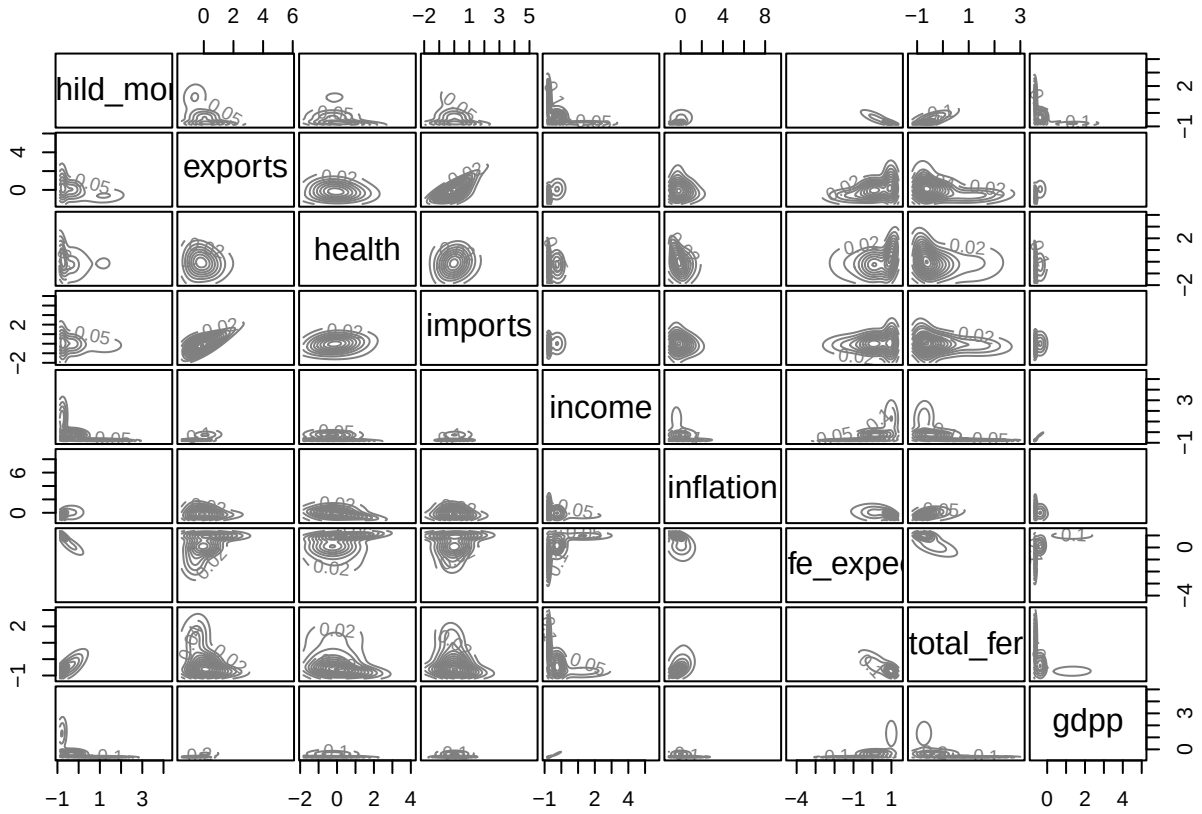
```
plot(mcl_res, what = "classification")
```




```
plot(mcl_res, what = "uncertainty")
```




```
plot(mcl_res, what = "density")
```



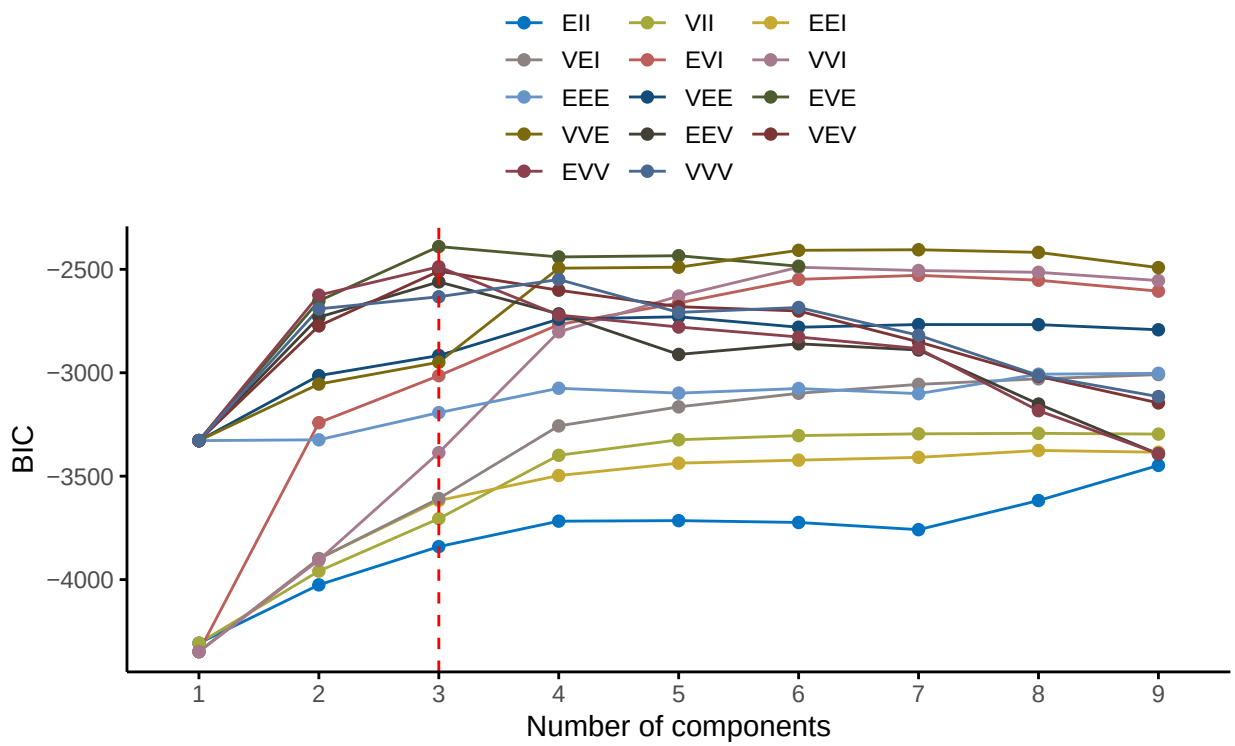
The pariplot shows the classification in the original dimensional space. While this visualization provides an overview of the data, it is not particularly readable due to the high dimensionality and overlapping points. However, as observed in the previous analysis, some of the most significant variables for classification are `gdpp` and `income`, which strongly influence the clustering.

Other variables also contribute to the classification but to a lesser extent. These secondary variables might still play a role in defining the cluster structure but are less impactful in distinguishing the groups compared to `gdpp` and `income`.

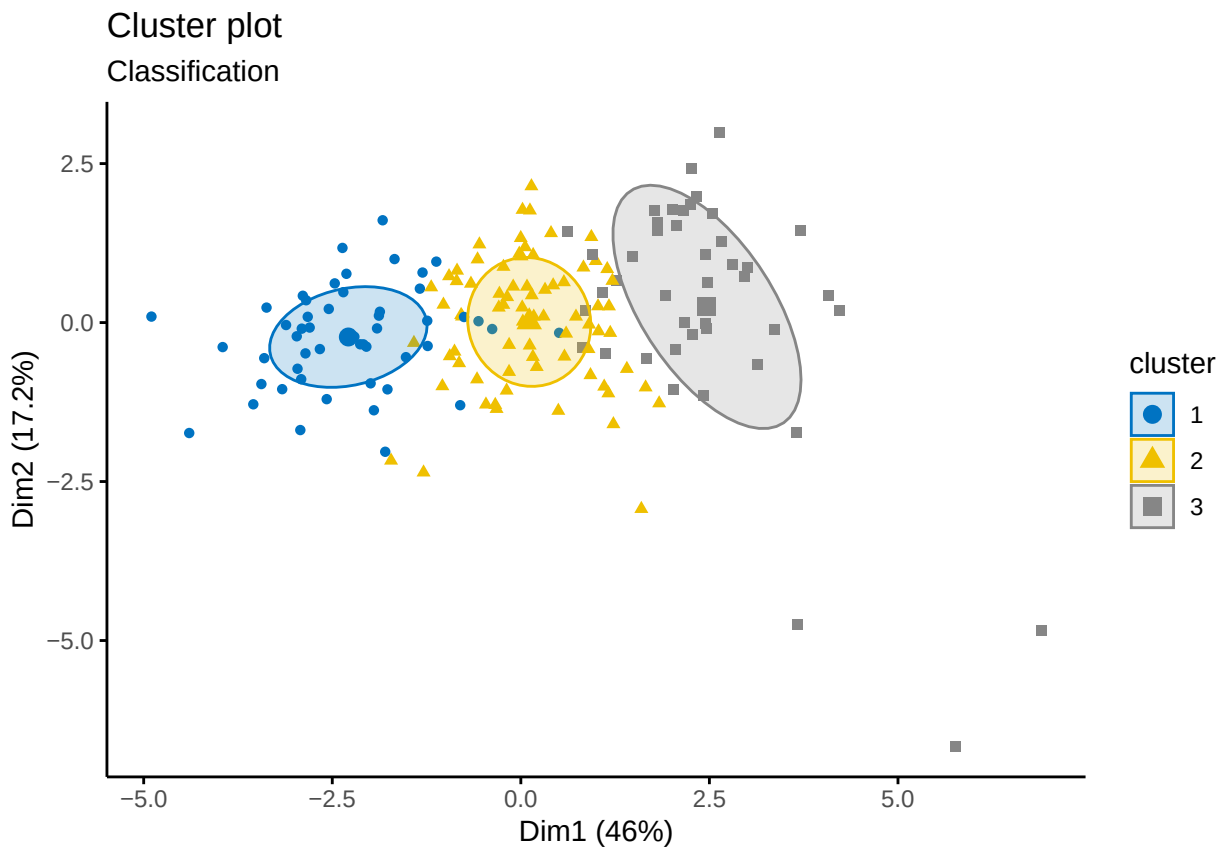
Focusing on the first two PCs variables can help simplify the interpretation of the clusters and provide clearer insights into the differences in welfare between nations.

```
fviz_mclust(mcl_res, "BIC", palette = "jco") +
  theme(legend.position = "top")
```

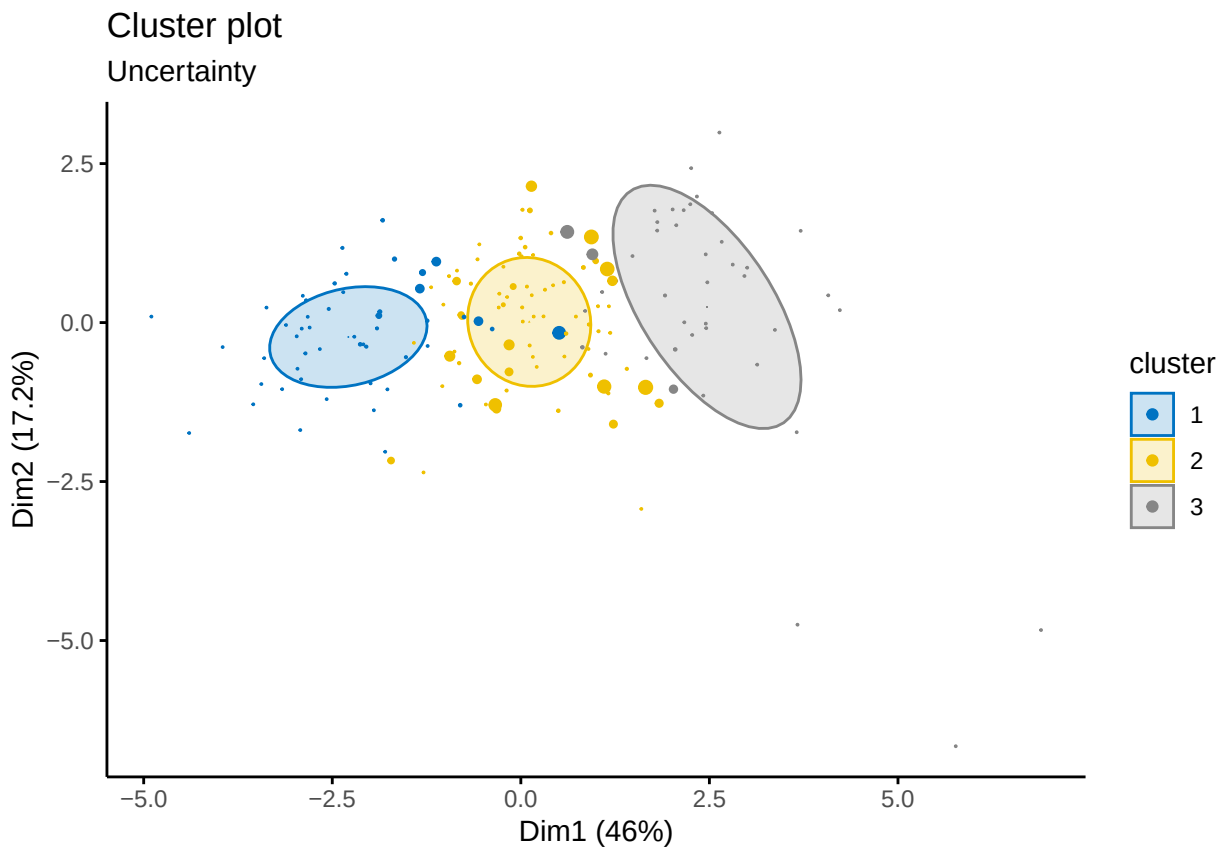
```
## Warning: `gather_()` was deprecated in tidyr 1.2.0.
## i Please use `gather()` instead.
## i The deprecated feature was likely used in the factoextra package.
##   Please report the issue at <https://github.com/kassambara/factoextra/issues>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
fviz_mclust(mcl_res, "classification", geom = "point", pointsize = 1.5, palette = "jco")
```




```
fviz_mclust(mcl_res, "uncertainty", palette = "jco")
```



Interpretation of the Plot A possible explanation for this classification could be as follows:

Cluster 1 (Blue):

- This cluster seems to represent the **poorer countries** with low values of **GDP per capita** (`gdpp`), `income`, and `life_expect`.
- These countries likely exhibit **high levels of child mortality** and **total fertility rates**, which are common indicators of underdeveloped nations.
- The cluster's tight grouping suggests relative homogeneity within this group.

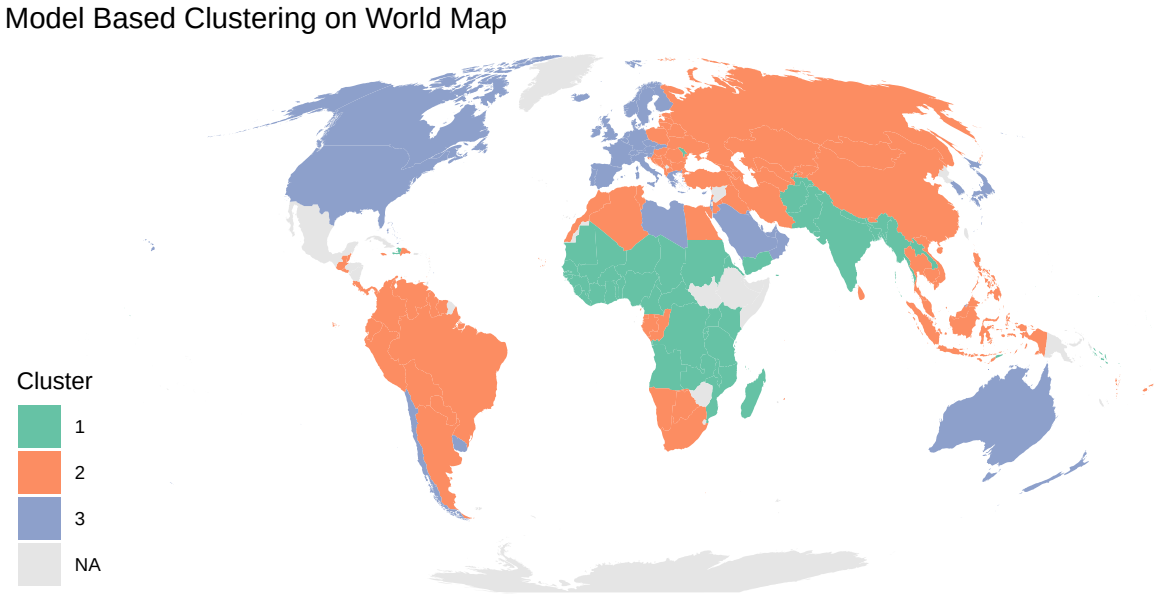
Cluster 2 (Yellow):

- This cluster likely represents **countries on the path to development**.
- There is noticeable **variability along Dim2 (17.2%)**, which may reflect differences in **trade-related variables**, such as imports and exports.
- These nations might have **moderate levels of GDP, income, and life expectancy**, indicating progress but with significant variability.

Cluster 3 (Gray):

- This cluster appears to encompass the **most developed countries**, characterized by **high values of Dim1 (46%) and a lot of variability in the Dim2 (17.2%),** in which we observe some significant negative values in the bottom-right corner of the plot.
- These countries are likely **wealthy**, with strong economic indicators such as **high GDP per capita and income**, and **low levels of child mortality and fertility**.

```r
cluster_values <- unname(mcl_res$classification)

cluster_data <- tibble(
  region  = rownames(mcl_res$data),
  cluster = as.factor(cluster_values)
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
                         "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                         "United Kingdom" = "UK",
                         "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(long <= 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")

ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Model Based Clustering on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



Mixture of Regression Model

The mixture of regression models represents a flexible framework designed to capture heterogeneity in data by assuming that observations are generated from a mixture of distinct subpopulations (or clusters). Each subpopulation is modeled by a separate regression relationship, allowing the model to explain the relationship between the response variable and the covariates within each cluster.

Unlike traditional regression models, which assume a single global relationship for the entire dataset, mixture of regression models are well-suited for cases where the underlying data structure is inherently heterogeneous. This framework is particularly useful when the data comes from a population that can be divided into subgroups with unique characteristics or patterns.

We can have those two situations in term of framework to use:

- Fixed Covariate models
- **Random Covariate models (*Cluster Weighted models*)**

Some premises abvout the implementations

I decided to apply these models out of personal curiosity, despite the fact that they are specifically designed for probability distributions of covariates defined on $\mathbb{R}$ (such as the Normal and t-distributions). Unfortunately, the attribute I'm working with is defined on $\mathbb{R}^+$ (as most of my variables are, by nature). Therefore, the application of these models is technically not correct. However, given this limitation, I proceeded with the analysis, being fully aware of the theoretical shortcomings of this implementation. For the documentation, I reviewed materials that describe the use of these models with distributions defined on $\mathbb{R}$, which influenced my decision to proceed with this approach.

Fixed Covariate Model

Formally, the model assumes that the probability density of the response variable y given the covariates x can be expressed as a weighted sum of K cluster-specific regression models:

$$p(y \mid x; \vartheta) = \sum_{k=1}^{K} \pi_k p(y \mid x; \beta_k, \gamma_k)$$

Where:

- π_k correspond to the mixing proportion for the k-th cluster ($\sum_{k=1}^{K} \pi_k = 1$)
- β_k correspond to the covariates for the regression model in cluster k
- γ_k correspond to the additional paramether for the ditribution of y

Model framework

The expected value for the model is:

$$\mathbb{E}(y|x, k; \beta_k) = \mu_{Y|k}(x; \beta_k)$$

$$\mu_{Y|k}(x; \beta_k) = \beta_{k0} + \beta_{k1} x_1 + ... + \beta_{kd} x_d$$

Where d is the number of covariates.

NB: In this application I assumed linear relationship between the response and covariates.

For instance, when a multivariate normal distribution is used, the model can be expressed as:

$$p(y \mid x; \beta_k, \gamma_k) = \mathcal{N}(y; \mu_{Y|k}(x; \beta_k), \Sigma_k)$$

Here, $\mu_{Y|k}(x; \beta_k)$ is the mean function of the k -th cluster, and $_k$ is the cluster-specific variance.

So, the computation in the E-step (responsability computation) of **EM** algorithm will be:

$$\hat{z}_{ik} = \frac{\pi_k p(y|x; \beta_k, \gamma_k)}{\sum_{j=1}^{K} \pi_j p(y|x; \beta_j, \gamma_j)}$$

Random Covariate Model (Cluster Weighted model)

The **random covariate model**, or **Cluster-Weighted Model (CWM)** (Ingrassia, Minotti, and Punzo 2014), extends the fixed covariate model by incorporating the distribution of the covariates X into the framework. This approach models the joint distribution of (y, x), rather than only the conditional distribution of y given x.

Formally, the joint probability density is expressed as:

$$p(y, x; \vartheta) = \sum_{k=1}^{K} \pi_k p(x; \theta_k) p(y \mid x; \beta_k, \gamma_k)$$

Where:

- π_k: Mixing proportion for cluster k, satisfying $\sum_{k=1}^{K} \pi_k = 1$.
- $p(x; \theta_k)$: The marginal density of the covariates x in cluster k, parameterized by θ_k.

- $p(y \mid x; \beta_k, \gamma_k)$: The conditional density of the response y in cluster k, given the covariates x.

This model assumes that the observations belong to K distinct clusters, where each cluster is characterized by:

1. A marginal distribution for the covariates (x).
2. A conditional regression model describing y given x.

Model framework The expected value of y, given x and the cluster k, remains:

$$\mathbb{E}(y|x, k; \beta_k) = \mu_{Y|k}(x; \beta_k)$$

The conditional mean $\mu_{Y|k}(x; \beta_k)$ can be expressed as:

$$\mu_{Y|k}(x; \beta_k) = \beta_{k0} + \beta_{k1}x_1 + \cdots + \beta_{kd}x_d$$

Where d is the number of covariates.

However, the CWM framework introduces flexibility by allowing the covariates x to be generated from cluster-specific distributions $p(x; \theta_k)$.

For instance, $p(x; \theta_k)$ might be modeled using a multivariate normal distribution:

$$p(x; \theta_k) = \mathcal{N}(x; \mu_{X|k}, \Sigma_k)$$

In this case, $\mu_{X|k}$ and Σ_k represent the mean vector and covariance matrix for x in cluster k, respectively.

In this case the E-step will be:

$$\hat{z}_{ik} = \frac{\pi_k p(y|x; \beta_k, \gamma_k) p(x; \theta_k)}{\sum_{j=1}^{K} \pi_j p(y|x; \beta_j, \gamma_j) p(x; \theta_j)}$$

Parsimonious model for CWM

Model	$\boldsymbol{X}\|\Omega_g$		$Y\|\boldsymbol{x}, \Omega_g$		Number of free parameters		
Identifier	Density	Constraint	Density	Constraint	$\boldsymbol{X}$	$Y\|\boldsymbol{x}$	weights
tt-VV	t	Variable	t	Variable	$G\left(d + \frac{d(d+1)}{2} + 1\right)$	$G\,(d+3)$	$G-1$
tt-VE	t	Variable	t	Equal	$G\left(d + \frac{d(d+1)}{2} + 1\right)$	$d+3$	$G-1$
tt-EV	t	Equal	t	Variable	$d + \frac{d(d+1)}{2} + 1$	$G\,(d+3)$	$G-1$
NN-VV	Normal	Variable	Normal	Variable	$G\left(d + \frac{d(d+1)}{2}\right)$	$G\,(d+2)$	$G-1$
NN-VE	Normal	Variable	Normal	Equal	$G\left(d + \frac{d(d+1)}{2}\right)$	$d+2$	$G-1$
NN-EV	Normal	Equal	Normal	Variable	$d + \frac{d(d+1)}{2}$	$G\,(d+2)$	$G-1$
tN-VV	t	Variable	Normal	Variable	$G\left(d + \frac{d(d+1)}{2} + 1\right)$	$G\,(d+2)$	$G-1$
tN-VE	t	Variable	Normal	Equal	$G\left(d + \frac{d(d+1)}{2} + 1\right)$	$d+2$	$G-1$
tN-EV	t	Equal	Normal	Variable	$d + \frac{d(d+1)}{2} + 1$	$G\,(d+2)$	$G-1$
Nt-VV	Normal	Variable	t	Variable	$G\left(d + \frac{d(d+1)}{2}\right)$	$G\,(d+3)$	$G-1$
Nt-VE	Normal	Variable	t	Equal	$G\left(d + \frac{d(d+1)}{2}\right)$	$d+3$	$G-1$
Nt-EV	Normal	Equal	t	Variable	$d + \frac{d(d+1)}{2}$	$G\,(d+3)$	$G-1$

This table (Ingrassia, Minotti, and Punzo 2014) illustrates the possible models for the conditional distribution of Y and the covariates X. Two distributions are considered (Gaussian and t-distribution), with parameters that can either vary across components or remain equal. The table also provides a clear breakdown of the number of parameters required for each model configuration.

Model implementation with `flexCWD()`

For the implementation I've decided to try at first two bivariate implementation of the model and after an application in the whole dimension extracted trought the PCA.

For all the implementations are used the **NN-VV** model, with paramethers:

$$G(d + \frac{d(d+1)}{2}) \; + \; G(d+2) \; + \; G - 1$$

```
fit_cwm1 <- cwm(data$total_fer~imports,
                familyY = "gaussian",
                data,
                k = 2:6,
                seed = 333)
```

1. Implementation with `tot_fert` and `imports`

```
##
## Estimating model with k=2, familyY=gaussian ********************************************
## Estimating model with k=3, familyY=gaussian ********************************************
## Estimating model with k=4, familyY=gaussian ********************************************
## Estimating model with k=5, familyY=gaussian ********************************************
## Estimating model with k=6, familyY=gaussian ********************************************
##
## Best model according to AIC is obtained with k = 6 group(s) and family gaussian(identity)
##
## Best model according to AICc, AICu, AIC3 is obtained with k = 3 group(s) and family gaussian(identity
##
## Best model according to AWE, BIC, CAIC, ICL is obtained with k = 2 group(s) and family gaussian(ident
```

```
summary(fit_cwm1)
```

```
## ----------------------------------
## Best fitted model according to BIC
## ----------------------------------
##  loglikelihood   n df     BIC
##           -252 167  7 -539.82
##
## Clustering table:
##   1   2
## 113  54
##
## Prior: comp.1 = 0.64174, comp.2 = 0.35826
##
## Distribution used for GLM: gaussian(identity). Parameters:
##
## Component 1
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  2.2322994  0.0794874 28.0837 < 2.2e-16 ***
```

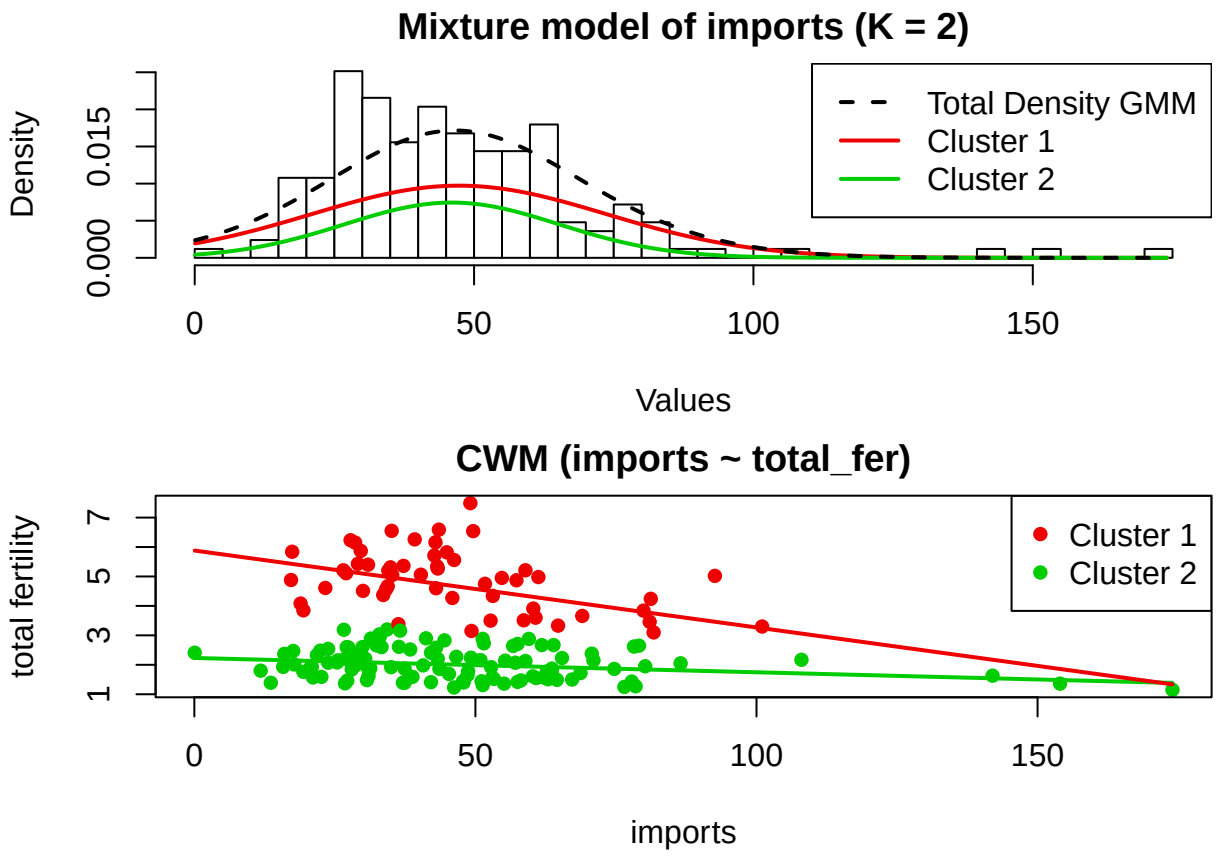
```
## imports     -0.0048675  0.0015051 -3.2341  0.001474 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma = 0.48657
##
## Component 2
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  5.8810364  0.1973679 29.7973 < 2.2e-16 ***
## imports     -0.0261211  0.0037521 -6.9617 7.567e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma = 1.0674
```

This is the best model according to the BIC, which identifies an optimal number of components as 2. The `summary` function provides all component parameters $(\pi_k, \beta_{0k}, \beta_{1k}, \sigma_k)$. Those parameters are in respect to the response varaible (`total_fer`). These parameters pertain to the conditional distribution $p(y|x; \beta_k, \sigma_k)$. Also all the β are significant.

We observe a negative relationship of varying magnitudes across the clusters, along with a significant difference in variance. For better interpretation, a representation is required that visualizes all the models for Y and the covariates X.

```r
cluster_colors <- c("red2", "green3")

## Paramethers
prior <- fit_cwm1$models[[1]]$prior
clusters <- fit_cwm1$models[[1]]$cluster
muX <- tapply(fit_cwm1$data$imports, clusters, mean)
sdX <- tapply(fit_cwm1$data$imports, clusters, sd)

x_vals <- seq(min(fit_cwm1$data$imports), max(fit_cwm1$data$imports), length.out = 500)

densities <- sapply(1:length(muX), function(i) {
  prior[i] * dnorm(x_vals, mean = muX[i], sd = sdX[i])
})

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

## Histogram
hist(fit_cwm1$data$imports, breaks = 30, probability = TRUE,
     col = "white", border = "black",
     main = "Mixture model of imports (K = 2)",
     xlab = "Values", ylab = "Density")

for (i in 1:ncol(densities)) {
  lines(x_vals, densities[, i], col = cluster_colors[i], lwd = 2)
}

overall_density <- rowSums(densities)
lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)

legend("topright", legend = c("Total Density GMM",
                              paste("Cluster", 1:length(prior))),
col = c("black", cluster_colors[1:length(prior)]), lwd = 2, lty = c(2, rep(1, length(prior))))
```

```r
## Scatterplot
plot(fit_cwm1, col = cluster_colors[clusters],
     xlab = "imports", ylab = "total fertility",
     main = "CWM (imports ~ total_fer)", pch = 16)

legend("topright", legend = paste("Cluster", 1:length(prior)),
       col = cluster_colors[1:length(prior)], pch = 16)
```

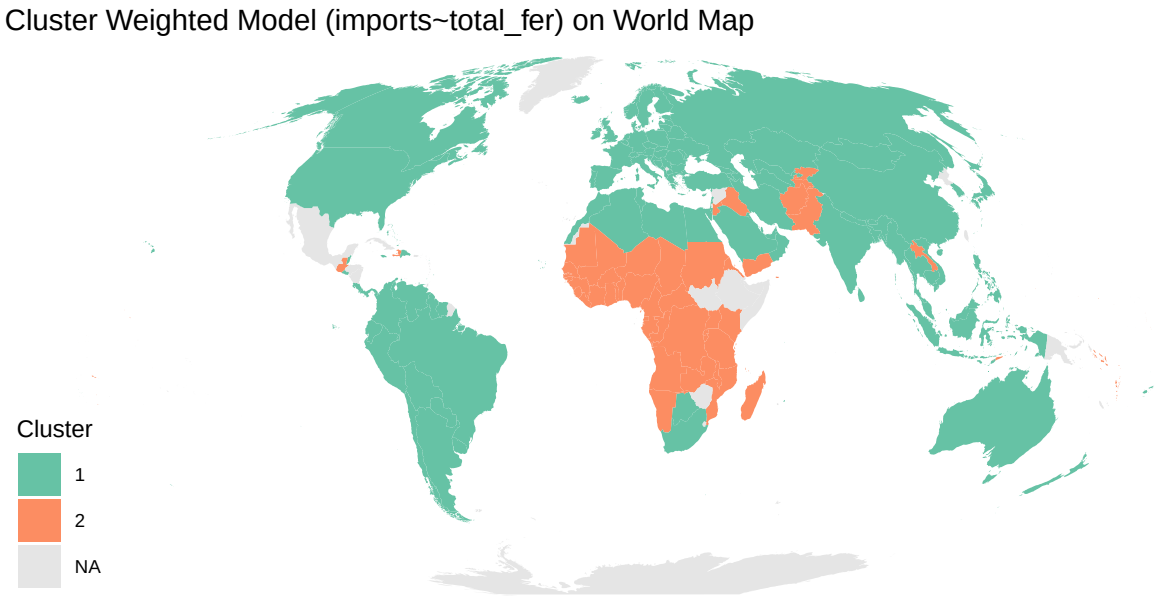


This model employs a Normal distribution for both $p(y|x;\beta_k,\sigma_k)$ and $p(x;\theta_k)$, where θ_k corresponds to the Gaussian model parameters. All parameters are variable, as the simplicity of the model makes it unnecessary to constrain any parameters to be equal. Thus, the parsimonious model used here is **NN-VV**

First of all, we notice that the classification of observations is primarily driven by the `total_fert` dimension. The linear model appears to fit both clusters well, capturing the distinct relationships between the two variables. Regarding the models for X, there is a similarity in terms of location, as the two distributions overlap, but a noticeable difference in variance is present.

```r
cluster_data <- tibble(
  region  = rownames(fkmed3_res$clus),
  cluster = as.factor(clusters)
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
```

```
                          "Cote d'Ivoire" = "Ivory Coast",
                          "Kyrgyz Republic" = "Kyrgyzstan",
                          "Lao" = "Laos",
                          "Macedonia, FYR" = "North Macedonia",
                          "Micronesia, Fed. Sts." = "Micronesia",
                          "Slovak Republic" = "Slovakia",
                          "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                          "United Kingdom" = "UK",
                          "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(long <= 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")

ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Cluster Weighted Model (imports~total_fer) on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



We can observe that the classification differentiates between poorer countries and mid-developed to developed countries. Given that we are using only two variables, this classification appears highly significant.

Moreover, since we used a **CWM model**, we can extract additional information from the partition itself. This allows us to better understand the relationship between the selected variables and generalize the classification, thanks to the probabilistic nature of the model.

```
fit_cwm2 <- cwm(total_fer~health,
                data,
                familyY = "gaussian",
                k = 2:6,
                seed = 333)
```

2. Implementation with `total_fer` and `health`

```
##
## Estimating model with k=2, familyY=gaussian ************************************
## Estimating model with k=3, familyY=gaussian ************************************************************
## Estimating model with k=4, familyY=gaussian ************************************************************
## Estimating model with k=5, familyY=gaussian ************************************************************
## Estimating model with k=6, familyY=gaussian ************************************************************
##
## Best model according to AIC, AICc, AICu, AIC3 is obtained with k = 4 group(s) and family gaussian(id
##
## Best model according to AWE, BIC, CAIC, ICL is obtained with k = 2 group(s) and family gaussian(iden
```

```r
summary(fit_cwm2)
```

```
## ----------------------------------
## Best fitted model according to BIC
## ----------------------------------
##  loglikelihood   n df     BIC
##         -248.1 167  7 -532.04
##
## Clustering table:
##   1   2
## 114  53
##
## Prior: comp.1 = 0.65784, comp.2 = 0.34216
##
## Distribution used for GLM: gaussian(identity). Parameters:
##
## Component 1
##              Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  2.600198   0.098119 26.5004 < 2.2e-16 ***
## health      -0.083225   0.013080 -6.3627 1.877e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma = 0.46039
##
## Component 2
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.000728   0.229755 21.7655   <2e-16 ***
## health      -0.041089   0.032657 -1.2582   0.2101
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma = 1.1354
```

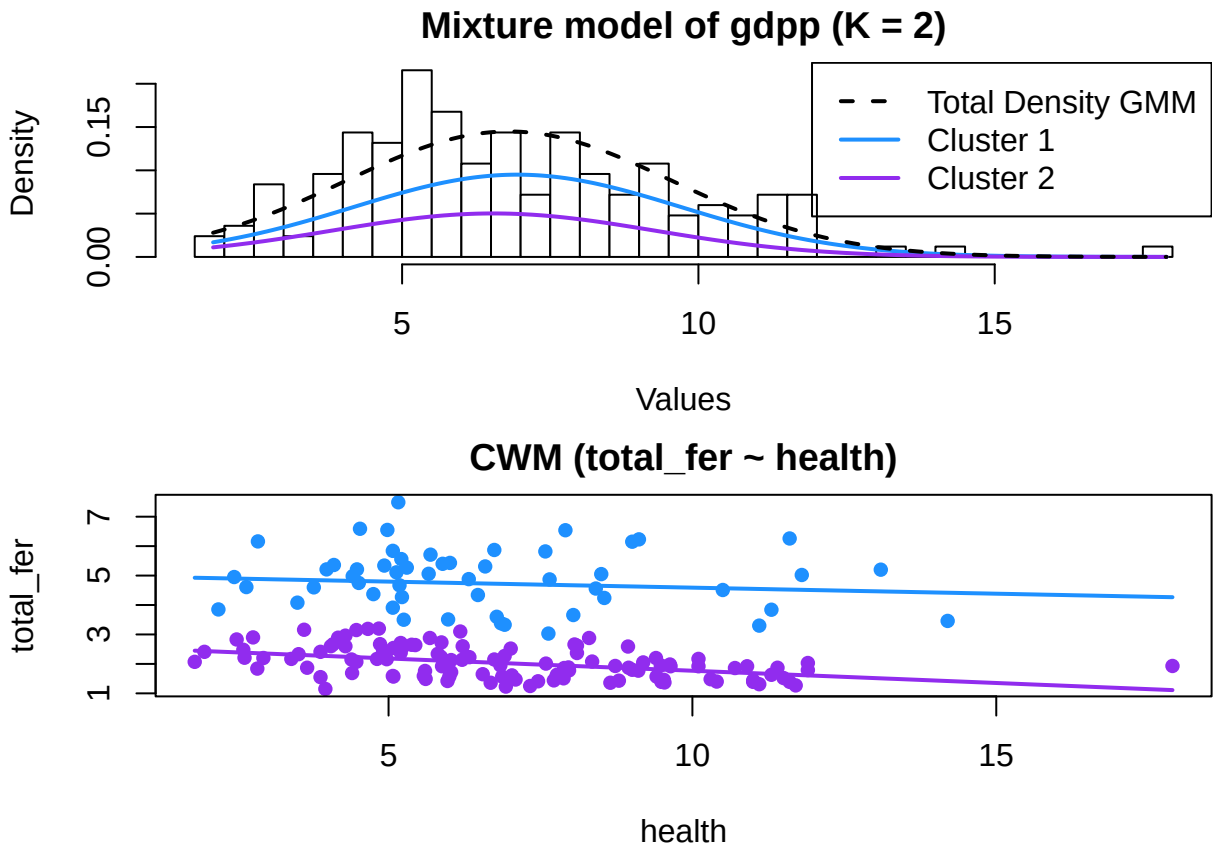
The **CWM output** identifies two clusters: **Cluster 1 (114 observations, 65.8%)** and **Cluster 2 (53 observations, 34.2%)**. The model uses a **Gaussian distribution with an identity link**, and the key variable is **health**.

- **Cluster 1:**
  - Strong **negative relationship** between health and the outcome (**-0.083**, $p < 0.001$), meaning higher health values correspond to lower outcomes.
  - **Smaller sigma (0.460)** suggests a more homogeneous group.
- **Cluster 2:**
  - **Weaker and non-significant** relationship with health (**-0.041**, $p = 0.21$).
  - **Higher sigma (1.1354)** indicates greater variability.

Cluster 1 shows a clear negative effect of health on the outcome, while Cluster 2 exhibit a lower negative trend. The larger variance in Cluster 2 suggests more heterogeneity, possibly indicating a less distinct classification.

```r
cluster_colors <- c("dodgerblue", "purple2")

## Paramethers
prior <- fit_cwm2$models[[1]]$prior
clusters <- fit_cwm2$models[[1]]$cluster
muX <- tapply(fit_cwm2$data$health, clusters, mean)
```

```r
sdX <- tapply(fit_cwm2$data$health, clusters, sd)

x_vals <- seq(min(fit_cwm2$data$health), max(fit_cwm2$data$health), length.out = 500)

densities <- sapply(1:length(muX), function(i) {
  prior[i] * dnorm(x_vals, mean = muX[i], sd = sdX[i])
})

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

## Histogram
hist(fit_cwm2$data$health, breaks = 30, probability = TRUE,
     col = "white", border = "black",
     main = "Mixture model of gdpp (K = 2)",
     xlab = "Values", ylab = "Density")

for (i in 1:ncol(densities)) {
  lines(x_vals, densities[, i], col = cluster_colors[i], lwd = 2)
}

overall_density <- rowSums(densities)
lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)

legend("topright", legend = c("Total Density GMM",
                              paste("Cluster", 1:length(prior))),
col = c("black", cluster_colors[1:length(prior)]), lwd = 2, lty = c(2, rep(1, length(prior))))

## Scatterplot
plot(fit_cwm2, col = cluster_colors[clusters],
     xlab = "health", ylab = "total_fer",
     main = "CWM (total_fer ~ health)", pch = 16)
```

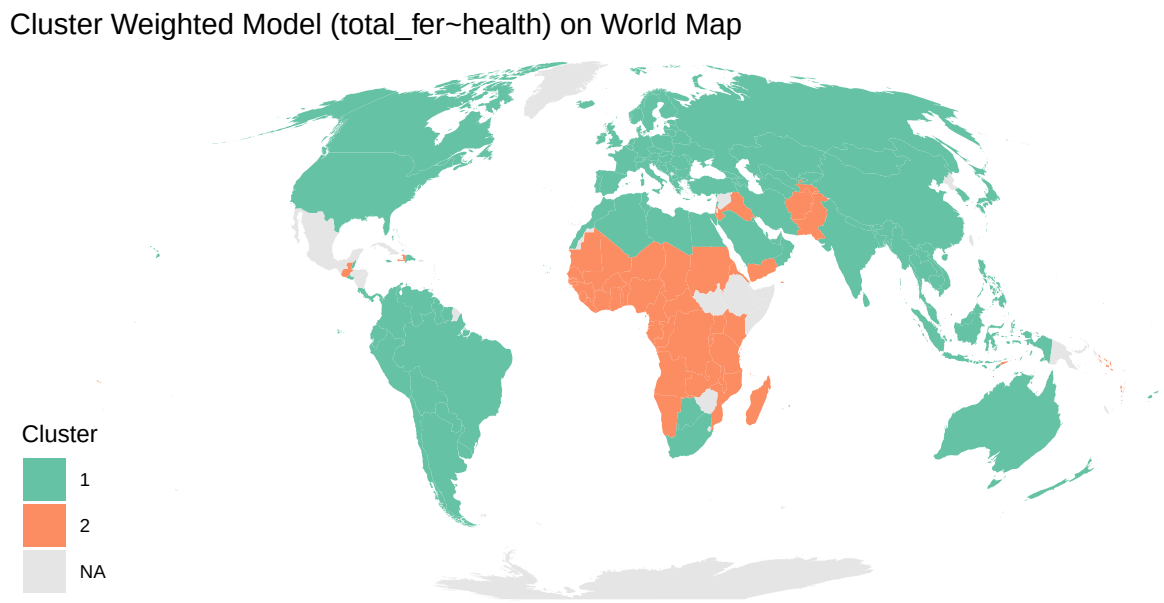


In this case, we observe that the slopes for both models are similar ($\beta_{11} = -0.08$ and $\beta_{12} = -0.04$) even though the second slope estimate is not significant. The intercepts are different, as are the probabilities (π_1, π_2) , influenced by the differing sizes of the clusters. The partitioning appears to be based on the Y-dimension. Furthermore, the first cluster shows greater dispersion, while the second is more concentrated.

Similarly, the usage of parsimonious models is not necessary due to the simplicity of the model (**NN-VV**).

```
cluster_data <- tibble(
  region  = rownames(fkmed3_res$clus),
  cluster = as.factor(clusters)
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
                         "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                         "United Kingdom" = "UK",
                         "United States" = "USA"))
```

```r
world_map <- map_data("world") %>%
  filter(long <= 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")

ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Cluster Weighted Model (total_fer~health) on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



The classification is quite similar to the previous one, which strengthens our conclusions about the partition and enhances the robustness of our analysis.

```r
attach(pca_data)
fit_cwm_pca <- cwm(PC1 ~ PC2,
                   pca_data,
                   familyY = 'gaussian',
                   k = 2:6,
```

```
                    seed = 333)
```

3. Implementation in PC1 and PC2 space

```
##
## Estimating model with k=2, familyY=gaussian ************************************************************
## Estimating model with k=3, familyY=gaussian ************************************************************
## Estimating model with k=4, familyY=gaussian ************************************************************
## Estimating model with k=5, familyY=gaussian ************************************************************
## Estimating model with k=6, familyY=gaussian ************************************************************
##
## Best model according to AIC, AICc, AIC3 is obtained with k = 4 group(s) and family gaussian(identity)
##
## Best model according to AICu, AWE, BIC, CAIC, ICL is obtained with k = 2 group(s) and family gaussian
```

```
summary(fit_cwm_pca)
```

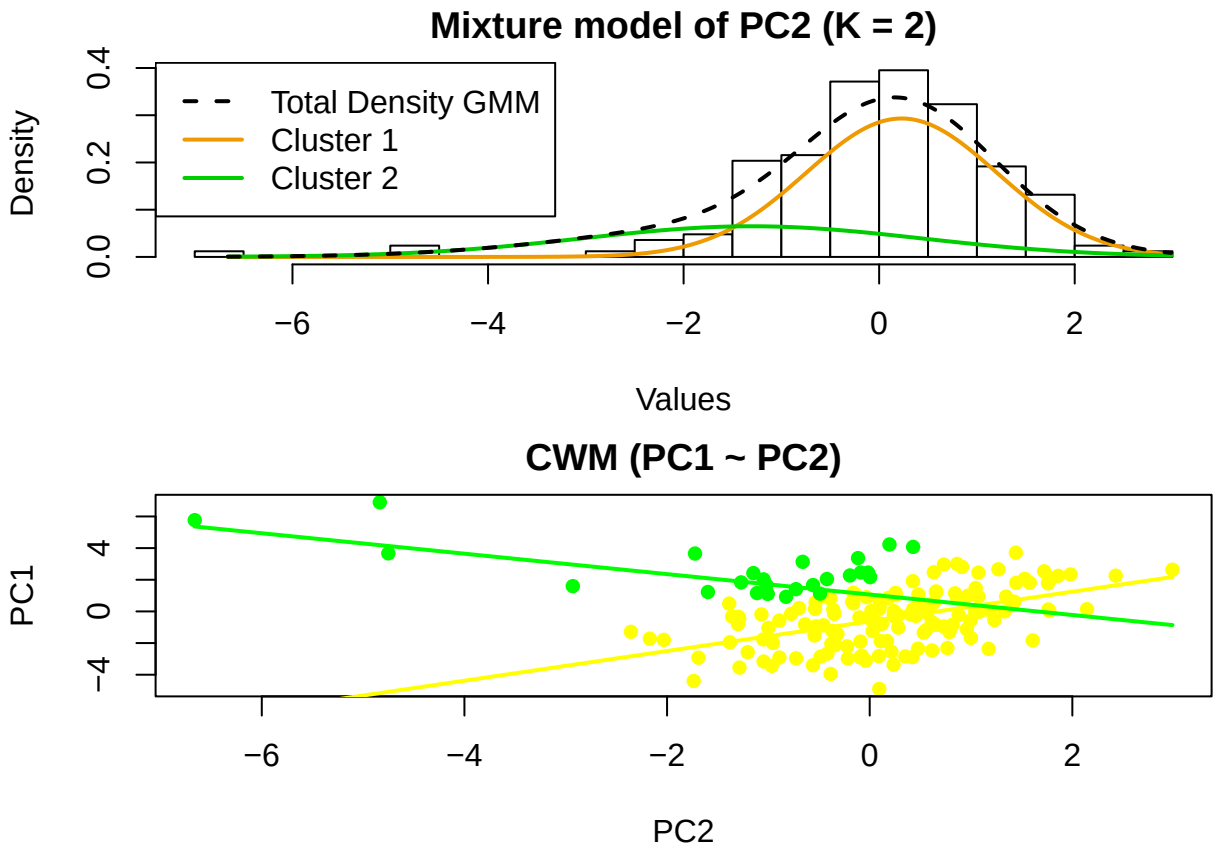
```
## ----------------------------------
## Best fitted model according to BIC
## ----------------------------------
##  loglikelihood   n df     BIC
##         -339.4 167  7 -714.63
##
## Clustering table:
##   1   2
## 142  25
##
## Prior: comp.1 = 0.7183, comp.2 = 0.2817
##
## Distribution used for GLM: gaussian(identity). Parameters:
##
## Component 1
##             Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -0.62787    0.12778 -4.9137 2.139e-06 ***
## PC2          0.93737    0.12686  7.3893 6.994e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma = 1.6271
##
## Component 2
##              Estimate Std. Error t value  Pr(>|t|)
## (Intercept)  1.065613   0.116922  9.1139 2.607e-16 ***
## PC2         -0.645570   0.069025 -9.3527 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## sigma = 1.4716
```

The **CWM output** shows two clusters, with Cluster 1 being much larger (142 observations) than Cluster 2 (25 observations). The model uses a **Gaussian distribution with an identity link**.

- **Cluster 1:** PC2 has a **strong positive effect** (0.94), meaning higher PC2 values lead to higher outcomes.
- **Cluster 2:** PC2 has a **strong negative effect** (-0.65), indicating an opposite trend.

The **small size of Cluster 2** raises questions about the stability of the classification.

```r
cluster_colors <- c("orange2", "green3")

## Paramethers
prior <- fit_cwm_pca$models[[1]]$prior
clusters <- fit_cwm_pca$models[[1]]$cluster
muX <- tapply(fit_cwm_pca$data$PC2, clusters, mean)
sdX <- tapply(fit_cwm_pca$data$PC2, clusters, sd)

x_vals <- seq(min(fit_cwm_pca$data$PC2), max(fit_cwm_pca$data$PC2), length.out = 500)

densities <- sapply(1:length(muX), function(i) {
  prior[i] * dnorm(x_vals, mean = muX[i], sd = sdX[i])
})

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

## Histogram
hist(fit_cwm_pca$data$PC2, breaks = 30, probability = TRUE,
     col = "white", border = "black",
     main = "Mixture model of PC2 (K = 2)",
     xlab = "Values", ylab = "Density")

for (i in 1:ncol(densities)) {
  lines(x_vals, densities[, i], col = cluster_colors[i], lwd = 2)
}

overall_density <- rowSums(densities)
lines(x_vals, overall_density, col = "black", lwd = 2, lty = 2)

legend("topleft", legend = c("Total Density GMM",
                             paste("Cluster", 1:length(prior))),
col = c("black", cluster_colors[1:length(prior)]), lwd = 2, lty = c(2, rep(1, length(prior))))

## Scatterplot
plot(fit_cwm_pca,
     xlab = "PC2", ylab = "PC1",
     main = "CWM (PC1 ~ PC2)", pch = 16)
```



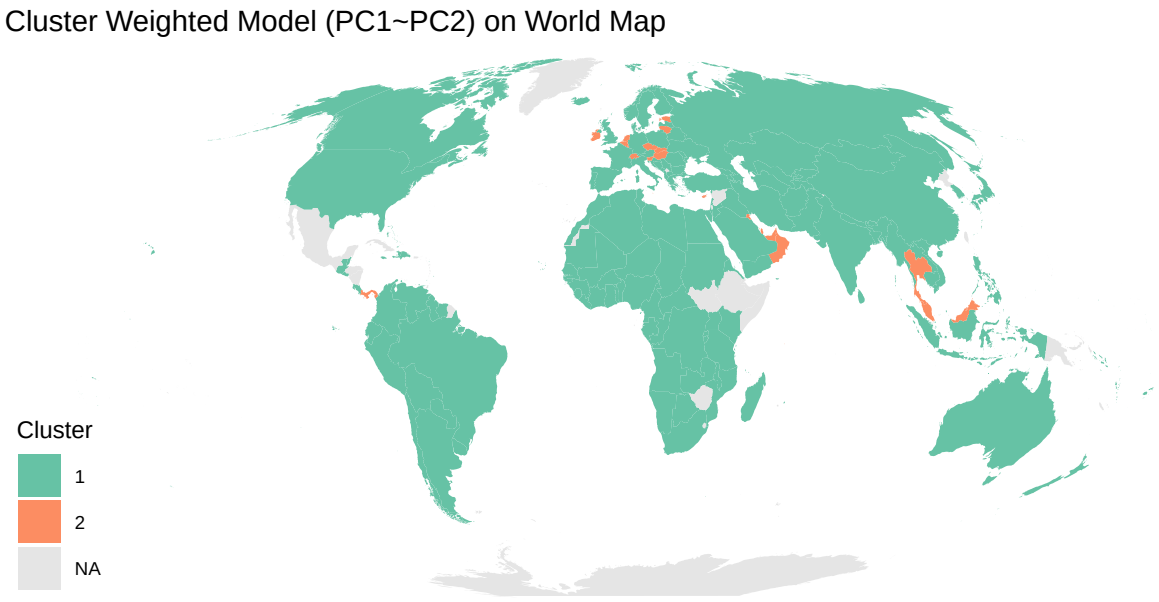
In this plot, the green cluster corresponds to the second cluster in both plots, whereas the green and magenta clusters together correspond to the first cluster. We can observe that one component is significantly more important than the others, particularly in the first cluster, which contains a much larger number of observations.

The two clusters generated exhibit an opposite relationship with PC1, which is primarily composed of socio-economic indices, while PC2 is largely influenced by import and export variables.

In my opinion, the previous applications of this method were more effective than this one. Here, the clusters overlap significantly, and using a linear function may not allow us to achieve a meaningful partition in this dimension.

```
cluster_data <- tibble(
  region  = rownames(fkmed3_res$clus),
  cluster = as.factor(clusters)
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
```

```
                          "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                          "United Kingdom" = "UK",
                          "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(long <= 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")

ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "Cluster Weighted Model (PC1~PC2) on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



This partition appears quite strange and, in my opinion, does not make much sense in terms of interpretability.

Conclusion

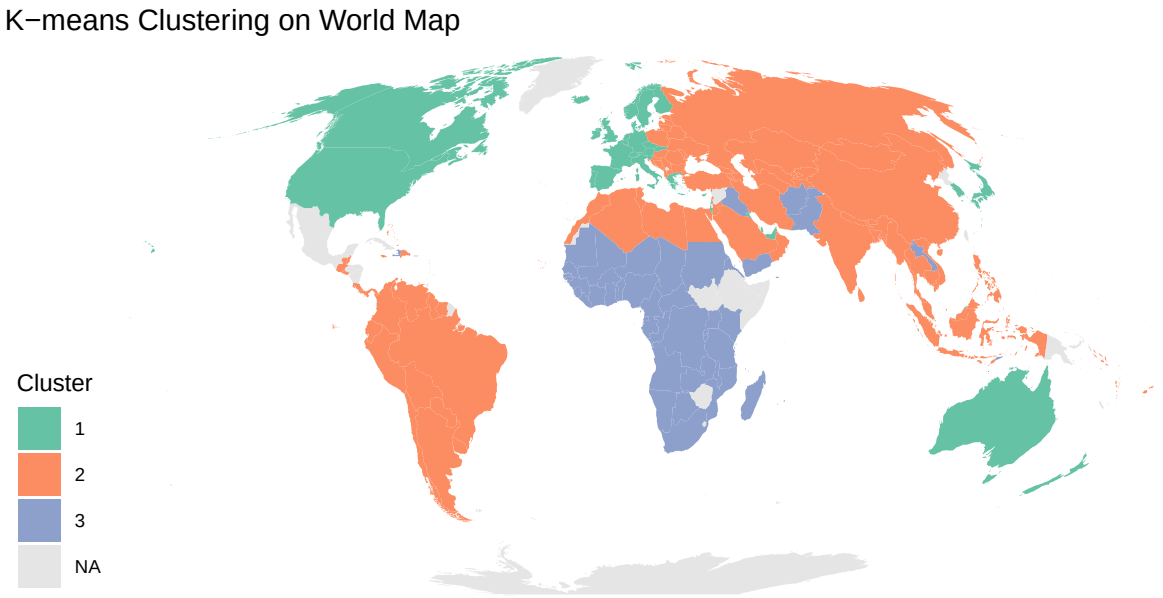
For this analysis, various **partitioning techniques** were used, starting with **hard partitions** like agglomerative clustering and K-means, followed by **soft partitions** such as fuzzy methods, model-based clustering, and finally **Clustering Weighted Models (CWM)**. The results varied significantly, with some partitions lacking clear interpretability.

In general, clustering was challenging because the data **did not exhibit well-separated clusters**, a common characteristic of **socio-economic data**. This led to different partitions depending on the method used, helping me better understand each approach.

I found the application of **sophisticated models** particularly interesting, even in a **simplified context**, as they highlighted the **flexibility and generalization** capabilities of these methods. In particular, **CWM was stimulating to explore**, despite not being the most suitable model for this dataset.

Overall, I believe **model-based clustering provided the best generalization**, offering **meaningful partitions** and fitting **multivariate normal models** to the clusters. However, to draw clear conclusions, **hard partitioning** is necessary, and in this regard, **K-means produced a solid result**.

```
cluster_data <- tibble(
  region  = names(km_res$cluster),
  cluster = as.factor(km_res$cluster)
)

cluster_data <- cluster_data %>%
  mutate(region = recode(region,
                         "Antigua and Barbuda" = "Antigua",
                         "Congo, Dem. Rep." = "Democratic Republic of the Congo",
                         "Congo, Rep." = "Republic of Congo",
                         "Cote d'Ivoire" = "Ivory Coast",
                         "Kyrgyz Republic" = "Kyrgyzstan",
                         "Lao" = "Laos",
                         "Macedonia, FYR" = "North Macedonia",
                         "Micronesia, Fed. Sts." = "Micronesia",
                         "Slovak Republic" = "Slovakia",
                         "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
                         "United Kingdom" = "UK",
                         "United States" = "USA"))

world_map <- map_data("world") %>%
  filter(!long > 180)

merged_map <- world_map %>%
  left_join(cluster_data, by = "region")


ggplot(merged_map, aes(fill = cluster, map_id = region)) +
  geom_map(map = world_map) +
  expand_limits(x = world_map$long, y = world_map$lat) +
  coord_map("mollweide") +
  theme_map() +
  labs(
    title = "K-means Clustering on World Map",
    fill = "Cluster"
  ) +
  scale_fill_brewer(palette = "Set2", na.value = "grey90")
```



With the following interpretation:

Cluster 1 (Red) This cluster likely represents **wealthier countries**. These countries score high on PC1, which is strongly associated with high GDP per capita (`gdpp`), income, and life expectancy (`life_expec`), while exhibiting low child mortality (`child_mort`) and total fertility rates (`total_fer`). On PC2, these countries also show moderate-to-high values, reflecting significant participation in international trade (`imports` and `exports`). Collectively, these traits suggest socio-economically advanced nations with robust global economic integration.

Cluster 2 (Green) The second cluster appears to represent **transitional countries**. These countries are positioned between Clusters 1 and 3, with medium scores on PC1 and PC2. They likely exhibit intermediate socio-economic characteristics, such as moderate GDP, income, and life expectancy, along with average trade activity. This group serves as a bridge, sharing some features of both wealthier and poorer countries.

Cluster 3 (Blue) This cluster is indicative of **poorer countries**. These countries have negative scores on PC1, reflecting low GDP, income, and life expectancy, paired with high levels of child mortality and fertility. Their low scores on PC2 further suggest limited involvement in international trade, with relatively low values for imports and exports. Overall, this group comprises nations facing significant socio-economic challenges.

Appendix

Below is a three-dimensional scatter plot (using the `plotly` library) that allows us to better visualize the cluster separation and data variability.

```
# pc <- prcomp(scaled_data)$x[, 1:3]
#
# colors <- rainbow(length(unique(mcl_res$classification)))
```

```
#
# plot_ly(
#   x = pc[,1],
#   y = pc[,2],
#   z = pc[,3],
#   color = as.factor(mcl_res$classification),
#   colors = colors,
#   type = "scatter3d",
#   mode = "markers") %>%
# layout(
#   scene = list(
#     xaxis = list(title = 'PC1'),
#     yaxis = list(title = 'PC2'),
#     zaxis = list(title = 'PC3')
#   )
# )
```

Below is a simple representation of three models (different from the previous ones), fitted in the PCA space. This shows how the bivariate model appears. Unfortunately, I was unable to use the model fitted in the original space, as I couldn't transform the parameters in a way that would allow for visualization of the model in the three-dimensional PCA space.

```
# pca_res <- prcomp(data, center = TRUE, scale. = TRUE)
# pca_data <- pca_res$x[, 1:3]
#
# mcl_res_pca <- Mclust(pca_data, G = 3)
#
# means <- mcl_res$parameters$mean
# sigmas <- mcl_res$parameters$variance$sigma
#
# x_seq <- seq(min(pca_data[,1]), max(pca_data[,1]), length.out = 50)
# y_seq <- seq(min(pca_data[,2]), max(pca_data[,2]), length.out = 50)
# grid <- expand.grid(x_seq, y_seq)
# colnames(grid) <- colnames(pca_data)[1:2]
#
# dens_list <- list()
# for (k in 1:3) {
#   dens_list[[k]] <- matrix(
#     dmvnorm(grid, mean = means[1:2, k], sigma = sigmas[1:2, 1:2, k]),
#     nrow = length(x_seq), ncol = length(y_seq)
#   )
# }
#
# p <- plot_ly()
# for (k in 1:3) {
#   p <- p %>% add_surface(
#     x = x_seq, y = y_seq, z = dens_list[[k]],
#     colorscale = 'Cividis',
#     opacity = 0.7,
#     showscale = FALSE
#   )
# }
#
# p <- p %>% layout(
```

```
#   title = "Probabilistic models G=3 (PCA)",
#   scene = list(
#     xaxis = list(title = "PC1"),
#     yaxis = list(title = "PC2"),
#     zaxis = list(title = "Density")
#   )
# )
#
# p
```

References

Ingrassia, Salvatore, Simona C. Minotti, and Antonio Punzo. 2014. "Model-Based Clustering via Linear Cluster-Weighted Models." *Computational Statistics &Amp; Data Analysis* 71 (March): 159–82. https://doi.org/10.1016/j.csda.2013.02.012.

Rohan. 2021. "Unsupervised Learning on Country Data." https://www.kaggle.com/datasets/rohan0301/unsupervised-learning-on-country-data.

Stasinopoulos, Mikis D., Robert A. Rigby, Gillian Z. Heller, Vlasios Voudouris, and Fernanda De Bastiani. 2017. *Flexible Regression and Smoothing Using GAMLSS in r*. Springer.