NetSAM User Guide

5/15/2021

1 Introduction

The last decade of systems biology research has demonstrated that networks rather than individual genes govern the onset and progression of complex diseases. Meanwhile, real world complex networks usually exhibit hierarchical organization, in which nodes can be combined into groups that can be further combined into larger groups, and so on over multiple scales. Thus, identifying the hierarchical organization of a network becomes indispensable in complex disease studies. A traditional and useful method for revealing hierarchical architecture of network is hierarchical clustering, which groups data over a variety of scales by creating a hierarchical tree. However, hierarchical clustering has three major limitations.

- There are many different leaf node orderings consistent with the structure of a hierarchical tree, and hierarchical clustering does not optimize the ordering.
- Hierarchical clustering does not assess the statistical significance of the modular organization of a network.
- It does not specify relevant hierarchical levels and modules at different scales.

To address these limitations, we developed the NetSAM (Network Seriation and Modularization) package which identifies the hierarchical modules from a network (network modularization) and find a suitable linear order for all leaves of the identified hierarchical organization (network seriation). NetSAM takes an edge-list representation of a weighted or unweighted network as an input and generates as files that can be used as an input for the one-dimensional network visualization tool NetGestalt (`http://www.netgestalt.org`) or other network analysis. NetSAM uses random walk distance-based hierarchical clustering to identify the hierarchical modules of the network and then uses the optimal leaf ordering (OLO) method to optimize the one-dimensional ordering of the genes in each module by minimizing the sum of the pair-wise random walk distance of adjacent genes in the ordering. The detailed description of the NetSAM method can be found in our published Nature Methods paper "NetGestalt: integrating multidimensional omics data over biological networks" (`http://www.nature.com/nmeth/journal/v10/n7/full/nmeth.2517.html`.

The NetSAM package can also generate correlation network (e.g. co-expression network) based on the input matrix data, perform seriation and modularization analysis for correlation network and calculate the associations between the sample features and modules or identify the associated GO terms for the modules.

2 Environment

NetSAM requires R version 3.0.0 or later, which can be downloaded from the website `http://www.r-project.org`. Because the seriation step requires pair-wise distance between all nodes, NetSAM is memory consuming. We recommend to use the 64 bit version of R to run the NetSAM. For networks with less than 10,000 nodes, we recommend to use a computer with at least 8GB memory. Using our computer with 2.7 GHz Intel Core i5 processor and 8GB 1333 MHz DDR3 memory, NetSAM took 402 seconds to analyze the HPRD network (`http://www.hprd.org`) with 9198 nodes. For networks with more than 10,000 nodes, a computer with at least 16GB memory is recommended. NetSAM package requires the following packages: `igraph` (>=0.6-1), `seriation` (>=1.0-6), `WGCNA` (>=1.34.0), `doParallel` (>=1.0.10), `foreach` (>=1.4.0), `tools` (>=3.0.0), `biomaRt` (>=2.18.0), `GO.db` (>=2.10.0), `R2HTML` (>=2.2.0) and `survival` (>=2.37-7), which can be installed as follows.

```
install.packages("igraph")
install.packages("seriation")
install.packages("WGCNA")
install.packages("snow")
install.packages("doSNOW")
install.packages("foreach")
source("http://bioconductor.org/biocLite.R")
biocLite("biomaRt")
biocLite("GO.db")
install.packages("R2HTML")
install.packages("survival")
```

3 Network Seriation and Modularization

After building up the basic environment mentioned above, the users can install the NetSAM package and use it to analyze networks.

```
library("NetSAM")
inputNetworkDir <- system.file("extdata", "exampleNetwork.net", package = "NetSAM")
outputFileName <- paste(getwd(), "/NetSAM", sep = "")
result <- NetSAM(inputNetwork = inputNetworkDir, outputFileName = outputFileName,
    outputFormat = "nsm", edgeType = "unweighted", map_to_genesymbol = FALSE,
    organism = "hsapiens", idType = "auto", minModule = 0.003, stepIte = FALSE,
    maxStep = 4, moduleSigMethod = "cutoff", modularityThr = 0.2,
    ZRanNum = 10, PerRanNum = 100, ranSig = 0.05, edgeThr = (-1),
    nodeThr = (-1), nThreads = 3)
```

3.1 Input

- `inputNetwork` is the network under analysis. `inputNetwork` can be the directory of the input network file including the file name with "net" extension. If `edgeType` is "`unweighted`", each row represents an edge with two node names separated by a tab or space. If `edgeType` is "`weighted`", each row represents an edge with two node names and edge weight separated by a tab or space. `inputNetwork` can also be a data object in R (data object must be igraph, graphNEL, matrix or data.frame class).
- `outputFileName` is the name of the output file.
- `outputFormat` is the format of the output file. "`nsm`" format can be used as an input to NetGestalt (`http://www.netgestalt.org`), "`gmt`" format can be used to do other network analysis (e.g. as an input in GSEA (Gene Set Enrichment Analysis) to do module enrichment analysis) and "`none`" means the function will not output a file.
- Because pathway enrichment analysis in NetGestalt is based on gene symbol, setting `map_to_genesymbol` as `TRUE` can transform other ids in the network into gene symbols and thus allow users to do functional analysis based on the identified modules. If the input network is not a biology network or users do not plan to do enrichment analysis in the NetGestalt, users can set `map_to_genesymbol` as `FALSE`. The default is `FALSE`.
- `organism` is the organism of the input network. Currently, the package supports the following nine organisms: `hsapiens`, `mmusculus`, `rnorvegicus`, `drerio`, `celegans`, `scerevisiae`, `cfamiliaris`, `dmelanogaster` and `athaliana`. The default is "`hsapiens`".
- `idType` is the id type of the input network. MatSAM will use BiomaRt package to transform the input ids to gene symbols based on `idType`. User can also set `idType` as "`auto`" that means MatSAM will automatically search the id type of the input data. However, this may take 10 minutes depending on the users' network connection speed. The default is "`auto`".

- `minModule` is the minimum percentage of nodes in a module. The minimum size of a module is calculated by multiplying `minModule` by the number of nodes in the whole network. If the size of a module identified by the function is less than the minimum size, the module will not be further partitioned into sub-modules. The default is 0.003 which means the minimum module size is `30` if there are $10,000$ nodes in the whole network. If the minimum module size is less than 5, the minimum module size will be set as 5. The `minModule` should be less than 0.2.
- Because NetSAM uses random walk distance-based hierarchical clustering to reveal the hierarchical organization of an input network, it requires a specified length of the random walks. If `stepIte` is `TRUE`, the function will test a range of lengths ranging from 2 to `maxStep` to get the optimal length. Otherwise, the function will directly use `maxStep` as the length of the random walks. The default `maxStep` is 4. Because optimizing the length of the random walks will take a long time, if the network is too big (e.g. the number of edges is over $200,000$), we suggest to set `stepIte` to `FALSE`.
- To test whether a network under consideration has a non-random internal modular organization, the function provides three options for parameter `moduleSigMethod`: "`cutoff`", "`zscore`" and "`permutation`". "`cutoff`" means if the modularity score of the network is above a specified cutoff value, the network will be considered to have internal organization and will be further partitioned. For "`zscore`" and "`permutation`", the function will first generate a set of random modularity scores. Based on a unweighted network, the function uses the edge switching method to generate a given number of random networks with the same number of nodes and an identical degree sequence and calculates the modularity scores for these random networks. Based on a weighted network, the function shuffles the weights of all edges and calculate the modularity scores for network with random weights. Then, "`zscore`" method will transform the real modularity score to a z score based on the random modularity scores and then transform the z score to a p value assuming a standard normal distribution. The "`permutation`" method will compare the real modularity score with the random ones to calculate a p value. Finally, under a specified significance level, the function determines whether the network can be further partitioned. The default is "`cutoff`".
- `modularityThr` is the threshold of modularity score for the "`cutoff`" method. The default value is 0.2.
- `ZRanNum` is the number of random networks that will be generated for the "`zscore`" calculation. The default value is 10.
- `PerRanNum` is the number of random networks that will be generated for the "`permutation`" p value calculation. The default value is 100.
- `ranSig` is the significance level for determining whether a network has non-random internal modular organization for the "`zscore`" or "`permutation`" methods.
- If the network is too big, it will take a long time to identify the modules. Thus, the function provides the option to set the threshold of number of edges and nodes as `edgeThr` and `nodeThr`. If the size of network is over the threshold, the function will stop and the users should change the parameters and re-run the function. We suggest to set the threshold for node as $12,000$ and the threshold for edge as $300,000$. The default value is -1 which means there is no limitation for the network.
- `nThreads` is the number of cores used for parallel processing. The default value is 3.

3.2 Output

If output format is "`nsm`", the function will output not only an "`nsm`" file but also a list object containing module information, gene order information and network information. If output format is "`gmt`", the function will output the "`gmt`" file and a matrix object containing the module and annotation information.

4 Network Analyzer

The `NetAnalyzer` function calculates the degree, clustering coefficient, betweeness and closeness centrality for each node and the shortest path distance for each pair of nodes. The function can also plot the distributions

for these measurements.

```
library("NetSAM")
inputNetwork <- system.file("extdata", "exampleNetwork.net", package = "NetSAM")
outputFileName <- paste(getwd(), "/NetSAM", sep = "")
NetAnalyzer(inputNetwork, outputFileName, "unweighted")
```

4.1 Input

- `inputNetwork` is the path to the network file under analysis or a data object. If it is a path to the network file, the file must have `.net`extension. If `edgeType` is `unweighted`, each row represents an edge with two node names separated by a tab or space. If edgeType is `weighted`, each row represents an edge with two node names and edge weight separated by a tab or space. If `inputNetwork` is a data object, if must be of one of the following classes: `igraph`, `graphNEL`, `matrix` or `data.frame`.
- `edgeType` can be either "`unweighted`" or "`weighted`"
- `outputFileName` is the name of the output file.

4.2 Output

The `NetAnalyzer` function will output two "`txt`" files and five "`pdf`" files. Two "`txt`" files contain degree, clustering coefficient, betweeness and closeness centrality for each node and the shortest path distance for each pair of nodes. Five "`pdf`" files are the distributions of these measurements.

5 mergeDuplicate

The `mergeDuplicate` function will merge the duplicate Ids in the matrix and return the matrix with unique Ids. This function can also used to merge the duplicate mapped Ids when transforming the Ids of data matrix to other Ids.

```
library("NetSAM")
inputMatDir <- system.file("extdata", "exampleExpressionData_nonsymbol.cct",
    package = "NetSAM")
inputMat <- read.table(inputMatDir, header = TRUE, sep = "\t", stringsAsFactors = FALSE,
    check.names = FALSE)
mergedData <- mergeDuplicate(id = inputMat[, 1], data = inputMat[,
    2:ncol(inputMat)], collapse_mode = "maxSD")
```

5.1 Input

- `id` are the duplicated Ids. It should be a vector object.
- `data` is the corresponding data matrix that has the same number of rows with id and should be a matrix or data.frame object.
- `collapse_mode` is the method to collapse duplicate ids. "`mean`", "`median`", "`maxSD`", "`maxIQR`", "`max`" and "`min`"represents the mean, median, max standard deviation, max interquartile range, maximum and minimum of values for ids in each sample respectively. The default value is "`maxSD`".

5.2 Output

The function returns a data matrix with unique Ids.

6 Mapping other ids to gene symbols

To perform enrichment analysis in NetGestalt, the gene ids in each module should be gene symbols. The `mapToSymbol` function can transform other ids from a gene list, network, matrix, sbt file or sct file to gene symbols.

```
library("NetSAM")
print("transform ids from a gene list to gene symbols...")
geneListDir <- system.file("extdata", "exampleGeneList.txt", package = "NetSAM")
geneList <- read.table(geneListDir, header = FALSE, sep = "\t", stringsAsFactors = FALSE)
geneList <- as.vector(as.matrix(geneList))
geneList_symbol <- mapToSymbol(inputData = geneList, organism = "hsapiens",
    inputType = "genelist", idType = "affy_hg_u133_plus_2")

print("transform ids in the input network to gene symbols...")
inputNetwork <- system.file("extdata", "exampleNetwork_nonsymbol.net",
    package = "NetSAM")
network_symbol <- mapToSymbol(inputData = inputNetwork, organism = "hsapiens",
    inputType = "network", idType = "entrezgene", edgeType = "unweighted")

print("transform ids in the input matrix to gene symbols...")
inputMatDir <- system.file("extdata", "exampleExpressionData_nonsymbol.cct",
    package = "NetSAM")
matrix_symbol <- mapToSymbol(inputData = inputMatDir, organism = "hsapiens",
    inputType = "matrix", idType = "affy_hg_u133_plus_2", collapse_mode = "maxSD")

print("transform ids in the sbt file to gene symbols...")
inputSBTDir <- system.file("extdata", "exampleSBT.sbt", package = "NetSAM")
sbt_symbol <- mapToSymbol(inputData = inputSBTDir, organism = "hsapiens",
    inputType = "sbt", idType = "affy_hg_u133_plus_2")

print("transform ids in the sct file to gene symbols...")
inputSCTDir <- system.file("extdata", "exampleSCT.sct", package = "NetSAM")
sct_symbol <- mapToSymbol(inputData = inputSCTDir, organism = "hsapiens",
    inputType = "sct", idType = "affy_hg_u133_plus_2", collapse_mode = "min")
```

6.1 Input

- `inputData`: `mapToSymbol` function supports five different types of data: "`genelist`", "`network`", "`matrix`", "`sbt`" and "`sct`". For "`genelist`" type, `inputData` should be a vector containing gene ids. For "`network`" type, `inputData` can be the path to the input network file and the file must have a "`.net`" extension. If `edgeType` is "`unweighted`"", each row represents an edge with two node names separated by a tab or space. If edgeType is"`weighted`", each row represents an edge with two node names and edge weight separated by a tab or space. `inputNetwork` can also be a data object (data object must be `igraph`, `graphNEL`, `matrix` or `data.frame` class). For"`matrix`" type, `inputData` should be a path to a file with extension "`cct`" or "`cbt`" or a matrix or data.frame object. The data should have column names. If the data do not have any row name, the first column of the data should be the ids. For "`sbt`" type, `inputData` should be a path to a file with extension "`.sbt`". For "`sct`" type, inputData should be a directory containing a file name with extension "`.sct`". The detail information of "`cct`" , "`cbt`", "`sbt`", "`sct`" format can be found in the NetGestalt user manual (`http://www.netgestalt.org`).
- `inputType`: the type of the input data. see detailed information in `inputData` parameter.
- `idType`: see `idType` in `NetSAM` function
- `collapse_mode` is the method used to collapse duplicate ids. See details in `mergeDuplicate` section.

For SCT file, we suggest to use "max" or "min" to collapse duplicate ids in the statistic data.

- is_outputFile: If is_outputFile is TRUE, the function will output the transformed data to a file. The default value is FALSE.
- outputFileName: the output file name.
- verbose: whether the function reports extra information. The default value is TRUE.

6.2 Output

The function will output a object with transformed data. If the ids in the input data can not be transformed to gene symbols, the function will output NULL. If outputFileName is TRUE, the functionsaves the transformed data to a file.

7 Construction of correlation network

The MatNet function can be used to construct a correlation network based on the input matrix.

```
library("NetSAM")
inputMatDir <- system.file("extdata", "exampleExpressionData.cct",
    package = "NetSAM")
matNetwork <- MatNet(inputMat = inputMatDir, collapse_mode = "maxSD",
    naPer = 0.7, meanPer = 0.8, varPer = 0.8, corrType = "spearman",
    matNetMethod = "rank", valueThr = 0.6, rankBest = 0.003, networkType = "signed",
    netFDRMethod = "BH", netFDRThr = 0.05, idNumThr = (-1), nThreads = 3)
```

7.1 Input

- inputMat is the path to a file with extension ".cct" or a matrix or data.frame object. The data should have column names. If the data do not have any row name, the first column of the data should be the ids.
- If the input matrix data contains the duplicate ids, the function will collapse duplicate ids based on the collapse mode. "mean", "median", "maxSD" and "maxIQR" represents the mean, median, max standard deviation or max interquartile range of id values in each sample respectively. The default value is "maxSD".
- To remove ids with missing values in most of samples, the function calculates the percentage of missing values in all samples for each id and removes ids with over naPer$\times$100% missing values in all samples. The default value is 0.7.
- To remove ids with low values, the function calculates the mean of values for each id in all samples and keeps top meanPer$\times$100% ids based on the mean values. The default value is 0.8.
- Based on the remained ids filtered by meanPer, the function can also remove less variable ids by calculating the standard deviation of values for each id in all samples and keeping top varPer$\times$100% ids based on the standard deviation. The default value is 0.8.
- corrType: a character string indicating which correlation coefficient is to be computed for each pair of ids. The function supports "spearman" (default) or "pearson" method.
- MatNet function supports three methods to construct correlation network: "value","rank" and "directed".
  - "value": the correlation network only keeps id pairs with correlations over cutoff threshold valueThr;

  - "rank": for each id A, the function first selects ids that significantly correlate with id A and then extracts a set of candidate neighbors (the number of ids is rankBest) from the significant set that are most similar to id A. Then, for each id B in the candidate neighbors of id A , the function also extracts the same number of ids that are significantly correlated and most similar to id B. If id A is also the candidate neighbors of id B, there will be an edge between id A and id B. Combining all edges can construct a correlation network;
  - "directed": the function will only keep the most significant id for each id as the edge. A directed correlation network is constructed by combining all edges.

- valueThr: the correlation cutoff threshold for "value" method
- rankBest is the percentage of ids that are most similar to one id for "rank" method. The default value is 0.003 which means the "rank" method will select top 30 most similar ids for each id if the number of ids in the matrix is $10,000$.
- networkType: if set as "unsigned", the correlation of all id pairs will be changed to absolute values. The default value is "signe".
- netFDRMethod: the p value adjustment methods for "rank" and "directed" methods. The default value is "BH".
- netFDRThr is FDR threshold for identifying significant pairs for "rank" and "directed" methods.
- idNumThr: If the matrix contains too many ids, it will take a long time and use a lot of memory to identify the modules.
  Thus, the function provides the option to set the threshold of number of ids for further analysis. After filtering by meanPer and varPer, if the number of ids is still larger than idNumThr, the function will select top idNumThr ids with the largest variance. The default value is -1, which means there is no limitation for the matrix.

7.2 Output

The function will output a matrix with two columns.

8 Construction of consensus network

To increase robustness against errors in data, a bootstrapping procedure is used to construct a consensus network.

```
library("NetSAM")
inputMatDir <- system.file("extdata", "exampleExpressionData.cct",
    package = "NetSAM")
data <- read.table(inputMatDir, header = TRUE, row.names = 1, stringsAsFactors = FALSE)
net <- consensusNet(data = data, organism = "hsapiens", bootstrapNum = 10,
    naPer = 0.5, meanPer = 0.8, varPer = 0.8, method = "rank_unsig",
    value = 3/1000, pth = 1e-06, nMatNet = 2, nThreads = 4)
```

8.1 Input

- data should contain a file name with extension cct or cbt or a matrix or data.frame object in R. The first column and first row of the cct or cbt file should be the row and column names, respectively and other parts are the numeric values. The detail information of cct or cbt format can be found in the manual of NetGestalt (http://www.netgestalt.org). A matrix or data.frame object should have row and column names and only contain numeric or integer values.
- organism is the organism of the input network. Currently, the package supports the following nine organisms: hsapiens, mmusculus, rnorvegicus, drerio, celegans, scerevisiae, cfamiliaris,

dmelanogaster and athaliana. The default is "hsapiens".

- bootstrapNum: Number of bootstrap data sets generated. Default is 100.
- naPer: To remove ids with missing values in most of samples, the function calculates the percentage of missing values in all samples for each id and removes ids with over naPer missing values in all samples. The default value is 0.5.
- meanPer: To remove ids with low values, the function calculates the mean of values for each id in all samples and remains top meanPer ids based on the mean. The default value is 0.8.
- varPer: Based on the remaining ids filtered by meanPer, the function can also remove less variable ids by calculating the standard deviation of values for each id in all samples and remaining top varPer ids based on the standard deviation. The default value is 0.8.
- method: Method used for constructing correlation network with MatNet. Currently supports "rank", "value" and "rank_unsig". Default is "rank_unsig".
- value: The corresponding value set for method. Default is 0.003.
- pth: p-value threshold for including an edge. Default is 1.0e-6.
- nMatNet: The number of concurrent running MatNet processes, default is 2.
- nThreads: consensusNet function supports parallel computing based on multiple cores. The default is 4.

8.2 Output

The function will output a matrix with two columns.

9 Test input data format

The testFileFormat function will test the format of the input data matrix and annotation data and return the standardized data matrix and sample annotation data.

```
library("NetSAM")
inputMatDir <- system.file("extdata", "exampleExpressionData.cct",
    package = "NetSAM")
sampleAnnDir <- system.file("extdata", "sampleAnnotation.tsi", package = "NetSAM")
formatedData <- testFileFormat(inputMat = inputMatDir, sampleAnn = sampleAnnDir,
    collapse_mode = "maxSD")
```

9.1 Input

- inputMat: a file name or a matrix or data.frame object.
- sampleAnn: a file name or a data.frame object. If the users set inputMat as "", the function only tests format of sample annotation data. If the users set sampleAnn as"", the function only tests format of data matrix.
- If the input data contains the duplicate ids, the function will collapse duplicate ids based on the collapse_mode. "mean", "median", "maxSD" and "maxIQR" represents the mean, median, max standard deviation or max interquartile range of values for ids in each sample. The default is "maxSD".

9.2 Output

If there is no format error, the function will return the standardized data matrix and sample annotation data. Otherwise, it will output the detailed sources of errors.

10 Identification of the associations between sample features and modules

The `featureAssociation` function can be used to calculate the associations between sample features in the input sample annotation data and the modules identified by `NetSAM` or `MatSAM` functions.

```
library("NetSAM")
inputMatDir <- system.file("extdata", "exampleExpressionData.cct",
    package = "NetSAM")
sampleAnnDir <- system.file("extdata", "sampleAnnotation.tsi", package = "NetSAM")
data(NetSAMOutput_Example)
outputHtmlFile <- paste(getwd(), "/featureAsso_HTML", sep = "")
featureAsso <- featureAssociation(inputMat = inputMatDir, sampleAnn = sampleAnnDir,
    NetSAMOutput = netsam_output, outputHtmlFile = outputHtmlFile,
    CONMethod = "spearman", CATMethod = "kruskal", BINMethod = "ranktest",
    fdrmethod = "BH", pth = 0.05, collapse_mode = "maxSD")
```

10.1 Input

- `inputMat` should be a path to a "`cct`" file or a `matrix` or `data.frame` object. The data should contain column names. If the data do not have any row name, the first column of the data should be the ids.
- `sampleAnn` should a path to a "`tsi`" file extension or a `data.frame` object. The detail information of "`tsi`" format can be found in the NetGestalt manual . The first row of the sample annotation data is the feature names. The second row is feature types. The function supports four types: `BIN` (binary feature, such as male and female), `CAT` (category feature, such as stage i, stage ii and stage iii), `CON` (continuous feature, such as age) and `SUR` (survival data, such as overall survival). The third row is the feature categories. If there is no category information for the features, the sample information will start from the third row. The first column is the sample names.
- `NetSAMOutput` is the list object from the NetSAM or MatSAM functions.
- `outputHtmlFile` is the output directory of the HTML file.
- `CONMethod` is the method used to calculate the associations between modules and continuous features. The function provides two methods: "`spearman`" and "`pearson`" and the default value is "`spearman`".
- `CATMethod` is the method used to calculate the associations between modules and category features. The function provides two methods: "`anova`" and "`kruskal`" and the default value is "`kruskal`".
- `BINMethod` is the method used to calculate the associations between modules and binary features. The function provides two methods: "`ttest`" and "`ranktest`" and the default value is "`ranktest`".
- `fdrmethod` is the FDR used method for identifying the significantly associated GO terms. The default value is "`BH`".
- `pth` is the threshold of the p values to be identified as significant associations.
- If the input matrix data contains the duplicate ids, the function will collapse duplicate ids based on `collapse_mode`. "`mean`", "`median`", "`maxSD`" and "`maxIQR`" represents the mean, median, max standard deviation or max interquartile range of values for ids in each sample. The default is "`maxSD`".

10.2 Output

The function will output a `data.frame` object and a HTML file to show the significant associations.

11 Identification of the associated GO terms for the modules

The `GOAssociation` function can be used to identify the associated GO terms for the modules identified by `NetSAM` or `MatSAM` functions.

```
library("NetSAM")
data(NetSAMOutput_Example)
outputHtmlFile <- paste(getwd(), "/GOAsso_HTML", sep = "")
GOAsso <- GOAssociation(NetSAMOutput = netsam_output, outputHtmlFile = outputHtmlFile,
    organism = "hsapiens", fdrmethod = "BH", fdrth = 0.05, topNum = 5)
```

11.1 Input

- `NetSAMOutput`: the list object from the `NetSAM` or `MatSAM` functions.
- `outputHtmlFile`: the output directory for the HTML file.
- `organism`: the organism type of the input data matrix that has been used to identify the modules. Currently, the package supports the following nine organisms: `hsapiens`, `mmusculus`, `rnorvegicus`, `drerio`, `celegans`, `scerevisiae`, `cfamiliaris`, `dmelanogaster` and `athaliana`. The default is value is "`hsapiens`".
- `outputType`: the type of output associated GO terms . The function supports two types:
  - `significant`: all GO terms that are significantly associated under a certain FDR threshold
  - `top`: the function first sorts all GO terms based on their hypergeometric test p values and then selects top GO terms as the associated terms. The default value is `significant`.
- `fdrmethod`: the FDR method for identifying the significantly associated GO terms. The default is "`BH`".
- `fdrth`: the FDR threshold.
- `topNum`: the number of selected top GO terms.

11.2 Output

The function will output a `data.frame` object and a HTML file to show the associated GO terms for each module.

12 Identification of correlation modules

The `MatSAM` function can identify the hierarchical correlation modules.

```
library("NetSAM")
inputMatDir <- system.file("extdata", "exampleExpressionData.cct",
    package = "NetSAM")
sampleAnnDir <- system.file("extdata", "sampleAnnotation.tsi", package = "NetSAM")
outputFileName <- paste(getwd(), "/MatSAM", sep = "")
matModule <- MatSAM(inputMat = inputMatDir, sampleAnn = sampleAnnDir,
    outputFileName = outputFileName, outputFormat = "msm", organism = "hsapiens",
    map_to_symbol = FALSE, idType = "auto", collapse_mode = "maxSD",
    naPer = 0.7, meanPer = 0.8, varPer = 0.8, corrType = "spearman",
    matNetMethod = "rank", valueThr = 0.6, rankBest = 0.003, networkType = "signed",
    netFDRMethod = "BH", netFDRThr = 0.05, minModule = 0.003, stepIte = FALSE,
    maxStep = 4, moduleSigMethod = "cutoff", modularityThr = 0.2,
    ZRanNum = 10, PerRanNum = 100, ranSig = 0.05, idNumThr = (-1),
    nThreads = 3)
```

12.1 Input

- `inputMat` should be a path to a "`cct`" file or a `matrix` or `data.frame` object. The data should contain column names. If the data do not have any row name, the first column of the data should be the ids.
- `sampleAnn` should a path to a "`tsi`" file extension or a `data.frame` object. The detail information of "`tsi`" format can be found in the NetGestalt manual . The first row of the sample annotation data is the feature names. The second row is feature types. The function supports four types: `BIN` (binary feature, such as male and female), `CAT` (category feature, such as stage i, stage ii and stage iii), `CON` (continuous feature, such as age) and `SUR` (survival data, such as overall survival). The third row is the feature categories. If there is no category information for the features, the sample information will start from the third row. The first column is the sample names.
- `outputFormat` is the format of the output file. "`msm`" format can be used as an input to NetGestalt; "`gmt`" format can be used to perform other network analysis (e.g. as an input to GSEA (Gene Set Enrichment Analysis) to perform module enrichment analysis); "`multiple`" means that the function will output five files: a ruler file containing gene order information, a hmi file containing module information,a net file containing correlation network information, a cct file containing the filtered data matrix, and a tsi file containing the sample annotation with standardized format; and "`none`" means that the function will not output any file.
- If `map_to_symbol` is `TRUE`, the function will first change the input id to gene symbol and collapse multiple ids with the same gene symbol based on the collapse mode method before identifying correlation network. The default value is `FALSE`.
- `matNetMethod` specifies the method used to construct correlation network, which has two options: "`value`" and "`rank`". Details can be found in the argument description of the `MatNet` function.

The description of other arguments can be found in the argument description of the proceeding functions.

12.2 Output

The function will output a list object containing module information, gene order information, correlation network and filtered matrix based on the ids in the network. The function will also output two HTML files that contain the significant associations between sample features and modules identified by `featureAssociation` function and associated GO terms for the modules identified by `GOAssociation` function.

Note: When calling `featureAssociation` function, `MatSAM` uses the default parameters. When calling the GOAssociation function, MatSAM sets "`ouputType`" to "`top`" and "`topNum`" to 1. User can use the list object returned by `MatSAM` as the input to these two functions to perform further analysis with different parameters.