Chapter 9 DISTATIS

The notion of distance is essential because many statistical techniques are equivalent to the analysis of a specific distance table. For example, principal component analysis and metric multidimensional scaling analyze Euclidean distances, correspondence analysis deals with a $\chi 2$ distance matrix, and discriminant analysis is equivalent to using a Mahalanobis distance. To define a distance is equivalent to defining rules to assign positive numbers between pairs of objects. The most important distances for statistics are: Euclidean, generalized Euclidean (which include $\chi 2$ and Mahalanobis), Minkowsky (which include the sorting and the symmetric difference distances), and the Hellinger distances.

Distatis is a procedure that combines bootstraps estimation (to estimate the variability of experimental conditions)it integrates the distances matrices and represents results as maps. Reliability estimates are expressed as tolerance intervals which reflect the accuracy of the assignments of scans to experimental categories and as confidence intervals which generalize standard hypothesis testing.

The purpose of this study is to determine how trained and untrained listeners sort western classical melodies into clusters based on perceived similarities. Listerners at three expertise levels sorted MIDI and natural excerpts from the piano music of Bach, Mozart, and Beethoven. We analyzed the data using DISTATIS which showed an effect of composer with both MIDI and natural stimuli and an effect of pianist with natural stimuli. However there was only a weak effect of music training.

Task Participants sorted excerpts into three clusters according to whether a single composer could have written the pieces in the group. Participants could have listen to each excerpt as many times as they wanted to. We investigated the effects of composer painist and music training on sorting. To analyze the data we applied DISTATIS, a recent adaption of multi-dimensional scaling specifically adapted to reveal the perceived dissimilarity among items, as well as to investigate group differences.

```
knitr::opts_chunk$set(echo = TRUE)
rm(list = ls())
graphics.off()
```

9.1 DATASET

Rows: [composer].[conductor].[ID]

Columns: participants

Row # 38: music experience

Paino music from Bach Mozart Beethoven 36 excerpts from CD recordings with 3 from each composer by each of 4 pianists: Arrau, Barenboim Pires Richter- Excerpts were 9 to 15 s long. We presented the stimuli as audio icons arranged randomly on a PowerPoint slide.

We cateogrize the level of expertise as follows:

Musicians a) musical training =5 years and above N=10 Moderate

musicians b) musical training = 1 to 4 years N= 17

Nonmusicians c) musical training = less than 1 year N= 10.

```
library(Matrix)
```

```
##
## Attaching package: 'Matrix'
```

```
## The following object is masked from 'package:tidyr':
##
##     expand
```

```
library(factoextra)
```

```
## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13E
```

```
#install.packages("factoextra")
library(DistatisR)
```

```
## Loading required package: car
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

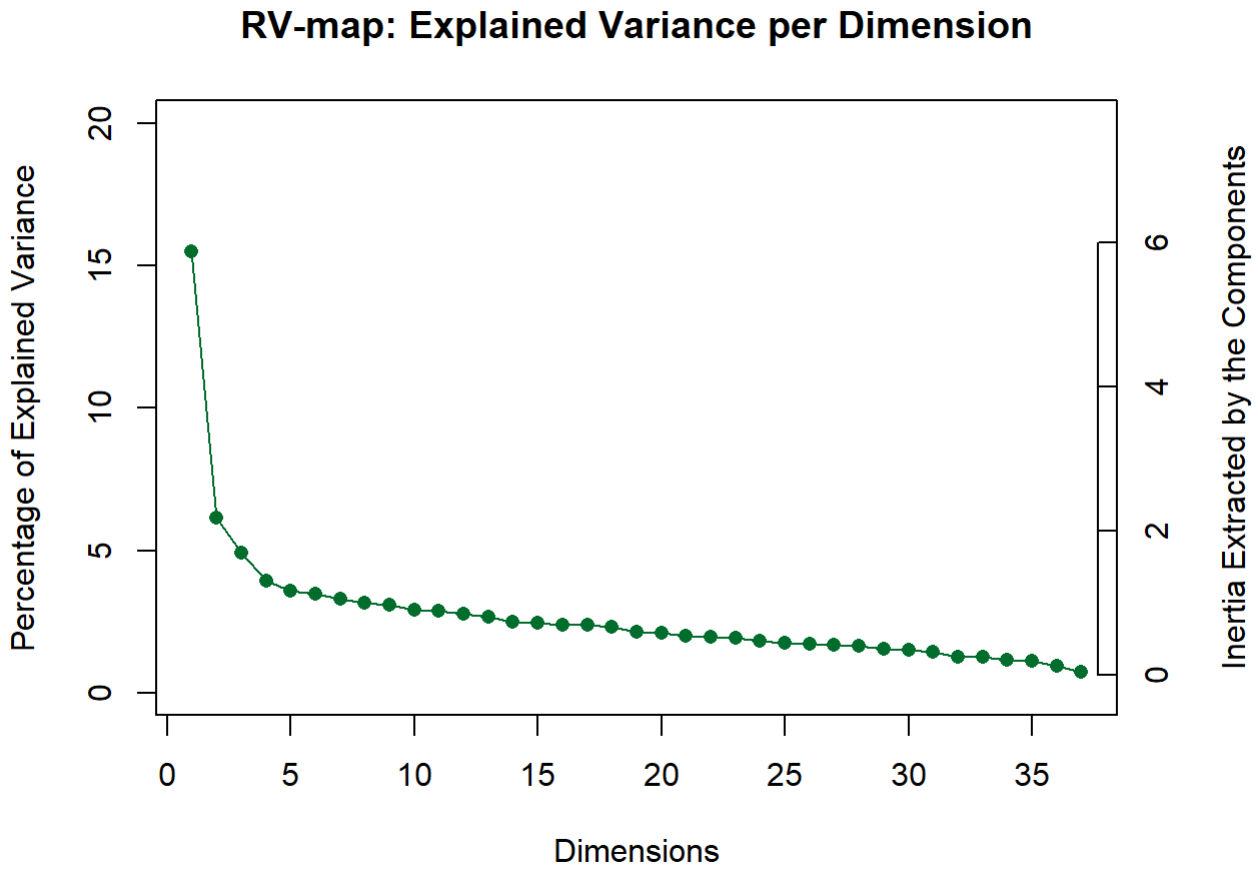
```r
library(PTCA4CATA)
library(prettyGraphs)
library(ExPosition)
Raw_Data <- read.csv('natural36_constrained_forR(9).csv', row.names = 1)
Sorting_Data <- Raw_Data[-38,]
Design_Data <- Raw_Data[38,]
Description <- Raw_Data
nVar <- ncol(Description)
k <- 1
desc <- Description[,k ]
# Create a 0/1 group matrix with ExPosition::makeNominalData()
nominal <- makeNominalData(as.data.frame(desc))
# get the colors
color4Judges.list <- prettyGraphs::createColorVectorsByDesign(nominal)
DistanceCube <- DistanceFromSort(Sorting_Data)
resDistatis <- distatis(DistanceCube)
```

9.2 SCREE PLOT.

```r
# Get the factors from the Cmat analysis
G <- resDistatis$res4Cmat$G
# Compute the mean by groups of HJudges
Means.tmp <- aggregate(G, list(desc), mean)
Means <- Means.tmp[,2:ncol(Means.tmp )]
rownames(Means) <- Means.tmp[,1]
# Get the bootstrap estimates
BootCube <- PTCA4CATA::Boot4Mean(G, design = desc,
niter = 100,
suppressProgressBar = TRUE)
 #head(BootCube)



#PROJECTVOC IS NOT WORKING ASK JU



F_j <- resDistatis$res4Splus$PartialF
alpha_j <- resDistatis$res4Cmat$alpha
# create the groups of Judges
#groupsOfJudges <- substr(names(alpha_j),1,1)
groupsOf <- desc
code4Groups <- unique(groupsOf)
nK <- length(code4Groups)
# initialize F_K and alpha_k
F_k <- array(0, dim = c(dim(F_j)[[1]], dim(F_j)[[2]],nK))
dimnames(F_k) <- list(dimnames(F_j)[[1]],
dimnames(F_j)[[2]], code4Groups)
alpha_k <- rep(0, nK)
names(alpha_k) <- code4Groups
Fa_j <- F_j
# A horrible loop
for (j in 1:dim(F_j)[[3]]){ Fa_j[,,j] <- F_j[,,j] * alpha_j[j] }

for (k in 1:nK){
```

```
lindex <- groupsOf == code4Groups[k]
alpha_k[k] <- sum(alpha_j[lindex])
F_k[,,k] <- (1/alpha_k[k])*apply(Fa_j[,,lindex],c(1,2),sum)
}
```

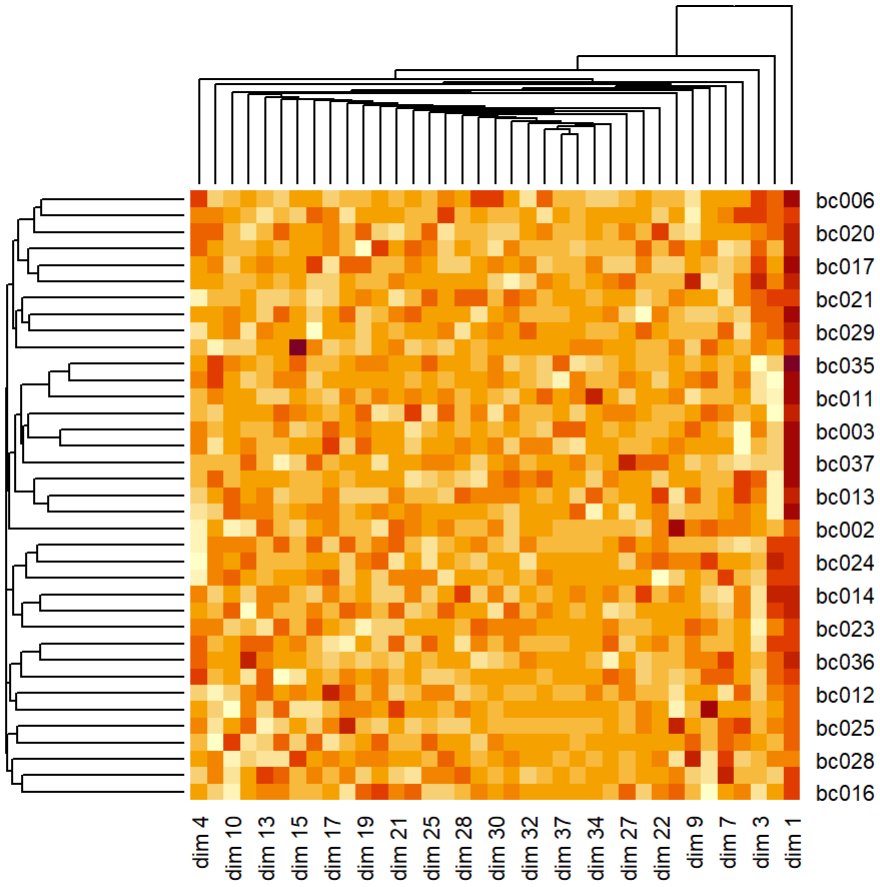
```
# 5.A. A scree plot for the RV coef. Using standard plot (PTCA4CATA)
scree.rv.out <- PlotScree(ev = resDistatis$res4Cmat$eigValues,
title = "RV-map: Explained Variance per Dimension")
```




```
a1.Scree.RV <- recordPlot() # Save the plot
```

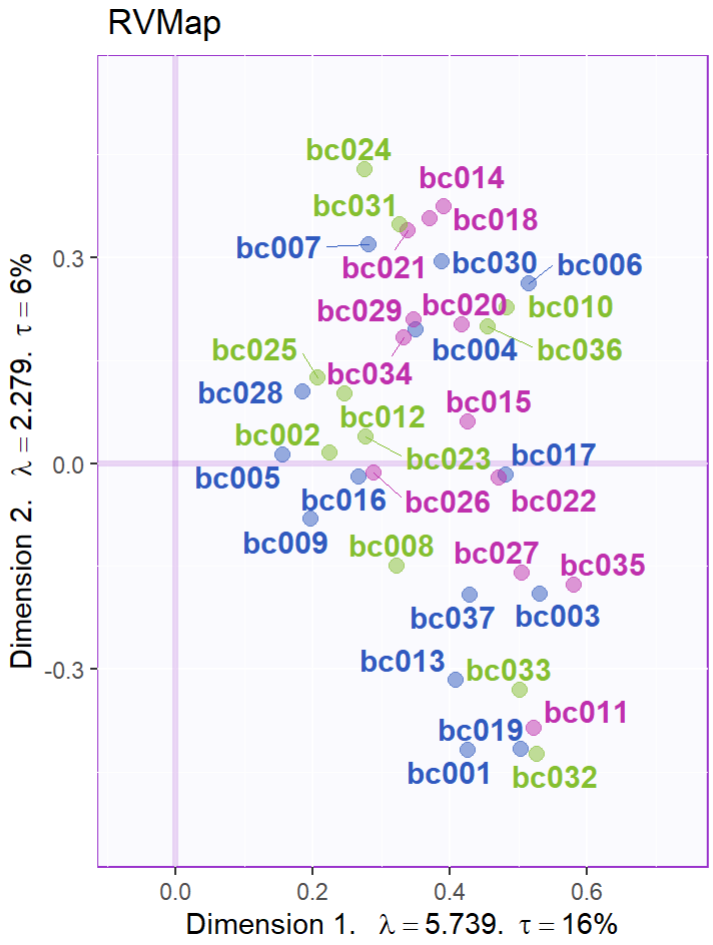
9.3 HEATMAP

```
# Create the layers of the map
gg.rv.graph.out <- createFactorMap(X = resDistatis$res4Cmat$G,
axis1 = 1, axis2 = 2,
title = " RVMap",
col.points = color4Judges.list$oc,
col.labels = color4Judges.list$oc)
# create the labels for the dimensions of the RV map
labels4RV <- createxyLabels.gen(
lambda = resDistatis$res4Cmat$eigValues ,
tau = resDistatis$res4Cmat$tau,
axisName = "Dimension ")
# # Create the map from the layers
# Here with lables and dots
a2a.gg.RVmap <- gg.rv.graph.out$zeMap + labels4RV
# Here with colored dots only
a2b.gg.RVmap <- gg.rv.graph.out$zeMap_background +
gg.rv.graph.out$zeMap_dots + labels4RV

heatmap(resDistatis$res4Cmat$G)
```



9.4 THE ASSESSOR MATRIX

```
print(a2a.gg.RVmap )
```



##BOOTSTRAP INTERVALS

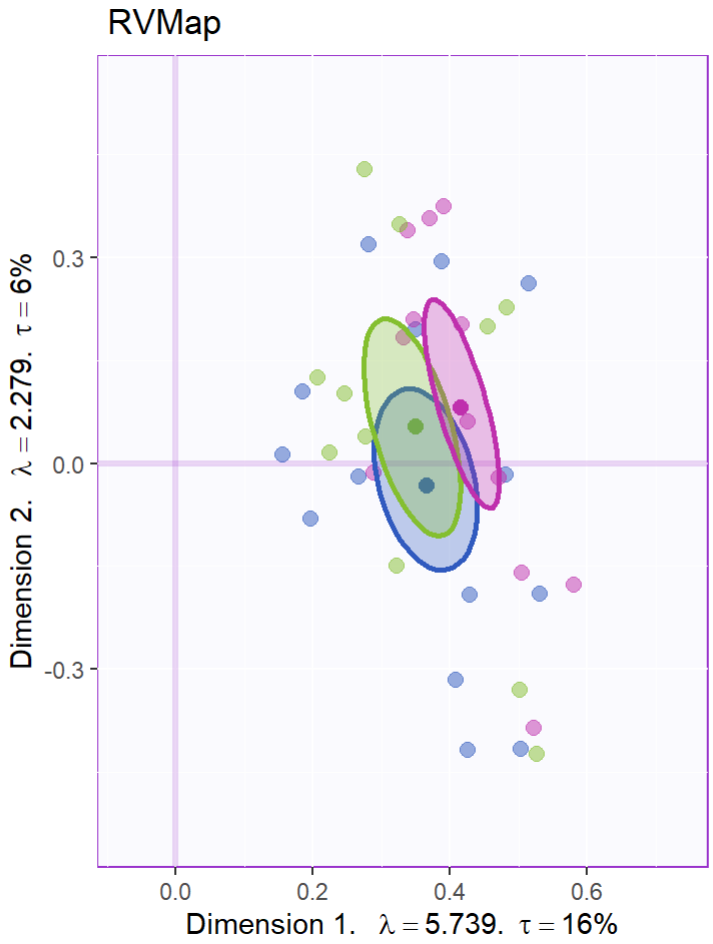
```
# First the means
# A tweak for colors
in.tmp <- sort(rownames(color4Judges.list$gc), index.return = TRUE)$ix
col4Group <- color4Judges.list$gc[in.tmp]
#
gg.rv.means <- PTCA4CATA::createFactorMap(Means,
axis1 = 1, axis2 = 2,
constraints = gg.rv.graph.out$constraints,
col.points = col4Group ,
alpha.points = 1, # no transparency
col.labels = col4Group)
#
dimnames(BootCube$BootCube)[[2]] <-
paste0('dim ',1: dim(BootCube$BootCube)[[2]])
#c('Dim1','Dim2')
9
```

```
## [1] 9
```

```
GraphElli.rv <- MakeCIEllipses(BootCube$BootCube[,1:2,],
names.of.factors = c("dim 1","dim 2"),
col = col4Group,
p.level = .95)
a2d.gg.RVMap.CI <- a2b.gg.RVmap + gg.rv.means$zeMap_dots + GraphElli.rv

knitr::kable(Means[,1])
```

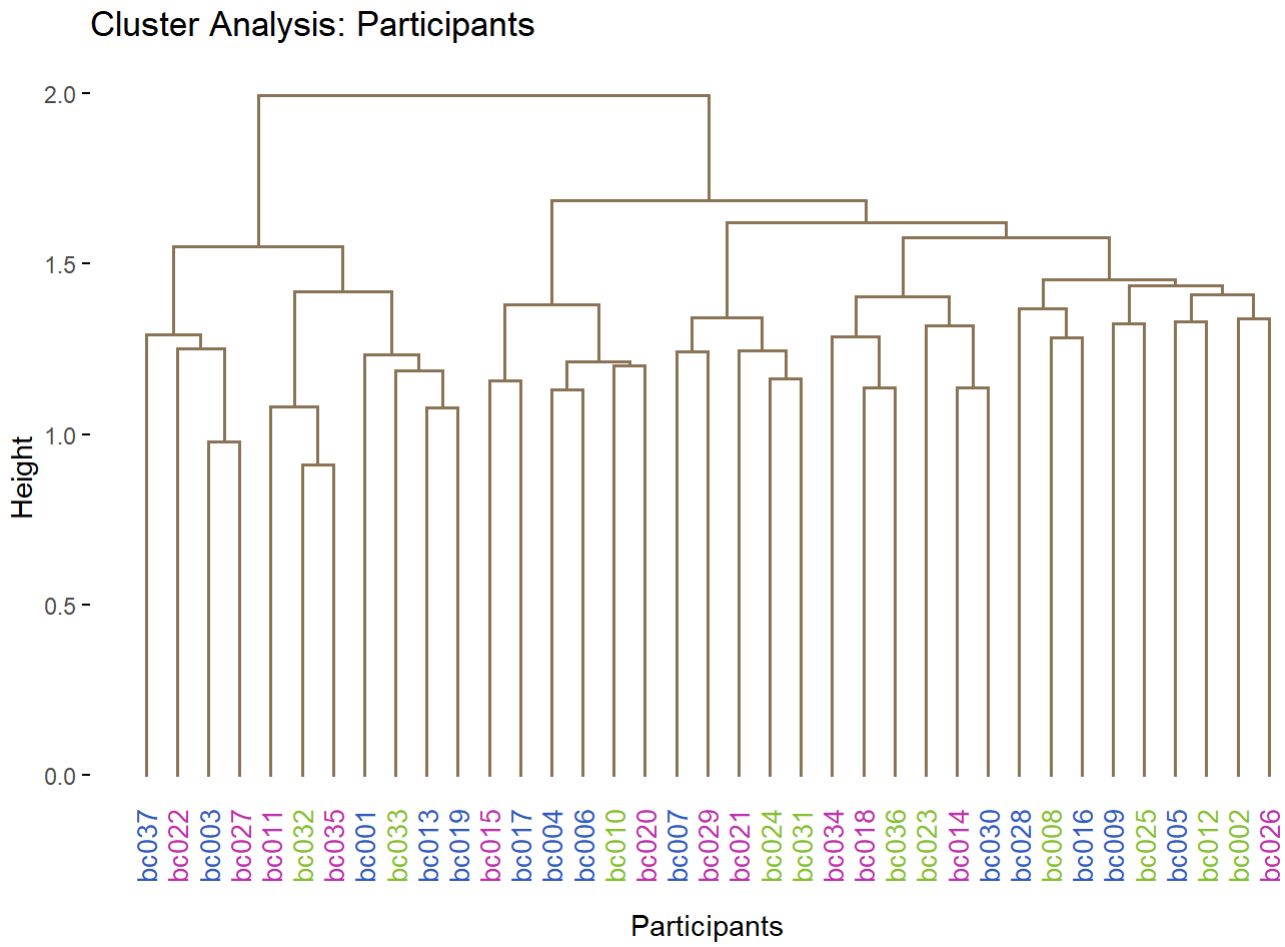
	x
	0.3657973
	0.3498342
	0.4159971

```
print(a2d.gg.RVMap.CI )
```

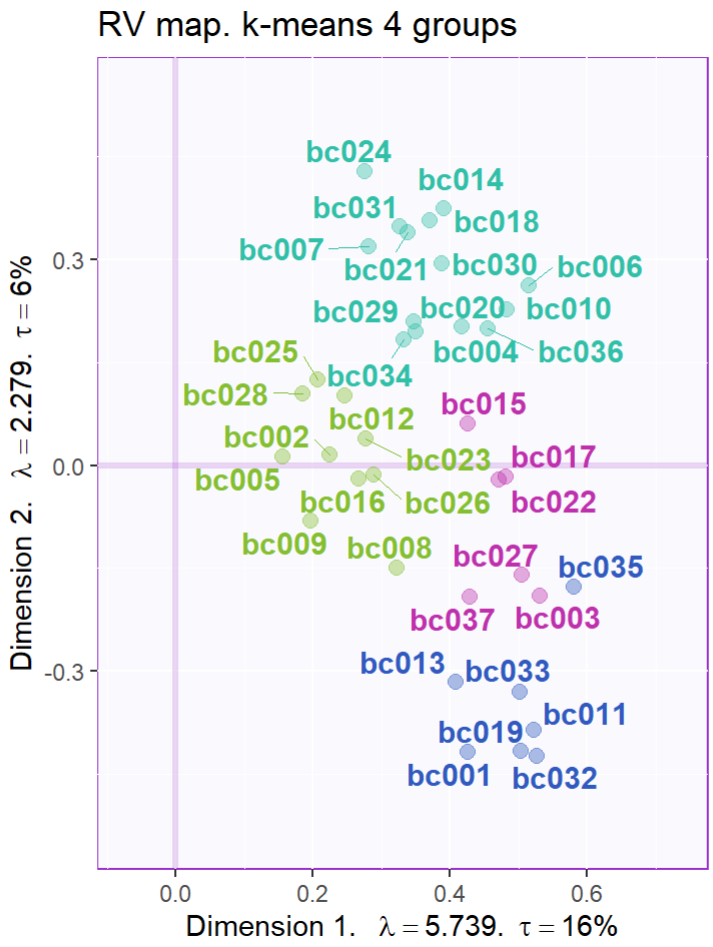


##CLUSTER ANALYSIS OF PARTICIPANTS/RV map. k-means 4 groups/SCREE PLOT

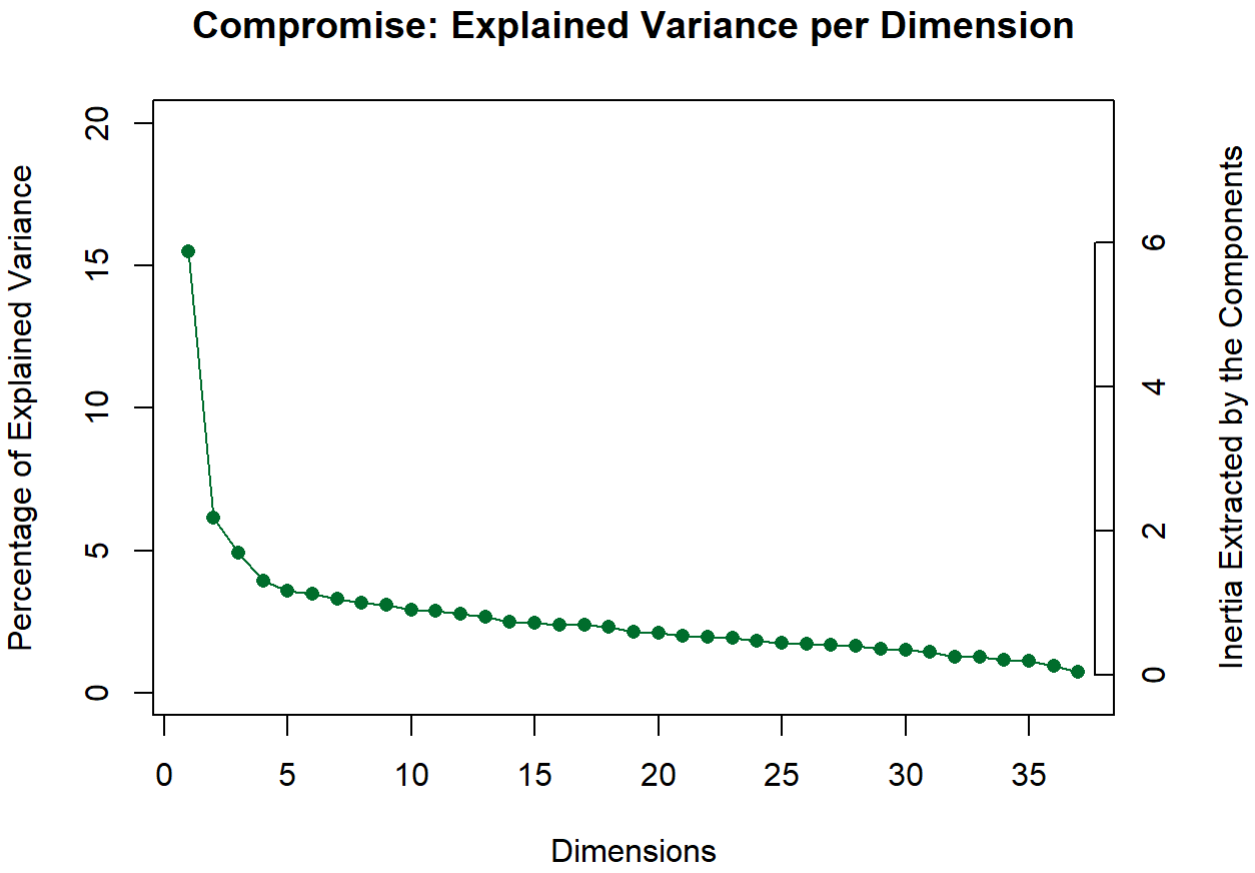
```
D <- dist(resDistatis$res4Cmat$G, method = "euclidean")
fit <- hclust(D, method = "ward.D2")
a05.tree4participants <- fviz_dend(fit, k = 1,
k_colors = 'burlywood4',
label_cols = color4Judges.list$oc[fit$order],
cex = .7, xlab = 'Participants',
main = 'Cluster Analysis: Participants')
print(a05.tree4participants)
```


Cluster Analysis: Participants
Height
2.0
1.5
1.0
0.5
0.0
bc037
bc022
bc003
bc027
bc011
bc032
bc035
bc001
bc033
bc013
bc019
bc015
bc017
bc004
bc006
bc010
bc020
bc007
bc029
bc021
bc024
bc031
bc034
bc018
bc036
bc023
bc014
bc030
bc028
bc008
bc016
bc009
bc025
bc005
bc012
bc002
bc026
Participants

```r
# First plain k-means
set.seed(42)
participants.kMeans <- kmeans(x = G , centers = 4)
#_____________________________________________________________________
# Now to get a map by cluster:
col4Clusters <- createColorVectorsByDesign(

makeNominalData(
as.data.frame(participants.kMeans$cluster) ))
baseMap.i.km <- PTCA4CATA::createFactorMap(G,
title = "RV map. k-means 4 groups",
col.points = col4Clusters$oc,
col.labels = col4Clusters$oc,
constraints = gg.rv.graph.out$constraints,
alpha.points = .4)
a06.aggMap.i.km <- baseMap.i.km$zeMap_background +
baseMap.i.km$zeMap_dots + baseMap.i.km$zeMap_text + labels4RV


baseMap.i.km <- PTCA4CATA::createFactorMap(G,
title = "RV map. k-means 4 groups",
col.points = col4Clusters$oc,
col.labels = col4Clusters$oc,
constraints = gg.rv.graph.out$constraints,
alpha.points = .4)
a06.aggMap.i.km <- baseMap.i.km$zeMap_background +
baseMap.i.km$zeMap_dots + baseMap.i.km$zeMap_text + labels4RV
print(a06.aggMap.i.km)
```




```
scree.S.out <- PlotScree(
ev = resDistatis$res4Cmat$eigValues,#########################ERROR ERROR
title = "Compromise: Explained Variance per Dimension")
```



9.5 I-MAP

```
# 4.1 Get the bootstrap factor scores (with default 1000 iterations)
BootF <- BootFactorScores(resDistatis$res4Splus$PartialF)
```

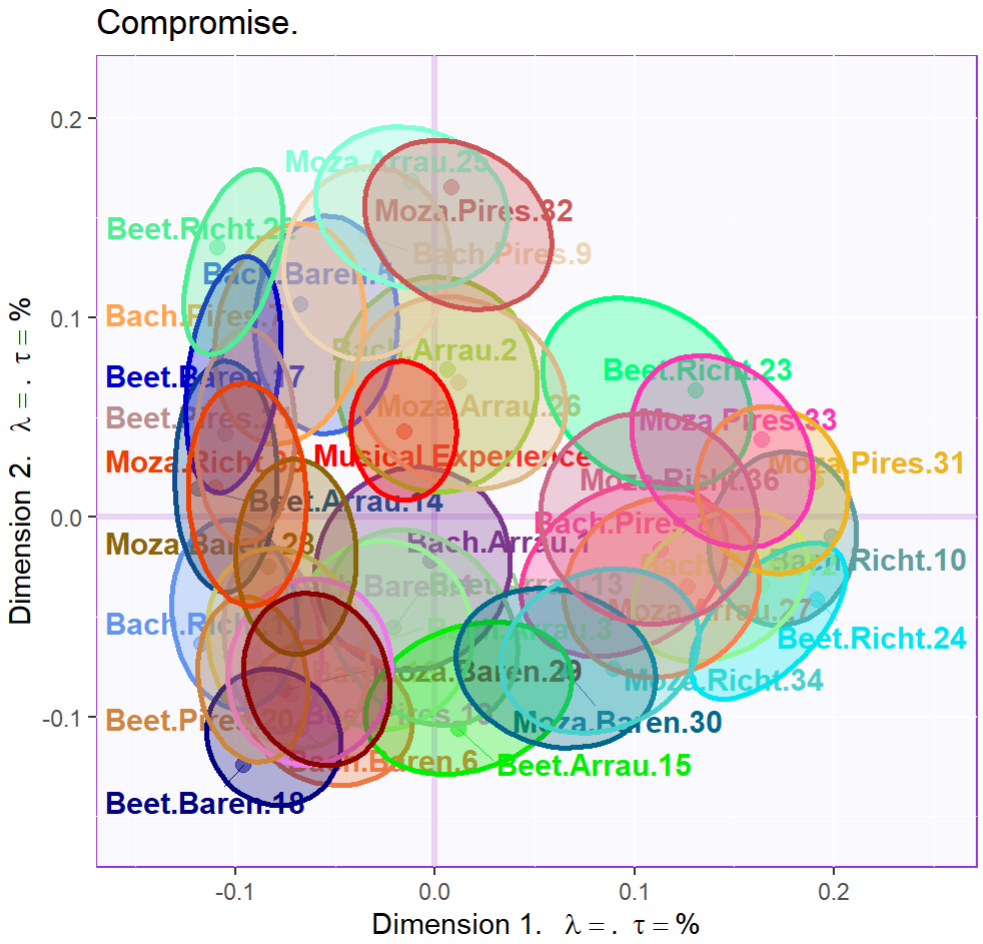
```
## [1] Bootstrap On Factor Scores. Iterations #:
## [2] 1000
```

```r
# 5.2 a compromise plot
# General title for the compromise factor plots:
genTitle4Compromise = 'Compromise.'
# To get graphs with axes 1 and 2:
h_axis = 1
v_axis = 2
# To get graphs with say 2 and 3
# change the values of v_axis and h_axis
color4Products <- # Create color for the Products from prettyGraph
prettyGraphsColorSelection(n.colors = nrow(resDistatis$res4Splus$F))
gg.compromise.graph.out <- createFactorMap(resDistatis$res4Splus$F,
axis1 = h_axis,
axis2 = v_axis,
title = genTitle4Compromise,
col.points = color4Products ,
col.labels = color4Products)

gg.compromise.graph.out <- createFactorMap(resDistatis$res4Splus$F,
                                           axis1 = h_axis,
                                           axis2 = v_axis,
                                           title = genTitle4Compromise,
                                           col.points = color4Products ,
                                           col.labels = color4Products)
label4S <- createxyLabels.gen(x_axis = h_axis, y_axis = v_axis,lambda = Design_Data, tau =



b2.gg.Smap <- gg.compromise.graph.out$zeMap + label4S
#
# 5.4 a bootstrap confidence interval plot
# 5.3 create the ellipses
gg.boot.graph.out.elli <- MakeCIEllipses(
data = BootF[,c(h_axis,v_axis),],
names.of.factors =
c(paste0('Factor ',h_axis),
paste0('Factor ',v_axis)),
```

```
  col = color4Products,
  )
  # Add ellipses to compromise graph
  b3.gg.map.elli <- gg.compromise.graph.out$zeMap + gg.boot.graph.out.elli + label4S
  print(b3.gg.map.elli)
```

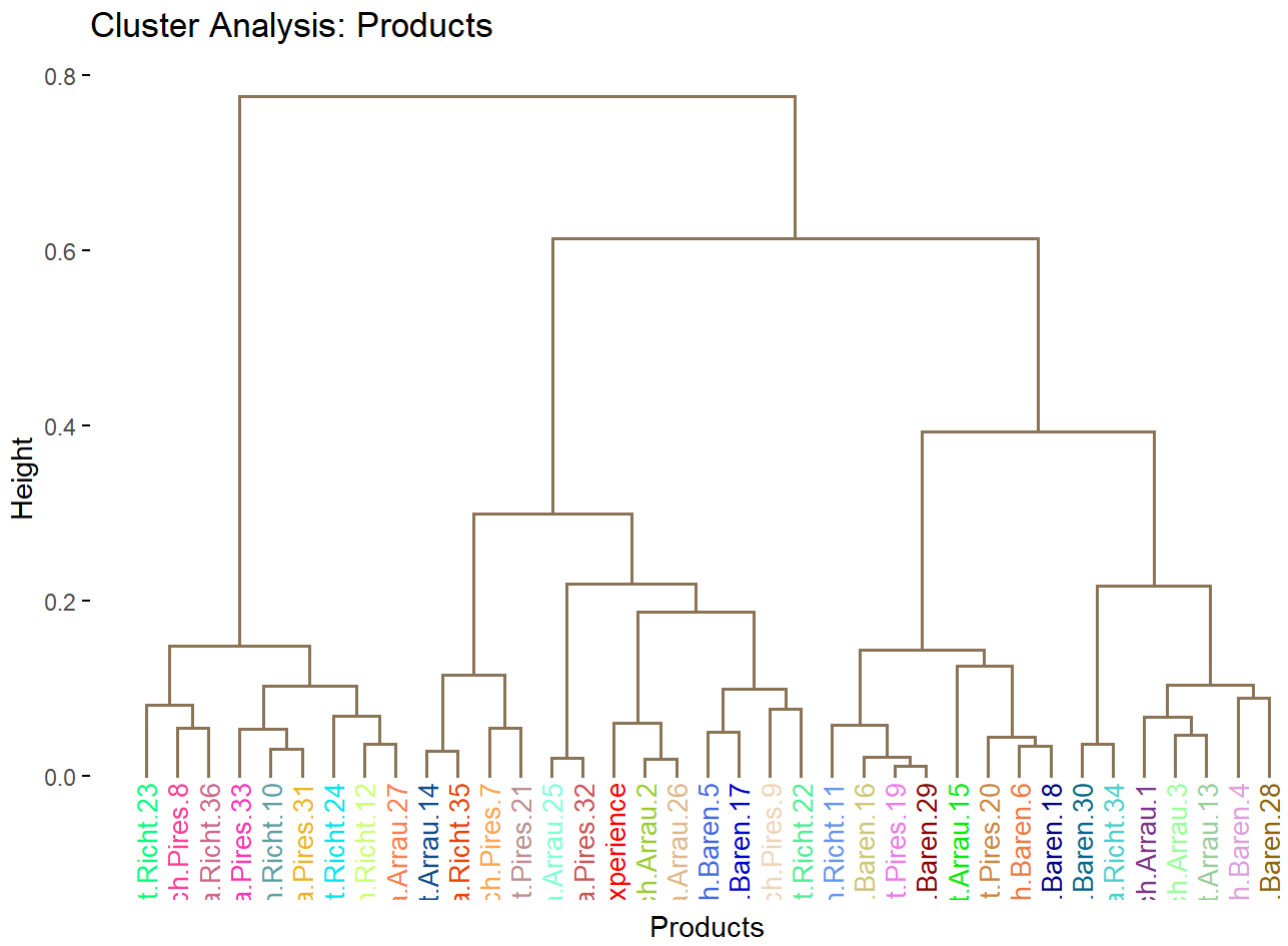
```
## Warning: Removed 3 rows containing non-finite values (stat_ellipse).
```

```
## Warning: Removed 3 rows containing non-finite values (stat_ellipse).
```



9.6 CLUSTERS ANALYSIS

```
nFac4Prod = 3
D4Prod <- dist(resDistatis$res4Splus$F[,1:nFac4Prod], method = "euclidean")
fit4Prod <- hclust(D4Prod, method = "ward.D2")
b3.tree4Product <- fviz_dend(fit4Prod, k = 1,
k_colors = 'burlywood4',
label_cols = color4Products[fit4Prod$order],
cex = .7, xlab = 'Products',
main = 'Cluster Analysis: Products')
print(b3.tree4Product)
```



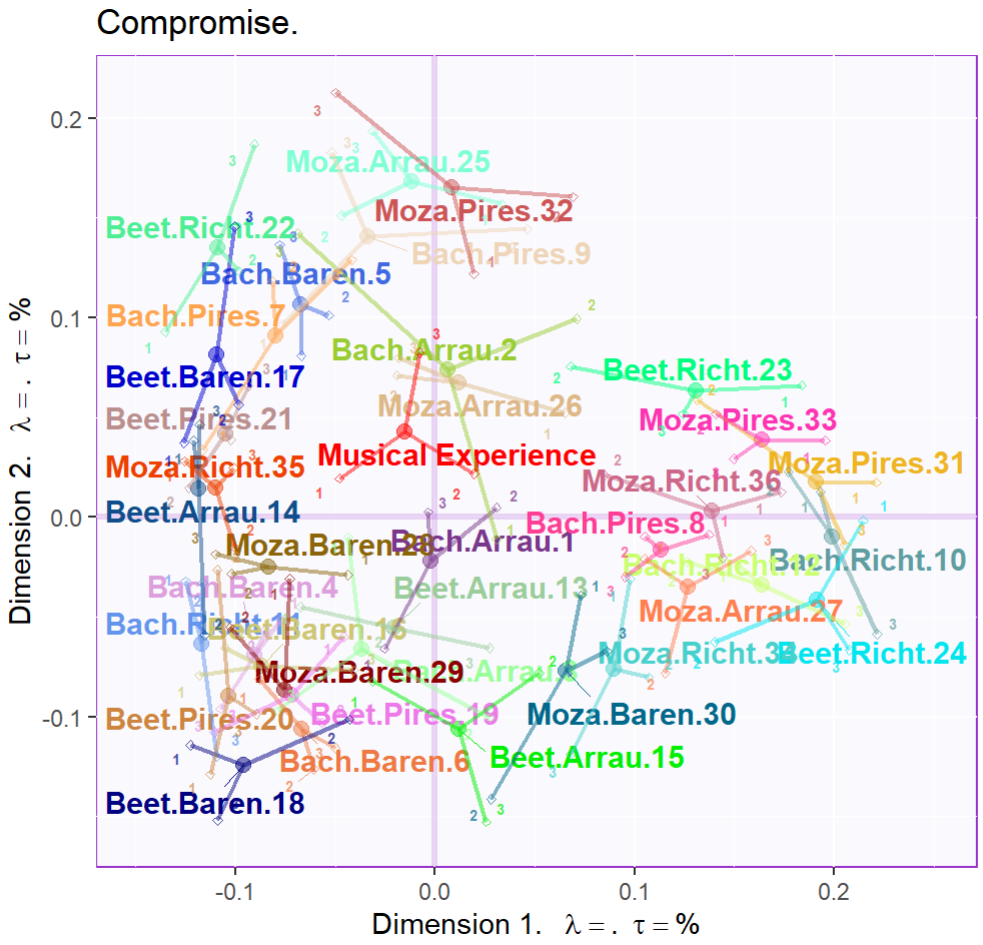
##PARTIAL FACTOR SCORES

```
# get the partial map
map4PFS <- createPartialFactorScoresMap(
factorScores = resDistatis$res4Splus$F,
partialFactorScores = F_k,
axis1 = 1, axis2 = 2,
colors4Items = as.vector(color4Products),
names4Partial = dimnames(F_k)[[3]], #
font.labels = 'bold')
d1.partialFS.map.byProducts <- gg.compromise.graph.out$zeMap +map4PFS$mapColByItems + label
d2.partialFS.map.byCategories <- gg.compromise.graph.out$zeMap +
map4PFS$mapColByBlocks + label4S
print(d1.partialFS.map.byProducts )
```

```
## Warning: Removed 1 rows containing missing values (geom_segment).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing missing values (geom_text_repel).
```

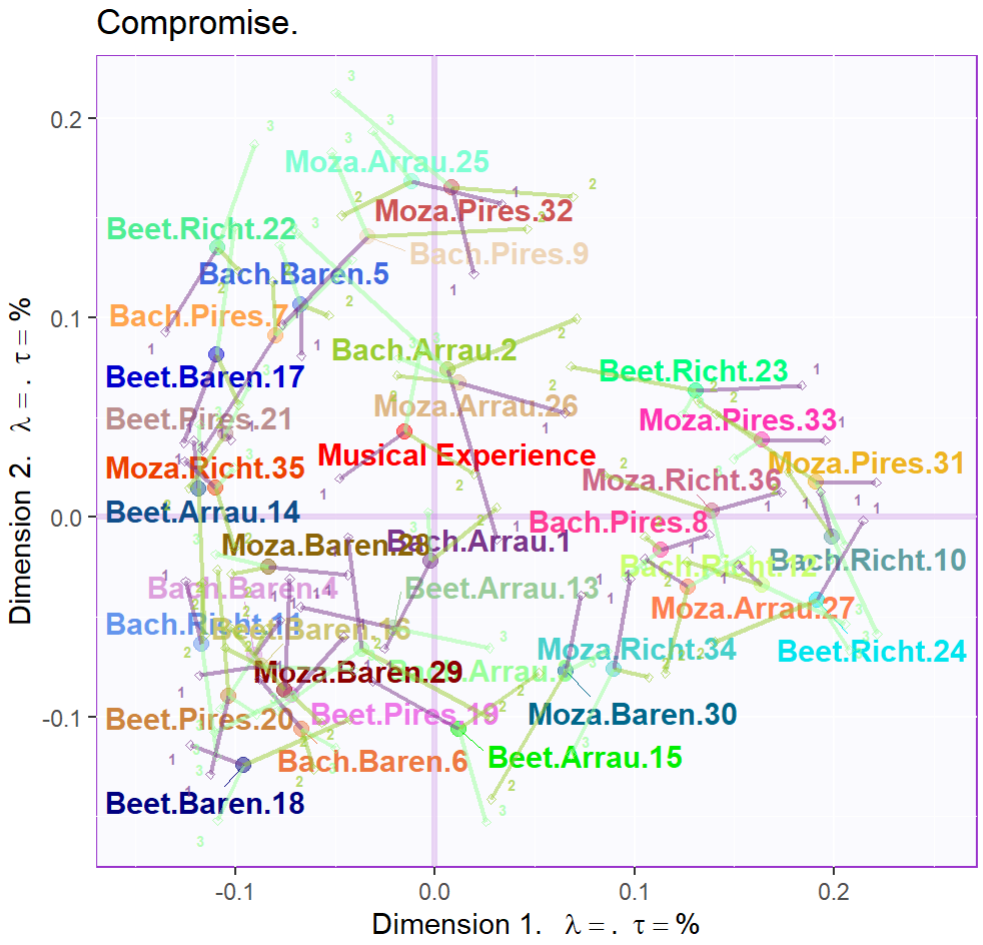



```
print(d2.partialFS.map.byCategories)
```

```
## Warning: Removed 1 rows containing missing values (geom_segment).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```

```
## Warning: Removed 1 rows containing missing values (geom_text_repel).
```



9.7 SUMMARY.

Participants were able to strongly differentiate Mozarts excerpts from Beethoven, with bach falling in between those two and Richters performances of the three composers were clustered relatively close to the Mozart region of the solution, indicating their clarity and balance: in contrast, those of Barenboim were clustered in the Beethoven region, indicating their sumptuousness and passion.