Lab 9 – Part 2 – Distance-based Redundancy Analysis (db-RDA)

Distance-based redundancy analysis (dbRDA) is a method for carrying out constrained ordinations on data using non-Euclidean distance measures. The usual methods for constrained ordinations (CCA, RDA) use Euclidean distance, but this distance measure has been found to be inappropriate for some types of data. dbRDA circumvents this issue using a three-step process: first, a distance matrix is calculated using the distance measure of choice. Next, a principle coordinates analysis (PCoA) is done on the matrix. Finally, the eigenvalues obtained in the PCoA are plugged into an RDA. The function "capscale" is used R to carry out distance-based redundancy analyses. The procedure provides you with a pseudo-F value, which is a measure of the significance of the overall analysis. There are also options for finding out how much variation is accounted for by each axis, as well as for finding out which explanatory variables, and which axes, are significant.

There is one caveat you must be aware of before using this procedure. First, dbRDA (and RDAs in general) assume that dependent variables respond in a linear fashion to changes in your predictor variables. However, in many cases (particularly in ecology), responses to predictor variables are unimodal rather than linear. If this is the case for your dataset, you should look into doing a "distance-based CCA" instead. Distance-based CCA is not covered in this lab, but can also be carried out using capscale. It appears to be a matter of adding .cca to your function commands. If you are interested in this method, more details are available at: http://cc.oulu.fi/~jarioksa/softhelp/vegan/html/capscale.html.

9.1 db-RDA using "capscale" function in the "vegan" package of R

- For this exercise download "dbRDA.zip"; inside this you will find the dataset "AB_Climate_Trees.csv" (this is the dataset that we used in Lab 5). Set up your workspace and working directory with the file specified above; import the data file "AB_Climate_Trees.csv" into R. When doing a db-RDA, it is necessary to specify which columns are your environmental data and community data.

```
library(vegan)

AB_Climate_Trees=read.csv("AB_Climate_Trees.csv")
fix(AB_Climate_Trees)
row.names(AB_Climate_Trees)=AB_Climate_Trees$ECOSYS
species=AB_Climate_Trees[,11:23]
environment=AB_Climate_Trees[,3:10]
fix(species)
fix(environment)
```

- If you looked at this datatable you might have also noticed that some of ecosystems do not have any trees (such as the grasslands). Capscale cannot analyze dataset with zero species in an entire row. To get around this we can add a very small number to every species value in the dataset.

```
species001=(species + 0.001)
fix(species001)
```

- Now that we are going to proceed with a db-RDA, we have to decide which distance measure to use. You can choose the distance measure you want to use/feel is best for your data. One way we can do this by looking at the rank correlations between dissimilarity indices and gradient separation; the higher the value the better.

```
rankindex(environment, species001, indices = c("euc", "man", "gow",
"bra", "kul"),stepacross= FALSE, method = "spearman")
```

- The Kulczynski measure looks like a good distance measure to use. We'll proceed with the Bray-Curtis distance measure for this example though since it's one we are familiar with. Feel free to try the different measures to see what they do to the data.

```
dbRDA=capscale(species001 ~ MAT+MWMT+MCMT+TD+lnMAP+lnMSP+lnAHM+lnSHM,
environment, dist="bray")
plot(dbRDA)
anova(dbRDA)
```

9.2 Permutation tests to access significance of constraints

- Now that we have done our analysis, it would be nice to know what is significant.

```
anova(dbRDA)        ## overall test of the significance of the analysis
anova(dbRDA, by="axis", perm.max=500)     ## test axes for significance
anova(dbRDA, by="terms", permu=200) ## test for sig. environ. variables
```

9.3 Transforming negative eignvalues

- If you want to get rid of the negative eigenvalues when you are doing your analysis here are a few of ways you can:

1) Add a constant:

```
dbRDA_add=capscale(species001 ~ MAT+MWMT+MCMT+TD+lnMAP+lnMSP+lnAHM
+lnSHM,environment, dist="bray",add=TRUE)
plot(dbRDA_add)
anova(dbRDA_add)
```

2) Take the square roots of dissimilarities

```
dbRDA_sqrt=capscale(species001 ~MAT+MWMT+MCMT+TD+lnMAP+lnMSP+lnAHM
+lnSHM,environment,dist="bray",sqrt.dist=TRUE)
plot(dbRDA_sqrt)
anova(dbRDA_sqrt)
```

3) Do a square root transformation, Wisconsin double standardization (this emphasizes the environmental variables):

```
dbRDA_metaMDS=capscale(species001 ~ MAT+MWMT+MCMT+TD+lnMAP+lnMSP+lnAHM+
lnSHM, environment, dist="bray", metaMDS=TRUE, sqrt.dist=TRUE)
plot(dbRDA_metaMDS)
anova(dbRDA_metaMDS)
```

9.4 Modifying db-RDA plot (optional; included for use on your own data if interested)

- Now that you know what axes and environmental variables are significant, you may want to modify your graphs to change the vector sizes or to only include the important axes. See the script below for the code to calculate the loadings and correlations.

```
## getting the scores
scores.rda(dbRDA)             ## getting the scores from the analysis;
notice species and sites are together
scores_dbRDA=scores.rda(dbRDA)

site_scores=scores_dbRDA$sites     ## separating out the site scores;
get CAP1 and CAP2 scores
fix(site_scores)
species_scores=scores_dbRDA$species   ## separating out the species
scores
fix(species_scores)


## calculating loadings/environmental correlations with the axes
site_scores_environment=cbind(site_scores,environment) ## merge
correlations=cor(site_scores_environment)    ## calculate correlations
fix(correlations)
correlations2=correlations[3:10,1:2] ## the loadings we are interested
fix(correlations2)

## these environmental variables are the only significant ones
correlations3=correlations2[1:3,1:2]
fix(correlations3)
```

9.5 Doing this analysis using a CCA instead of a RDA

You may want to do this analysis using a CCA instead of a RDA. The “capscale” function has a RDA set as the default but you can opt to use a CCA instead by adding “.cca” to your code. For example, “anova.cca(**dbRDA**)” instead of “anova(**dbRDA**)”. “?capscale” in R for the details.