Using Optimal Classification for multidimensional scaling analysis of linguistic data

William Croft and Jason Timm, *University of New Mexico*

Version of 11/23/2013

1. The use of multidimensional scaling in linguistic analysis

What is multidimensional scaling?

Multidimensional scaling (MDS) is a technique for visualizing the relationships among data that are similar to each other on very many dimensions. For example, meanings of words such as indefinite pronouns are similar to each other by virtue of being expressed by the same indefinite pronoun in one language or another (Haspelmath 1997). If one compares indefinite pronouns of a large number of languages, the meanings they express are similar to each other on a very large number of "dimensions", namely all the different indefinite pronouns of all the languages. MDS reduces the large number of dimensions of similarity to a small number—typically one or two dimensions—which can then be visually displayed and interpreted by the analyst.

The example of indefinite pronouns illustrates one use of MDS in linguistic analysis: the analysis of complex patterns of crosslinguistic comparison. MDS is a mathematical technique which has been computationally implemented, and so can handle large and complex datasets that are difficult to analyze by hand in the semantic map model which is currently used for typological analysis (Haspelmath 1997, 2003; but see Regier et al. 2013). A description of the use of MDS for this type of crosslinguistic analysis can be found in Croft and Poole (2008) and Croft (2010). A more general and mathematical discussion of MDS and the Optimal Classification algorithm (see below) can be found in Poole (2000, 2005).

What kind of linguistic data is MDS useful for?

We gave the example of using MDS for crosslinguistic analysis above. This is not the only type of linguistic data that MDS could be used for. However, for MDS to be useful, the data has to be of the following type:

•It must be interpretable as measuring degrees of similarity between data. In the crosslinguistic example, meanings are considered to be similar to each other to the degree that they are likely to be expressed by the same word, grammatical inflection or construction. This is fundamentally a fact about human categorization. MDS has been used for over half a century in psychology for analyzing human categorization, including linguistic categorization. MDS can be used for many types of linguistic categorization phenomena.

•The data must be of high dimensionality and of high variability. That is, the data points are interpreted as being similar or different along many different dimensions. If the data is similar only along one or two dimensions, then one can usually interpret it without

using a computer algorithm. If the data is similar among only a few dimensions, then it may all "clump" together in uninteresting ways. For example, if one is comparing 100 verbs in terms of how they occur in just four argument structure constructions, the four constructions probably won't divide up the verbs in a very fine-grained way (and the patterns may be perceivable without using MDS).

What is the Optimal Classification MDS algorithm? How does it work?

The Optimal Classification algorithm is used with data that is nominal, that is, in the form of a 'yes'/'no' answer. For example, meanings can be classified as to whether they are expressed by a particular linguistic form ('yes') or not ('no'). This type of data indirectly measures similarity by virtue of shared 'yes'/'not' answers among data points.

The usual algorithms for doing MDS are dissimilarity algorithms, that is, the input that they require is a square dissimilarity matrix of pairwise comparison of each pair of data points in the dataset, for example, comparing two meanings as to whether they are expressed by the same linguistic form or not. Direct measurements of linguistic similarity, e.g. in similarity judgement experiments, can be analyzed using dissimilarity MDS algorithms, such as SMACOF (also available in R). Direct measurements of similarity may be numerical rather than nominal.

Much linguistic data, such as categorization data or distributional data (of words in grammatical constructions, for example), is not in the form of pairwise similarity measurements, yet possesses theoretically significant similarity relations (e.g. similarity of categories or distributional patterns). Also, if the data is lopsided (e.g. a linguistic form is used for either a small number of meanings or for the vast majority of meanings), the dissimilarity matrix values are compressed and the analysis does not do a good job of discriminating the data points.

The Optimal Classification (OC) algorithm was developed by Keith Poole for analyzing parliamentary voting. As its name implies, OC optimizes correct classification of the data points, that is, it tries to make the model of each linguistic category as accurate as possible in terms of which data points the linguistic category includes (the "Yeas" in parliamentary parlance) and which data points the category excludes (the "Nays"). Equally important, OC is an unfolding algorithm, which directly models the categorization of individual data points (e.g. meanings), without having to construct pairwise comparisons of the categories and thus saving a step in coding nominal ('yes'/'no') data. OC does not suffer from the problems with lopsided data that dissimilarity algorithms have (see Croft 2010:7-9).

In order to understand mathematically exactly how the OC algorithm works, consult Poole (2000, 2005). In general terms, the goal of MDS is to give a spatial model of the data points (the data whose similarity you want to measure) in such a way that all the categories that you use for defining similarity "cut" the data points as accurately as possible, so that the data points that are in the category are on one side of the cutting point (in a one-dimensional model) or cutting line (in a two-dimensional model) for the category, and the data points that are excluded from the category are on the other side of the cutting point/line. The algorithm progressively approximates the best positioning of all the data points in the spatial model and the best placement of all the cutting

points/lines, until the optimally correct classification of the data by the categories is achieved.

How does MDS compare to other similar multivariate analyses?

MDS analyzes multidimensional data using an unsupervised distance method (Baayen 2008:118). It is unsupervised in the sense that one does not specify in advance the categories or groupings that the data will fall into. It contrasts with analyses such as classification trees and discriminant analysis in which the user specifies what categories the data are to be fit into (ibid., 148-63). MDS is most useful when one does not have an idea in advance of the clustering of the data.

MDS is a distance method in that it represents similarity directly, in terms of Euclidean distances in the spatial model it produces. MDS represents similarity in a spatial model of a low number of spatial dimensions that are specified by the user (one, two, or three dimensions, for example; how to choose the dimensionality of the model is discussed in §3). MDS captures all of the variance of the data in the specified number of dimensions.

MDS contrasts with eigenanalysis methods, such as principal components analysis (PCA), factor analysis and correspondence analysis (Baayen 2008:118-36; Woods et al. 1986, ch. 15). Principal components analysis and factor analysis are parametric, while correspondence analysis is nonparametric (and therefore most comparable to MDS). Eigenanalysis methods take a matrix of data of the same sort as used in MDS, and converts it to another matrix of the same dimensionality, but where the dimensions and their values in the matrix have the following properties: each dimension is uncorrelated with every other dimension; the dimensions have values such that the first dimension ('principal component') accounts for the most variance in the data, the second dimension accounts for the next most variance, and so on.

Eigenanalysis methods therefore differ from distance methods in several important ways that impact on how the results of these analyses are interpreted (Croft and Poole 2008:13-14). This is important to remember because MDS and PCA or correspondence analysis results are often displayed graphically in similar ways, that is, in a two-dimensional space. An MDS analysis in two dimensions and a PCA/correspondence analysis displayed in two dimensions are very different in interpretation. First, an MDS analysis in two dimensions represents all of the variance in the data (relative to the degree of fit of the spatial model to the data), whereas a PCA/correspondence analysis in two dimensions represents only the variance in the first two principal components; variance accounted for in the remaining principal components is not represented. Second, an MDS analysis in two dimensions is a true (Euclidean) spatial representation of the structure of the data. A PCA/correspondence analysis, in contrast, is a visual representation of the variance captured in the first two principal components.

One consequence of this second difference is how one interprets the geometry of the spatial representation. In MDS, the distances in the two-dimensional space directly represent degree of similarity in some qualitative dimension or dimensions. All distances in the spatial representation of similarity are interpretable. Therefore, the analysis is invariant under translation and rotation. In other words, in the interpretation of the two-dimensional display of the data, one can place the origin anywhere, and orient the axes of

the two qualitative dimensions in any way, depending on how the analyst interprets the data (see for example the orientation of the tense and aspect axes in the MDS analysis of the tense-aspect data in Croft and Poole 2008:26, Figure 8). In PCA/correspondence analysis, the two dimensions cannot be directly interpreted geometrically. Thus, the qualitative interpretation must respect the actual origin and axes of the two-dimensional representation of the first two principal components. In fact, one cannot even compare diagonal distances in a two-dimensional visual display of the first two dimensions of a correspondence analysis: 'only the distances between row points, and correspondingly, between column point are interpretable in this manner; the distances between row points and column points cannot be interpreted' (http://www.statsoft.com/textbook/correspondence-analysis/).

2. How to run the MDS OC R programs

The MDS OC R programs are slightly modified versions of the MDS OC R programs developed by Keith Poole for analyzing parliamentary voting data. Thus, the names of variables and outputs reflect that origin. For example, the data points whose similarity we are interested in measuring are 'legislators', and the linguistic properties used to measure the similarity of the data points are 'roll-calls'. The similarity or categorization of the data points by the linguistic properties is called 'Yea' (the data point falls in the category) or 'Nay' (the data point does not fall in the category).

Downloading and running R

R is an open-source programming language that is widely used for statistical applications. If you do not have R on your computer already, download the R programming language from http://www.r-project.org, and install it on your computer. Once you have installed it, launch R. A window called R Console will appear.

Mac: the first time you run something in R, it will ask you to install the X.Org X Window System from xquartz.org, another open-source effort. Install it.

PC: the X.Org X Window System is not required.

The format of the data input

The data input is in a matrix consisting of the data whose similarity you are interested in is presented in rows, and the linguistic factors which are interpreted as contributing to the measurement of similarity of the data is presented in columns. Thus, the data you have must be converted into that form. For example, in the study of indefinite pronouns above, the Romanian indefinite pronoun data may have originally been given as follows:

va-:	specific known, specific unknown, irrealis nonspecific, question, conditional
vre- -un:	question, conditional, indirect negation
ori-:	comparative, free choice
ni-:	indirect negation, direct negation

The data must be converted into something like the following two-dimensional matrix:

	va-	*vre- -un*	*ori-*	*ni-*
Specific known	1	6	6	6
Specific unknown	1	6	6	6
Irrealis nonspecific	1	6	6	6
Question	1	1	6	6
Conditional	1	1	6	6
Comparative	6	6	1	6
Free choice	6	6	1	6
Indirect negation	6	1	6	1
Direct negation	6	6	6	1

That is, for all meanings (whose similarity we want to measure), we have to specify for each linguistic form whether the form is used for the meaning or not, and present it in a two-dimensional matrix. Numerals must be used instead of 'Yes' or 'No'. Similarity is represented by numerical closeness, so 1 is 'Yes' and 6 is 'No'; missing data is coded as 9.

One relatively easy way to do this is to create the file in a spreadsheet program such as Microsoft Excel, since you can enter data and manipulate rows and columns easily in the spreadsheet. When you are ready, save the file as a tab-delimited text file. Make sure there are no special characters (i.e. characters that are not ASCII characters) in the file before saving it as a text file.

The program expects one column specifying the rows. If you have more than one column, for example a column identifying the function and another column with a one- or two-letter code for the function that you want displayed in the graphic output, as below, then you will have to remove the first column before creating the text file. The output file presents the data in the same order, so you can re-insert the column with the identifying information without confusion (see discussion of graphic output in §3).

		va-	*vre- -un*	*ori-*	*ni-*
Specific known	sk	1	6	6	6
Specific unknown	su	1	6	6	6
Irrealis nonspecific	ir	1	6	6	6
Question	qu	1	1	6	6
Conditional	cd	1	1	6	6
Comparative	cp	6	6	1	6
Free choice	fr	6	6	1	6
Indirect negation	in	6	1	6	1
Direct negation	dn	6	6	6	1

NB: OC works best with datasets with a lot of categories (i.e. columns). In fact, there is a setting in the OC program (minvotes) that specifies a minimum number of categories in the dataset. If the data does not have enough categories, then the program will not produce output. The OC program has a variable, minVOTES, which allows you to lower

(or for that matter, increase) the minimum number of categories required in the dataset. The default value for minVOTES is 10 categories. However, if the data has few categories that discriminate the items in the dataset, then the result may not be useful because the effects of chance distributions (or errors or missing data) may dominate in the result, so that the spatial model represents more noise than signal.

The OC program also has a setting (lop) that excludes lopsided categories, that is, categories that either include all but a small number of items (say, all but one or two items) or exclude all but a small number of items (e.g. applies only to one or two items). Highly lopsided categories do not provide as much information in discriminating items, so leaving them out of the analysis may not affect the overall result, especially if there are a lot of categories to begin with. The OC program has a variable, LOP, which allows you to lower (or raise) the degree of lopsidedness of a category before it is excluded from the analysis. The default value for LOP is .025, that is, categories that are as lopsided as 97.5% vs. 2.5% are left out of the analysis. It is probably acceptable to lower LOP to as low as .005 (i.e., 99.5% to 0.5%) and still get reliable results (Keith Poole, pers. comm; recall that OC handles lopsided data better than dissimilarity algorithms). Of course, a category that includes all the items or excludes all the items does not tell you anything about the data, and should be excluded from the analysis.

The values of minVOTES and/or LOP are likely to need to be adjusted only if you have few categories to divide the items in your dataset. It is better to reduce LOP to include more categories than to reduce minVOTES to allow for a smaller set of categories, other things being equal.

Executing the program with the data file

In order to execute the program, you must launch R and open the program:

Mac: launch R, then select Open Document in the File menu; then select the program.
PC: launch R, then select Open Script in the File Menu; then select the program.

The program will appear in a new window (i.e. a different window from the R Console window). There are comments in the R program; comment lines begin with #. The information below is also in the comments in the program.

(1) Before you run OC for the first time, you will have to install three packages, by executing the following commands in the R Console window (these instructions are in the comments near the beginning of the OC program):

```
install.packages("pscl")
install.packages("oc")
install.packages("gdata")
```

When you install the first package, you will probably be asked to choose a site from which to get the package; choose a site that is near you (in the same country, or the same or nearby state in the USA). Sometimes a site will not have a current version of the packages, which have to be updated when R is updated; if that is the case, try another

site. (*PC users*: when you install the first package, two pop-up boxes will appear: select "yes" for both. All this does is create a folder to store the packages.)

You only have to do step (1) the first time that you run OC.

As a user, you must replace two items in the R code (these instructions are in the comments near the beginning of the OC program):

(2) You must insert the full name of the location of the folder where the input data file is and where the output files will be written. An example of the name of a folder location is given as a comment line in the program, but the names are slightly different for a Mac and a PC:

Mac: To find out the full name of the location of the folder with the input data file, select the input data file and select Get Info in the File menu (or type command-I). The full name of the location is after "Where:" in the information window. Cut and paste that over the sample file location in the R program.
PC: To find out the full name of the location of the folder with the input data file, right-click on the input data file and select Properties. The full name of the location is after "Location:" in the information window. The file location will have the following format: `C:\Users\John\Desktop\Data`. Cut and paste that over the sample file location in the R program. However, when you paste the file location into R, you will need to add backslashes: `C:\\Users\\John\\Desktop\\Data`.

(3) You must insert the name of the input data file. Again, replace the name of the existing data file in the R program with the name of the input data file you want to run. Again, a sample of the filename is given as a comment line in the R program.

(4) If you want to adjust the value of minVOTES and/or LOP, you must change the values in the program at this step. as noted above, the default value of minVOTES is ten categories, and the default value of LOP is .025 (i.e. 2.5% lopsided).

Now you are ready to run the program:

Mac: Select the entire program in the window by typing command-A. Then select Execute in the Edit menu (in R).
PC: Select Run All in the Edit menu (in R).

The data from the execution of the program will appear in the R Console window. When the program is done, the output files will appear in the folder that you specified.

A very important point is that the spatial model that is produced will not be exactly the same each time you run the same data in the same number of dimensions. As noted in §1, an MDS algorithm produces successively more accurate approximations of the best configuration of data points and cutting points/lines for the number of spatial dimensions chosen. So all of your analyses should be based on a single run of the input data through the R program. If you rerun the input data, for example because you made some changes in the coding, then you should redo all of the qualitative analyses. In fact, the overall

spatial model will not differ very much from one run to the next, so your overall qualitative analysis should remain essentially the same.

3. How to interpret the results of the OC MDS analysis

This section describes the information in the output files of the program, and how to interpret them.

The fitness tests

For any statistical analysis of data, one must evaluate how well the model fits the data. In the case of the MDS OC algorithm, two fitness tests are used (Croft and Poole 2008:12-13):

(1) % Correct Classification. This is the percentage of correct classification of the data, that is, how often data points in the rows in the input data file is correctly classified as belonging or not belonging to a category specified by a column in the input data file. Since the point of MDS is to model all the variance in the data in a low-dimensional space, the fit is unlikely to be perfect, but it should be very high, since the algorithm optimizes correct classification.

(2) Aggregate Proportional Reduction of Error (APRE). This is how much the categorization of the data by the categories in the columns (as constructed by the program) is an improvement on the null model. The null model is categorizing all the data as belonging in the category or categorizing all the data as outside the category. The formula is given below (Croft and Poole 2008:12):

$$\frac{\text{Total tokens in minority category} - \text{total errors}}{\text{Total tokens in minority category}}$$

So a bad model of the category will not be a major improvement in accuracy of categorization over the null model. Also, if the data is lopsided (and/or the data set is large), then the APRE will not be large, because a correct model will in fact be not that far off from the null model. The APRE is an aggregation of the proportional reduction of error for each category specified in the columns.

For an MDS analysis, one must also decide which number of dimensions is the best for interpreting the data. (This is not unlike deciding the number of principal components that should be used in a PCA.) This decision is made by comparing how well a one-dimensional model, a two-dimensional model, and a three-dimensional model fit the data. When adding another dimension to the MDS analysis does not give a major improvement in the Correct Classification and APRE, then it is not worth including the additional dimension.

The practical reality is that applications of MDS in psychology, political science and linguistics generally require only one or two dimensions: either adding a second dimension is not much of an improvement over the one-dimensional model, or adding a

third dimension is not much of an improvement over the two-dimensional model. So the R program here assumes that the desirable model will be in one or two dimensions. In order to calculate the fitness statistics, the OC program runs the MDS analysis in one, two and three dimensions, and produces a file giving the (Correct) Classification and APRE for those three models. If the input file is called `Data.txt`, the file with the fitness statistics will be called `Data_Class_File.txt`. You can inspect the fitness statistics in this file in order to determine which number of dimensions is best, one or two.

The program also outputs files with the results from both the one-dimensional and two-dimensional MDS analyses. You can choose which one you want to analyze. In some cases, it is useful to analyze the one-dimensional model as well as the two-dimensional model, even if the two-dimensional model is a significant improvement on the one-dimensional model: the one-dimensional model may still depict a useful, qualitatively interpretable analysis. The following subsections describe the results from the one-dimensional and two-dimensional analyses and how to interpret them.

One-dimensional analysis: output files

Let us consider the simpler, one-dimensional analysis first. There are two output files for the one-dimensional analysis, one describing the data points (i.e. the data whose similarity you want to measure) and one describing their linguistic properties (i.e. the linguistic categories that classify or categorize the data). If the input file is called `Data.txt`, the first file for the one-dimensional analysis is called `Data_1D_X_File.txt`; we will informally call this the "X" file. The second file for the one-dimensional analysis is called `Data_1D_Z_File.txt`; we will informally call this the "Z" file.

The X file presents all the data points in the input file in the rows, in the order that the data points were give in the input data file, along with analytical information. The analytical information in the columns was designed for both the one-dimensional analysis and the two-dimensional analysis, so some columns will be irrelevant for the one-dimensional analysis. The analytical information in the X file is as follows:

`rank`	the rank order of the points on the one dimension. In some cases, points will be tied.
`correctYea`	the number of categories from the columns in the input data where the data point was correctly included in the category
`wrongYea`	the number of categories from the columns in the input data where the MDS analysis wrongly classifies them as in the category (Yea) when in fact they are not in the category (Nay)
`wrongNay`	the number of categories from the columns in the input data where the MDS analysis wrongly excludes them from the category (Nay) when in fact they are in the category (Yea)
`correctNay`	the number of categories from the columns in the input data where the data point was correctly excluded from the category
`volume`	irrelevant for the one-dimensional analysis
`coord1D`	identical to the rank in the one-dimensional analysis

The Z file presents all the categories in columns in the input file in the rows, in the order that the columns occurred in the input data file. It also gives analytical information about the categories in the columns of the input data file. The analytical information in the Z file is as follows:

`correctYea`	the number of data points where the data point was correctly included in the category
`wrongYea`	the number of data points where the MDS analysis wrongly classifies them as in the category (Yea) when in fact they are not in the category (Nay)
`wrongNay`	the number of categories from the columns in the input data where the MDS analysis wrongly excludes them from the category (Nay) when in fact they are in the category (Yea)
`correctNay`	the number of data points where the data point was correctly excluded from the category
`PRE`	the proportional reduction of error in classification (categorization) of the data from the null model by this particular category. NA means that the category was close to unanimous (all Yea or all Nay) and therefore was dropped from the MDS analysis.
`normVector1D`	irrelevant for the one-dimensional analysis
`midpoints`	the point in the one-dimensional ranking where the category divides the data points between Yea (in the category) and Nay (not in the category)

Interpreting the output: one-dimensional model

The best way to analyze the output is to read the text file into a spreadsheet program such as Excel so that you can manipulate the rows and columns.

The one-dimensional model is relatively straightforward because it is simply a rank order of the data along a single dimension (not unlike an implicational hierarchy or scale, in typological analyses). Thus the best way to analyze the result is to reorder the X file by rank order, and interpret the result. As noted above, the rank order may include "ties", that is, two data points at the same rank in the order. It is important to observe that the one-dimensional scalar model gives only a rank order of the data points; it does not provide a distance metric between points in the rank order. For example, from data points in rank order 3, 4, 5, one can only infer that 3 is closer to 4 than to 5 and 5 is closer to 4 than to 3; one cannot infer that 5 is "twice as far" from 3 than it is from 4.

The Z file gives information about which categories in the columns in the data input file actually contributed to the MDS analysis, and of those that did, the degree to which each category is an improvement on the null model. The categories in which the PRE value is 'NA' did not contribute to the MDS analysis. The categories in which the PRE is zero categorized all the points as being in the category or excluded from the category; that is the null model, and so the MDS modeling of that category is no improvement on the null model. The larger the PRE, the greater the improvement. A PRE of 1 means

100% correct classification of the data points for that category, and is therefore a complete improvement on the null model.

Finally, the midpoints can also be ranked, showing exactly where the category divides the data between Yea and Nay (in the MDS model; so classification will not always be perfect). One can reorder the Z file by the midpoint value to get a ranking of the categories with respect to what qualitative parameter you think underlies the one dimension.

Two-dimensional analysis: output files

The two-dimensional analysis also produces X and Z output files, named `Data_2D_X_File.txt` and `Data_2D_Z_File.txt` (again, assuming the input file is called `Data.txt`). The analytical information that was irrelevant in the one-dimensional model is now relevant or informative in the two-dimensional model.

In addition, two graphs are produced to visually represent the data in the X file and the data in the Z file. This is because the data points are now distributed in a two-dimensional space. As we noted in §1, MDS produces a true spatial model, so all distances in the two-dimensional model are interpretable by the analyst. The categories from the columns in the data input file that include/exclude the data points in the rows of the data input file are represented as straight lines bisecting the two-dimensional space. These lines are called cutting lines. (In the one-dimensional model, the categories are represented as cutting points that bisect the single dimension. In a three-dimensional model, the categories would be represented as cutting planes that bisect the three-dimensional space.) The lines separating the Yeas from the Nays must be bisections of the space because this is a Euclidean spatial model that represents categories linearly.

The X file gives information about the position of the points in two dimensions as well as the error data:

`rank`	the rank order of the points on the first dimension of the two-dimensional model. This is not necessarily the same as the rank in the one-dimensional model, since some of the variance is captured by the second dimension in the two-dimensional model.
`correctYea`	the number of categories from the columns in the input data where the data point was correctly included in the category
`wrongYea`	the number of categories from the columns in the input data where the MDS analysis wrongly classifies them as in the category (Yea) when in fact they are not in the category (Nay)
`wrongNay`	the number of categories from the columns in the input data where the MDS analysis wrongly excludes them from the category (Nay) when in fact they are in the category (Yea)
`correctNay`	the number of categories from the columns in the input data where the data point was correctly excluded from the category
`volume`	the volume of the polytope formed by all the cutting lines that "box in" a data point in the two-dimensional space. A data point could be positioned anywhere in the polytope that boxes it in. The position of some data points is very precisely specified if the

	cutting lines completely box it in; in this case the volume will be very small. For other data points, the polytope is large and the position of the data point is not very precisely specified. The data point could be positioned anywhere in the polytope. This may be relevant in the qualitative interpretation of the MDS model.
`coord1D`	the coordinate of the data point on the first dimension. The origin is in the center of the spatial model, for reasons to be explained below.
`coord2D`	the coordinate of the data point on the second dimension

There are two graphics displays of the data points from the X file. The first, `Data_Points_Fig1.jpeg`, simply displays each data point as a point in the two-dimensional spatial model. The second, `Data_Points_Fig2.jpeg`, displays each data point with the label for the row from the data input file. The second file is relatively easy to read if there are very few data points (as in the indefinite pronouns example). If there are a large number of data points, it is advisable to run the program with one-letter codes for each data point, where the codes correspond to some property of the data points that you believe is relevant to the qualitative interpretation of the MDS analysis. Even so, the letter codes will be superimposed if the data points have the same coordinates in the spatial model. (Your letter codes may not uniquely identify each point, but as noted in §2, you can re-insert the column with the identifying information in the X file, or more precisely, in the spreadsheet you derived from the X file.) As discussed below, you will have to look at the coordinates of each point in the X file in order to fully interpret the results.

The Z file gives information about the position of the cutting line for each category defined by a column in the input data file.

`correctYea`	the number of data points where the data point was correctly included in the category
`wrongYea`	the number of data points where the MDS analysis wrongly classifies them as in the category (Yea) when in fact they are not in the category (Nay)
`wrongNay`	the number of data points where the MDS analysis wrongly excludes them from the category (Nay) when in fact they are in the category (Yea)
`correctNay`	the number of data points where the data point was correctly excluded from the category
`PRE`	the proportional reduction of error in classification (categorization) of the data from the null model by this particular category. NA means that the category was close to unanimous (all Yea or all Nay) and therefore was dropped from the MDS analysis.
`normVector1,2D`	The normal vector for the cutting line is a vector starting at the origin. `normVector1D` and `normVector2D` give the coordinates of the endpoint of the vector. (This is why the origin is at the center of the two-dimensional model.)

`midpoints` the midpoint of the normal vector. The cutting line is perpendicular to the normal vector and intersects it at its midpoint. Thus, one cannot easily read off the cutting line from the information in the Z file, but the cutting line is clearly displayed in the graphs.

There are also two graphics displays of the cutting lines from the Z file (combined with the data points from the X file). The first, `Data_CuttingLines_Fig1.jpeg`, displays the points, and all of the cutting lines (called a Coombs mesh). The red arrows point in the direction which indicates the points that are included in the category defined by the cutting line. The second, `Data_CuttingLines_Fig2.jpeg`, labels the cutting lines with the names provided in the data input file.

Interpreting the output: two-dimensional model

Interpretation of the two-dimensional spatial model is more complex than interpreting a one-dimensional model. First, one must try to identify the overall dimensions of variation in the data. If there are few data points, then one can simply use `Data_Points_Fig2.jpeg` and look at the spatial arrangement of the data point labels. If there are a lot of data points, then one can use the data points graphics display in conjunction with the X file. It is useful to read the X file into a spreadsheet program like Excel and then successively sort the rows by the `coord1D` and `coord2D` columns until you have placed data points that are close together in the space next to each other in the X spreadsheet file. Then qualitative similarities and dimensions should emerge naturally.

The data points are likely to form clusters. These clusters will, hopefully, make qualitative sense, in that the analyst can find qualitative properties that are associated with the cluster. However, this is actually *not* the best way to interpret the results of an MDS analysis. (Hierarchical cluster analysis is an unsupervised method for identifying clusters of data if that's what you want to do; see Woods et al. 1986:254-61; Baayen 2008:138-48). An MDS analysis provides spatial *dimensions* of significant variation in the data. Data points that are close to each other are similar, and those that are far apart are dissimilar; but the similarity is a property of the geometric dimensions of the spatial model, and should represent more or less continuous variation on qualitative dimensions. The clustering is an effect of data points that happen to be quite similar to each other on the relevant qualitative dimensions. The qualitative dimensional interpretation of MDS is discussed in Croft (2010:11-13; see also Woods et al. 1986:262, who nevertheless confusingly include MDS in a chapter discussing clustering methods).

There are several common types of dimensional interpretations of a two-dimensional MDS spatial model. The simplest is when the two dimensions of the spatial model correspond to two independent dimensions of variation in the data. An example of this interpretation is the analysis of the tense-aspect data in Croft and Poole (2008:26, Fig. 8). Tense and aspect are orthogonal dimensions in the two-dimensional model.

Another common type of dimensional interpretation of a two-dimensional model is found when the data is distributed in a horseshoe ("U" or "C" like) shape. An example of this interpretation is the indefinite pronoun data. This data is in a more-or-less ranked order, as can be seen in the semantic map model representation in Croft and Poole (2003:3, Fig. 1, from Haspelmath 1997:64). One end of the horseshoe is the most specific

indefinite pronoun type and the other end is the least specific (negative and free choice). The adposition data discussed in Croft (2010:9-12) is also a horseshoe, extending from the most extreme "on"-type relation (figure supported vertically by gravity on an extended flat ground) to the most extreme "in"-type relation (figure completely enclosed by the ground).

The horseshoe shape emerges when the linguistic categories for the data include data points in the middle of the ranking but not either endpoint of the ranking. Since cutting lines (which represent the categories) must be straight—MDS is a linear, Euclidean spatial model—then the spatial arrangement of the data points must be curved into a horseshoe shape. If the interpretation is truly a horseshoe, then there should be no cutting lines including data points at both of the ends of the ranking but not data points in the middle of the ranking (that is, there should be a gap at the open end of the horseshoe shape where the arrows of all the cutting lines are pointing away). This can be observed in the display of all cutting lines for the indefinite pronoun data in Croft and Poole (2008:19, Fig. 7; gap at the top) and the corresponding display for the adposition data in Croft (2010:11, Fig. 5; gap at the left).

A third common type of dimensional interpretation is a distribution of the data points in a circular shape. An example of a circular dimension of linguistic data is the arrangement of semantic types of predicates with respect to their aspectual behavior in Croft (2012:166, Fig. 4.4). Semantic types of predicates lend themselves to alternative aspectual construals, typically a common pair of aspectual construals for each class of predicate; those pairs of construals form a circle of typical aspectual construals of predicates as described in Croft (2012:165-71). In this case, there are no "ends" defined by categories (cutting lines) facing away from each other.

Finally, one may want to examine the arrangement of the categories themselves, which are represented as cutting lines in the data, and their significance in the two-dimensional model. This information can be found in the Z file. As noted in the section on interpreting a one-dimensional model, cutting lines with NA in the PRE column did not contribute to the analysis; cutting lines with zero in the PRE column were no better than the null model; and the higher the PRE, the greater the improvement over the null model is provided by the cutting lines.

MDS provides the most information when there is a large number of cutting lines that categorize (classify) the data points in many different ways. However, the Coombs mesh of all the cutting lines is difficult to read when there are dozens or hundreds of cutting lines. We would often be interested in viewing only a few cutting lines, or even just one.

There is a second script which allows you to display just a subset of the cutting lines for a given two-dimensional analysis. It uses the X and Z files from a run of the analysis, and the input data file (for the labels of the cutting lines). The input data file and the X and Z files must be in the folder where the output of this program will be produced. As with the main analysis program, the user must first insert some information into the R program before executing it:

(1) The name of the input data file. The program will automatically use the X and Z files derived from the MDS run of the input data file.

(2) The numbers of the columns from the data input file whose cutting lines you want to display. You may enter individual column numbers (x,y,…) separated by commas, a column range (x:y), or a combination of the two.

The output will display all the points, the cutting lines for the selected columns, and the labels of the cutting lines.

References

Baayen, R. Harald. 2008. *Analyzing linguistic data: a practical introduction to statistics using R*. Cambridge: Cambridge University Press.

Croft, William and Keith T. Poole. 2008. Inferring universals from grammatical variation: multidimensional scaling for typological analysis. *Theoretical Linguistics* 34.1-37. [available by emailing wcroft@unm.edu]

Croft, William. 2010. Relativity, linguistic variation and language universals. *CogniTextes* 4.303 [http://cognitextes.revues.org/303/]

Croft, William. 2012. *Verbs: aspect and causal structure.* Oxford: Oxford University Press.

Haspelmath, Martin. 1997. *Indefinite pronouns*. Oxford: Oxford University Press.

Haspelmath, Martin. 2003. The geometry of grammatical meaning: semantic maps and cross-linguistic comparison. *The new psychology of language*, *vol. 2*, ed. Michael Tomasello, 211-42. Mahwah, N. J.: Lawrence Erlbaum Associates.

Poole, Keith T. 2000. Non-parametric unfolding of binary choice data. *Political Analysis* 8(3).211-237.

Poole, Keith T. 2005. *Spatial models of parliamentary voting*. Cambridge: Cambridge University Press.

Regier, Terry, Naveen Khetarpal and Asifa Majid. 2013. Inferring semantic maps. *Linguistic Typology* 17.89-105.

Woods, Anthony, Paul Fletcher & Arthur Hughes. 1986. *Statistics in language studies*. Cambridge: Cambridge University Press.