Diversity monitoring

KEY

10/17/2021

## Monitoring diversity

We will be working with data from 11 years of counts of birds at the San Dieguito River estuary, conducted between 2010 and 2020. The counts are done monthly, but I summed the numbers counted each year of each bird species so that we can focus on annual changes.

### The SDRP bird survey data

The data is in file sdrp\_waterbirds.xlsx, which you should have downloaded into the project folder for today. There are two worksheets you will use, sdrp\_waterbirds and guilds. We need to import both:

library(readxl) data.frame(read\_excel(“sdrp\_waterbirds.xlsx”, sheet = “sdrp\_waterbirds”)) -> birds data.frame(read\_excel(“sdrp\_waterbirds.xlsx”, sheet = “guilds”)) -> guilds

Import the data:

library(readxl)  
data.frame(read\_excel("sdrp\_waterbirds.xlsx", sheet = "sdrp\_waterbirds")) -> birds  
data.frame(read\_excel("sdrp\_waterbirds.xlsx", sheet = "guilds")) -> guilds

If you open the birds data you’ll see that it has a column for year, and a column with the four letter codes for 75 species of bird detected at the estuary. These are only species that use the estuary habitats, so many species that were counted but are not estuary species were omitted. The four letter codes are standard abbreviations, curated by the American Ornithologist’s Union, aside from a few species that are difficult to tell apart in the field and were thus lumped together (Short-billed Dowitcher and Long-billed Dowitcher are lumped as Dowitcher sp. with the column name DOSP, as are Western and Clark’s Grebe - WCGR - and Royal and Elegant Tern - ROET).

You can see what species these four letter codes refer to if you open the guilds data - the first two column of this data set is the four letter code and the species common name that it pertains to. The other two columns in guilds shows the species’ diet, and its foraging method - we will use the diet information when we assess functional diversity, but will not use the foraging method information today.

### Calculate species diversity

We will stick with the Shannon diversity index as our measure of diversity for this exercise. It is one of the older options, and it incorporates species richness (i.e. how many species are present) and species evenness (i.e. how similar the relative abundances are for the species present).

All of the functions we will need are in the package vegan. If you have not installed it already, switch to the Packages tab, click Install, and install “vegan” (if you’re using Citrix it should be installed already).

Once vegan is installed you can load it:

library(vegan)

library(vegan)

## Loading required package: permute

## Loading required package: lattice

## This is vegan 2.5-7

To calculate diversity we just need to use the diversity() function on the columns containing the species. There is only one column in the birds data that doesn’t have species data, so we could use use birds[,-1] to drop the first column with year in it. But, I prefer to make lists of the column names that we want to include, and then use the species list to select the columns from the data set - it is more clearer and more self-documenting, and makes trouble shooting commands easier.

To get the list of species, we just need to ask for the names() of the columns in the birds data, but then drop the first one so that only the four letter codes will be selected, and not year:

names(birds)[-1] -> species

names(birds)[-1] -> species

You should see an object called species in your environment that has 75 species codes in it.

Now to calculate Shannon diversity we just need to use this species list object to select the columns from birds, and then use those columns in the diversity() function (note that the square brackets have a comma before species - this is needed, as we can enter both the rows we sant to extract and the columns, and leaving the rows position inside the brackets tells R to include every row):

diversity(birds[,species]) -> birds$shannon birds[,c(“year”,“shannon”)]

diversity(birds[,species]) -> birds$shannon  
  
birds[,c("year","shannon")]

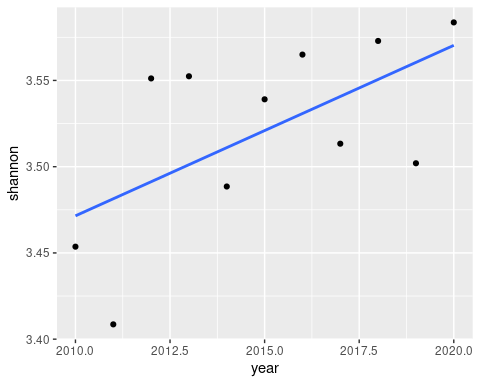
## year shannon  
## 1 2010 3.453623  
## 2 2011 3.408583  
## 3 2012 3.551160  
## 4 2013 3.552411  
## 5 2014 3.488506  
## 6 2015 3.539053  
## 7 2016 3.564980  
## 8 2017 3.513323  
## 9 2018 3.572887  
## 10 2019 3.501979  
## 11 2020 3.583642

The birds data set now has 78 variables, with the last one being a new column called shannon that contains the Shannon diversity index values for each year. The command birds[,c(“year”,“shannon”)] extracts the year and shannon columns from birds and displays them side by side so you can see them. You’ll see that there is some variation in diversity each year, but not a clear trend upward or downward. We can graph this using:

library(ggplot2) ggplot(birds, aes(x = year, y = shannon)) + geom\_point() + geom\_smooth(method = “lm”, se = F)

library(ggplot2)  
ggplot(birds, aes(x = year, y = shannon)) + geom\_point() + geom\_smooth(method = "lm", se = F)

## `geom\_smooth()` using formula 'y ~ x'



It appears that diversity is increasing over time. We can confirm this by running regression analysis on this relationship:

lm(shannon ~ year, data = birds) -> birds.shannon.lm anova(birds.shannon.lm)

lm(shannon ~ year, data = birds) -> birds.shannon.lm  
anova(birds.shannon.lm)

## Analysis of Variance Table  
##   
## Response: shannon  
## Df Sum Sq Mean Sq F value Pr(>F)   
## year 1 0.010745 0.0107448 5.1763 0.04895 \*  
## Residuals 9 0.018682 0.0020758   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

The increase in diversity over tiem is (barely!) statistically significant.

Since the Shannon index incorporates both species richness and evenness, we can add richness and evenness to the birds data to see which of the two seems to be responsible for the increase in diversity over time. Richness is just a count of non-zero species each year:

specnumber(birds[,species]) -> birds.richness

specnumber(birds[,species]) -> birds$richness

There is not a built-in function in vegan for evenness, but it can be calcuated from shannon and richness as:

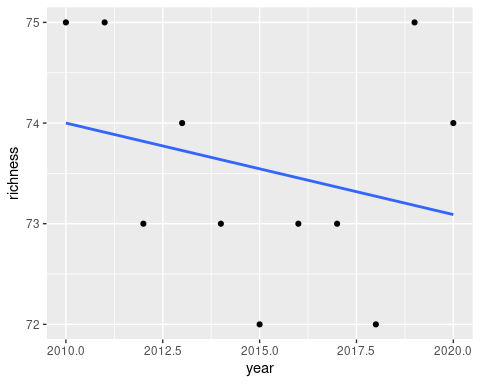
birdsshannon/log(birds$richness)

birds$evenness <- birds$shannon/log(birds$richness)

We can now make graphs of each of these components of the Shannon index - see if you can do this on your own, the command is just like the one you used to graph shannon:

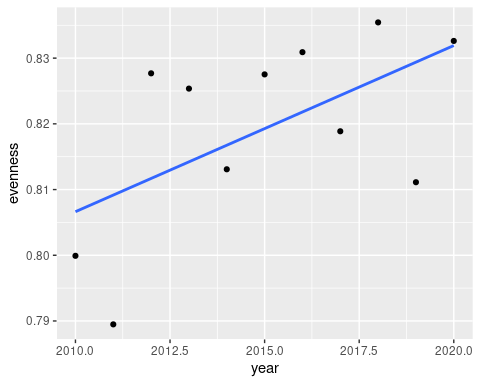
ggplot(birds, aes(x = year, y = richness)) + geom\_point() + geom\_smooth(method = "lm", se = F)

## `geom\_smooth()` using formula 'y ~ x'



ggplot(birds, aes(x = year, y = evenness)) + geom\_point() + geom\_smooth(method = "lm", se = F)

## `geom\_smooth()` using formula 'y ~ x'



So, interestingly, evenness has been going up, but richness has declined slightly (from 76 to 74) - the increase in diversity has to be due to an increase in the similarity in relative frequencies among the species that are present, with less numerical dominance by a few species, because the number of species has not increased over time.

## Functional diversity

Functional diversity refers to the diversity of species that play different ecological roles in the community. A very important functional difference between species is their diet - what a species eats affects how it interacts with other species, and an ecosystem that supports a wide range of different diets is meeting the needs of species with different niche requirements.

Groups of species that share functional characteristics are called “guilds”, and we will use the kind of food species eat to define guilds for these data. The column “Diet” in the guilds data set gives the diet for each species present in the data set. We will use this information in two different ways: 1) to split the species into their guilds, and assess diversity of species within each guild over time and 2) to sum the abundance of species within each guild, and then calculate the diversity of the guilds collectively over time. We will look at the within-guild diversity first.

First, we can split the guilds data set by diet to get separate lists of species for each diet:

split(guilds, guilds$Diet) -> diets

If you look at the diets object in the Environment tab you’ll see that you now have a “list” (which is a kind of R data set made up of named elements), with a subset of the guilds data for each diet - there is a named element for each diet, with the columns Code, Species, Diet, and Foraging.method in each one. We really only need the species codes within each diet guild, so this diets list gives us what we need.

Next, we will make use of a function in R that can apply a command repeatedly to each named element of a list, called sapply(). To get diversity within each guild:

sapply(diets, FUN = function(x) diversity(birds[,x$Code])) -> within.diet.shannon

within.diet.shannon

sapply(diets, FUN = function(x) diversity(birds[,x$Code])) -> within.diet.shannon  
  
within.diet.shannon

## Carnivore Granivore Herbivore Insectivore Mudflat.invertebrates Omnivore  
## [1,] 1.643774 1.0487206 1.1708544 2.095864 1.582191 1.977839  
## [2,] 1.534369 1.0759484 1.1279336 1.911512 1.586580 1.797794  
## [3,] 1.576117 1.0562079 1.1044136 1.962671 1.531769 2.092400  
## [4,] 1.469795 0.9874636 1.0410266 2.019088 1.532274 2.167889  
## [5,] 1.503018 0.9238894 0.9137314 2.174278 1.563213 2.131611  
## [6,] 1.605042 1.0649948 0.8983031 2.112954 1.336271 2.134680  
## [7,] 1.557263 0.7841056 0.9414134 2.230707 1.342605 2.195192  
## [8,] 1.462298 0.7370385 0.9326701 2.203962 1.478025 1.966129  
## [9,] 1.379517 0.7857993 0.9542687 2.140285 1.253236 2.221396  
## [10,] 1.265369 0.5005767 0.9369800 2.312042 1.372397 2.154421  
## [11,] 1.408208 0.8643207 1.1196023 2.237123 1.365989 1.992220  
## Piscivore  
## [1,] 1.718933  
## [2,] 2.202290  
## [3,] 2.101706  
## [4,] 1.996051  
## [5,] 1.772442  
## [6,] 2.300684  
## [7,] 2.212759  
## [8,] 2.071146  
## [9,] 2.184216  
## [10,] 2.083462  
## [11,] 2.213195

What just happened? The sapply() function works like so:

* The first argument is a list with named elements. sapply() extracts one at a time and feeds each to the function defined by FUN =. Since the named elements include the codes for species in each diet guild, we will make use of the Code variable to extract the birds that are in each guild, and then apply a function to them.
* FUN = defines the function to be applied. We want to calculate diversity(), of the birds that are listed in the Code variable. We are defining the function as a function of x, so x takes the place of the named element from diets. That is, the first named element is Carnivore, so the Carnivore data is extracted and fed to the function we define with the name x. We want to use the birds identified by the Code column, so using birds[ , x$Code] extracts the species in the Carnivore diet guild, which are then used to calculate diversity.
* This process is repeated for each of the diets, and then the results are bundled together and returned as a matrix.

To plot this pattern we need to first convert the matrix to a data frame, “stack” the columns, name the columns in the stacked data, and add the year back to the final data set. Stacking means that we want to have a single column for the shannon diversity values, and use another column to indicates the diet guild rather than having each guild in a separate column. These steps are done with the following commands:

data.frame(within.diet.shannon) -> within.diet.shannon # Convert the matrix to a data frame

stack(within.diet.shannon) -> within.diet.shannon.stacked # Stack the columns names(within.diet.shannon.stacked) <- c(“shannon”,“diet”) # Name the columns birdsyear # Add year to the stacked data

data.frame(within.diet.shannon) -> within.diet.shannon # Convert the matrix to a data frame  
  
stack(within.diet.shannon) -> within.diet.shannon.stacked # Stack the columns  
names(within.diet.shannon.stacked) <- c("shannon","diet") # Name the columns  
birds$year -> within.diet.shannon.stacked$year # Add year to the stacked data

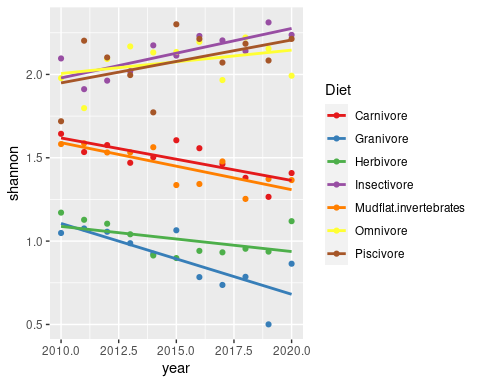
One thing that seems like it shouldn’t work but does is to assign the year column from birds to the stacked data set as a variable. If you look at within.diet.shannon.stacked you’ll see that there is a column for year that repeats for each guild, but in the birds data set there are only 11 years. As long as the number of rows in within.diet.shannon.stacked is a multiple of the rows in birds R will just use the years repeatedly. Since we stacked columns that had a row for each year this worked perfectly.

When we have the groups identified by a column we can use that column to graph the groups separately. To plot the within-diet guild diversity over time do the following:

ggplot(within.diet.shannon.stacked, aes(x = year, y = shannon, color = diet, group = diet)) + geom\_point() + geom\_smooth(method = “lm”, se = F) + scale\_color\_brewer(name = “Diet”, palette = ‘Set1’)

ggplot(within.diet.shannon.stacked, aes(x = year, y = shannon, color = diet, group = diet)) + geom\_point() + geom\_smooth(method = "lm", se = F) + scale\_color\_brewer(name = "Diet", palette = 'Set1')

## `geom\_smooth()` using formula 'y ~ x'



You’ll see both that the guilds differ in their diversities (i.e. the lines are at different heights on the graph), and that some guilds are increasing in diversity over time (omnivores, piscivores, and insectivores), while others are flat to decreasing (carnivores, mudflat invertebrate eaters, herbivores, and granivores).

To calculate the diversity of the diet guilds we need to first sum the counts for the species within each guild by year. This will give us total numbers of each of the diet guilds each year, from which we will be able to calculate Shannon diversity.

We can use sapply() again to calculate the sums of species within each guild. We will wrap the sapply() function inside of the data.frame() function this time so that the output is converted to a data frame in a single command:

data.frame(sapply(diets, FUN = function(x) rowSums(birds[,x$Code]))) -> diet.sums diet.sums

data.frame(sapply(diets, FUN = function(x) rowSums(birds[,x$Code]))) -> diet.sums  
  
diet.sums

## Carnivore Granivore Herbivore Insectivore Mudflat.invertebrates Omnivore  
## 1 1454 2296 3839 3012 1019 5954  
## 2 1798 3350 4936 5303 1293 8822  
## 3 1333 1882 4104 3115 1368 5628  
## 4 845 1623 3098 3164 1461 3561  
## 5 1160 833 2864 2346 1240 2421  
## 6 1069 946 2784 3019 1650 3328  
## 7 1359 1103 2800 2891 1188 3082  
## 8 1743 1246 3093 2467 1018 3768  
## 9 1429 852 2497 2562 1223 2649  
## 10 1687 1093 3353 2354 994 3725  
## 11 1735 1805 3512 3478 1027 5229  
## Piscivore  
## 1 6157  
## 2 2771  
## 3 4065  
## 4 3704  
## 5 2989  
## 6 1354  
## 7 1479  
## 8 3769  
## 9 2202  
## 10 2820  
## 11 4079

You’ll see that this operation summed all the birds seen in each diet guild. We can now calculate the diversity of the guilds using the diversity() function:

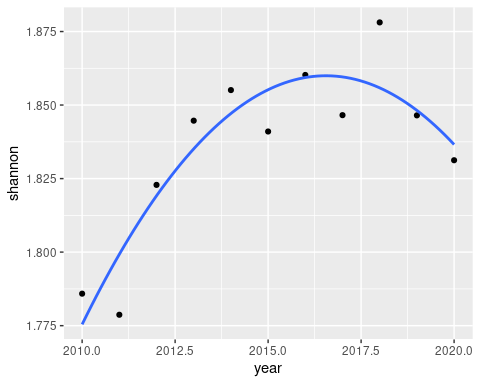
diversity(diet.sums) -> guild.shannon data.frame(year = birds$year, shannon = guild.shannon) -> guild.diversity

diversity(diet.sums) -> guild.shannon  
  
data.frame(year = birds$year, shannon = guild.shannon) -> guild.diversity

These two commands first calculated Shannon diversity from the data in diet.sums and assigned them to a vector called guild.diverstity, and then made a data frame from the year column in the birds data set and the guild.diversity vector. We can plot the guild diversity by year using:

ggplot(guild.diversity, aes(x = year, y = shannon)) + geom\_point() + geom\_smooth(method = “lm”, se = F, formula = y ~ poly(x, 2))

ggplot(guild.diversity, aes(x = year, y = shannon)) + geom\_point() + geom\_smooth(method = "lm", se = F, formula = y ~ poly(x, 2))



I included a function for the geom\_smooth() command this time, because the default for method = “lm” is a straight line, which fits the data poorly. Using y ~ poly(x, 2) fits a 2nd degree polynomial (i.e. a quadritic) to the data, which does a better job of capturing that guild diversity increased and then leveled off or possibly started to decline.

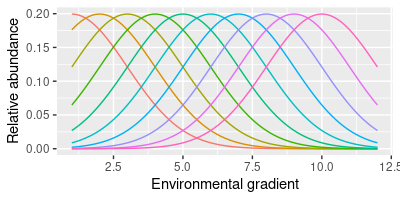
If you stop here don’t forget to knit and upload the Word document to complete the assignment.

## Species composition - optional

Diversity is fine as a way of measuring change in a community, but it’s possible for diversity to be unchanged as community composition is changing. Composition refers to the set of species that are present - two communities that are made up of completely different species can nevertheless have identical Shannon diversities if they contain the same number of species with the same relative frequencies.

The methods to statistically measure changes in composition are multivariate, meaning they use more than a single response variable. I teach a class in multivariate analysis, and have the time to explain how the methods work. If you’re interested in seeing how it is possible to analyze changes in species composition, please proceed.

Just like with diversity, there are various ways to analyze species comoposition. In general, the methods are called ordinations, in that they seek to put the species in order relative to their presence in the samples. Conceptually, we are looking for a pattern of species increasing to a peak and decreasing as they respond to their physiological tolerances or habitat preferences along some environmental gradient, like so:



Species along a gradient

Some methods just look for patterns of change in species composition, but don’t try to explain what the gradient is that the species might be responding to - these are indirect gradient analysis methods, in that a gradient is assumed to be responsible for the change but it is not explicitly included in the analysis. Direct gradient analysis methods use one or more variables that predict the change in species composition. We will use a direct gradient analysis method called canonical correspondence analysis (CCA), with year as our “gradient” to see whether composition is changing over time.

We will use the cca() function in vegan. The cca() function operates much like lm(), in that we will use a formula to show which variables are to be related to which, but since this is a multivariate method we will have multiple response variables instead of just one. The command is:

cca(birds[,species] ~ year, data = birds) -> birds.cca

birds.cca

cca(birds[,species] ~ year, data = birds) -> birds.cca  
  
birds.cca

## Call: cca(formula = birds[, species] ~ year, data = birds)  
##   
## Inertia Proportion Rank  
## Total 0.2530 1.0000   
## Constrained 0.0560 0.2213 1  
## Unconstrained 0.1970 0.7787 9  
## Inertia is scaled Chi-square   
##   
## Eigenvalues for constrained axes:  
## CCA1   
## 0.056   
##   
## Eigenvalues for unconstrained axes:  
## CA1 CA2 CA3 CA4 CA5 CA6 CA7 CA8 CA9   
## 0.06365 0.05524 0.02827 0.01413 0.01190 0.00991 0.00538 0.00527 0.00325

The basic output gives us some information about how much variation in species composition is explained by year. The measure used for variation with multivariate data is called inertia, and you can see that the “Constrained” inertia is 0.05, while the “Total” inertia is 0.25 - the proportion of the total explained by year is thus 0.05/0.25 = 0.2213, or about 22.3%. That’s a fair amount, considering how much variability there is in abundance of species. we can confirm that this is statistically significant using an equivalent to an ANOVA:

anova(birds.cca, by = “terms”)

anova(birds.cca, by = "terms")

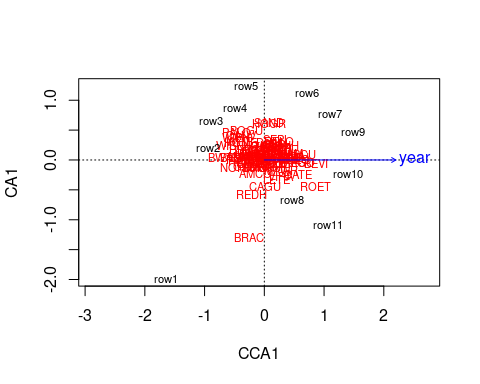
## Permutation test for cca under reduced model  
## Terms added sequentially (first to last)  
## Permutation: free  
## Number of permutations: 999  
##   
## Model: cca(formula = birds[, species] ~ year, data = birds)  
## Df ChiSquare F Pr(>F)   
## year 1 0.055998 2.5582 0.002 \*\*  
## Residual 9 0.197007   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

This is not a conventional analysis of variance - rather, it is based on a permutation test, which means that vegan randomly shuffles the bird data relative to year (this is called a permutation), conducts a CCA, and then records how much inertia is explained by year when the data data are randomly shuffled. It repeats this process many times, and with enough repetitions it’s possible to see whether our observed amount of inertia explained is big compared to what would occur by random chance. With a p-value of 0.001 and 999 random permutations our observed variation explained was bigger than any of the randomly shuffled versions - we would reject the null of no relationship between species composition and year, and can go on to try to understand what the change in composition has been.

With a highly multivariate data set like this, the challenge is in the interpretation of the patterns. There are both graphical and numerical ways of understanding the results. First the graphical.

The classic graph for an ordination plots the scores for species (i.e. columns) and samples (i.e. rows) in the same graph, with an arrow indicating the effect of the predictor variable(s). We only have one predictor, so only the position of species along the x-axis should be interpreted, and the arrow for year will need to lie flat along it. The plot is obtained with:

plot(birds.cca)



Since year is pointing to the right the species on the right side of the graph are becoming more common over time, while those on the left are becoming less common. The row labels indicate the individual samples relative to their species composition - you can see that row1 (2010) is the firthest to the left, and as you move to the right you move through the years more or less in order (row11, 2020, is not the furthest to the right, but the high numbers are on the right side of the graph and the low numbers are on the left). The closer together the row labels are right to left (ignoring up and down) the more similar their specie compositions are, and the species that tend to be found within them are the ones closest to them.

Thes biplots are neat, but with 75 species they are a little hard to interpret, and it’s difficult to properly disregard the y-axis (if we had two or more predictors we would have additional CCA axes and we could interpret both the x and y positions meaningfully, but the y-axis is a pattern of turnover in species that isn’t related to year, so it’s not of interest to us here). To understand what CCA1 means in terms of changes in species abundances we can calculate the species abundance with the scores for each year - species that are strongly positively correlated with CCA1 scores are increasing over time while those that are strongly negatively correlated with CCA1 are decreasing over time.

First we can get the CCA1 scores:

scores(birds.cca, display = “wa”, choices = 1) -> cca1.scores

scores(birds.cca, display = "wa", choices = 1) -> cca1.scores

The display = “wa” argument selects the scores for the years (wa refers to weighted average of the species, which are used to calculate the scores), and choices = 1 selects only the first CCA axis, CCA1, which is the only one available for this analysis.

We can add the years as a column in cca1.scores to make it easier to correlate the birds to year and CCA1:

data.frame(year = birds$year, cca1.scores)

data.frame(year = birds$year, cca1.scores) -> cca1.scores

We can then correlate all the species with these scores, and with year so that we can confirm that the correlations with CCA1 reflect changes over time (using View() will open the correlations so that you can see them in a tab - you can then sort on columns to put the CCA1 and year columns in order, and see which species had the highest and lowest correlations with each - you’ll need to come back to this tab to continue):

View(cor(birds[,species],cca1.scores))

View(cor(birds[,species],cca1.scores))

You’ll see that some species are increasing and others decreasing over time.

That’s it! Knit and upload your file to complete the assignment.