DAPC

2023-05-05

1. Load packages

```
library(DESeq2)
library(tidyverse)
library(adegenet)
```

2. Data transformation

2.a. Import data sets The following data sets are provided by Dr. Rebecca L. Young.

```
frog_raw_counts_greaterthan5 <- read.table(file = "/stor/work/Bio321G_RY_Spring2023/MiniProjects/Poison
                                           row.names = 1,
                                           header = TRUE,
                                           sep = ",")

frog_metadata <- read.delim(file = "/stor/work/Bio321G_RY_Spring2023/MiniProjects/PoisonFrogLivers/Meta
```

2.b. Normalization Since the gene length is not provided, we will use DeSeq2 to normalize our raw counts

```
dds <- DESeqDataSetFromMatrix(countData=frog_raw_counts_greaterthan5, design = ~condition,
                              colData=frog_metadata)

dds <- estimateSizeFactors(dds)

normalized_counts_greater_5 <- counts(dds, normalized = TRUE)
normalized_counts_greater_5 <- as.data.frame(normalized_counts_greater_5)   # Convert to data.frame
```

3. Discriminant Analysis Principal Component (DAPC)

DAPC can be used to infer the number of clusters of genetically related individuals. In this multivariate statistical approach variance in the sample is partitioned into a between-group and within- group component,

in an effort to maximize discrimination between groups. In DAPC, data is first transformed using a principal components analysis (PCA) and subsequently clusters are identified using discriminant analysis (DA).

In other words, the goal of DAPC is to find a linear combination of features that characterizes or separates two or more classes of objects or events. While PCA focuses on overall variation, DAPC allows us to focus specifically on finding variation between groups.

```
x <- normalized_counts_greater_5 %>%
  t()   # Transpose the matrix

PC_x <- prcomp(x)   # Calculate the PCs

PCs_x <- data.frame(PC_x$x) %>%
  rownames_to_column(var = "sample_id")

head(PCs_x, 10)
```

3.a. Calculate the PCs

```
##      sample_id       PC1         PC2         PC3        PC4         PC5
## 1   A1.6854_S1 -39562.41   -342.5773  -9404.7575 -2073.1319   3305.1680
## 2  A3.6830_S17  21909.17  27432.2706  -8311.4987 -1777.0711 -11904.6700
## 3   B1.6855_S2  71864.29  10319.9164   1552.8794  1712.2488   2081.8945
## 4  B2.6813_S10  18565.00 -13279.3162   6328.9341 -1895.6037  -2819.1610
## 5  B3.6832_S18   7140.44 -14883.7208  -5635.1653  3048.1746    274.9454
## 6   C1.6872_S3 -18569.83   -296.5142  -5994.8365 -1842.0134   3851.0146
## 7  C2.6821_S11 -23194.78 -22919.1079    221.8982  7814.0353  -4806.9118
## 8  C3.6835_S19 -13121.78  -1836.0159 -10054.3150  -231.2435   1617.8117
## 9   D1.6863_S4  81802.01   5185.5556   4677.7633  -193.3651   2021.8303
## 10 D2.6822_S12  69158.21  11080.2149   4600.0748  3547.2010    998.4706
##           PC6        PC7         PC8        PC9       PC10        PC11
## 1   3536.1971 -3685.3603 -1874.67620  262.12854  -771.0563  1593.94782
## 2  -2022.9889 -1031.0081  -111.07274  116.52500  -157.5630   -18.53249
## 3   1823.5021  -426.0612   133.64454  718.78058 -1328.5323   292.11726
## 4    803.8003  2457.5121 -3407.07094 -871.51686 -1417.1613   551.76973
## 5  -1972.6157  3658.5318 -1114.77746 1127.91413 -1335.1018  -569.91620
## 6  -1246.1407 -2199.0722    68.05519 -806.05795  -968.7915  -302.13275
## 7   4920.5982 -1945.0609  1929.57188  -27.43375  -107.8694 -1288.72222
## 8   1156.5282  4388.7478  2178.49084  281.50823  2453.7572   629.57654
## 9  -1267.8385 -1294.2309 -1517.26271 3528.18086  1643.8254 -1434.76485
## 10  1501.0252  -704.6848   502.03412 -704.49876   746.7140  1150.37772
##            PC12        PC13       PC14        PC15        PC16       PC17
## 1   -861.311380   541.70561 -913.06251    85.11898  397.166369 -639.24666
## 2      7.380523   -44.96605   43.91261    14.88528   -6.435194  -21.42287
## 3   -926.788212  -832.44594   29.58555  1378.59841 -460.290638 1190.53482
## 4    936.206540  1553.29916  849.74595   121.73696  564.854821  212.44162
## 5  -2191.008381  -620.60140  695.20608  -445.95992 -383.587327 -479.47799
## 6   1430.572363  -572.47558 1346.25492   558.15052   -9.569817  375.21058
## 7     21.884458   860.42629 -141.12892  -176.08788  -33.115249  294.13914
## 8    420.231837   546.56404  392.03775   921.44236  650.218390  -80.26483
## 9    422.733715   673.50165 -426.62006   163.34380  444.987201 -133.59878
## 10  -273.432763 -1416.43456  802.03862 -1424.12220 1240.828080   29.92035
```

```
##           PC18          PC19
## 1     74.14985  2.332797e-11
## 2     11.53935  3.227832e-12
## 3    680.90310 -2.043285e-12
## 4    335.57861  1.041319e-11
## 5   -398.64053 -1.209406e-11
## 6  -1220.18891  3.211840e-12
## 7   -310.60342  6.223478e-12
## 8    285.34503 -1.921234e-11
## 9   -387.67161  7.029710e-13
## 10    27.69222 -1.132524e-10
```

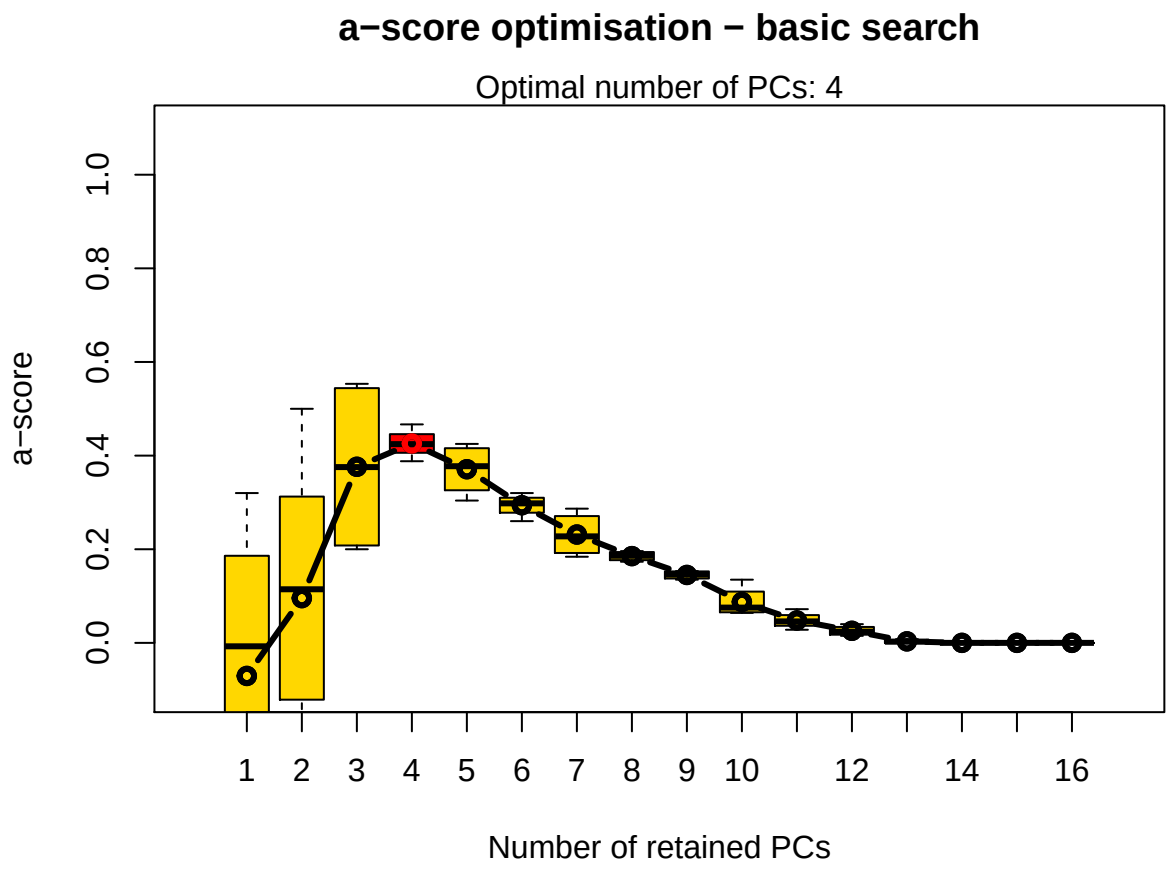
```
# Remove rows containing the two anomalies using the results of the PCA
PCs_x <- PCs_x %>%
  subset(sample_id != c("H2.6848_S16", "A3.6830_S17"))

normalized_counts_greater_5 <- normalized_counts_greater_5 %>%
  dplyr::select(-c("H2.6848_S16", "A3.6830_S17"))

# Add another column which provides the species related to each gene
PCs_x <- PCs_x %>%
  mutate(Species = case_when(
    sample_id == "A1.6854_S1" ~ "E_anthonyi",
    sample_id == "B1.6855_S2"~ "E_anthonyi",
    sample_id == "C1.6872_S3" ~ "E_anthonyi",
    sample_id ==  "D1.6863_S4"~ "E_anthonyi",
    sample_id == "E1.6870_S5"~ "E_anthonyi",
    sample_id == "F1.6803_S6" ~ "E_boulengeri",
    sample_id == "G1.6806_S7" ~ "E_boulengeri",
    sample_id == "H1.6807_S8"~ "E_boulengeri",
    sample_id == "B2.6813_S10" ~ "E_boulengeri",
    sample_id ==  "C2.6821_S11"~ "E_machalilla",
    sample_id == "D2.6822_S12"~ "E_machalilla",
    sample_id == "E2.6826_S13" ~ "E_machalilla",
    sample_id == "F2.6845_S14" ~ "E_machalilla",
    sample_id == "G2.6847_S15_1" ~ "E_machalilla",
    sample_id == "H2.6848_S16" ~ "E_machalilla",
    TRUE ~ "E_tricolor"))   # The remaining ones are E_tricolor

# Add another column which provides the skin coloration related to each species
PCs_x <- PCs_x %>%
  mutate(Skin_Coloration = case_when(
    Species == "E_anthonyi" ~ "Aposematic",
    Species == "E_tricolor" ~ "Aposematic",
    Species == "E_machalilla" ~ "Cryptic",
    Species == "E_boulengeri" ~ "Cryptic"))

# Add another column which provides the localities at which the Epipedobates were collected
PCs_x <- PCs_x %>%
  mutate(Locality = case_when(
```

```
    sample_id == "A1.6854_S1" ~ "Pasaje", sample_id == "B1.6855_S2" ~ "Pasaje",
    sample_id == "C1.6872_S3" ~ "Uzchurummi", sample_id ==  "D1.6863_S4" ~ "Moromoro",
    sample_id == "E1.6870_S5" ~ "Uzchurummi", sample_id == "F1.6803_S6" ~ "LaPerla",
    sample_id == "G1.6806_S7" ~ "LaPerla", sample_id == "H1.6807_S8" ~ "LaPerla",
    sample_id == "B2.6813_S10" ~ "Bilsa", sample_id ==  "C2.6821_S11"~ "5 de agosto",
    sample_id == "D2.6822_S12" ~ "5 de agosto", sample_id == "E2.6826_S13" ~ "5 de agosto",
    sample_id == "F2.6845_S14" ~ "Jouneche", sample_id == "G2.6847_S15_1" ~ "Jouneche",
    sample_id == "H2.6848_S16" ~ "Jouneche", sample_id == "A3.6830_S17" ~ "Guanujo",
    sample_id == "C3.6835_S19" ~ "Guanujo", sample_id == "B3.6832_S18" ~ "Guanujo",
    sample_id == "D3.6843_S20" ~ "ChazoJuan"))

# Add another column which provides the sex of the Epipedobates
PCs_x <- PCs_x %>%
  mutate(Sex = case_when(
    sample_id == "A1.6854_S1" ~ "F", sample_id == "B1.6855_S2"~ "M",
    sample_id == "C1.6872_S3" ~ "F", sample_id ==  "D1.6863_S4"~ "M",
    sample_id == "E1.6870_S5"~ "F", sample_id == "F1.6803_S6" ~ "F",
    sample_id == "G1.6806_S7" ~ "F", sample_id == "H1.6807_S8"~ "F",
    sample_id == "B2.6813_S10" ~ "F", sample_id ==  "C2.6821_S11"~ "F",
    sample_id == "D2.6822_S12"~ "M", sample_id == "E2.6826_S13" ~ "M",
    sample_id == "F2.6845_S14" ~ "F",sample_id == "G2.6847_S15_1" ~ "F",
    sample_id == "H2.6848_S16" ~ "F",sample_id == "A3.6830_S17" ~ "F",
    sample_id == "C3.6835_S19" ~ "F",sample_id == "B3.6832_S18" ~ "F",
    sample_id == "D3.6843_S20" ~ "M"))

# Make first column as row names
PCs_sample <- PCs_x[,-1]
rownames(PCs_sample) <- PCs_x[,1]
```

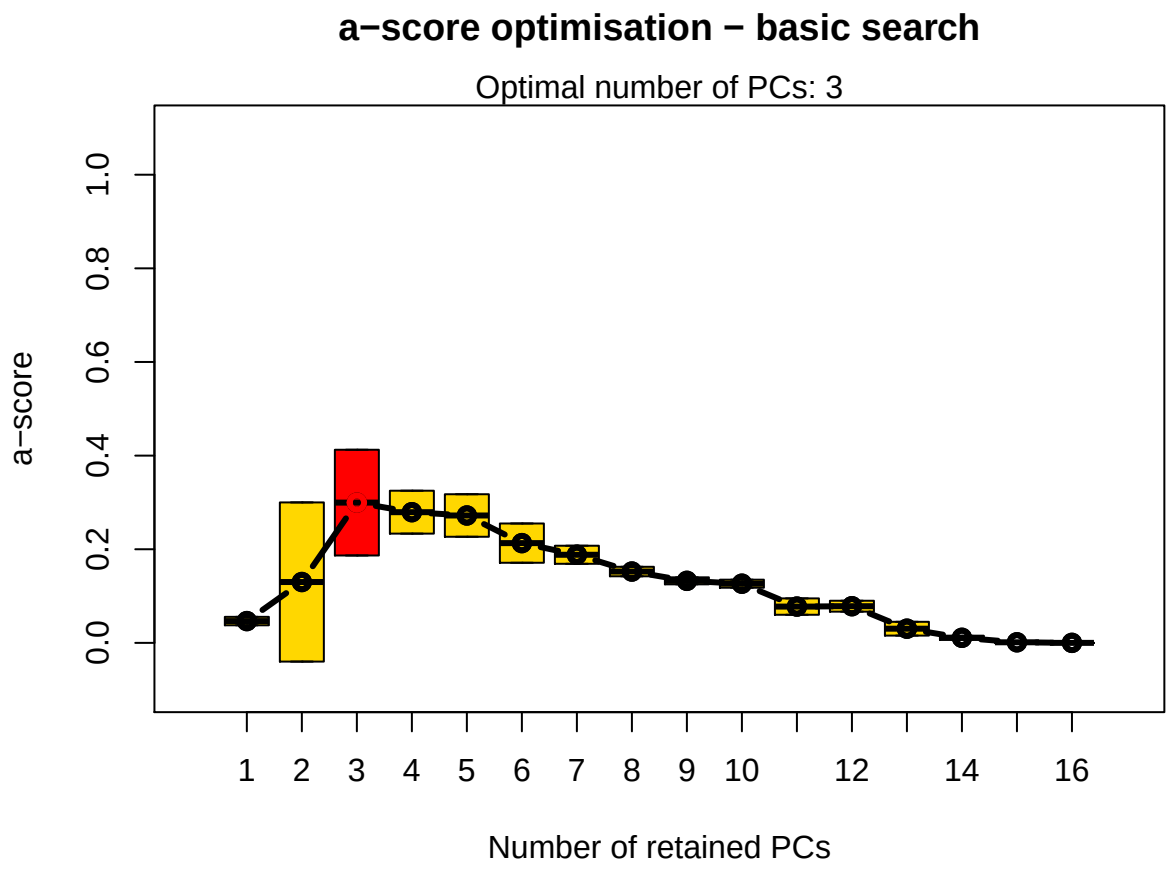
3.b. Data transformation

3.c. Calculate a-score To retain the optimal number of PCs, we'll run the a-score optimization (*Note*: The number of discriminant analysis `n.da` = n -1, where n is the number of different groups (e.g., species (n = 4), sex (n = 2)))

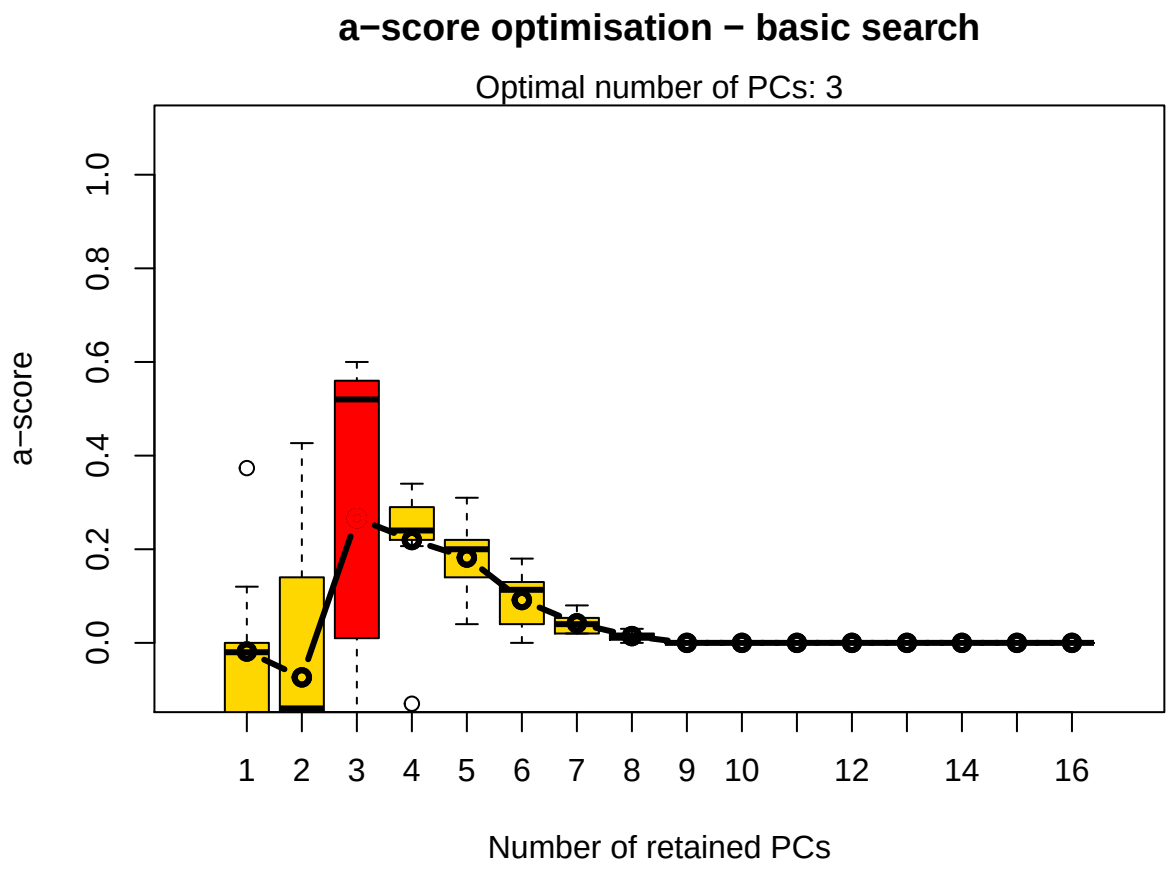
```
set.seed(100) # Since DAPC analysis use randomized groups to calculate a.score, we'll set seed

# Calculate optimal number of PCs for species
dapcTemp_species <- dapc(t(normalized_counts_greater_5), PCs_sample$Species,
                         perc.pca = 100, n.da = 3)    # n.da = 4(species) - 1 = 3
ascore_species <- optim.a.score(dapcTemp_species, smart = FALSE, n.sim = 50)    # Optimal number of PCs:
```



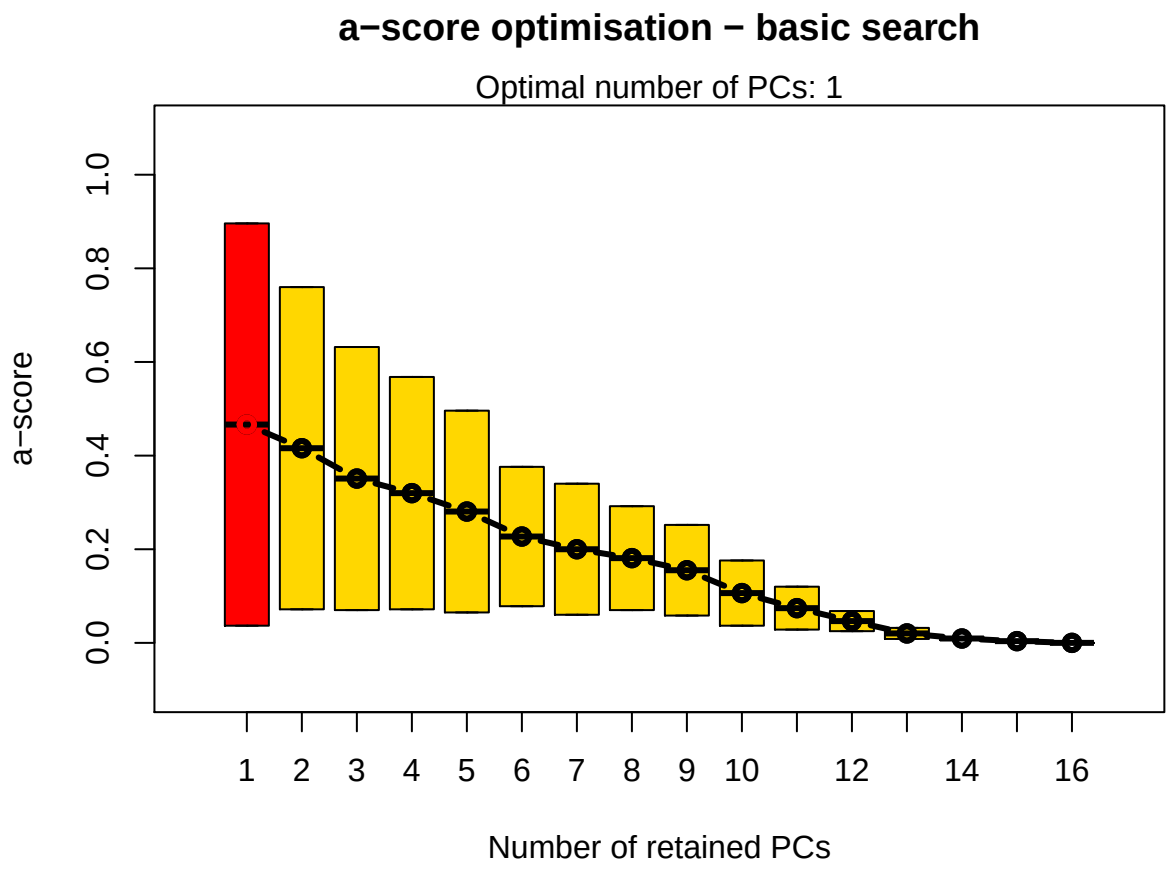

```
# Calculate optimal number of PCs for skin coloration
dapcTemp_skin_coloration <- dapc(t(normalized_counts_greater_5), PCs_sample$Skin_Coloration,
                                 perc.pca = 100, n.da = 1)    # n.da = 2(skin coloration) - 1 = 1
ascore_skin_coloration <- optim.a.score(dapcTemp_skin_coloration, smart = FALSE, n.sim = 50)    # Optima
```




```r
# Calculate optimal number of PCs for locality
dapcTemp_skin_locality <- dapc(t(normalized_counts_greater_5), PCs_sample$Locality,
                               perc.pca = 100, n.da = 18)   # n.da = 19 (localities) - 1 = 18
ascore_locality <- optim.a.score(dapcTemp_skin_locality, smart = FALSE, n.sim = 50)   # Optimal number
```

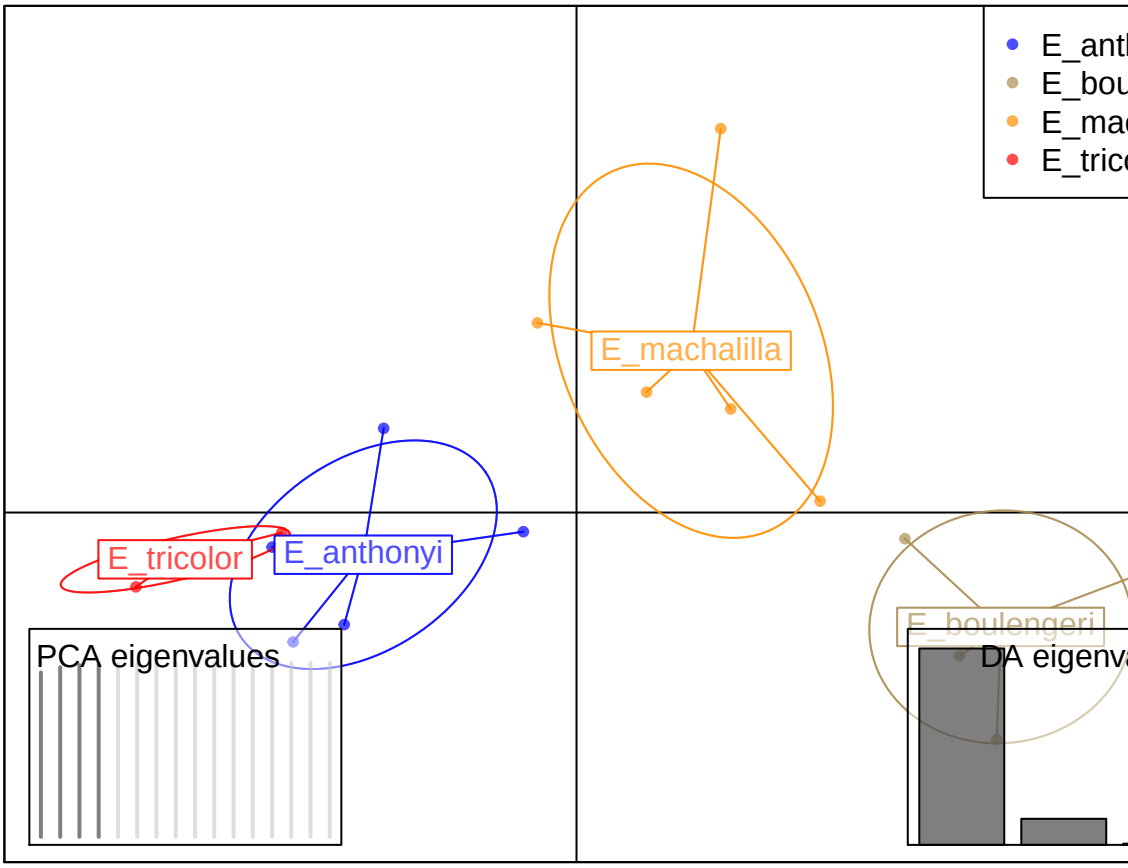



```r
# Calculate optimal number of PCs for sex
dapcTemp_sex <- dapc(t(normalized_counts_greater_5), PCs_sample$Sex,
                     perc.pca = 100, n.da = 1)    # n.da = 2(sex) - 1 = 1
ascore_sex <- optim.a.score(dapcTemp_sex, smart = FALSE, n.sim = 50)    # Optimal number of PCs: 1
```

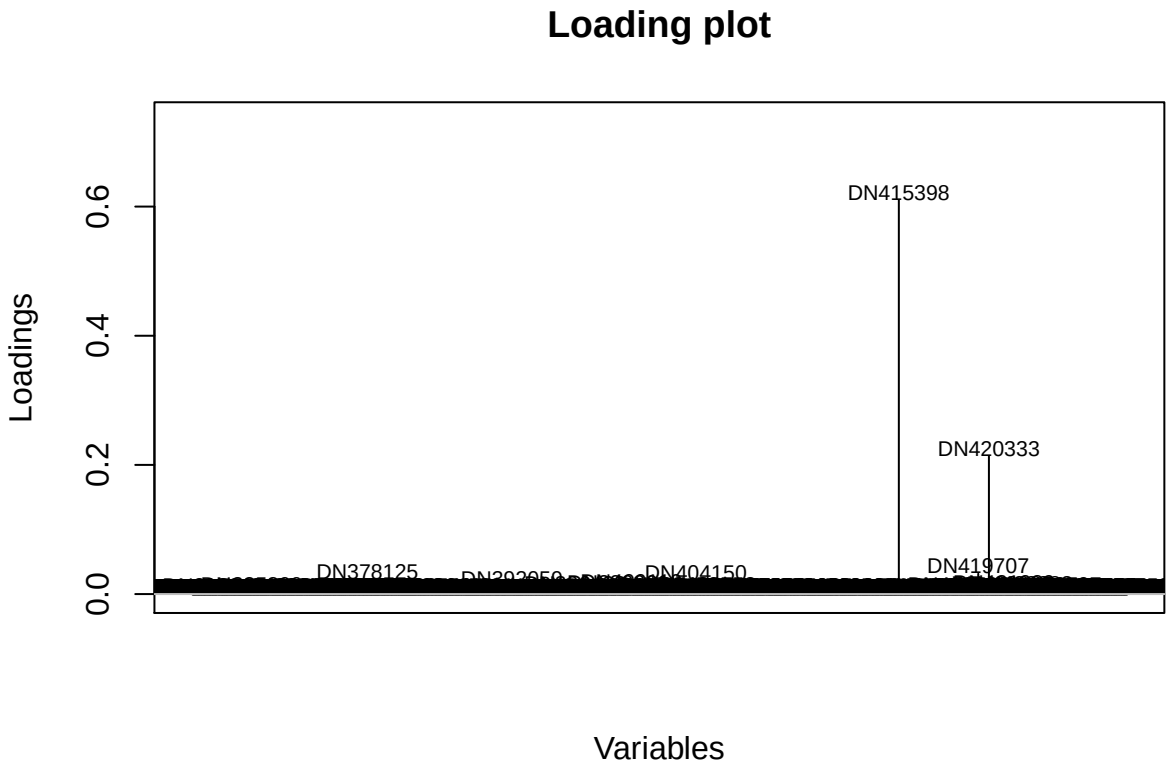


4. Visualization - DAPC

```
{ # Wrap output (e.g., plots and text) in one chunk using {}
  set.seed(100)

  dapc_species <- dapc(t(normalized_counts_greater_5), PCs_sample$Species,
                       n.pca = 4, n.da = 3)
  scatter.dapc(dapc_species, scree.pca = TRUE, scree.da = TRUE, legend = TRUE)
  loadingplot(dapc_species$var.contr)  # var.contr: A data.frame giving the contributions of
                                       # original variables to the principal components of DAPC
  print(paste("The proportion of conserved variance is", dapc_species$var))
}
```



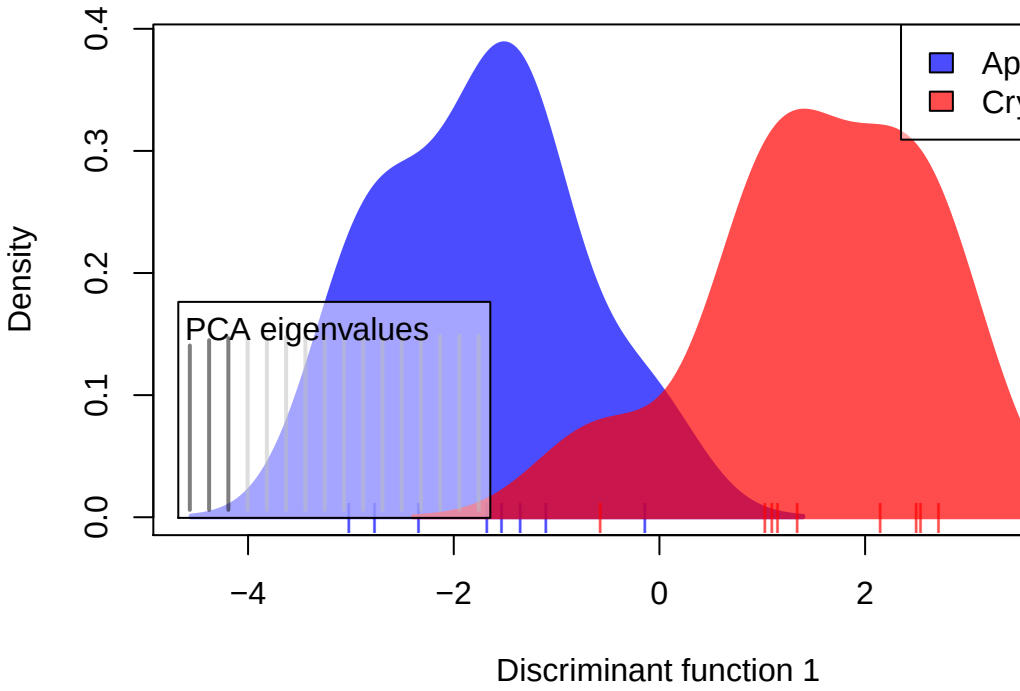
4.a. DAPC by species



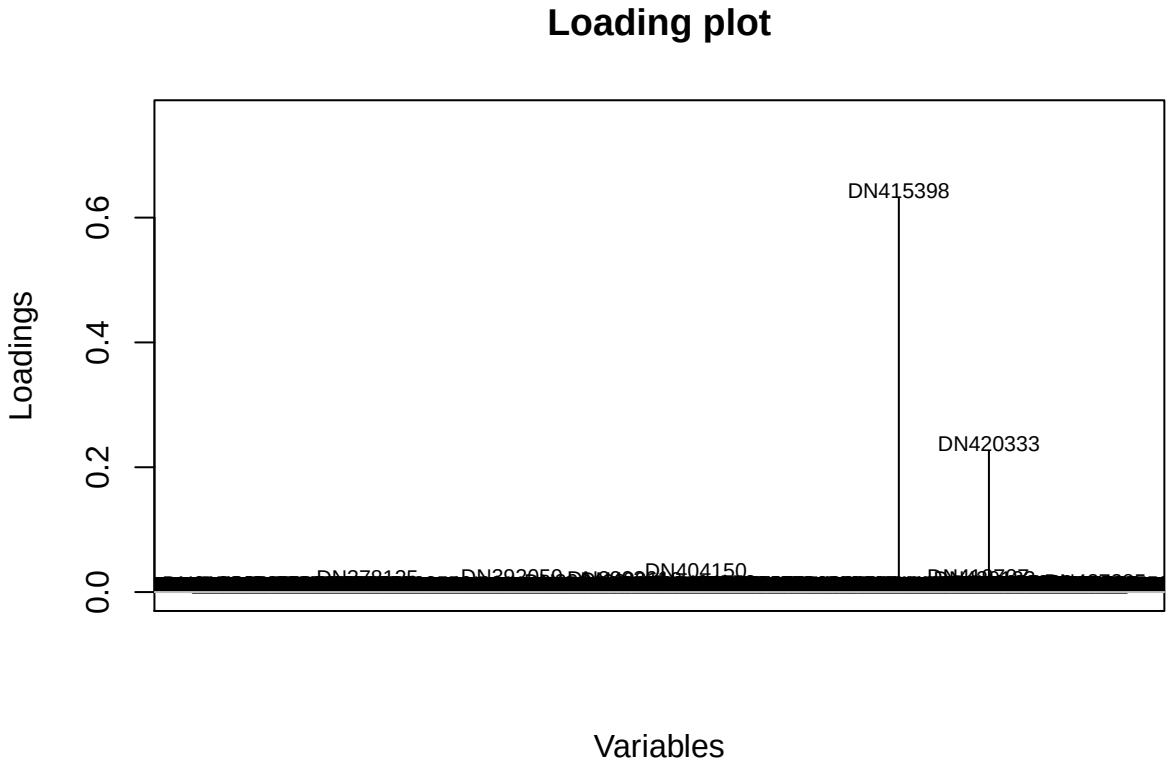
```
## [1] "The proportion of conserved variance is 0.991437635964515"
```

We can see that the DAPC clearly separates the four species and up to 99% of the variance is conserved, performing a much better job than the PCA. If we looked at the loading plot, we can see that the two genes DN415398 and DN420333 are the main genes that drive the separation. However, the gene DN415398, which has the mapping P4HB_MAN1A1_HKDC1_HK3_HK2_HK1, are found in every category in our GO analysis, suggesting that it's a common gene. On the other hand, the gene DN420333, which has the mapping SERPINA12_SERPINA10, is not identified in our DGE or GO analysis.

```
{
  set.seed(100)

  dapc_skin_coloration <- dapc(t(normalized_counts_greater_5), PCs_sample$Skin_Coloration,
                               n.pca = 3, n.da = 1)
  scatter.dapc(dapc_skin_coloration, scree.pca = TRUE, scree.da = TRUE, legend = TRUE)
  loadingplot(dapc_skin_coloration$var.contr)
  print(paste("The proportion of conserved variance is", dapc_skin_coloration$var))
}
```



4.b. DAPC by skin coloration

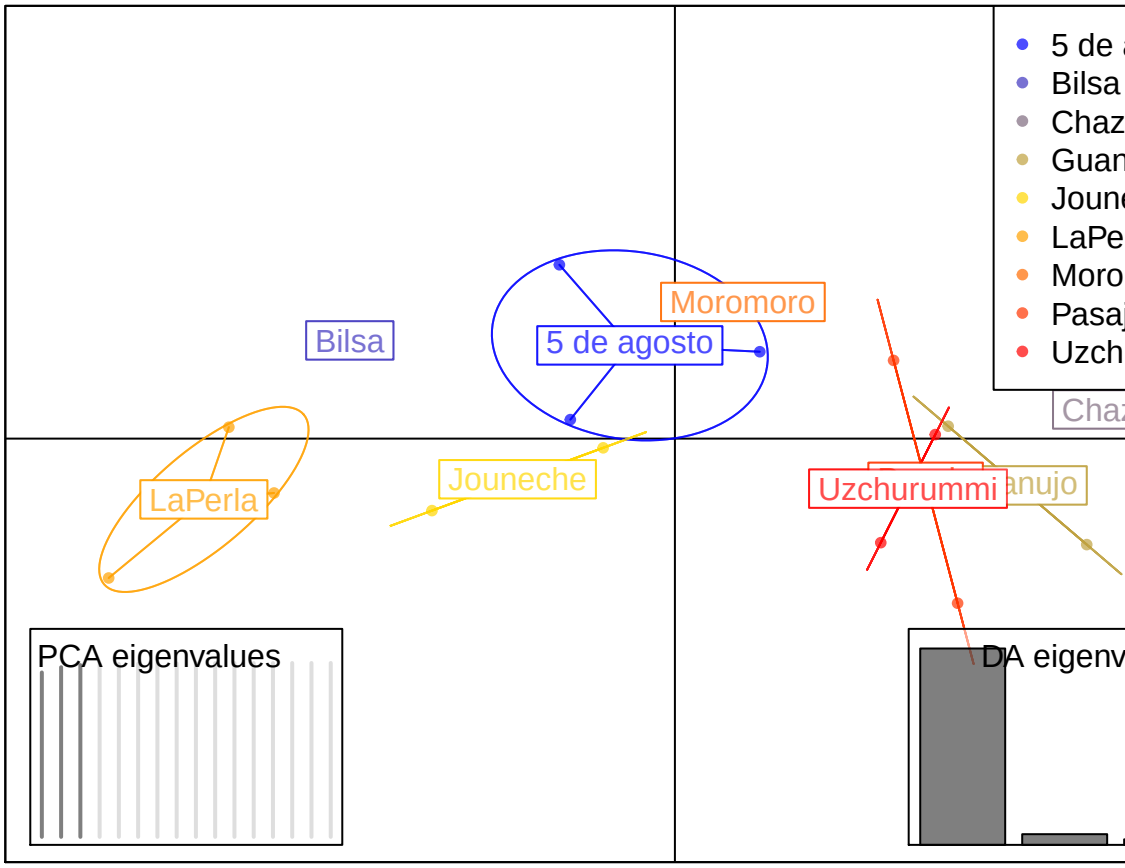


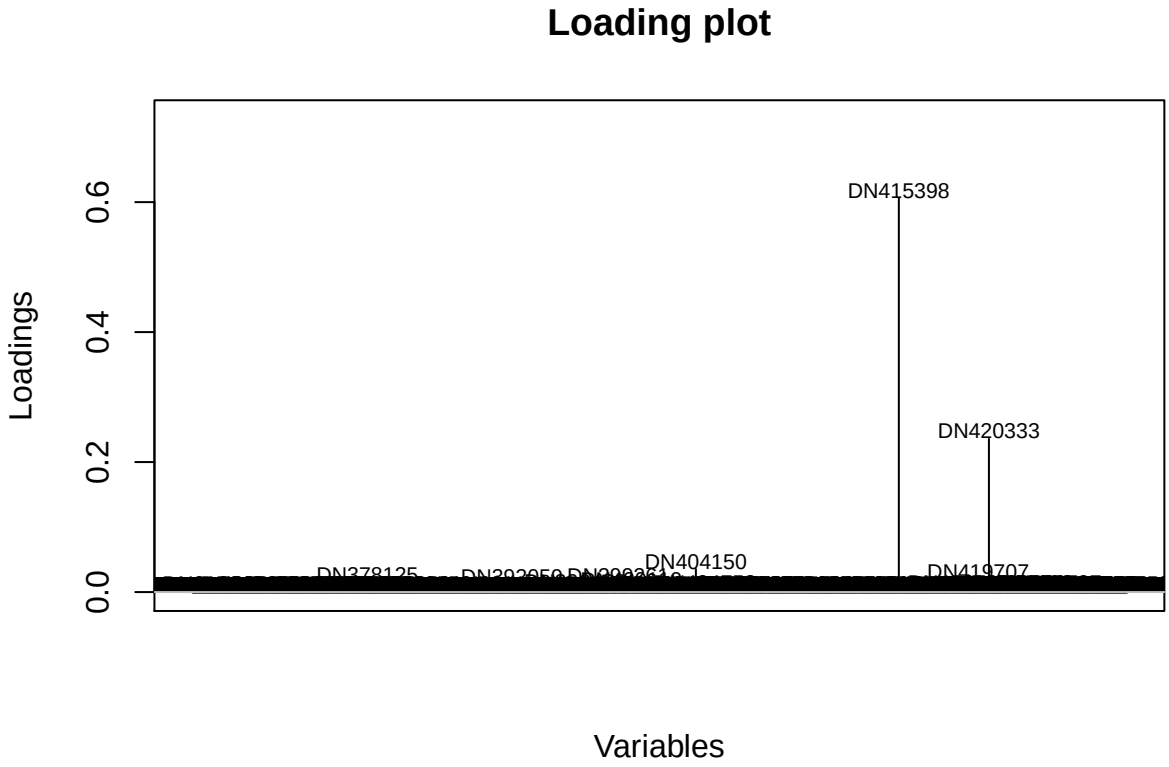
```
## [1] "The proportion of conserved variance is 0.987767357365527"
```

Again, we can see that the DAPC clearly separates the two skin coloration and up to 98% of the variance is conserved. If we looked at the loading plot, we can see that the two genes DN415398 and DN420333 are the main genes that drive the separation. However, we found the same trend where the two genes DN415398 and DN420333 are the main genes that drive the separation again.

```
{
  set.seed(100)

  dapc_locality <- dapc(t(normalized_counts_greater_5), PCs_sample$Locality,
                        n.pca = 3, n.da = 18)
  scatter.dapc(dapc_locality, scree.pca = TRUE, scree.da = TRUE, legend = TRUE)
  loadingplot(dapc_locality$var.contr)
  print(paste("The proportion of conserved variance is", dapc_locality$var))
}
```

4.c. DAPC by locality

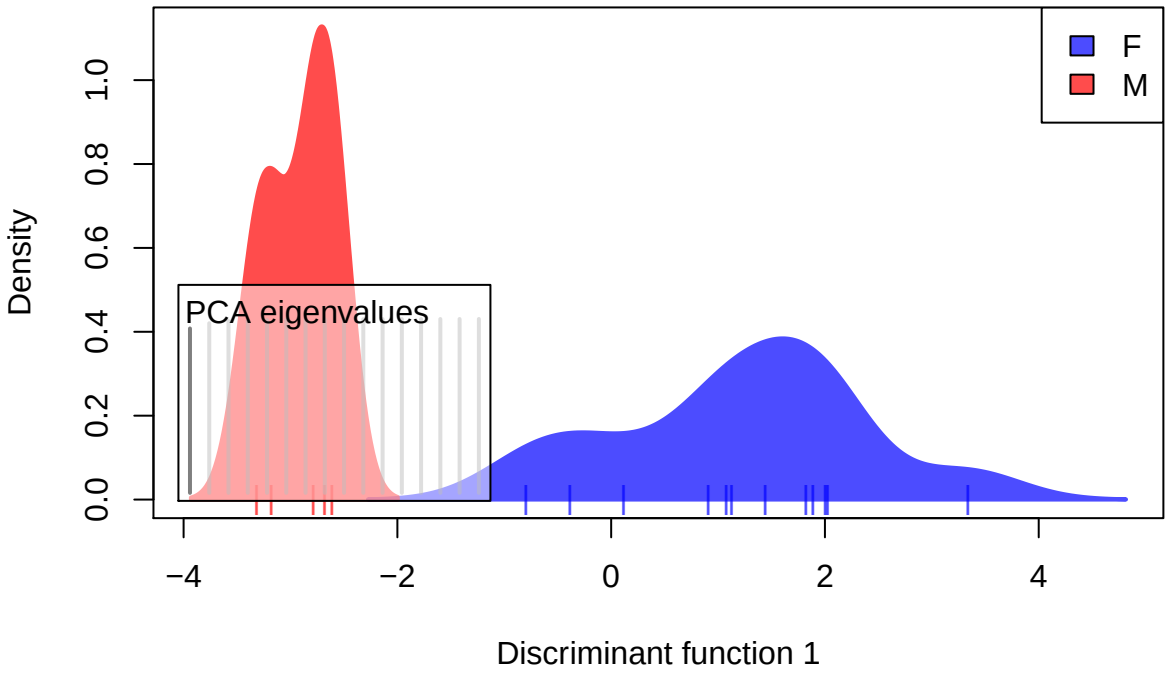




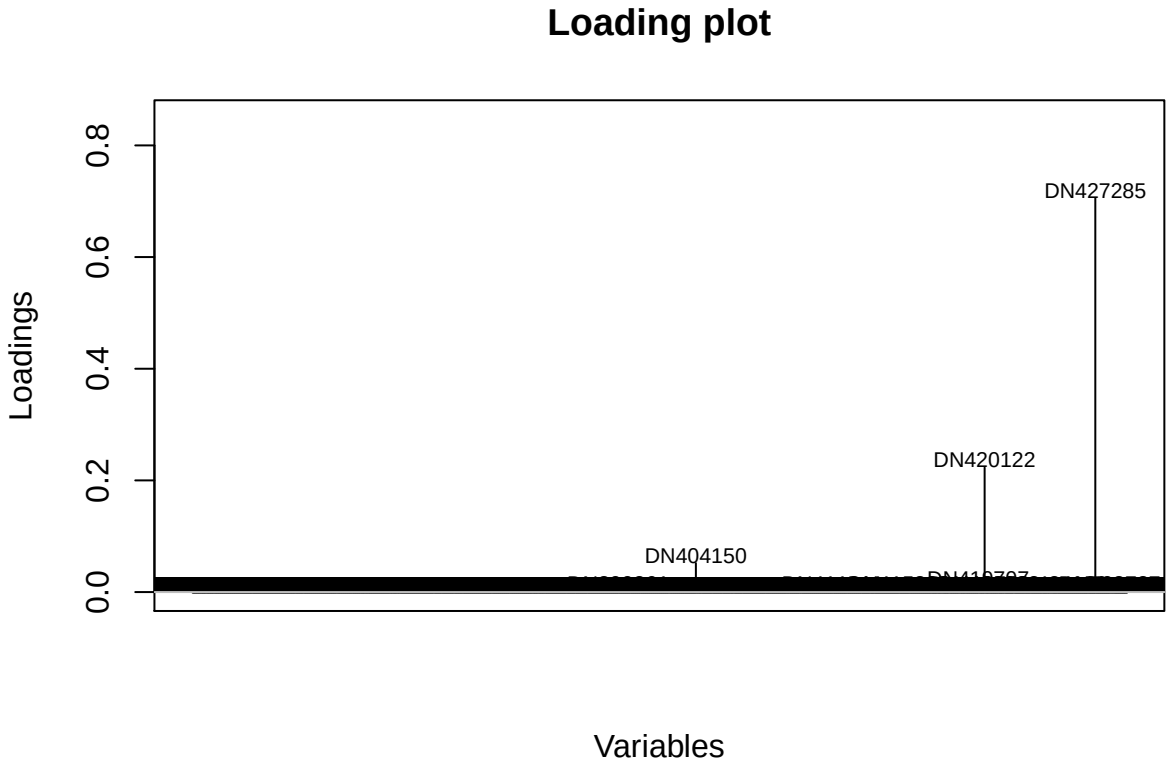
```
## [1] "The proportion of conserved variance is 0.987767357365527"
```

Same pattern as above

```
{
  set.seed(100)

  dapc_sex <- dapc(t(normalized_counts_greater_5), PCs_sample$Sex,
                   n.pca = 1, n.da = 1)
  scatter.dapc(dapc_sex, scree.pca = TRUE, scree.da = TRUE, legend = TRUE)
  loadingplot(dapc_sex$var.contr)
  print(paste("The proportion of conserved variance is", dapc_sex$var))
}
```



4.d. DAPC by sex



```
## [1] "The proportion of conserved variance is 0.945380078045848"
```

This time, the separation between male and female is clear and the proportion of conserved variance remains high (although slightly lower than previous ones), the genes responsible for driving the separation are different - DN427285, DN420122, and DN404150. However, DN 427285 doesn't have a mapping, while both DN420122 (mapping: DROSHA) and DN404150 (mapping: OMP_KIAA0226) are not identified in our DGE or GO analysis

Overall, the genes that primarily drive the difference between the 4 groupings (species, skin coloration, locality, and sex) are either common genes, not found in our DGE or GO analysis (conflicting result), or doesn't have a mapping. Therefore, it's not possible to pinpoint any specific gene(s) that results in the separation between Cryptic and Aposematic before further investigations are carried out.