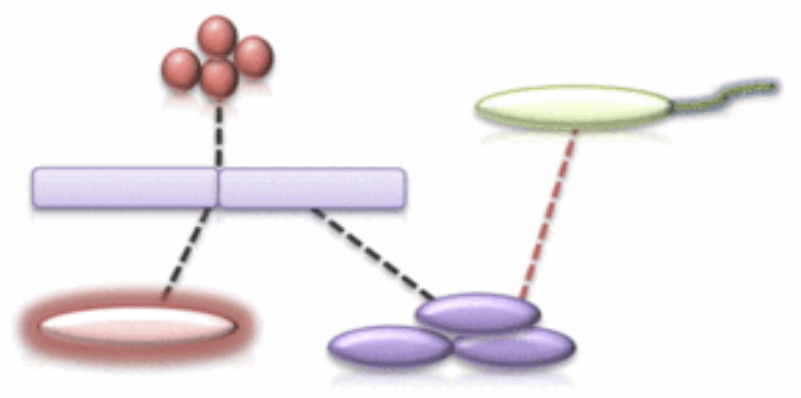
CoNet documentation

Documentation for the Cytoscape plugin CoNet.

CoNet computes significant cooccurrence or mutual exclusion between items (matrix rows), whose presence/absence or abundance was observed repeatedly (matrix columns) and visualize the result as a network.



In case of questions, contact Karoline Faust:
karoline.faust@vib-vub.be

This document has been generated from html with prince.

Index

Main menu.

CoNet is a versatile tool to compute co-occurrences and mutual exclusions between items. A basic run looks something like this:

1. Open the Data menu and select the input matrix file.
2. Open the Methods menu and select the methods to execute and their thresholds.
3. Open the Merge menu and specify how the results of the different methods should be combined
4. Click the **GO** button

When the GO button is clicked, a progress menu appears allowing to cancel the network construction job. It disappears upon completion of the job. During computation, no further jobs can be launched.
In more advanced runs, randomizations can be carried out (Randomization menu).

Demo

The "Demo" button computes co-occurrences on an example data set (the oral samples collected in Costello et al., Science, 2009). Running the demo does not change the current configuration of CoNet, neither is the demo influenced by CoNet's current configuration. The demo data and settings accompany the CoNet distribution.

Command line call

CoNet is accompanied by a command line tool, which is recommended if thousands of randomizations have to be carried out on large matrices. The button "Generate command line call" displays a window that shows the command line call corresponding to the current setting of CoNet. Note that Rserve-related configuration cannot be set via command line directly, but need to be specified in the (optional) configuration file of the CoNet command line tool.

Settings loading/saving

CoNet settings can also be saved to or loaded from a file via the button "Settings loading/ saving". This saves the user from repeating a tedious CoNet configuration.

GDL network loading

The "Load GDL network" box allows displaying networks in GDL format generated with the command line tool. Select the GDL network file by clicking the "Load" button and then push **GO**. To undo a GDL file selection, click "Load" again and then "Cancel". This will clear the GDL file selection.

Network display

Result networks are displayed with a number of attributes.
Network attribute:
The **Comment** stores details of the network generation as well as the command line call.

Node attributes:

- **Label** The name of the node that is displayed on the node.
- **abundance** Each node corresponds to a row in the input matrix. The abundance is the row sum. When the input matrix was processed in any way, this refers to the row sum in the processed matrix.
- **samplecount** The samplecount is the number of values of the corresponding row that are neither zero nor missing values. When the input matrix was processed in any way, this refers to the number of occurrences in the processed matrix.
- **degree** The number of links of the node.
- **posdegree** The number of positive links of the node.
- **negdegree** The number of negative links of the node.
- **unknowndegree** The number of links of the node having unknown interaction type.
- **impliesdegree** The number of links of a node contributed by association mining.
- **oriID** The identifier of the row to which this node corresponds.
- **canonicalName** An attribute set by Cytoscape that stores the node identifier.
- **isafeature** This attribute is only set in case features are provided and allows to differentiate between feature nodes and other nodes.
- **matrix_number** This attribute is only set if two input matrices are provided and indicates the origin of the node from matrix 1 or 2.
- **shiftgroup** This attribute is only set if a lag is specified and groups all shifted versions of one row together.

Additional node attributes are added when metadata are provided, such as the lineage or the node group.

Edge attributes:

- **Label** The edge identifier.
- **cooc_method** The co-occurrence method that contributed this edge. There may be more than one co-occurrence method.
- **weight** The edge weight. When no randomization is run, this is the score of the co-occurrence method. In case of several methods per edge, the weight represents the combined score.
- **interactionType** The edge type. Can be either co-presence (positive interaction) or mutual exclusion (negative interaction). Co-presence edges are colored in green, mutual exclusion edges in red.
- **method_number** The number of methods supporting an edge. This attribute is not set for multi-graphs.

- **method_scores** The scores of methods supporting an edge. This attribute is not set for multi-graphs.
- **methodname_scores** The method name versus its score for all methods supporting an edge. This attribute is not set for multi-graphs.
- **methodname_interactiontype** The method name versus the interaction type (positive, negative or unknown) it predicted. This attribute is not set for multi-graphs.
- **interaction** An attribute set by Cytoscape that refers to the source of the edge. Here, it is set to the same value as cooc_method.
- **canonicalName** An attribute set by Cytoscape that stores the edge identifier.

When the network was randomized, the following edge attributes are added:

- **pval** The p-value computed from the random score distribution.
- **qval** The q-value computed from the p-value when benjaminihochberg or bonferroni are selected as multiple testing correction strategies.
- **sig** The significance computed from the p-value when compute_eval is selected as multiple testing correction strategy.
- **methodname_pval** The method name versus its p-value for all methods supporting an edge. This attribute is not set for multi-graphs.
- **randdistribMean** The mean of the random score distribution.
- **randdistribMedian** The median of the random score distribution.
- **randdistribSD** The standard deviation of the random score distribution.
- **randdistribNumNonNaNScores** The number of iterations that did not result in missing values.
- **nulldistribmean** When both permutation and bootstrap distribution were computed, the mean of the permutation distribution.
- **nulldistribmedian** When both permutation and bootstrap distribution were computed, the median of the permutation distribution.
- **nulldistribsd** When both permutation and bootstrap distribution were computed, the standard deviation of the permutation distribution.
- **oriScore** The value of the edge weight before randomization.

When the network is a randomized merged multigraph, the following edge attribute is added:

- **pval-MERGE_STRATEGY** The p-value computed by merging measure-specific p-values with the selected merging strategy (e.g. pval-brown-merge).

In case of a merged multigraph, multiple-testing-corrected q-values are computed from the merged p-values. The q-values are also stored under the "weight" attribute.

So when you do not randomize, your edge attribute of choice to weight edges is "weight", when you randomize, it is "pval", when you merge p-values, it is pval-MERGE_STRATEGY and when you correct for multiple testing, it is either "qval" (for bonferroni and benjaminihochberg) or "sig" (for E-value computation).

Limitation

CoNet expects small-sized (hundreds of rows) to medium-sized (thousands of rows) matrices as input. If filtering steps are selected that reduce the matrix size, larger matrices can be provided as well. If you want to infer networks from matrices with more than five thousand rows, CoNet is not a suitable tool for you. You might however combine it with other tools by first clustering a huge input matrix with an external tool and then submitting aggregated clusters to CoNet. For matrices with a size close to CoNet's limit, it is best to avoid selecting slow inference methods such as mutual information or Kendall's correlation and to carry out the inference on command line.

Error report

No tool is without bugs. CoNet errors usually cause an error report to be generated, which you can send if you think it's relevant. You can also check the cytoscape-generated log file "output.log" in the Cytoscape application folder.

Load an input matrix

Open input matrix

To select the input matrix from your local file system, click "Select file" and browse the file tree. The first time you select a file or a folder after launching CoNet, it may take a bit of time before your file tree is visualized. After having selected a file, click "Open". The file path is displayed in the field above the "Select file" button. If you want to clear the selected file, click "Select File", then "Cancel". Alternatively, you can paste the full path of your file into the text field below the "Select file" button.
You can optionally load a **second input matrix**. In this case, co-occurrence is computed between all cross-matrix row combinations (and never between two rows belonging to the same matrix). However, the first and second matrix are expected to have an equal number of columns. This results in a bipartite network with two node types, each node type corresponding to one input matrix.
Note that metadata have to include all rows of the first and second input matrix. For instance, the first input matrix could represent Eukaryotes and the second Bacteria sampled for the same number of locations. A combined metadata file could then provide the lineages of both Eukaryotes and Bacteria. The delimiter and transpose options explained below apply to both input matrices.
When two matrices are read in, each of their rows is assigned a group attribute called "matrix_number", with different values for the first and second matrix. This group attribute can be entered into the "Metadata and Feature loading menu", to use group options (see below).

Matrix property computation

If you enable "Compute matrix info" and push the GO button, some basic properties of the input matrix are displayed along with configuration tips. For instance, the row and column number, minimal and maximal column sums as well as total possible edge number is displayed. The total possible edge number is computed according to the formula: n*(n-1)/2, where n is the row number (note that this formula is not valid for association rule mining). Matrix properties are computed after selected preprocessing and filter steps have been applied.

Parsing options

By default, strength of co-occurrence is computed between rows. If it should be computed between columns, activate the **"Transpose data matrix"** check box. If the matrix transposing is enabled, feature columns should match the columns of the transposed matrix (i.e. the rows of the input matrix) and metadata identifier should refer to the rows of the transposed matrix.

The columns of the input matrix are by default expected to be tab-delimited. However, the column separator can be changed by entering the desired delimiter string in the text field below **"Change default column delimiter"**.

The table can be optionally given in the **QIIME taxon count table format**, which can be generated automatically from biom files. A converter for this task is available at http://biom-format.org/. CoNet accepts QIIME OTU tables with and without taxonomy. When an OTU table with taxonomy is given (where the last column holds the lineages in the format k__KINGDOM;p__PHYLUM;c__CLASS;o__ORDER;f__FAMILY;g__GENUS;s__SPECIES), metadata and metadata attributes are automatically set from this table (overriding any previously set metadata), which allows the usage of metadata-dependent CoNet options such as higher-level taxon assignment or the suppression of relationships between OTUs belonging to the same genus. The taxonomy is also parsed automatically from QIIME phylotype tables where the lineage is included in the row name (example for a row name: k__Bacteria; p__Bacteroidetes; c__Bacteroidia; o__Bacteroidales; f__Prevotellaceae; g__Prevotella). Note however that the OTU/phylotype table should not contain features, i.e. non-taxon rows. Note also that the QIIME taxon count table format, if enabled, is assumed to be valid also for the second input matrix, if one is given. If two matrices are provided, lineages will be parsed only from the first. To enable the QIIME taxon count table format, activate the **"Table obtained from biom file"** check box. With this format enabled (and lineage information provided), taxon nodes will provide values for the phylum, class, order, family, genus and species attribute, although these values may be set to none.

Input matrix format

In general, each row should appear only once in your matrix, i.e. each row name should be unique. Lines preceded by # are treated as comment lines and omitted. The input matrix can be a pure numeric matrix with tab-delimited entries, such as the small example below:

```
9.4         5.4         10.1
13.5         4.4         11.9
14.3         13.4         8.9
```

It is also possible to specify a matrix with row names and without column names. The row name column is expected to be the first column and row names should not be purely numeric or blank. Example:

```
taxon1        1.2        4.4        3.2        0.0        1.1
taxon2        2.2        4.3        1.1        0.0        1.2
taxon3        1.3        2.1        3.1        2.3        1.1
```

The matrix can also provide row and column names. In this case, it is expected that the column names form the first row and that the row name column has itself a column name. Row and column names should not be purely numeric or blank. A small example illustrates this:

```
colnames         col1         col2         col3         col4
tax1         1.2         1.0         3.2         0.0
tax2         2.2         2.3         1.4         0.0
tax3         1.3         2.1         3.1         2.3
tax4         1.2         1.4         2.1         2.4
tax5         2.3         2.1         4.0         3.1
tax6         2.5         2.4         2.3         1.3
```

Note that OTU tables generated from biom files are also accepted (check out the "Parsing options").

Treatment of missing values

Missing values have to be indicated using NaN. For example:

```
9.4         NaN         10.1
13.5         4.4         11.9
14.3         13.4         NaN
```

Treatment of special characters

Special characters in row names (or in the first metadata column, which represents row names, see below) are replaced by a dash to avoid confusion with CoNet's own special characters. However, the original row names are stored as values of the node attribute "oriId".

Matrix type

The matrix can be of three different types. An abundance matrix has continuous values, a count matrix has integer values and an incidence matrix has only two values representing

presence (1) and absence (0). Methods like the hypergeometric distribution and association rule mining can only be applied to the incidence matrix and will produce an error if applied to an abundance matrix.

Time series data

CoNet's methods are not tailored to deal with time series. However, CoNet allows to compute lagged similarities via the option "Specify maximum lag". If a value larger than zero is set for this option, input data are treated as time series. For each time series, shifted versions are generated up to the given maximum shift. This allows to detect associations between two species with similar, but shifted abundances. Take for example the following time series observed for species A and B:

```
speciesA        1        2        3        4        5
speciesB        2        3        4        5        6
```

If a lag of 1 is specified, then CoNet would generate rows

```
speciesA-0      1        2        3        4
speciesA-1       2        3        4        5
speciesB-0      2        3        4        5
speciesB-1       3        4        5        6
```

If missing values are treated, the shifted rows would be generated as follows:

```
speciesA-0      1        2        3        4        5
speciesA-1       2        3        4        5        NaN
speciesB-0      2        3        4        5        6
speciesB-1       3        4        5        6        NaN
```

CoNet does not compute auto-correlations, e.g. no similarity is computed between speciesA-0 and speciesA-1. In addition, similarities between the same shift (larger than zero) are also not computed, e.g. the combination speciesA-1, speciesB-1 is omitted. Beware that specifying a lag greatly increases the runtime as well as the risk of false positives.

Metadata and Feature Loading

If you click on the "Metadata and features menu", a sub-menu will be opened. The four functions of this sub-menu are explained below:

1. **Metadata** Metadata are row attributes. For instance, if the input matrix rows represent species and we want to associate each species with its genus, we can upload a metadata matrix consisting of two tab-delimited columns: the first listing the row names of the input matrix and the second listing the genera. More

than one attribute per row can be up-loaded, by providing a matrix with several tab-delimited columns. The attribute names have to be given in the order in which they appear in the metadata matrix in the text field below "Enter metadata column names". If several attribute names are given, they have to be separated by a slash. For instance, if we want to load values for the attributes "lineage" and "taxon" on the rows, we need to up-load a matrix with 3 columns (row names, lineage attribute values and taxon attribute values) and specify the two attribute names separated by slash in the metadata name input field.
One column of the metadata file may assign group memberships. For instance, a column assigning body sites to each row groups rows body-site-wise. If a **group attribute** has been specified, parent-child relationships between taxa across different groups are allowed and column-wise normalization and renormalization (see randomization) are carried out group-wise.
A **second group attribute** can be provided, which does not have any of the effects of the first group attribute. Instead, specifying this group attribute prevents intra-group links, i.e. links between nodes sharing the same value for this group attribute are forbidden. This is useful to prevent links between too closely related taxa.

2. **Lineage** The lineage is a specific row attribute that provides the taxonomic classification, from most generic to most specific, e.g. Bacteria--Bacteroidetes--Bacteroidia--Bacteroidales--Prevotellaceae--Prevotella. The two dashes ('--') between the taxonomic level are the lineage separator, which can be changed using option **lineage separator**. The lineage column metadata name is fixed to "lineage". It is recommended that, if both lineage and taxon are provided, the lineage includes the taxon itself as last entry, e.g. if the taxon is OTU-12345, the lineage could be Bacteria--Bacteroidetes--OTU-12345.
When a matrix consists of taxa on the same level and the lineages are given in the metadata, higher-level taxa rows can be automatically assigned by summing lower-level taxon rows. The option **explore links between higher-level taxa** enables automatic assignment of higher taxon rows from the lineages, which allows to compute correlations between higher-level taxa. This option however assumes that all taxa in the input matrix are on the same taxonomic level and that taxonomic levels have different names (Beware: Actinobacteria is the name of the class and the phylum, so the name would need a modification such as Actinobacteria_class to differentiate between class and phylum). It is advisable to use this option together with the parent-child exclusion filter.
3. **Row combination filters.** Specifying metadata allows to apply filters during network building that are listed here:
    1. The **parent-child exclusion** filter prevents the formation of links between taxa that have a parent-child relationship, such as in *Pasteurellaceae* versus *Pasteurella*. This has an impact on the run-time and the (non-edge-wise) p-value calculation. To apply this filter, a **lineage** and a **taxon** attribute are expected (with metadata column names "lineage" and "taxon" respectively). The taxon is expected to correspond

to the last entry of the lineage. For instance, one line in the metadata file (with a lineage and a taxon column) could look like this:

```
id1         Bacteria--Actinobacteria         Actinobacteria
```

2. The **Within-group relationships only** filter can be applied to prevent links between items belonging to different groups. It requires the first group attribute to be set.
3. The **Between-group relationships only** filter can be applied to prevent links between items belonging to the same group. It requires the first group attribute to be set.

4. **Features** Features are additional rows that differ from the other rows of the input matrix. For instance, if the matrix describes count data of species, the features could capture physical properties such as the temperature or pH. In the network, feature nodes are displayed with another shape than the other nodes. Features should be set separately, since they need to be excluded from some operations, such as column-wise normalization.
   The feature matrix is expected to be tab-delimited and to contain continuous data when the input matrix is of count or abundance type or of binary data when the input matrix is an incidence (i.e. presence/absence) matrix. Otherwise, the rules for input matrix parsing also apply to feature matrix parsing, i.e. row and column names should not be purely numeric, special characters will be treated and lines preceded by # are interpreted as comment lines and skipped. It should not contain negative values, because these cannot be treated correctly by Kullback-Leibler and other measures that interpret standardized abundances as probabilities (if you only plan to use correlations, negative values are fine). In addition, constant features (i.e. those that have the same value across all samples) should also be avoided. If **transpose** is enabled, the feature matrix is transposed, i.e. columns are parsed as rows and vice versa. If **match samples** is enabled, feature samples are matched to input samples. Samples not present in the feature matrix will be discarded from the input matrix and vice versa. In addition, samples in the feature matrix are ordered in the same way as the input matrix. Thus, this option can be activated if the feature samples are in a different order than those of the input matrix. Note that matching is carried out after transposing, if both options are enabled. Note that the feature matrix is expected to have the same samples in the same order as the input matrix (after transposing), except when matching is enabled.

   The following feature-specific filters are provided:
   1. Activating the first filter excludes features from association rule mining.
   2. Activating the second filter keeps only features in the network that are supported by Spearman correlation.

Save output network

Saving the resulting network to a given location is an option that allows saving results in formats not supported by Cytoscape. If no output file is selected, the network is visualized in Cytoscape without being saved to a file.

To specify an output file location, select first a folder via button "Select folder", then click on the folder into which the output file should be saved and click "Choose". The selected folder will be displayed in the text area below this button. Then type a file name in the text area below "Type file name". To undo the specification of an output file, remove the file name or push the button "Select folder" and then push Cancel.

Network formats

The GDL (GraphDataLinker) format is a custom format that stores networks along with their node and edge attributes. In contrast to all Cytoscape-supported formats, it preserves multi-edges. GDL networks can be loaded into Cytoscape via a button in the main menu. The "tab" format is a tab-delimited custom format that consists of two parts: a node part listing all nodes and their attribute values, one by line, and an edge part listing all edges and their attribute values, one by line. Here is a small example:

```
;NODES          genus

a          Escherichia

b          Yersinia

c          Salmonella

d          Mycoplasma

;ARCS          score

a          b          1.1

b          c          2.4

c          a          3
```

The other formats are supported by the following tools:

- **VisML** This is the input format of the network and pathway analysis platform VisANT.

- **Dot** This is the input format of the command line graph visualization software GraphViz.

Output network properties

Each output network has a "Comment" and a "CALL" entry. The comment is a long string that lists the details of network generation, whereas the CALL string precedes the command line call that was used to generate the network.

Preprocess and filter input matrix

Input filtering

In some situations, it is useful to filter the input matrix. For instance, if the matrix represents bacterial abundances, noise may contribute much more than biological reasons to variances in low-abundant bacteria abundances. In addition, low-abundant species contain many zero values, which are ambiguous: they mean either that the taxon does not occur at all in the sample or that it was below detection limit. Zeros require a special treatment: either by "smoothing" (e.g. Witten-Bell smoothing) which tries to "guess" values from the remaining data or by pseudo-counts. If a row is dominated by zero entries and missing values, it is best discarded. Below, a number of filters are listed that allow discarding problematic rows. These filters can also be combined.

1. **row_minsum** The sum of the values of each row should be equal to or larger than the specified minimum. To specify the minimum, write the value in the text field next to the check box and press enter.
2. **row_minocc** Each row should have at least the specified number of non-zero, non-missing values.
3. **col_minsum** The sum of the values of each column should be equal to or larger than the specified minimum.
4. **col_minocc** Each column should have at least the specified number of non-zero, non-missing values.
5. **minzeropairs** Each row pair should have at least the specified number of non-double-zero value pairs when computing its similarity score. This filter avoids computing high scores based on double absences.

Keep sum of filtered rows

Instead of simply discarding filtered rows, sum them into a single row that is kept for further processing steps.

Standardization

If the columns of the matrix represent different samples, standardization over the samples is necessary. A number of standardization strategies are available, which can be applied to columns and/or to rows. Features are excluded from standardization. If a group attribute has been specified (see data menu help), column-wise standardization strategies are carried out group-wise. Note that rows or columns with zero sum will be removed from the input matrix. If both filtering and standardization is specified, filtering is carried out first, followed by standardization.

1. **col_norm** Each column is divided by its sum, converting abundances in column-wise proportions.

2. **col_downsample** Each column is down-sampled such that its sum is equal to the sum of the column with the smallest sum. This option is only available for count matrices.
3. **row_stand** Standardize rows such that each row x is transformed as follows: x_stand = (x-mean(x))/sd(x).
4. **row_stand_robust** Standardize rows as in row_stand, but estimate the mean by the median and the standard deviation with the interquartile range normalized by (qnorm(0.75) - qnorm(0.25) ~ 1.35), where qnorm is the quantile function of the standard normal distribution (robust standardization follows an R-script by Jacques van Helden).
5. **row_norm** Each row is divided by its sum, converting abundances in row-wise proportions.
6. **row_downsample** Each row is down-sampled such that its sum is equal to the sum of the row with the smallest sum. This option is only available for count matrices.
7. **log2** Take the logarithm to basis 2 of each matrix entry.

To incidence conversion

When an abundance or count matrix is given, but incidence matrix methods (such as the hypergeometric distribution) shall be applied as well to the input matrix, the matrix can be converted into an incidence matrix for these methods. The following methods are available for conversion (features are excluded from the conversion and subsequent analysis steps):

1. **user** Each matrix value equal to or above the threshold is considered as presence (and set to one) and each value below is considered as absence (and set to zero).
2. **quantiles** The row value equal to or above the given row-specific quantile is considered as presence (and set to one) and the row value below is considered as absence (and set to zero). Quantiles should be set between 0 and 1. For instance, if the quantile was set to 0.75, a row value is considered as present only if it is equal to or above the third row quantile.
3. **eval** Each row value is transformed into a z-score by subtracting the row mean from it and dividing it by the row standard deviation. With the normal distribution, each z-score is converted into a p-value, which is multiplied by the column number to obtain a multiple-test corrected E-value. An E-value below or equal to the given threshold is considered as presence, else as absence.

Output filtering

Edges can represent positive ("copresence") or negative ("mutual exclusion") relationships between the items. The output filter allows keeping only copresence or only mutual exclusion edges.

Select measures and specify their thresholds

A set of methods to infer relationships between items can be selected. In order to select a measure of interest, activate its corresponding check box.

Correlations

Correlation measures all have a range of [-1,+1], with -1 for the strongest negative relationship (anti-correlation), 0 for the neutral case and +1 for the strongest positive relationship. Briefly stated, Pearson is assuming a linear relationship between the data, whereas Spearman and Kendall are rank-based and thus do not assume linearity. It should be pointed out that Pearson, Spearman and Kendall are not defined in case the standard deviation is zero. Thus, if two taxa have constant abundances, they will not be linked by these correlation measures. For large matrices, the computation of Kendall can take a prohibitive amount of time and should better be carried out on command line. For the formulas and in-depth discussions, we refer to the respective Wiki pages: Pearson, Spearman, Kendall
Note that all three correlations are sensitive to the so-called double-zero problem (Legendre & Legendre), that is a vector pair with many matched zeros will receive a higher score than the same vector pair without them. Since the interpretation of zeros is often ambiguous, this is a serious drawback when dealing with sparse data.
In addition, correlation results may be strongly biased when applied to normalized data (see Aitchison 2003). Renormalization (see the randomization menu) can reduce this bias.

Similarities

Similarities range from 0 to 1 or infinity. The higher their score, the more similar are two objects.

- The **Steinhaus similarity** is the inverse of the Bray Curtis dissimilarity and is defined as
  $S(x,y) = 2W/(A+B)$, where A = sum of vector x, B = sum of vector y and W = sum of minimal values, i.e. $\min(x_i,y_i)$
  It is bounded between 0 and 1.
- The **mutual information** has been defined and implemented in a variety of ways. However, different estimations of mutual information give different results (Fernandes & Gloor). The default in CoNet is a fast implementation of mutual information by Jean-Sebastien Lerat. A re-implementation of the ARACNE matlab script, which depends on a parameter (the Gauss kernel width), is also available. With Rserve enabled, one can also select the implementation that is

part of the minet package in R (see config menu). When computing mutual information with minet or the default, abundance matrices or normalized matrices require a discretization step, which can be selected in the config menu. Note that the discretization step computes the bin number as the square root of the sample number, thus mutual information with discretization requires a sufficiently large sample number. Whereas extreme values in most measures are indicative of co-presence at one end and mutual exclusion at the other, an extremely low mutual information (close to zero) signals independence, whereas strong co-presence and mutual exclusion patterns receive likewise high mutual information values. Thus, when mutual information is combined with the "Top and Bottom" option explained below, the double number of top edges is returned instead of top and bottom edges. Note that for the same reason, mutual information cannot assign the interaction type.

- The **variance of log-ratios** was introduced by Aitchison and is a measure of dissimilarity between two components across compositions, i.e. two species across samples. It is defined as:
  $D(x,y)=var(log(x_i/y_i))$
  Aitchison suggested a transformation that scales the variance of log-ratios between 0 and 1:
  $S(x,y) = 1-exp(-sqrt(D(x,y)))$
  This transformation converts the measure into a similarity, with 0 corresponding to a lack of proportional relationship and 1 perfect proportional relationship. Note that a pseudocount is added to deal with zero values. However, the variance of log ratios depends on the selected pseudocount, thus variance of log ratios computed on different zero-containing data are only comparable if the same pseudocount was used.
- The **distance of correlation** (also known as Brownian correlation) is a similarity measure in the range of 0 to 1. In contrast to correlation measures, 0 flags statistical independence. Like mutual information, an interaction type cannot be assigned with this measure. For this reason, if this measure is used together with the "Top and Bottom" option, the double number of top edges is returned. The distance of correlation has been implemented following the implementation in the R package energy (DCOR). For more information, see the Wiki entry.
- The **Hilbert-Schmidt independence criterion (HSIC)** is another general measure of dependency (see this Wiki entry). As in the case of mutual information and distance correlation, no interaction type can be assigned and the option "Top and Bottom" returns simply the double number of top edges. A HSIC of zero flags independence, and the higher the HSIC, the stronger the dependence.

Dissimilarities

Dissimilarities range from 0 to 1 or infinity. The higher their score, the more dissimilar are two objects. A distance (or metric) has to fulfill the following criteria: 1) It should

never be negative, 2) It should be zero only if objects are identical, 3) It should be symmetric, e.g. d(x,y)=d(y,x) and 4) It should fulfill the triangle inequality (d(x,z) <= d(x,y)+d(y,z)). Dissimilarities do not meet the fourth criterion.

- **Euclidean distance** is defined as
  $D(x,y) = sqrt(SUM_i(x_i - y_i)^2)$ and goes from 0 to infinity.
  Each row is divided by its sum, such that it adds up to one.
- **Bray Curtis dissimilarity** This is the inverse of the Steinhaus similarity described above and is bounded between 0 and 1. The formula is: $D(x,y) = 1 - 2W/(A+B)$, where A = sum(x), B = sum(y) and W = $sum(min(x_i,y_i))$
  Each row is divided by its sum, such that it adds up to one.
- **Hellinger distance** is computed as follows:
  $D(x,y) = sqrt(SUM_{i=1}(sqrt(x_i) - sqrt(y_i))^2)$, where x and y are supposed to sum to one
  Each row is divided by its sum, such that it adds up to one. The Hellinger distance is closely related to the Kullback-Leibler divergence.
- **Kullback-Leibler dissimilarity**
  $D(x,y) = SUM_{i=1}(x_i*log(x_i/y_i)+y_i*log(y_i/x_i))$, where x and y have been standardized to sum to one.
  Note that a pseudocount is added to deal with zero values.
- **Jensen-Shannon dissimilarity**
  $D(x,y) = 1/2*SUM_{i=1}(x_i*log(x_i/M) + y_i*log(y_i/M))$, where $M = (x_i+y_i)/2$ and where x and y have been standardized to sum to one.
  Note that a pseudocount is added to deal with zero values.

Incidence methods

In case the matrix is of type "incidence" (i.e. only contains presence/absence values) or a conversion to the incidence type has been specified in the preprocessing menu, a number of methods specific to incidence matrices are available.

- The **hypergeometric distribution** is often used to compute the p-value of the overlap between two sets, taking into account the set sizes. The one-tailed version of the hypergeometric test is also known as Fisher's exact test. Here, the p-values of the hypergeometric distribution are multiple-test adjusted using the E-value correction (that is multiplication of p-values with the test number) and then converted into significances (sig=-log10(p-value)). The higher the significance, the smaller the likelihood that the predicted edge is due to chance. Thus, the significance behaves like a similarity measure. Strictly speaking, it is not a similarity, since significances can be negative, but the significance threshold is usually set to zero or larger.
- The **Jaccard distance** also measures set overlap and is defined for two sets A and B as:
  D(A,B) = 1 - size(intersection(A,B))/size(union(A,B))
  The Jaccard distance ranges from 0 to 1 (see Wiki entry).

- **Association rule mining (not supported for Windows)** enumerates logical rules in presence/absence data sets. These rules can span any item number, but due to multiple-testing and run-time issues, one should restrict association rule mining to low item numbers. Numerous filters have been developed to retain only rules of interest. CoNet wraps Christian Borgelt's implementation of the apriori algorithm (Agrawal et al.). For the meaning of the filters, we refer users to the documentation of apriori. By default, if confidence is not selected as a filter, it is set to 50, whereas support (if not selected) is set to 10. Using association rule mining requires apriori to be installed (see configuration). Note that association rule mining is currently the only method that returns a directed network.

Network inference with Minet

Minet is an R package that implements three popular network inference algorithms used in genetic regulatory network inference, namely CLR (Faith et al.), ARACNE (Margolin et al.) and MRNET (Meyer et al. 2007). These algorithms work on the basis of a similarity matrix, where the similarity is usually mutual information. Note that using a correlation measure instead requires the data to be normally distributed. Using minet requires Rserve to be enabled. Minet's strategies for mutual information estimation and discretization (required for abundance or normalized matrices) can be set in the configuration menu. Note that by default, mutual information is not computed in minet, but using ARACNE's algorithm. The default can be changed in the config menu.

Threshold setting

In order to set method-specific thresholds, two options are available:

- **Manually:** The user can set manually the threshold via the sliders and text fields next to the measures. Text fields are up-dated upon slider movement and vice versa.
- **Automatically:** Click the "Automatic threshold setting" button on the bottom of the Methods menu to open the Threshold setting menu. The user can specify a certain edge number or quantile in the text field below the "Edge selection" choice. CoNet will then set thresholds automatically such that the corresponding edge number is included in the output network. The quantile is expected to be in the range of 0 to 1. For example, if the quantile is set to 0.05, the threshold will include the 5% top-scoring edges.
  If the edges with lowest scores should be included as well, activate the "Top and bottom" check box. This is of interest to capture exclusion for measures other than correlation. This option will be ignored for measures that cannot support it (such as mutual information). For these measures, an equivalent number of top edges is selected instead. Make sure to disable "Top and Bottom" and to clear the edge selection parameter to disable automated threshold setting.

If "Force intersection" is enabled, thresholds are set such that resulting network has the selected number of intersection edges. This option can be used in combination with network merge strategy "intersection", to obtain an intersection network of given size (see Merge menu help).
Automatically set thresholds can be optionally saved to a file called "thresholds.txt" in a user-selected folder. To select the folder, click the button "Select folder", click on a folder in the file tree and then click "Choose". In the folder selection mode, files in the file tree are not clickable.
Note that it is always possible to store the thresholds as part of a CoNet settings file (see Settings loading/saving) after completion of the network computation task. With such a settings file, there is no need to recompute the thresholds. However, the **guessing parameter should be deleted** (such that the field is empty) **before the current settings are saved** or the threshold guessing line in the settings file should be removed, **to avoid recomputing the thresholds**.
When loading lower and upper thresholds from a settings file, only one of them will be displayed for each measure in the interface, however both will be applied during network inference.

Visualize score distributions

Score distributions of selected measures can be visualized by selecting an export folder and activating the "Export distribution" check box in the Threshold setting menu. To select an export folder, click the "Select" button and click on a folder, then click "Choose". The score distributions will be exported into a single pdf file called "score_distributions.pdf" in the selected folder. When this option is activated, pushing the "GO" button in the main menu will not result in a network, but instead the user is asked whether to open the pdf file with the distribution plots. Pdf files are displayed using the Adobe Acrobat Viewer. In case the user confirms opening the pdf file, Acrobat Viewer asks whether the user accepts the License Agreement and upon acceptance displays the pdf file. When score visualization is enabled, previously set thresholds are lost.

Note that threshold guessing and score visualization are not supported for minet, hypergeometric distribution and association rule mining.

Merge networks and edge scores

To combine the output of several methods, different strategies can be considered, which are listed below. Independently of the merge strategy choice, the method(s) supporting a link between two items are always listed in the resulting network as edge attribute values.

Multigraph

When the "multi-graph" checkbox is enabled, outputs of different methods are not combined, but displayed in a multi-graph, i.e. if more than one method supports a link between two items, those items are linked by more than one edge. If "multi-graph" is not enabled, the default score merge strategy is carried out instead. "Multi-graph" is currently by default enabled.

Score merge

When different methods support the same item pair, their scores can be combined into one score in different ways. Thus, the score merge is carried out edge-wise. It is only carried out when more than one network inference method has been selected. Note that in case of restoring networks from randomization files, the contributing methods and their individual scores cannot be restored.

1. **scoremethodnum** Simply put the number of supporting methods as score.
2. **scoremean** First, distances (D) are converted to similarities (S) by: S = maximum_observed_value - D. Each method-specific score is then converted into a percentage with respect to its maximal observed score, even for bounded measures. Finally, the combined score is obtained as the mean over the modified method scores.
3. **scoresum** As for scoremean, but instead of the mean the sum is computed.

Network merge

The results of the different measures can be considered as alternative networks, which can be merged in various ways (with and without enabling "multi-graph"). The following options are available:

1. **union** The network union keeps all edges contributed by all methods.
2. **intersection** The network intersection keeps only edges supported by all methods.
3. **separate** Separate networks are instantiated for each method. If this option is selected, no score merging is carried out.
4. **majority** Only edges supported by a majority of methods are kept.
5. **minsupport** An edge is only kept if supported at least by the specified method number (see input field below). The default minimal support is 2.

The **minimum number of methods that should support a link** is only taken into account if the network merge strategy is set to **minsupport**.

P-value assignment and multiple-testing correction

Background

For a matrix with n rows and a symmetric measure, n*(n-1)/2 tests are carried out during network inference. Thus, with increasing row number, it is more likely to find significant relationships by chance alone. There are various approaches to correct for this multiple testing issue, most of which rely on the adjustment of p-values. The p-value describes the probability that a relationship is inferred due to chance, thus the smaller the p-value, the higher the significance of a relationship. P-values can be computed from a random score distribution. In order to calculate such a "background" edge score distribution given a null hypothesis, random data are generated a selected number of times.

Random score distribution options

Randomization routine

First, the randomization routine needs to be chosen. The default "none" disables any randomization test. The "edgeScores" routine generates a set of edge-specific score distributions, whereas the routine "lallich" is an implementation of BS_FD (bootstrap-based false discovery) routine and is recommended for association rule mining. This routine takes a false discoveries number parameter, which is set to one. Note that the "lallich" routine ignores the resampling parameter (which is fixed to bootstrap for this routine) and requires the significance level as parameter (which can be provided via the p-value threshold, e.g. 0.05). In addition, it does not keep bottom edges (i.e. edges with extremely low similarity or high distance scores, see option "Top and bottom" in the methods menu). Note also that the "lallich" routine computes global score distributions for each method. In contrast to the edge-specific routine, the "lallich" routine does not assign p-values to the edges. Instead, it adjusts the original thresholds and discards edges with scores below the adjusted thresholds. The adjusted thresholds are stored in the network comment attribute. Since "lallich" does not assign p-values, the multiple-testing correction options are not applicable.
Note that only the "edgeScore" routine allows computing edge-specific p-values. The p-values are one-sided, that is a very high p-value is indicative of mutual exclusion (e.g. for a dissimilarity, the score was much higher than expected at random). P-values above 0.5 are converted by subtracting the p-value from one prior to multiple test correction. However, if the p-value does not agree with the initial interaction type (e.g. a high p-value for a co-presence edge), the edge is discarded. P-values are computed as follows:

- For **bootstrap**, the p-value of the expected null value (e.g. 0 for Pearson) under the normal distribution is computed, using the mean and standard deviation of

the bootstrap distribution as parameters of the normal distribution. In short, we compute how likely it is to observe the null value given the distribution around the observed score. In case the measure has no proper null value (which is the case for unbounded measures), the null value is obtained by shuffling the vector pair.

- For **shuffling (permutation)**, the p-value is computed distribution-free by counting the number of times the observed edge score is smaller/larger (depending on the measure) than the random edge scores, using the formula p-value = (r+1)/(n+1), where n is the number of randomized scores and r the number of randomized scores smaller/larger than the observed score.
- If **permutation and bootstrap** are combined (see "Loading null distributions"), the p-value of the null value under the normal distribution is computed, using the mean and standard deviation of the bootstrap distribution as parameters of the normal distribution. The null value is the mean of the null distribution.

Resampling

By default, the matrix is randomized by permuting the order of columns row-wise ("shuffle_rows"), which preserves the row sum. Alternative re-sampling procedures are the permutation of row order column-wise ("shuffle_cols"), to permute rows as well as columns ("shuffle_both"), or to bootstrap.

Iteration number

The iteration number specifies how often the randomization routine is carried out, i.e. how many values the random score distribution contains.

P-value merging

If the computation of a multi-edge network and edge-wise randomizations are enabled, method-specific p-values of multi-edges connecting the same node pair can be merged using the strategies listed below. Note that p-value filtering and multiple-test correction, if selected, are carried out after the merge, i.e. on the merged p-values.

- **Brown** Brown's method works like Fisher-merge (see formula below), but divides the result by a correction factor before calculating the Chi^2 p-value. The correction factor takes the correlation of measure scores into account, see Brown 1975 for details. The correction factor is calculated when calculating thresholds for a given edge number or quantile, so Brown's method cannot be used with manually set thresholds.
- **Fisher-merge** Formula: $Chi^2 = -2\ SUM_{i=1}^{k}(\log(p_i))$

  The p-value of the merged edge can then be computed from a Chi^2 distribution with 2*k degrees of freedom. See the Wiki entry for more information on Fisher's method.

- **Simes** Keep the minimum p-value.
- **mean** Keep the mean of the p-values.
- **geometric_mean** Keep the geometric mean of the p-values.
- **median** Keep the median of the p-values.
- **max** Keep the maximum p-value.

Note that p-values of edges filtered out initially are missing for the p-value merge. In Brown's and Fisher's method, these missing p-values are implicitly set to one, but they are not considered in any other p-value merge method. The best is to enable the **Force intersection** option in the Threshold setting sub menu of the Methods menu when using a p-value merge option.

Renormalization

This option is recommended when the input matrix is normalized column-wise and correlation measures are selected. Renormalization (Sathirapongsasuti*, Faust* et al., PLoS Comp Bio 8(7): e1002606, 2012) is a strategy that has been empirically shown to counter the compositionality bias. The compositionality bias affects correlation measures and is introduced when normalizing across samples (see Aitchison). The principle is to repeat sample-wise normalization for each item pair in each randomization round, such that the effect of all other items on this pair are captured. Renormalization is only compatible with the "shuffle_rows" resampling routine. Attention: Renormalization is very time-consuming and best run on command line. It is not applied to measures known to be robust to compositional bias (Bray Curtis, Hellinger, Kullback-Leibler, Variance of log-ratios) or to incidence matrix measures. Note that association rule mining cannot be combined with renormalization. Renormalization is only possible for column-wise normalization by column sum division. Note that features are excluded from renormalization. If groups were specified, renormalization is carried out group-wise. When combining renormalization with pre-processing, permutation and renormalization is carried out on the matrix resulting from all of these pre-processing steps. Without renormalization, resampling is carried out on the original input matrix and the preprocessing steps are repeated for each iteration.

Variance pooling

When permutation and bootstrap distributions are combined, their variances can be pooled to account for different iteration numbers using the following formula: pooled_sd = sqrt((var(permutation) + var(bootstrap))/2), which computes the mean variance of the bootstrap and permutation distribution. The standard deviation of the pooled variance instead of the bootstrap standard deviation is then used to compute the combined p-value. To enable variance pooling, activate the "Pool variances" check box.

Discarding unstable edges

Some edges might be only significant due to outliers and have much lower scores once the outliers are removed. Edge-specific bootstrapping results in a confidence interval for each edge. If the observed edge score is not within the 2.5 and 97.5 percentile of the bootstrap distribution, it can be considered as exceptionally high (due to outliers) and the edge is removed.

P-value options

Once p-values have been obtained from the randomization routine, they can be multiple-test adjusted using various approaches. For each approach, a lower and optionally an upper threshold for the edge p-value can be applied. All edges with p-values above the lower threshold (defaults to 0.05) are discarded.

- **E-value** The E-value is calculated by multiplying the p-value with the number of tests. An E-value of one means that one predicted relationship is expected to be due to chance. The E-value is converted into a significance by computing -log10(E-value).
- **Bonferroni** The Bonferroni-corrected p-value is obtained by dividing the threshold p-value by the number of tests and retaining only those p-values that are below the *adjusted* threshold.
- **Benjamini-Hochberg** The method by Benhamini and Hochberg adjusts the false discovery rate (FDR), which controls the expected number of false positives among all significant relationships. When applied to multigraphs, BH is carried out with the Yekutieli correction for dependent p-values. The dependency arises because multiple measures may support the same relationship. For multigraphs with merged p-values or normal graphs, BH is carried out without dependency correction. See this Wiki entry for more details.

Saving and re-using randomized scores

Since the repeated computation of random scores can be extremely time-consuming, it is possible to save the originally inferred network along with its random scores. To do so, select a folder by clicking the "Select folder" button on the bottom left side of the menu ("Save"). This will open the file tree, where you can click a folder and then click "Choose". Below the folder selection button, there is a text field "Specify file name", where you can input the name of the file into which scores are to be saved. Then click the "Save randomizations to file" checkbox.
The network can later be restored by selecting the file with the randomization results via the "Open file" button in the middle right of the menu ("Load" box). To clear a selected randomization file, click the "Open file" button again and push "Cancel".
Note that the CoNet settings should match the settings used to generate the randomization file (except for the p-value thresholds or metadata). For saving settings, click the

"Settings loading/saving" button in the main menu.
Loading a network from a randomization file requires at least the following settings:

- the input matrix
- the randomization routine used to generate the data
- the resampling method
- the selected measures and for the "lallich" randomization routine their thresholds
- the p-value threshold
- if initial thresholds were set with "topbottom" enabled (see Methods menu): the upper p-value threshold
- if selected during randomization: multigraph and p-value merge strategy

and the following options can be enabled:

- if the resampling method is bootstrap, unstable edges can be filtered
- if a null distribution has been generated with permutation, it can be loaded together with the bootstrap distribution (see below)
- multiple-test correction of p-values

Loading null distributions

Optionally, edge-specific p-values can be computed from two distributions: the null distribution generated by shuffling and the bootstrap distribution, from which a confidence interval can be derived. The p-value is then obtained from both distributions as described above. In order to compute p-values by combining two randomizations, the following steps are necessary:

1. Configure and run CoNet to compute the null distribution (using one of the shuffling options and, optionally, renormalization). The null distribution should be saved to a file as explained above (using the "Save" box).
2. Configure and run CoNet to compute the bootstrap distribution and to set combined p-values. The previously calculated null distribution can be set in the "Load null distributions" box.
3. For future runs, both distributions can be reloaded; the null distribution via "Load null distribution" and the bootstrap distribution via "Load randomization file". Thus, there is no need for re-computing any of these distributions when changing the p-value treatment.

Configure CoNet

CoNet by default runs without Rserve, but enabling Rserve allows the usage of more advanced implementations of some measures, especially the minet package for mutual information.

Rserve configuration

By default, Rserve is disabled. Enable it by activating the "Enable Rserve" check box (assuming you have installed and started the Rserve server outside of CoNet). The host and port of the Rserve server can be specified in the host and port text fields.

Rserve installation

Rserve is a server that allows to connect to R from within other programming languages. If you do not have access to an Rserve server, you can install and run it locally on your machine as follows:

1. Start R (it can be obtained from http://www.r-project.org/)
2. In the R console, type:

   ```
   install.packages("Rserve")
   ```

3. After installation of the Rserve package, load it with:

   ```
   library(Rserve)
   ```

4. Then start the Rserve server with the command:

   ```
   Rserve(args="--no-save")
   ```

5. Rserve is now running locally on your machine with default parameters (host=127.0.0.1, port=6311)

More detailed information on Rserve installation is given here.

Implementation selection

For mutual information (MI), the choice of implementation affects the results. Three implementations are offered, the default is the implementation by Jean Sebastien Lerat (option jsl). Alternatively, MI can be computed with a re-implementation of the ARACNE implementation (the original Matlab code is available here) which has one parameter, the standard deviation of the Gaussian kernel (which can be set as one of the global constants). However, ARACNE cannot deal with missing values. Last but not

least, MI can be computed using the minet package in R. This requires Rserve to be enabled. Minet is by default run with the "mi.shrink" estimator. Discretization strategy and estimator can be selected in the Minet settings.

Association rule mining binary

CoNet wraps the apriori association mining algorithm developed by Agrawal et al. and implemented by Christian Borgelt. To use apriori in combination with CoNet, please download it from here and indicate the directory in which you placed the binary. This can be done by clicking "Select folder", then click the folder of choice and finally click "Choose". Alternatively, you can add the binary to your path environment variable (/usr/bin or /usr/local/bin). Note that association rule mining is not supported for Windows.

Missing value treatment

Missing values can either be ignored (the default) or treated by omission ("pairwise_omit"). In the latter case, if a score is computed between two matrix rows, all value pairs that contain a missing value are omitted. Non-pairwise computations such as the ARACNE implementation of mutual information do not handle missing values and can therefore not used in combination with "pairwise_omit". Pair-wise omission of missing values may shorten the vectors to such an extend that the computed score is meaningless. To prevent this, a minimum number of missing-value free pairs can be specified, which should however not exceed the number of columns in the input matrix.

Global constants

The Gauss kernel width is a parameter of the ARACNE implementation of mutual information. The pseudo-count is added to zeros in measures like Kullback-Leibler and Jensen-Shannon dissimilarity to prevent negative infinity to occur. Note that the variance of log ratios and to a lesser extent Kullback-Leibler are sensitive to the value of the pseudocount, thus the same pseudocount needs to be selected to compare their scores across different data sets. However, pseudocounts are adjusted automatically when they are larger than the smallest non-zero value in the matrix. Thus, it is better to compare Kullback-Leibler and variance of log ratio networks not on the score level, but on the p-value level.

Mutual information settings

Set minet default value for mutual information estimation (see the minet R package documentation for details). In addition, the discretization method for both minet and mutual information as implemented in Jean Sebastien Lerat's library can be selected. The selected discretization method is only applied if the matrix is an abundance matrix or if a

count matrix has been transformed into a continuous matrix by a normalization step. The number of bins is always determined by taking the square root of the column number.

Speed-up

From version 1.0b1 onwards, CoNet comes with a major re-implementation of its core to make it faster. However, it is possible to switch back to the old implementation, in case users want to reproduce previous results with the old core. The re-implementation covers the measures Pearson, Spearman, Kendall, Hellinger, Euclid, Steinhaus, Bray Curtis, Kullback-Leibler and Variance of Log-ratios and adds the Hilbert-Schmidt independence criterion (which is not available with the old core). The re-implementation also includes renormalization for these measures. For Kullback-Leibler, the score range (but not the edge ranks) changed, such that old threshold files cannot be re-used with versions 1.0b1 to 1.0b3. However, from version 1.0b3.1 onwards, the Kullback-Leibler score range is the same as for the alpha version. From version 1.0b2 onwards, mutual information and Jensen-Shannon dissimilarity are also available (with the old core, only the ARACNE implementation of mutual information is available and Jensen-Shannon is not supported). Pairwise count/abundance measures not covered by the speed-up, which include logged Euclidean and Chi-Square distance, are removed. Note that the speed-up does not cover any of the incidence measures.

Note on reproducibility
Networks constructed with and without speed-up (using the full pipeline, e.g. p-values computed from both renormalized permutations and bootstraps) do not overlap perfectly. The Jaccard indices of edge overlap vary, but are typically between 0.5 and 0.6 (the maximum being one). Thus, if networks should be reproduced that were generated with the alpha version of CoNet, disable the speed-up. Also note that networks may vary slightly when re-computed with the same version of CoNet, since p-values are computed by randomization (typically this variation should not be larger than 5% in terms of Jaccard index of edge overlap).

Load or save CoNet settings.

This menu allows you to save your current CoNet settings, so if you want to re-do an experiment after having closed CoNet, you do not need to click again all the buttons involved.

Load CoNet settings

Select a configuration file by clicking on "Select file". Then, click the "Apply settings in selected file" button to replace the current CoNet setting by the setting in the file. All previous settings will be lost!

Save CoNet settings

Clicking on "Select folder" will open the file browser. Select the folder into which you want to save the settings file by clicking on it, then click "Choose". The path of the selected folder will appear below the "Select folder" button. Finally, enter the name of the settings file in the text field below "Enter file name". Clicking the "Save current settings to file" button will save the current settings to the file of chosen name in the selected folder.

Restore CoNet default settings

If you want to start "from scratch", click the button "Set default", and all parameter values in CoNet will be re-set to their default values. All previous settings will be lost!

CoNet on command line

Using CoNet on command line - step by step tutorial

When a high number of iterations is requested (especially with renormalization enabled), CoNet is best run on command line. To ease the command line call of CoNet, the Cytoscape plugin allows generating the command line call from the current settings.

Prerequisites

This tutorial demonstrates how to run CoNet on command line. It assumes that you have downloaded and unzipped the CoNet.zip file. It also assumes that you have Java Runtime Environment 1.6 or higher installed on your machine (if Cytoscape 2.8 runs on your computer, you don't need to worry about this).

Optional: Set the Classpath

Optionally, you can add CoNet.jar located in the lib folder of CoNet to your class path. Java will find the jar by looking up this class path variable. Check this tutorial on how to do this on different systems. Example for MacOS and UNIX-based systems (on command line):

```
export LIB=/Users/me/Documents/CoNet/lib
export CLASSPATH=${CLASSPATH}:${LIB}/CoNet.jar
```

Example for Windows (in command prompt window):

```
set CLASSPATH=%CLASSPATH%;C:\Users\me\Documents\CoNet\lib\CoNet.jar;
```

Tutorial steps

1. Start Cytoscape and open the CoNet plugin. Load the demo settings (the file is located in the demo folder of the CoNet directory).
2. Open CoNet's Data menu and select "Costello_2009_oral.txt" located in the demo folder as input matrix.
3. Click the "Generate command line call" button in CoNet's main menu. This will open another window with some text in it.
4. Copy the line(s) of the text starting with "java". These lines represent the commands you want to carry out. They call CoNet's command line version with the parameters you have selected or loaded in the Cytoscape plugin.
5. If you did not set the class path, paste the command into an editor and replace "java" by "java -cp C:\Users\me\Documents\CoNet\lib\CoNet.jar" in Windows and by "java -cp /Users/me/Documents/CoNet/lib/CoNet.jar" in MacOS/UNIX.

For either OS, please do not copy the example paths, but give the path in which your CoNet.jar is located. Then copy the modified command.

6. Open a command shell. In Windows, you can do so by clicking "Start", then "Run...". A window will open into which you write "cmd". This will open the command prompt window. Alternatively, you can also open the "Command prompt" program located in "Accessories". In MacOS, you can open the terminal application located in /Applications/Utilities.
7. In the command shell, go to the demo folder of the CoNet directory using the command "cd". Example for MacOS:

   ```
   cd /Users/me/Documents/CoNet/demo
   ```

   Example for Windows:

   ```
   cd C:\Users\me\Documents\CoNet\demo
   ```

8. Paste the command. On MacOS/UNIX, you can add an ampersand (&) at the end of the command to send it to the background
9. Push Enter to start the execution of the command.
10. After a few seconds, a network file starting with "cooccurrence" and ending with ".gdl" should have been generated in the demo folder.
11. You can load this network into Cytoscape by selecting it using the "Load" button in CoNet's main menu and then pushing "GO".

Command line tips and tricks

Command line help

You can get a short and a long version of the command line help. For the short version, please type:

```
java be.ac.vub.bsb.cooccurrence.cmd.CooccurrenceAnalyser -h
```

For the long version, type:

```
java be.ac.vub.bsb.cooccurrence.cmd.CooccurrenceAnalyser -H
```

Runtime memory

You can increase java runtime memory with option -Xmx. Example:

```
java -Xmx2000M be.ac.vub.bsb.cooccurrence.cmd.CooccurrenceAnalyser -h
```

Alias

In MacOS/UNIX, you can set an alias to shorten the CoNet call on command line. For this, add the line below to your bash shell configuration file:

```
alias conet="java be.ac.vub.bsb.cooccurrence.cmd.CooccurrenceAnalyser"
```

Then you can call the program as:

```
conet -h
```

Running CoNet on command line with a configuration file

Some options of the CoNet Cytoscape plugin are given to CoNet via a configuration file. These concerns the following options (the corresponding command line option is given in brackets):

- RserveHost (host)
- RservePort (port)
- LineageSeparator (lineage_separator)
- MiImplementation (mi_implementation)
- NoRserveDependency (no_rserve)
- PseudoCounts (pseudocount)
- Poolvar (poolvar)
- DisableSpeedup (disable_speedup)

In addition, a number of other variables can be set via the configuration file (check the command line help for more details).
Here's an example for a configuration file:

```
############ CONET CONFIG #############
# rserve config
rserve_host=127.0.0.1
rserve_port=6311
# phylogenetic lineage
lineage_separator=--
# mutual information computation
mi_implementation=minet
minet_r_batch=true
# access to R
no_rserve=false
no_r=false
# p-value computation
poolvar=false
```

```
# speed-up disabled
disable_speedup=true
```

This configuration file indicates that CoNet makes use of Rserve at the specified host and port. It then specifies the lineage separator string (which is needed for taxon metadata specifying phylogenetic lineages). Furthermore, it enables mutual information computation in minet and, since R is needed for minet usage, allows calls to R via command line and via Rserve. In addition, the configuration switches off the CoNet speed-up, so that CoNet uses the previous (slower) implementation of various similarity measures (disable_speedup).

Example for CoNet with configuration file

Here is an example of running CoNet command line with a configuration file. The example assumes that Rserve is running and minet is installed in R. Please copy the command into one line.

```
java be.ac.vub.bsb.cooccurrence.cmd.CooccurrenceAnalyser
-Z CoNetConfig.txt --method ensemble
--input /CoNet/demo/Costello_2009_oral.txt --matrixtype abundance
--ensemblemethods correl_pearson/dist_bray/
dist_kullbackleibler/sim_mutInfo --minetdisc equalfreq
--format gdl --nantreatment pairwise_omit
--nantreatmentparam 5 --networkmergestrategy union
--stand col_norm --multigraph
--ensembleparams correl_pearson~upperThreshold=0.72/
correl_pearson~lowerThreshold=-0.6/dist_bray~upperThreshold=0.89/
dist_bray~lowerThreshold=0.25/dist_kullbackleibler~upperThreshold=7.16/
dist_kullbackleibler~lowerThreshold=0.45/sim_mutInfo~lowerThreshold=0.6
--filter row_minocc --filterparameter 5.0
--output cooccurrenceNetworkDemo.gdl
```

where CoNetConfig.txt is a file located in the directory where the command is carried out. CoNetConfig.txt has the following content:

```
############ CONET CONFIG #############
# rserve config
no_rserve=false
rserve_host=127.0.0.1
rserve_port=6311
# mutual information computation with minet
mi_implementation=minet
```

Advanced users: Submitting CoNet jobs to a SGE cluster (MacOS/UNIX)

CoNet can send jobs to a SunGridEngine (SGE) cluster (command line option -b). This feature is needed when thousands of permutation iterations need to be carried out. A number of options in the configuration file allow to manage the cluster submission capabilities of CoNet (see list below). Most important is the job_num option, which defines how iterations are split into jobs. For example, if 1,000 permutations need to be carried out, a job_num of 100 will result in the submission of 100 jobs, each run with 10 iterations. In consequence, 100 temporary score files will be created (having 10 lines each), which will be merged into a final randomization score file after completion of all jobs. When submitting jobs to a SGE cluster, the lib_dir, jar_file, temp_dir, queue, job_num and memory options should all be set, and it is advisable to set both keep_tmpscores and no_rserve to true. Here's an example configuration:

```
############ CONET CONFIG #############
no_rserve=true
no_r=true
####### cluster configuration
# location of the CoNet jar
lib_dir=/path/to/my/conet/lib/folder/
# name of the CoNet jar
jar_file=CoNet.jar
# temporary score files are stored here
tmp_dir=/path/to/my/temp/folder
# SGE queue name
queue=all.q
# memory allocated to java (via option -Xmx)
memory=4000
# number of jobs into which iterations are splitted
job_num=100
# keep the temporary score files
keep_tmpscores=true
# dry run: test run that does not submit jobs to the cluster
dry_run=false
# do not keep launcher scripts
keep_scripts=false
```

When the CoNet command exits before completion of the jobs, temporary score files can be concatenated after completion of the jobs using command line option --restorefromscorefolder

List of cluster options

Here is the list of all cluster-related options, to be set via the configuration file.

- lib_dir = location of the CoNet jar file (not including the jar file name)
- jar_file = the name of the jar file
- tmp_dir = location of temporary score files
- keep_tmpscores = if true, temporary score files are kept in the temp directory
- job_num = the number of jobs submitted to the cluster (the number of iterations run in each job is determined automatically from the number of requested iterations)
- queue = the name of the queue
- memory = the memory in MB allocated to the job (defaults to 1000)
- user_cmd = a bash command carried out before the job is started (e.g. user_cmd=module load java). The user command can include additional SGE directives (e.g. user_cmd=#$ -m ae -M me@somewhere.com). Several commands can be given by emulating the new line separator with three question marks (e.g. user_cmd=. /etc/profile.d/modules.sh???module load java).
- launch_dir = directory in which cluster submission is launched (defaults to the current directory)
- keep_scripts = if true, job submission scripts are kept
- dry_run = if true, job submission scripts are created but not launched
- quiet_run = if true, output and error streams will be directed to /dev/null, so the creation of job-specific log files is suppressed

Advanced users: Parallelizing CoNet (MacOS/UNIX)

CoNet randomization can be split in several jobs to speed it up. Conet supports this parallelization in two ways: SGE-cluster submission (see above) and user-made wrappers. If you would like to run several CoNet jobs in parallel using your own wrapper, the following options are of interest to you: You can specify -f with value "randscore". This causes CoNet to write random scores instead of a network to the output. The original network can be provided via option -I. If it is provided (and -f is set to randscore), the original network will not be recomputed, but read in from the file given via -I. If the file given via -I does not exist yet, the original network will be exported to this file. Thus, you can set up parallelization in the following way: The first step is to compute the original network scores:

```
java -Xmx2000m -cp CoNet.jar
be.ac.vub.bsb.cooccurrence.cmd.CooccurrenceAnalyser
-i input.txt -f randscore -E correl_spearman/dist_bray --method ensemble
--ensembleparamfile thresholds.txt --multigraph --pvaluemerge brown
-F rand --iterations 1 -g 0.05 --resamplemethod shuffle_rows
-I oriscores.txt -K edgeScores --scoreexport
--output randomScores.0 > oriscores.log &
```

Next, you can create N jobs (where i is the job index going from 1 to N), by launching the following command N times:

```
java -Xmx2000m -cp CoNet.jar
be.ac.vub.bsb.cooccurrence.cmd.CooccurrenceAnalyser
-i input.txt -f randscore -E correl_spearman/dist_bray --method ensemble
--ensembleparamfile thresholds.txt --multigraph --pvaluemerge brown
-F rand --iterations 10 -g 0.05 --resamplemethod shuffle_rows
-I oriscores.txt -K edgeScores --scoreexport
--output randomScores.i > randscores_i.log &
```

Finally, you can merge all separately generated random score files (randomScores.i with i from 1 to N) by appending them to the original score file (oriscores.txt), thus creating your final permutation or bootstrap score file. If all your random score files are in one directory and if their names start with "randomScores.", you can let CoNet do the final merge. For this, point to the random score directory by setting tmp_dir in the configuration file to this directory and add option --restorefromscorefolder on command line. An example command could look like this:

```
java -Xmx2000m -cp CoNet.jar
be.ac.vub.bsb.cooccurrence.cmd.CooccurrenceAnalyser
-i input.txt -f gdl -E correl_spearman/dist_bray --method ensemble
--ensembleparamfile thresholds.txt --multigraph --pvaluemerge brown
-F rand --iterations 100 -g 0.05 --resamplemethod shuffle_rows
-I oriscores.txt -K edgeScores --restorefromscorefolder
-Z CoNetConfig.txt --output network.gdl > restore.log &
```

Advanced users: Ensemble Pipeline Bash Script (MacOS/UNIX)

In the cmd folder of the CoNet distribution, there is an example for a bash script that runs the steps of the ensemble part of the pipeline published in PLoS Computational Biology 8, e1002606. The example bash script uses input data from the third CoNet tutorial. It assumes that it is located in a folder that has two sub-folders "Input" and "Output", where input files (input matrix and metadata) are located in the input folder and the output files will be written to the output folder. Don't forget to give the script execution permission (chmod 755 cooc.sh).
The bash script runs all required network construction steps in one go, that is computation of initial thresholds, generation of renormalized permutation and bootstrap scores and final network construction. The renormalized permutation score computation is the longest step, it takes around 5 minutes on an 8GB RAM machine.
You can run a first test by setting PERMUT, BOOT and RESTORE to false and enabling COMPUTE_THRESHOLDS and TEST instead.
If you want to enable CLUSTER, make sure you have the SGE cluster management

system installed. Then, add required cluster options to the CoNetConfig.txt and CoNetConfigBoot.txt configuration files. Both configuration files should specify the location of the CoNet jar file (lib_dir and jar_file) as well as the requested job number (job_num) and memory (memory). The CoNetConfig.txt configuration file should in addition point to the location of the directory where temporary permutation score files will be saved (tmp_dir). Likewise, the CoNetConfigBoot.txt configuration file should point to a different location, where temporary bootstrap score files will be saved. Note that in case you keep temporary score files (keep_tmpscores=true), you can restore random score distributions from these files using option --restorefromscorefolder later on.

References, resources and related tools

References

1. Agrawal, R., Imielinski, T. & Swami, A. "Mining Association Rules between Sets of Items in Large Databases" in ACM SIGMOD Conference (eds. Buneman, P. & Jajodia, S.) 207-216 (ACM Press, 1993).
2. Aitchison, J., "A Concise Guide to Compositional Data Analysis" in CDA Workshop Girona (2003).
3. Brown, M.B., "A Method for Combining Non-Independent, One-Sided Tests of Significance." Biometrics 31, 987-992 (1975).
4. Costello, E.K. et al. "Bacterial Community Variation in Human Body Habitats Across Space and Time." Science 326, 1694-1697 (2009).
5. Ellson, J., Gansner, E.R., Koutsofios, E., North, S.C. & Woodhull, G. in GRAPH DRAWING SOFTWARE (Springer-Verlag, 2003).
6. Faith J.J., et al. "Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles." PLoS Biol, 5:54-66 (2007).
7. Hu, Z. et al. "VisANT 3.5: multi-scale network visualization, analysis and inference based on the gene ontology." Nucleic Acids Research 37, W115-W121 (2009).
8. Lallich S., Teytaud O., & Prudhomme E. "Statistical inference and data mining: false discoveries control." 17th Compstat Symposium of the IASC:325-336 (2006).
9. Legendre, P. & Legendre, L. "Numerical ecology" (Elsevier Science B.V., Amsterdam, 1983).
10. Margolin, A.A. et al. "ARACNE: An Algorithm for the Reconstruction of Gene Regulatory Networks in a Mammalian Cellular Context." BMC Bioinformatics, 1-15 (2006)
11. Meyer, P.E., Lafitte, F. & Bontempi, G. "minet: A R/Bioconductor Package for Inferring Large Transcriptional Networks Using Mutual Information." BMC Bioinformatics, 1-10 (2009)
12. Meyer P.E., et al. "Information-theoretic inference of large transcriptional regulatory networks." EUROSIP J. Bioinform. Syst. Biol. 79879 (2007).

Resources

Below, resources are listed of which CoNet makes use.

- **apriori** http://www.borgelt.net/apriori.html
  association rule mining implementation
- **Apache Commons Math** http://commons.apache.org/math/
  java mathematics and statistics library

- **ARACNE matlab script** http://icbp.stanford.edu/software/FastPairMI/
  fast implementation of mutual information
- **Colt** http://acs.lbl.gov/software/colt/
  statistical computing with java
- **Java statistical classes (jsc)** http://www.jsc.nildram.co.uk/
  statistical computing with java
- **Lobo browser** http://lobobrowser.org/java-browser.jsp
  pure java browser
- **minet** http://www.bioconductor.org/packages/release/bioc/html/minet.html
  network inference, computation of mutual information
- **Rserve** http://www.rforge.net/Rserve/
  java-R bridge

Links to some related tools

- **ARACNE (standalone)** http://wiki.c2b2.columbia.edu/califanolab/index.php/Software/ARACNE
  gene regulatory network inference algorithm, Califano lab
- **BANJO (standalone and Java library)** http://www.cs.duke.edu/~amink/software/banjo/
  Banjo is a software application and framework for structure learning of static and dynamic Bayesian networks, Alexander J. Hartemink and colleagues
- **BNW** http://compbio.uthsc.edu/BNW/sourcecodes/home.php
  Bayesian Network Webserver for Biological Network Modeling, by Jesse D. Ziebarth, Anindya Bhattacharya and Yan Cui
- **compare-profiles (RSAT tool suite, command line tool)** http://rsat.ulb.ac.be/rsat
  compare all pairs of rows in a matrix using one of several measures, by Jacques van Helden
- **Community Analyzer (stand alone)** http://metagenomics.atc.tcs.com/Community_Analyzer/
  explores inter-microbial interactions across metagenomes, TATA Consultancy Services, Bio-Sciences Division
- **ccrepe** http://huttenhower.sph.harvard.edu/ccrepe
  R package designed to detect significant correlations in compositional data, Huttenhower lab
- **Cyni Toolbox (Cytoscape plugin)** http://apps.cytoscape.org/apps/cynitoolbox
  Cytoscape Network Inference Toolbox puts together several tools that allow inferring networks from bio data, Benno Schwikowski and Oriol Guitart Pla
- **efficient estimation of covariance and (partial) correlation** http://strimmerlab.org/software/corpcor/
  R package with shrinkage estimator for covariance matrix, Strimmer lab
- **ExpressionCorrelation (Cytoscape plugin)** http://www.baderlab.org/Software/ExpressionCorrelation

computes a similarity network from either the genes or conditions in an expression matrix, Bader lab

- **Fast Local Similarity Analysis (stand alone)** http://hallam.microbiology.ubc.ca/fastLSA/install/index.html
  computes a directed similarity network from time series data, Hallam lab
- **Extended Local Similarity Analysis (LSA) (standalone and as part of the Galaxy pipeline)** http://meta.usc.edu/softs/lsa/
  computes a directed similarity network from time series data, Sun lab
- **GeneNet (R package)** http://strimmerlab.org/software/genenet/index.html
  GeneNet is an R package for learning high-dimensional dependency networks from genomic data (e.g. gene association networks), Strimmer lab
- **MENA (web server)** http://ieg2.ou.edu/MENA
  Molecular Ecological Network Analysis Pipeline, Zhou lab
- **MONET (Cytoscape plugin and web service)** http://monet.kisti.re.kr
  regulatory network inference algorithm based on Bayesian network learning, by Phil Hyoun Lee and Doheon Lee
- **NetCutter (stand alone)** http://muller.group.ifom-ieo-campus.it/
  co-occurrence networks identification and analysis, by Heiko Mueller and Francesco Mancuso
- **Picante (R package)** http://picante.r-forge.r-project.org/
  phylogeny and trait diversity, community null models, by Peter Cowan, Matthew Helmus and Steven Kembel
- **SparCC (stand alone)** https://bitbucket.org/yonatanf/sparcc
  Python module for computing correlations in compositional data, by Yonatan Friedman
- **WGCNA** http://labs.genetics.ucla.edu/horvath/CoexpressionNetwork/Rpackages/WGCNA/
  R package for weighted correlation network analysis, by Peter Langfelder and Steve Horvath