**Non-metric Multidimensional Scaling**

Non-metric multidimensional scaling (NMDS) is one of the most flexible forms of community analysis. It is a form of ordination that capably handles data that are not normally distributed, and it does not require that there be a linear relationship among variables. It also works well if there are multiple, simultaneous gradients of variance. Until recently, NMDS was not done much because it took greater computational power than was available in most settings. Now, however, NMDS is one of the most commonly used forms of ordination.

NMDS is unique among ordination techniques because a small number (N) of ordination axes are chosen prior to analysis and then data are then fitted to those axes. A matrix of object distances (dissimilarities) is calculated first (using any distance metric of your choosing), and then ranks of the distances among all objects are calculated. The original distances between pairs of sample units (based on the original distance matrix you used) are ranked, and this rank order becomes the target rank order. For a given number of axes, which you specify, sampling units are then randomly assigned ordination scores (i.e., they are mapped onto axes). A Euclidean distance matrix is then built based on these new ordination scores. The NMDS algorithm then finds a configuration of objects in the chosen N-dimensional ordination space that best matches differences in the ranks. (NMDS uses only rank orders of intersample dissimilarities and not their magnitudes.) The relationship between the original distances and the new distances should be monotonic (i.e., increasing ordination distance = increasing observed distance); if it isn’t, then the ordination scores must be adjusted until monotonicity is achieved. A **stress** parameter is computed to measure the lack of fit between object distances in the ordination space and the calculated dissimilarities among objects: this indicates how much the ordination scores have to be adjusted to achieve monotonicity. Because the ordination scores have now been adjusted (i.e., they have been remapped), a new Euclidean distance matrix is calculated from them, the sampling units are ranked again, and the relationship between the two sets of distances is again evaluated for monotonicity. The NMDS algorithm iteratively repositions the objects in the ordination space to minimize stress. This process is continued until the improvement in monotonicity is marginal and stress has reached its lowest level. The resulting ordination axes have no particular order of importance (unlike PCA, where the first axis represents the greatest variance in the data, followed by PCA2, etc. in decreasing order of importance); they simply represent the best possible mapping of positions. They do not indicate the percentage of variance explained by each axis (unlike PCA). And the axes do not have a direct, interpretable relationship to the response variables; they represent a best fit of the patterns of redundancy in the original dataset. This is because NMDS is not an eigenvector-based gradient analysis technique; instead, it is a mapping method to represent ranks of pairwise dissimilarities among objects. Thus, its ordination axes do not correspond to a particular gradient in the original data.

Put another way, NMDS is an iterative search for the best positions of *n* entities on *k* dimensions (axes) that minimizes the stress of the *k*-dimensional configuration. The iterative calculations are based on an *n* x *n* distance matrix calculated from the *n* x *p* data matrix (where *n* is the number of rows [entities, usually sites or sampling units] and *p* is the number of columns [attributes, usually environmental variables] in the data matrix). Stress is defined as 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. (Stress is a measure of departure of monotonicity in the relationship between the distance in the original *p*-dimensional space and distance in the reduced *k*-dimensional ordination space.)

Axis 1

Axis

2

Site 1

Site 3

Site 5

Site 9

Site 6

Site 8

Site 7

Site 4

Site 2

Unlike other forms of ordination, each set of NMDS axes does not correspond to a particular gradient in the original data. Instead, NMDS represents ranks of pairwise dissimilarities among objects. NMDS 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. To do so, NMDS uses an iterative algorithm where initial estimates of the positions of samples are adjusted to minimize the stress until further iterations do 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** (i.e., that is, solutions that are not the best solution but that are better than all nearby solutions). Because the initial sample unit ordination scores are randomly assigned, every time you run an NMDS (using a different random number seed to start the process), you will get a different solution. Thus, NMDS is not an analytical process that produces a unique solution. (**This is seen as NMDS’s chief weakness.**) This seems especially common with high-dimensional solutions (*k* > 4). The main approach to minimize this problem is to try multiple random starts (with different random number seeds) and to keep the best result (lowest stress). You are never guaranteed that you have found the single best solution, but if the majority of your results achieve similarly low levels of stress, then you can be reasonably certain that you have at least found a good solution.

**NMDS makes few data assumptions**. (In contrast, for example, PCA assumes linear relationships.) In addition, it **allows the use of any distance/dissimilarity measure of the samples**, unlike other methods that require specific distance measures (e.g. Euclidean in PCA, Chi-square in Correspondence Analysis). If your data have multiple gradients of variance, then NMDS is probably your best bet in representing patterns. In addition, NMDS is not affected strongly by outliers, irregularly spaced observations along the underlying gradient, or a moderate level of noise in the data. These factors make NMDS well-suited for a wide variety of applications in ecology.

Be advised, however, that **NMDS is computationally demanding, especially for large datasets. (This is NMDS’s other main weakness**, but one that is becoming less problematic with increasing computational capability.)

Stress tends to increase with sample size and the number of response variables.

There are several packages that do NMDS, such as metaMDS() in *vegan*, isoMDS() in *MASS*, and nmds()in *labdsv*. We will explore all three today, starting with nmds()in *labdsv*.

**Examples:**

**Calculating an NMDS:**

Open a new RStudio session (with your class working directory) with the following libraries:

*labdsv*

*MASS*

*MVA*

*optpart*

*picante*

*stats*

*vegan*

Read in the Bryce Canyon vegetation data:

veg <- read.table("bryceveg.R", header=TRUE)

Now you need to **construct a dissimilarity (distance) matrix. To decide which method to use to do so**, you normally would go through a process like the following (refresh your memory from the “Ordination” lesson if needed):

-it is a homogeneous dataset 🡪 that eliminates Gower

-data aren’t counts 🡪 that eliminates Steinhaus

-there are lots of 0’s 🡪 that means that Euclidean, Manhattan, Canberra aren’t the best choices

-Mahalanobis isn’t needed here

-Chi-square de-emphasizes abundant species, of which there are some in this dataset, so that’s not the best choice

Therefore, Bray-Curtis or Hellenger might be good choices here. Bray-Curtis is more flexible, so that’s what we’ll use.

NMDS works on a dissimilarity/distance matrix as input, so create a dissimilarity/distance matrix using the Bray-Curtis metric:

dis.bc <- vegdist(veg, method="bray")

To calculate a NMDS, use the nmds()function on that dissimilarity/distance matrix:

bc.nmds <- nmds(dis.bc)

bc.nmds

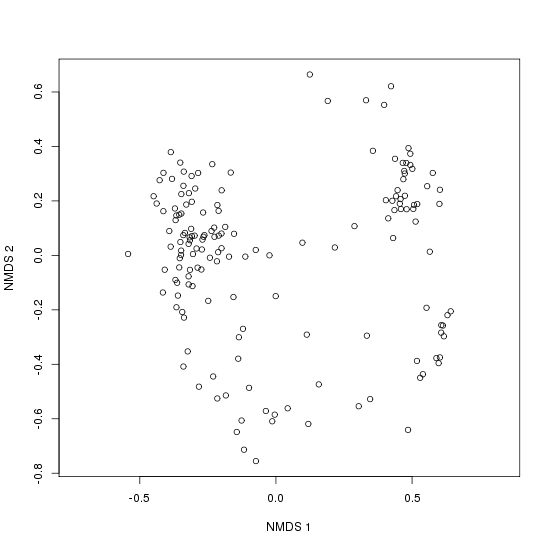
As you read through the output, you’ll find the stress value when the algorithm converges, in this case at 19.27838:

$stress

[1] 19.27838

Is this a low value or stress, or a high one? Stress increases both with the number of samples and the number of variables in your dataset. Thus, a larger dataset will necessarily result in a higher stress value, so use caution when comparing stress among datasets. In order to determine whether this value of stress is high for THIS dataset, we must compare this value to stress values from other ordination configurations (i.e., different numbers of axes). First, plot the outcome:

plot(bc.nmds)



Unlike in PCA, the axes in NMDS cannot be interpreted like axis 1 explains the greatest amount of variance, axis 2 explains the next greatest amount of variance, etc.

When you calculate the NMDS you specify the number of dimensions (axes) you want. (The default is 2.) Be advised, however, the first 2 dimensions of a 3-dimensional NMDS are not the same as a 2-dimensional NMDS. Remember, it's not a geometric projection; it's trying to minimize stress, and it will take advantage of however many dimensions you give it. The ordination is sensitive to the number of dimensions that is chosen, so this must be made with care. Choosing too few dimensions will force multiple axes of variation to be expressed on a single ordination dimension. Choosing too many dimensions can cause a single source of variation to be expressed on more than one dimension. One way to choose an appropriate number of dimensions is perform ordinations of progressively higher numbers of dimensions, and then compare them.

**To select an appropriate number of dimensions (axes):**

The number of dimensions *k* is a crucial parameter in NMDS. Unlike other forms of ordination, as dimensions are added, the configuration of the other dimensions changes. However, even the people who developed the NMDS technique indicated that there is no firm statistical criterion for selecting an appropriate number of dimensions. An appropriate number of dimensions should be determined by tabulating final stress vs. number of dimensions. Choose a number of axes beyond which reductions in stress are small (<5%). Sometimes this is easily detected by plotting stress (Y) vs. dimensions (X) in what’s called a **scree plot**, and seeing where the resulting pattern shows an “elbow.” For example:

Dimensions

Stress

1

6

5

4

3

2

0

50

40

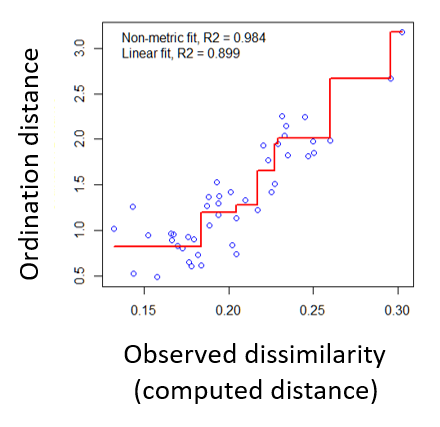
30

20

10

In this scree plot, the first two axes provide greater reductions in stress than other dimensions, so we can conclude that a 2-dimensional solution is best for this dataset.

Most functions make a different form of plot, called a **stress plot** (or Shepard plot):



This plot shows scatter around the regression between the interpoint distances in the final ordination against their original dissimilarities. Large scatter around the regression line indicates that the reduced number of dimensions from the ordination does not do a good job of representing the original dissimilarities of the data.

To make a long story short, you need to examine multiple dimensions to find the optimal number of reduced dimensions. To specify more dimensions than the default of 2, simply specify the desired dimension as the second argument, e.g. for 4 dimensions:

bc4d.nmds <- nmds(dis.bc, 4)

bc4d.nmds

$stress

[1] 10.43317

This value is lower than the default 2 dimensions we had before. Stress usually decreases with an increasing number of dimensions, but keeping the number of dimensions < 4 is advised to minimize the possibility of **overfitting**.

At this point, to settle on an suitable number of axes (dimensions) based on low stress, you need to know whether a given value of stress is a “good” one (low) or not.

**What are “good” (i.e., low) values of stress?**

Stress values are typically reported as values between 0-100 (with the reported stress values having been multiplied by 100 to obtain a %). So our value of 10.43317 means 10.43317%. That seems low based on a scale of 0-100, but what is a “good” (i.e., truly low) value of stress? Here are some rules of thumb for interpreting stress from Kruskal (1964):

< 2.5 Excellent

2.5 to 5 Good

5 to 10 Fair

> 20 Poor

and from Clarke (1993):

< 5 Excellent, but rarely achieved

5 to 10 Good

10 to 20 Flawed

< 15 Acceptable

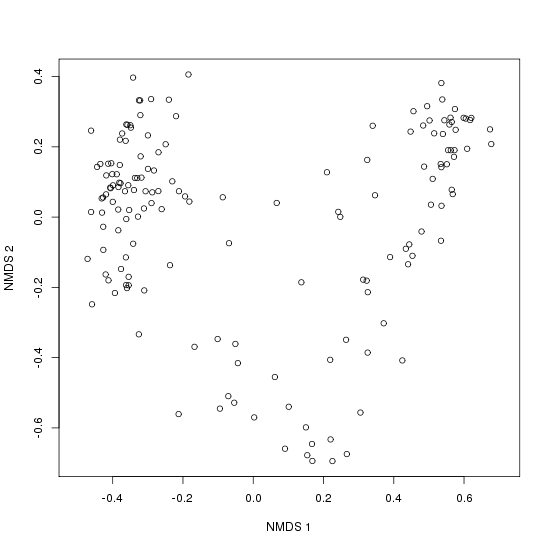
> 20 Poor

Zuur et al. (2007) consider values below 0.1 (i.e., 10) to indicate a good to excellent ability to resolve patterns in your data.

So even though our value of 10.43317 was low, relatively speaking, it would be seen as fair to poor by Kruskal (1964) and flawed by Clarke (1993). That may seem disheartening, but most “good” (i.e., tidy, interpretable) ecological community datasets have solutions with stress between 10 and 20 (McCune and Grace 2002). And remember that these are some general rules of thumb, but they are only that rather than fixed rules.

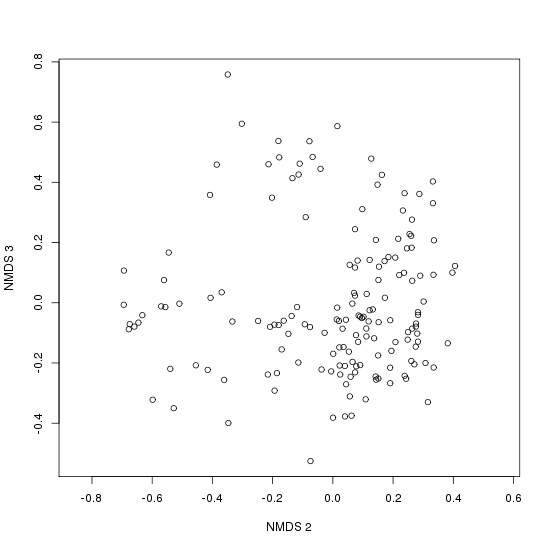
So we’ll keep our 4-dimensional model. Now plot it:

plot(bc4d.nmds)



To plot dimensions higher than 1 and 2 (the default), simply include the dimension number in the plot() function (although the plot will still be planar; recall that NMDS dimensions are not geometric dimensions):

plot(bc4d.nmds,2,3)



Having many dimensions can compromise interpretability of the results: variation will be spread over all of the axes, defeating the primary purpose of ordination, which is to express variation in as few dimensions as needed to represent the covariation among as many attributes as possible. **Overfitting** occurs when you use a dimensionality that is too high for the number of sampling units, and should be avoided.

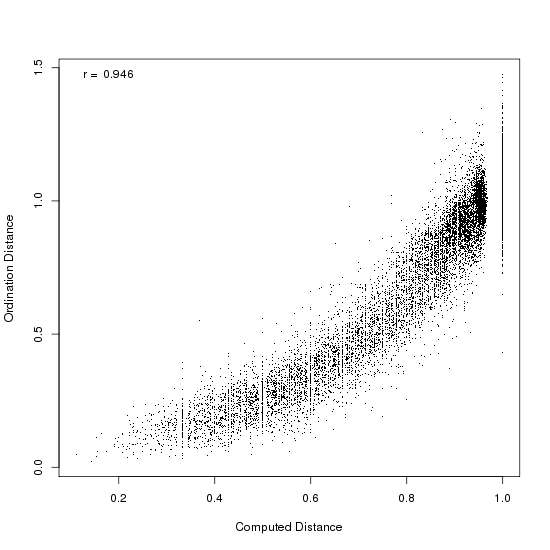
Most ordination methods are analytical and therefore result in a single unique solution to a set of data. In contrast, NMDS iteratively seeks a solution and stops when an acceptable solution has been found or after some pre-specified number of attempts. As a result, an NMDS ordination is not a unique solution; a subsequent NMDS of the same data will likely result in a somewhat different ordination.

**How to test whether your NMDS is doing better than random chance:**

A Monte Carlo (i.e., a randomization) test can be used to evaluate whether the NMDS is extracting stronger axes than expected by chance. Be advised that this can be very time-consuming, even for as few as only 20 runs, if your dataset is large. You compare the stress obtained using your data with the stress from multiple runs of randomized versions of your data. (The data are randomly shuffled within columns after each run.) You can assess the significance (p-value) of this difference where if *A* is the number of randomized runs with final stress < observed minimum stress and *B* is the number of randomized runs, then p-value = (1 + *A*)/(1 + *B*). Here’s how we will perform this procedure.

We start by examining the correlation between the calculated dissimilarities and the plotted values (after all, that's what it's trying to maximize). We can do that with the *labdsv* ordcomp()function and generating a stress plot:

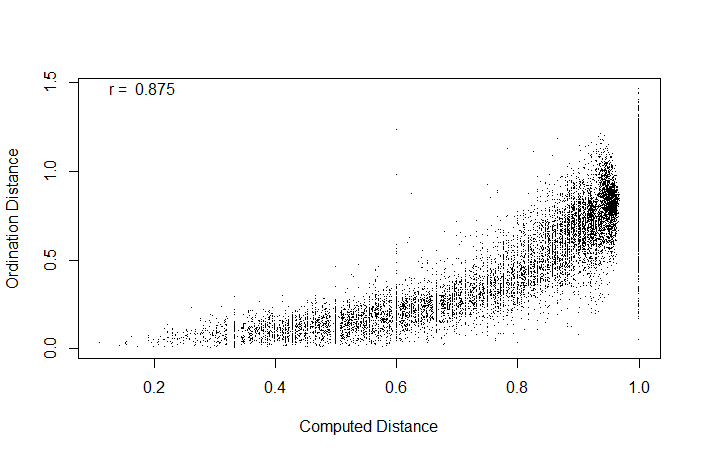
ordcomp(bc4d.nmds,dis.bc,dim=4)



The dots are plotted for each pair of sample plots, and the correlation is printed in the upper left corner (r = 0.946, very strong!). It's not quite linear, but remember, it's not a projection. In only two dimensions, it's

ordcomp(bc.nmds,dis.bc)

r = 0.875, which is still pretty darn good:



(These are the types of stress plots produced for products of *labdsv*’s nmds(); later today you’ll make stress plots for *vegan*’s metaMDS(), which are not Monte Carlo-cased but rather take a regression approach.)

The NMDS algorithm starts with a random configuration; to pick the best set from random starts, you can use the bestnmds()function in *labdsv*:

test <- bestnmds(dis.bc,k=4,itr=10)

[1] 10.43317 27.04523 27.08192 27.06957 11.01079 27.03676 27.05923

[8] 27.05944 27.05560 27.06456

best result = 1

with stress = 10.43317

The results are stress values based on iterations (10 was specified in the itr=10 argument).

The default number of iterations for bestnmds() is 50, but random starts often take a while to settle down, so bumping up this number is advised. A larger number also helps the ordination from settling into a local minimum, but the tradeoff is that it slows down processing time.

The best result of the set is saved into whatever appears on the left side of the assignment arrow, in this case the R object I named test.

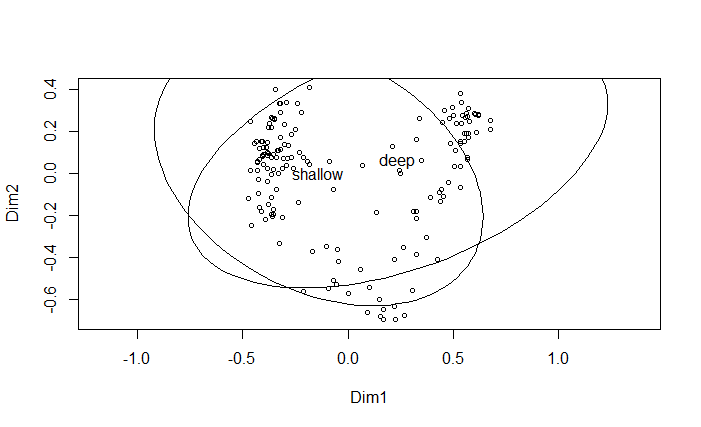
**Putting the NMDS model to use:**

Once you have a model that is reasonable, you can then use the axes as dependent variables and environmental variables as independent variables. For example, to then examine effect of soil depth from the site x environment dataset brycesite.R:

site <- read.table("brycesite.R", header=TRUE)

ordiplot(bc4d.nmds, display = "sites", type = "points")

ordiellipse(bc4d.nmds, site$depth, conf = 0.95, label = TRUE)



This was an example I used in the “Introduction to Ordination” lesson, and in the video for that lesson I explained that NMDS wasn’t likely an appropriate choice for this specific analysis (examining whether soil depth could explain variation in the site x species data) because depth is a categorical variable and thus not on a continuous gradient. The overlap in the ellipses indicates either that soil depth is not a factor that accounts for differences in plants by sites…OR that NMDS isn’t an appropriate choice for this type of analysis.

This is an important point: just because you can get an analysis to work and can produce a graph, that doesn’t mean that it is an appropriate analysis for your data or questions.

If you want to apply your NMDS with respect to some environmental predictor (independent) variable, and IF that variable is numeric rather than categorical, then you can make use of the rankindex() command in *vegan* to determine which of five distance metrics (Euclidean, Manhattan, Gower, Bray-Curtis, and Kulczynski) is best for your data. For example, if we wanted to examine whether variation in species distributions at Bryce Canyon was a function of elevation, we could use the following:

rankindex(site$elev, veg)

euc man gow bra kul

0.11753141 0.13300762 0.06486742 0.36723163 0.37047115

The highest value is considered the best, which in this case is Kulczynski. This is a metric I spent no time on in the “Introduction to Ordination” lesson; it is an alternative to the Jaccard metric and likewise is used with binary data. I have seldom seen it used. The Bray-Curtis solution is nearly as high, however, and so is preferable. We will use it in a demo of *vegan*’s metaMDS(). This function automatically transforms your data if needed and checks solution robustness:

bryce.nmds <- metaMDS(veg, distance = "bray")

The output is a list of stress values from iterations (default is 20 but can be changed by adding trymax= ). You can examine this list to see the best stress value obtained. However, since NMDS starts with a random seeding, you can run that same line of code multiple times and will see different best values! And sometimes you’ll see \*\*\* No convergence at the end of the list, indicating that no stable solution was found. (See more about this in next section below.) In my example, I ran the code, the best stress value was 0.1405363, and a solution was reached. For the stress value, metaMDS() reports stress as a proportion, so you have to multiply the result by 100 to get a value comparable to the rules of thumb we covered earlier. The value of 14.0 would be fair to poor by Kruskal (1964) and flawed according to Clarke (1993), but remember that most ecological community datasets have stress values of 10-20 (McCune and Grace 2002), so this is workable.

Now we need to determine how many dimensions is best, so I’m going to ask for stress values for 2-5 dimensions:

for(i in 2:5) print(metaMDS(veg, distance="bray", k=i, trace=FALSE)$stress\*100)

[1] 14.05384

[1] 10.35248

[1] 8.22879

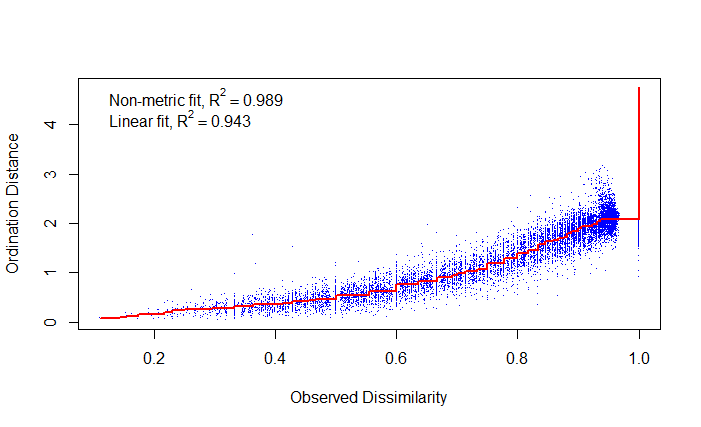
[1] 6.933714

As was expected, the more dimensions there are, the lower the stress (but the tradeoff is the danger of overfitting). All of the values of stress are acceptable. In this case, the biggest drop was from k=2 (top line) to k=3 (2nd line), so k=3 is the best and most parsimonious solution, so recalculate the NMDS with k=3:

bryce.nmds <- metaMDS(veg, distance = "bray", k=3)

The result is a solution with lower stress (0.103535 = 10.35% in my example). Make a stress plot to examine its goodness of fit:

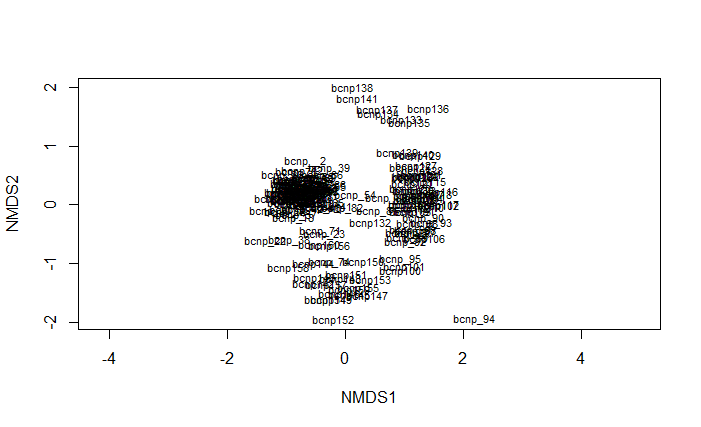
stressplot(bryce.nmds)



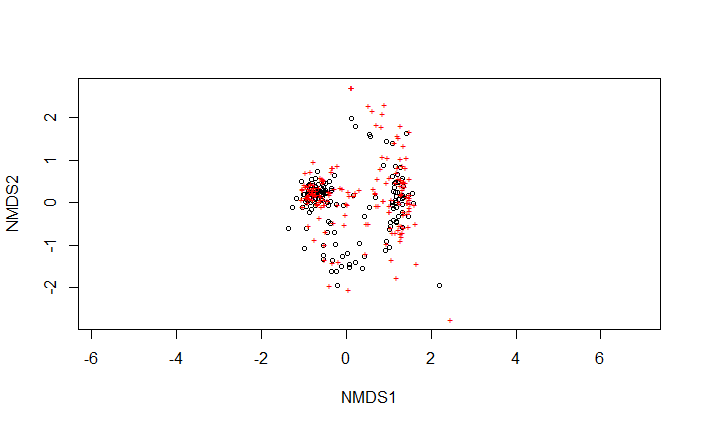
The red line is the regression line, and the cloud of points around it is pretty good. The R2 values also indicate good fit for our k=3 model.

Now let’s plot the result of the NMDS, using regular plotting as well as metaMDS’s own plot functions:

plot(bryce.nmds, type="t", display="sites")



op = plot(bryce.nmds, type="p", display=c("site", "species"))



Notice how there is a site that is a sort of outlier in the lower right, and a site that is like that, too. You can label the diagram like so:

orditorp(bryce.nmds, display="sites", col="black", cex=0.55)

orditorp(bryce.nmds, display="species", col="red", cex=0.55)

but the result is really “busy”! (There were 160 sites and 169 species, after all.)

**What does it mean if your NMDS cannot converge on a stable solution?**

The stability of the solution can be examined with a plot of stress (Y) vs. iteration number (X). This plot normally shows a decline before settling on an asymptote. Sometimes, however, no such stability will be found. This can occur by overfitting. You can sometimes fix this problem by increasing the number of iterations (e.g. by altering the trymax= statement in metaMDS) or by transforming the data and re-doing the whole procedure.

**What can cause problems with your NMDS (and what can you do about them)?**

-outliers (solution: you should have detected them and culled them before starting analysis)

-data dominated by a single super-abundant species (solution: nothing perfect; you can conduct analyses with and without that species and compare the output)

-very small datasets (< 10 samples) create randomizations that are too conservative and that produce configurations for which NMDS finds solutions with zero stress (solution: do not do the randomization test, and interpret your results very cautiously)

-data with many zeros (solution: you can try doing randomizations with a VERY large number of random starts, but even that may not perform well)

**Summary of the NMDS procedure:**

-Do multiple runs with your real data, each with a random starting configuration. A “run” consists of a series of solutions, going in descending order of dimensionality (from highest number of axes down to one axis). Accumulate statistics on the final stress for each dimensionality.

-Do multiple runs with randomized data. Before each run, the data from the main matrix should be shuffled within columns. Each run will use a different random starting configuration. Accumulate statistics on the final stress for each dimensionality.

-Choose the best solutions, as defined by a particular starting configuration and number of dimensions: select the best solution for each dimensionality as that with the lowest final stress score from a real run.

-Select the dimensionality by comparing the final stress values among the best solutions (one single best for each dimensionality). Additional dimensions are considered useful if the reduce the final stress by at least 5%. Select the highest dimensionality that meets this criterion. At that dimensionality, the final stress must be lower than that for 95% of the randomized runs (i.e., p-value < 0.05 for the Monte Carlo test). If this criterion is not met, choose the next lowest dimensionality that does so.

Thus, the basic principles behind NMDS are: select an appropriate number of dimensions (axes), seek low stress, use a Monte Carlo randomization procedure to compare real data to a null model, and avoid unstable solutions.

**Another example:**

The dataset hsere.csv has abundances of plants collected on 10 transects. These transects represent a gradient of pH; those data are in hsere\_ph.csv. Read in the data:

hsere <- read.csv("hsere.csv", header=TRUE, row.names=1)

Notice that this spreadsheet has the transects (sites) as columns and the species as rows, which is not R’s preferred way of dealing with things. So transpose the data:

thsere <- t(hsere)

Now read in the pH data:

hsereph <- read.csv("hsere\_ph.csv", header=TRUE, row.names=1)

They also need to be transposed:

thsereph <- t(hsereph)

Now you need to make a dissimilarity (distance) matrix on the abundance data.

To decide which method to use, examine the data:

-it is a homogeneous data matrix consisting of integers 🡪 that eliminates Gower

-there are lots of 0’s 🡪 that means that Euclidean, Manhattan, Canberra aren’t the best choices

-Mahalanobis isn’t needed here

-Chi-square de-emphasizes abundant species, of which there are some in this dataset, so that’s not the best choice

Therefore, Bray-Curtis, Hellenger, or Steinhaus might be good choices here. Bray-Curtis is the default method in *vegan* because it’s so flexible and appropriate for so many datasets.

#Make a Bray-Curtis distance matrix and do NMDS:

thsere.bc <- vegdist(thsere, method="bray")

This time let’s use isoMDS() from *MASS*:

thsere.nmds <- isoMDS(thsere.bc)

thsere.nmds

The resulting stress value is super-low, excellent! So you now may think, let me plot this with plot(thsere.nmds)…but that results in an error! The output from isoMDS() is a little different than what we did before and needs to be plotted differently:

This sets up the plot frame:

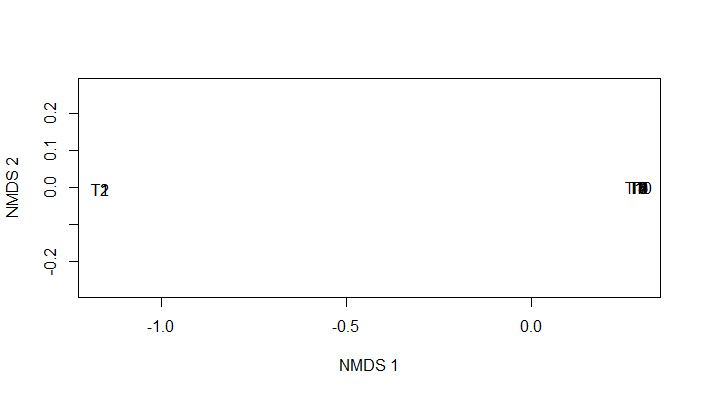
plot(thsere.nmds$points, type="n", xlab="NMDS 1", ylab="NMDS 2", asp=1)

(The asp=1 argument keeps a 1:1 aspect ratio.)

And this populates the frame with the transect names:

text(thsere.nmds$points, labels=rownames(thsere.nmds$points))

But the result is hideous:



There is separation of sites, but only on one axis. So this ordination is not informative even though it had a very low stress value! (That low stress is likely because only one axis was involved.)

In situations like this, the distance matrix is usually the culprit. Even though we went through a logical selection process, this ordination isn’t informative. Therefore, let’s default to the second most commonly used distance metric, Euclidean:

thsere.eu <- vegdist(thsere, method="euclidean")

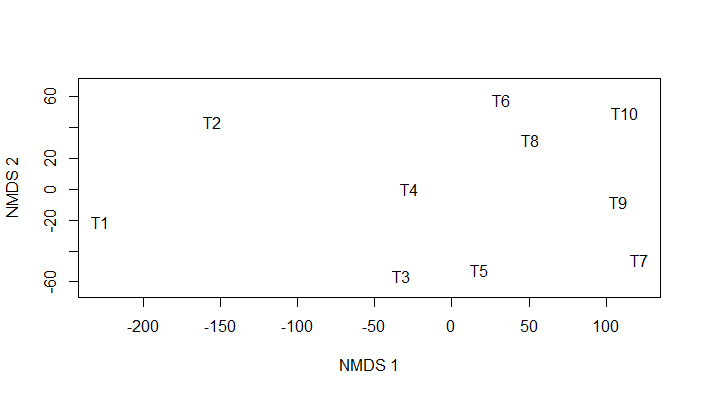
thsere2.nmds <- isoMDS(thsere.eu)

thsere2.nmds

The stress value is higher than before but is still excellent. Now plot:

plot(thsere2.nmds$points, type="n", xlab="NMDS 1", ylab="NMDS 2", asp=1)

text(thsere2.nmds$points, labels=rownames(thsere2.nmds$points))



What a difference!

Notice the difference in scales between axis 1 and 2, so axis 1 is still the main driver. So let’s now do a (nonparametric = Spearman) correlation between axis 1 and pH:

cor(scores(thsere2.nmds)[,1],thsereph, method="spearman")

The result (-0.9272727) indicates there is a very strong negative association, which makes sense when you look back at hsere\_ph.csv: the lowest pH values were found in transects 8, 9, and 10 and the highest values were in T1 and T2.

Look at how the stress changes as you add more dimensions:

for(i in 1:1) print(isoMDS(thsere.eu, k=1, trace=FALSE)$stress)

for(i in 1:1) print(isoMDS(thsere.eu, k=2, trace=FALSE)$stress)

for(i in 1:1) print(isoMDS(thsere.eu, k=3, trace=FALSE)$stress)

for(i in 1:1) print(isoMDS(thsere.eu, k=4, trace=FALSE)$stress)

You can see a big drop in stress from k = 1 to k = 2, and the values get better (as expected) with increasing values of k. The stress values of k > 2 are all good to excellent.

**References:**

Clarke, K.R. 1993. Non-parametric multivariate analyses of changes in community structure. Australian Journal of Ecology 18:117-143.

Kruskal, J.B. 1964. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. Psychometrika 29:1-27.

McCune, B., and J.B. Grace. 2002. *Analysis of Ecological Communities*. MJM, Gleneden Beach, OR.

Zuur, A.F., E.N. Ieno, and G.M. Smith. 2007. *Analysing Ecological Data*. Springer, New York, NY.

**Assignment:** due 0800 Monday, April 12

Start a fresh RStudio session. Remember to set your working directory to your course folder and use the same package libraries as we used today.

Read in the grassland.community.csv site x species file from the course website as an object named comm (with header = TRUE, row.names = 1) and its associated site x environment file (plot.metadata.csv) as an object named metadata (with header = TRUE, row.names = 1).

We’ll use these data to explore the metaMDS() function from *vegan*. Recall that this function automatically transforms your data if needed and checks solution robustness:

comm.bc.mds <- metaMDS(comm, dist = "bray")

**Q1. What transformations/standardizations (if any) did metaMDS() do on the data?**

**Q2. What was the stress? Is that value good? (Justify your answer.)**

**Q3. Determine the optimal number of dimensions and then re-do the NMDS with that number of dimensions.**

**Q4. What is the stress of this new NMDS model? (It should be better than before.)**

**Q5. Now assess goodness of ordination fit with a stress plot.**

Now plot site scores as text:

ordiplot(comm.bc.mds, display = "sites", type = "text")

Notice how fescue and mixedgrass habitats separate along the first NMDS axis (the “X-axis”).

What about species by habitats?

ordipointlabel(comm.bc.mds)

What a mess! We can do better. First, start with a clean template:

mds.fig <- ordiplot(comm.bc.mds, type = "none")

Now plot just the samples, colored by habitat; pch=19 means circle symbols:

points(mds.fig, "sites", pch = 19, col = "green", select = metadata$habitat == "Fescue")

points(mds.fig, "sites", pch = 19, col = "blue", select = metadata$habitat == "Mixedgrass")

Add confidence ellipses around habitat types:

ordiellipse(comm.bc.mds, metadata$habitat, conf = 0.95, label = TRUE)

**Q6. Interpret the output.**

Make an RMarkdown Word file of your work and turn that in. Be sure to include your answers to the questions asked! Turn in your assignment as a Word document via email to [iroro.tanshi@ttu.edu](mailto:iroro.tanshi@ttu.edu) no later than 8:00 a.m. on Monday of next week. In your email, please include the following as the Subject line:

Assignment on NMDS