Constrained Ordination (Guerrero Negro)

Back to Table of Contents

All of the code in this page is meant to be run in `R` unless otherwise specified.

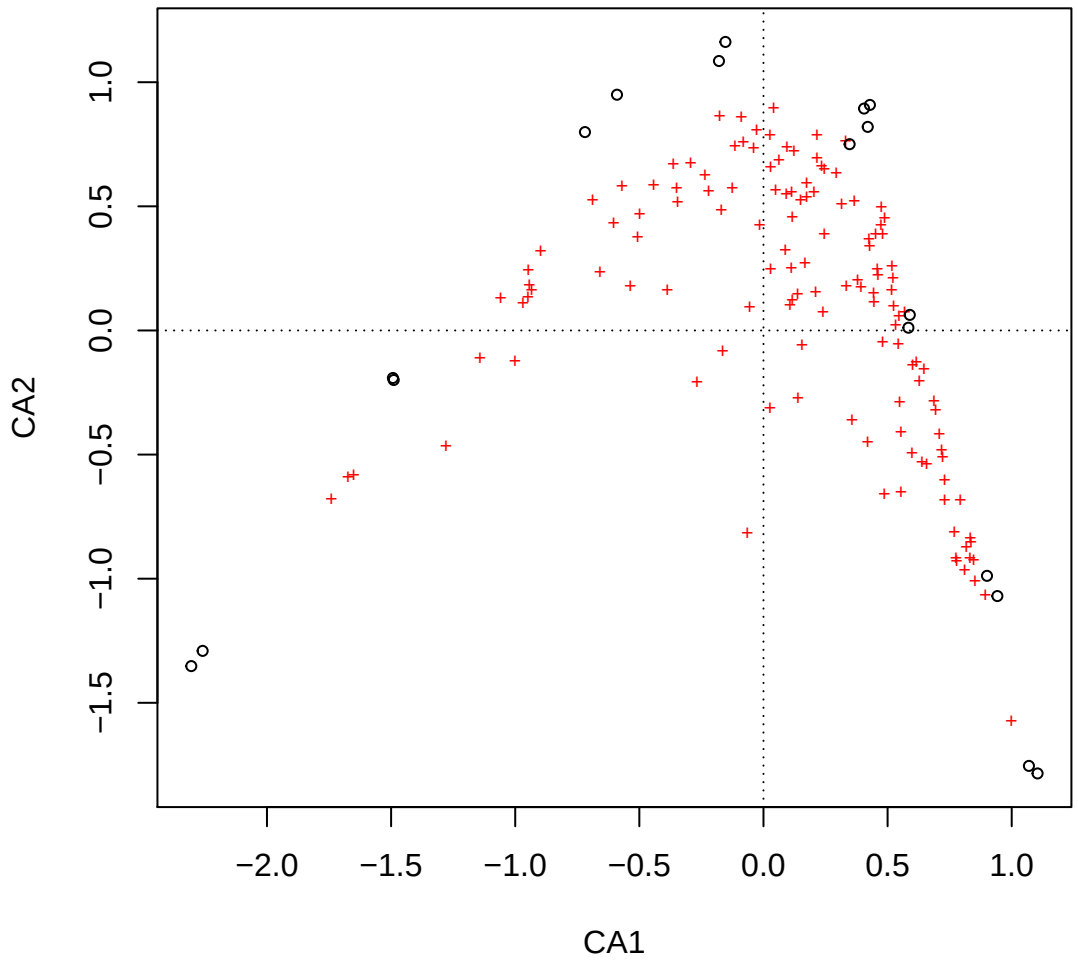
Load data and calculate distance metrics. For more explanations of these commands see Beta diversity

```
library('biom',quietly=TRUE, warn=FALSE)
library('vegan',quietly=TRUE, warn=FALSE)

# load biom file
otus.biom <- read_biom('otu_table_json.biom')

# Extract data matrix (OTU counts) from biom table
otus <- as.matrix(biom_data(otus.biom))

# transpose so that rows are samples and columns are OTUs
otus <- t(otus)

# convert OTU counts to relative abundances
otus <- sweep(otus, 1, rowSums(otus),'/')

# load mapping file
map <- read.table('map.txt', sep='\t', comment='', head=T, row.names=1)

# find the overlapping samples
common.ids <- intersect(rownames(map), rownames(otus))

# get just the overlapping samples
otus <- otus[common.ids,]
map <- map[common.ids,]

# Keep only OTUs present in at least 50% of samples
# This is fairly aggressive but will reduce the clutter in biplots
otus <- otus[,colMeans(otus>0)>.5]
```

Regular Correspondence Analysis

We have seen ordination using "Chi-square" distances and PCoA. But there is another interpretation of this approach. It is essentially equivalent to doing "Correspondence analysis," which tries to put the samples in order along the x-axis so that all species have a unimodal response to the primary gradient. In other words, each species should peak in abundance only one time somewhere in the middle of the gradient, or at one of the ends of the gradient, and should not have additional peaks anywhere along the gradient. If there is truly an ordering that makes this possible, then correspondence analysis will find it. We will use the `vegan` package to run correspondence analysis. We can also plot a biplot using vegan by calling `plot()` on the resulting CA object.

```
# run CA using vegan command
my.ca <- cca(otus)

plot(my.ca)
```



What fraction of total inertia is explained by each axis?

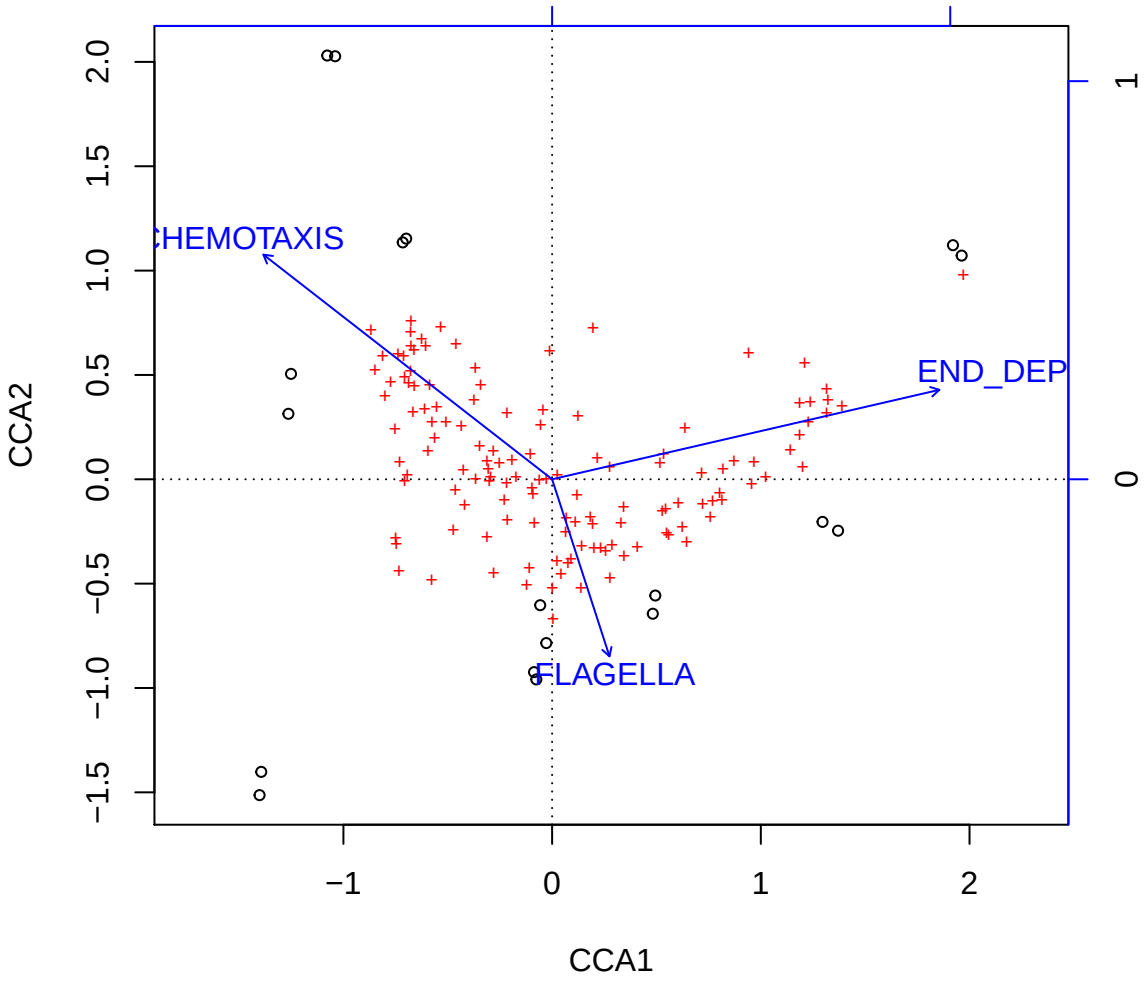
```
my.ca$CA$eig/my.ca$tot.chi
```

```
##         CA1         CA2         CA3         CA4         CA5         CA6
## 0.406798811 0.229194869 0.108818250 0.066303469 0.048693205 0.035283841
##         CA7         CA8         CA9        CA10        CA11        CA12
## 0.020154887 0.014594913 0.014142673 0.011368061 0.009541120 0.007872942
##        CA13        CA14        CA15        CA16        CA17
## 0.007518104 0.006628815 0.005971017 0.004739655 0.002375368
```

Constrained correspondence analysis.

Now we can perform "Direct Gradient Analysis," in which we relate species directly to environmental variable. According to Mike Palmer, "Canonical Correspondence Analysis is the marriage between CA and multiple regression." Like CCA, CA maximizes the correlation between species scores and sample scores. However, in CCA the sample scores are constrained to be linear combinations of environmental variables. Therefore CCA must explain less variation than pure CA.

```
# run CA using vegan command
my.cca <- cca(otus ~ END_DEPTH + CHEMOTAXIS + FLAGELLA, data=map)

plot(my.cca)
```



What fraction of total inertia is explained by each axis in CCA? Compare this to the fraction of total inertia explained by CA.

```
my.cca$CCA$eig/my.cca$tot.chi
```

```
##       CCA1       CCA2       CCA3
## 0.31237954 0.10001388 0.03609433
```

Assessing significance

We can compare the variance explained by the constrained and unconstrained correspondence analyses in the first axis. We want to see that constrained CA explains a good fraction of the explainable variation.

```
a <- my.ca$CA$eig/my.ca$tot.chi
b <- my.cca$CCA$eig/my.cca$tot.chi

# Test what fraction of CA1 variance is explained in CCA1
b[1]/a[1]
```

```
##      CCA1
## 0.7678969
```

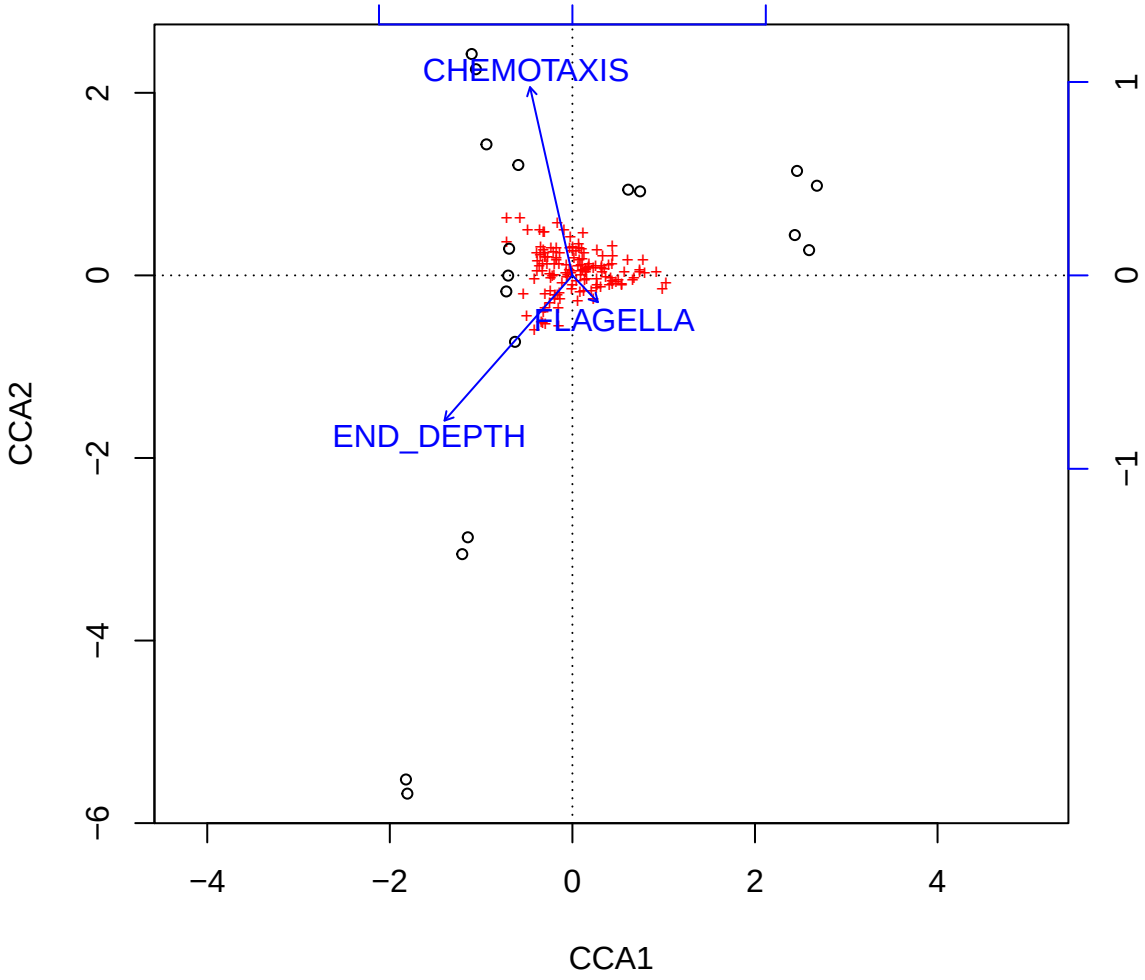
We can also simulate random data by shuffling or permuting the metadata values. We will shuffle them together to preserve correlations between metadata variables. If we shuffle them 10,000 times and calculate the variance explained in CCA axis 1 each time, we can compare this to the observed variation explained to get a p-value.

```
# store the observed value
obs.val <- my.cca$CCA$eig[1]/my.cca$tot.chi

# Perform 999 randomized CCAs
mc.vals <- replicate(999,{my.cca <- cca(otus ~ END_DEPTH + CHEMOTAXIS + FLAGELLA, data=map[sample(1:nrow

# include the observed value as one of the "null" values to be conservative
mc.vals <- c(mc.vals, obs.val)

# What fraction of the randomized values was greater than the observed value?
# this is the p-value
mean(c(obs.val, mc.vals) >= obs.val)
```

```
## [1] 0.002997003
```

Note that a randomized CCA does not look very good.

```
my.cca <- cca(otus ~ END_DEPTH + CHEMOTAXIS + FLAGELLA, data=map[sample(1:nrow(map)),])
plot(my.cca)
```


CHEMOTAXIS
FLAGELLA
END_DEPTH
CCA1
CCA2
−4
−2
0
2
4
−6
−4
−2
0
2
1
0
−1