Morphometric identification of an unknown honey bee colony: an example from north India

Supplementary document with statistical analysis

Hardeep Kaur Sajad Ahmad Ganie Adam Tofilski

2024-11-11

Libraries

```
# calculations
library(geomorph) # GPA
library(shapes) # OPA
library(MASS) # LDA
library(IdentiFlyR) # identification data
# plotting and visualization
library(ggplot2) # plots
ggplot2::theme_set(theme_light())
library(ggforce) # geom_ellipse
library(ggrepel) # geom_label_repel
library(rnaturalearth) # maps
library(raster) # raster
library(ggspatial) # annotation_scale
# input and output
library(readxl) # read XLSX
library(officer) # read DOCX
library(xml2) # read XML
library(XML) # write XML
```

Variable names

In order to avoid mistakes it is safer to use defined variables instead of numbers, for example, columns names instead of indexes

```r
p = 19 # number of landmarks
k = 2  # number of dimensions, in this case 2 for coordinates (x, y)

# create coordinates names used by IdentiFly
xyNames = c("x1", "y1")
for (i in 2:p) {
  xyNames = c(xyNames, paste0("x", i))
  xyNames = c(xyNames, paste0("y", i))
}
xyNames
```

```
 [1] "x1"  "y1"  "x2"  "y2"  "x3"  "y3"  "x4"  "y4"  "x5"  "y5"  "x6"  "y6"
[13] "x7"  "y7"  "x8"  "y8"  "x9"  "y9"  "x10" "y10" "x11" "y11" "x12" "y12"
[25] "x13" "y13" "x14" "y14" "x15" "y15" "x16" "y16" "x17" "y17" "x18" "y18"
[37] "x19" "y19"
```

```r
# define which landmarks are connected by lines in wireframe graph
link.x <- c(1, 1, 2, 2, 3,  3, 4,  4,  5, 6, 7,  7,  7, 8,  9,  9, 10, 11, 11,
            12, 13, 14, 15, 16, 17)
link.y <- c(2, 3, 4, 5, 6, 19, 6, 10, 12, 8, 8, 14, 19, 9, 10, 15, 11, 12, 16,
            13, 18, 15, 16, 17, 18)
links.apis <- cbind(link.x, link.y)

# define Bustamente links
link.x <- c(1, 1, 2, 3, 3, 4,  4,  5, 6, 7,  7, 8,  9,  9, 10, 11, 11, 12, 13,
            13, 14, 15, 15, 16, 17)
link.y <- c(2, 6, 3, 4, 8, 5, 10, 12, 7, 8, 16, 9, 10, 14, 11, 12, 13, 19, 14,
            15, 17, 18, 19, 17, 18)
links.Bustamente <- cbind(link.x, link.y)
```

Functions

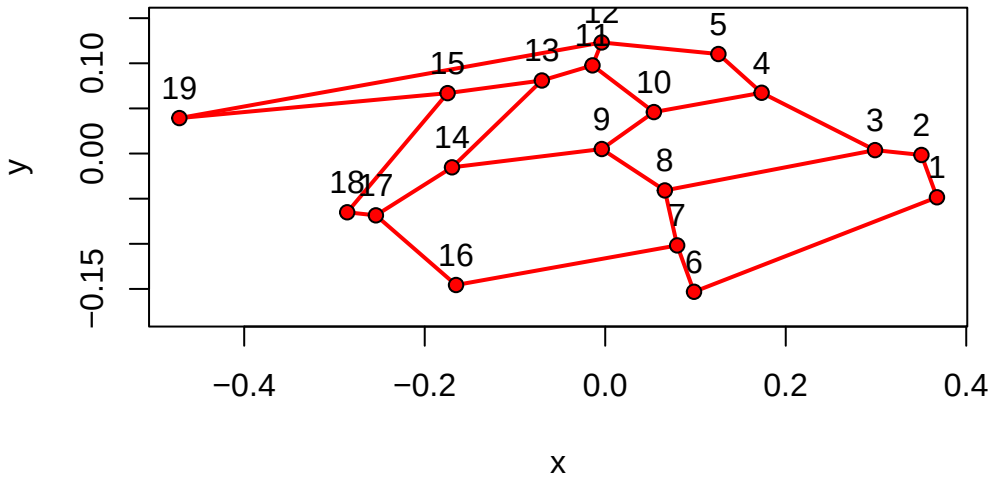
```r
# convert vector to XML
vector2XML = function(vec.in, id, XML.out){
  vec.in = as.character(vec.in)
  vec.in = paste(vec.in, collapse = "\t")
  vec.in = c("\t", vec.in, "\t") # pad with tabulators for spreadsheet
  XML.out$addTag("vector", vec.in, attrs = c(id=id))
}

# convert matrix to XML
matrix2XML = function(mat.in, id, XML.out){
```

```
  XML.out$addTag("matrix", close = FALSE, attrs = c(id=id))
  header = as.character(colnames(mat.in))
  header = paste(header, collapse = "\t")
  header = c("\t", header, "\t")
  XML.out$addTag("header", header)

  for (i in 1:nrow(mat.in)) {
    vector2XML(mat.in[i,], rownames(mat.in)[i], XML.out)
  }
  XML.out$closeTag() # matrix
}
```

Apis species classification data

The classification is based on tabular data obtained from Bustamante et al. (2021)

```
# download ESM1 from Bustamente et al. 2021 and save as Bustamente2021-ESM1.docx
download.file("https://static-content.springer.com/esm/art%3A10.1007%2Fs13592-021-00857-7/
              destfile = "Bustamente2021-ESM1.docx", method="curl")

# download ESM2 from Bustamente et al. 2021 and save as Bustamente2021-ESM2.xlsx
download.file("https://static-content.springer.com/esm/art%3A10.1007%2Fs13592-021-00857-7/
              destfile = "Bustamente2021-ESM2.xlsx", method="curl")

# download ESM3 from Bustamente et al. 2021 and save as Bustamente2021-ESM3.xlsx
download.file("https://static-content.springer.com/esm/art%3A10.1007%2Fs13592-021-00857-7/
              destfile = "Bustamente2021-ESM3.xlsx", method="curl")

# extract reference configuration from sheet "Aligned Sample" in ESM3
Bustamente2021.reference = read_excel("Bustamente2021-ESM3.xlsx",
                                      sheet = "Aligned Sample",
                                      range = "A1:AL2")
Bustamente2021.reference = as.data.frame(Bustamente2021.reference)
colnames(Bustamente2021.reference) = xyNames
rownames(Bustamente2021.reference) = c("reference")

reference.3D <- arrayspecs(Bustamente2021.reference, p, k)
# plot landmarks from Bustamante et al. (2021)
plotAllSpecimens(reference.3D,links=links.Bustamente,label=TRUE,
                 plot.param = list(pt.bg = "black", pt.cex = 0.5,
                                   mean.bg = "red", mean.cex=1, link.col="red",
                                   txt.pos=3, txt.cex=1))
```



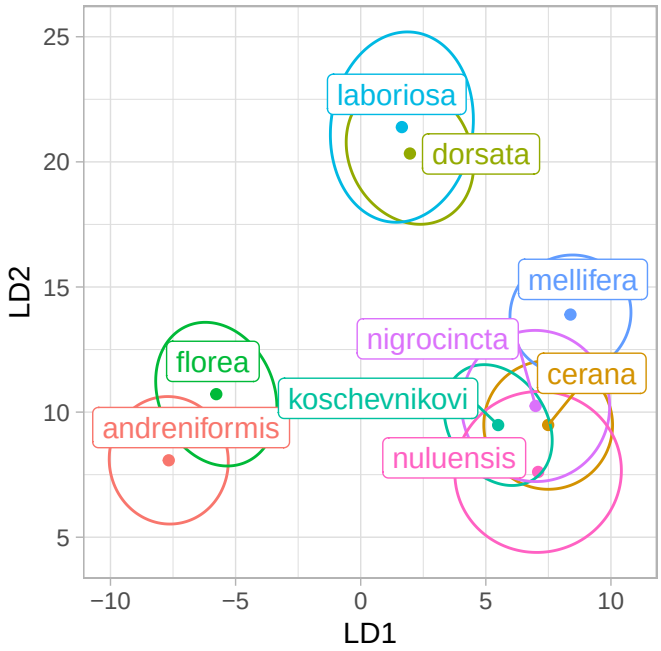

```
# extract coefficients from sheet "LD's" in ESM3
Bustamente2021.coefficients = read_excel("Bustamente2021-ESM3.xlsx",
                                         sheet = "LD's", range = "A1:I39")
Bustamente2021.coefficients = as.data.frame(Bustamente2021.coefficients)
Bustamente2021.coefficients = Bustamente2021.coefficients[,-c(1)]
Bustamente2021.coefficients = t(Bustamente2021.coefficients)
colnames(Bustamente2021.coefficients) = xyNames

# extract class means from ESM1; the table is in 4 parts
ESM1 <- read_docx("Bustamente2021-ESM1.docx")
content <- docx_summary(ESM1)

fragment = content[content$doc_index == 5, ]
fragment = data.frame(text = fragment$text, row_id = fragment$row_id,
                      cell_id = fragment$cell_id)
tab5 = reshape(fragment, idvar = "row_id", timevar = "cell_id", direction = "wide")
tab5 = tab5[-c(1:2), -c(1)]

fragment = content[content$doc_index == 7, ]
fragment = data.frame(text = fragment$text, row_id = fragment$row_id,
                      cell_id = fragment$cell_id)
tab7 = reshape(fragment, idvar = "row_id", timevar = "cell_id", direction = "wide")
tab7 = tab7[-c(1), -c(1)]

fragment = content[content$doc_index == 10, ]
fragment = data.frame(text = fragment$text, row_id = fragment$row_id,
```

```
                      cell_id = fragment$cell_id)
tab10 = reshape(fragment, idvar = "row_id", timevar = "cell_id", direction = "wide")
tab10 = tab10[-c(1), -c(1)]

fragment = content[content$doc_index == 13, ]
fragment = data.frame(text = fragment$text, row_id = fragment$row_id,
                      cell_id = fragment$cell_id)
tab13 = reshape(fragment, idvar = "row_id", timevar = "cell_id", direction = "wide")
tab13 = tab13[-c(1), -c(1)]

rownames(tab5) = tab5[,1]
tab5 = tab5[,-c(1)]
Bustamente2021.means = cbind(tab5, tab7[,2:11], tab10[,2:11], tab13[,2:9])
colnames(Bustamente2021.means) = xyNames
Bustamente2021.means[xyNames] = sapply(Bustamente2021.means[xyNames], as.numeric)

# extract covariances matrices from ESM2; each matrix is in separate sheet
groups = rownames(Bustamente2021.means)
# create list of empty dataframes
Bustamente2021.covariances = c(rep(data.frame(), length(groups)))
for (i in 1:length(groups)) {
  sheet.name = paste0(groups[i], " ginv")
  Bustamente2021.covariances[[i]] = read_excel("Bustamente2021-ESM2.xlsx",
                                               sheet = sheet.name, range = "B1:I9")
  names(Bustamente2021.covariances)[i] <- groups[i]
}

# Create XML file
XML = xmlOutputDOM("identifly", attrs=c(version="1.8"))
XML$addTag("prototype", close=TRUE,
           attrs = c(file="Bustamente2021-prototype.dw.png"))

XML$addTag("lda", close=FALSE)

# reorder landmarks
Bustamente2IdentiFly = c(18, 17, 15, 14, 16, 13, 12, 11, 10, 9, 8, 7, 6, 5,
                         4, 3, 2, 1, 19)
reference = reorder2D(Bustamente2021.reference, Bustamente2IdentiFly)
matrix2XML(reference, "reference", XML)

means = reorder2D(Bustamente2021.means, Bustamente2IdentiFly)
matrix2XML(means, "means", XML)

XML$addTag("covariances", close=FALSE)
```

```
for (i in 1:length(Bustamente2021.covariances)) {
  matrix2XML(Bustamente2021.covariances[[i]],
             names(Bustamente2021.covariances)[i], XML)
}
XML$closeTag() # covariances

coefficients = reorder2D(Bustamente2021.coefficients, Bustamente2IdentiFly)
matrix2XML(coefficients, "coefficients", XML)

XML$closeTag() # lda

# add prototype for IdentiFly software
XML$addTag("prototype", close=TRUE,
           attrs = c(file="apis-worker-prototype.dw.png"))

saveXML(XML$value(), file="apis-species.dw.xml",
        prefix = "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n")
```

```
[1] "apis-species.dw.xml"
```

```
# Verify the XML file
id.data = xml2gmLdaData("apis-species.dw.xml")
means.LD = id.data$means %*% t(id.data$coefficients)
covEllipses(means.LD, id.data$covariances)
```



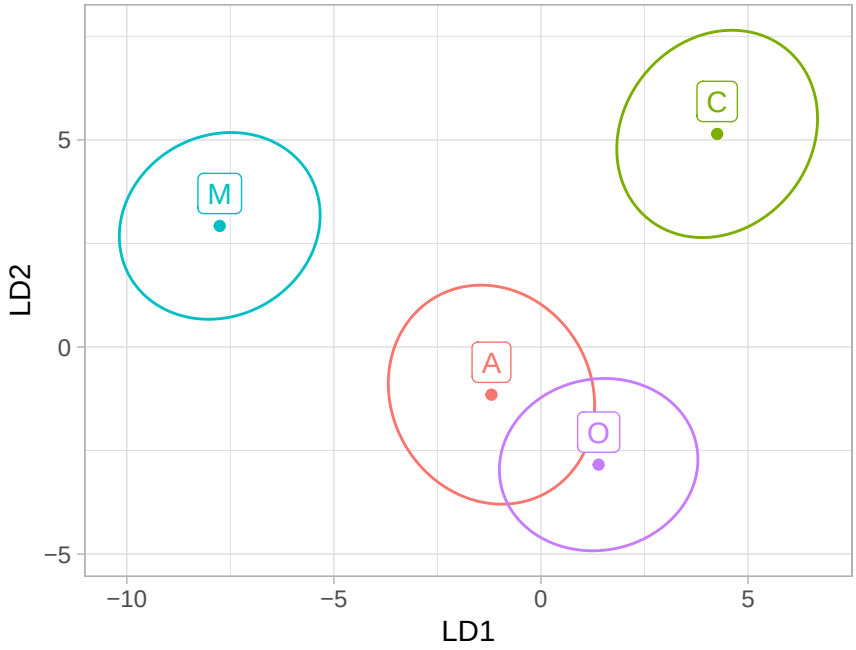
Honey bee lineage classification data

The classification is based on raw data obtained from Nawrocka et al. (2018a) and Nawrocka et al. (2018b)

```
data(lineages)
grVec = lineages$lineage
coordinates = lineages[,-1] # remove the first column
gmLdaData2xml(coordinates, grVec, "apis-mellifera-lineage.dw.xml")
```

```
[1] "apis-mellifera-lineage.dw.xml"
```

```
# Verify the XML file
id.data = xml2gmLdaData("apis-mellifera-lineage.dw.xml")
means.LD = id.data$means %*% t(id.data$coefficients)
covEllipses(means.LD, id.data$covariances)
```



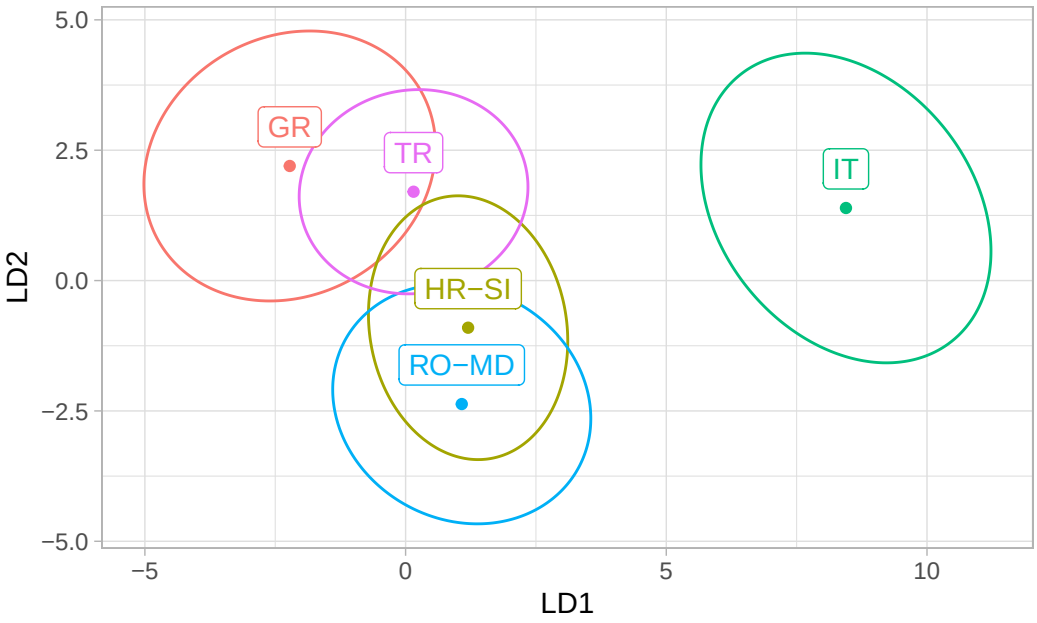
Region classification data

The classification is based on raw data obtained from Oleksa et al. (2022), Oleksa et al. (2023) and Nawrocka et al. (2018b)

```r
wings.C = read.csv("https://zenodo.org/record/7244070/files/GR-raw-coordinates.csv")
wings.C = rbind(wings.C, read.csv("https://zenodo.org/record/7244070/files/HR-raw-coordina
wings.C = rbind(wings.C, read.csv("https://zenodo.org/record/7244070/files/MD-raw-coordina
wings.C = rbind(wings.C, read.csv("https://zenodo.org/record/7244070/files/RO-raw-coordina
wings.C = rbind(wings.C, read.csv("https://zenodo.org/record/7244070/files/SI-raw-coordina
wings.C = rbind(wings.C, read.csv("https://zenodo.org/record/7244070/files/TR-raw-coordina

# extract classifier from file name
wings.C$sample = substr(wings.C$file, 1, 7)

# Convert 2D array into a 3D array
wings.C.3D <- arrayspecs(wings.C[xyNames], p, k)
dimnames(wings.C.3D)[[3]] <- wings.C$file
# Align the coordinates using Generalized Procrustes Analysis
GPA.lin <- gpagen(wings.C.3D, print.progress = FALSE)
# Convert 3D array into a 2D array - opposite to arrayspecs
wings.C.aligned <- two.d.array(GPA.lin$coords)
colnames(wings.C.aligned) = xyNames

# average data within samples
wings.C <- aggregate(wings.C[xyNames],
                     by = list(wings.C$sample),
                     FUN = mean)
rownames(wings.C) <- wings.C$Group.1
wings.C <- subset(wings.C, select = -c(Group.1))

# add Italy
subspecies = substr(rownames(lineages), 1, 5)
IT = subset(lineages, subspecies == "C-lig")
IT <- subset(IT, select = -c(lineage))
rownames(IT) <- gsub("C-lig", "IT", rownames(IT))
wings.C = rbind(wings.C, IT)

# extract region classifier
wings.C$country = substr(rownames(wings.C), 1, 2)
wings.C$region = ifelse(wings.C$country == "HR"
                        | wings.C$country == "SI"
                        , "HR-SI", wings.C$country)
wings.C$region = ifelse(wings.C$country == "RO"
                        | wings.C$country == "MD"
                        , "RO-MD", wings.C$region)

gmLdaData2xml(wings.C[xyNames], wings.C$region, "apis-mellifera-lineage_c_region.dw.xml")
```

```
[1] "apis-mellifera-lineage_c_region.dw.xml"
```

```
# Verify the XML file
id.data = xml2gmLdaData("apis-mellifera-lineage_c_region.dw.xml")
means.LD = id.data$means %*% t(id.data$coefficients)
covEllipses(means.LD, id.data$covariances)
```



Landmark coordinates from India

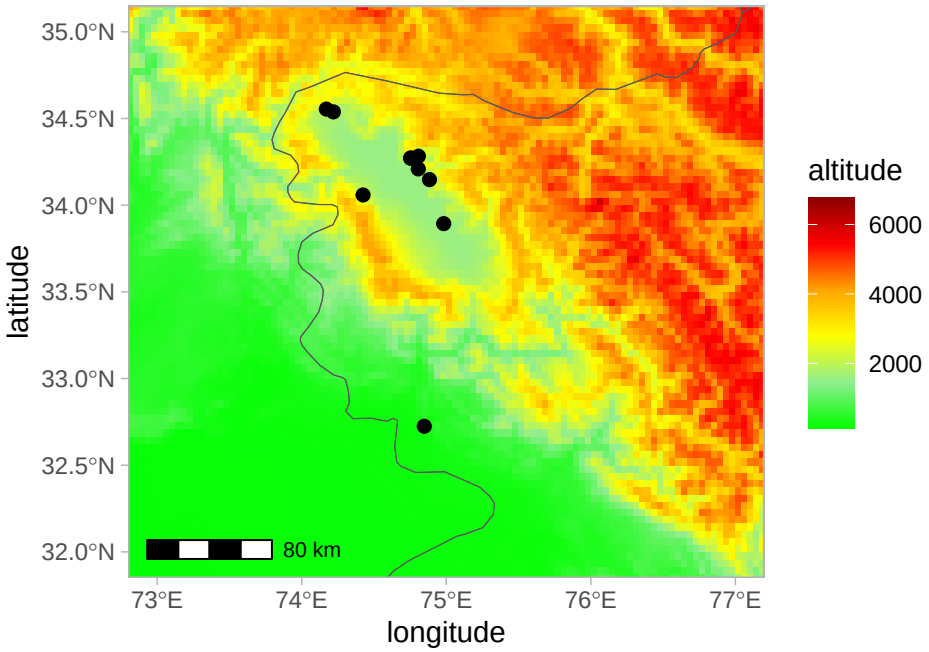
Read raw coordinates of 19 landmarks from Zenodo (Kaur et al., 2023)

```
wings <- read.csv("https://zenodo.org/record/8071014/files/IN-raw-coordinates.csv")

# extract individual and sample classifier from file name
wings$ind = substr(wings$file, 1, 14)
wings$sample = substr(wings$file, 1, 7)
head(wings, 2)
```

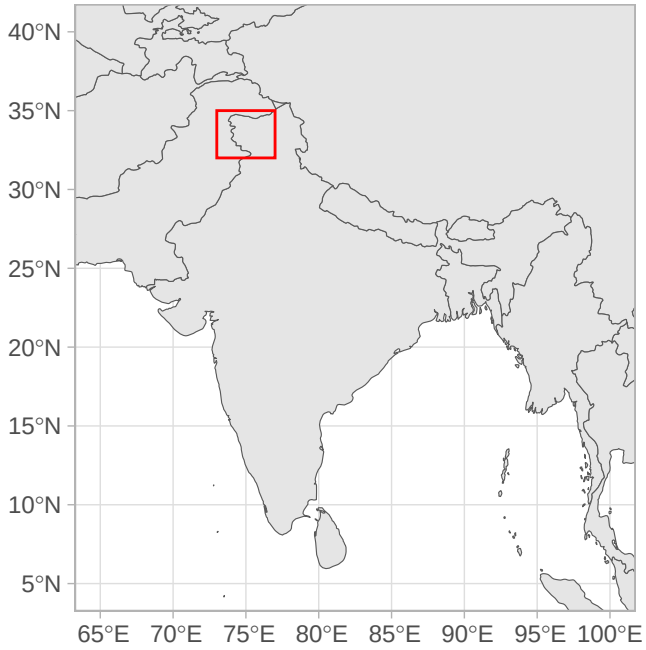
```
                    file  x1  y1  x2  y2  x3  y3  x4  y4  x5  y5  x6  y6  x7
1 IN-0001-000243-L.dw.png 195 169 213 166 268 246 261 192 267 108 337 250 386
2 IN-0001-000243-R.dw.png 180 251 193 243 272 290 231 242 214 166 333 266 392
   y7  x8  y8  x9  y9 x10 y10 x11 y11 x12 y12 x13 y13 x14 y14 x15 y15 x16 y16
1 282 369 258 406 224 382 204 422 171 427 131 438  99 451 270 479 235 559 194
2 272 367 258 386 210 357 204 379 156 364 120 361  85 448 236 458 189 512 119
  x17 y17 x18 y18 x19 y19            ind  sample
1 593 191 601 161  82 244 IN-0001-000243 IN-0001
2 542 101 539  72 104 369 IN-0001-000243 IN-0001
```

Map of sampling locations in India

```
# Read geographic coordinates
geo.data <- read.csv("https://zenodo.org/record/8071014/files/IN-data.csv")
sample.geo.data <- aggregate(cbind(geo.data$latitude, geo.data$longitude),
                             by = list(geo.data$sample), FUN = mean)
sample.geo.data <- reshape::rename(
  sample.geo.data, c(Group.1 = "sample", V1 = "latitude", V2 = "longitude"))

# Read elevation data
# In order to download the TIF file uncomment the two lines below
# download.file("https://geodata.ucdavis.edu/climate/worldclim/2_1/base/wc2.1_2.5m_elev.zi
# unzip("wc2.1_2.5m_elev.zip")
elev <- raster("D:/WorldClim/wc2.1_2.5m_elev.tif")

elev.color <- colorRampPalette(c("green", "lightgreen", "yellow",
                                 "orange", "red", "darkred"))

x.min = 73
x.max = 77
y.min = 32
y.max = 35
e <- extent(x.min-1, x.max+1, y.min-1, y.max+1)
elev = crop(elev, e)
elev.df  <-  data.frame(rasterToPoints(elev))
colnames(elev.df) = c("longitude", "latitude", "altitude")

world <- ne_countries(scale = "medium", returnclass = "sf")

ggplot(data = world) +
  geom_raster(data = elev.df, aes(longitude, latitude, fill = altitude)) +
  scale_fill_gradientn(colours = elev.color(100)) +
  geom_sf(fill = NA) +
  geom_point(data = sample.geo.data, aes(x = longitude, y = latitude), size = 2 )+
  coord_sf(xlim = c(x.min, x.max), ylim = c(y.min, y.max)) +
  annotation_scale(location = "bl", width_hint = 0.2)
```

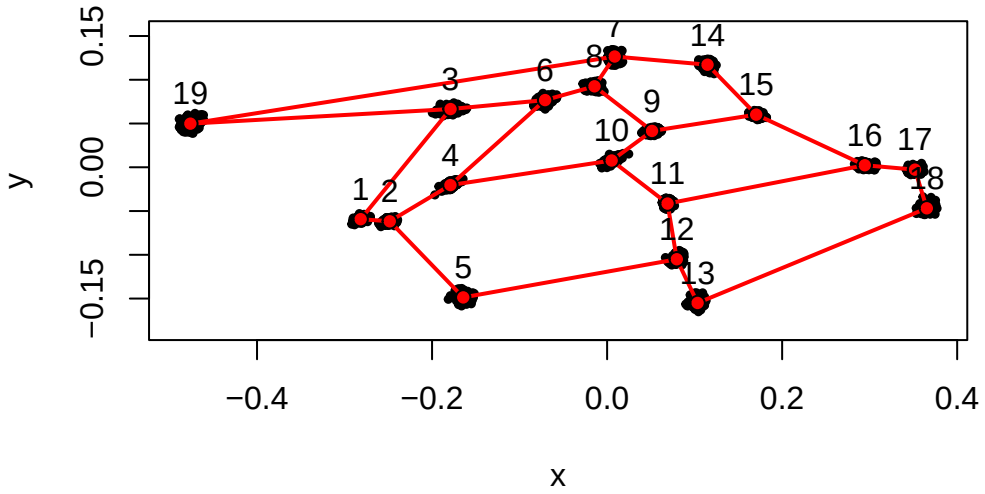



```
ggplot(data = world) +
  geom_sf() +
  annotate("rect", xmin = x.min, xmax = x.max, ymin = y.min, ymax = y.max,
           alpha = 0, color= "red") +
  coord_sf(xlim = c(65, 100), ylim = c(5, 40))
```



GPA alignment

```
# Convert from 2D array to 3D array
wings.coords <- arrayspecs(wings[xyNames], p, k)
dimnames(wings.coords)[[3]] <- wings$file

# Align the coordinates using Generalized Procrustes Analysis
GPA.IN <- gpagen(wings.coords, print.progress = FALSE)
consensus.IN = GPA.IN$consensus

# plot landmarks after alignment
plotAllSpecimens(GPA.IN$coords,links=links.apis,label=TRUE,
                 plot.param = list(pt.bg = "black", pt.cex = 0.5,
                                   mean.bg = "red", mean.cex=1, link.col="red",
                                   txt.pos=3, txt.cex=1))
```

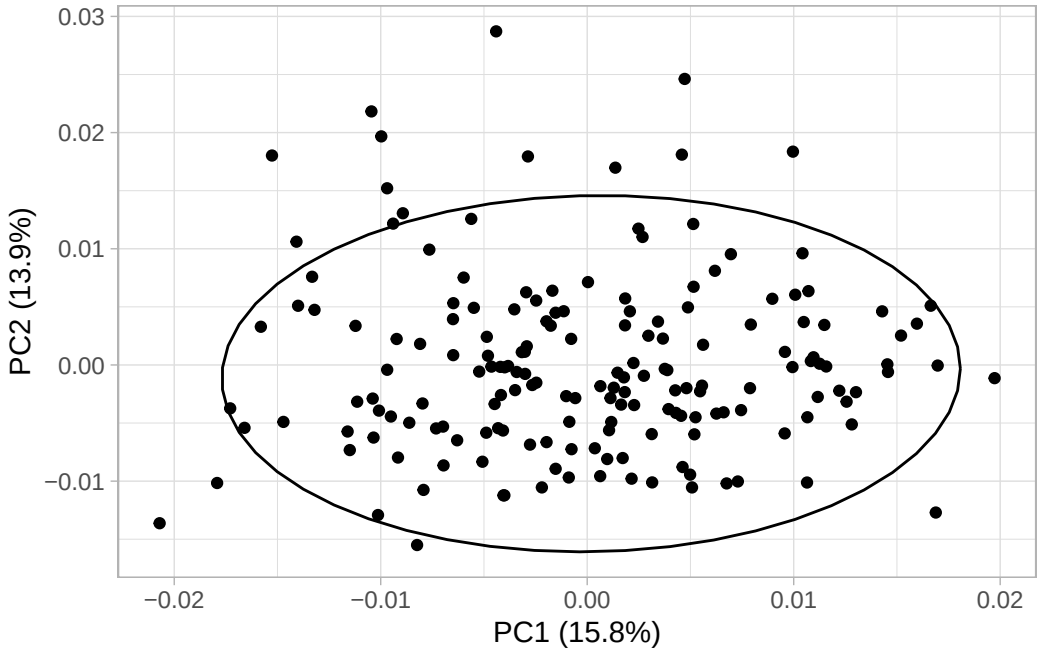



```
# Convert from 3D array to 2D array
wings.aligned.IN <- two.d.array(GPA.IN$coords)
colnames(wings.aligned.IN) = xyNames

# Average aligned coordinates within individuals
aligned.ind.IN <- aggregate(wings.aligned.IN, by = list(wings$ind), FUN = mean)
rownames(aligned.ind.IN) = aligned.ind.IN$Group.1
aligned.ind.IN <- subset(aligned.ind.IN, select = -c(Group.1))
```

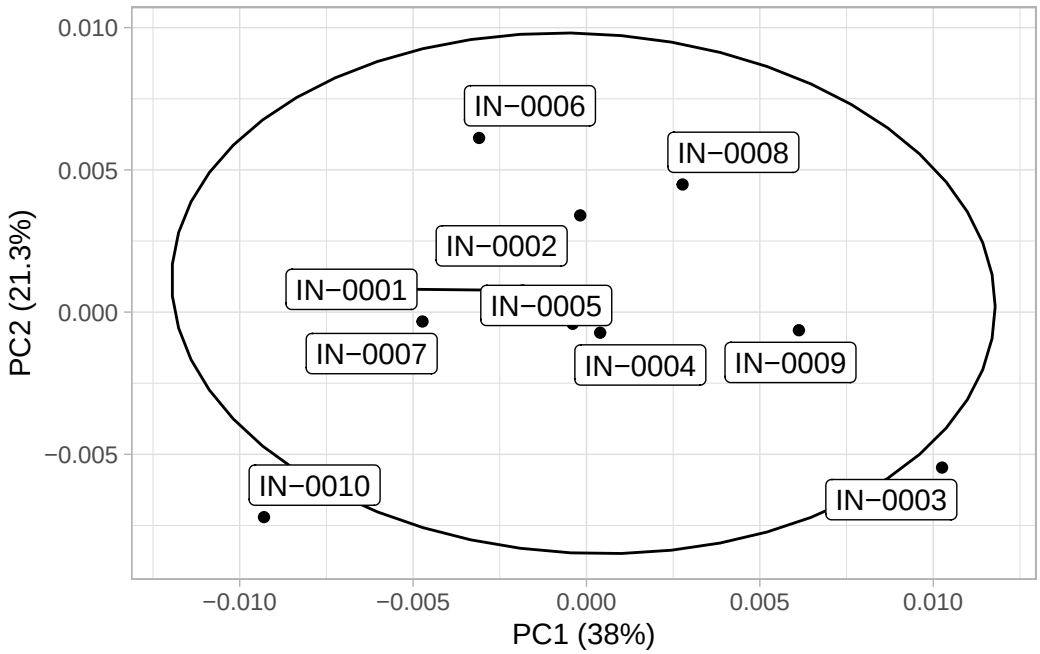
```r
# Average aligned coordinates within samples
sample.aligned <- aggregate(wings.aligned.IN, by = list(wings$sample), FUN = mean)
rownames(sample.aligned) = sample.aligned$Group.1
sample.aligned <- subset(sample.aligned, select = -c(Group.1))
```

Principal component analysis

```r
# PCA based on individuals
ind.pca <- prcomp(aligned.ind.IN[ , xyNames])
ind.pca.scores <- as.data.frame(ind.pca$x)

# create plot labels
variance.tab <- summary(ind.pca)$importance
variance <- variance.tab["Proportion of Variance", "PC1"]
variance <- round(100 * variance, 1)
label.x <- paste0("PC1 (", variance, "%)")
variance <- variance.tab["Proportion of Variance", "PC2"]
variance <- round(100 * variance, 1)
label.y <- paste0("PC2 (", variance, "%)")

ggplot(ind.pca.scores, aes(x = PC1, y = PC2)) +
  geom_point() +
  stat_ellipse() +
  xlab(label.x) + ylab(label.y)
```



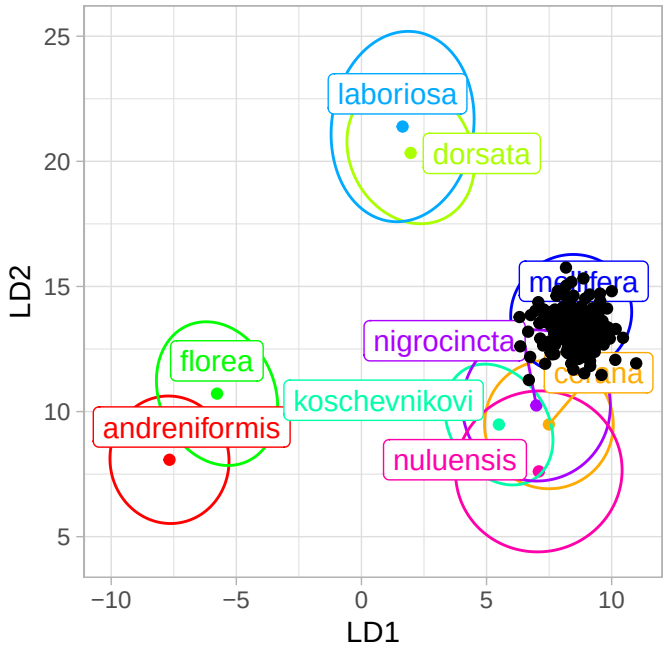
```
# PCA based on samples
sample.pca <- prcomp(sample.aligned[ , xyNames])
sample.pca.scores <- as.data.frame(sample.pca$x)

# create plot labels
variance.tab <- summary(sample.pca)$importance
variance <- variance.tab["Proportion of Variance", "PC1"]
variance <- round(100 * variance, 1)
label.x <- paste0("PC1 (", variance, "%)")
variance <- variance.tab["Proportion of Variance", "PC2"]
variance <- round(100 * variance, 1)
label.y <- paste0("PC2 (", variance, "%)")

ggplot(sample.pca.scores, aes(x = PC1, y = PC2)) +
  geom_point() +
  stat_ellipse() +
  xlab(label.x) + ylab(label.y) +
  geom_label_repel(label = rownames(sample.pca.scores))
```



Classification as species

```
# Classification of workers from India as species from genus *Apis*
id = xml2id("apis-species.dw.xml",
```

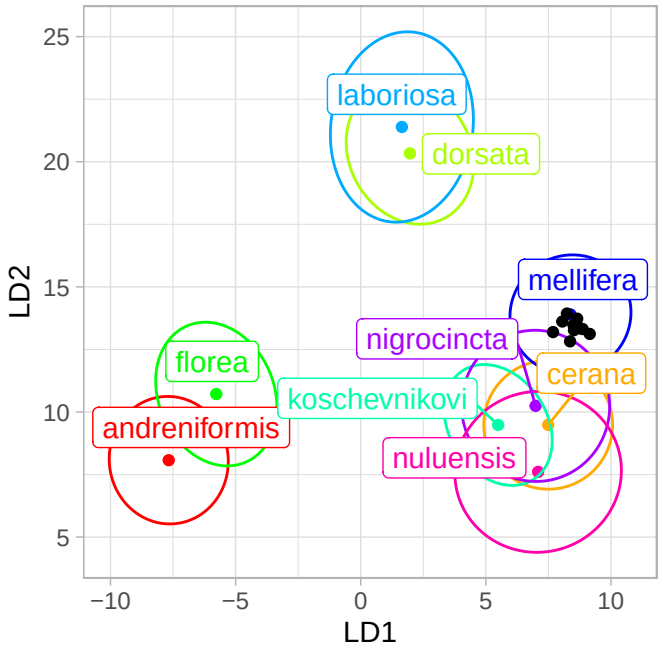
```
                  aligned.ind.IN,
                  average = FALSE)

id$plot
```




```
out.class = as.data.frame(id$id)
# All workers were classified as Apis mellifera
table(out.class$group)
```

```
mellifera
      175
```

```
# Classification of samples from India as species from genus *Apis*
id = xml2id("apis-species.dw.xml",
                  sample.aligned,
                  average = FALSE)
id$plot
```

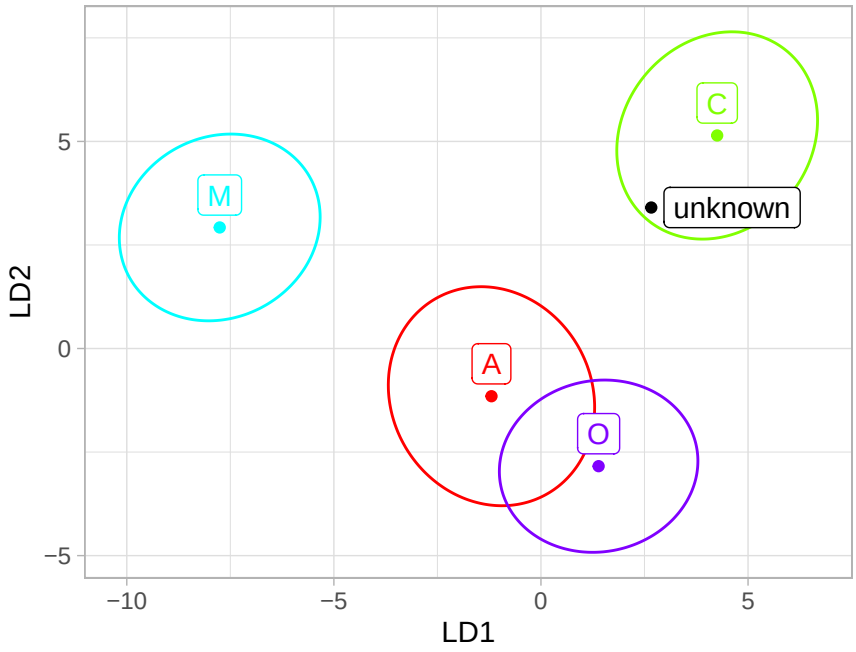



```
id$id
```

```
            group       P
IN-0001 mellifera 0.18315
IN-0002 mellifera 0.27792
IN-0003 mellifera 0.29372
IN-0004 mellifera 0.50807
IN-0005 mellifera 0.43472
IN-0006 mellifera 0.08873
IN-0007 mellifera 0.20471
IN-0008 mellifera 0.29236
IN-0009 mellifera 0.17865
IN-0010 mellifera 0.17901
```

Classification as lineages

```
# Classification of average of all samples from India as honey bee lineages
id = xml2id("apis-mellifera-lineage.dw.xml",
                 sample.aligned,
                 average = TRUE)
id$plot
```



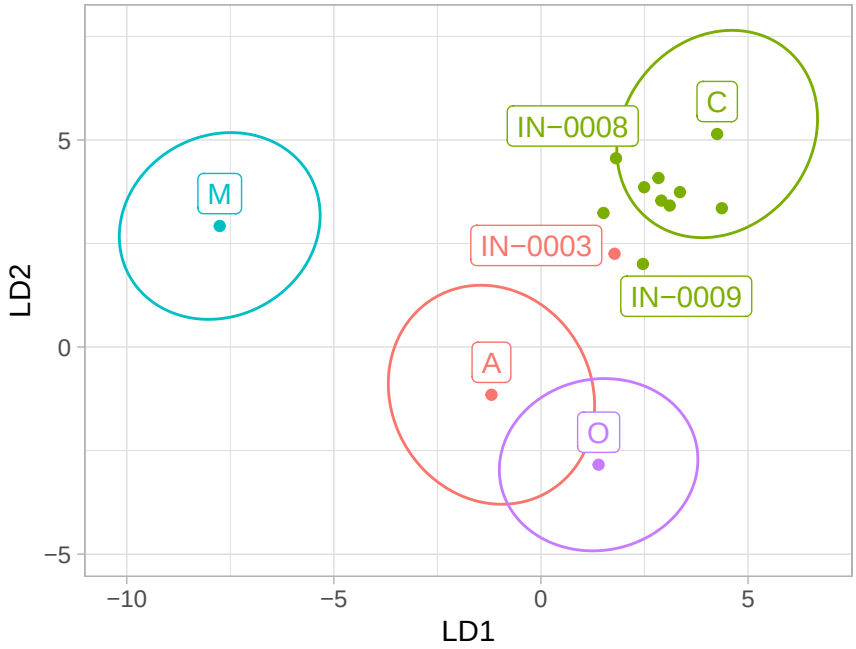

```
id$id
```

```
     MD2         P
A  37.86 5.995e-09
C  13.17 1.381e-03
M 115.07 1.032e-25
O  71.54 2.926e-16
```

```
# Classification of samples from India as honey bee lineages
id = xml2id("apis-mellifera-lineage.dw.xml",
                sample.aligned,
                average = FALSE)
# one sample IN-0003 was classified as lineage A
id$id
```

```
        group         P
IN-0001     C 1.055e-03
IN-0002     C 6.053e-03
IN-0003     A 2.139e-05
IN-0004     C 1.866e-01
IN-0005     C 4.528e-04
IN-0006     C 3.292e-02
IN-0007     C 9.116e-05
IN-0008     C 3.557e-03
IN-0009     C 6.220e-06
IN-0010     C 1.201e-05
```

```
# labeled samples in dimensions 1 and 2
id.data = xml2gmLdaData ("apis-mellifera-lineage.dw.xml")
means.LD = id.data$means %*% t(id.data$coefficients)
covEllipses (means.LD, id.data$covariances, 1, 2) +
  geom_point(id$LDx, mapping=aes(x = LD1, y = LD2, color = id$id$group)) +
  geom_label_repel(id$LDx, mapping=aes(x = LD1, y = LD2,
                                       color = id$id$group, label = rownames(id$LDx)))
```

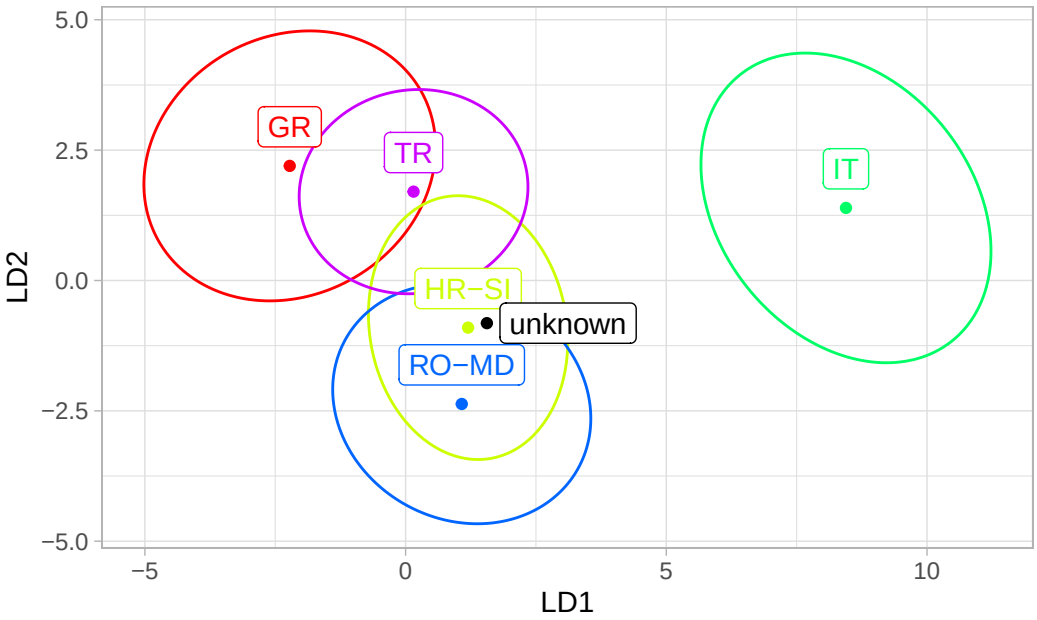



```
# Classification of samples IN-0003 as honey bee lineages
id = xml2id("apis-mellifera-lineage.dw.xml",
                 sample.aligned["IN-0003",],
                 average = TRUE)
id$id
```

```
     MD2         P
A  21.50 2.139e-05
C  29.78 3.411e-07
M 106.10 9.156e-24
O  53.54 2.364e-12
```

Classification as regions

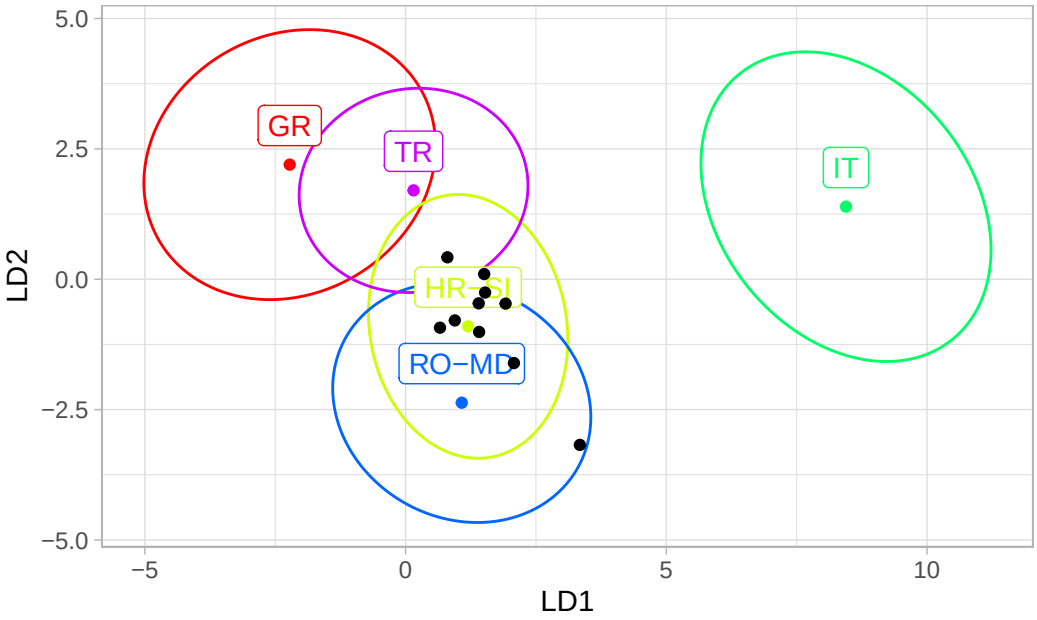
```
# Classification of average of all samples from India as regions in lineage C

id = xml2id("apis-mellifera-lineage_c_region.dw.xml",
                  sample.aligned,
                  average = TRUE)
id$plot
```




```
id$id
```

```
          MD2         P
GR     22.591 4.914e-05
HR-SI   8.410 3.825e-02
IT    110.800 7.380e-24
RO-MD   9.727 2.103e-02
TR     33.193 2.933e-07
```

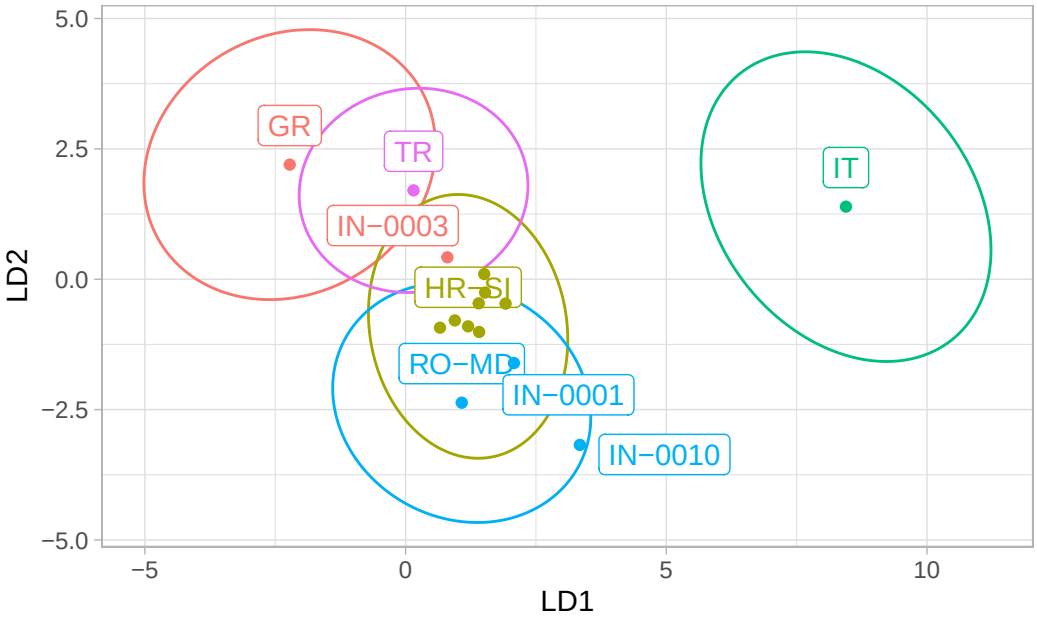
```
# Classification of samples from India as regions in lineage C

id = xml2id("apis-mellifera-lineage_c_region.dw.xml",
                  sample.aligned,
                  average = FALSE)
id$plot
```



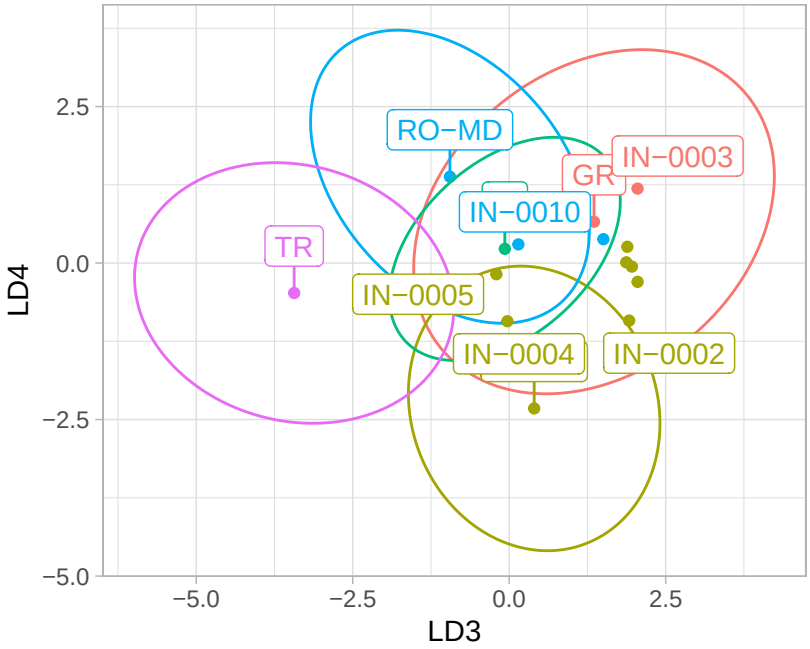

```
id$id
```

```
        group        P
IN-0001 RO-MD 0.017219
IN-0002 HR-SI 0.067142
IN-0003    GR 0.009275
IN-0004 HR-SI 0.404255
IN-0005 HR-SI 0.049901
IN-0006 HR-SI 0.003501
IN-0007 HR-SI 0.019839
IN-0008 HR-SI 0.010062
IN-0009 HR-SI 0.012511
IN-0010 RO-MD 0.032266
```

```
id.data = xml2gmLdaData("apis-mellifera-lineage_c_region.dw.xml")
means.LD = id.data$means %*% t(id.data$coefficients)
# labeled samples in dimensions 1 and 2
covEllipses(means.LD, id.data$covariances, 1, 2) +
  geom_point(id$LDx, mapping=aes(x = LD1, y = LD2, color = id$id$group)) +
  geom_label_repel(id$LDx, mapping=aes(x = LD1, y = LD2,
                                       color = id$id$group, label = rownames(id$LDx)))
```




```
# labeled samples in dimensions 3 and 4
covEllipses(means.LD, id.data$covariances, 3, 4) +
  geom_point(id$LDx, mapping=aes(x = LD3, y = LD4, color = id$id$group)) +
  geom_label_repel(id$LDx, mapping=aes(x = LD3, y = LD4,
                                       color = id$id$group, label = rownames(id$LDx)))
```



```r
# Classification of samples IN-0003 as honey bee lineages
id = xml2id("apis-mellifera-lineage_c_region.dw.xml",
                 sample.aligned["IN-0003",],
                 average = TRUE)
id$id
```

```
         MD2         P
GR     11.51 9.275e-03
HR-SI  22.19 5.953e-05
IT    132.75 1.383e-28
RO-MD  19.26 2.421e-04
TR     36.76 5.165e-08
```

References

Bustamante, T., Fuchs, S., Grünewald, B., & Ellis, J. D. (2021). A geometric morphometric method and web application for identifying honey bee species (*Apis* spp.) using only forewings. Apidologie, 52(3), 697-706. https://doi.org/10.1007/s13592-021-00857-7

Kaur H., Ganie S. A., & Tofilski A. (2023). Fore wings of honey bee (*Apis mellifera*) from Jammu and Kashmir, India [Data set]. Zenodo. https://doi.org/10.5281/zenodo.8071014

Nawrocka, A., Kandemir, İ., Fuchs, S., & Tofilski, A. (2018a). Computer software for identification of honey bee subspecies and evolutionary lineages. Apidologie, 49(2), 172-184. https://doi.org/10.1007/s13592-017-0538-y

Nawrocka, A., Kandemir, İ., Fuchs, S., & Tofiilski, A. (2018b). Dataset: Computer software for identification of honey bee subspecies and evolutionary lineages. Apidologie, 49, 172–184. https://doi.org/10.5281/zenodo.7567336

Oleksa, A., Căuia, E., Siceanu, A., Puškadija, Z., Kovačić, M., Pinto, M.A., Rodrigues, P.J., Hatjina, F., Charistos, L., Bouga, M., Prešern, J., Kandemir, I., Rašić, S., Kusza, S., & Tofilski, A. (2022). Collection of wing images for conservation of honey bees (*Apis mellifera*) biodiversity in Europe [Data set]. Zenodo. https://doi.org/10.5281/zenodo.7244070

Oleksa, A., Căuia, E., Siceanu, A., Puškadija, Z., Kovačić, M., Pinto, M. A., Rodrigues, P. J., Hatjina, F., Charistos, L., Bouga, M., Prešern, J., Kandemir, İ., Rašić, S., Kusza, S., Tofilski, A. (2023). Honey bee (*Apis mellifera*) wing images: a tool for identification and conservation. GigaScience, 12, giad019. https://doi.org/10.1093/gigascience/giad019

Information about session

```r
sessionInfo()
```

```
R version 4.3.2 (2023-10-31 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 11 x64 (build 22000)

Matrix products: default


locale:
[1] LC_COLLATE=Polish_Poland.utf8  LC_CTYPE=Polish_Poland.utf8
[3] LC_MONETARY=Polish_Poland.utf8 LC_NUMERIC=C
[5] LC_TIME=Polish_Poland.utf8

time zone: Europe/Warsaw
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] XML_3.99-0.16       xml2_1.3.6          officer_0.6.3
 [4] readxl_1.4.3        ggspatial_1.1.9     raster_3.6-26
 [7] sp_2.1-1            rnaturalearth_0.3.4 ggrepel_0.9.4
[10] ggforce_0.4.1       ggplot2_3.5.1       IdentiFlyR_0.1.0
[13] MASS_7.3-60         shapes_1.2.7        geomorph_4.0.6
[16] Matrix_1.6-1.1      rgl_1.2.1           RRPP_1.4.0

loaded via a namespace (and not attached):
 [1] gtable_0.3.4            xfun_0.41               htmlwidgets_1.6.3
 [4] lattice_0.22-6          vctrs_0.6.4             tools_4.3.2
 [7] generics_0.1.3          parallel_4.3.2          tibble_3.2.1
[10] proxy_0.4-27            fansi_1.0.5             pkgconfig_2.0.3
[13] KernSmooth_2.23-22      rematch_2.0.0           uuid_1.1-1
[16] scatterplot3d_0.3-44    lifecycle_1.0.4         compiler_4.3.2
[19] farver_2.1.1            textshaping_0.3.7       munsell_0.5.0
[22] terra_1.7-55            minpack.lm_1.2-4        codetools_0.2-19
[25] htmltools_0.5.7         class_7.3-22            yaml_2.3.7
[28] pillar_1.9.0            openssl_2.1.1           classInt_0.4-10
[31] rnaturalearthdata_0.1.0 nlme_3.1-163            zip_2.3.0
[34] tidyselect_1.2.0        digest_0.6.33           sf_1.0-14
[37] dplyr_1.1.4             labeling_0.4.3          polyclip_1.10-6
[40] fastmap_1.1.1           grid_4.3.2              colorspace_2.1-0
[43] cli_3.6.1               magrittr_2.0.3          base64enc_0.1-3
[46] utf8_1.2.4              e1071_1.7-13            ape_5.7-1
[49] withr_2.5.2             scales_1.3.0            rmarkdown_2.25
[52] httr_1.4.7              jpeg_0.1-10             cellranger_1.1.0
[55] ragg_1.2.6              askpass_1.2.0           evaluate_0.23
```

```
[58] knitr_1.45              rlang_1.1.2             Rcpp_1.0.11
[61] glue_1.6.2              DBI_1.1.3               tweenr_2.0.2
[64] reshape_0.8.9           rstudioapi_0.15.0       jsonlite_1.8.7
[67] plyr_1.8.9              R6_2.5.1                systemfonts_1.0.5
[70] units_0.8-4
```