

Multivariate Statistics for Wildlife and Ecology Research

NRC 621

Lab 4: Correspondence Analysis and Nonmetric Multidimensional Scaling

The purpose of this lab is to give you hands-on experience using R to conduct a Correspondence Analysis (CA and DCA) and Nonmetric Multidimensional Scaling Analysis (NMDS) of a multivariate data set in support of your first lab project.

Background

Ecologists have long since abandoned reliance on PCA as an effective means of conducting an unconstrained ordination due to a number of well-known shortcomings. In particular, the underlying linear model of PCA has proven too restrictive for most ecological applications, in particular those involving the analysis of community (i.e., samples-by-species) data sets. In this second (and last) unconstrained ordination lab, we will use Correspondence Analysis (CA; also known as Reciprocal Averaging, RA) and its detrended version (Detrended Correspondence Analysis, DCA), in addition to Nonmetric Multidimensional Scaling (NMDS), to reduce the distribution of sample points to 2 or 3 dimensions, and plot the results in "ordinations." As before, depending on how successful we are at reducing the data set, we can seek patterns among the distribution of plots in ordination space, and explore possible environmental correlates with these. Remember, however, that at most we can establish correlation among the variables, not causation.

Correspondence Analysis and Detrended Correspondence Analysis:

Correspondence analysis is one of two names given to a method of ordination that was widely employed in the 70's and 80's. In one of the more interesting aspects of the history of vegetation science, it was soon determined that CA was independently developed by Hill in England (under the name reciprocal averaging) and by Benzecri in France (under the name analyse factorielle des correspondences). Legendre and Legendre (1998) trace its origins back farther than that outside of ecology.

CA can be calculated by a number of different algorithms, including those based on eigenanalysis and those based on reciprocal averaging. Recall that PCA is specifically an eigenanalysis of a correlation or covariance matrix, and PCO is an eigenanalysis of a broad variety of symmetric dissimilarity/distance matrices. These two methods both involve the linear mapping of samples into a reduced ordination space; they differ in the metric employed: Euclidean and non-Euclidean, respectively. CA is essentially an eigenanalysis of a Chi-square distance matrix. A Chi-square distance matrix is defined by the deviation from expectation. Similar to the Chi-square statistic, it is defined as:

$$(\text{observed} - \text{expected}) / \sqrt{\text{expected}}$$

where observed and expected are abundances of species in sample plots. Species which occur in

a sample plot in an abundance greater than expected have positive values, and species which occur less than expected or which are absent have negative values. Expected values are based on the row and column sums of the vegetation matrix. The use of the chi-square metric results in a weighted linear mapping of samples (and species) into a reduced ordination space, where the particular weighting scheme (doubly weighting based on row and column totals) assumes a unimodal (rather than linear) species response to the underlying gradients.

In contrast to the other eigenvector ordination methods, CA provides a configuration for species as well as sample plots in the ordination. That is, species centroids (the hypothetical mode of their distribution along the eigenvectors) can be plotted as well as the locations of the plots. This dual nature is one of the characteristics that first attracted ecologists to CA.

We can perform CA much as we did PCA, using the original data matrix (as opposed to a dissimilarity/distance matrix), and there are a number of different implementations of CA in R. A function called `ca()` is included in the `mva` library. The `CoCoAn` library includes CA as a special case of CCA in the function `CAIV()`, and the `vegan` library includes CA as a special case of `decorano()` or `cca()`. The method presented here is due to Legendre and Legendre (1998), who calculate CA as an eigenanalysis of a particular matrix.

CA is a much better and more robust method for community ordination than principal components analysis. However, with long ecological gradients it suffers from some drawbacks or “faults” which were corrected in Detrended Correspondence Analysis (DCA):

1. Single long gradients appear as curves or arcs in ordination: the solution is to detrend the later axes by making their means equal along segments of previous axes.
2. Sites are packed more closely at gradient extremes than at the center: the solution is to rescale the axes to equal variances of species scores.
3. Rare species seem to have an unduly high influence on the results: the solution is to down-weight rare species.

All these three separate tricks are incorporated in the `vegan` function `decorana()`, which is a faithful port of Mark Hill’s original program with the same name.

Note, while DECORANA was the ordination method of choice in the 70's and 80's, it was severely criticized for some of the "detrending" algorithms. While they were arguably inelegant (if not heavy-handed), empirically they were regarded as a significant improvement on CA. Part of the reason that the controversy died is that Canonical Correspondence Analysis (CCA) came along and rendered many of the arguments moot. CCA is presented in a later lab.

Nonmetric Multidimensional Scaling:

Principal Coordinates Analysis (PCO, also known as Metric Multidimensional Scaling, MDS) is an eigenvector technique to project a dissimilarity/distance matrix to fewer dimensions. It is possible to prove that this projection is the best possible rigid geometric projection of the data in

n -dimensional dissimilarity space into a reduced k -dimensional ordination space. That means that it is not possible to obtain a projection of strictly parallel lines from a higher dimension to a lower dimension that maximizes the variance of points along the axes as well as PCO.

However, it's not necessarily true that a rigid geometric projection is the best MAPPING, as opposed to projection, from a higher space to a lower. By mapping, it is meant simply that there is a one-to-one correspondence between the points in high-dimensional space and low dimensional space, but that we can violate the parallel lines of geometry at will. Rather than attempting to maximize variance along the axes, instead we can try to maximize the correlation of distances in the graph to the calculated dissimilarities/distances in the matrix.

In contrast to PCO, there is not a single best algorithm for achieving the maximum correlation, and many algorithms have been developed as solutions to this problem. We will focus on just one: Kruskal's non-metric multidimensional scaling (or NMDS). Kruskal's NMDS attempts to minimize something called "stress", the square root of the ratio of the squared differences between a monotonic transformation of the calculated dissimilarities/distances and the plotted distances and the sum of the plotted distances squared. It sounds more complicated than it is. Essentially, it tries to maximize the rank correlation between the calculated dissimilarities/distances and the plotted distances, allowing tied distances to not have identical plotted distances, only sequential ranks.

NMDS employs an iterative algorithm where initial estimates of the positions of samples are adjusted to minimize the stress until further iterations do not achieve a sufficient improvement. Accordingly, NMDS is somewhat sensitive to the initial positions, and in fact sometimes settles into a local optimum that is not the best solution. This seems especially common with high dimensional solutions ($k > 4$). There are several approaches to minimize the problem. One approach is to try multiple random starts, and to keep the best result (lowest stress). You are never guaranteed that you have found the best solution, but if the majority of your results achieve similar low levels of stress, then you can be reasonably certain that you have at least found a GOOD solution, if not the theoretical best. Another alternative, and one used as the default in R, is to use a PCO as the initial position, and then to improve on that. We already know that PCO is the optimal geometric projection, so that starting there should be a good start. It also guarantees that the NMDS solution will be no worse than the PCO.

NMDS overcomes many of the problems faced by PCA, CA and DCA and is therefore the method of choice among many ecologists. Despite the appealing nonparametric nature of NMDS, it is not necessarily a panacea for unconstrained ordination of ecological data sets. In particular, the dependence of the solution on the dimensionality chosen, the lack of ordering of the extracted axes, and the difficulty in evaluating the solution in terms of variance explained, make this technique a challenge for many users.

NMDS was contributed to S-Plus and R by Venables and Ripley as part of the MASS library, and is called isoMDS in their library. It is their efficient isoMDS routine that establishes the default of starting with a PCO. Once again, we will make use of the functions available in the

vegan library. Specifically, we will use the metaMDS() function which is simply a wrapper for the isoMDS function, but many additional features.

Provided Data Set

The provided sample data set is taken from research conducted in the Drift Creek Basin in the central Oregon Coast Range during 1990 (McGarigal and McComb, unpubl. data). This data set is described in detail in the second lab handout. Briefly, the full data set includes 164 observations (rows) associated with the sample points/plots and 63 bird species variables (columns) and 48 habitat (environment) variables (columns), although the latter are organized into three logical groups of environmental variables: geomorphology (14 variables), landscape composition (20 variables) and landscape heterogeneity (14 variables).

Exercise

In this exercise, you will work in your groups to analyze your data set using CA/DCA and NMDS. If you are not using the provided data set, birdhab, and are instead using one of your own, you will need to adjust the following procedures accordingly to suit the characteristics of your data set.

1. Set up your R work session.

From the course website download the sample data set to a local workspace or copy your own data set to the local workspace. In addition, download the mvstats R library from the course website to your local workspace.

Open R and change the current working directory to your local workspace.

On the computer lab PC's, install the following libraries from a nearby CRAN mirror and load them by typing:

```
library(vegan)
library(MASS)
library(fields)
library(scatterplot3d)
library(mgcv)
library(akima)
library(ellipse)
```

In addition, source the mvstats library (which is not a formal library) by typing:

```
source('mvstats.R')
```

Finally, import the sample data set as modified in the second lab set by typing:

```
birdhab<-read.csv('birdhab.adjusted.csv',header=TRUE)
```

where 'birdhab.adjusted.csv' is the modified original data set. Now that we've got the data into a data frame, we are ready to start the analysis. But first, to make things a little easier later, let's create a couple of work data sets that contain only the species variables and habitat variables, respectively.

```
rip.bird<-subset(rip.birdhab,select=AMGO:WWPE)  
rip.hab<-subset(rip.birdhab,select=VAREA:CONEDGE)
```

Note, the environmental variable set can contain both numeric variables and factors (categorical), although in this case rip.hab contains all numeric variables. In this case, the grouping variable SUB (subbasin) remains in the original data set, birdhab, only.

2. Evaluate suitability of the data set.

First, make sure the data set is suitable for CA/DCA/NMDS.

- single set of interdependent species variables (i.e., no distinction between independent and dependent variables).
- continuous (e.g., percent cover), presence/absence or count variables; all variables should be measured on the same scale.
- every sample entity must be measured on the same set of variables (i.e., no missing data).

Ideally enough samples should be taken to adequately describe each distinctive community or ecological condition and in community data sets to ensure that each species optima is estimated accurately and precisely by the sample data set (i.e., enough to insure stable parameter estimates). Note, in contrast to PCA, there are no sample-to-variable ratio considerations. In fact, it is quite common in community data sets for the number of species (variables) to exceed the number of samples.

To review the dimensions of the data sets, type:

```
dim(rip.bird)  
dim(rip.hab)
```

3. Evaluate the model assumptions.

CA/DCA is based on an underlying theoretical model which has a number of assumptions that must be met for valid statistical inference. NMDS, on the other hand, is a nonparametric procedure that is mostly free of restrictive assumptions (but see below).

Independent random sample:

For valid statistical inference, all procedures require that the samples be independent. If our

goals are descriptive, as is often the case, this assumption can be relaxed. The best solution to this requirement is a sound study design.

Outliers:

CA/DCA are especially sensitive to outliers, in particular, rare species and samples containing only rare species have a disproportionate effect on the ordination due to the use of the chi-square distance metric, which doubly weights observations by row and column totals. Here, we may want to create two data sets, one with and one without outliers (samples and species) and compare the ordination results below to assess the effects of the outliers on the ordination. NMDS is less sensitive because only the rank order in dissimilarities is used, but outliers can be problematic for the Monte Carlo randomization tests that are often used to assess the significance of the extracted axes.

Response Function Shape:

The quality of the ordination by CA/DCA is dependent on how well species' distributions along the underlying gradients can be represented as *unimodal* distributions. This is not an unreasonable assumption for most species if the sampled environmental gradient is long. Often, however, the gradient length is too short for many species to exhibit a complete rise and fall in abundance. There is no simple way to fully assess this assumption. However, once the CA/DCA has been conducted, we can examine these relationships through overlays of each species on the ordination plot.

NMDS, on the other hand, is a nonparametric procedure that makes no assumption of linearity or unimodal response functions. Instead, the assumption of linearity is replaced with the less problematic assumption of a monotonic relationship between distances in the original p -dimensional space and distances in the reduced k -dimensional ordination space. The best way to assess this assumption is with the shepard plot (see below).

4. Conduct the CA/DCA/NMDS and examine the results.

Correspondence Analysis (CA or RA):

We will use the `cca()` function in the `vegan` library, which allows for both unconstrained correspondence analysis (CA) as well as the constrained version, canonical correspondence analysis (CCA), which we will come back to in a later lab. The `cca()` function accepts matrices or data frames, computing "scores" for each sample and each species in the matrix or data frame. The scores are derived from the corresponding eigenvectors or, equivalently, from the weighted averages of the opposing scores. For example, species scores represent the weighted average of the sample scores and the sample scores can be calculated as weighted averages of the species scores.

The `cca()` function expects the input matrix or data frame to contain only the samples and species

of interest. Thus, make sure the data set has just the desired set of samples and species, and make sure that the data have already been transformed, if appropriate. Note, CA has a built in chi-square distance, so a row or column standardization may not be appropriate. To conduct the CA, simply type:

```
rip.bird.ca <- cca(rip.bird)
rip.bird.ca
```

The results of a CA are complex and stored in a list, including:

- eigenvalues (inertia) associated with each axis
- eigenvectors (sample and species scores)

Eigenvalues and Inertia:

Recall that in PCA eigenvalues represent ‘variance’ explained. In CA, eigenvalues represent ‘inertia’, where total inertia equals the Chi-squared statistic of the data matrix standardized to unit total. Thus, the eigenvalues are something akin to variance, but they are not variances per se. The chi-square statistic is a measure of association between samples and species. If species are not independently distributed among samples, so that species tend to co-occur in samples, the chi-square statistic will be large. The larger the chi-square statistic, the greater the correspondence of species’ distributions among samples, and vice versa. The total inertia can be calculated directly by computing the Pearson’s chi-square statistic of the data matrix after standardizing to the matrix total, as follows:

```
chisq.test(rip.bird/sum(rip.bird))
```

You should not pay any attention to P-values which are certainly misleading, but notice that the reported X-squared is equal to the total inertia.

To see the inertia (eigenvalue) of each axis, type:

```
rip.bird.ca$CA$eig
```

Note the syntax above. The results of the CA were stored in an object called ‘rip.bird.ca’. This object is a list containing several components, including one called ‘CA’. This component is actually another list containing several components, including the eigenvalues (‘eig’). The call requests the list component named ‘eig’ in the list component named ‘CA’ in the object named ‘rip.bird.ca’.

We may wish to test the statistical significance of the first several eigenvalues using a Monte Carlo (randomization) test, as follows:

```
ordi.monte(rip.bird, 'ca', dim=5)
```

We can look at this same information graphically with the `ordi.scree()` function:

```
ordi.scree(rip.bird.ca)
```

Eigenvectors (sample and species scores):

To see the raw sample and species eigenvectors for the first three axes, type:

```
rip.bird.ca$CA$u[,1:3] - sample eigenvectors ('u')
```

```
rip.bird.ca$CA$v[,1:3] - species eigenvectors ('v')
```

Alternatively, to see the eigenvectors scaled by the corresponding eigenvalues, type:

```
rip.bird.ca$CA$u.eig[,1:3]
```

```
rip.bird.ca$CA$v.eig[,1:3]
```

The eigenvalues and scaled eigenvectors can also be output with the `summary()` function, that has some additional features that allow us to change the scaling of the samples and species scores. For example,

```
summary(rip.bird.ca,scaling=1,axes=3)
```

outputs only the first three axes and uses scaling option 1, which scales the sample scores by the eigenvalues and leaves the species scores unscaled.

Note, in contrast to PCA, the structure correlations (i.e., linear correlations between the species, in this case, and the ordination axes) and final communality estimates are not meaningful because we are assuming a unimodal response function. In contrast to PCA, in CA there is little emphasis on trying to interpret the axes on the basis of which species load highly. Instead, we are interested in the relative positions of species in the ordination space and, reciprocally, the relative positions of samples in the ordination space.

Ordination plots:

Recall that there are lots of interesting options for displaying relationships in ordination plots, including:

- simple joint plots of samples and species
- displaying related samples (groups) on ordination plots
- fitting and displaying extrinsic variables on ordination plots

All of the plotting functions used in the PCA lab are applicable here and operate identically. Just substitute the CA object (`rip.bird.ca`) for the PCA object. For example,

```
ordiplot(rip.bird.ca)
```

produces a joint plot of the first two axes. CA has several conventions for scaling the sample and species scores. Recall that CA is a weighted averaging method (at least conceptually). In scaling type 1, the locations of the sample plots are calculated weighted averages of species scores; i.e., as the means of the species scores for species that occur in the plot, weighted by species abundance. Thus, species scores often lie outside the range of plot scores in the graph. In scaling type 2, species scores are calculated as weighted averages of samples scores; i.e., the means of the plot scores in which the species occurs, weighted by its abundance in each plot. In this case, sample plots often lie outside the range of species scores. Note, when we take a weighted average, the range of averages shrinks from the original values. The shrinkage factor is equal to the eigenvalue of ca, which has a theoretical maximum of 1. Thus, in scaling type 1, the sample scores are scaled by the eigenvalues, whereas in scaling type 2, the species scores are scaled by the eigenvalues. In scaling type 3, the sample and species scores are scaled symmetrically; both are scaled by square root of eigenvalues

One thing we might want to look at more carefully is the effect of rare species (and samples with low very totals). CA is known to be sensitive to rare species and outlier plots, and you should be vigilant in screening for such high-leverage samples and species. Programs like DECORANA (Hill and Gaugh, 1980)(see below) have options to "downweight" rare species to minimize the problem of species as outliers, but CA does not have a similar option. One way to evaluate this is to plot the species symbol proportional to the total species abundance, as follows:

```
sp.total<-colSums(rip.bird)
ordiplot(rip.bird.ca,choices=c(1,2),type='none')
orditorp(rip.bird.ca,display='sites',col='blue')
points(rip.bird.ca,display='species',col='red',cex=100*sp.total/sum(sp.total))
```

The first line computes species (column) totals, the second line adds the frame for the ordination plot, the third line plots the sample points as either text or open blue circles, and the fourth line add species points as open red circles proportional in size to the relative abundance of each species. Note, the 100 is a scaling factor that you will likely need to adjust up or down to suite your data set (via trial and error).

Note whether the rare species (with small circles) lie predominantly on the outside of the plot; i.e., the periphery of the sample points. If so, you may want to try deleting these species and evaluating the sensitivity of the ordination to these points. These can easily be done in a few steps, as follows:

```
temp<-rip.bird>0
sp.pres<-apply(temp,2,sum)
rip.bird.ca2<-cca(rip.bird[,sp.pres>=10])
ordiplot(rip.bird.ca2,choices=c(1,2),type='none')
orditorp(rip.bird.ca2,display='sites',col='blue',pch=19)
```

```
orditorp(rip.bird.ca2,display='species',col='red',pch=19)
```

This calculates a CA using only species that occur 10 or more times. The first line stores a logical (TRUE or FALSE) result for the expression “is abundance >0”. The second line creates a vector of frequency of occurrence for each species by summing the TRUE’s (equal to 1’s) in each column (species). The third line calls the `cca()` function, but uses subscripts to select only species where frequency of occurrence (`sp.pres`) is ≥ 10 .

Detrended Correspondence Analysis (DCA):

The execution of DCA and the examination of the results are almost identical to those described already for CA and PCA, and we will not repeat the procedures again here. Briefly,

```
rip.bird.dca<-decorana(rip.bird) – detrended correspondence analysis
rip.bird.dca – print results
rip.bird.dca<-decorana(rip.bird,iweigh=1) – downweighting rare species
rip.bird.dca – print new results
rip.bird.dca$evals – to print eigenvalues
rip.bird.dca$rproj – to print sample scores
rip.bird.dca$cproj – to print species scores
summary(rip.bird.dca,axes=3) – summary of eigenvalues and scaled scores
ordiplot(rip.bird.dca) – ordination plot
and all the other usual enhanced ordination plots and overlays.
```

Nonmetric Multidimensional Scaling (NMDS):

The execution of NMDS is slightly different than PCA/CA/DCA. To begin, ideally we would like to have some a priori knowledge of the underlying dimensionality of the data set. When you calculate the NMDS you can specify the number of dimensions you want. In contrast to PCO, however, the first 2 dimensions of a 3 dimensional NMDS are not the same as a 2 dimensional NMDS. Remember, it's trying to minimize stress, and it will take advantage of however many dimensions you give it, and it's not a geometric projection. If we assume a final dimensionality of 3, then we can call the `metaMDS()` function as follows:

```
rip.bird.nmds3<-metaMDS(rip.bird,distance='bray',k=3,trymax=50,autotransform=FALSE)
```

Note, if the data set is large, the computations can take a long time. The `metaMDS()` function has lots of arguments, so see the help files for more options. Basically, the arguments used above include a required data frame object (`rip.bird`), a distance measure (‘bray’ by default), a dimensionality (2 by default), the maximum number of random starts in search of a stable solution, and a choice not to automatically conduct a square root transformation and a Wisconsin double standardization (see `data.stand` for more details) on the data if certain thresholds are exceeded.

Alternatively, if we have no idea what the final dimensionality should be, we can be guided by a scree plot of stress versus the number of dimensions. The scree plot can take a very long time to generate if the data set is large, so choose your time to execute this function carefully:

```
ordi.scree(rip.bird,distance='bray',k=6,trymax=50,autotransform=FALSE)
```

This function basically calls the metaMDS function as before, but this time it calls it once for each number of dimensions and then plots the final stress value against the number of dimensions.

Once the final number of dimensions has been decided upon, a Monte Carlo randomization test of the final stress value can be conducted as follows:

```
nmds.monte(rip.bird,k=3,distance='bray',trymax=50,autotransform=FALSE)
```

How good a job does NMDS do? Well, one way to determine this is to look at the correlation between the calculated dissimilarities and the plotted values (after all, that's what it's trying to maximize). Specifically, we can plot the relationship between original dissimilarities and Euclidean distances in the ordination species using the stressplot() function:

```
stressplot(rip.bird.nmds3)
```

Function stressplot draws so-called Shepard plot where ordination distances are plotted against community dissimilarities and the fit is shown as monotone step function. In addition, stressplot shows two correlation like statistics of goodness of fit. The correlation based on stress simply is $R^2 = 1 - S^2$. The “fit-based R^2 ” is the correlation between the fitted values (\hat{d}) and ordination distances \tilde{d} , or between the step line and the points. This should be linear even when the fit is strongly curved and is often known as the “linear fit”. These two correlations are both based on the residuals in the Shepard plot, but they differ in the null model. In linear fit, the null model is that all ordination distances are equal, and the fit is a flat horizontal line. This sounds sensible, but you need $N - 1$ dimensions for the null model of N points, and the null model is geometrically impossible in the ordination space. The basic stress uses the null model where all observations are put in the same point, which is geometrically possible.

We might expect some sample points to be poorly fit by the ordination and it might be useful to identify them. The goodness() function measures the goodness-of-fit for individual samples and we can use this information to scale the points in the ordination plot:

```
rip.bird.nmds3.gof<-goodness(rip.bird.nmds3)
rip.bird.nmds3.gof
p<-ordiplot(rip.bird.nmds3,display='sites',type='n')
points(rip.bird.nmds3,display='sites',pch=19,col='blue',cex=rip.bird.nmds3.gof)
title(main='Oregon Streamside Birds NMDS: larger circles indicate poorer fit')
identify(p,'sites')
```

NMDS produces sample scores which are the coordinates of the samples in the k-dimensional ordination space, and these are stored in a the result object list in the component named 'points'. To see the sample scores, type:

```
rip.bird.nmds3$points
```

Similarly, species scores can be calculated as weighted averages of the samples, and they are stored in the component named 'species', accessed by typing:

```
rip.bird.nmds3$species
```

Lastly, all of the standard plotting functions used previously to view and enhance ordination plots and to plot fitted vectors and surfaces of extrinsic variables are applicable here. We won't repeat the procedures again here.

Well, I don't know about you, but I am exhausted! Let's call it quits even though there is so much more that we could do. If you are an over-achiever and are really loving this stuff, try out the `procrustes()` and `protest()` functions to compare the results of different ordinations.