Please do not circulate, cite, or quote this draft without permission of the authors.

RTaxometrics: An R Package for Taxometric Analysis

Shirley B. Wang and John Ruscio

The College of New Jersey

Author Note

Shirley B. Wang and John Ruscio, Department of Psychology, The College of New Jersey.

Shirley B. Wang is now at the Department of Psychology, Harvard University.

Correspondence concerning this article should be addressed to John Ruscio, Department of Psychology, The College of New Jersey, Ewing, NJ, 08628. Contact: ruscio@tcnj.edu

Abstract

This article presents and illustrates the RTaxometrics package for taxometric analysis, a family of data-analytic procedures that can be used to differentiate categorical and dimensional data. The primary output of taxometric analysis is graphical, and the interpretation of curve shapes can be rather subjective. To reduce this subjectivity, one can perform parallel analyses of categorical and dimensional comparison data, calculate the Comparison Curve Fit Index (CCFI), and examine the consistency of results across multiple taxometric procedures. Investigators have been performing taxometric analysis in these ways using Ruscio's (2016) R code, and the RTaxometrics package replaces this with more modular, efficient, well-documented functions grounded in the implementation options that are used frequently and empirically supported by simulation research. The package provides more user-friendly output and incorporates a recent development known as a CCFI profile that can rigorously test the structure of the data and, if it appears to be categorical, provide an estimate of the taxon base rate. This paper reviews the functions available in the RTaxomerics package and demonstrates their use.

Keywords: taxometric analysis, CCFI, R, computer software

RTaxometrics: An R Package for Taxometric Analysis

Social and behavioral scientists often wonder whether people differ on any given construct by belonging to discrete groups or by varying along a continuum. For instance, do depressed and non-depressed individuals form two separate groups of people, or does everyone fall along a continuous spectrum of depression? Individual differences on any given psychological construct may follow a categorical or dimensional structure. How one chooses to conceptualize and measure a construct, and whether this is congruent with its true latent structure, can have important consequences for theory, research, and practice (Ruscio, Haslam, & Ruscio, 2006). Rather than choosing to conceptualize or measure a construct based on conventional practices or preferences for categories or dimensions, this structural distinction can be addressed empirically (Meehl, 1992).

Meehl's (1995) taxometric method empirically differentiates between categorical and dimensional data using several non-redundant data-analytic procedures. Meehl developed this method to test for a schizotype taxon, or a group of individuals hypothesized to carry genetic liability for schizophrenia (Meehl, 1990). Since its development, taxometric analysis has been widely applied in psychopathology research to examine the latent structure of eating disorders (Thomas et al., 2015), anxiety disorders (Marcus, Sawaqdeh, & Kwon, 2014), mood disorders (Ahmed, Green, Clark, Stahl, & McFarland, 2011), alcohol use disorders (Kerridge, Saha, Gmel, & Rehm, 2013), and attention deficit/hyperactivity disorder (Marcus, Norris, & Coccaro, 2012), as well as more broadly in the social and behavioral sciences (e.g., flashbulb memories, Lanciano & Curci, 2012; crime-related constructs, Walters, 2012).

Within the overarching framework of the taxometric method, dozens of data-analytic procedures have been introduced. A small handful has emerged as the most popular and well-

studied set of taxometric procedures. These include mean above minus below a cut (MAMBAC; Meehl & Yonce, 1994), maximum eigenvalue (MAXEIG; Waller & Meehl, 1998), maximum covariance (MAXCOV; Meehl & Yonce, 1996), and latent mode (L-Mode; Waller & Meehl, 1998). The primary output of these procedures is graphical in nature, requiring users to judge whether it appears more similar to the prototypical curve shapes for categorical or dimensional data. Although this avoids the potential pitfalls of null hypothesis statistial testing, judging curve shapes introduces an unfortunate amount of subjectivity into taxometric research. Compounding the challenge of performing a rigorous taxometric analysis is that each procedure can be performed in a variety of ways and empirical guidance for making implementation deisions was slow to develop because simulation studies required that taxometric experts judge the output of each analysis. This severely constrained the size and scope of methodological research on taxometric analysis.

To enable powerful simulation studies that could provide some guidance for performing taxometric analyses, and to reduce the subjectivity in the interpretation of taxometric results, Ruscio, Ruscio, and Meron (2007) developed a technique in which graphs for the empirical data could be interpreted relative to those for parallel analyses of categorical and dimensional comparison data. The goal was to hold constant all relevant aspects of the empirical data (e.g., sample size, number of variables, marginal distributions, correlation matrices) while varying only the structure used to generate each population of comparison data (i.e., categorial vs. dimensional). By analyzing many random samples drawn from each population, the typical results for each structure could be examined along with the variation attributable to normal sampling error. Plotting results for empirical data alongside those for both types of comparison

data provided a more appropriate reference point than the prototypical curves for each structure that had been generated using a narrow range of fairly ideal data parameters.

To further reduce the subjectivity of these graphs, and especially to afford automated simulation research on taxometric methodology, Ruscio et al. (2007) developed the Comparison Curve Fit Index (CCFI). The CCFI quantifies the extent to which the results for the empirical data are a closer match to those for the categorical or dimensional comparison data. Values can range from 0 (strongest support for dimensional structure) to 1 (strongest support for categorical structure), with .50 representing the most ambiguous outcome possible. A number of simulation studies show that the CCFI effectively differentiates between categorical and dimensional data across a wide range of challenging data conditions (see Ruscio, Ruscio, & Carney, 2011, for an overview).

Another cornerstone of Meehl's taxometric method is the use of multiple non-redundant data-analytic procedures to check the consistency of findings (Meehl, 1995). For a long time, researchers' approaches to consistency testing were uneven, at best. Practice was guided only by a shared ideal that had not been operationalized. Ruscio, Walters, Marcus, and Kaczetow (2010) used the CCFI to specify and evaluate several operationalizations of consistency testing. The best method among those they tested was to obtain CCFI values using multiple taxometric procedures and then calculate and interpret the mean CCFI. Values above .50 support categorical structure, values below .50 support dimensional structure. When a single threshold at .50 is used in this way, there are inevitable errors (i.e., categorical data that yield a CCFI below .50 or dimensional data that yield a CCFI above .50). Research suggests that the error rate should be low provided that the data are appropriate for taxometric analysis (see the sections on "Checking the Data", below), but users can reduce it further by treating values close to .50 as ambiguous. For example,

treating CCFIs from .40 to .60 as ambiguous, and reaching no conclusion, will eliminate most errors that would otherwise have been made. Using a more narrow range of ambiguous CCFIs (e.g., from .45 to .55) will yield fewer ambiguous findings, but at the cost of an increase in the error rate.

Performing Taxometric Analysis in R

Mainstream statistical software does not include taxometric analysis, so a number of investigators created their own special-purpose code through the years. To check our impression that Ruscio's (2016) R code for taxometric analysis had become the most popular, we performed a review of 37 taxometric studies published from 2011 to 2016. In each case, the researchers used Ruscio's taxometric programs. This code has been available free of charge since 2000 via Ruscio's professional web site (ruscio.pages.tcnj.edu). The code was originally written in the commercial S+ language and soon converted for use in the R computing environment.

This code has been updated many times, with the results that one might expect of an incremental, evolutionary process. The original formulation and structure remains, buried beneath a variety of add-ons and modifications. The code's growth rendered it increasingly difficult to read or update, much less to reorganize in more modular and efficient ways. Moreover, even as the practice of taxometric analysis began to converge on best practices supported by methodological research, the difficulty of making substantial changes to the inelegant code meant that some outdated options remained and some new techniques had not been incorporated.

The current project was a complete reworking of Ruscio's R code for taxometric analysis. We followed the modern style conventions of R programming and documentation to produce an R package that is distributed in the standard way, rather than through a personal web site.

Functions in the RTaxometrics package are well documented and programmed to be readable, streamlined, and run-time efficient. A total of 42 functions are used in a modular manner, which eliminates redundancy in the code and renders its operations transparent. Only 5 of these functions should be accessed directly by users (each of these is demonstrated below), and their use and output was designed to be user-friendly. Details regarding the new code are provided below, followed by illustrations of its use.

Function Defaults and Procedural Implementation

A review of literature on empirically supported guidelines for taxometric analysis was conducted to determine options that the new code should include, as well as appropriate default choices. To help understand what follows, readers unfamiliar with taxometric analysis should consult a tutorial such as Ruscio et al. (2011).

Because the MAXEIG and MAXCOV procedures produce remarkably similar results (Ruscio et al., 2010) and should not be used as consistency tests, a single function is provided to perform MAXEIG, but not MAXCOV. In the event that only two variables are provided for analysis, the MAXSLOPE procedure is performed instead of MAXEIG.

The following default choices have been set for procedural implementation. For the MAMBAC procedure, default settings include variables used in all input-output pairings (`assign.MAMBAC` = 1), cuts starting and ending at 25 points from either extreme (`n.end` = 25), and 50 total cuts (`n.cuts` = 50). For the MAXEIG procedure, default settings include each variable serving as an input variable once (`assign.MAXEIG` = 1) and overlapping windows at .90 (`overlap` = .90). For the L-Mode procedure, default settings include searching for the left mode beyond -.001 (`mode.l` = -.001) and searching for the right mode beyond .001 (`mode.r` = .001).

Appendix A provides a complete list of options that can be specified, along with default settings and any required minimum or maximum values.

Several aspects of the MAMBAC, MAXEIG, L-Mode, and MAXSLOPE procedures are implemented in only one way in the new functions, meaning that users would have to modify the code itself to select alternatives. For the MAMBAC procedure, cuts are located based on case numbers rather than scores on the input variable. For the MAXEIG procedure, cases are divided into subsamples along the input variable using overlapping windows, rather than the non-overlapping intervals of traditional MAXCOV analyses (though users can set the overlap parameter as low as 0 so that the windows effectively become intervals). Likewise, the MAXEIG function only calculates eigenvalues, not covariances. The L-Mode procedure performs a factor analysis and then calculates factor scores using Bartlett's (1937) weighted least squares method. Finally, the MAXSLOPE procedure uses Cleveland's (1979) LOWESS (LOcally WEighted Scatterplot Smoother).

Streamlining the Code and Increasing Run-Time Efficiency

A review of 37 recently published taxometric studies was conducted to find ways to streamline the program code by removing calculations and output that are seldom or never reported. For example, because it has become common practice to standardize variables prior to analysis, the option not to do so was removed. Likewise, users can no longer request averaged curves for each variable in the analysis or Bayesian probabilities of group membership.

In addition to removing rarely used options, several additional steps were taken to streamline the code. First, numerous parameters are shared across all functions involved in performing taxometric analysis. Rather than repeatedly specifying these parameters when one function calls subsidiary functions, the new R package bundles these parameters into a single list

object that is passed between functions (see Appendix A for a complete list of shared and procedure-specific parameters included within this object). This makes the code more readable as well as making future modifications relatively simple. In addition, whenever possible the new functions minimize the use of loops and use the smallest objects possible to decrease memory usage and run time.

Perhaps the most significant improvement, from a run-time perspective, involves the use of comparison data. Standard practice in taxometric analysis calls for the generation and parallel analysis of artificial comparison data (Ruscio et al., 2007). Generating the necessary populations of categorical and dimensional comparison data, from which random samples are taken for parallel analysis, can take as long or longer than performing all of the taxometric analyses. This step used to be done separately for each taxometric procedure. To improve this process, the current code begins by generating the populations of comparison data and stores these for use by all taxometric procedures that are subsequently performed.

Checking the Data

Taxometric analysis requires that data meet several requirements in order to reach accurate and informative conclusions (Meehl, 1995; Ruscio et al., 2010). These include total sample size ($N \geq 300$), size and base rate of the putative taxon ($n_t \geq 50$ and $P \geq .10$), number of variables ($k \geq 2$), number of ordered categories per variable ($C \geq 4$), between-group validity of each variable ($d \geq 1.25$), and within-group correlations among variables ($r_{wg} \leq .30$). Although it is desirable for data sets to meet each of these requirements, a number of simulation studies have shown that borderline values on some of these criteria, or failure to meet one or more criteria, may be offset by especially favorable characteristics on other criteria in the same data set (Ruscio et al., 2011).

Analyses to check whether data were appropriate for taxometric analysis were previously completed within the functions for taxometric procedures themselves. For example, if one constructed a set of variables and submitted it to taxometric analysis, the output would include information about the between-group validity and within-group correlations of these variables. Incorporating this into the taxometric functions themselves may have been convenient, but it also may have muddied the distinction between checking whether the data are appropriate for analysis and performing the analysis itself. To make this clearer, the RTaxometrics package includes a new `CheckData()` function intended to be run before any taxometric procedures. This function examines and provides output bearing on each of the characteristics listed above. If data do not meet one or more of these requirements, the function provides warning notes in the output (e.g., "This is smaller than the recommended minimum of $N = 300$").

Status Updates

The two main functions to perform taxometric analyses—`RunTaxometrics()` and `RunCCFIProfile()`—begin by performing a variety of checks on the data and the parameter specifications. As the checks are passed, the user is notified of progress. Many potential problems are anticipated in the code, which will make changes as needed and alert the user. The functions begin by checking the data in three ways. The first check is for missing data, performing a listwise deletion if any is found. The second check—for `RunTaxomerics()`, but not for `RunCCFIProfile()`—is that the final column contains a variable classifying each case as a member of the taxon (coded as 2) or complement (coded as 1) group. The third check ensures that each variable contains nonzero variance, adding a small amount of random variance to any variable(s) if necessary. The functions then check that all parameters are specified in acceptable ways, as detailed in Appendix A. If necessary, changes are made and reported.

After checking the data and the parameter specifications, the functions perform analyses of the empirical data as well as generating populations of comparison data and analyzing random samples from each. Each time a key step is completed, this progress is reported. Not only can it be reassuring to know that particular steps have run successfully, but especially for a time-consuming analysis (e.g., a large sample with many variables) it can be helpful to track progress.

CCFI Profiles

A recent development in taxometric methodology involves performing analyses using a series of populations of categorical comparison data that vary in the base rate of the taxon. The purpose is to examine how the CCFI changes when known groups differ in their relative size. Ruscio, Carney, Dever, Pliskin, and Wang (2017) found that creating a CCFI profile using a range of base rates for categorical comparison data (from .025 to .075, in increments of .025) provided two key benefits.

First, using CCFI profiles improves base rate estimation relative to what can be obtained using formulas for each taxometric procedure. If the results support an inference of categorical structure, locating the peak in the CCFI profile provides a clue about the taxon base rate. It is expected that this peak will emerge for the population of categorical comparison data generated using a base rate rate close to that for the empirical data. Because a discrete series of base rates is used to generate the CCFI profile, and also because each CCFI contained therein will be subject to sampling error, the profile is smoothed before locating its peak. The location of the peak in this smoothed curve is then used as the base rate estimate. Ruscio et al. (2017) found that this decreases bias and increases precision for the MAMBAC, MAXEIG, and L-Mode procedures.

Second, a weighted mean of the CCFI values in the profile improves the ability of CCFI to differentiate between categorical and dimensional data. A single CCFI value is useful, but like

any statistic it is subject to sampling error. Averaging values reduces the amount of sampling error and yields an aggregate CCFI that even more effectively differentiates between categorical and dimensional data. The weighting scheme is based on the distance from each data point to the estimate of the taxon base rate, thus giving more weight to points nearer the estimated base rate.

In addition to the `RunTaxometrics()` function that performs taxometric analysis for a sample of data, the RTaxometrics package includes the `RunCCFIProfile()` function to perform a series of taxometric analyses, generate a CCFI profile, and use this to calculate aggregate CCFI values and estimate the taxon base rate. Below, the use of these and other functions is illustrated.

Using the RTaxometrics Package

R is available as a free download from any of the sites listed at http://cran.r-project.org/mirrors.html. The R computing environment contains a wide array of mathematical, statistical, and graphing tools, and most R commands require entering text. R organizes information using a "console" window for entering these commands and displaying text results, as well as additional windows that are created as needed for presenting graphical output. R is case sensitive. For instance, whereas `install.packages()` will run the appropriate function, `Install.Packages()` will not be recognized. The commands used in R are called *functions*, and users may specify different parameters for each function using *arguments*.

To install the RTaxometrics program in R (R Development Core Team, 2010), users should first install and then load the package:

```
> install.packages("RTaxometrics")
> library(package = "RTaxometrics")
```

The RTaxometrics package contains 42 functions, but only 5 should be called directly by users. `CheckData()` should be run prior to any taxometric analysis to ensure that the data are appropriate for taxometric analysis. `RunTaxometrics()` performs taxometric analyses for a

sample of data. `RunCCFIProfile()` performs a series of taxometric analyses to generate a CCFI profile. `CreateData()` generates a sample of categorical or dimensional data. `ClassifyCases()` assigns cases to groups using the base-rate classification method (Ruscio, 2009). Text output that appears in the console window may be copied and pasted into other files or applications, and graphical output can be copied and pasted or printed as-is. Unfortunately, R does not allow users to modify elements of a graph or reorganize graphs within a graph sheet. If users would like axes labeled or scaled differently, for example, they must edit the code that performs the function.

Creating a Data Set

The `CreateData()` function creates an artificial data set based on either categorical or dimensional structure, including within-group correlations, skew, and/or ordered categorical values if desired. The program returns a data object containing the variables and a final column containing group membership (1 = complement, 2 = taxon). For dimensional data, this final column is created using the `ClassifyCases()` function described below, and the codes do not correspond to actual groups. Artificial data can be useful for getting to know the taxometric programs and becoming familiar with their output by conducting analyses using data sets whose characteristics are known.

For example, suppose we wanted to create a categorical data set by running the `CreateData()` function. We'll assign these data to the object "`x`" so that they can be provided to other functions:

```
> x <- CreateData("cat")
```

By specifying the argument "`cat`", the function will create a categorical data set. Alternatively, users may specify certain parameters when creating a data set:

```
> x <- CreateData("cat", n = 500, p = .3)
```

See Appendix B for full details on default settings and optional parameter specifications for the `CreateData()` function.

Checking the Data

The `CheckData()` function checks whether the data are appropriate for taxometric analysis. Users should ensure that the data set is a matrix object including one variable per column, followed by a final column containing case classification coded as 1 = complement, 2 = taxon. If the data set does not include this final classification column, users can run the `ClassifyCases()` function described below to assign cases to groups. Users should assign their data to an object. For example, to import data from a comma-separated text file:

```
> x <- read.csv("mydata.txt")
```

After the data is imported and contains a classification column, running the `CheckData()` function is relatively straightforward:

```
> CheckData(x)

Sample size:  N = 600
Taxon base rate:  P = 0.5
Taxon size:  n = 300
Complement size:  n = 300
Number of variables:  k = 4

Distributions:

   M SD Skewness Kurtosis
v1 0  1     0.01    -0.51
v2 0  1     0.08    -0.43
v3 0  1     0.01    -0.44
v4 0  1    -0.09    -0.25

Validities:

     Cohen's d
v1        1.92
v2        1.79
v3        1.97
v4        1.89
```

```
Mean      1.89

Within-group correlations (taxon):

      v1    v2    v3    v4
v1  1.00  0.04  0.03 -0.05
v2  0.04  1.00 -0.06  0.08
v3  0.03 -0.06  1.00  0.07
v4 -0.05  0.08  0.07  1.00
Mean = 0.02

Within-group correlations (complement):

      v1    v2   v3    v4
v1  1.00 -0.02 0.04 -0.04
v2 -0.02  1.00 0.07  0.07
v3  0.04  0.07 1.00  0.07
v4 -0.04  0.07 0.07  1.00
Mean = 0.03
```

This function displays text only, meaning that its output should not be assigned to an object. If one or more data requirements are not met, the program will print warnings. To illustrate, if we create a data set using `CreateData()` with `d = 1.0`, the `CreateData()` function will include the following output and warning:

```
Validities:

     Cohen's d
v1        1.14
v2        0.73
v3        1.23
v4        0.97
Mean      1.02
  * One or more values below the recommended minimum of d = 1.25.
```

Classifying Cases

If users wish to perform taxometric analyses on a data set that does not include a final column containing classification, the `ClassifyCases()` function may be used to assign cases to groups based on a specified base rate. This function assigns cases to groups using the base-rate classification technique (Ruscio, 2009). Specifically, cases are sorted according to their total

scores on all available variables, and then the highest-scoring cases are assigned to the taxon such that the proportion of taxon members equals the specified base rate. This function requires two arguments to be specified: `x` (the data set) and `p` (the base rate):

```
> x <- ClassifyCases(x, p = .5)
```

This function returns the data matrix with an additional final column containing classification (1 = complement, 2 = taxon) appended at the end. Users should note that if the base rate locates the threshold dividing taxon and complement members at a score shared by many cases, these cases will all be assigned to the same group. This may mean that the group sizes as classified do not precisely reproduce the specified base rate. For instance, if a total of 200 cases are to be assigned to the taxon and 190 score below 12, with the next 30 scores tied at 12, this means that none of these 30 cases will be assigned to the taxon (had it been only, say, 15 scores tied at 12, then all of them would have been assigned to the taxon). If the threshold dividing taxon and complement members is identified by a tied score that equals the maximum score, then all cases at this score are assigned to the taxon (otherwise, all cases in the entire sample would be assigned to the complement); likewise, if the tied score equals the minimum score, all cases at this score are assigned to the complement.

Running Taxometric Analyses

The `RunTaxometrics()` function performs taxometric analyses for a sample of data. If the supplied (empirical) data set contains three or more variables ($k \geq 3$), the function will automatically run the MAMBAC, MAXEIG, and L-Mode procedures. If the supplied data set contains only two variables, the function will automatically run only the MAMBAC and MAXSLOPE procedures. Otherwise, users may also specify which procedures they wish to perform by specifying the `MAMBAC`, `MAXEIG`, `L-Mode`, and `MAXSLOPE` parameters as `TRUE` or `FALSE`. This function requires one argument to be specified: `x` (the data set), although users may also

choose to specify a variety of other shared and procedure-specific parameters (see Appendix A for details):

```
> RunTaxometrics(x)
```

While running, the function will provide users with updates on the status of program execution, as well as remind users that the `CheckData()` function should first be run to determine whether data are appropriate for taxometric analysis:

```
STATUS OF PROGRAM EXECUTION

Checking for missing data
Checking classification variable
Checking for variance
Checking program parameters
Generating population of dimensional comparison data
Generating population of categorical comparison data
  Generating taxon
  Generating complement
Analyzing empirical data
Analyzing samples of dimensional comparison data
Analyzing samples of categorical comparison data

  Note: Users should run the CheckData() function to evaluate whether
        data appear to be adequate for taxometric analysis.
```

The function will next provide text output in the R console, and graphical output in a separate window. The detailed text summary of the analyses includes an overview of relevant shared and procedure-specific parameters, as well as CCFI values and base rate estimates by procedure and averaged across procedures:

```
TAXOMETRIC ANALYSIS RESULTS

Summary of shared analytic specifications
  sample size:  600
  number of variables:  4
  comparison data population size:  1e+05
  comparison data samples:  100
  comparison data taxon base rate:  0.5
  replications:  1
```

```
Summary of MAMBAC analytic specifications
  cuts:  50 evenly-spaced cuts beginning 25 cases from either extreme
  indicators:  all possible input-output pairs
  number of curves:  12

Summary of MAXEIG analytic specifications
  subsamples:  50 windows that overlap 0.9
  indicators:  input = one variable, output = all other variables
  number of curves:  12

Summary of L-Mode analytic specifications
  position beyond which to search for left mode:  -0.001
  position beyond which to search for right mode:  0.001

Comparison Curve Fit Index (CCFI)
  MAMBAC:  0.897
  MAXEIG:  0.821
  L-Mode:  0.912
  mean:  0.877

  Note: CCFI values can range from 0 (dimensional) to 1 (categorical).
        The further a CCFI is from .50, the stronger the result.

Base Rate Estimates:
  MAMBAC:  0.45
  MAXEIG:  0.444
  L-Mode:
    based on location of left mode:  0.47
    based on location of right mode:  0.56
    mean:  0.515
  mean:  0.47

  Note: There is no evidence-based way to use base rate estimates to
        differentiate categorical and dimensional data. They should
        only be used if evidence supports categorical structure.
```

The graphical output (see Figure 1) includes panels of curves with results for the empirical data (dark line) superimposed above the results for the categorical comparison data, and then the results for the dimensional comparison data. Results for comparison data sets are summarized by plotting the middle 50% of data points as a gray band and light lines that show the minimum and maximum values. In this example, both the text and graphical output support a

categorical structure (CCFI > .50), which is correct, and estimate the taxon base rate to be approximately .47, which is very close to the true value of .50.

Generating a CCFI Profile

The `RunCCFIProfile()` function bears many similarities to the `RunTaxometrics()` function, with the main difference being that it performs a series of taxometric analyses to generate a CCFI profile. The default and optional choices of taxometric procedures to be performed are the same as for the `RunTaxometrics()` function. By default, this function will generate 39 populations of categorical comparison data from $P = .025$ to $P = .975$, although users may adjust these parameters by specifying values for `min.p`, `max.p`, and `num.p`. This function requires just one argument to be specified: `x` (the data set), although users may also choose to specify a variety of other shared and procedure-specific parameters (see Appendix A for details). Users should only include the specific columns that contain data, and not case classifications. In the case of our illustrative data object `x`, this means submitting `x[, 1:4]` rather than simply `x`. If the latter was submitted, the case classification variable in the 5th column would be treated as another variable in the analysis. The following command would generate a CCFI profile:

```
> RunCCFIProfile(x[, 1:4])
```

As before, this function will provide users with updates on the status of program execution as well as a reminder about the `CheckData()` function:

```
STATUS OF PROGRAM EXECUTION

Checking for missing data
Checking for variance
Checking program parameters
Analyzing empirical data
Generating population of dimensional comparison data
Analyzing samples of dimensional comparison data
Generating populations of categorical comparison data and analyzing samples
  p = 0.025
  p = 0.05
```

```
p = 0.075
p = 0.1
p = 0.125
p = 0.15
p = 0.175
p = 0.2
p = 0.225
p = 0.25
p = 0.275
p = 0.3
p = 0.325
p = 0.35
p = 0.375
p = 0.4
p = 0.425
p = 0.45
p = 0.475
p = 0.5
p = 0.525
p = 0.55
p = 0.575
p = 0.6
p = 0.625
p = 0.65
p = 0.675
p = 0.7
p = 0.725
p = 0.75
p = 0.775
p = 0.8
p = 0.825
p = 0.85
p = 0.875
p = 0.9
p = 0.925
p = 0.95
p = 0.975

Note: Users should run the CheckData() function to evaluate whether
      data appear to be adequate for taxometric analysis.
```

The detailed text summary of the analyses includes an overview of relevant shared and procedure-specific parameters, as well as aggregate CCFI values and base rate estimates calculated using profiles for each procedure as well as the mean profile across procedures. These

aggregate CCFI values represent the overall mean CCFI value averaged with 25% of the nearest data points.

```
TAXOMETRIC ANALYSIS RESULTS

Summary of shared analytic specifications
  sample size:  600
  number of variables:  4
  comparison data population size:  1e+05
  comparison data samples:  100
  replications:  1

Summary of MAMBAC analytic specifications
  cuts:  50 evenly-spaced cuts beginning 25 cases from either extreme
  indicators:  all possible input-output pairs
  number of curves:  12

Summary of MAXEIG analytic specifications
  subsamples:  50 windows that overlap 0.9
  indicators:  input = one variable, output = all other variables
  number of curves:  12

Summary of L-Mode analytic specifications
  position beyond which to search for left mode:  -0.001
  position beyond which to search for right mode:  0.001

Aggregate Comparison Curve Fit Index (CCFI)
  mean profile:  0.729
  MAMBAC profile:  0.785
  MAXEIG profile:  0.704
  L-Mode profile:  0.699

  Note: CCFI values can range from 0 (dimensional) to 1 (categorical).
        The further a CCFI is from .50, the stronger the result.
        Aggregate CCFI values are a weighted mean of all CCFI values
        in the profile.

Base Rate Estimates
  mean profile:  0.518
  MAMBAC profile:  0.531
  MAXEIG profile:  0.518
  L-Mode profile:  0.512

  Note: There is no evidence-based way to use base rate estimates to
        differentiate categorical and dimensional data. They should
        only be used if evidence supports categorical structure.
```

The graphical output (see Figure 2) includes a single CCFI profile graph that plots CCFI values across the range of categorical comparison data base rates. This graph includes the mean CCFI across procedures (dark line with dots), as well as the CCFI values for each procedure (`M` for MAMBAC, `X` for MAXEIG, `L` for L-Mode, and `S` for MAXSLOPE). A horizontal reference line is plotted at CCFI = .50. In this example, both the text and graphical output support a categorical structure (CCFI > .50), which is correct, and estimate the taxon base rate to be .518, which is close to its true value of .50 (and a bit closer than the earlier estimate of .47 provided by the `RunTaxometrics()` function).

Conclusion

This article introduced the new RTaxometrics package and provided a tutorial on its use. The package has been developed from scratch and includes many improvements from previous versions of Ruscio's (2016) taxometric program code. RTaxometrics has been written to be more modular, more easily modified and updated, more readable, and more efficient in execution of functions and procedures, as well as to provide more user-friendly output. The initial RTaxometrics package was called version 2.0 in recognition of the fact that this is a comprehensive overhaul of Ruscio's code, which had evolved in many ways over the course of 16 years. The original code is implicitly version 1.0, with updates constituting unspecified versions between 1.0 and 2.0.

Along with a host of subsidiary functions that users should not call directly, the RTaxometrics package provides 5 core functions with which users can generate artificial data sets (`CreateData()`), assign cases to groups (`ClassifyCases()`), check whether data are appropriate for taxometric analysis (`CheckData()`), perform taxometric analysis for a sample of data (`RunTaxometrics()`), and perform taxometric analyses to generate a CCFI profile

(`RunCCFIProfile()`). The generation and analysis of CCFI profiles is a novel technique that was not included in earlier taxometric program code. CCFI profiles rigorously test for the existence of groups in empirical data and estimate their size with less bias and greater precision than conventional techniques (Ruscio et al., 2017). Whether using the `RunTaxometrics()` or the `RunCCFIProfile()` function, we anticipate that the RTaxometrics package will allow researchers to more easily and effectively perform their taxometric analyses.

References

Ahmed, A. O., Green, B. A., Clark, C. B., Stahl, K. C., & McFarland, M. E. (2011). Latent structure of unipolar and bipolar mood symptoms. *Bipolar Disorders, 13*(5-6), 522-536. doi:10.1111/j.1399-5618.2011.00940.x

Bartlett, M. S. (1937). The statistical conception of mental factors. *British Journal Of Psychology*, *28,* 97-104.

Cleveland, W. S. (1979). Robust locally-weighted regression and smoothing scatterplots. *Journal of the American Statistical Association, 74,* 829-836. doi:10.2307/2286407

Kerridge, B. T., Saha, T. D., Gmel, G., & Rehm, J. (2013). Taxometric analysis of DSM-IV and DSM-5 alcohol use disorders. *Drug And Alcohol Dependence, 129*(1-2), 60-69. doi:10.1016/j.drugalcdep.2012.09.010

Lanciano, T., & Curci, A. (2012). Type or dimension? A taxometric investigation of flashbulb memories. *Memory, 20*(2), 177-188. doi:10.1080/09658211.2011.651088

Marcus, D. K., Norris, A. L., & Coccaro, E. F. (2012). The latent structure of attention deficit/hyperactivity disorder in an adult sample. *Journal Of Psychiatric Research, 46*(6), 782-789. doi:10.1016/j.jpsychires.2012.03.010

Marcus, D. K., Sawaqdeh, A., & Kwon, P. (2014). The latent structure of generalized anxiety disorder in midlife adults. *Psychiatry Research, 215*(2), 366-371. doi:10.1016/j.psychres.2013.12.011

Meehl, P.E. (1990). Toward an integrated theory of schizotaxia, schizo- typy, and schizophrenia. *Journal of Personality Disorders, 4*, 1–99.

Meehl, P. E. (1992). Factors and taxa, traits and types, differences of degree and differences in kind. *Journal of Personality, 60*(1), 117-174. doi:10.1111/j.1467-6494.1992.tb00269.x

Meehl, P. E. (1995). Bootstraps taxometrics: Solving the classification problem in psychopathology. *American Psychologist*, *50*(4), 266-275. doi:10.1037/0003-066X.50.4.266

Meehl, P. E., & Yonce, L. J. (1994). Taxometric analysis: I. Detecting taxonicity with two quantitative indicators using means above and below a sliding cut (MAMBAC procedure). *Psychological Reports*, *74*(3, Pt 2), 1059-1274.

Meehl, P. E., & Yonce, L. J. (1996). Taxometric analysis: II. Detecting taxonicity using covariance of two quantitative indicators in successive intervals of a third indicator (Maxcov procedure). *Psychological Reports*, *78*(3, Pt 2), 1091-1227. doi:10.2466/pr0.1996.78.3c.1091

Ruscio, J. (2009). Assigning cases to groups using taxometric results: An empirical comparison of classification techniques. *Assessment*, *16*(1), 55-70. doi:10.1177/1073191108320193

Ruscio, J. (2016). Taxometric programs for the R computing environment: User's manual. Available on the world wide web at http://ruscio.pages.tcnj.edu/quantitative-methods-program-code/

Ruscio, J., Carney, L., Dever, L., Pliskin, M., Wang, S.B. (2017). Using the Comparison Curve Fit Index (CCFI) in taxometric analyses: Averaging curves, standard errors, and CCFI profiles. *Manuscript submitted for publication.*

Ruscio, J., Haslam, N., & Ruscio, A. M. (2006). *Introduction to the taxometric method: A practical guide*. Mahwah, NJ: Lawrence Erlbaum Associates.

Ruscio, J., Ruscio, A. M., & Carney, L. M. (2011). Performing taxometric analysis to distinguish categorical and dimensional variables. *Journal Of Experimental Psychopathology*, *2*(2), 170-196. doi:10.5127/jep.010910

Ruscio, J., Ruscio, A. M., & Meron, M. (2007). Applying the bootstrap to taxometric analysis: Generating empirical sampling distributions to help interpret results. *Multivariate Behavioral Research*, *42*(2), 349-386. doi:10.1080/00273170701360795

Ruscio, J., Walters, G. D., Marcus, D. K., & Kaczetow, W. (2010). Comparing the relative fit of categorical and dimensional latent variable models using consistency tests. *Psychological Assessment*, *22*(1), 5-21. doi:10.1037/a0018259

Thomas, J. J., Eddy, K. T., Ruscio, J., Ng, K. L., Casale, K. E., Becker, A. E., & Lee, S. (2015). Do recognizable lifetime eating disorder phenotypes naturally occur in a culturally Asian population? A combined latent profile and taxometric approach. *European Eating Disorders Review, 23*(3), 199-209. doi:10.1002/erv.2357

Waller, N. G., & Meehl, P. E. (1998). *Multivariate taxometric procedures: Distinguishing types from continua*. Thousand Oaks, CA, US: Sage Publications, Inc.

Walters, G. D. (2012). Taxometrics and criminal justice: Assessing the latent structure of crime-related constructs. *Journal Of Criminal Justice, 40*(1), 10-20. doi:10.1016/j.jcrimjus.2011.11.003

Appendix A

The following parameters for the `RunTaxometrics` and `RunCCFIProfile` functions are shared across all taxometric procedures (MAMBAC, MAXEIG, L-Mode, MAXSLOPE). All subsidiary functions will automatically run with the following defaults, unless otherwise specified by users. Although there is flexibility in adjusting these parameters, some minimum and maximum values are often required. For example, the minimum size of populations of comparison data is 10,000; if users set `n.pop` to a value < 10,000, it will automatically be reset to 10,000 (and the user will be notified of this change).

`seed:` The random number seed provided prior to analysis of empirical data as well as prior to generating each population of comparison data (if comparison data are used); this allows users to create exact replications of analyses. The default value is 1.

`n.pop:` The size of populations of comparison data. The default value is 100,000, and the minimum value is 10,000.

`n.samples:` The number of samples drawn from each population of comparison data; Generating multiple sets of comparison data is strongly encouraged. The default value is 100, and the minimum value is 10.

`reps:` The number of times to resort tied scores and redo calculations, which are averaged to obtain final results. If no tied scores are found, the default and minimum values are 1; if tied scores are found, the default and minimum values are 10.

`min.p:` The minimum base rate used for generating a CCFI profile. The default value is .025, and the minimum value is 0.025.

`max.p`: The maximum base rate used for generating a CCFI profile. The default value is .975, and the maximum value is .975.

`num.p`: The number of base rates used for generating a CCFI profile. The default value is 39, and the minimum value is 20.

The following parameters are specific to the MAMBAC procedure:

`MAMBAC`: Whether the MAMBAC procedure is performed (default = TRUE).

`assign.MAMBAC`: Whether the variables are used in all input-output pairings (`assign.MAMBAC` = 1) or one variable at a time is used as the output variable with all remaining variables summed to form the corresponding input variable (`assign.MAMBAC` = 2). The default value is 1.

`n.cuts`: The number of cuts along the input variable. The default value is 50, and the minimum value is 25.

`n.end`: The number of cases at each extreme along the input variable before making the first and last cuts. The default value is 25, and the minimum value is 10.

The following parameters are specific to the MAXEIG procedure:

`MAXEIG`: Whether the MAXEIG procedure is performed (default = TRUE).

`assign.MAXEIG`: Whether the variables are used in all input-output triplets (`assign.MAXEIG` = 1), each variable serves as input once with all remaining variables serving as the correspond output variables (`assign.MAXEIG` = 2), or two variables at a time are used as the output variables with all remaining variables summed to form the

corresponding input variable (`assign.MAXEIG` = 3). The default value is 1.

`windows:` The number of overlapping windows. The default value is 50, and the minimum value is 10.

`overlap:` The proportion of overlap between windows. The default value is .90, and the minimum value is 0.

The following parameters are specific to the L-Mode procedure:

`LMode:` Whether the L-Mode procedure is performed (default = TRUE).

`mode.l:` The position beyond which to search for the left mode. The default value is -.001, and this value must be a negative number.

`mode.r:` The position beyond which to search for the right mode. The default value is .001, and this value must be a positive number.

The following parameters are specific to the MAXSLOPE procedure:

`MAXSLOPE:` Whether the MAXSLOPE procedure is performed (default = FALSE).

Appendix B

The `CreateData` function allows users to create artificial datasets of known structure (categorical or dimensional), with the following data parameters. Unless otherwise specified, the `CreateData` function will run with the following default values.

`str`: The type of data to be generated. This argument has no default value; users must specify either "dim" to generate a sample of dimensional data or "cat" (or anything else) to generate a sample of categorical data.

`n`: Sample size. The default value is 600.

`k`: Number of variables. The default value is 4.

`p`: Taxon base rate. The default value is .5.

`d`: Standardized mean difference between groups. The default value is 2.

`r`: Correlation among variables. The default value is 0.

`r.tax`: Correlation among variables within the taxon. The default value is 0.

`r.comp`: Correlations among variables within the complement. The default value is 0.

`skew`: Amount of skew to be applied to variables. The default value is 0.

`cuts`: Number of values to use when generating ordered categorical data. The default value is 0.

`seed`: Random number seed; specifying the same seed enables users to generate and analyze identical data sets. The default value is 1.

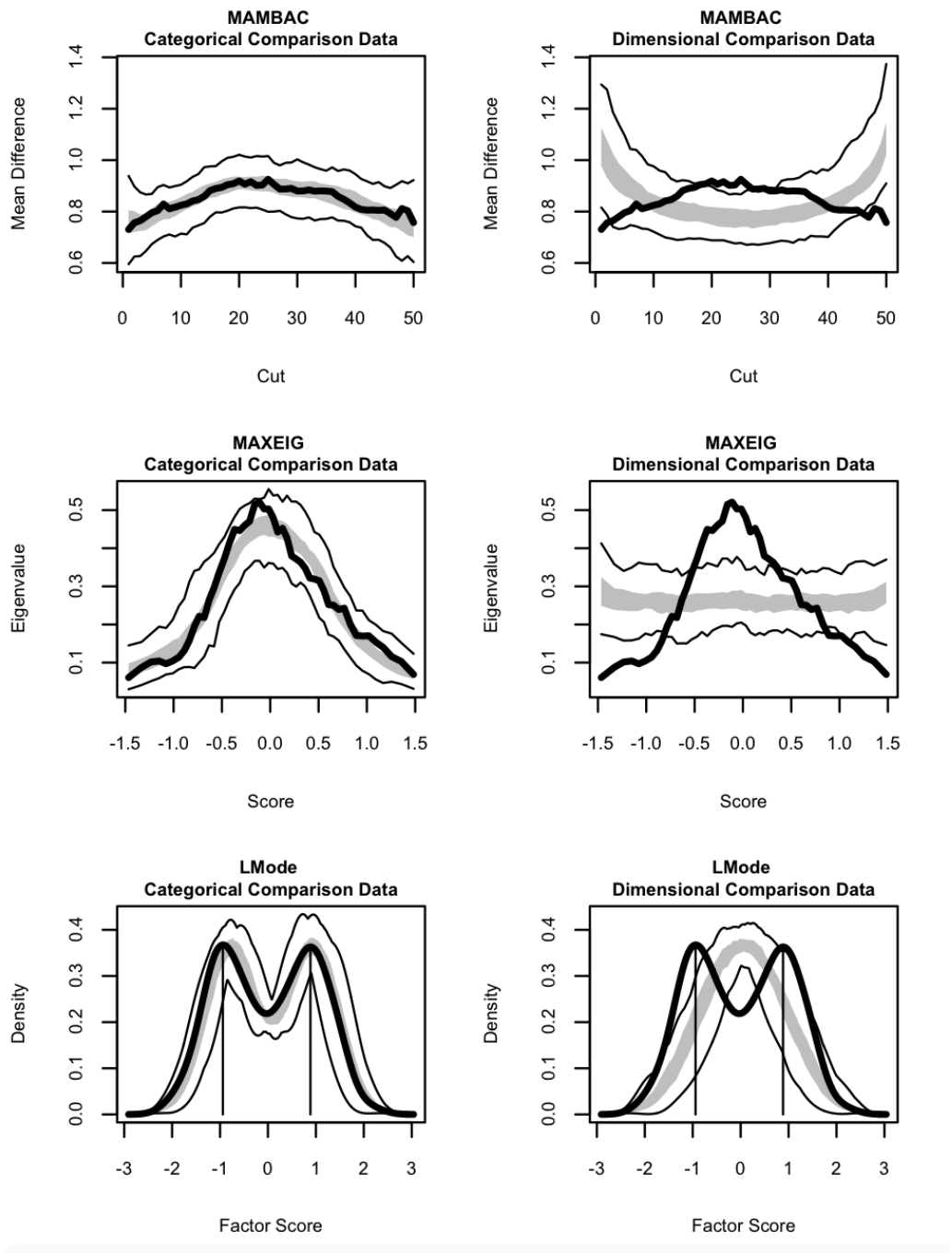

Figure 1. Graphical output from `RunTaxometrics`; panel of curves with results for empirical data superimposed above results for comparison data. Results support categorical data structure.

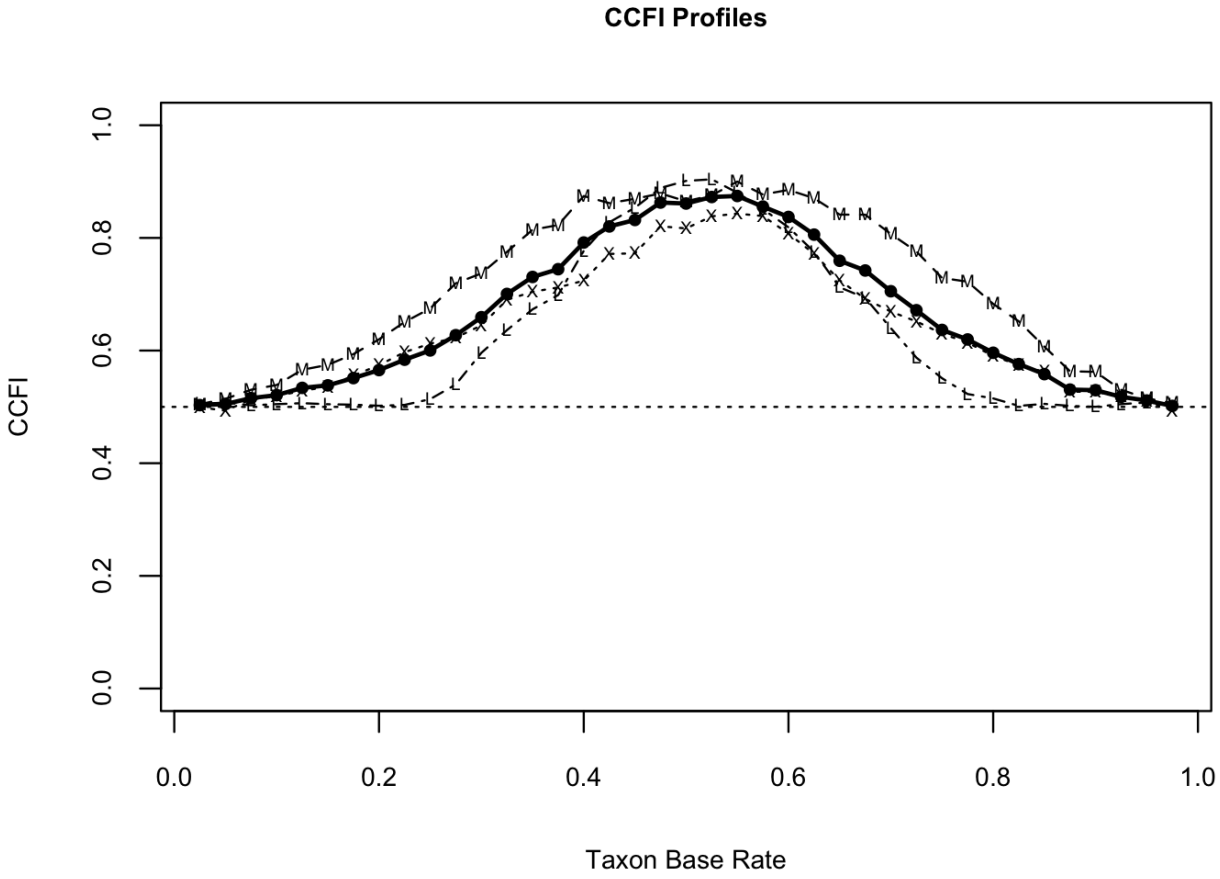

Figure 2. Graphical output from `RunCCFIProfile`; profile of CCFI values averaged across procedures, as well as for MAMBAC (M), MAXEIG (X), and L-Mode (L) procedures. The peak of each fitted curve supports categorical data structure and a taxon base rate of approximately .50.