FactoQGIS script

Florent Demoraes, December 2018

This is the content of the file named "**Typological_Analysis_PCA_PCA_and_HAC.rsx**" that must be stored in the following folder: C:\Users\...\...\ qgis2\processing\rscripts

The script is documented both in English and French.

```
##FactoQGIS=group
##Language=selection Francais;English
##Working_directory_Mandatory=Folder
##Layer=Vector
##Row_name=Field Layer
##Active_variables_Must_be_numeric=multiple Field Layer
##Scale_data=selection TRUE;FALSE
##Number_of_axes_to_be_kept_for_PCA=Number 5
##Number_of_axes_to_be_kept_for_HAC=Number 2
##Number_of_clusters_to_be_kept_for_HAC=Number 5
##Metric_to_be_used_to_build_the_tree=selection euclidean;manhattan
##Aggregation_method_to_be_used_to_define_clusters=selection
ward;average;single;complete
##Eigen_Value_Table=output table
##Variable_Coordinates_Table=output table
##Layer_with_Clusters=output vector

# Chargement des packages necessaires. NB les packages de R qui sont appeles dans les
scripts doivent avoir ete installes au prealable dans le logiciel R (ou via R Studio)
# Loading required packages. Please note that they must have previously been installed in R
# Pour executer l'ACP
# To perform the PCA
library(FactoMineR)
# Pour produire des graphiques esthetiques issus de l'ACP
# To produce aesthetic graphs and plots deriving from the PCA
library(factoextra)
# Pour traiter les chaines de caracteres
# To handle character strings
library(stringr)
# Pour exporter les resultats vers Excel
# To save the results to Excel
library(openxlsx)
# Pour generer un fichier html contenant les resultats
# To generate a html file with the results
library(R2HTML)
# Pour creer une matrice restituant la qualite de representation des variables (cos2 des
variables sur toutes les dimensions)
# To create a matrix to visualize the quality of representation for variables (cos2 of the
variables on every dimension)
library(corrplot)
```

```
# Recuperation et formatage des valeurs saisies dans la boite de dialogue par l'utilisateur
# Recovering and formatting the values entered into the dialog box by the user

# Pour recuperer dans une chaine de caracteres le chemin de l'espace de travail saisi par l'utilisateur
# To assign to a character string the path of the working directory entered by the user
Working_Directory <- as.character(Working_directory_Mandatory)

# Pour specifier l'espace de travail saisi par l'utilisateur
# To set working directory entered by the user
setwd(Working_Directory)

# Pour recuperer dans une chaine de caracteres le champ choisi par l'utilisateur contenant les entetes de ligne (identifiant des unites spatiales)
# To assign to a character string the user-selected field containing the row names (spatial units identifier)
Row_name<- as.character(Row_name)

# Pour recuperer dans une chaine de caracteres les variables actives quantitatives selectionnees par l'utilisateur
# To assign to a character string the quantitative active variables selected by the user
ListVar<-as.character(Active_variables_Must_be_numeric)

# Pour creer un vecteur avec le nom des variables actives quantitatives selectionnees par l'utilisateur
# To create a vector with the name of the quantitative active variables selected by the user
ListVar<-unlist(str_split(ListVar, ";"))

# Pour enregister le nombre de facteurs a garder pour l'ACP saisi par l'utilisateur
# To save the number of factors entered by user to be kept for the PCA
nf<-as.numeric(Number_of_axes_to_be_kept_for_PCA)

# Pour enregister le nombre de classes a garder pour la CAH saisi par l'utilisateur
# To save the number of clusters entered by user to be kept for the HAC
Nb_clust<-as.numeric(Number_of_clusters_to_be_kept_for_HAC)

# Pour enregister le nombre de facteurs a garder pour la CAH saisi par l'utilisateur
# To save the number of factors entered by user to be kept for the HAC
HAC_nf<-as.numeric(Number_of_axes_to_be_kept_for_HAC)

# Pour creer un parametre booleen correspondant aux choix de l'utilisateur de centrer et de reduire ou non les donnees
# To create a boolean parameter which translates the choice set by the user to scale and center or not the data
Scale_data
Scales<-c("TRUE","FALSE")
Scales
Scale<-Scales[Scale_data+1]
Scale
```

```
if (Scale == 'TRUE'){scale <- 1} else {scale <- 0}
scale

# Pour creer un parametre correspondant a la methode d'agregation a retenir afin de constituer les classes
# To create a parameter which translates the method of aggregation set by the user to define clusters
Aggregation_method_to_be_used_to_define_clusters
methodes<-c("ward","average","single","complete")
methodes
methode<-methodes[Aggregation_method_to_be_used_to_define_clusters+1]
methode

# Pour creer un parametre correspondant au type de distance a retenir afin de construire le dendrogramme
# To create a parameter which translates the distance type set by the user to build the tree
Metric_to_be_used_to_build_the_tree
distances<-c("euclidean","manhattan")
distances
distance<-distances[Metric_to_be_used_to_build_the_tree+1]
distance

# Importation du jeu de donnee
# To import the dataset
dataset<-as.data.frame(Layer)
# Pour rajouter la colonne Row_name au jeu de donnee
# To add Row_name column to the dataset
dataset<-cbind(dataset, Layer[[Row_name]])
# Pour renommer la colonne Row_name en ID
# To rename Row_name column as ID
names(dataset)[names(dataset) == "Layer[[Row_name]]"] <- "ID"
# Pour specifier l'attribut ID comme rowname
# To set ID attribute as rowname
row.names(dataset)<-dataset$ID
colnames(dataset)

# Creation d'un sous-ensemble a partir du tableau initial sur la base des champs selectionnes (variables quantitatives actives)
# To create a subset from the initial table based on the user-selected fields (quantitative active fields)
dataset<-dataset[names(dataset)[names(dataset) %in% ListVar]]

# Creation dans l'espace de travail d'un fichier html dans lequel on va enregistrer les graphiques et tableaux
# To create in the working directory a html file that will contain all the output plots and tables
directory<-getwd()
HTMLoutput=file.path(directory,"PCA_Results.html")

# Lancement de l'ACP
# To perform the PCA
```

```
res.pca <-PCA(dataset, quanti.sup=NULL, quali.sup=NULL, ind.sup=NULL,
scale.unit=scale, graph=TRUE, ncp=nf)

# Creation d'un objet avec les differents resultats issus de l'ACP relatifs aux variables
# To create an object containing the results from the PCA for variables
var <- get_pca_var(res.pca)
var

# Creation d'un tableau avec les valeurs propres et export xlsx
# To create a table with the Eigen values and to export it to xlsx
eig.val <- get_eigenvalue(res.pca)
write.xlsx(eig.val, "EigenValue.xlsx", asTable=FALSE, col.names = TRUE, row.names =
TRUE, append = FALSE, showNA = FALSE)

# Affichage de la table des valeurs propres dans QGIS
# To display the Eigen values table in QGIS
Eigen_Value_Table=eig.val

# Creation d'un fichier png et enregistrement du graphe des gains d'inertie dedans
# To create a png file and to save the scree plot (gain of inertia) into it
graph1="GrapheGainInertie.png"
png("GrapheGainInertie.png", bg = "transparent", width = 1200, height = 1000, units = "px",
pointsize = 12)
fviz_eig(res.pca, addlabels = TRUE, ylim = c(0, 50), font.main = 20, font.submain = 18,
font.x = 14, font.y = 14)
dev.off()

# Creation d'un tableau avec les coordonnees des variables sur les axes et export xlsx
# To create a table with the coordinates of the variables on the axes and to export it to xlsx
VarCoord<-as.data.frame(var$coord)
# Tri du tableau sur le premier axe
# To sort the table on the first axe
VarCoord<-VarCoord[order(VarCoord$Dim.1, decreasing = TRUE), ]
write.xlsx(VarCoord, "VarCoord.xlsx", asTable=FALSE, col.names = TRUE, row.names =
TRUE, append = FALSE, showNA = FALSE)

# Affichage du tableau avec les coordonnees des variables sur les axes dans QGIS
# To display the coordinates of the variables on the axes in a table in QGIS
Variable_Coordinates_Table=VarCoord

# Creation d'un fichier png et enregistrement du graphe des variables dedans
# To create a png file that will contain the PCA plot of the variables
graph2="GrapheVariables.png"
png("GrapheVariables.png", bg = "transparent", width = 500, height = 500, units = "px",
pointsize = 24)
fviz_pca_var(res.pca, col.var = "black", repel = TRUE)
dev.off()

# Creation d'un fichier png et enregistrement dedans d'une matrice restituant la qualite de
representation des variables (cos2 des variables sur toutes les dimensions)
```

```
# To create a png file that will contain the quality of the representation of the variables into a matrix (cosine of the variables on every dimensions)
graph3="QualiteRepresentation.png"
png("QualiteRepresentation.png", bg = "transparent", width = 1000, height = 1500, units = "px", pointsize = 24)
corrplot(var$cos2, is.corr=FALSE, caption="Qualite Representation")
dev.off()

# Creation d'un objet avec les differents resultats issus de l'ACP relatifs aux individus
# To create an object that will contain all the PCA results for individuals
ind <- get_pca_ind(res.pca)
ind

# Creation d'un fichier png et enregistrement du graphe des individus dedans
# To create a png file that will contain the PCA plot of the individuals
graph4="GrapheIndividus.png"
png("GrapheIndividus.png", bg = "transparent", width = 800, height = 800, units = "px", pointsize = 24)
fviz_pca_ind(res.pca)
dev.off()

# Relancer l'ACP en ne gardant que les n premiers facteurs
# To perfom again the PCA only by keeping the n first factors
res.pca2 <-PCA(dataset, quanti.sup=NULL, quali.sup=NULL, ind.sup=NULL, scale.unit=scale, graph=TRUE, ncp=HAC_nf)

# Calcul d'une CAH sur les n premiers facteurs choisis par l'utilisateur
# To perform a HAC on the n first factors chosen by the user
res.HCPC<-HCPC(res.pca2, nb.clust=Nb_clust, consol=FALSE, graph=FALSE, metric=distance, method=methode)

# Creation d'un fichier png et enregistrement de l'arbre hierarchique dedans
# To create a png file that will contain the hierarchical tree
graph5="Dendrogram.png"
png("Dendrogram.png", bg = "transparent", width = 800, height = 800, units = "px", pointsize = 24)
fviz_dend(res.HCPC,
cex = 0.7,                     # Label size
palette = "jco",               # Color palette see ?ggpubr::ggpar
rect = TRUE, rect_fill = TRUE, # Add rectangle around groups
rect_border = "jco",           # Rectangle color
labels_track_height = 0.8      # Place for labels
)
dev.off()

# Creation d'un fichier png et enregistrement de l'arbre hierarchique 3D dedans
# To create a png file that will contain the 3D hierarchical tree
graph6="Dendrogram3D.png"
png("Dendrogram3D.png", bg = "transparent", width = 800, height = 800, units = "px", pointsize = 24)
```

```
plot.HCPC(res.HCPC, choice='3D.map', ind.names=FALSE, centers.plot=FALSE,
angle=60, axes=c(1, 2))
dev.off()

# Conversion de l'objet res.HCPC data.clust en dataframe
# To convert the object res.HCPC data.clust into a dataframe
res.HCPC.dataclust <- as.data.frame(res.HCPC[["data.clust"]])

# Creation d'une colonne contenant le code des unites spatiales a partir de l'argument
row.names
# Create a column containing the ID of the spatial units using the row.names argument
# Utile uniquement au final pour controler le resultat dans la couche en sortie (verification de
l'appariement)
# Only useful in the end to control the result in the output layer (to check if the matching
between rows is ok)
res.HCPC.dataclust$ID <- row.names(res.HCPC.dataclust)

# Choix des variables dans la couche en sortie pour eviter qu'elles ne soient dupliquees. Ce
qui nous interesse dans le dataframe res.HCPC.dataclust c'est le champ "clust", donc on
enleve toutes les variables actives qui sont deja contenues dans Layer
# To choose the variables in the output layer to avoid their duplication. What concern us in
the dataframe res.HCPC.dataclust is the "clust" field, so we remove all the active variables
which are already contained in Layer
res.HCPC.dataclust<-
res.HCPC.dataclust[names(res.HCPC.dataclust)[!names(res.HCPC.dataclust) %in% ListVar]]

# Reformatage des dataframes en SpatialPolygonsDataFrames pour pouvoir les afficher dans
QGIS et specification de l'argument match.ID = F pour que l'appariement se fasse
automatiquement
# To reformat the dataframes into SpatialPolygonsDataFrames so as to be able to display
them in Qgis. We also specify the argument match.ID = F so that the matching will
automatically be done
dataset_export <- as.data.frame(Layer)
s_poly1 <- SpatialPolygonsDataFrame(Layer, dataset_export, match.ID = F)
s_poly2 <- SpatialPolygonsDataFrame(Layer, res.HCPC.dataclust, match.ID = F)

# Jointure des deux SpatialPolygonsDataFrames pour avoir tous les champs de la couche
Layer + la colonne clust
# To merge the two SpatialPolygonsDataFrames so as to have all the fields of the initial Layer
+ clust field
s_poly_Final <-cbind(s_poly1, s_poly2)

# Affichage de la couche contenant les clusters dans QGIS
# To display the layer containing the clusters in QGIS
Layer_with_Clusters=s_poly_Final

# Choix de la langue (francais ou anglais) pour les titres des tableaux et graphiques dans le
fichier html en sortie
# To select the language (French or English) for the tables and bar plots captions in the output
html file
```

```
langues<-c("Francais","English")
langues
Language
langue<-langues[Language+1]
langue

if (langue == 'Francais'){
# Ajout d'un titre pour le tableau des valeurs propres a inserer dans le fichier html
PCA_Results (reinitialisation du fichier html a chaque fois que le script est execute avec
l'option append = FALSE)
HTML("<br>Tableau des valeurs propres ===>", file= "PCA_Results.html",
append=FALSE)
# Insertion du tableau des valeurs propres dans le fichier html PCA_Results
HTML(eig.val, file= "PCA_Results.html", append=TRUE)

# Ajout d'un titre pour le graphe des gains d'inertie a inserer dans le fichier html PCA_Results
HTML("<br>Graphe des gains d'inertie ===>", file= "PCA_Results.html", append=TRUE)
# Insertion du graphe des gains d'inertie dans le fichier html PCA_Results
HTMLInsertGraph(graph1,file=HTMLoutput, append = TRUE)

# Ajout d'un titre pour le graphe des variables a inserer dans le fichier html PCA_Results
HTML("<br>Graphe des variables (premier plan factoriel, axes 1 et 2) ===>", file=
"PCA_Results.html", append=TRUE)
# Insertion du graphe des variables dans le fichier html PCA_Results
HTMLInsertGraph(graph2,file=HTMLoutput, append = TRUE)

# Ajout d'un titre pour la matrice restituant la qualite de representation a inserer dans le fichier
html PCA_Results
HTML("<br>Qualite de representation des variables (Cos2) ===>", file=
"PCA_Results.html", append=TRUE)
# Insertion de la matrice de la qualite de representation dans le fichier html PCA_Results
HTMLInsertGraph(graph3,file=HTMLoutput, append = TRUE)

# Ajout d'un titre pour le graphe des individus a inserer dans le fichier html PCA_Results
HTML("<br>Graphe des individus (premier plan factoriel, axes 1 et 2) ===>", file=
"PCA_Results.html", append=TRUE)
# Insertion du graphe des individus dans le fichier html PCA_Results
HTMLInsertGraph(graph4,file=HTMLoutput, append = TRUE)

# Ajout d'un titre pour l'arbre hierarchique a inserer dans le fichier html PCA_Results
HTML("<br>Arbre hierarchique ===>", file= "PCA_Results.html", append=TRUE)
# Insertion de l'arbre hierarchique dans le fichier html PCA_Results
HTMLInsertGraph(graph5,file=HTMLoutput, append = TRUE)

# Ajout d'un titre pour pour l'arbre hierarchique sur le plan factoriel a inserer dans le fichier
html PCA_Results
HTML("<br>Arbre hierarchique sur le premier plan factoriel ===>", file=
"PCA_Results.html", append=TRUE)
# Insertion de l'arbre hierarchique sur le plan factoriel dans le fichier html PCA_Results
HTMLInsertGraph(graph6,file=HTMLoutput, append = TRUE)
```

```
# Ajout d'un titre pour les graphiques de description des classes par les variables a inserer dans le fichier html PCA_Results (reinitialisation du fichier html a chaque fois que le script est execute avec l'option append = FALSE)
HTML("<br>Graphiques de description des classes par les variables ( v-test > |1.96| ) ===>", file= "PCA_Results.html", append=TRUE)

# Recuperation dans des dataframes de la description de chacune des N classes par les variables
# To save into dataframes the description of each one of the N clusters by the variables
for (i in 1:Nb_clust){

# Nommage des dataframes
# To name the dataframes
name<- paste("classe",i, sep="") # La fonction paste permet de concatener differents elements

# Recuperation dans des dataframes de la description de chacune des N classes par les variables
# To save into dataframes the description of each one of the N clusters by the variables
assign(name,as.data.frame(res.HCPC$desc.var$quanti[[i]])) # La fonction assign permet d'assigner un nom a une valeur/un element

# Nommage des fichiers png
# To name the png files
name_png<- paste("Graphe_classe",i,".png",sep="")

# Creation des fichiers png qui contiendront les histogramme horizontaux decrivant les classes par les variables
# To create png files that will contain the bar plots describing the clusters by the variables
png(name_png, bg = "transparent", width = 800, height = 800, units = "px", pointsize = 24)
fichier<-get(name)
barplot((fichier$v.test), names = row.names(fichier), col = "black", border = "white", horiz = TRUE, las = 1, xlim = c(-10, 10), cex.names=0.55, main = paste("Classe", i, sep=" "))
dev.off()

# Insertion des graphiques png dans le fichier html PCA_Results
# To insert the bar plots into the PCA_Results html file
HTMLInsertGraph(name_png,file=HTMLoutput, append = TRUE)

}

# Ajout d'un titre pour les tableaux de description des classes par les variables a inserer dans le fichier html PCA_Results
HTML("<br>Tableaux donnant la description des classes par les variables ===>", file= "PCA_Results.html", append=TRUE)
# Insertion des tableaux de description des classes par les variables dans le fichier html PCA_Results
HTML(head(res.HCPC$desc.var$quanti), file= "PCA_Results.html", append=TRUE)

# Affichage du fichier html PCA_Results dans un navigateur
```

```
browseURL("PCA_Results.html")

}else # to add captions for plots and tables in English
{# To add into the PCA_Results html file a caption for the table of the eigen values (the
append = FALSE option allows to reset the html file each time the script is executed)
HTML("<br>Table of the Eigen values ===>", file= "PCA_Results.html", append=FALSE)
# To insert the table of the Eigen values into the PCA_Results html file
HTML(eig.val, file= "PCA_Results.html", append=TRUE)

# To add into the PCA_Results html file a caption for the scree plot (gain of inertia)
HTML("<br>Scree plot (Gain of inertia) ===>", file= "PCA_Results.html", append=TRUE)
# To insert the scree plot (gain of inertia) into the PCA_Results html file
HTMLInsertGraph(graph1,file=HTMLoutput, append = TRUE)

# To add into the PCA_Results html file a caption for the factorial map of variables
HTML("<br>First factorial map showing the variables (axes 1 and 2) ===>", file=
"PCA_Results.html", append=TRUE)
# To insert the factorial map showing the variables into the PCA_Results html file
HTMLInsertGraph(graph2,file=HTMLoutput, append = TRUE)

# To add into the PCA_Results html file a caption for the matrix showing the quality of the
representation of the variables
HTML("<br>Quality of the representation of the variables (Cos2) ===>", file=
"PCA_Results.html", append=TRUE)
# To insert the quality representation matrix into the PCA_Results html file
HTMLInsertGraph(graph3,file=HTMLoutput, append = TRUE)

# To add into the PCA_Results html file a caption for the factorial map showing the
coordinates of the individuals
HTML("<br>First factorial map showing the coordinates of the individuals (axes 1 and 2)
===>", file= "PCA_Results.html", append=TRUE)
# To insert the factorial map showing the coordinates of the individuals into the PCA_Results
html file
HTMLInsertGraph(graph4,file=HTMLoutput, append = TRUE)

# To add into the PCA_Results html file a caption for the Hierarchical cluster tree
HTML("<br>Hierarchical cluster tree ===>", file= "PCA_Results.html", append=TRUE)
HTMLInsertGraph(graph5,file=HTMLoutput, append = TRUE)

# To add into the PCA_Results html file a caption for the 3D Hierarchical cluster tree on the
factor map
HTML("<br>Hierarchical cluster tree on the first factor map ===>", file=
"PCA_Results.html", append=TRUE)
HTMLInsertGraph(graph6,file=HTMLoutput, append = TRUE)

# To add into the PCA_Results html file a caption for the bar plots showing the variables
which best describe the clusters (the append = FALSE option allows to reset the html file each
time the script is executed)
HTML("<br>Bar plots showing the variables which best describe the clusters ( v-test > |1.96|
) ===>", file= "PCA_Results.html", append=TRUE)
```

```
# To save into dataframes the description of each one of the N clusters by the variables
for (i in 1:Nb_clust){

# To name the dataframes
name<- paste("classe",i, sep="")

# To save into dataframes the description of each one of the N clusters by the variables
assign(name,as.data.frame(res.HCPC$desc.var$quanti[[i]])) # La fonction assign permet
d'assigner un nom a une valeur/un element

# To name the png files
name_png<- paste("Cluster_barplot",i,".png",sep="")

# To create png files that will contain the bar plots showing the variables which best describe
the clusters
png(name_png, bg = "transparent", width = 800, height = 800, units = "px", pointsize = 24)
fichier<-get(name)
barplot((fichier$v.test), names = row.names(fichier), col = "black", border = "white", horiz =
TRUE, las = 1, xlim = c(-10, 10), cex.names=0.55, main = paste("Classe", i, sep=" "))
dev.off()

# To insert the bar plots into the PCA_Results html file
HTMLInsertGraph(name_png,file=HTMLoutput, append = TRUE)

}

# To add into the PCA_Results html file a caption for the tables which describe the clusters by
the variables
HTML("<br>Tables giving the description of the clusters by the variables ===>", file=
"PCA_Results.html", append=TRUE)
HTML(head(res.HCPC$desc.var$quanti), file= "PCA_Results.html", append=TRUE)

# Display the PCA_Results html file in a web browser
browseURL("PCA_Results.html")}
```