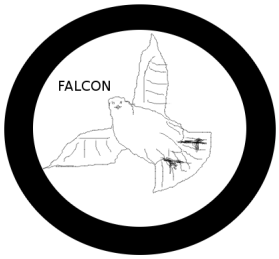
Instructions for using FALCON

Stephen J. Beckett*, Chris A. Boulton, Hywel T. P. Williams

April 15, 2014

College of Life and Environmental Sciences, University of Exeter, Exeter, UK



*author for correspondence: S.J.Beckett@exeter.ac.uk

Contents

1 Introduction **3**

2 How FALCON works **3**

3 Download and installation **5**
3.1 FALCON files and folders 5

4 Running FALCON in MATLAB **7**
4.1 To run FALCON: 7
4.2 Inputs 7
4.2.1 Loading a matrix 8
4.3 Interactive mode 8
4.4 Command line mode 11
4.4.1 Input options 12
4.5 Interpreting output 14

5 Running FALCON in Octave **18**
5.1 To run FALCON: 18
5.2 Known issues with FALCON in Octave 18

6 Running FALCON in R **19**
6.1 To run FALCON: 19
6.2 Inputs 19
6.2.1 Loading a matrix 20
6.3 Interactive mode 20
6.4 Command line mode 24
6.4.1 Input options 24
6.5 Interpreting output 26

7 Extending FALCON's selection of null models and nestedness measures **30**
7.1 FALCON in other programming languages... 30

1 Introduction

FALCON is a free software package for calculating and comparing nestedness in bipartite networks. FALCON is available to download from https://github.com/sjbeckett/FALCON. This document is a practical guide to using FALCON for calculating nestedness. A technical description of the theoretical background and methods used in FALCON is also available (see `FALCON_Manuscript.pdf`). This document is a guide for installing and using FALCON - users are encouraged to read the accompanying technical document to learn more about nestedness and its applications.

FALCON stands for "Framework for Adaptive ensembLes for the Comparison Of Nestedness". Bipartite networks are widely used in science to represent interactions between two kinds of entity. For example, in biology they are often used to represent plant-pollinator interactions, host-virus interactions, or site-species relations (see `FALCON_Manuscript.pdf`). Nestedness is a statistical property of bipartite networks which has been argued to capture various kinds of mechanistic relationship between entities (see `FALCON_Manuscript.pdf`). There are many different ways to measure nestedness in a bipartite network and many different null models can be used to calculate the statistical significance of a nestedness score. FALCON allows users to compare a selection of the most widely used nestedness measures and null models for a given bipartite network. FALCON returns nestedness scores and associated statistical significance (p-values) for a bipartite network given in the form of a binary matrix of connections. The user supplies the input matrix, chooses which nestedness measures and null models to use, and then FALCON does the rest.

FALCON is currently coded for use with MATLAB, Octave and R. MATLAB is proprietary software commonly used in industry and academia. A free clone of MATLAB called Octave is also available for which FALCON has been validated against. R is free software that is also in common use, especially for statistical projects. This document details instructions for how to download and use FALCON in MATLAB, Octave and R. To aid reading this document `folder locations are typed in green`, `variables are typed in red` and `file names are typed in blue`.

2 How FALCON works

FALCON requires a bipartite network to be supplied by the user in the form of a biadjacency matrix - that is, a two-dimensional matrix of values indicating the presence/absence of interactions between two classes of entity. Rows represent nodes of one class of entity and columns represent nodes of the other class of entity, e.g. rows might be plants and columns might be pollinators. The elements can be binary (1's and 0's) indicating whether a connection is present or absent, or real-valued weights representing the strength of interaction between nodes. Input matrices can be loaded or created in MATLAB/Octave/R by various methods. Users must choose whether they want to carry out a binary

analysis of their network, a weighted analysis, or both - this is necessary in order for FALCON to use appropriate null models.

Name	CodeID	Type
Swappable-Swappable (SS)	1	Binary
Fixed-Fixed (FF)	2	Binary
Cored-Cored (CC)	3	Binary
Degreeprobable-Degreeprobable(DD)	4	Binary
Equiprobable-Equiprobable (EE)	5	Binary
Binary Shuffle	-1	Quantitative
Conserve Row Totals (CRT)	-2	Quantitative
Conserve Column Totals (CCT)	-3	Quantitative
Row and Column Totals Average (RCTA)	-4	Quantitative

Table 1: Types of null models included in FALCON

Users can choose a number of different nestedness measures and can specify which null models FALCON should calculate statistical significance against. FALCON can be run in command line mode where all these arguments are submitted on a single line, or in interactive mode which asks users for each of the input arguments separately with explanations about it's relevance. Note that currently this interactive mode is limited to performing nestedness analysis on binary matrices using FALCON's default null models and measures. When all parameters have been chosen, FALCON performs the necessary calculations. Nestedness scores and statistical significances are then returned as output to the user. For full details see `FALCON_Manuscript.pdf`.

Name	Code	Type
NODF	NODF	Binary
WNODF	WNODF	Quantitative
WNODF-REVERSE	WNODF_REVERSE	Quantitative
Manhattan distance (used to calculate $\tau - temperature$)	MANHATTAN_DISTANCE	Binary
Spectral Radius	SPECTRAL_RADIUS	Both
JDM's nestedness	JDMnestedness	Binary
Nestedness temperature	NTC	Binary
Discrepancy	DISCREPANCY	Binary

Table 2: Types of measures included in FALCON

3 Download and installation

If you have not already downloaded FALCON, you can find the main project at https://github.com/sjbeckett/FALCON . GitHub is an online repository that is widely used for sharing and collaborating on coding based projects. You can download FALCON by clicking on the 'Download ZIP' button (which is in the bottom right of the screen at time of writing).

Once the ZIP file has downloaded, move the folder to somewhere where you wish to store the files and then extract the files from their compressed format. You should probably be able to find this in the options list after right clicking the folder. This will extract the various FALCON codes and related documentation in various subfolders under `FALCON`. Within FALCON are two subfolders `Documentation`, which lists various FALCON documentation and `MATLAB`, where the MATLAB software codes used in FALCON are stored.

3.1 FALCON files and folders

The folder `FALCON/MATLAB/` (also see `FALCON/R/`) contains multiple functions in separate files. These are briefly outlined below (see table 3 for a full list of files and dependencies). Inside the `MATLAB` subfolder are other subfolders; `MEASURES` is the folder where the codes to run different ways of measuring nestedness are stored, `NULLS` contains codes to create the different types of null models, `SOLVERS` contains the different types of ensemble generators that can be used in FALCON and `MISC` contains other 'miscellaneous functions' that are used for plotting, generating statistics, sorting matrices and evaluating whether nestedness increases with an increasing nestedness score.

In addition to these subfolders the software folder also includes the file `PERFORM_NESTED_TEST` which is the main file through which to use FALCON. Using the parameters supplied this file calls on the chosen solver, which then iteratively calls on the chosen null models and chosen measure to build an ensemble of null models for which statistics can be generated. There are examples of how to use `PERFORM_NESTED_TEST` in the file `examplescript`. There is also an interactive version of FALCON which asks for user input to questions in `InteractiveMode`. It contains a simple code that should allow for very simple computation of nestedness metrics through user input via questions at the command line.

File	Location	Description	Called from	Calls on
InteractiveMode	MATLAB (or R)	Asks for user input through questions which is sent to PERFOM_NESTED_TEST	-	PERFORM_NESTED_TEST
PERFORM_NESTED_TEST	MATLAB (or R)	Main file for running FALCON	InteractiveMode, examplescript	sortMAT, MEASURES, SOLVERS
examplescript	MATLAB (or R)	Examples of how to use PERFORM_NESTED_TEST	-	PERFORM_NESTED_TEST
MATRIXPLOT	MATLAB (or R)	Plots a bipartite network in matrix form	-	-
FIXEDMETHOD	METHODS	The fixed number ensemble solver	PERFORM_NESTED_TEST	NULLS, performEnsembleStats,plotEnsemble
ADAPTIVEMETHOD	METHODS	The adaptive ensemble solver	PERFORM_NESTED_TEST	NULLS, performEnsembleStats,plotEnsemble
NODF	MEASURES	The NODF measure	PERFORM_NESTED_TEST,NULLS	-
WNODF	MEASURES	The weighted NODF measure	PERFORM_NESTED_TEST,NULLS	-
WNODF_REVERSE	MEASURES	The reverse weighted NODF measure	PERFORM_NESTED_TEST,NULLS	-
SPECTRAL_RADIUS	MEASURES	The spectral radius measure	PERFORM_NESTED_TEST,NULLS	-
MANHATTAN_DISTANCE	MEASURES	The Manhattan distance measure	PERFORM_NESTED_TEST,NULLS	-
JDMnestedness	MEASURES	Nestedness measure based on dissassortativity	PERFORM_NESTED_TEST,NULLS	-
NTC	MEASURES	Nestedness temperature calculator measure	PERFORM_NESTED_TEST,NULLS	-
DISCREPANCY	MEASURES	The discrepancy measure	PERFORM_NESTED_TEST,NULLS	-
CREATEBINNULL1	NULLS	SS binary null model	METHODS	MEASURES, sortMATRIX
CREATEBINNULL2	NULLS	FF binary null model	METHODS	MEASURES, sortMATRIX
CREATEBINNULL3	NULLS	CC binary null model	METHODS	MEASURES, sortMATRIX
CREATEBINNULL4	NULLS	DD binary null model	METHODS	MEASURES, sortMATRIX
CREATEBINNULL5	NULLS	EE binary null model	METHODS	MEASURES, sortMATRIX
CREATEQUANTNULL1	NULLS	Binary shuffled quantitative null model	METHODS	MEASURES, sortMATRIX
CREATEQUANTNULL2	NULLS	Conserve row totals quantitative null model	METHODS	MEASURES
CREATEQUANTNULL3	NULLS	Conserve column totals quantitative null model	METHODS	MEASURES
CREATEQUANTNULL4	NULLS	Row & column total avergae quantitative null model	METHODS	MEASURES
MWWtest	MISC (MATLAB ONLY)	Performs Mann-Whitney U test	METHODS	-
NESTED_UP_OR_DOWN	MISC	Returns whether nestedness increases with increased nestedness score or not	METHODS	MEASURES
performEnsembleStats	MISC	Creates statistics about the measures found in null ensemble	METHODS	-
plotEnsemble	MISC	Plots a histogram of the measures found in null ensemble	METHODS	-
sortMATRIX	MISC	Sorts matrix into most nested configuration	PERFORM_NESTED_TEST,METHODS, NULLS	-
SHAPE_MATRIX	MISC	Creates a set matrix with one's above and zero's below a curve of given weighting	BENCHMARKING	-

Table 3: Outline of different functions and dependencies contained in FALCON. METHODS is shorthand for the functions in the METHODS folder, similarly NULLS is short for functions in NULLS and MEASURES short for functions in MEASURES. See `FALCON_Manuscript.pdf` for references to original papers describing the different nestedness measures and null models.

4 Running FALCON in MATLAB

You will require a copy of MATLAB to run the MATLAB version of FALCON. If you do not have one installed it may be possible to download a trial edition. If you cannot access a version of MATLAB you can try Octave (http://www.gnu.org/software/octave/), which is open source software that mimics much of MATLAB's functionality. See section section 5 for more on using FALCON with Octave.

4.1 To run FALCON:

1. Open MATLAB
2. Navigate to where you have stored the FALCON folder.
3. Navigate through to `FALCON/MATLAB/`

There are two methods for running FALCON: an interactive mode which asks the user for input parameters via a series of questions (code in `InteractiveMode.m`) and a command line mode where input parameters are given a single function call (code in `PERFORM_NESTED_TEST.m`). Examples of the command line mode are given in the file `examplescript.m`, whilst an example of interactive mode is given in section 4.3. The interactive mode (see section 4.3) is designed for ease of use and is intended to offer basic functionality to the casual user. The command line mode (see section 4.4) offers more functionality and is intended for users wishing to test particular measures and/or null models.

4.2 Inputs

FALCON takes six or seven inputs:

1. Input matrix (representing a bipartite network)
2. Choice of binary or quantitative nestedness analysis
3. Choice of sorting to maximise nestedness
4. Choice of nestedness measures
5. Choice of null models
6. Choice of fixed or adaptively chosen size for the null ensemble used to calculate statistical significance
7. Choice of whether to plot distributions of nestedness scores (command-line mode only)

In command-line mode, all six inputs are given as arguments in a function call to `PERFORM_NESTED_TEST.m`. In interactive mode, a series of questions asks the user to set each input in turn.

4.2.1 Loading a matrix

To run FALCON you need to have an input matrix for which you wish to calculate nestedness. Any rectangular matrix with zero and non-zero values can be used. There are various ways to create such a matrix in MATLAB.

You could write the matrix into MATLAB directly as shown in the code snippet below. Here `MY_MATRIX` is the variable name of the matrix being created, the square brackets indicate the enclosed is stored as an array, semi-colons indicate the start of each new row and 0's and 1's separated by spaces are the matrix elements:

```
>> MY_MATRIX= [ 1 0 0 1;0 1 0 1; 0 1 1 1; 0 0 1 1]
```

Entering the above code creates a matrix that looks like this:

$$\begin{matrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{matrix}$$

We use this matrix to illustrate the operation of FALCON in the rest of this document.

Alternatively, if you already have a matrix stored in a spreadsheet it is possible to load it into MATLAB directly. For example, to load a matrix from `test.csv` (a csv file can be created in a Microsoft Excel spreadsheet by choosing to save in the csv format) and save it to a new variable called `MY_MATRIX` you can run the command below:

```
>> MY_MATRIX = dlmread('test.csv')
```

Similar examples are also shown in `examplescript.m`.

There are various other ways to create or import an input matrix. You can search MATLAB's help documentation for 'importing data' to look at other data types that can be imported.

4.3 Interactive mode

Type `InteractiveMode` into the command window and hit enter. This will cause the script in `Interactive.m` to run. This script will ask you questions about the location of the input matrix you wish to assess and about which nestedness measures and null models you want to use. Interactive mode is currently only set up to evaluate nestedness of a single binary matrix (0's and 1's). Quantitative matrices can be analysed in command line mode (see section 4.4). If a non-binary matrix is supplied values not equal to 0 will map to 1's.

Before interactive mode is used you need to create an input matrix - type `restart` to return to the command line if you have not yet created a matrix. Alternatively, if you don't have an input matrix or just want to practise, interactive mode offers you the chance to create a random matrix by typing `generate` and pressing enter as the answer to the first question as shown below.

```
>> InteractiveMode

- Enter the variable name of the matrix you wish to test.  If a
variable is not yet assigned type "restart" and create a matrix
variable before continuing, or type "generate" to generate a
random matrix to test the code:

generate
```

If you have previously created a matrix to test type in the name of the variable containing your matrix (here we refer to this matrix as `MY_MATRIX`) and press enter.

```
>> InteractiveMode

- Enter the variable name of the matrix you wish to test.  If a
variable is not yet assigned type "restart" and create a matrix
variable before continuing, or type "generate" to generate a
random matrix to test the code:

MY_MATRIX
```

FALCON will now ask questions about how to analyse the nestedness of the input matrix, which should be answered as directed. The questions and some example answers are shown below. To move onto the next question you must press enter once you have entered your input. If you enter a command incorrectly, you will be prompted to correct your input. However, if you make a mistake (enter a valid but incorrect value) you will have to restart interactive mode after answering all the questions. This example specifies that `MY_MATRIX` should be measured by NODF and tested against the degreeprobable rows , degreeprobable columns (DD) null model using the adaptive ensemble solver:

```
- Do you wish to use the current matrix ordering or sort to
maximise nestedness?  The way the matrix is ordered makes a
difference to the NODF, DISCREPANCY and MANHATTAN DISTANCE
measures.  This choice is applied both to the input and null
model matrices.  Select 1 to sort or 0 to use current matrix
ordering:

1

- Which measure do you wish to use to measure nestedness?  If
unsure we suggest using NODF, one of the more popular nestedness
measures.  Select 1 for NODF , 2 for SPECTRAL RADIUS, 3 for
MANHATTAN DISTANCE (used to calculate TAU-TEMPERATURE), 4 for
JDMnestedness, 5 for NTC or 6 for DISCREPANCY.

1

- Do you want to add another measure?  (Y/N)n

- Which null model do you wish to use to measure nestedness?  If
unsure you could try using 1, where size and fill are conserved
but uniformly randomly shuffled.  Select 1 for SS , 2 for FF, 3
for CC, 4 for DD, 5 for EE.

4

- Do you want to add another null model?  (Y/N)N

- Are you happy to use the adaptive solver?  Adaptive solver (Y),
Fixed solver (N).

Y

Now performing the calculations on MY_MATRIX to find the NODF
score(s); and test it against null model(s) DD using the ADAPTIVE
solver.  Hold on whilst your output is calculated!
```

Once these answers have been supplied, the interactive mode version of FALCON passes these inputs and the input matrix to PERFORM_NESTED_TEST.m to calculate the nestedness of the input matrix and how significant the nestedness of this matrix is in comparison to those created by the null model. It is worth noting that it is possible to call measures multiple times - though you probably won't wish to do this!

After FALCON finishes the calculations, the user is shown the nestedness

of the input matrix relative to the distribution of nestedness scores from the null ensemble in a figure (e.g. figure 1). Outputs are also given numerically in the command window(see section 4.5 for more) and stored in a data object called `ind`. The data object `ind` has fields containing various kinds of information and output from the calculations, which can be accessed by typing `ind` at the command line. For example, `ind.Matrix.Matrix` contains the matrix that was entered by the user, while `ind.NestedConfig` contains the most nested configuration of the input matrix (by sorting by row and column degree) and the corresponding row and column indexes swaps from the input matrix, and `ind.Bin_t4` shows the statistics associated with the null ensemble, where the 4 indicates that null model 4 (DD) was used. More information on how output can be interpreted is given below.

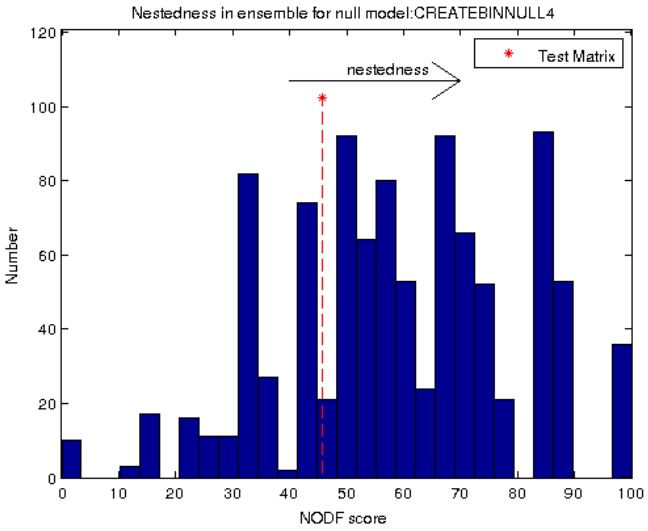


Figure 1: Output histogram of null model nestedness scores for `MY_MATRIX` measured by NODF using the DD null model. Red line indicates the nestedness score of `MY_MATRIX`.

4.4 Command line mode

Command-line mode bypasses the question-and-answer configuration in interactive mode and calls `PERFORM_NESTED_TEST.m` directly. This is useful when you are familiar with FALCON's operation or if you want to embed FALCON within another script. Command line mode also offers more input options than interactive mode. This section describes how to use `PERFORM_NESTED_TEST.m` in command line mode.

4.4.1 Input options

PERFORM_NESTED_TEST.m takes user options as input and uses them to work out which nestedness measure, which null models and which ensemble method should be used. After creating or loading an input matrix (here labelled MATRIX but could take any name) you should choose a variable name to save results to (here using store) then you can run the function as follows:

```
>> store = PERFORM_NESTED_TEST(MATRIX,binary,sortVar,MEASURE,nulls,ensNum,plotON);
```

where store is a data object that will contain output data.
The inputs are:

- MATRIX - the name of the matrix to be tested.
- binary - whether the nestedness analysis should be binary, quantitative, or both.
- sortVar - whether to order row and columns to achieve maximal nestedness of initial and null matrices (this matters to some measures).
- MEASURE - the name of the nestedness measure to be used.
- nulls - which null models are to be used
- ensNum - which ensemble method (adaptive or fixed) to use.
- plotON - whether the user would like to view a visual representation of output.

The binary option can take values 0, 1 or 2 (see below):

$$binary = \begin{cases} 0 & quantitative \\ 1 & binary \\ 2 & both \end{cases}$$

sortVar inidcates whether to use the users initial matrix ordering of rows or columns, which may be useful if for example nestedness is being tested along some type of ecological or environmental gradient, or whether the user wishes their input to be optimally packed for nestedness analaysis. For binary matrices sorting arranges rows and columns in degree decreasing order, where the degree is the number of precneses (ones) in a particular row or column. We note that the ordering of the matrix only makes a difference to some of the nestedness measures we consider here - as spectral radius and the JDMnestedness measure are invariant to matrix row and column ordering and the nestedness temperature calculator contains its own packging algorithm. We order quantitative matrices in the same fashion, but additional sorting is made for rows and columns that share the same degree. Rows (columns) with the greatest number of overlapping

interactions that are greater than other rows (columns), take precedent. For any rows (columns) that share the same number of greater overlapping elements precedence is then decided by the total row (column) totals. To use nestedness as a statistical property of the matrix we recommend using the sorting algorithm. To use the sorting algorithm specify that sortVar = 1 , else use sortVar=0.

MEASURE is the name of the nestedness measure to be used, given as a string in curly braces. The string is used to call functions saved in the folder FALCON/MATLAB/MEASURES . If there is a measure that we have not coded but you wish to study, you can code it yourself, save it here, and then call it using the MEASURE argument. It is important that the MEASURE argument is a string and its components are enclosed uses curly braces e.g. MEASURE = {'NODF'} or MEASURE = {'NODF','JDMnestedness'}. The measures of nestedness currently included in FALCON are given in table 2.
The input nulls decides which null models are to be used to calculate statistical significance of the nestedness score measured for the input matrix. The choice of null model is tied to whether the user wishes to test the matrix in a binary (unweighted interactions) or quantitative (weighted interactions) sense. The different values nulls can take are shown by the CodeID in table 1. Many null models can be tested with one call to the PERFORM_NESTED_TEST.m function by using a vector e.g. nulls = [1 3] would run the SS and CC null models . Alternatively, all possible null models (according to whether the input is being treated in a binary or quantitative context) can be called by setting nulls=[]. It is important the user chooses null models that are appropriate with their choice of nestedness measure i.e. measuring WNODF against a binary null model is non-informative. Outputs are stored in separate objects within store for each of the different null models (see section 4.5).
Each null model is encoded with a number (positive for binary nulls and negative for quantitative nulls) and new null models can be added to the folder FALCON/MATLAB/NULLS . In order to use your own null model you will also need to change some information in the PERFORM_NESTED_TEST.m function, this is outlined in section 7.

The setting of variable ensNum tells PERFORM_NESTED_TEST.m which ensemble method (adaptive or fixed) to run and an option to go with it. To run the adaptive method ensNum should be set as empty (ensNum=[]). However, if ensNum is not empty, the fixed method is called and the value it takes specifies the exact number of null models should be used in the ensemble to test for statistical significance of the nestedness found.

The final input argument plotON is used to indicate whether the user would like to view a histogram of the measures computed in the null ensemble where 1 indicates that the plot should be made and 0 indicates it should not. Plotting arguments can be found in the FALCON/MATLAB/MISC folder in plotEnsemble.m .

The following command would perform the same nested analysis as that described in the Interactive Mode section above.:

```
>> ind = PERFORM_NESTED_TEST(MATRIX,1,1,{'NODF'},4,[],1);
```

You should look at examplescripts.m for some examples of how to use FALCON in this way.

4.5 Interpreting output

Once you have run one of the above nestedness scripts you will be supplied with some sort of output that will look something like this:

```
ind =
   binary:  1
   sorting:  1
   MEASURE: {'NODF'}
   nulls:  4
   ensNum:  []
   plot:  1
   Matrix:  [1x1 struct]
   NestedConfig:  [1x1 struct]
   Bin_t4:  [1x1 struct]
   SignificanceTableSummary:  [0 0 1 0 0]

ind.Matrix =
   Matrix:  [1 0 0 1; 0 1 0 1; 0 1 1 1; 0 0 1 1 ]
   fill:  9
   connectance:  0.5625

ind.NestedConfig =
   DegreeMatrix:  [1 1 1 0; 1 0 0 1; 1 1 0 0; 1 0 1 0]
   Degreeindex_rows:  [3 1 2 4]
   Degreeindex_cols:  [4 2 3 1]

ind.Bin_t4 =
   EnsembleSize:  1000
   SignificanceTable:  [0 0 1 0 0]
   measures:  [1x1 struct]

ind.Bin_t4.measure{1} =
   MEASURE: 'NODF'
   NANcount:  0
```

```
Measure:  45.8333
pvalue:  0.7470
pvalueCorrected:  0
Mean:  58.9788
StandardDeviation:  20.4381
sampleZscore:  -0.6432
Median:  58.3333
minimum:  0
maximum:  100
NormalisedTemperature:  0.7771
NestednessUpOrDown:  'Up'
```

What does it mean? `ind` is where the results of FALCON are stored from section 4.3 (`store` is the corresponding variable for the examples in section 4.4). This object contains the parameters used in calling `PERFORMING_NESTED_TEST.m` (`ind.binary`, `ind.MEASURE`, `ind.nulls` and `ind.ensNum`) as described in section 4.4. In addition it contains the input matrix information in `ind.Matrix` with the initial matrix, it's fill (number of interactions present) and its connectance (proportion of all possible interactions that are present, i.e. fill divided by the product of the numbers of rows and columns), the nested configuration of this matrix (`ind.NestedConfig.DegreeMatrix`) and the orderings of rows and columns from the input matrix to achieve this (`ind.Degreeindex_rows` and `ind.Degreeindex_cols` respectively). See table 4 for a description of the output variables for the output from significance tests (here just `ind.Bin_t4`, where Bin stands for binary and t4 stands for test 4, the 4 indicating that null model 4 (DD) was used). It should be noted that the `NormalisedTemperature` calculated for the Manhattan distance is the $\tau-temperature$, where a $\tau-temperature$ greater than 1 indicates it is less nested than expected in comparison to the null model, whilst a $\tau - temperature$ less than 1 indicates it is more nested than expected according to the null model. This is due to more nested matrices having a lower Manhattan distance due to the matrix more likely to appear upper triangular - the shape (for a given number of elements) that will minimize Manhattan distance. This method of measuring nestedness therefore depends on whether the amount of nestedness decreases or increases with increasing score. Thus the above definitions for `NormalisedTemperature` defined for $\tau - temperature$ are reversed when looking at NODF, spectral radius or JDM nestedness (i.e. $\tau - temperature < 1$ indicates the input was less nested than expected) . However, these differences in the way these measures are defined are accounted for when considering the p-value. `pvalue` always represents the probability of attaining a more nested matrix than the one under consideration. There is a `SignificanceTable` within each null model, where each row corresponds to a different measure-null combination. The first column indicates whether the input matrix was found to be significantly nested at a level of $p<=0.001$, the second column represents whether the input was found to be significantly nested at a level of $p<0.05$, the third column indicates the input was not significant ($0.05<p<0.95$), the fourth column indicates the input was significantly

not nested at a level of p>0.95 and the final column indicates the input was significantly not nested at a level of p>=0.999. The SignificanceTableSummary is a sum of all the SignificanceTable's and allows an at a glance idea of how often the input matrix was found to be significantly nested or not.

Each binary null model set of results is stored in the object store.Bin_t# and every quantitative null model set of results is stored in the object ind.Qua_t#, where the # refers to the null models identifying number set in PERFORM_NESTED_TEST.m (the negative numbers used for quantitative nulls are shown as positives here). In addition the input matrix is saved as ind.Matrix.Matrix and the ordering of rows and columns used to compute nestedness are saved as ind.NestedConfig.Degreeindex_rows and ind.NestedConfig.Degreeindex_cols respectively. This shows rows and columns sorted into their most likely nested configuration by sorting them in terms of row and column degree respectively and removing rows and columns composed entirely of zero elements. This sorted matrix can be used by typing ind.NestedConfig.DegreeMatrix. Then the command MATRIXPLOT can be used to visualise this configuration as `MATRIXPLOT(ind.NestedConfig.DegreeMatrix)` .

Name	Description
fill	Number of precenses in the input matrix
connectance	$\frac{fill}{rows \times columns}$
EnsembleNum	Number of null models used in ensemble for significance testing
NANcount	Number of null models in ensemble that returned nestedness score with not a number. Want 0 of these!
EnsembleNum	Number of null models used in ensemble for significance testing
pvalue	The p-value found by significance testing (proportion of null models returning a higher nestedness score i.e the probability of attaining a higher nestedness score than Measure)
pvalueCorrected	Indicates (1) the case p=0, when p is modified to be p<1/EnsembleSize.
Mean	The mean average nestedness score returned by the null model ensemble
Median	The median nestedness score returned by the null model ensemble
minimum	The minimum of the measured value in the null ensmble
maximum	The maximum of the measured values in the null ensemble
NormalisedTemperature	The normalised nestedness temperature (= Measure/Mean)
StandardDeviation	The standard deviation of the test matrix from the ensemble
sampleZscore	The sample Z-score of the matrix against the ensemble
Measure	The nestedness score of the matrix itself against chosen measure
NestednessUpOrDown	Displays whether increased nestedness increases with increasing measure ('Up') or decreasing measure ('Down')

Table 4: Description of FALCON output

5 Running FALCON in Octave

We have tested FALCON's functionality in Octave v3.6.1 and found that most things work as intended. However, not all parts of FALCON are available. In earlier versions of Octave, functionality may be reduced further. Octave can be downloaded here: http://www.gnu.org/software/octave/.

5.1 To run FALCON:

Due to the similarity between MATLAB and Octave, we recommend Octave users of FALCON to follow the instructions given for running FALCON in MATLAB can be run on it in a very similar way to MATLAB section §4.

5.2 Known issues with FALCON in Octave

- We encountered difficulty when trying to run the NTC.m command as many of it's functions are MATLAB specific. We do not offer a fix for this problem and ask you to avoid using this function if using Octave. However, if you wish to use this function it is available in the *vegan* package for R as the function *nestedtemp*, which is the code we based our NTC.m function on and we recommend you to use this.

If you encounter any other problems whilst using Octave, please let us know!

6 Running FALCON in R

You will require a copy of R to run FALCON in R: it can downloaded from (http://www.r-project.org). In order to run nestedness analysis on the nested temperature calculator you will require a version 2.15.0 or later as the function used to run these calculations (`nestedtemp`) is taken from the vegan package (http://cran.r-project.org/web/packages/vegan/index.html). If you wish to use R with a GUI we recommend using RStudio (https://www.rstudio.com/).

6.1 To run FALCON:

1. Open R
2. Navigate to where you have stored the FALCON folder (this can be done using the setwd() function)
3. Navigate through to `FALCON/R/`

There are two methods for running FALCON: an interactive mode which asks the user for input parameters via a series of questions (code in `InteractiveMode.R`) and a command line mode where input parameters are given a single function call (code in `PERFORM_NESTED_TEST.R`). Examples of the command line mode are given in the file `examplescript.R`, whilst an example of interactive mode is given in section 6.3 . The interactive mode (see section 6.3) is designed for ease of use and is intended to offer basic functionality to the casual user. The command line mode (see section 6.4) offers more functionality and is intended for users wishing to test particular measures and/or null models.

6.2 Inputs

FALCON takes six or seven inputs:

1. Input matrix (representing a bipartite network)
2. Choice of binary or quantitative nestedness analysis
3. Choice of sorting to maximise nestedness
4. Choice of nestedness measures
5. Choice of null models
6. Choice of fixed or adaptively chosen size for the null ensemble used to calculate statistical significance
7. Choice of whether to plot distributions of nestedness scores (command-line mode only)

In command-line mode, all six inputs are given as arguments in a function call to `PERFORM_NESTED_TEST.R`. In interactive mode, a series of questions asks the user to set each input in turn.

6.2.1 Loading a matrix

To run FALCON you need to have an input matrix for which you wish to calculate nestedness. Any rectangular matrix with zero and non-zero values can be used. There are various ways to create such a matrix in R.

You could write the matrix into R directly as shown in the code snippet below. Here `MY_MATRIX` is the variable name of the matrix being created, the list of 0's and 1's separated by commas are the matrix elements (read columns in top-bottom, then right) which are aligned into a matrix with nrow rows:

```
> MY_MATRIX <- matrix( c(1,0,0,0,0,1,1,0,0,0,1,1,1,1,1) , nrow=4 )
```

Entering the above code creates a matrix that looks like this:

$$\begin{matrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{matrix}$$

We use this matrix to illustrate the operation of FALCON in the rest of this document.

Alternatively, if you already have a matrix stored in a spreadsheet it is possible to load it into R directly. Although it is possible to read in from a Microsoft Excel spreadsheet directly it is much easier to save the sheet to the .csv format and read in this file. Then `test.csv` can be read into R as follows:

```
> MY_MATRIX = read.csv('test.csv')
```

Similar examples are also shown in `examplescript.R`.

There are various other ways to create or import an input matrix. You can search R's help documentation for 'importing data' to look at other data types that can be imported.

6.3 Interactive mode

In order to run interactive mode the code instructions need to be sourced (or read into) R. To start interactive mode type the following command:

```
source('InteractiveMode.R')
```

This script will ask you questions about the location of the input matrix you wish to assess and about which nestedness measures and null models you want to use. Interactive mode is currently only set up to evaluate nestedness of a single binary matrix (0's and 1's). Quantitative matrices can be analysed in command line mode (see section 6.4). If a non-binary matrix is supplied values not equal to 0 will map to 1's.

Before interactive mode is used you need to create an input matrix - type `restart` to return to the command line if you have not yet created a matrix. Alternatively, if you don't have an input matrix or just want to practise, interactive mode offers you the chance to create a random matrix by typing `generate` and pressing enter as the answer to the first question as shown below.

```
> source('InteractiveMode.R')

- Enter the variable name of the matrix you wish to test.  If a
variable is not yet assigned type "restart" and create a matrix
variable this before continuing, or type "generate" to generate a
random matrix to test the code:

generate
```

If you have previously created a matrix to test type in the name of the variable containing your matrix (here we refer to this matrix as `MY_MATRIX`) and press enter.

```
> source('InteractiveMode.R')

- Enter the variable name of the matrix you wish to test.  If a
variable is not yet assigned type "restart" and create a matrix
variable this before continuing, or type "generate" to generate a
random matrix to test the code:

MY_MATRIX
```

FALCON will now ask questions about how to analyse the nestedness of the input matrix, which should be answered as directed. The questions and some example answers are shown below. To move onto the next question you must press enter once you have entered your input. If you enter a command incorrectly, you will be prompted to correct your input. However, if you make a mistake (enter a valid but incorrect value) you will have to restart interactive mode after answering all the questions. This example specifies that `MY_MATRIX` should be measured by NODF and tested against an degreeprobable rows / degreeprobable columns (DD) null model using the adaptive ensemble solver:

Once these answers have been supplied, the interactive mode version of FALCON passes these inputs and the input matrix to `PERFORM_NESTED_TEST.R` to calculate the nestedness of the input matrix and how significant the nestedness of this matrix is in comparison to those created by the null model. It is worth noting that it is possible to call measures multiple times - though you probably won't wish to do this!

```
- Do you wish to use the current matrix ordering or sort to
maximise nestedness?  The way the matrix is ordered makes a
difference to the NODF, DISCREPANCY and MANHATTAN DISTANCE
measures.  This choice is applied both to the input and null
model matrices.  Select 1 to sort or 0 to use current matrix
ordering:

1

- Which measure do you wish to use to measure nestedness?  If
unsure we suggest using NODF, one of the more popular nestedness
measures.  Select 1 for NODF , 2 for SPECTRAL RADIUS, 3 for
MANHATTAN DISTANCE (used to calculate TAU-TEMPERATURE), 4 for
JDMnestedness, 5 for NTC or 6 for DISCREPANCY.

1

- Do you want to add another measure?  (Y/N)n

- Which null model do you wish to use to measure nestedness?  If
unsure we suggest using 1, where size and fill are conserved but
uniformly randomly shuffled.  Select 1 for SS , 2 for FF, 3 for
CC, 4 for DD, 5 for EE.

4

- Do you want to add another null model?  (Y/N)n

- Are you happy to use the adaptive method?  Adaptive method (Y),
Fixed method (N).

y

Now performing the calculations on MY_MATRIX to find the NODF
score(s); and test it against null model(s) DD using the ADAPTIVE
method.
```

After FALCON finishes the calculations, the user is shown the nestedness of the input matrix relative to the distribution of nestedness scores from the null ensemble in a figure (e.g. figure 2). Outputs are also given numerically in the command window(see section 6.5 for more) and stored in a data object called `ind`. The data object `ind` has fields containing various kinds of information and output from the calculations, which can be accessed by typing `ind` at the command line. For example, `ind$Matrix$Matrix` contains the matrix that was entered by the user, while `ind$NestedConfig` contains the most nested configuration of the input matrix (by sorting by row and column degree) and the corresponding row and column indexes swaps from the input matrix, and `ind$Bin_t4` shows the statistics associated with the null ensemble, where the 4 indicates that null model 4 (DD) was used. More information on how output can be interpreted is given below.

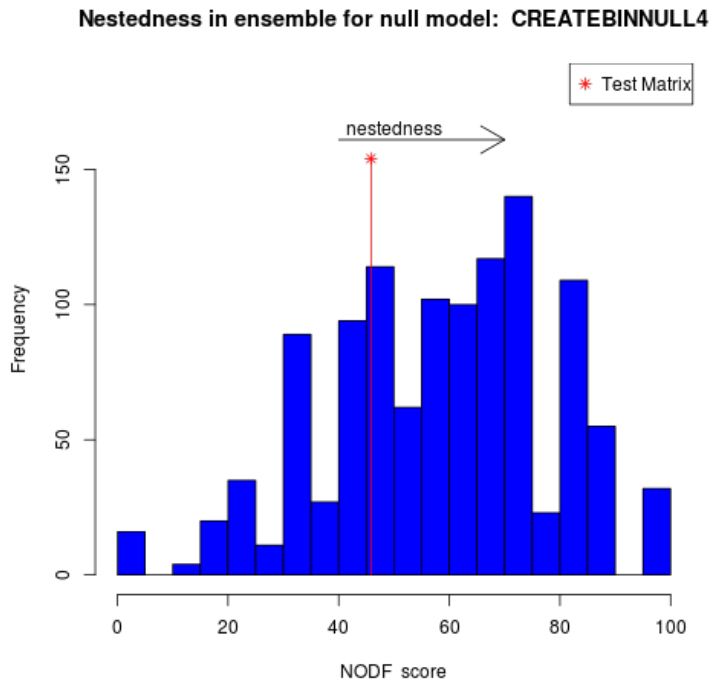


Figure 2: Output histogram of null model nestedness scores for `MY_MATRIX` measured by NODF using the DD null model. Red line indicates the nestedness score of `MY_MATRIX`. These scripts were run in R.

6.4 Command line mode

Command-line mode bypasses the question-and-answer configuration in interactive mode and calls PERFORM_NESTED_TEST.R directly. This is useful when you are familiar with FALCON's operation or if you want to embed FALCON within another script. Command line mode also offers more input options than interactive mode. This section describes how to use PERFORM_NESTED_TEST.R in command line mode. As with interactive mode the first thing you should do is source the code, using `source('PERFORM_NESTED_TEST.R')` .

6.4.1 Input options

PERFORM_NESTED_TEST.R takes user options as input and uses them to work out which nestedness measure, which null models and which solver should be used. After creating or loading an input matrix (here labelled MATRIX but could take any name) you should choose a variable name to save results to (here using store) then you can run the function as follows:

```
> store <- PERFORM_NESTED_TEST(MATRIX,binary,sortVar,MEASURE,nulls,ensNum,plotON)
```

where store is a data object that will contain output data.
The inputs are:

- MATRIX - the name of the matrix to be tested.
- binary - whether the nestedness analysis should be binary, quantitative, or both.
- sortVar - whether to order row and columns to achieve maximal nestedness of initial and null matrices (this matters to some measures).
- MEASURE - the name of the nestedness measure to be used.
- nulls - which null models are to be used
- ensNum - which solver (adaptive or fixed) to use.
- plotON - whether the user would like to view a visual representation of output.

The binary option can take values 0, 1 or 2 (see below):

$$binary = \begin{cases} 0 & quantitative \\ 1 & binary \\ 2 & both \end{cases}$$

sortVar inidcates whether to use the users initial matrix ordering of rows or columns, which may be useful if for example nestedness is being tested along

some type of ecological or environmental gradient, or whether the user wishes their input to be optimally packed for nestedness analaysis. For binary matrices sorting arranges rows and columns in degree decreasing order, where the degree is the number of precneses (ones) in a particular row or column. We note that the ordering of the matrix only makes a difference to some of the nestedness measures we consider here - as spectral radius and the JDMnestedness measure are invariant to matrix row and column ordering and the nestedness temperature calculator contains its own packging algorithm. We order quantitative matrices in the same fashion, but additional sorting is made for rows and columns that share the same degree. Rows (columns) with the greatest number of overlapping interactions that are greater than other rows (columns), take precedent. For any rows (columns) that share the same number of greater overlapping elements precedence is then decided by the total row (column) totals. To use nestedness as a statistical property of the matrix we recommend using the sorting algorithm. To use the sorting algorithm specify that sortVar = 1 , else use sortVar=0.

MEASURE is the name of the nestedness measure(s) to be used, given as a string list. The string(s) is used to call functions saved in the folder FALCON/R/MEASURES . If there is a measure that we have not coded but you wish to study, you can code it yourself, save it here, and then call it in the same way as other measures. It is important that the MEASURE argument is a string list and can be used as e.g. MEASURE = 'NODF' for a single measure or MEASURE = c('NODF','JDMnestedness') for multiple measures. The measures of nestedness currently included in FALCON are given in table 2.
The input nulls decides which null models are to be used to calculate statistical significance of the nestedness score measured for the input matrix. The choice of null model is tied to whether the user wishes to test the matrix in a binary (unweighted interactions) or quantitative (weighted interactions) sense. The different values nulls can take are shown by the CodeID in table 1. Many null models can be tested with one call to the PERFORM_NESTED_TEST.R function by using a list e.g. nulls = c(1, 3) would run the SS and CC null models . Alternatively, all possible null models (according to whether the input is being treated in a binary or quantitative context) can be called by setting nulls = c(). It is important the user chooses null models that are appropriate with their choice of nestedness measure i.e. measuring WNODF against a binary null model is non-informative. Outputs are stored in separate objects within store for each of the different null models (see section 4.5).
Each null model is encoded with a number (positive for binary nulls and negative for quantitative nulls) and new null models can be added to the folder FALCON/R/NULLS . In order to use your own null model you will also need to change some information in the PERFORM_NESTED_TEST.R function, this is outlined in section 7.

The setting of variable ensNum tells PERFORM_NESTED_TEST.R which solver (adaptive or fixed) to run and an option to go with it. To run the adaptive solver ensNum should be set as empty (ensNum = c()). However, if ensNum is not

empty, the fixed solver is called and the value it takes specifies the exact number of null models should be used in the ensemble to test for statistical significance of the nestedness found.

The final input argument `plotON` is used to indicate whether the user would like to view a histogram of the measures computed in the null ensemble where 1 indicates that the plot should be made and 0 indicates it should not. Plotting arguments can be found in the `FALCON/R/MISC` folder in `plotEnsemble.R` .

You should look at `examplescripts.R` for some examples of how to use FALCON in this way.

6.5 Interpreting output

Once you have run one of the above nestedness scripts you will be supplied with some sort of output that will look something like this:

```
> ind
$binary [1] 1
$sorting [1] 1
$MEASURE [1] "NODF"
$nulls [1] 4
$plot [1] 1
$Matrix $Matrix$Matrix
  [,1] [,2] [,3] [,4]
[1,] 1 0 0 1
[2,] 0 1 0 1
[3,] 0 1 1 1
[4,] 0 0 1 1
$Matrix$fill [1] 9
$Matrix$connectance [1] 0.5625
$NestedConfig $NestedConfig$DegreeMatrix
  [,1] [,2] [,3] [,4]
[1,] 1 1 1 0
[2,] 1 0 0 1
[3,] 1 1 0 0
[4,] 1 0 1 0
$NestedConfig$Degreeindex_rows [1] 3 1 2 4
$NestedConfig$Degreeindex_cols [1] 4 2 3 1
$Bin_t4 $Bin_t4$EnsembleSize [1] 1150
$Bin_t4$SignificanceTable
  [,1] [,2] [,3] [,4] [,5]
[1,] 0 0 1 0 0
$Bin_t4$NODF $Bin_t4$NODF$MEASURE [1] "NODF"
$Bin_t4$NODF$NANcount [1] 0
$Bin_t4$NODF$Measure [1] 45.83333
```

```
$Bin_t4$NODF$pvalue [1] 0.7426087
$Bin_t4$NODF$pvalueCorrected [1] 0
$Bin_t4$NODF$Mean [1] 58.64596
$Bin_t4$NODF$StandardDeviation [1] 20.34613
$Bin_t4$NODF$sampleZscore [1] -0.6297331
$Bin_t4$NODF$Median [1] 58.3333
$Bin_t4$NODF$minimum [1] 0
$Bin_t4$NODF$maximum [1] 100
$Bin_t4$NODF$NormalisedTemperature [1] 0.7815258
$Bin_t4$NODF$NestednessUpOrDown [1] "Up"
$Bin_t4$NODF$SignificanceTable
[1] 0 0 1 0 0
$SignificanceTableSummary [1] 0 0 1 0 0
```

What does it mean? `ind` is where the results of FALCON are stored from section 4.3 (`store` is the corresponding variable for the examples in section 4.4). This object contains the parameters used in calling `PERFORMING_NESTED_TEST.R` (`ind$binary`, `ind$MEASURE`, `ind$nulls` and `ind$ensNum`) as described in section 4.4. In addition it contains the input matrix information in `ind$Matrix` with the initial matrix, it's fill (number of interactions present) and its connectance (proportion of all possible interactions that are present, i.e. fill divided by the product of the numbers of rows and columns), the nested configuration of this matrix (`ind$NestedConfig$DegreeMatrix`) and the orderings of rows and columns from the input matrix to achieve this (`ind$Degreeindex_rows` and `ind$Degreeindex_cols` respectively). See table 5 for a description of the output variables for the output from significance tests (here just `ind$Bin_t4`, where Bin stands for binary and t4 stands for test 4, the 4 indicating that null model 4 (DD) was used). It should be noted that the `NormalisedTemperature` calculated for the Manhattan distance is the $\tau - temperature$, where a $\tau - temperature$ greater than 1 indicates it is less nested than expected in comparison to the null model, whilst a $\tau - temperature$ less than 1 indicates it is more nested than expected according to the null model. This is due to more nested matrices having a lower Manhattan distance due to the matrix more likely to appear upper triangular - the shape (for a given number of elements) that will minimize Manhattan distance. This method of measuring nestedness therefore depends on whether the amount of nestedness decreases or increases with increasing score. Thus the above definitions for `NormalisedTemperature` defined for $\tau - temperature$ are reversed when looking at NODF, spectral radius or JDM nestedness (i.e. $\tau - temperature < 1$ indicates the input was less nested than expected) . However, these differences in the way these measures are defined are accounted for when considering the p-value. `pvalue` always represents the probability of attaining a more nested matrix than the one under consideration. There is a `SignificanceTable` within each null model, where each row corresponds to a different measure-null combination. The first column indicates whether the input matrix was found to be significantly nested at a level of p<=0.01, the second column represents whether the input was found to be significantly

nested at a level of p<0.05, the third column indicates the input was not significant (0.05<p<0.95), the fourth column indicates the input was significantly not nested at a level of p>0.95 and the final column indicates the input was significantly not nested at a level of p>=0.99. The SignificanceTableSummary is a sum of all the SignificanceTable's and allows an at a glance idea of how often the input matrix was found to be significantly nested or not.

Each binary null model set of results is stored in the object store$Bin_t# and every quantitative null model set of results is stored in the object ind$Qua_t#, where the # refers to the null models identifying number set in PERFORM_NESTED_TEST.R (the negative numbers used for quantitative nulls are shown as positives here). In addition the input matrix is saved as ind$Matrix$Matrix and the ordering of rows and columns used to compute nestedness are saved as ind$NestedConfig$Degreeindex_rows and ind$NestedConfig$Degreeindex_cols respectively. This shows rows and columns sorted into their most likely nested configuration by sorting them in terms of row and column degree respectively and removing rows and columns composed entirely of zero elements. This sorted matrix can be used by typing ind$NestedConfig$DegreeMatrix. Then the command MATRIXPLOT (after sourcing it - source('MATRIXPLOT.R')) can be used to visualise this configuration as MATRIXPLOT(ind$NestedConfig$DegreeMatrix) .

Name	Description
fill	Number of precenses in the input matrix
connectance	$\frac{fill}{rows \times columns}$
EnsembleNum	Number of null models used in ensemble for significance testing
NANcount	Number of null models in ensemble that returned nestedness score with not a number. Want 0 of these!
EnsembleNum	Number of null models used in ensemble for significance testing
pvalue	The p-value found by significance testing (proportion of null models returning a higher nestedness score i.e the probability of attaining a higher nestedness score than Measure)
pvalueCorrected	Indicates (1) the case p=0, when p is modified to be p<1/EnsembleSize.
Mean	The mean average nestedness score returned by the null model ensemble
Median	The median nestedness score returned by the null model ensemble
minimum	The minimum of the measured value in the null ensmble
maximum	The maximum of the measured values in the null ensemble
NormalisedTemperature	The normalised nestedness temperature (= Measure/Mean)
StandardDeviation	The standard deviation of the test matrix from the ensemble
sampleZscore	The sample Z-score of the matrix against the ensemble
Measure	The nestedness score of the matrix itself against chosen measure
NestednessUpOrDown	Displays whether increased nestedness increases with increasing measure ('Up') or decreasing measure ('Down')

Table 5: Description of FALCON output

7 Extending FALCON's selection of null models and nestedness measures

You may also wish to introduce new functionality to the FALCON package, for example by including additional null models or additional measures of nestedness. This will include writing a function and saving it to the appropriate folder, where the arguments should be the same as given for the other functions there which we recommend using as a template. `SPECTRAL_RADIUS` would make a good nestedness measure template, whilst `CREATEBINNULL5` would make a good null model template. In the case of new null models being added to FALCON you should also modify `PERFORM_NESTED_TEST` .

This file has information about which types of measures and which null models are appropriate for each other based on whether they are binary or quantitative in nature.

If you are adding extra null models, you need to change several parts of `PERFORM_NESTED_TEST` :

- The variables `BnullNumber` and `QnullNumber` near the top in the commented REFERENCES section which represent the number of available binary and quantitative null models respectively and is used to error check input arguments.
- Adding code to tell the function to run your null model and store it in an appropriate object in the commented PERFORM TESTS section in a similar format to the current implementation.

If you extend FALCON, please let us know about it! We are keen to continue to develop FALCON by adding new functions, so we might want to include your additions in later versions.

7.1 FALCON in other programming languages...

If you'd be interested in contributing to the development of FALCON for other languages please get in touch!