Consider the Iris data set. Write a R code wich produces the partition matrix. Compute the gravity centers of the quantitative variables in the three classes using a matrix formula. With matrix multiplication P*P instead or percent*percent.

data(iris)
X<-iris[,1:4]
library(nnet)
C<-class.ind(irisDOLLARSpecies) Matrice partition
t((t(X) P*P C))/diag(t(C) P*P C)
image(C)

In this graphic we can see that there is 3 groups actually, this we lead us to our result. We need to understand why we do have 3 groups, there is actually 3 types of flowers so it s normal that there is 3 groups. We will use the kmeans method to try to plot our data :

kmeans.res <- iris P>P select(c(-Species,-Sepal.Length,-Sepal.Width)) P>P kmeans(3,nstart = 10)
cluster<-as.factor(kmeans.resDOLLARcluster)
centers <- as.data.frame(kmeans.resDOLLARcenters)

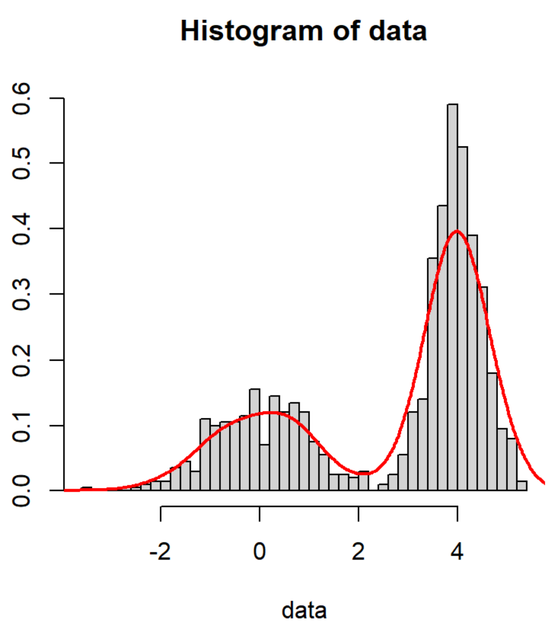
When the number of cluster is increasing the sum of square slowly tend to 0.

n<-1000
Proportions<-c(1,2)/3
parameters<-list()
parameters[[1]]<-list(mean=0,sd=1)
parameters[[2]]<-list(mean=4,sd=1/2)

simu-mixture<-function(n=100,parameters,Proportions)
z<-t(rmultinom(n,1,prob = Proportions))
z<-apply(z,1,which.max)
x<-matrix(0,n,1)
for (i in 1:length(z))
x[i,1]<-rnorm(1,mean=parameters[[z[i]]]DOLLARmean, sd=parameters[[z[i]]]DOLLARsd)
return(list(x=x,z=z))

hist(simu-mixture(n=1000, parameters=parameters, Proportions = Proportions)DOLLARx,breaks=50)

By doing this we create a vector with all of ours parameters and we can now use a histogram to plot our values :



We can see here that we have two major values that are highlight here, 0 and 4 the two means given for this mixture. Now we are using *K-Means* algorithm :

res<-kmeans(simulationDOLLARx,2,nstart = 30)

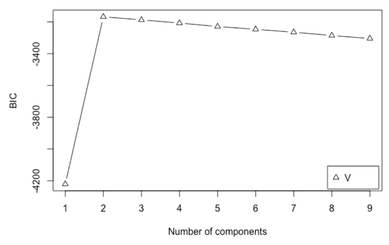
Mclust with model E and V :
So we do apply our Algorithm EV, we apply it with constant variance, here and see what we got :

model-Mclust-E <- Mclust(simulationDOLLARx, G=2, modelNames="E") equal variance
model-Mclust-EDOLLARparameters

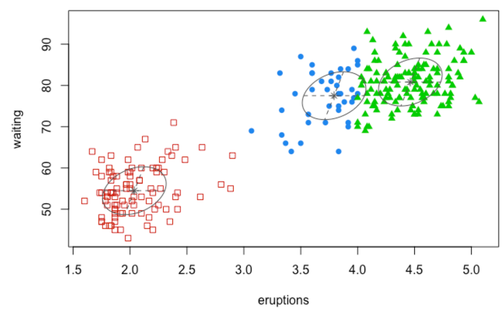
model-Mclust-V <- Mclust(simulationDOLLARx, G=2, modelNames="V") equal variance
model-Mclust-VDOLLARparameters

plot(model-Mclust-VDOLLARBIC)

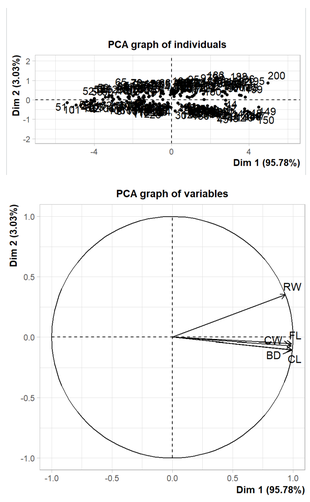
So after applying our algorithm, our goal is to compare each one of the 3 answers we got and try to understand why do we got same or close values. We do plot BIC, and see the difference between the number of composents :



data(faithful) , here we find the number of cluster
summary(faithful), actually Mclust choose it for us
plot(faithfulDOLLARwaiting, faithfulDOLLAReruptions)
model <- Mclust(faithful)
plot(model)

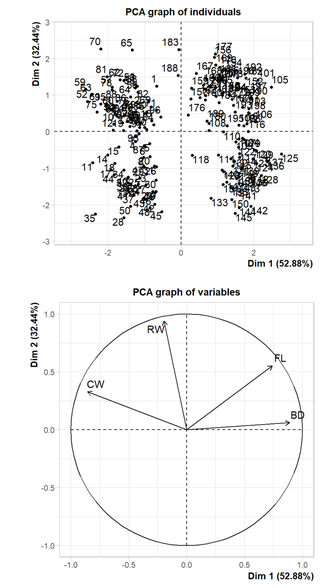


data(crabs)
crabsquant <- crabs[,4:8]
plot(crabsquant)
PCA(crabsquant)
prcomp(crabsquant)



We can see that one of our axis is explaining about 95 percent of our data. This result is obtained because those variable are correlated, to try to spread out our analysis we need to decorrelate it. So we need to find the most correlated variable to divide all of the others by this one in order to obtain a new dataset to work on, this is in sort of a normalization. In order to obtain this column u can plot the matrix of correlation. So we do :

crabsquant-decorel <-
crabsquant/crabsquantDOLLARCL
crabsquant-decorel <- crabsquant-decorel[,-3]
PCA-crabs <- PCA(crabsquant-decorel)



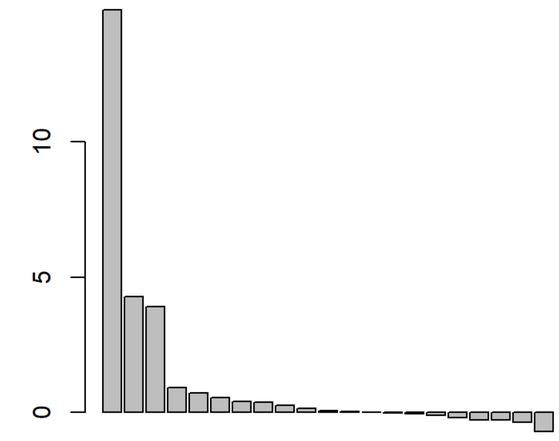
Blue/Male Orange/Male
Blue/Female Blue/Female

Correlation Circle

And now 85 percent of the variance is explained with 2 axis.

D = Dissimilarity Matrix. This matrix contains the distance between x_j and x_i.
This matrix has a diagonal null, is defined positive and symetric. Δ is a matrix with the square of every A coefficient. Cf behind for J and B matrix

tmp <- eigen(B)
J <- tmpDOLLvectors
A <- diag(tmpDOLLvalues)



In Rstudio, eigen value are ranged by decreasing values and we choose to keep 3 of them because they will mostly explained about 85 percent of the dataset. In order to reduce the dimension and to explain more of the dataset. The eigen vectors with the highest eigen value is the one that does explain the most about the model. Each vector will be orthogonal to the previous in order to create a space with m dimensions.

P = percent
library(dplyr)
spam-quant <- spam P>P select(-"type")
spam-quant-norm <- scale(spam-quant)
res.PCA <- PCA(spam-quant-norm, graph = F)
eigvalues <- data.frame(res.PCADOLLeig)
barplot(eigvaluesDOLLpercentage.of.variance, names.arg = row.names(eigvalues))
X <- res.PCADOLLindDOLLcoord
plot(X[,1], X[,2],
col = as.numeric(spamDOLLtype)+2)

m = 100 N =nrow(spam-quant-norm)
Tirage <- sample(1:N, m, replace = FALSE)
spam-subset = spam-quant-norm[Tirage,]
kpc <- kpca(., data = as.data.frame (spam-subset), kernel = "rbfdot", kpar = list(sigma=0.01))
kpvc <- pcv(kpc)
plot(rotated(kpc)[,1:2], xlab = "1st Principale Comp.", ylab = "2nd Principale Comp.")
barplot(eig(kpc)/sum(eig(kpc)))

Here we have done kernel PCA with a '*RBF*' a linear model, with now need to work with different type of kernel to compare our result. Around 25 percent of variance is explained so we need to apply few more methods. In the second plot with a better distribution of the values in the barplot.