Dimesion Reduction

2022-11-07

Load packages, clean environment, and set working directory

```r
# Libraries for preparing data for analysis
library(ape)
library(dplyr)
library(nlme)
library(tidyverse)
library(vroom)
library(readxl)
library(ggplot2)

# Libraries for PCA (principal components analysis)
library(vegan)
library(factoextra) #fviz_eig

#Libraries for NMF (non-negative matrix factorization)
# if (!requireNamespace("BiocManager", quietly = TRUE))
#   install.packages("BiocManager")
# BiocManager::install("Biobase", version = "3.16")
library(Biobase)
library(NMF)

#libraries for MCA
library(FactoMineR)
library(dplyr)
library(factoextra) #fviz_eig

#Clean environment
rm(list=ls())
graphics.off()

#Set working directory
setwd("~/Library/CloudStorage/OneDrive-WashingtonStateUniversity(email.wsu.edu)/Fernandez Lab/Projects
```

Prepare genomic data for dimension reduction

Let's clean our sequence data and explore the variability in the data!

```r
#Load genome annotations and trim
genes <- read_xlsx("OPVnew_nowwithVirus.xlsx", sheet="PoxHost")

#Rename variables and exclude unnecessary variables
```

```r
genes <- plyr::rename(genes, c("Virus"="VirusSpecies","Host Genus"="HostGenus","Host Species"="HostSpec
genes <- subset(genes, select=-c(HostSpecies))

#Correct sequence MT903347_1 - 'HostGenus' var lists Family name instead of Genus
genes$HostGenus <- ifelse(genes$HostGenus=="Gliridae","Graphiurus",genes$HostGenus)

#Add unique identifier
genes$rownames <- rownames(genes)
genes$Sequence <- paste(genes$Genome,genes$VirusSpecies,genes$HostGenus,sep="_",genes$rownames)
genes$rownames=NULL
genes <- genes %>% dplyr::select(Sequence, everything())

#View frequency of various virus species
prop_table <- subset(genes, select=-c(Sequence,Genome))
prop_table$Frequency = 1
prop_table <- aggregate(Frequency ~ VirusSpecies + HostGenus, data=prop_table, FUN=sum)
prop_table <- prop_table[order(prop_table[,c("VirusSpecies")],prop_table[,c("HostGenus")]) ,]
prop_table$Perc <- prop_table$Frequency/sum(prop_table$Frequency)*100
print(prop_table)
```

```
##                VirusSpecies   HostGenus Frequency       Perc
## 29 Abatino macacapox virus      Macaca         1  0.5076142
## 2            Akhmeta virus    Apodemus         3  1.5228426
## 22           Akhmeta virus        Homo         3  1.5228426
## 23         Alaskapox virus        Homo         1  0.5076142
## 6           Camelpox virus     Camelus         6  3.0456853
## 44       Cetaceanpox virus    Tursiops         1  0.5076142
## 1             Cowpox virus    Acinonyx         6  3.0456853
## 5             Cowpox virus  Callithrix         1  0.5076142
## 8             Cowpox virus      Castor         1  0.5076142
## 12            Cowpox virus     Cynomys         1  0.5076142
## 14            Cowpox virus  Dolichotis         1  0.5076142
## 15            Cowpox virus     Elephas         2  1.0152284
## 16            Cowpox virus       Equus         1  0.5076142
## 18            Cowpox virus       Felis        24 12.1827411
## 21            Cowpox virus Herpailurus         2  1.0152284
## 24            Cowpox virus        Homo        25 12.6903553
## 33            Cowpox virus    Microtus         3  1.5228426
## 35            Cowpox virus      Mungos         1  0.5076142
## 37            Cowpox virus      Myodes         1  0.5076142
## 40            Cowpox virus     Procyon         1  0.5076142
## 42            Cowpox virus      Rattus         9  4.5685279
## 43            Cowpox virus    Saguinus         1  0.5076142
## 45            Cowpox virus     Vicugna         5  2.5380711
## 25        Ectromelia virus        Homo         1  0.5076142
## 36        Ectromelia virus         Mus         2  1.0152284
## 7          Monkeypox virus       Canis         1  0.5076142
## 10         Monkeypox virus  Cricetomys         1  0.5076142
## 11         Monkeypox virus   Crocidura         1  0.5076142
## 13         Monkeypox virus     Cynomys         3  1.5228426
## 19         Monkeypox virus Funisciurus         2  1.0152284
## 20         Monkeypox virus  Graphiurus         2  1.0152284
## 26         Monkeypox virus        Homo        57 28.9340102
## 28         Monkeypox virus   Ictidomys        2  1.0152284
## 30         Monkeypox virus      Macaca        2  1.0152284
## 31         Monkeypox virus   Malacomys        1  0.5076142
## 38         Monkeypox virus         Pan        1  0.5076142
## 41        Raccoonpox virus     Procyon        2  1.0152284
## 32          Skunkpox virus    Mephitis        1  0.5076142
## 3           Vaccinia virus         Bos        2  1.0152284
## 4           Vaccinia virus     Bubalus        4  2.0304569
## 9           Vaccinia virus Chlorocebus        1  0.5076142
## 17          Vaccinia virus       Equus        1  0.5076142
## 27          Vaccinia virus        Homo        8  4.0609137
## 34           Volepox virus    Microtus        1  0.5076142
## 39           Volepox virus  Peromyscus        1  0.5076142
```

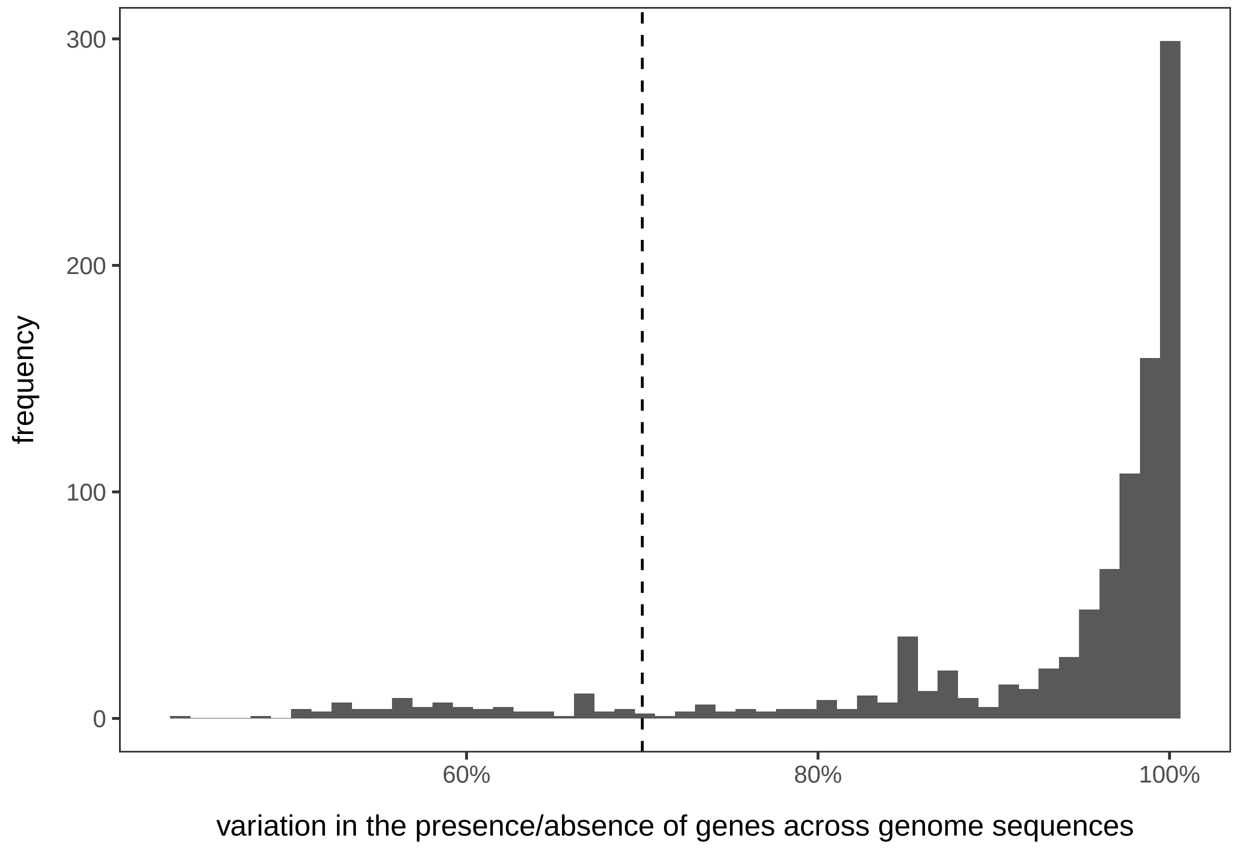
```
#Save frequency table to Output folder
# write.csv(table, "Output/gene_freq_table.csv")

#Create function (mode.prop) to assess variation in the presence/absence of OPV genes
mode.prop <- function(x) {
  ux <- unique(x[is.na(x)==FALSE])        # creates array of unique values
  tab <- tabulate(match(na.omit(x), ux))  # creates array of the frequency a unique value appears in a
  max(tab)/length(x[is.na(x)==FALSE])     # max-frequency / number of elements in each column that are
}

# Assess variation across columns (2 indicates columns)
vars=data.frame(apply(genes,2,function(x) mode.prop(x)),
                apply(genes,2,function(x) length(unique(x)))) # number of unique elements in each colum
vars$variables=rownames(vars)
colnames(vars) <- c("var","uniq","column")

# Trim
vars <- vars[-c(1,2), ]

# Any variables with no variation? If so drop
which(vars$var==1)
```

```
## integer(0)
```

```
# vars <- subset(vars,vars$var<1)

# Visualize distribution of variation
#png("Output/gene_variation.png", width=4,height=4,units="in",res=600)
gene_var <- ggplot(vars,
       aes(var))+
  geom_histogram(bins=50)+
  geom_vline(xintercept=0.70,linetype=2,size=0.5)+
  theme_bw()+
  theme(panel.grid.major=element_blank(),panel.grid.minor=element_blank())+
  theme(axis.title.x=element_text(margin=margin(t=10,r=0,b=0,l=0)))+
  theme(axis.title.y=element_text(margin=margin(t=0,r=10,b=0,l=0)))+
  labs(y="frequency",
       x="variation in the presence/absence of genes across genome sequences")+
  scale_x_continuous(labels=scales::percent)
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
```

```
#dev.off()
gene_var
```



```
# Clean environment
rm(prop_table, vars, gene_var, mode.prop)
```

(1) PCA of viral accessory genes

Using principal components analysis, can we distill the variables down to their most important features? Which genes contribute the most to each feature?

```
# Subset data and reformat as numeric matrix
# genes_mat <- subset(genes,select=-c(Genome,VirusSpecies,HostGenus))
# mat <- as.matrix(genes_mat[,-1])
# rownames(mat) <- genes_mat[,1] %>% pull()
# class(mat) <- "numeric"

#Apply PCA using stats::prcomp
pca1 <- prcomp(genes[,5:985])       #scaling/centering not appropriate
relvar <- pca1$sdev^2 / sum(pca1$sdev^2)
relvar_per <- round(relvar*100,1)
```

```r
#View summary results
# summary(pca1)
# View(pca$x) #sequence (individuals)
# View(pca$rotation) #genes (variables)

#Table of importance of components: Eigenvalue, SD, Proportion of Variance, Cumulative Prop
importance <- as.data.frame(t(summary(pca1)$importance))
importance$Eigenvalue <- importance$`Standard deviation`^2
importance <- importance %>% dplyr::relocate(Eigenvalue)
importance <- importance[c(1:10),]
### Eigenvalue: the variance explained by each PC

#Table of loadings: $rotation is the matrix of variable loadings where columns are eigenvectors
loadings <- as.data.frame(pca1$rotation)
loadings <- loadings[,c(1:10)]
loadings <- abs(loadings) #get absolute values
### Why are some loadings > |1|? Loading is the covariances/correlations b/w original vars and unit-sca

#For each dimension, create df of accessory genes
for(i in 1:ncol(loadings)){
  assign(colnames(loadings)[i], data.frame(loadings[,i]))
}

#Create list of dataframes of PC loadings
list <- colnames(loadings)
list_df = lapply(list, get)

#To each dataframe in that list, add corresponding gene name and sort in descending order (genes with h
for (i in 1:length(list)) {
  colnames(list_df[[i]]) <- "Loadings"
  list_df[[i]]$Gene <- rownames(loadings)
  list_df[[i]]=list_df[[i]][order(-list_df[[i]]$Loadings),]
}

#Drop loadings (only need ranking of genes)
for(i in 1:length(list)) {
  list_df[[i]]$Loadings=NULL
}

#Save PC gene rankings as table
rank_loadings <- data.frame(matrix(ncol=ncol(loadings), nrow=nrow(loadings)))
colnames(rank_loadings) <- colnames(loadings)
for(i in 1:length(list)) {
  rank_loadings[,i] = list_df[[i]]
}
print("Top 20 accessory genes with the largest loadings")
```

```
## [1] "Top 20 accessory genes with the largest loadings"
```

```r
head(rank_loadings,20)
```

```
##             PC1         PC2         PC3         PC4         PC5         PC6
## 1  SNB57677.1 AGR37027.1 AXN75245.1 URK21303.1 URK21279.1 SNB56391.1
## 2  SNB50228.1 SNB48500.1 AGY98600.1 SNB56391.1 CUI02483.1 URK21279.1
## 3  AGZ00427.1 SPN68915.1 AGZ00715.1 ATB56114.1 AGY97404.1 AKJ93648.1
## 4  AZY91520.1 UEC93297.1 AGY99636.1 ADZ30436.1 SNB49818.1 AOP31461.1
## 5  AGZ01043.1 ADZ29950.1 URF91580.1 AAY97376.1 UPV00262.1 AOP31711.1
## 6  AGZ00855.1 AZY91082.1 AGY98413.1 BDQ10418.1 QKE59858.1 AOP31501.1
## 7  SNB48439.1 QNP13069.1 BDQ10542.1 QJQ40180.1 AGY98413.1 AOP31502.1
## 8  ADZ29563.1 ATB55769.1 SNB49818.1 QNP13044.1 ADZ29563.1 AOP31289.1
## 9  SNB63702.1 QNP12693.1 UPV00262.1 ARR30464.1 QNP12533.1 AKJ93661.1
## 10 AGY97210.1 AGR37221.1 SNB48426.1 SNB58018.1 AZY89284.1 AKJ93663.1
## 11 ADZ30410.1 USG71453.1 AGY97404.1 AGR37813.1 URK21282.1 AOP31729.1
## 12 AZY91526.1 AGR37033.1 AGF36621.1 AAY97012.1 AZY89555.1 AOP31760.1
## 13 QCY54139.1 SNB57100.1 SNB50673.1 AGR38581.1 SNB63702.1 AOP31374.1
## 14 SNB48849.1 AGY98361.1 AGY99938.1 AAL40474.1 SNB48426.1 AOP31821.1
## 15 ADZ29558.1 AGZ00866.1 ADZ29950.1 SNB50029.1 SNB50029.1 AOP31412.1
## 16 QEQ49763.1 UPV00452.1 AGR35818.1 QKE59858.1 CRL86950.1 AKJ93790.1
## 17 AZY91347.1 SNB54318.1 AXN75207.1 AZY91520.1 QQA05472.1 AOP31428.1
## 18 SNB50795.1 QNP14477.1 SNB49398.1 SPN68107.1 AXN75245.1 AOP31857.1
## 19 ARR30773.1 ADZ29296.1 QEQ49504.1 AGZ01043.1 AGF36696.1 AOP31647.1
## 20 DAD53541.1 ADZ24189.1 AGY97427.1 QEQ49955.1 AGR38581.1 AOP31860.1
##             PC7        PC8        PC9       PC10
## 1    SNB50029.1 AGR37033.1 ARR30464.1 SNB50747.1
## 2    ADX22669.1 SNB57100.1 QNP12533.1 AGZ00866.1
## 3    AGR38581.1 AZY89555.1 AGZ00866.1 AGF36621.1
## 4    QKE59858.1 URK21282.1 AAY97012.1 SNB50157.1
## 5    QNP12533.1 AZY89284.1 AAL40474.1 USO09134.1
## 6    ARR30464.1 AZY90595.1 SNB50029.1 SPN68107.1
## 7    QJQ40180.1 AZY90734.1 AGR38581.1 QEQ49763.1
## 8    AAL40474.1 AGY99032.1 ATB55273.1 BDQ10517.1
## 9  CAB5514210.1 QNP14477.1 SPN68114.1 BDQ10418.1
## 10   QGQ59741.1 QQA05060.1 QQA05677.1 UEC93542.1
## 11   AAY97012.1 SPN68915.1 SPN68107.1 QNP13044.1
## 12   URG34914.1 SNB50673.1 ADZ24189.1 QKE59817.1
## 13   AGR34497.1 AGY99378.1 AZY89067.1 AAY97407.1
## 14   AGR37744.1 CRL86950.1 QNP13567.1 SOU90190.1
## 15   SNB50747.1 SNB49398.1 QQA05666.1 ADZ24189.1
## 16   AGR37813.1 AYN64771.1 AGR37813.1 QGQ59741.1
## 17   SNB49712.1 QQA05666.1 QNP14477.1 AGY97210.1
## 18   BDQ10517.1 AGY97342.1 AFH54710.1 ATB55769.1
## 19   CUI02483.1 QNP13044.1 ATB56114.1 AZY91526.1
## 20   SPN68107.1 QNP12344.1 QEQ49955.1 AGY98712.1
```

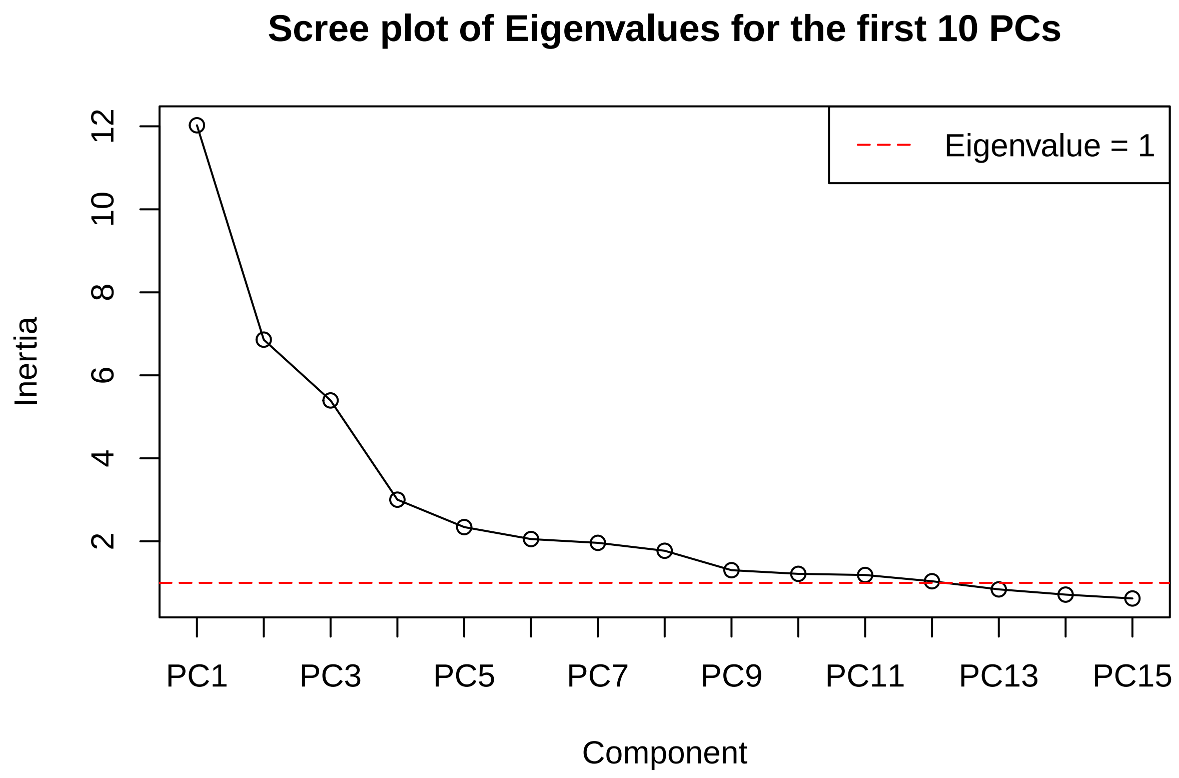
```
rm(list=ls(pattern="^PC"), list_df, loadings, importance, rank_loadings, i, list)
```

(1) PCA visualizations

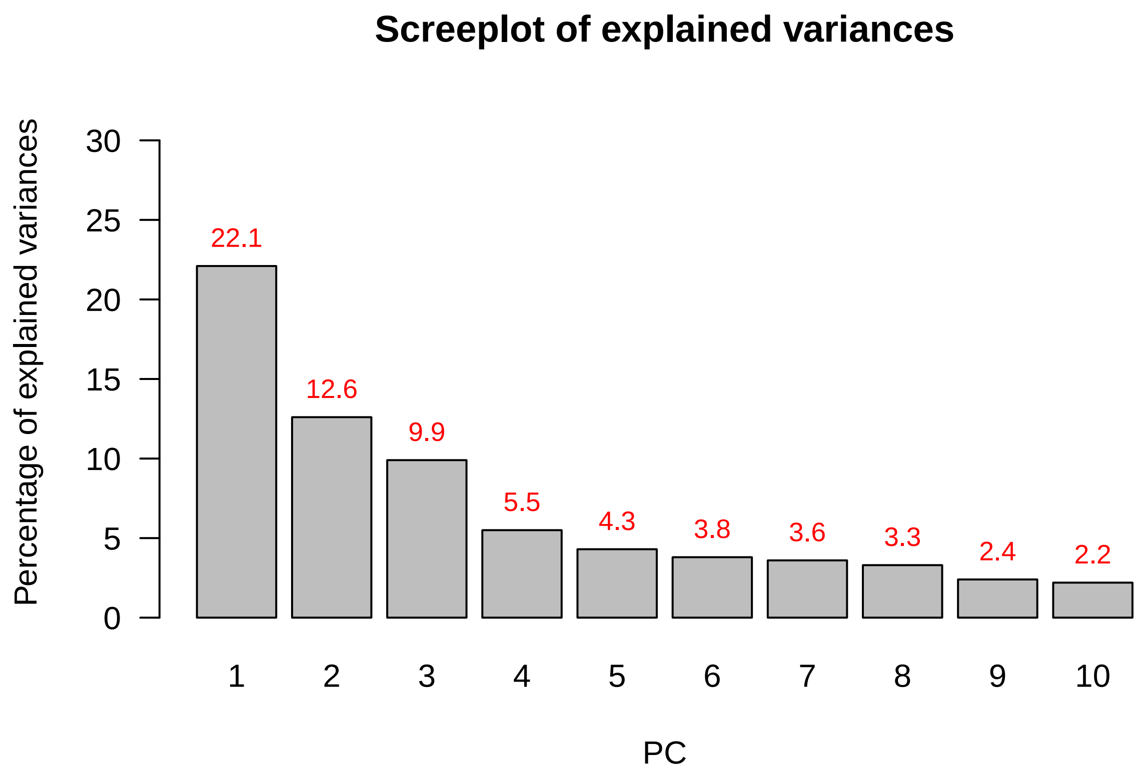
Do all of the dimensions spark joy?

```
#Vizualize variance: screeplots, cumulative variance, etc.
#Vizualize individuals/scores
#Vizualize variables/loadings: by virus family, etc.
#Vizualize centroid
#Visualize scores by cluster via hierarchical cluster analysis (k-means)
```

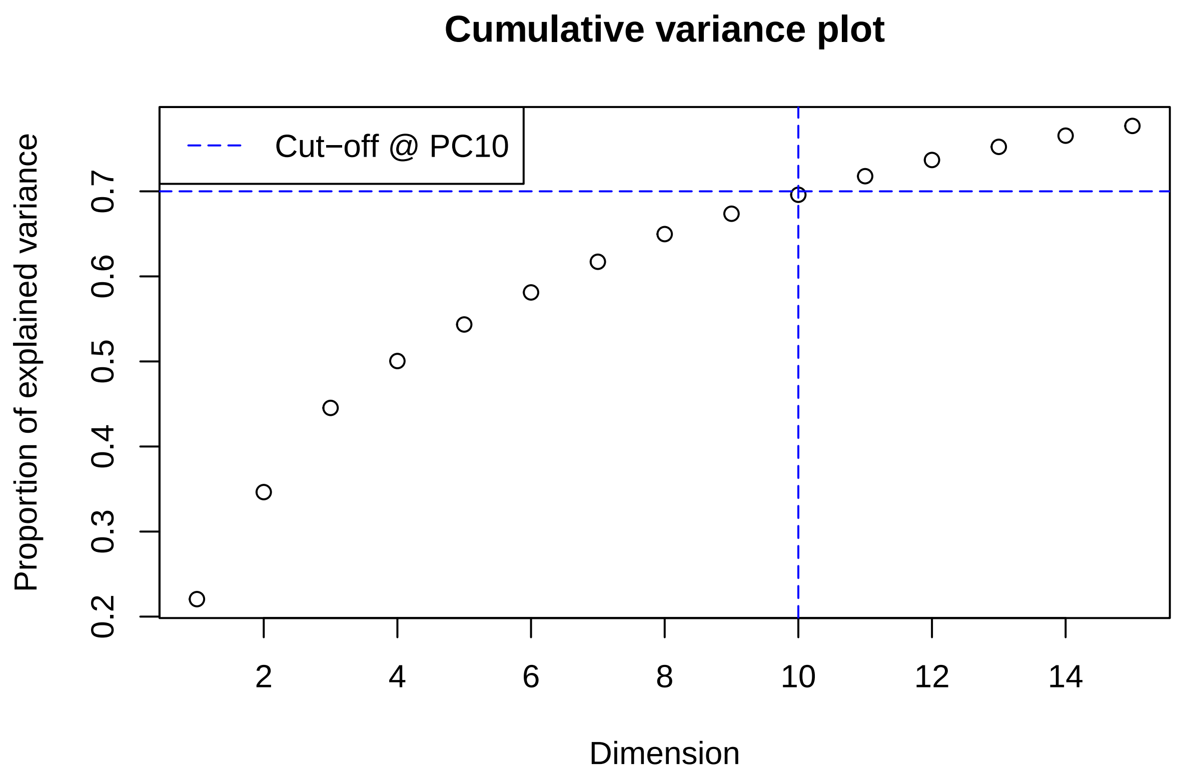
```
#Screeplot variance (eigenvalues) to show the decreasing rate at which variance is explained by additio
screeplot(pca1, type="lines", npcs=15, main="Scree plot of Eigenvalues for the first 10 PCs")
abline(h=1, col="red", lty=5)
legend("topright", legend=c("Eigenvalue = 1"), col=c("red"), lty=5, cex=1)
```



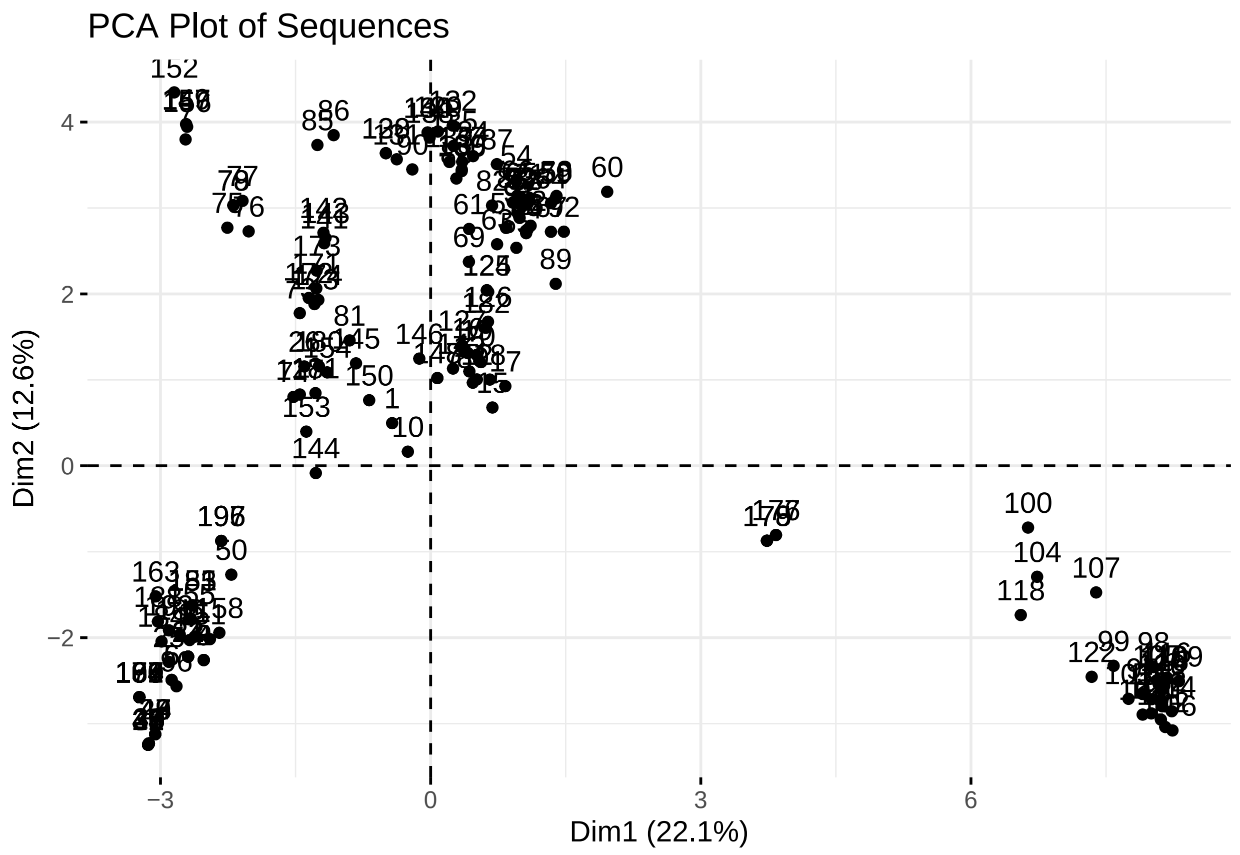
```
### suggests cutoff at PC10

#Screeplot cumulative variance to show the % variance explained by additional PCs
screeplot <- barplot(relvar_per[1:10], xlab='PC', ylab='Percentage of explained variances', main='Scree
text(screeplot, 0, y=relvar_per[1:10], label=relvar_per[1:10],cex=0.8, pos=3, col="red")
```



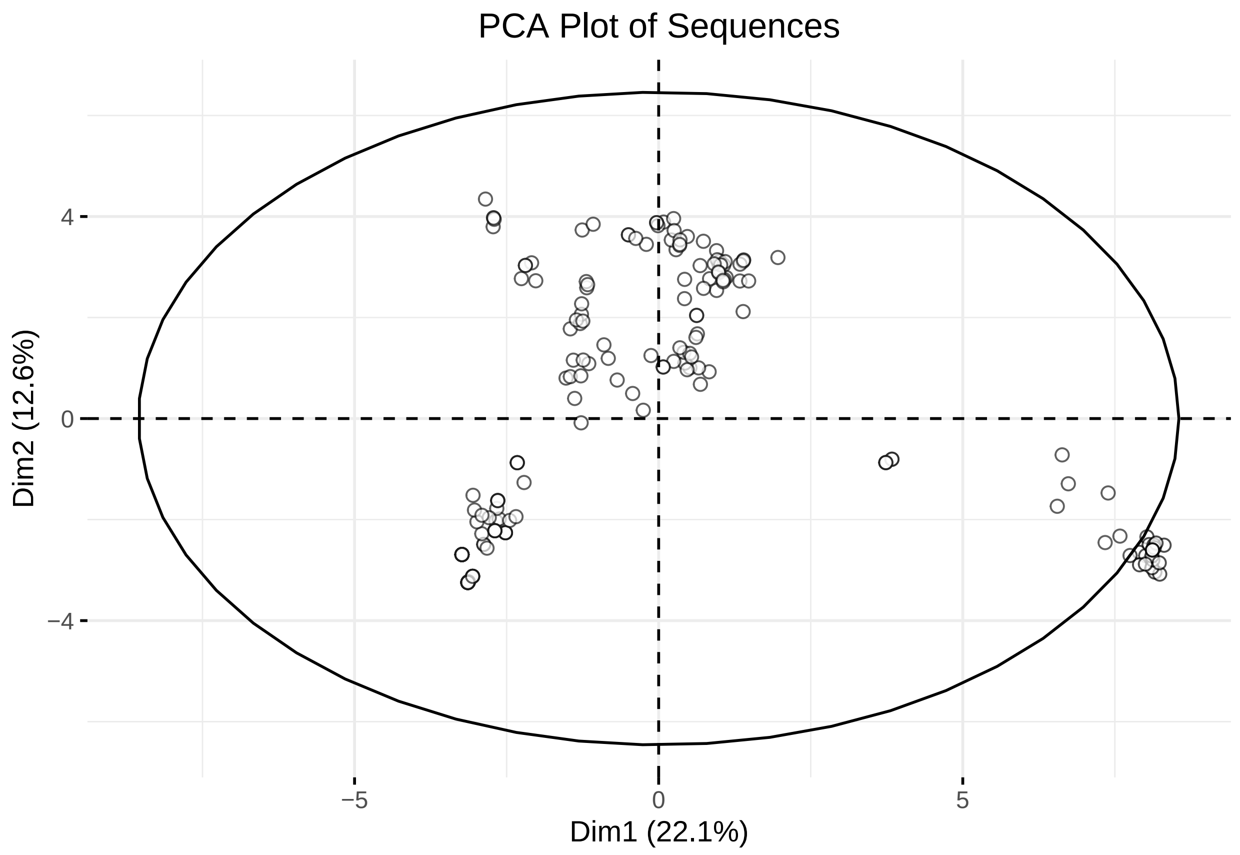
```
# fviz_eig(pca, choice=c("variance"), main = "Scree plot of explained variances") # these values agree

#Plot cumulative variance to show the proportion of variance explained with each add'l PC
cumpro <- cumsum(pca1$sdev^2 / sum(pca1$sdev^2))
plot(cumpro[0:15], xlab = "Dimension", ylab = "Proportion of explained variance", main = "Cumulative va
abline(v = 10, col="blue", lty=5)
abline(h = 0.7, col="blue", lty=5)
legend("topleft", legend=c("Cut-off @ PC10"), col=c("blue"), lty=5, cex=1)
```



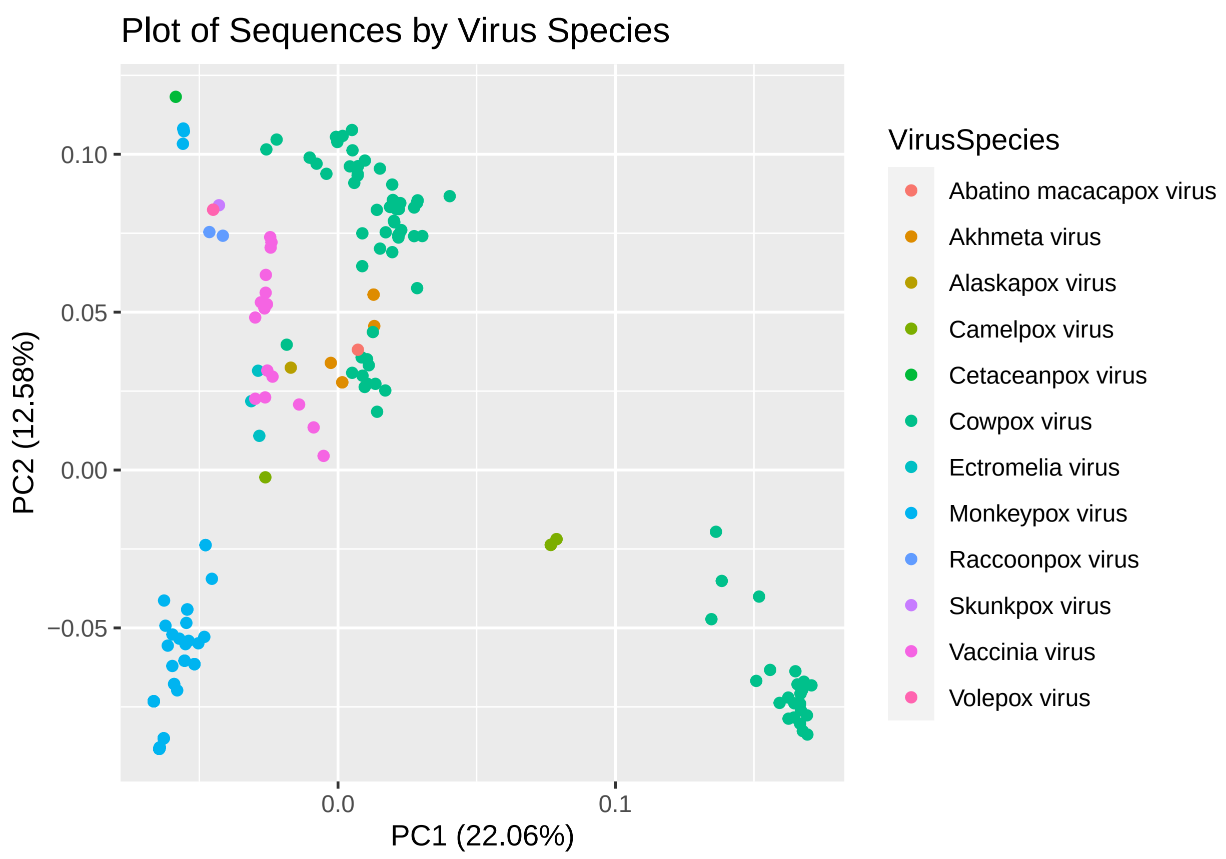
```
#Plot sequences
fviz_pca_ind(pca1) + ggtitle("PCA Plot of Sequences")
```



```
### with ellipses
fviz_pca_ind(pca1, geom.ind = "point", pointshape = 21, pointsize = 2,
             col.ind = "black", addEllipses = TRUE, label = "var",
             col.var = "black", palette = "rickandmorty", repel = TRUE,
             alpha.ind = 0.7) +
             ggtitle("PCA Plot of Sequences") + theme(plot.title = element_text(hjust = 0.5))
```

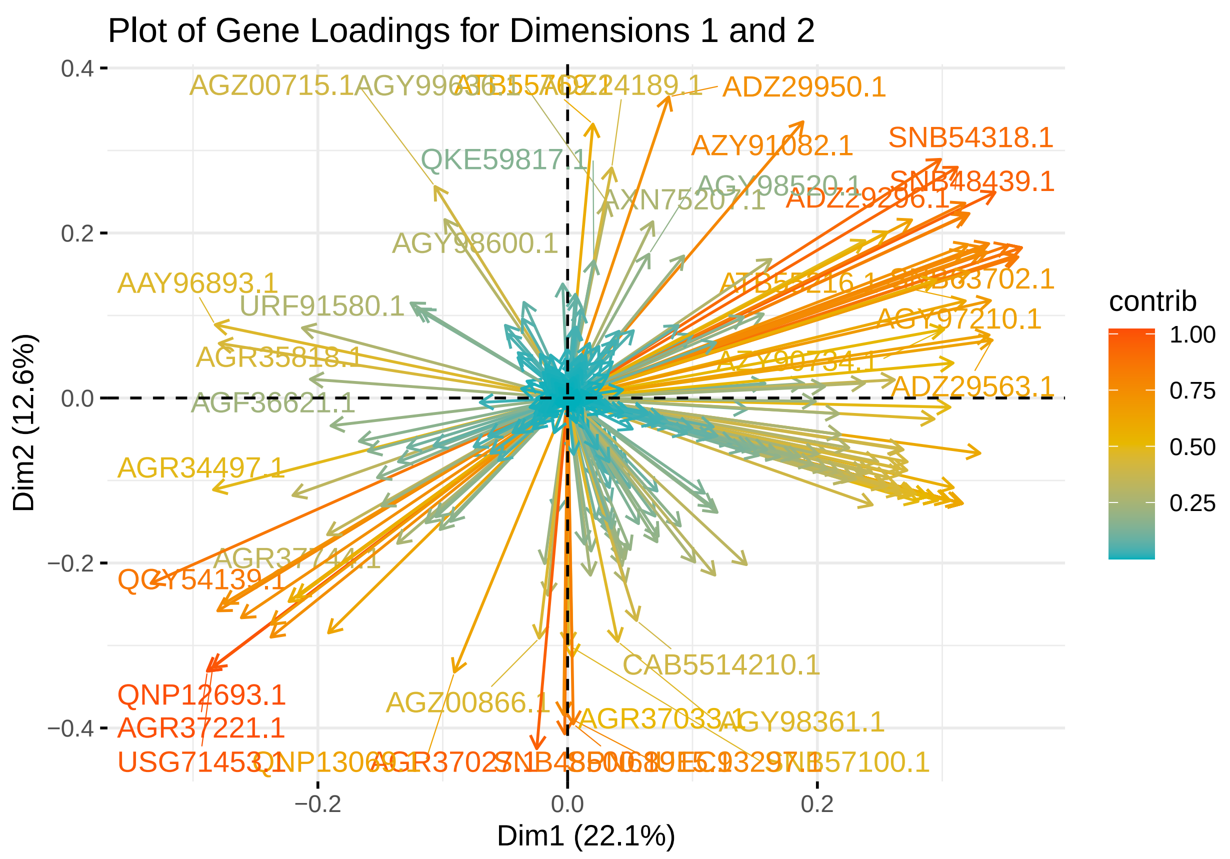


```
#Plot sequences by virus species for dim 1 and 2
library(ggfortify)
autoplot(pca1, data = genes, colour = 'VirusSpecies') + ggtitle("Plot of Sequences by Virus Species")
```



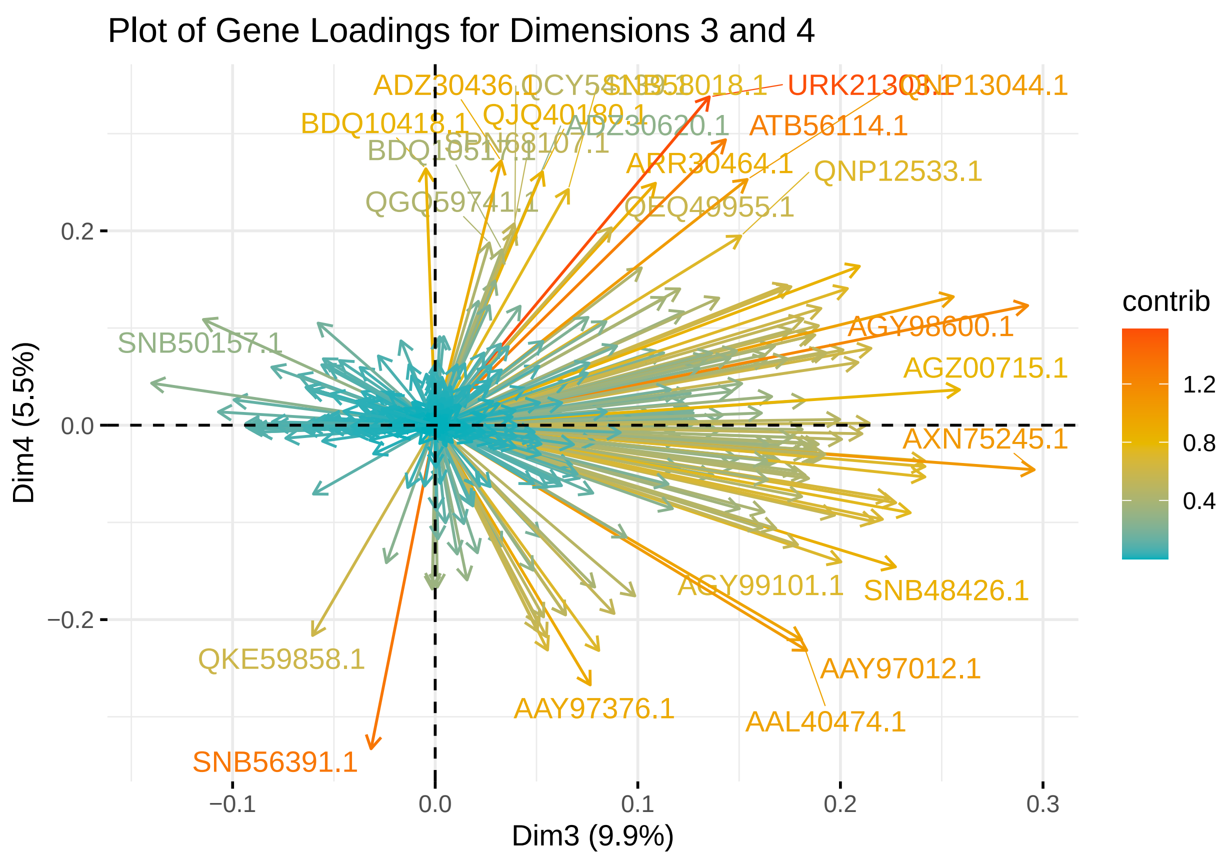
```
#Plot gene loadings for dim 1 and 2
fviz_pca_var(pca1,
             col.var = "contrib", # Color by contributions to the PC
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE,     # Avoid text overlapping
             label = c("ind", "ind.sup", "quali", "var", "quanti.sup")) +
             ggtitle("Plot of Gene Loadings for Dimensions 1 and 2")
```

```
## Warning: ggrepel: 943 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



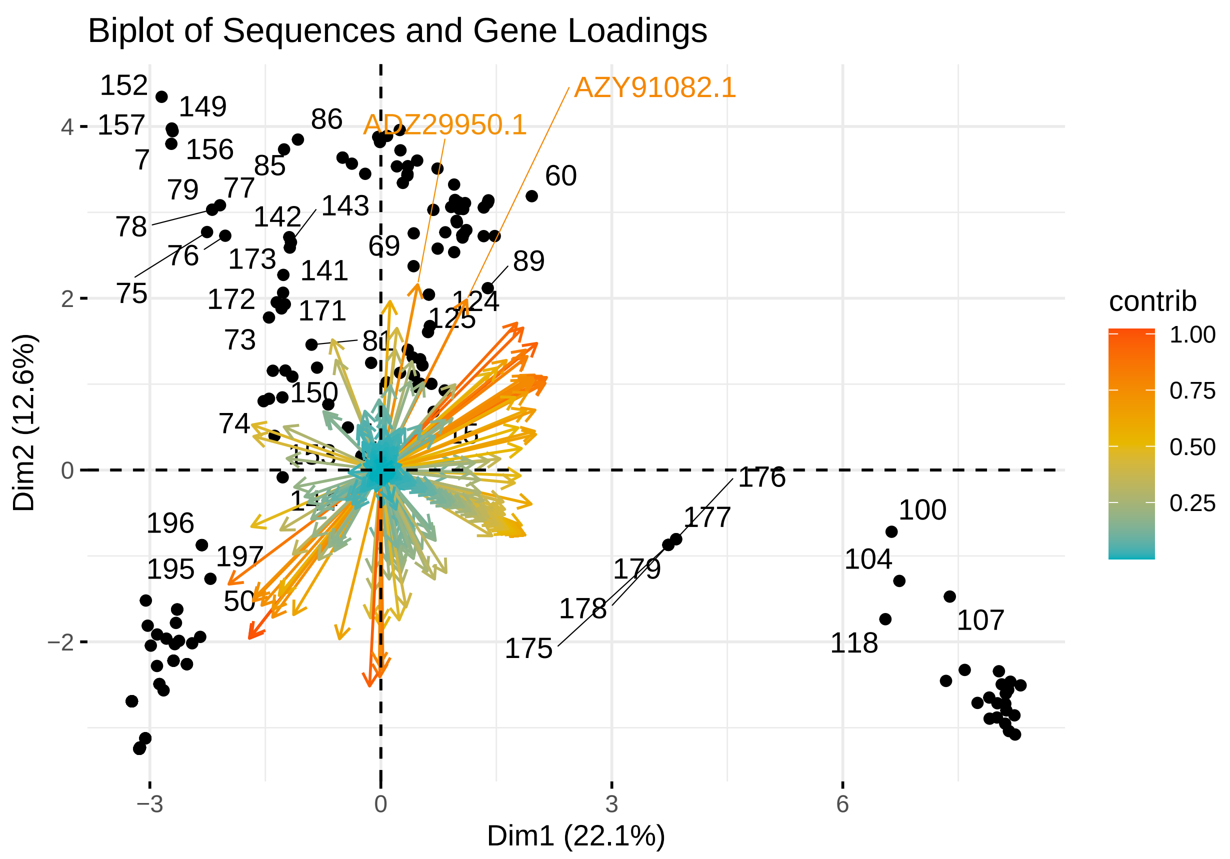
```
### Here we see PC1 has large positive associations with a number of AGs like ADZ29556.1, SNB51281.1, a
### QKE61192.1 - hypothetical protein [Vaccinia virus]
### QNP13375.1 - MPXV Viral membrane assembly proteins (VMAP) (Cop-A 30.5L)"

#Plot gene loadings for dim 3 and 4
fviz_pca_var(pca1, axes = c(3, 4),
             col.var = "contrib",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE) +
             ggtitle("Plot of Gene Loadings for Dimensions 3 and 4")
```

```
## Warning: ggrepel: 955 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



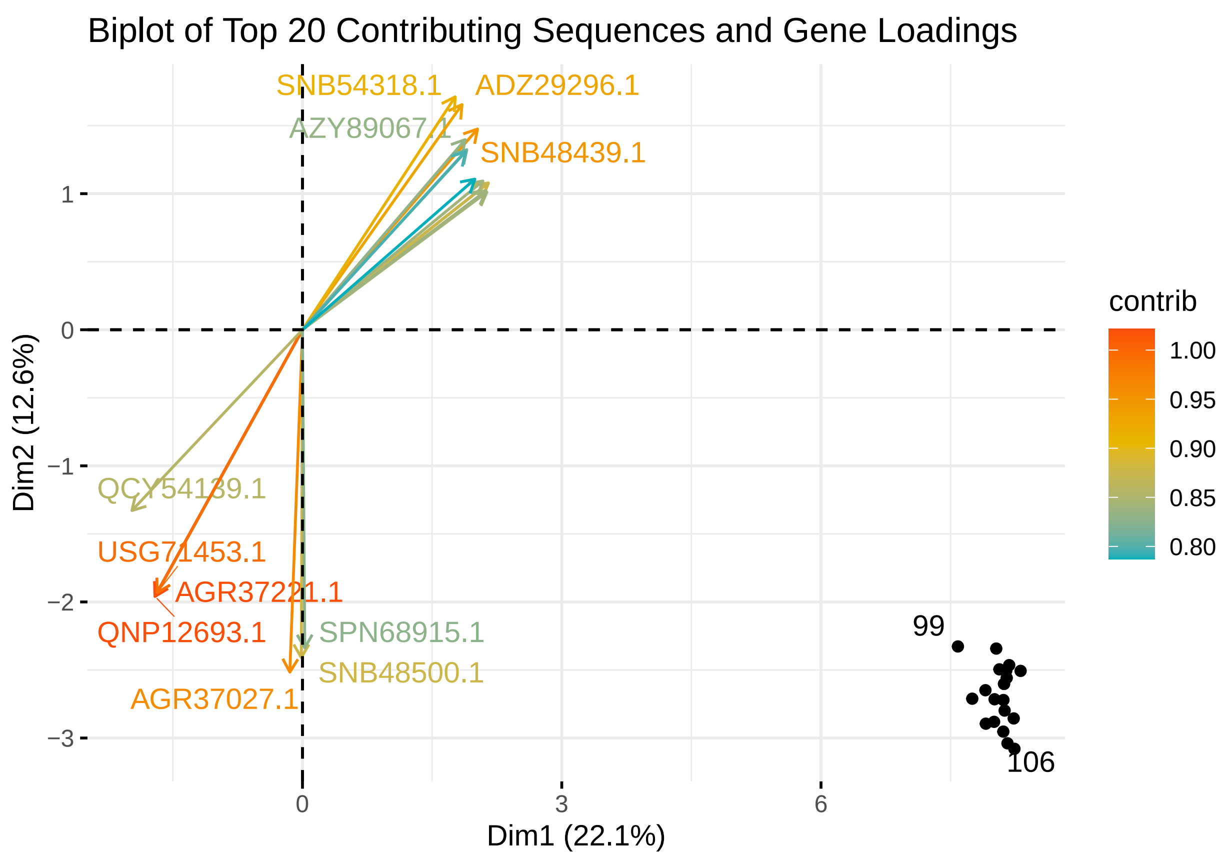
```
#Biplot sequences and gene loadings
fviz_pca_biplot(pca1, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
                repel = TRUE) +
                ggtitle("Biplot of Sequences and Gene Loadings")
```

```
## Warning: ggrepel: 152 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps

## Warning: ggrepel: 979 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
#Biplot top 20 influential scores and loadings
fviz_pca_biplot(pca1, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE, select.ind=list(contrib=20), select.var=list(contrib=20), max.overlaps=Inf)
             ggtitle("Biplot of Top 20 Contributing Sequences and Gene Loadings")
```

```
## Warning: ggrepel: 18 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps

## Warning: ggrepel: 9 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

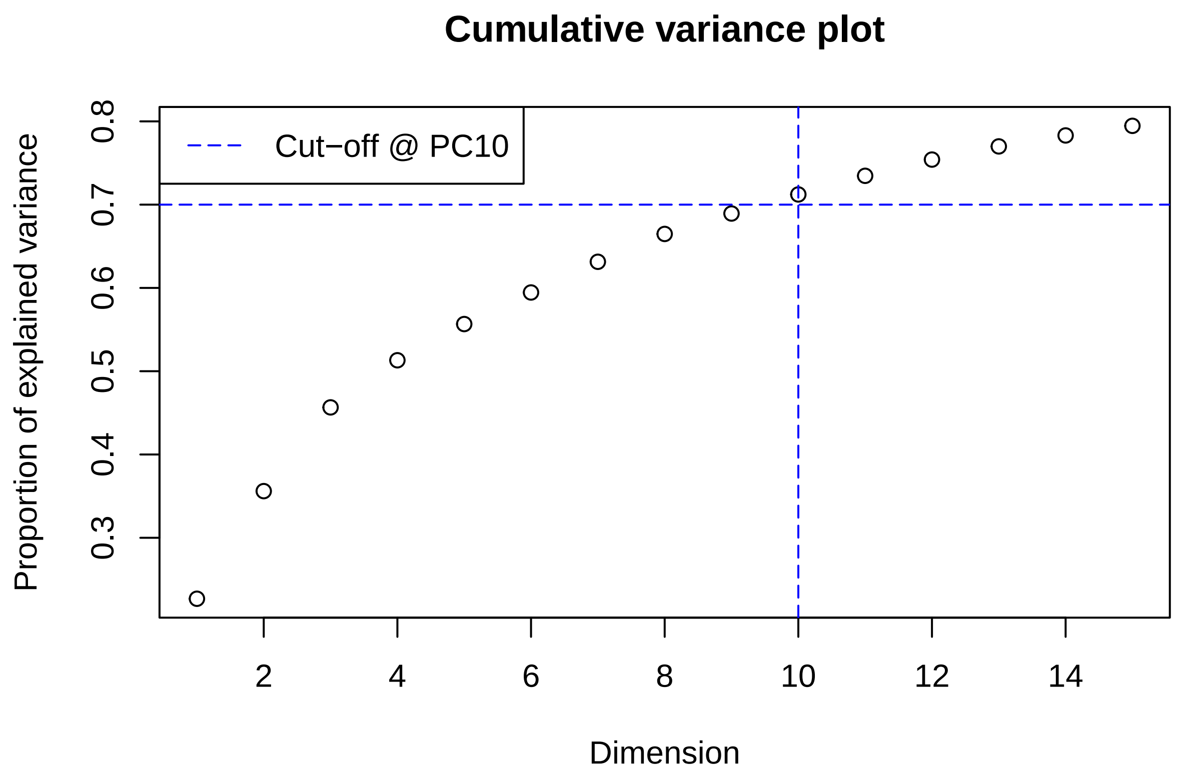


```
rm(cumpro)
```

(2) PCA Alternative Analysis

What happens when we exclude accessory genes present in only one virus species?

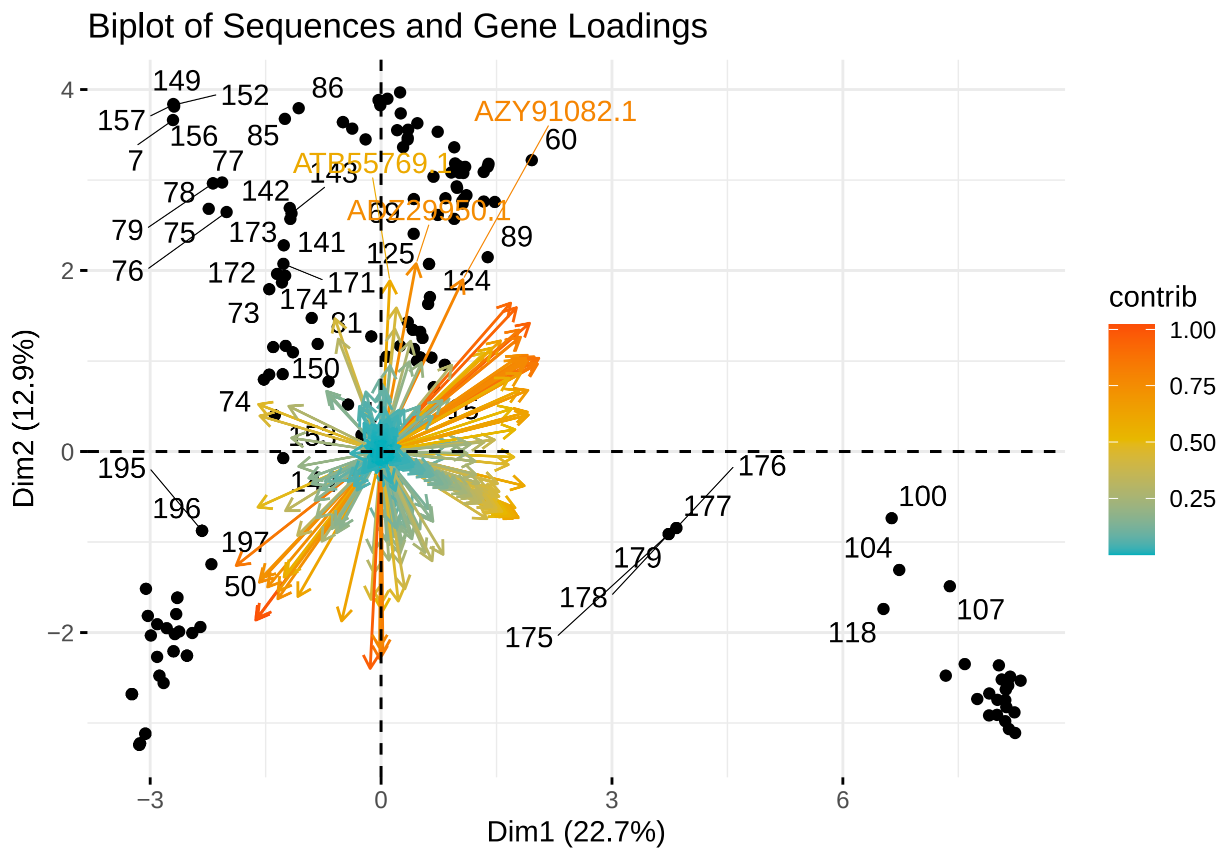
```
#Drop accessory genes that are present in only one virus species (all 0's except for one)
genes2 <- genes[c(1:4,4 + which(colSums(genes[-(1:4)])>1))]
### 985 variables to 686 variables

#Apply PCA using stats::prcomp
pca2 <- prcomp(genes2[,5:686])

#Plot cumulative variance to show the proportion of variance explained with each add'l PC
cumpro <- cumsum(pca2$sdev^2 / sum(pca2$sdev^2))
plot(cumpro[0:15], xlab = "Dimension", ylab = "Proportion of explained variance", main = "Cumulative va
abline(v = 10, col="blue", lty=5)
abline(h = 0.7, col="blue", lty=5)
legend("topleft", legend=c("Cut-off @ PC10"), col=c("blue"), lty=5, cex=1)
```



```r
#Biplot sequences and gene loadings
fviz_pca_biplot(pca2, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
               repel = TRUE) +
               ggtitle("Biplot of Sequences and Gene Loadings")
```

```
## Warning: ggrepel: 151 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

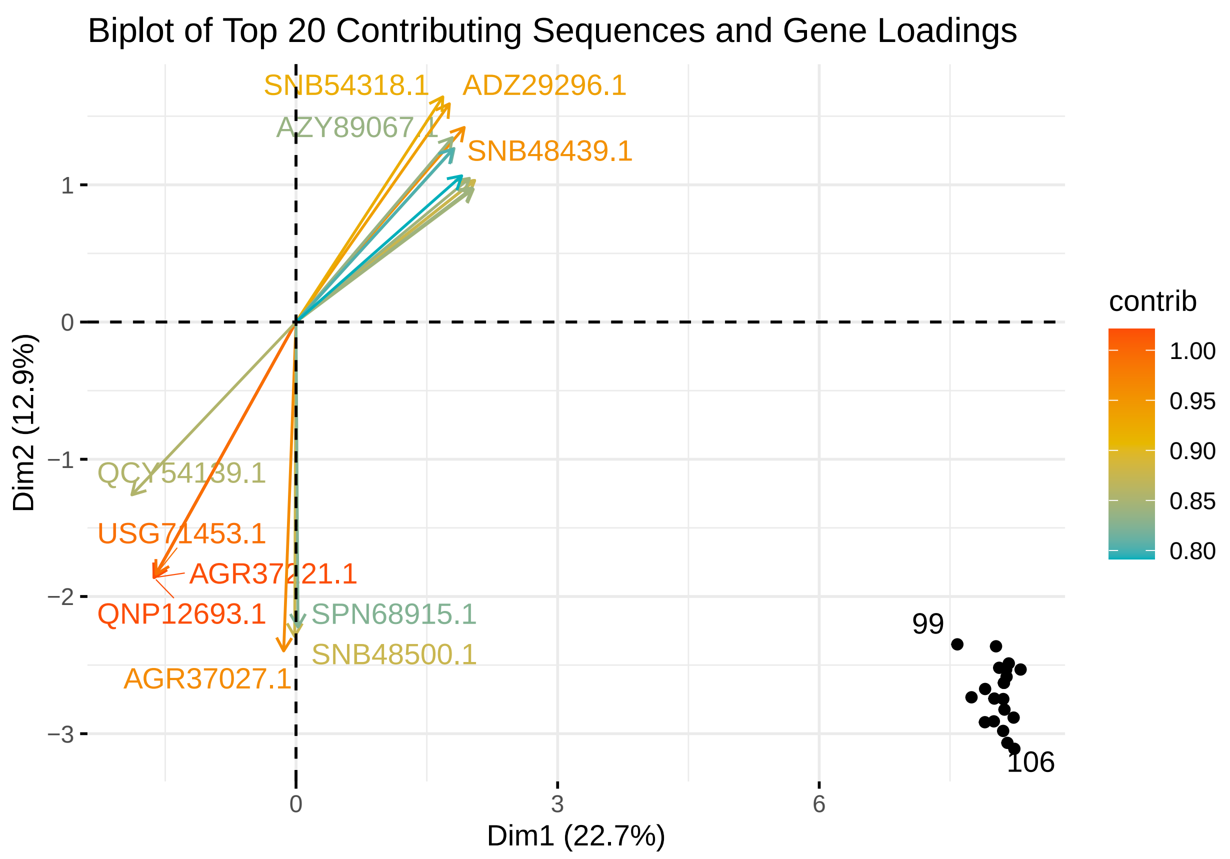
```
## Warning: ggrepel: 679 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
#Biplot top 20 influential scores and loadings
fviz_pca_biplot(pca2, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
                repel = TRUE, select.ind=list(contrib=20), select.var=list(contrib=20), max.overlaps=Inf) +
                ggtitle("Biplot of Top 20 Contributing Sequences and Gene Loadings")
```

```
## Warning: ggrepel: 18 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
## Warning: ggrepel: 9 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```r
### Summary: Compared to PCA.1, there's an increase in the proportion of variance explained by the firs
```

(3) PCA Alternative Analysis

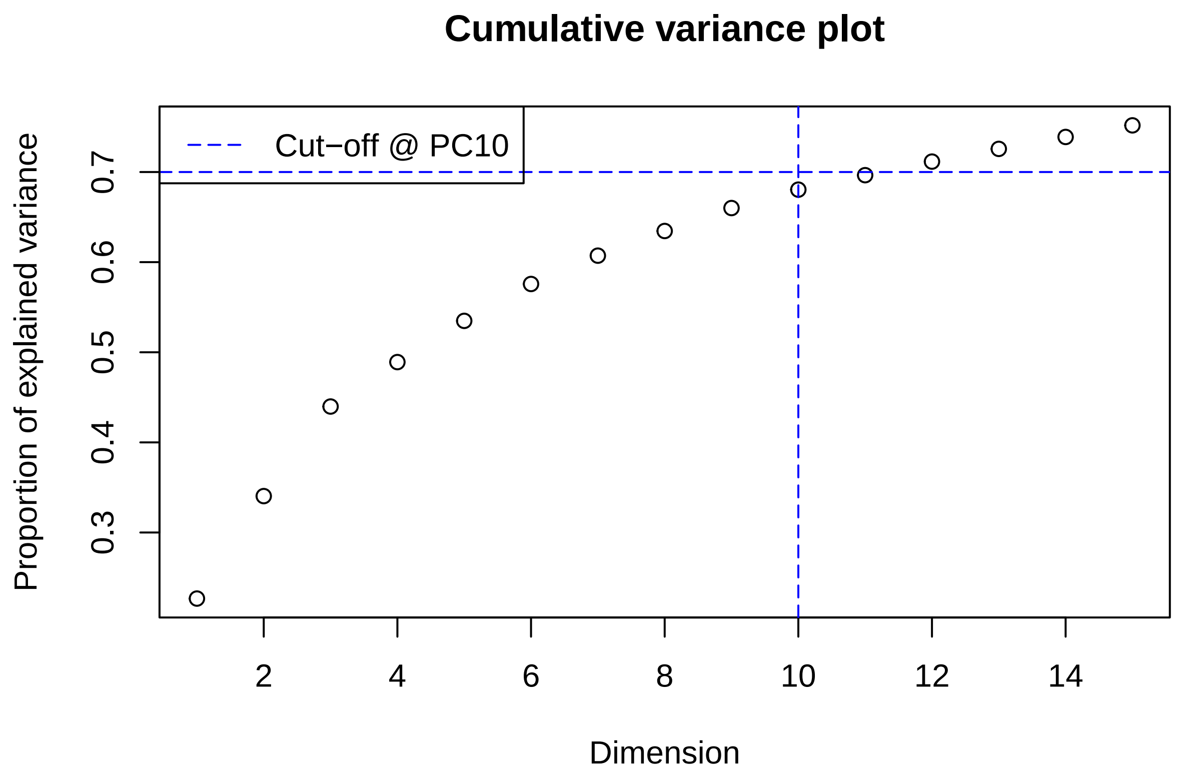
What happens when we drop duplicate observations within the same host-virus links (sequences with the same identical presence/absence of accessory genes as another sequence of the same host-virus link)?

```r
#Identify observations of the same host-virus links with identical presence/absence of accessory genes
genes3 <- genes
genes3$dup <- duplicated(genes3[,-c(1:2)])
table(genes3$dup)
```

```
##
## FALSE  TRUE
##   155    42
```

```r
### 42 dups

#Drop duplicate observations
genes3 <- genes3[genes3$dup==FALSE,]
genes3$dup=NULL
### 197 obs to 155 obs

#Apply PCA using stats::prcomp
```

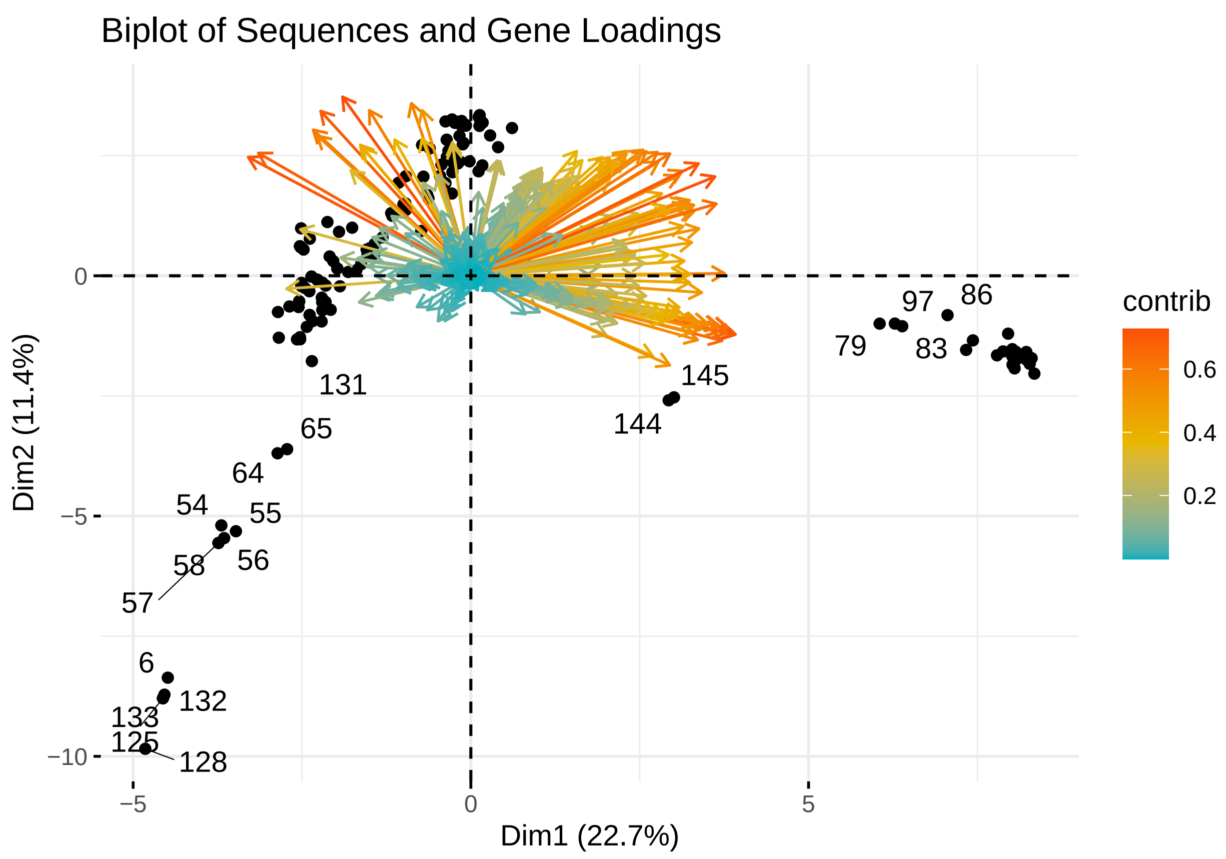
```
pca3 <- prcomp(genes3[,5:985])

#Plot cumulative variance to show the proportion of variance explained with each add'l PC
cumpro <- cumsum(pca3$sdev^2 / sum(pca3$sdev^2))
plot(cumpro[0:15], xlab = "Dimension", ylab = "Proportion of explained variance", main = "Cumulative va
abline(v = 10, col="blue", lty=5)
abline(h = 0.7, col="blue", lty=5)
legend("topleft", legend=c("Cut-off @ PC10"), col=c("blue"), lty=5, cex=1)
```



```
#Biplot sequences and gene loadings
fviz_pca_biplot(pca3, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
                repel = TRUE) +
                ggtitle("Biplot of Sequences and Gene Loadings")
```

```
## Warning: ggrepel: 136 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

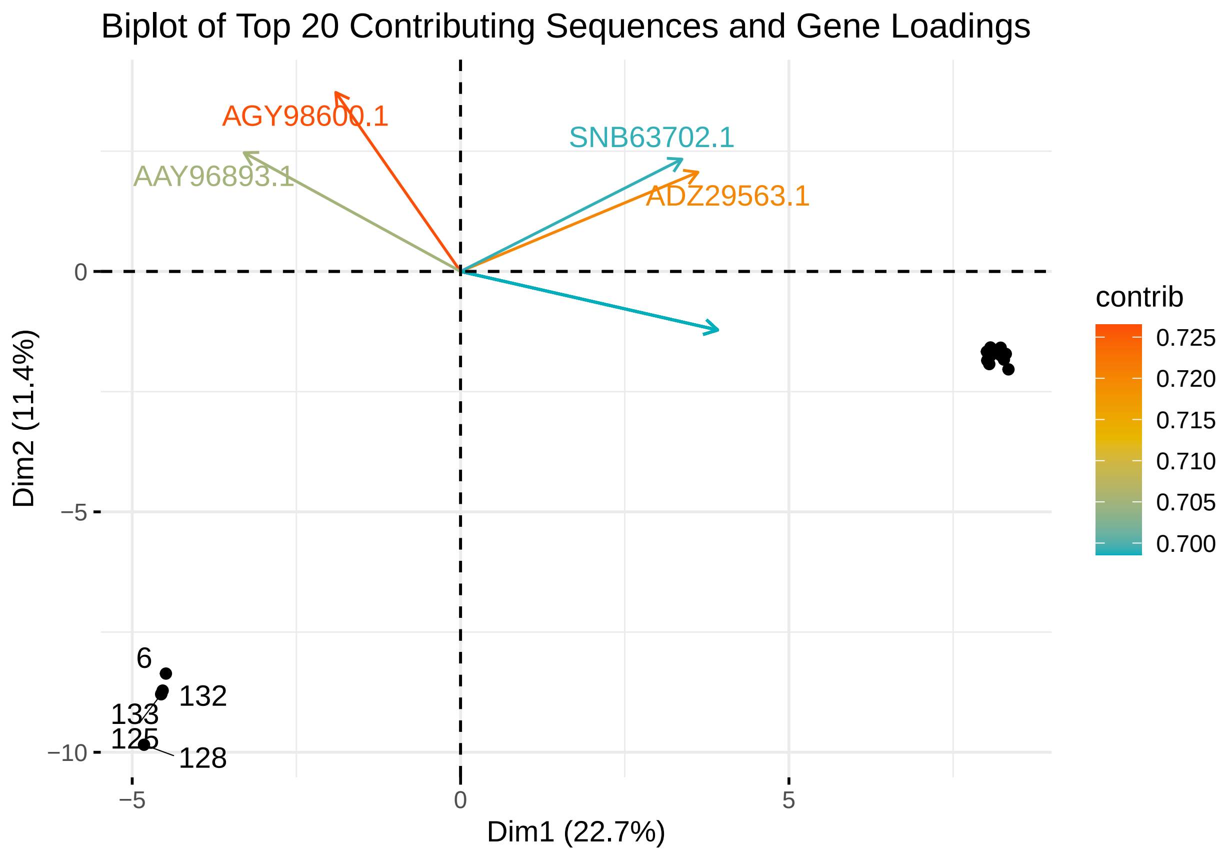
```
## Warning: ggrepel: 981 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
#Biplot top 20 influential scores and loadings
fviz_pca_biplot(pca3, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE, select.ind=list(contrib=20), select.var=list(contrib=20), max.overlaps=Inf)
             ggtitle("Biplot of Top 20 Contributing Sequences and Gene Loadings")
```

```
## Warning: ggrepel: 15 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
## Warning: ggrepel: 16 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



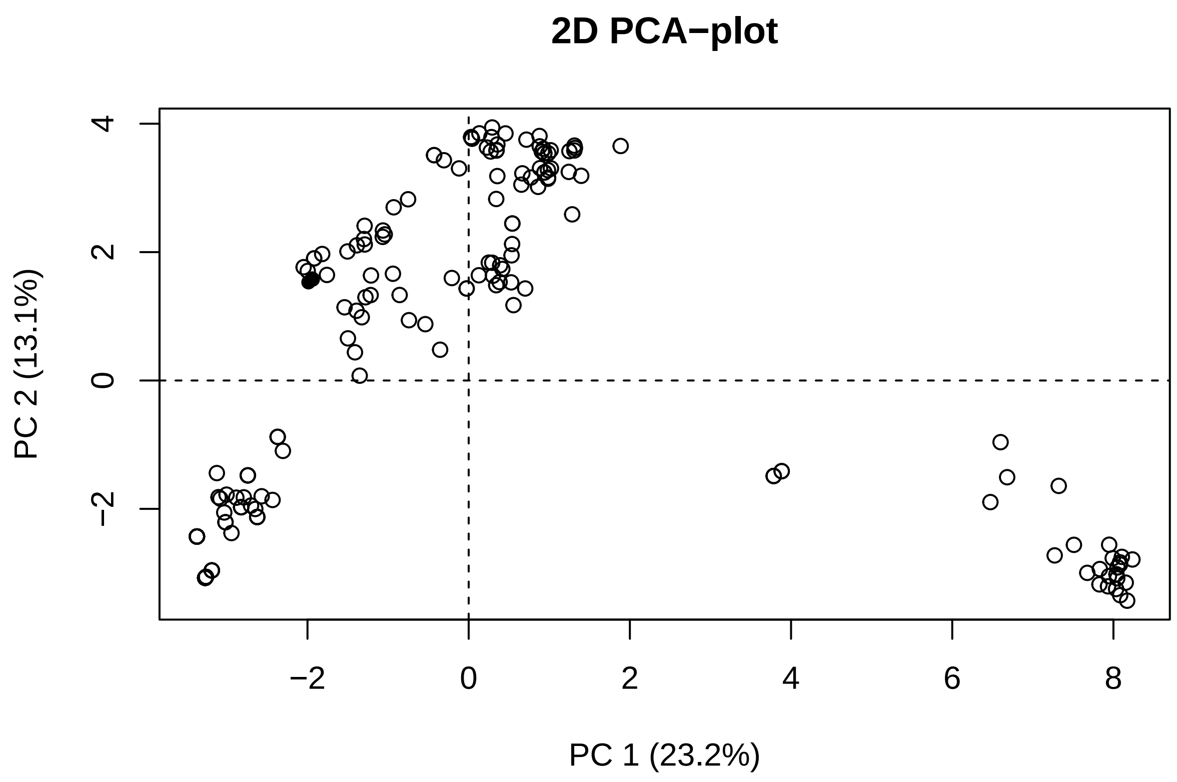
```
### Summary: Compared to PCA.1, there is a decrease in the proportion of variance explained by the firs
```

(4) PCA Alternative Analysis

What happens if we exclude the potential outliers from PCA, and then predict their scores and loadings?

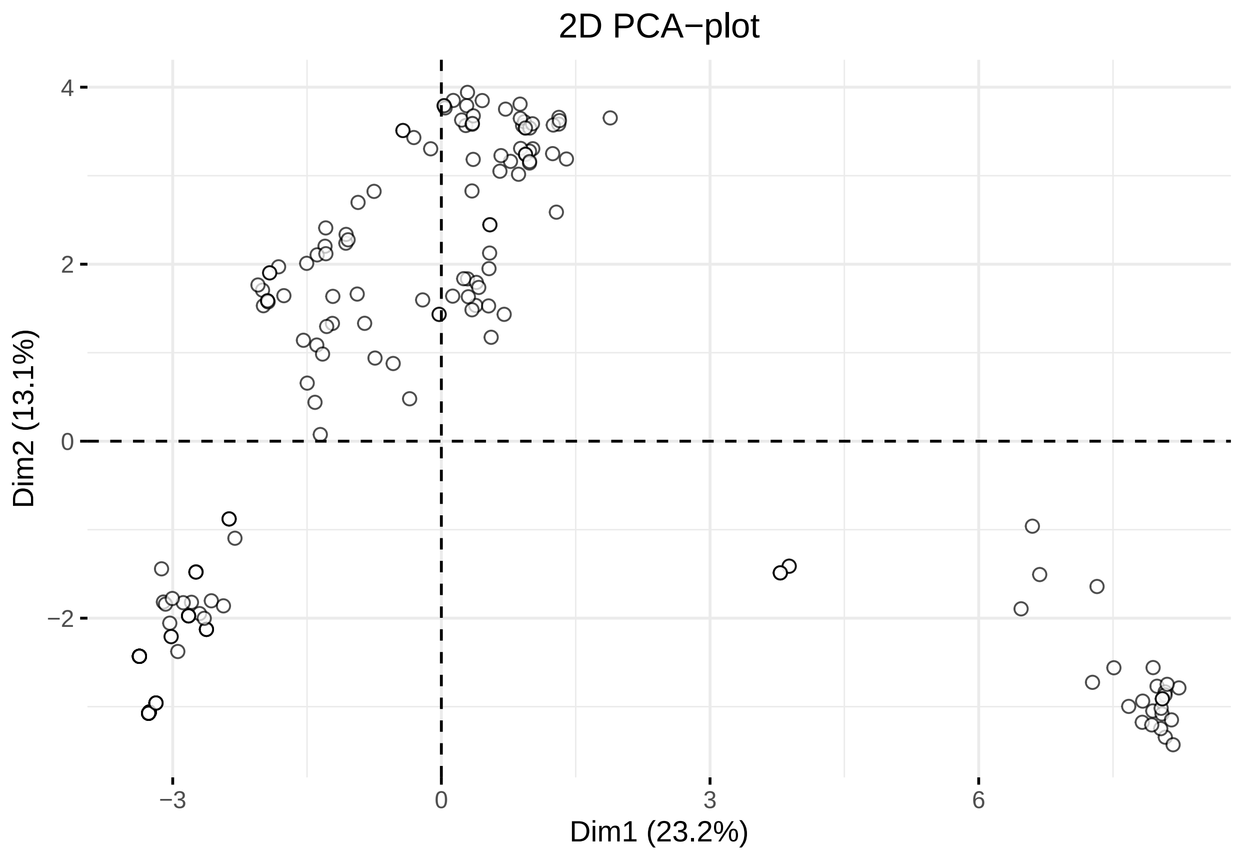
```
#Create df excluding outliers identified in PCA.3
genes4_in <- genes[!grepl("MT724769_1|MN346703_1|MT724770_1|DQ011155_1", genes$Genome),]

#Create df of outliers
genes4_out <- genes[grepl("MT724769_1|MN346703_1|MT724770_1|DQ011155_1", genes$Genome),]

#Apply PCA using stats::prcomp
pca4_in <- prcomp(genes4_in[,5:985])
relvar <- pca4_in$sdev^2 / sum(pca4_in$sdev^2)
relvar_per <- round(relvar*100,1)

#Prediction of PCs for outliers
pred <- predict(pca4_in, newdata=genes4_out)
pca4_pred <- pca4_in
pca4_pred$x <- rbind(pca4_pred$x, pred)

#Plot of individuals w/ outliers in shaded bullets
COLOR <- c(1:length(unique(genes$VirusSpecies)))
PCH <- c(1,16)
```

```r
pc <- c(1,2)
plot(pca4_in$x[,pc], cex=PCH[1],
     xlab=paste0("PC ", pc[1], " (", relvar_per[pc[1]], "%)"),
     ylab=paste0("PC ", pc[2], " (", relvar_per[pc[2]], "%)")
)
points(pred[,pc], pch=PCH[2]) + abline(h = 0, v=0, lty = 2) +
title("2D PCA-plot") + theme(plot.title = element_text(hjust = 0.5))
```



```
## NULL
```

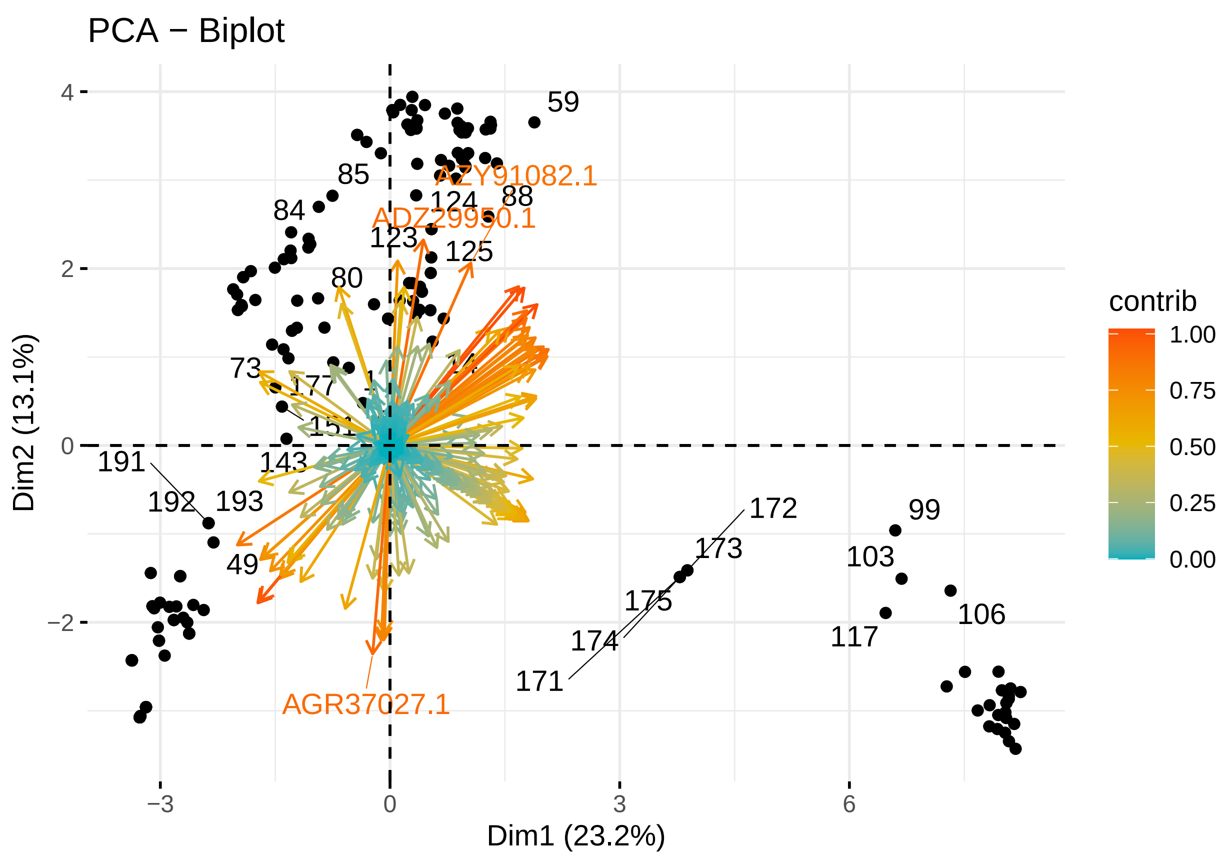
```r
#Plot of individuals w/ all unshaded bullets
fviz_pca_ind(pca4_pred, geom.ind = "point", pointshape = 21, pointsize = 2,
             col.ind = "black", addEllipses = F, label = "var",
             col.var = "black", palette = "rickandmorty", repel = TRUE,
             alpha.ind = 0.7) +
      ggtitle("2D PCA-plot") + theme(plot.title = element_text(hjust = 0.5))
```



```
#Biplot of individuals and variables
fviz_pca_biplot(pca4_pred, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
                repel = TRUE)
```

```
## Warning: ggrepel: 169 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

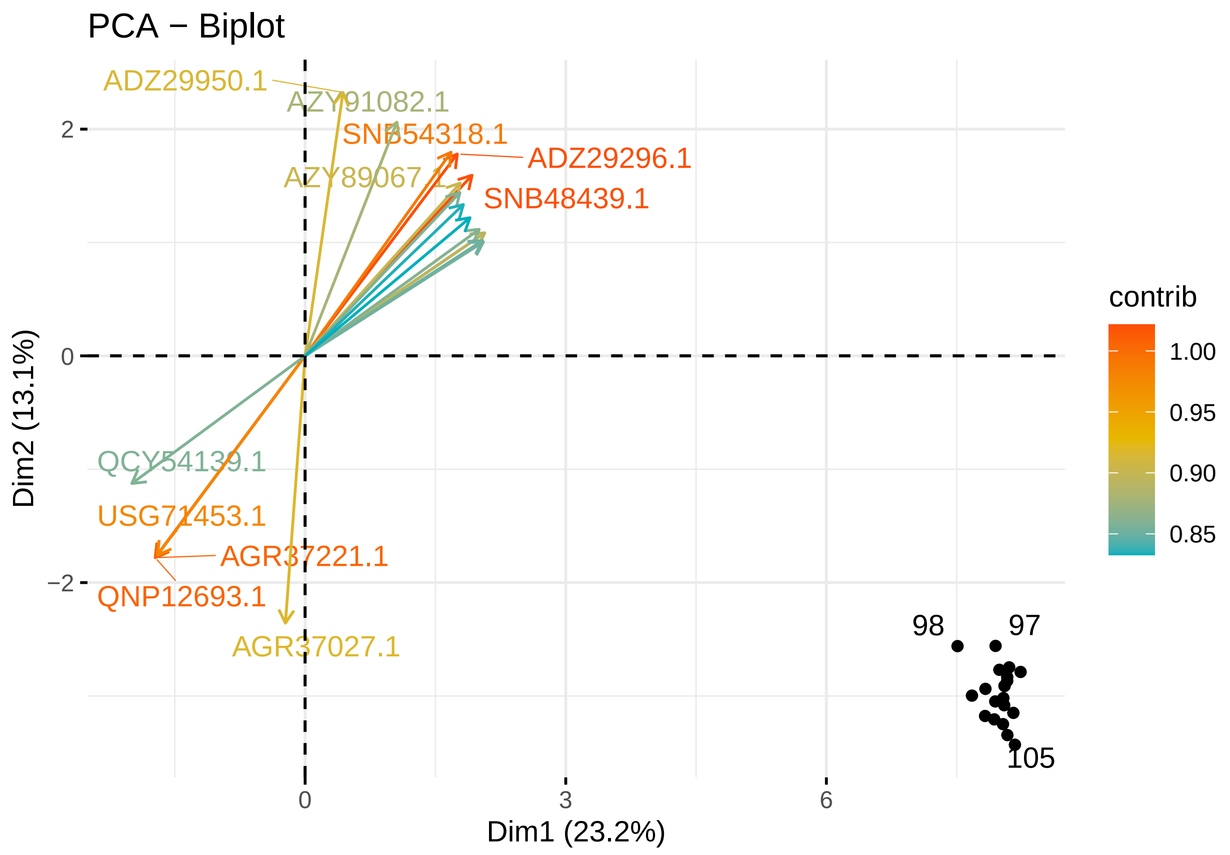
```
## Warning: ggrepel: 978 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
#Biplot of top 20 contributing individuals and variables
fviz_pca_biplot(pca4_pred, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
                repel = TRUE, select.ind=list(contrib=20), select.var=list(contrib=20))
```

```
## Warning: ggrepel: 17 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

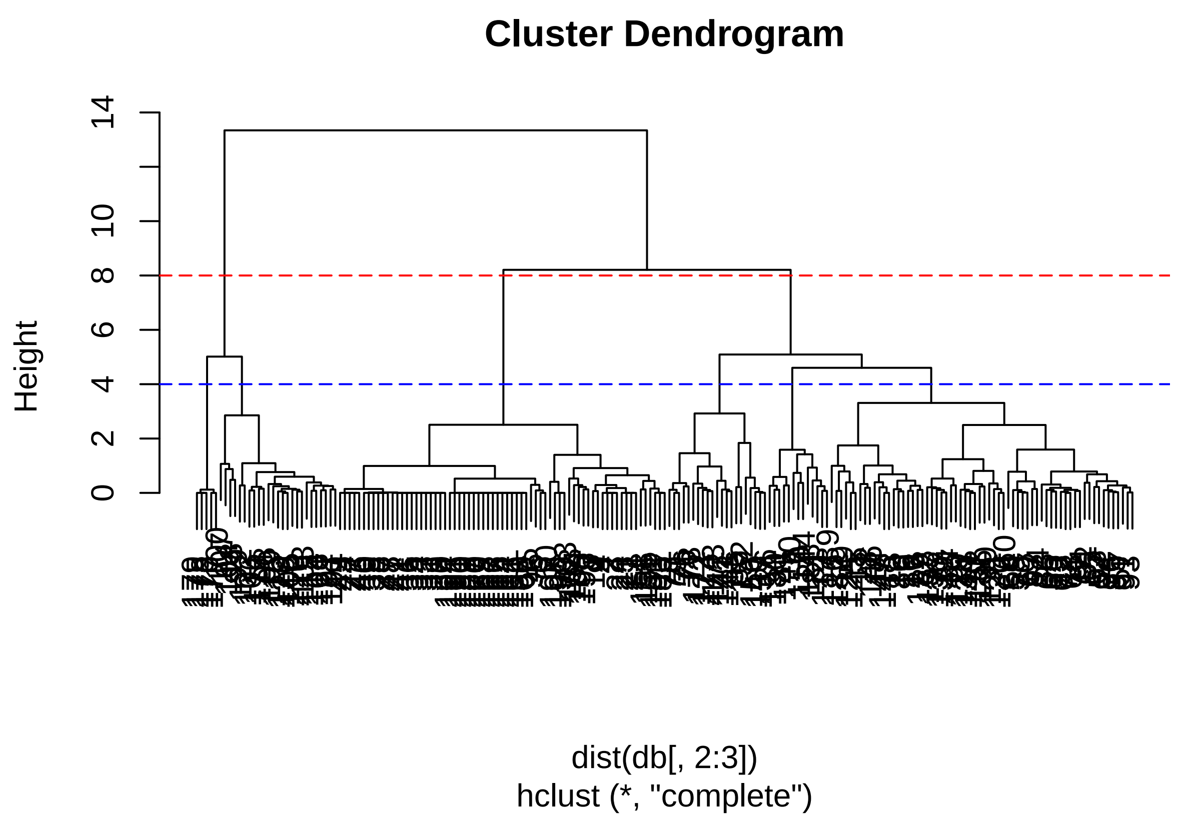
```
## Warning: ggrepel: 9 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
#Clean environment
rm(list=setdiff(ls(), c("genes", "pca1")))

### Summary: Predicted scores of outliers cluster in the fourth quadrant with other sequences. As in pr
```

PCA Hierarchical Cluster Analysis

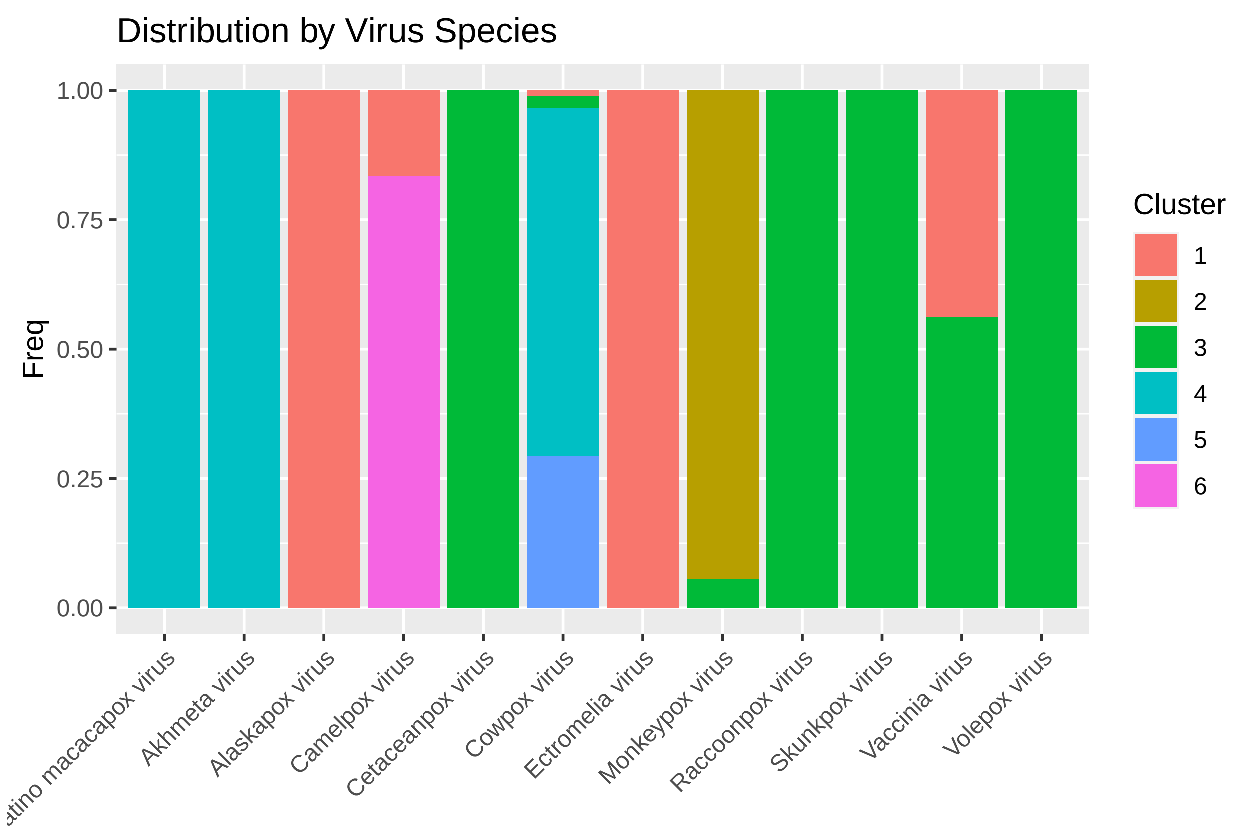
```
#Extract coordinates for individual sequences
ind.coord <- pca1$x
rownames(ind.coord) <- 1:nrow(ind.coord)
db <- cbind(genes$VirusSpecies, ind.coord)

#HCA on a set of dissimilarities for objects being clustered, wherein each object is assigned its own c
clusters <- hclust(dist(db[,2:3]))
plot(clusters)
abline(h = 8, col="red", lty=5)
abline(h = 4, col="blue", lty=5)
```



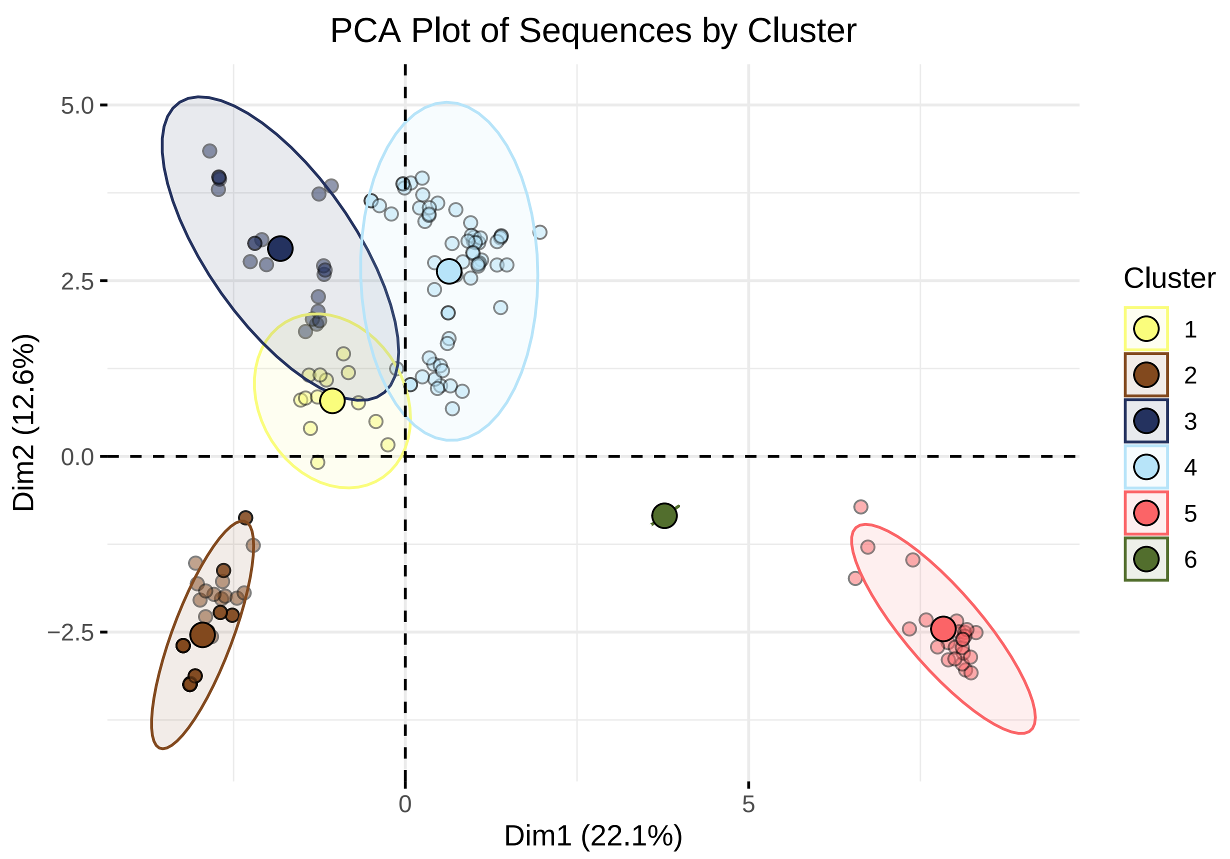
```r
#Cluster cut
clusterCut <- cutree(clusters, 6)
table(clusterCut)
```

```
## clusterCut
##  1  2  3  4  5  6
## 13 69 21 64 25  5
```

```r
#Prop tables by virus species
mytable<-table(clusterCut, genes$VirusSpecies)
mytable2 <- data.frame(prop.table(mytable,2))
ggplot(mytable2, aes(x = Var2, y = Freq, fill = clusterCut)) +
  geom_col() +
  labs(fill='Cluster') +
  theme(axis.title.x = element_blank(), axis.text.x = element_text(angle=45,hjust=1)) +
  ggtitle("Distribution by Virus Species")
```

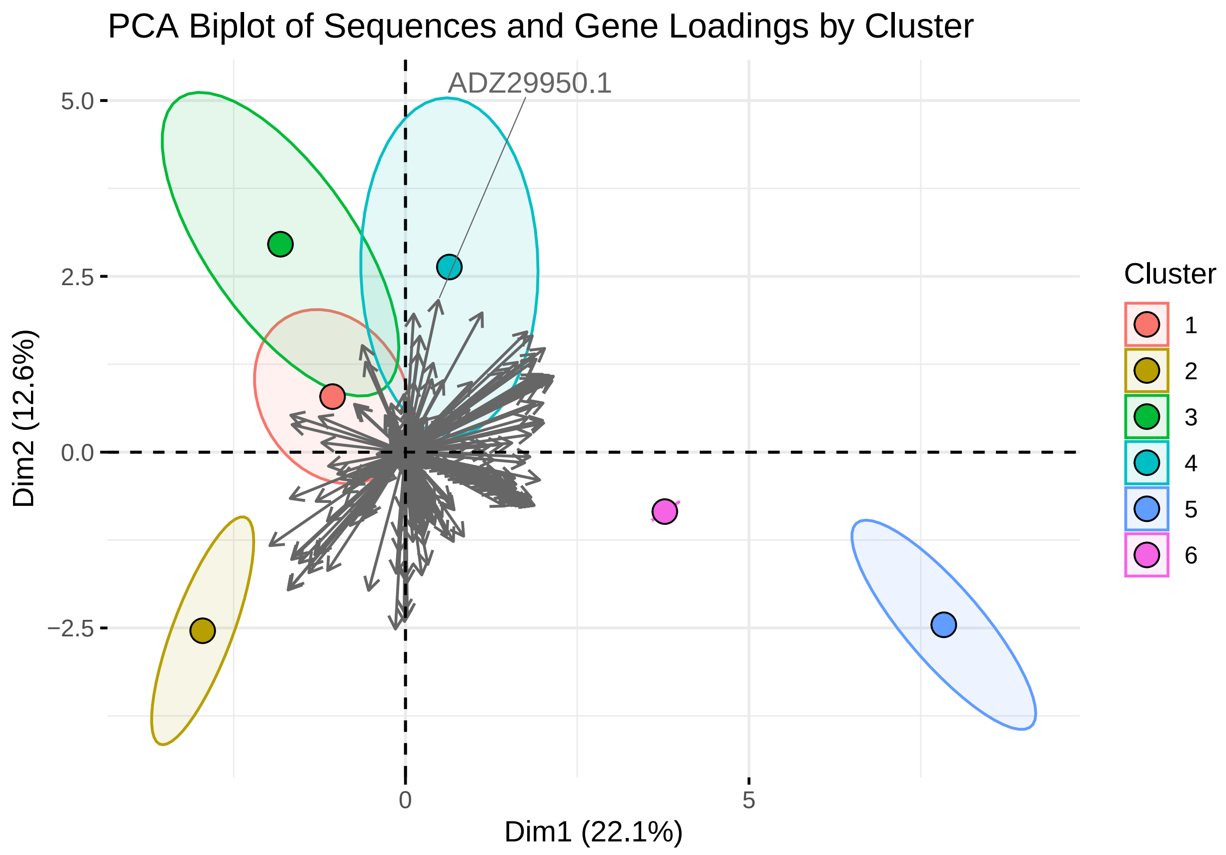


```
#Re-run PCA to color by cluster
  #add cluster to original db
  genes1<-data.frame(cbind(genes,clusterCut))
  genes1$clusterCut <- as.factor(genes1$clusterCut)

#Run PCA as before, but now grouping by cluster
pca1 <- prcomp(genes1[,5:985])       #scaling/centering not appropriate
# pca_relvar <- pca$sdev^2 / sum(pca$sdev^2)
# pca_relvar_per <- round(pca_relvar*100,1)

fviz_pca_ind(pca1, geom.ind = "point", pointshape = 21,
             pointsize = 2,
             fill.ind = genes1$clusterCut,
             col.ind = "black",
             addEllipses = TRUE,
             label = "var",
             col.var = "black",
             palette = "rickandmorty",
             repel = TRUE,
             legend.title = "Cluster",
             alpha.ind = 0.5) +
  ggtitle("PCA Plot of Sequences by Cluster") +
  theme(plot.title = element_text(hjust = 0.5))
```



```
fviz_pca_biplot(pca1, geom.ind = "point", pointshape = 21,
             pointsize = 2,
             fill.ind = genes1$clusterCut,
             col.ind = "black",
             label = "var",
             repel = TRUE,
             legend.title = "Cluster",
             addEllipses = TRUE,
             pallete = "lancet",
             alpha.ind = 0,
             col.var = "grey40") +
    ggtitle("PCA Biplot of Sequences and Gene Loadings by Cluster")
```

```
## Warning: ggrepel: 980 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
db.vf <- dplyr::select(genes1, Sequence, clusterCut)
# write.csv(db.vf,"clusters.csv", row.names = F)

#Clean environment
rm(clusters, db, db.vf, genes1, ind.coord, pca1, clusterCut, mytable, mytable2)
```

Non-Negative Matrix Factorization

PCA: Goal is to reduce dimensions while maintaining maximal variance - Each PC is a linear combination of uncorrelated attributes/features - # of PCs is limited by # of samples using the eigenvalue decomposition method (?) NMF: Like PCA, except coefficients in linear combination (weight of each base) must be non-negative - Explains the dataset through factoring into two non-negative matrices in such a way that the distance between the original matrix and subset matrices are minimized - Update rule implemented by NMF algorithms are multiplicative instead of additive - multiplicative update rules can hold nonnegativity easily with nonnegativity initalization - The number of learned basis experiments is not as limited by number of samples - NMF is stochastic (PCA is deterministic) - Can be much more stable and well-specified reconstruction when assumptions are appropriate - Excellent for separating out additive factors

```
#Resources: https://rpubs.com/JanpuHou/300168; https://aarmey.github.io/ml-for-bioe/public/Wk4-Lecture7

#Run NMF defaulting to 'brunet' algorithm and 'random' seed on initialization
genes_mat <- subset(genes,select=-c(Genome,VirusSpecies,HostGenus))
mat <- as.matrix(genes_mat[,-1])
rownames(mat) <- genes_mat[,1] %>% pull()
```

```r
class(mat) <- "numeric"

start_time <- Sys.time()
# nmf <- nmf(genes[,5:10], 6, "brunet") #Time difference of 0.2662811 secs
nmf <- nmf(genes[,5:100], 96, "brunet") #Time difference of 17.2904 secs
end_time <- Sys.time()
end_time - start_time
```

```
## Time difference of 18.08848 secs
```

```r
#Summarize results
nmf #explore object
```

```
## <Object of class: NMFfit>
##  # Model:
##   <Object of class:NMFstd>
##   features: 197
##   basis/rank: 96
##   samples: 96
##  # Details:
##   algorithm:  brunet
##   seed:  random
##   RNG: 10403L, 568L, ..., 2033055286L [b2dbe2d415da902b53105edcb4c4184d]
##   distance metric:  'KL'
##   residuals:  1.025041
##   Iterations: 2000
##   Timing:
##      user  system elapsed
##    16.935   0.137  17.080
```

```r
fit(nmf) #retrieve fitted model
```

```
## <Object of class:NMFstd>
## features: 197
## basis/rank: 96
## samples: 96
```

```r
V.hat <- fitted(nmf) #retrieve estimated target matrix and its dimensions
dim(V.hat)
```

```
## [1] 197  96
```

```r
summary(nmf)
```

```
##              rank sparseness.basis  sparseness.coef  silhouette.coef
##       96.00000000       0.27598885       0.26161709      -0.03010824
## silhouette.basis        residuals            niter              cpu
##       -0.06725388       1.02504111    2000.00000000      16.93500000
##          cpu.all             nrun
##       16.93500000       1.00000000
```

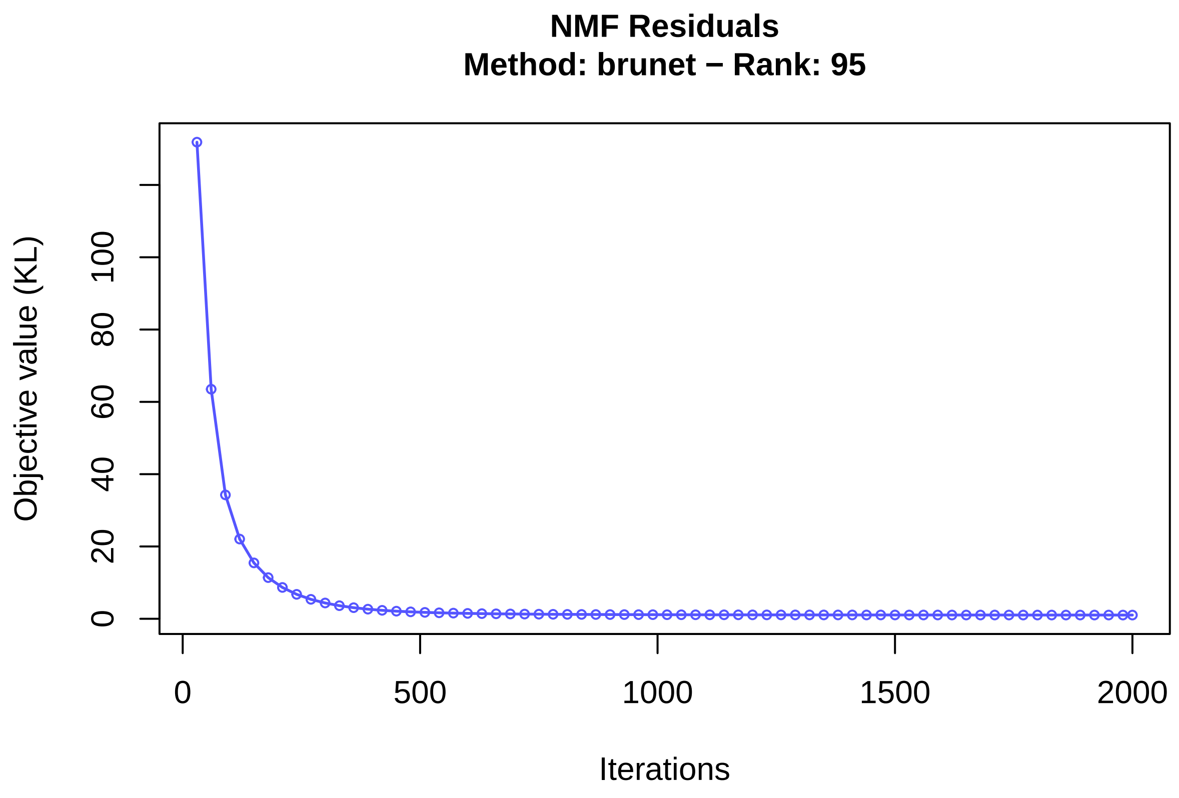
```
#Perform multiple runs to achieve stability because the seeding method is stochastic (random): the retu
start_time <- Sys.time()
nmf_multi <- nmf(genes[,5:100], 96, nrun=5, .opt='v') #.opt='v' tries to run in parallel using all core
```

```
## NMF algorithm: 'brunet'

## Multiple runs: 5

## Mode: parallel (7/8 core(s))

## Runs: |                                                          Runs: |
## System time:
##    user  system elapsed
##  97.876   1.950  28.128
```

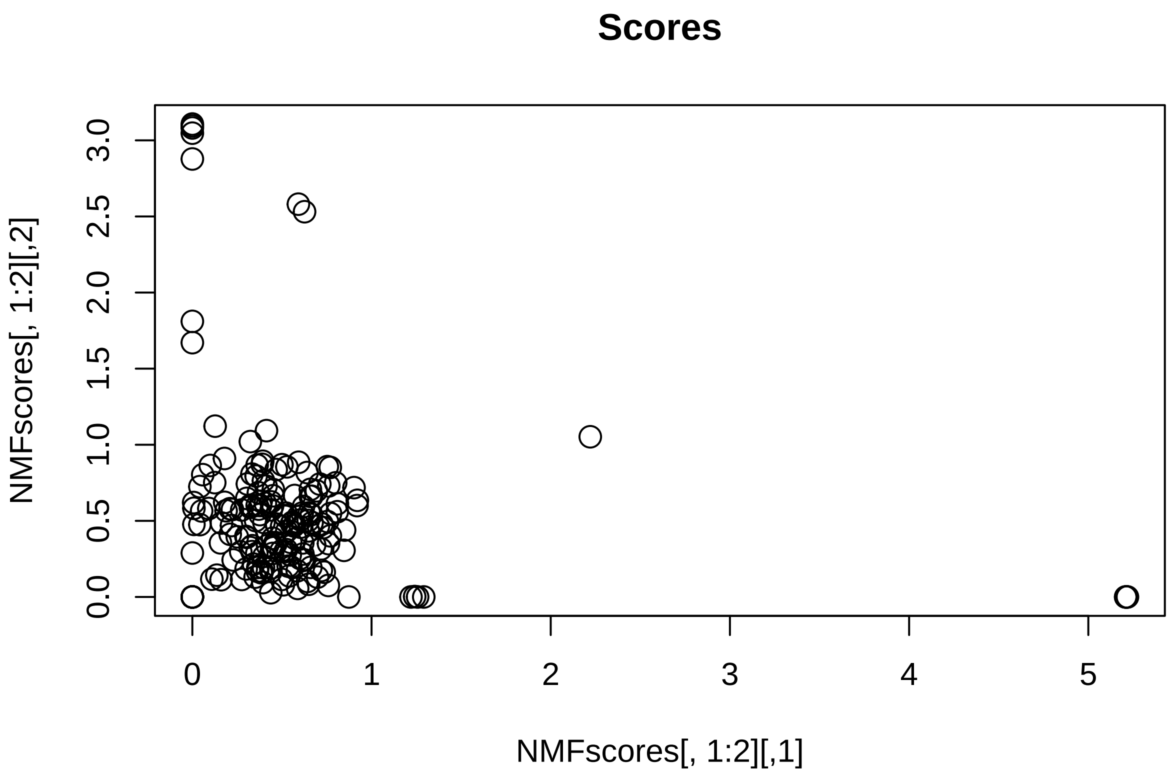
```
end_time <- Sys.time()
end_time - start_time #Time difference of 25.05052 secs
```

```
## Time difference of 28.6035 secs
```

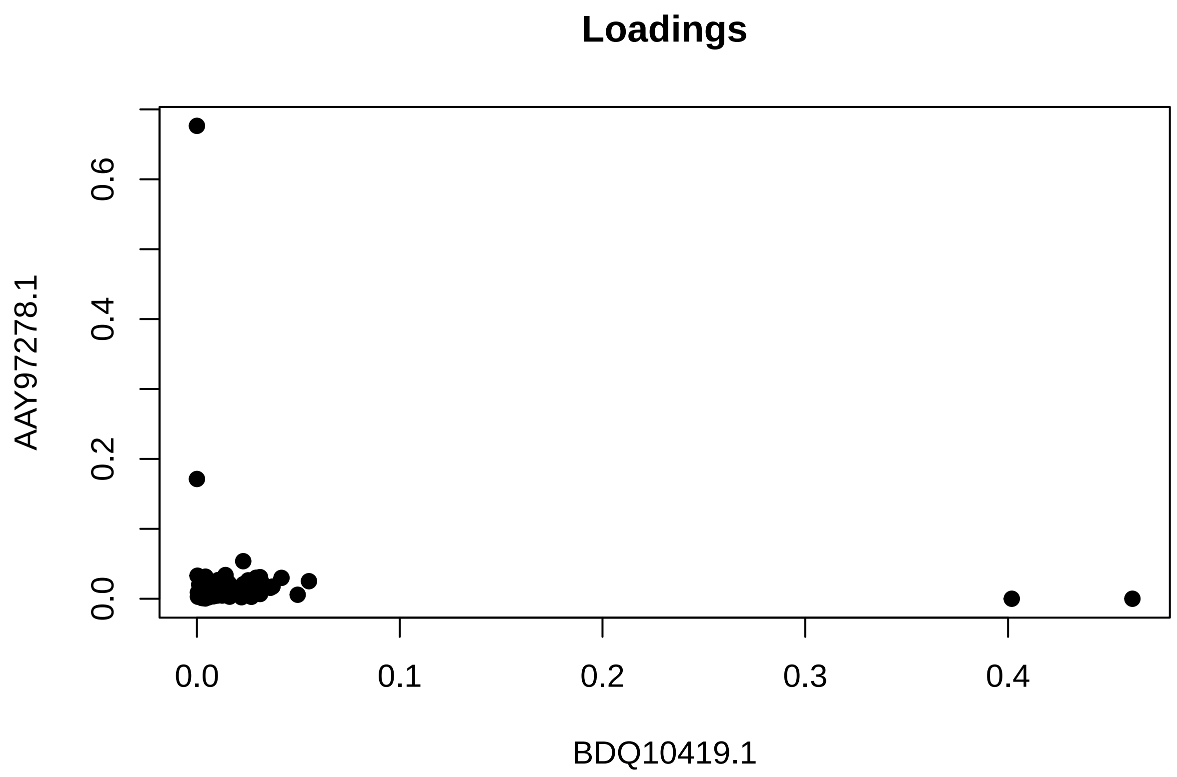
```
nmf <- nmf(genes[,5:100], 95, .opt='t')
plot(nmf)
```



```
NMFscores <- nmf_multi@fit@W
NMFloadings <- nmf_multi@fit@H

plot(NMFscores[,1:2],  # x and y data
     pch=21,           # point shape
     cex=1.5,          # point size
     main="Scores"     # title of plot
)
```



```
plot(NMFloadings[,1:2],   # x and y data
     pch=21,              # point shape
     bg="black",          # point color
     cex=1,               # point size
     main="Loadings"      # title of plot
)
```



MCA of viral accessory genes

Multiple Correspondence Analysis (MCA) for dimension reduction of categorical variables.

```
#Subset data and reformat gene variables as factor
###Note: Actual data for MCA is pending. For the purposes of this exercise, I am manipulating the prese
genes_cat <- subset(genes,select=-c(Genome,VirusSpecies,HostGenus))
genes_cat[] <- lapply(genes_cat, as.character)
rownames <- genes$Sequence
genes_cat[,-1] <- lapply(genes_cat[,-1], factor)
genes_cat$Sequence=NULL
rownames(genes_cat) <- rownames
```

```
## Warning: Setting row names on a tibble is deprecated.
```

```
#str(genes_cat)

#Apply MCA using FactoMineR::MCA
mca = MCA(genes_cat, graph = FALSE)
# pca_relvar <- pca$sdev^2 / sum(pca$sdev^2)
# pca_relvar_per <- round(pca_relvar*100,1)

#List and summarize MCA results
print(mca)
```

```
## **Results of the Multiple Correspondence Analysis (MCA)**
## The analysis was performed on 197 individuals, described by 981 variables
## *The results are available in the following objects:
##
##    name              description
## 1  "$eig"            "eigenvalues"
## 2  "$var"            "results for the variables"
## 3  "$var$coord"      "coord. of the categories"
## 4  "$var$cos2"       "cos2 for the categories"
## 5  "$var$contrib"    "contributions of the categories"
## 6  "$var$v.test"     "v-test for the categories"
## 7  "$ind"            "results for the individuals"
## 8  "$ind$coord"      "coord. for the individuals"
## 9  "$ind$cos2"       "cos2 for the individuals"
## 10 "$ind$contrib"    "contributions of the individuals"
## 11 "$call"           "intermediate results"
## 12 "$call$marge.col" "weights of columns"
## 13 "$call$marge.li"  "weights of rows"
```

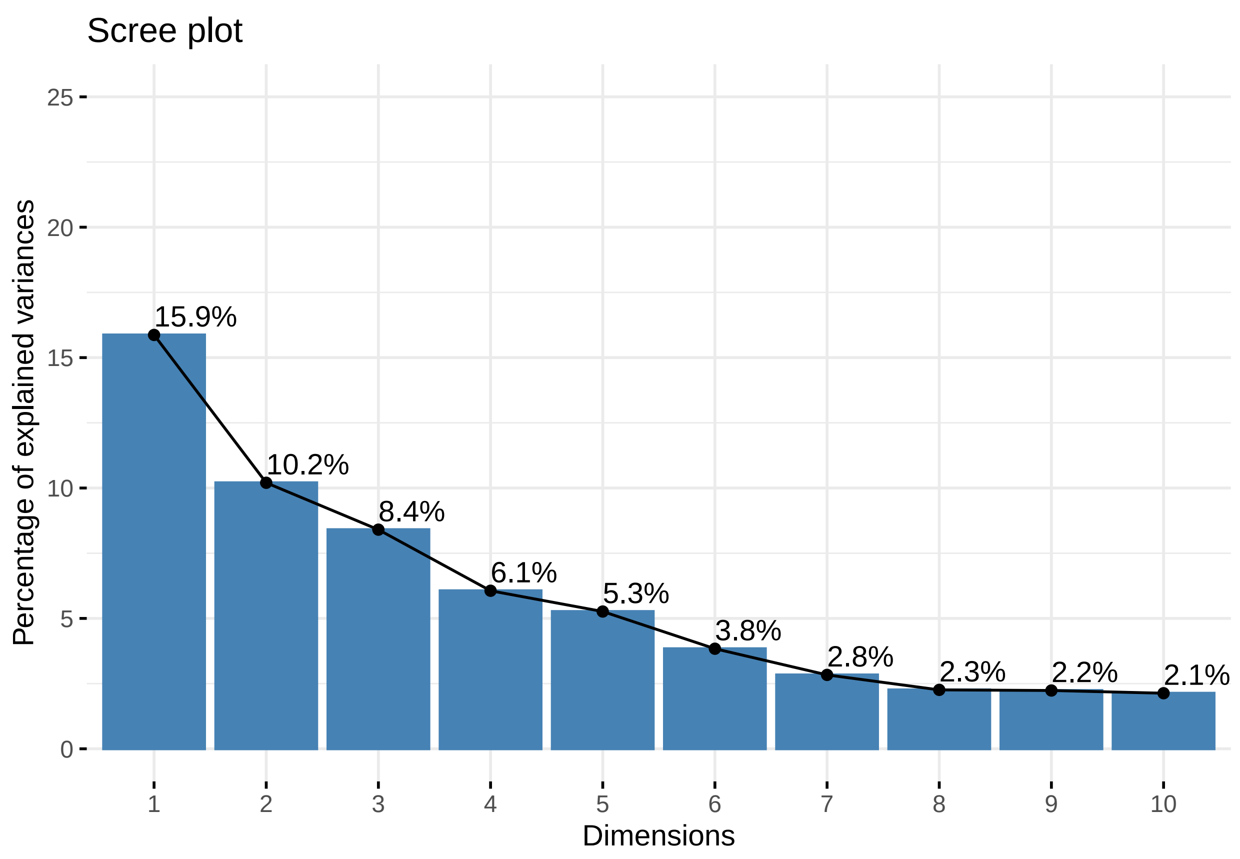
```
# summary(mca)
head(mca$ind$coord) #sequence (individuals)
```

```
##         Dim 1       Dim 2       Dim 3      Dim 4        Dim 5
## 1 -0.07348303 0.002279623 -0.07993247 0.07256654  0.006247373
## 2 -0.05469522 0.010009442 -0.18044039 0.06687608 -0.002436931
## 3 -0.05469522 0.010009442 -0.18044039 0.06687608 -0.002436931
## 4 -0.05469522 0.010009442 -0.18044039 0.06687608 -0.002436931
## 5 -0.06100347 0.004533480 -0.19985039 0.07484893  0.001739553
## 6 -0.06100347 0.004533480 -0.19985039 0.07484893  0.001739553
```

```
head(mca$var$coord) #genes (variables)
```

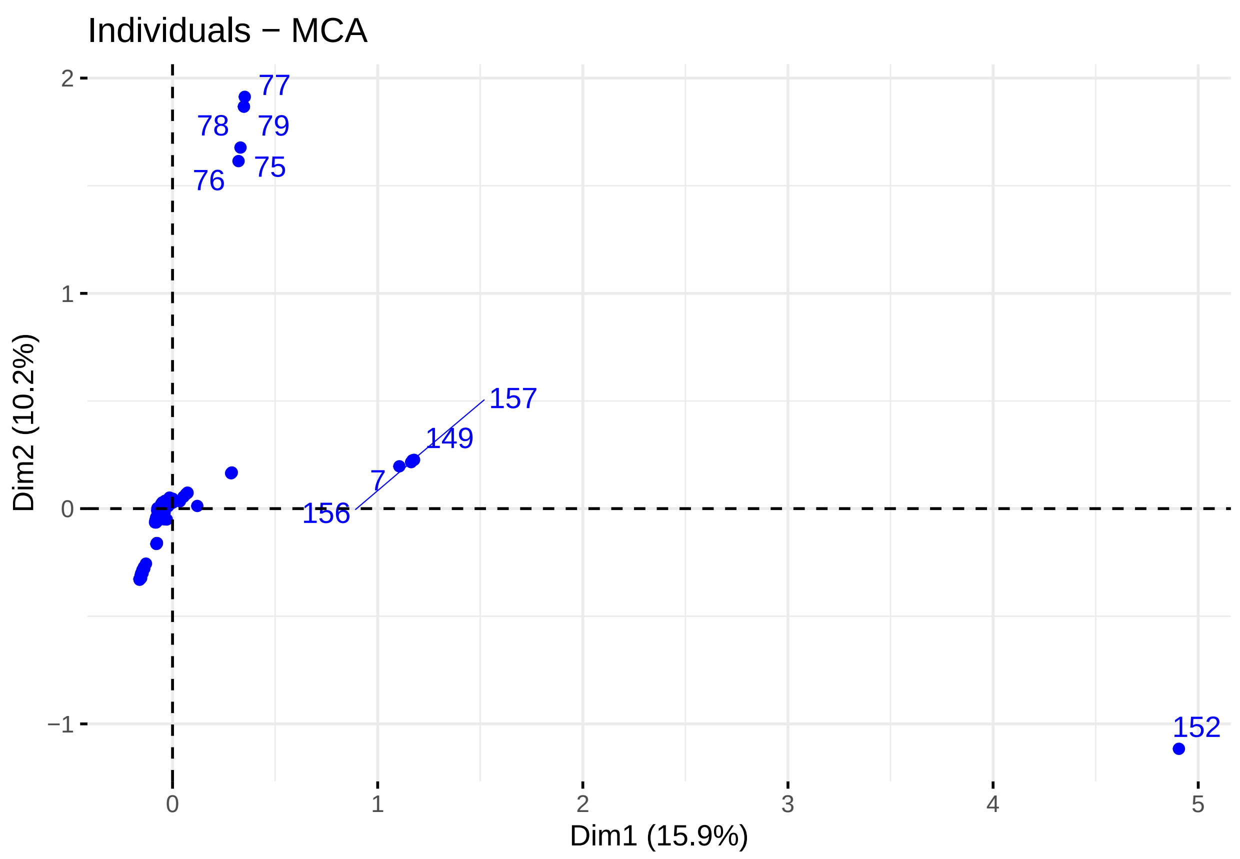
```
##                    Dim 1        Dim 2         Dim 3         Dim 4         Dim 5
## BDQ10419.1_0  7.54493918 -1.439016813  0.1672657656 -0.0461567526 -0.0516897146
## BDQ10419.1_1 -0.07738399  0.014759147 -0.0017155463  0.0004734026  0.0005301509
## AAY97278.1_0  2.93728181  0.696055379 -0.3359158345 -6.0877394658 -0.4089330107
## AAY97278.1_1 -0.04542188 -0.010763743  0.0051945748  0.0941403010  0.0063237064
## BDQ10401.1_0  4.13293482 -0.459503058 -0.0274152067 -0.7827244719 -0.0577198941
## BDQ10401.1_1 -0.08565668  0.009523379  0.0005681908  0.0162222688  0.0011962672
```

```
#Screeplot - Variance (Eigenvalues)
#mca$eig
fviz_eig(mca, addlabels = TRUE, ylim = c(0, 25))
```



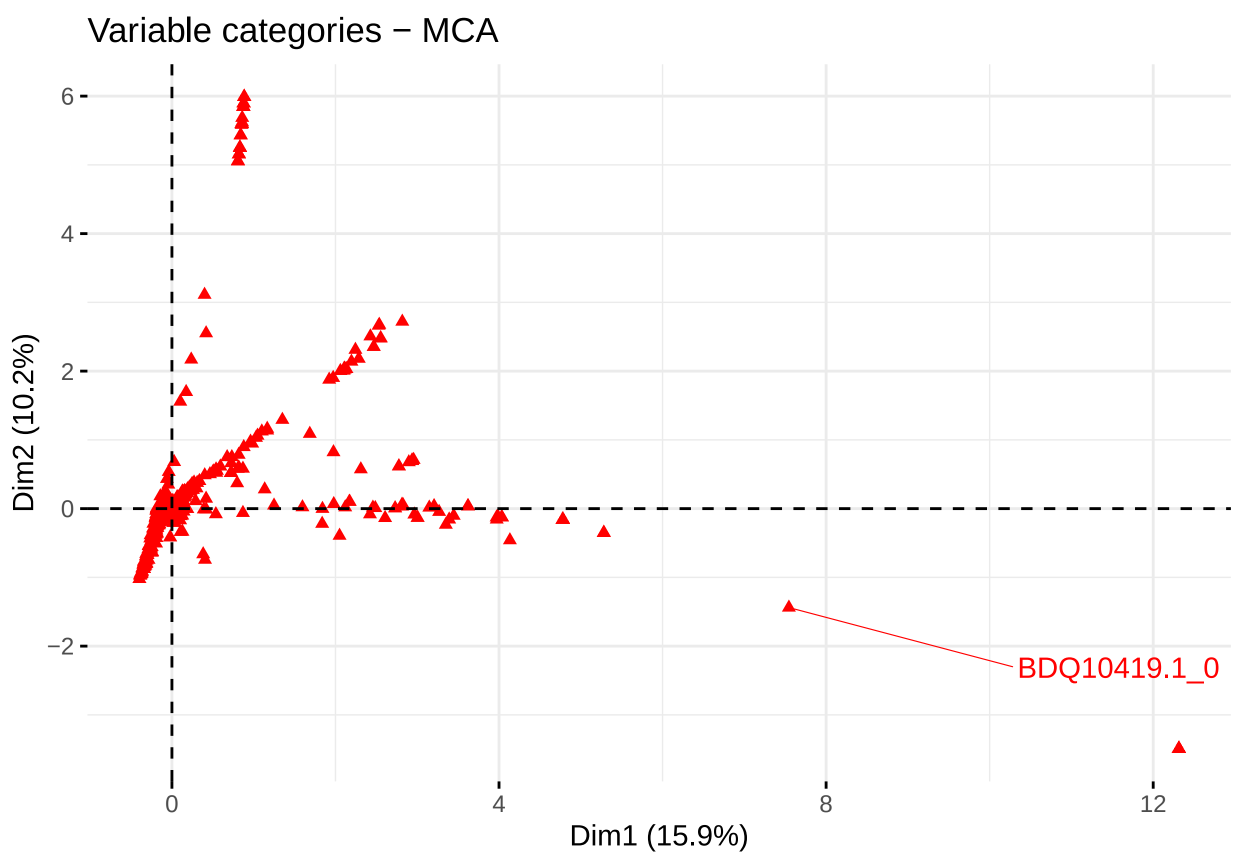
```
#Plots of individuals
fviz_mca_ind(mca, repel=TRUE)
```

```
## Warning: ggrepel: 187 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```r
#Plots of MCA variables 1 and 2
fviz_mca_var(mca, repel = TRUE) ##
```

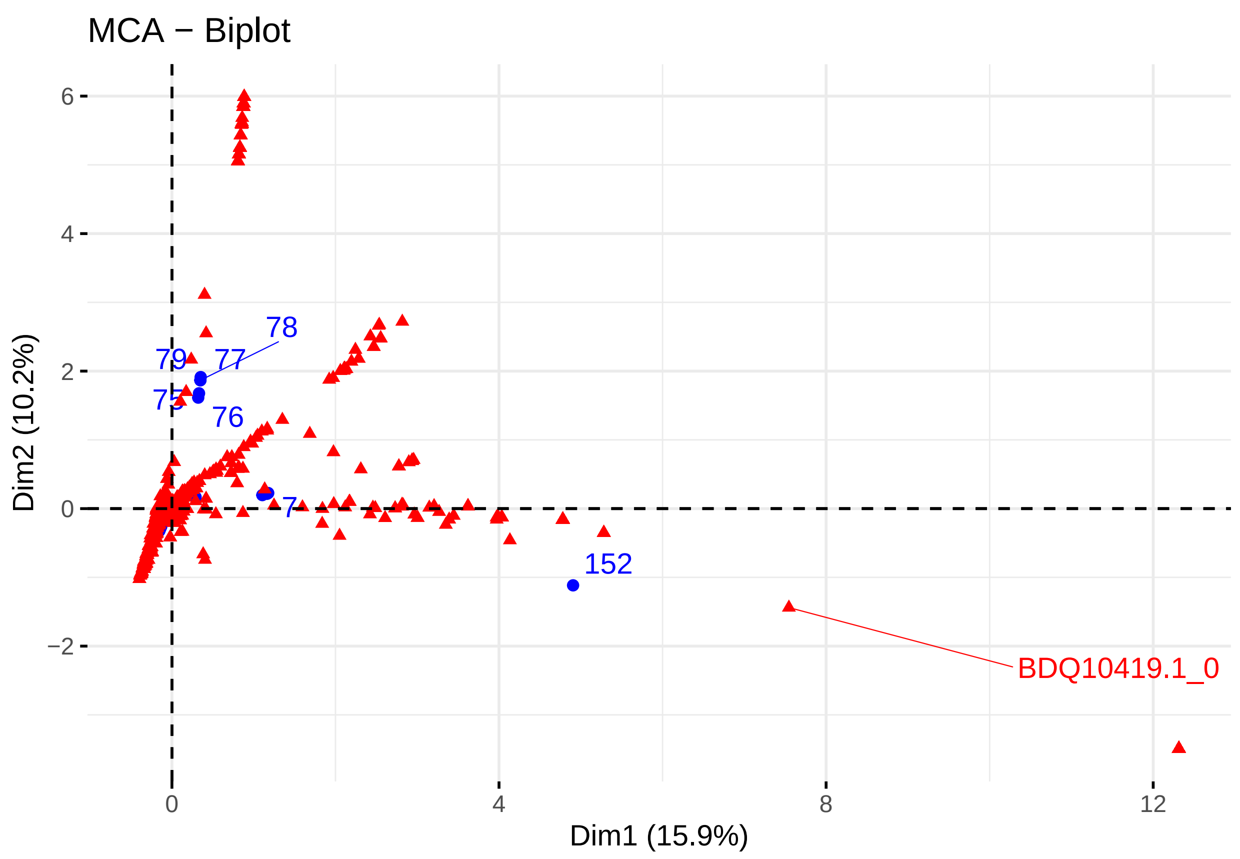
```
## Warning: ggrepel: 1961 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



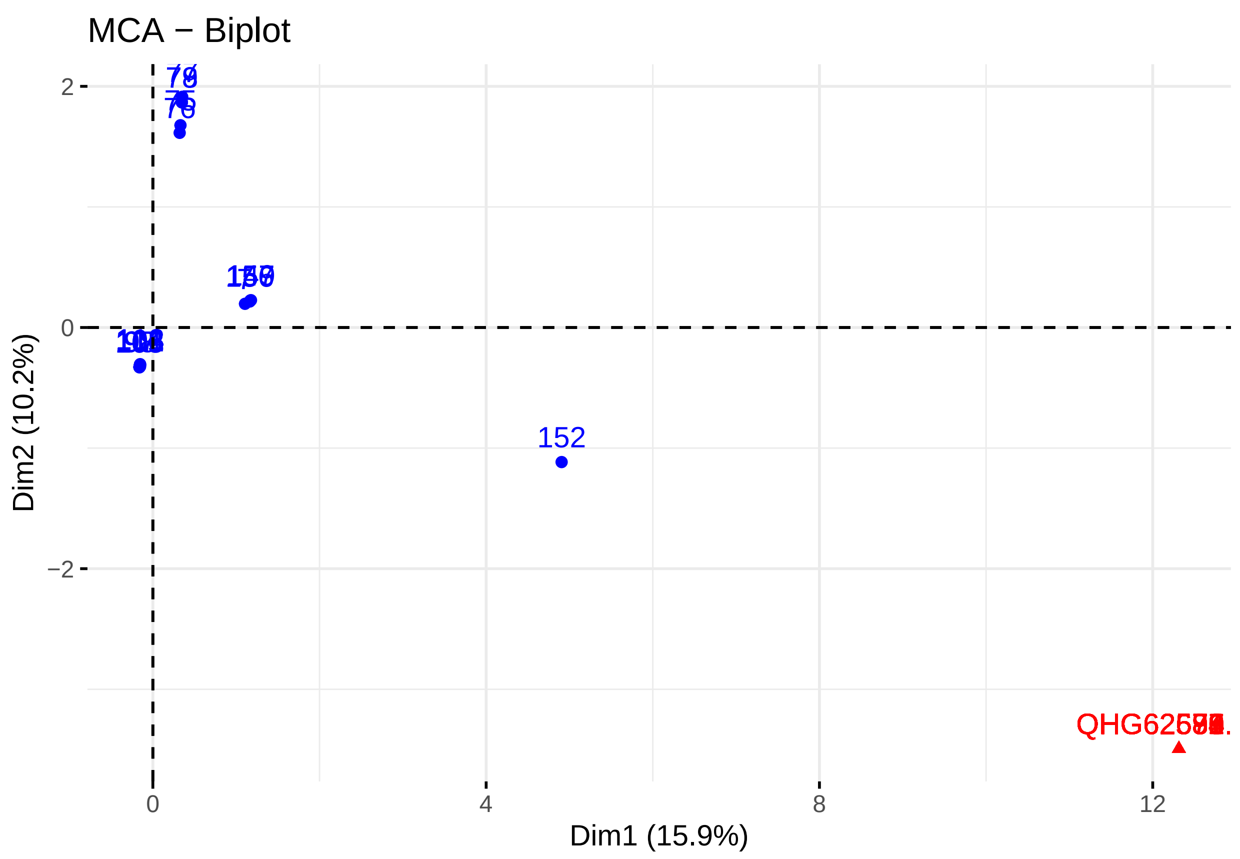
```
#Biplot
fviz_mca_biplot(mca, repel = TRUE)
```

```
## Warning: ggrepel: 190 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

```
## Warning: ggrepel: 1961 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
fviz_mca_biplot(mca, repel = FALSE, select.ind=list(contrib=20), select.var=list(contrib=20))
```



```
#Clean environment
rm(genes,genes_cat,genes_mat,genes_tax,mat,mca,pc_cutoff)
```

```
## Warning in rm(genes, genes_cat, genes_mat, genes_tax, mat, mca, pc_cutoff):
## object 'genes_tax' not found

## Warning in rm(genes, genes_cat, genes_mat, genes_tax, mat, mca, pc_cutoff):
## object 'pc_cutoff' not found
```