```
### Essential AA LDA Demo

# start by loading the necessary libraries
library(MASS)
library(stats)
library(car)
library(ellipse)
library(ggplot2)
library(ggnewscale)

# let's make sure everything has loaded correctly
# running these lines of code should not produce any error messages
?lda()
?predict()
?ellipse()
?shapiro.test()
?leveneTest()

# set your working directory
setwd("/Users/AlexiBesser/Desktop")

# read in your data!
# there will be 2 csv files:
# (1) The producer file (Prod.csv)
# (2) The consumer file (Cons.csv)

# read in the producer data csv file and name it "prod"
prod <- read.csv("Prod.csv")

# read in the consumer data csv file and name it "con"
con <- read.csv("Cons.csv")

# we don't need the "X" column in these data frames
prod$X <- NULL
con$X <- NULL

# let's practice subsetting our data by creating separate dataframes
kelp <- subset(prod, spp == "kelp", select = c("ID", "ile", "leu", "lys", "phe", "thr", "val", "site"))
fito <- subset(prod, spp == "fito", select = c("ID", "ile", "leu", "lys", "phe", "thr", "val", "site"))
red <- subset(prod, spp == "red", select = c("ID", "ile", "leu", "lys", "phe", "thr", "val", "site"))
green <- subset(prod, spp == "green", select = c("ID", "ile", "leu", "lys", "phe", "thr", "val", "site"))

# let's merge two of these subsetted dataframes back together
greenfito <- rbind(fito, green)

# calculate the mean and standard deviations for a few essential AA
kelp.ile.mean <- mean(kelp$ile)
kelp.ile.sd <- sd(kelp$ile)

fito.ile.mean <- mean(fito$ile)
fito.ile.sd <- sd(fito$ile)

red.ile.mean <- mean(red$ile)
red.ile.sd <- sd(red$ile)

green.ile.mean <- mean(green$ile)
```

```
green.ile.sd <- sd(green$ile)

# test for normality (p-values above 0.05 indicate normality)
shapiro.test(kelp$ile)
shapiro.test(fito$ile)
shapiro.test(red$ile)
shapiro.test(green$ile)

# test for homogeneity of variance (p-values above 0.05 indicate homogeneity of variance)
leveneTest(ile ~ spp, data = prod)

# perform an ANOVA to look for differences in ile d13C values among species
ile.aov <- aov(ile ~ spp, data = prod)
summary(ile.aov)

# perform pairwise comparisons using Tukey Honest Significant Differences
TukeyHSD(ile.aov)

# make and save pdfs of box plots of producer ile d13C values
pdf("ProducerIled13CBoxplot.pdf", width = 12, height = 8)
boxplot(kelp$ile, fito$ile, red$ile, green$ile, names = c("Kelp", "POM", "Red", "Green"), xlab = "Primary
Producer", ylab = "Ile d13C")
dev.off()

# Okay! So now let's try our LDA model
# we'll start by creating an LDA with our producer essential AA data
prod_LDA <- lda(spp ~ ile + lys + val + thr + phe + leu, data = prod)

# let's look at which AA are driving the patterns!
prod_LDA

# a couple of other notes:
# (1) the 'Prior probabilities of groups' represents how many individuals of each species are present (e.g.,
35% of our samples are phytoplankton)
# (2) the 'Coefficients of linear discriminants' gives you information on how important each AA is for
#     distinguishing among groups along each linear discriminant axis - Larger absolute values mean those
AA are important in driving that LD
# (3) the 'Proportion of trace' tells you how much variation is explained by each LD axis

# now, let's calculate an error rate for our producer data set
# this is done with a cross validation approach
# in this function, the line 'CV = TRUE' makes the LDA perform a jacknifed (leave one out) model fit
prod_LDA_error <- lda(spp ~ ile + lys + val + thr + phe + leu, data = prod, CV = TRUE)

# now let's create a table which compares the classification of the LDA model to the actual species
ct.prod <- table(prod$spp, prod_LDA_error$class)
ct.prod

# the total percent of samples correctly classified is the sum of the diagonal of this table
sum(diag(prop.table(ct.prod)))

# this line of code tells us what % of each species is being correctly classified
diag(prop.table(ct.prod, 1))
```

```
# Let's plot these data!!!
# first, we need to export the new values of our individuals along the LD coordinates
# we could calculate this ourselves using the information in the LDA function
# but the predict() function in base R will do this for us! Woohoo!

# here we create a dataframe which contains these coordinates
datPred <- data.frame(Group = prod$spp, predict(prod_LDA)$x)
as.factor(datPred$Group)
str(datPred)


# PLOTTING THE DATA
ggplot(datPred, aes(x = LD1, y = LD2, color = Group)) +
  geom_point(size = 6) +
  theme_classic() +
  stat_ellipse(geom="path", aes(color = Group),
               alpha = 1,
               show.legend = FALSE,
               level = 0.95, size =1)


# it's bothering me that the red algae are purple
ggplot(datPred, aes(x = LD1, y = LD2, color = Group)) +
  geom_point(size = 6) +
  theme_classic() +
  scale_color_manual(values = c('#56B4E9', '#009E73', '#999999', '#D55E00')) +
  stat_ellipse(geom="path", aes(color = Group),
               alpha = 1,
               show.legend = FALSE,
               level = 0.95, size =1)

ggsave("LDADemoEE.pdf", plot = last_plot(), width = 10, height = 10)


# now let's classify the consumers based on their essential AA d13C values and save this into a
dataframe
consumers <- predict(prod_LDA, con[,c(2:7)])
datPred2 <- data.frame(Group = con$spp, consumers$x)

# we can look at how many individuals from each species were classified with each producer group
ct.consumers <- table(con$spp,consumers$class)
ct.consumers



# JUST THE SUBTIDAL SPECIES (to make it easier to visualize)
datPred3 <- datPred2[!(datPred2$Group == "ACH"|
                       datPred2$Group == "ENI"|
                       datPred2$Group == "TNI"|
                       datPred2$Group == "HHE"|
                       datPred2$Group == "COR"|
                       datPred2$Group == "PPU"|
                       datPred2$Group == "TAT"|
                       datPred2$Group == "CCH"|
                       datPred2$Group == "ZOOP"|
                       datPred2$Group == "film"),]
```

```
# plot the consumers on top of the producers
ggplot(datPred, aes(x = LD1, y = LD2, color = Group)) +
  geom_point(size = 6, alpha = 0) +
  theme_classic() +
  stat_ellipse(geom="path", aes(color = Group),
               alpha = 1,
               show.legend = FALSE,
               level = 0.95, size = 2) +
  geom_point(data = datPred3, size = 6, aes(color = Group)) +
  scale_color_manual(values = c('#000000', '#FFC000', '#0072B2', '#CC79A7', '#56B4E9', '#009E73',
                                '#C69F78', '#92D050', '#999999', '#D55E00'))
```