Package 'bingat'

October 12, 2022

Type Package

Title Binary Graph Analysis Tools

Version 1.3

Date 2017-07-05

Author Terrence Brooks, Berkley Shands, Skye Buckner-
Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Maintainer Berkley Shands <rpackages@biorankings.com>

Depends R (>= 3.1.0)

Imports matrixStats, network, gplots, doParallel, foreach, parallel,
vegan, stats, graphics

Description Tools to analyze binary graph objects.

License Apache License (== 2.0)

LazyData yes

NeedsCompilation no

Repository CRAN

Date/Publication 2017-07-05 18:30:37 UTC

R topics documented:

bingat-package   *Binary Graph Analysis Tools*

Description

Tools for analyzing binary graphs, including calculating the MLE of a set of binary graphs, comparing MLE of sets of graphs, regression analysis on sets of graphs, using genetic algorithm to identify nodes and edges separating sets of graphs, and generating random binary graphs sampled from the Gibbs distribution.

Details

The following are the types of binary graphs that are accepted:

1. adjMatrix: An entire binary adjacency matrix as a single vector

2. adjMatrixLT: The upper or lower triangle of a binary adjacency matrix as a single vector

3. diag: The diagonal vector on a binary adjacency matrix

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

References

1. Stat Med. 2015 Nov 25. doi: 10.1002/sim.6757. Gibbs distribution for statistical analysis of graphical data with a sample application to fcMRI brain images. La Rosa PS1,2, Brooks TL1, Deych E1, Shands B1,3, Prior F4, Larson-Prior LJ4,5, Shannon WD1,3.

---

braingraphs *Brain Graph Data Set*

Description

A data set containing 38 brain scans each with 20 total nodes.

Usage

```
data(braingraphs)
```

Format

The format is a data frame of 400 rows by 38 columns, with each column being a separate subject and each row being a different edge between 2 nodes. Each column is a 20 by 20 matrix of brain connections transformed into a vector. A value of 1 indicates that subject had a connection at that edge.

---

calcDistance *Calculate the Distance Between Vectors*

Description

This function calculates the distance between two vectors.

Usage

```
calcDistance(x, y, type = "", method = "hamming")
```

Arguments

x, y	Vectors of the same length that contain 1's and 0's.
type	The type of graph being used (adjmatrix or adjmatrixlt). See 'Details'
method	The distance metric to use, currently only "hamming" is supported.

Details

If the type = "adjMatrix" is used, the value will be divided by 2 to account for duplicate comparisons. Otherwise the type does not affect the output.

Value

A single number indicating the distance between the two input vectors.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

dist <- calcDistance(braingraphs[,1], braingraphs[,2], "adjMatrix")
dist
```

---

estGStar   *Estimate G-Star*

Description

This function estimates the g-star graph for a given set of graphs.

Usage

```
estGStar(data)
```

Arguments

data   A data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).

Value

A single vector that is the gstar is returned.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

braingstar <- estGStar(braingraphs)
braingstar[1:25]
```

estLogLik

Estimate the Log Likelihood Value

Description

This function estimates log likelihood value for a given graph.

Usage

```
estLogLik(data, type, gstar, tau, g = NULL)
```

Arguments

data	A data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
type	The type of graph being used (adjmatrix or adjmatrixlt).
gstar	A single vector to estimate the likelihood for.
tau	A single value used in estimating the likelihood.
g	Deprecated. Replaced with gstar for clarity.

Value

The log-likelihood value of the data.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

braingstar <- estGStar(braingraphs)
braintau <- estTau(braingraphs, "adjMatrix", braingstar)
brainll <- estLogLik(braingraphs, "adjMatrix", braingstar, braintau)
brainll
```

---

estMLE *Estimate the MLE Parameters*

Description

This function estimates the MLE parameters g-star and tau for a given set of graphs.

Usage

```
estMLE(data, type)
```

Arguments

data   A data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).

type   The type of graph being used (adjmatrix or adjmatrixlt).

Details

Essentially this function calls both `estGStar` and `estTau` and returns the results.

Value

A list containing g-star and tau named gstar and tau respectively.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

brainmle <- estMLE(braingraphs, "adjMatrix")
brainmle
```

---

estTau   *Estimate Tau*

Description

This function estimates tau for a given set of graphs.

Usage

```
estTau(data, type, gstar)
```

Arguments

data	A data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
type	The type of graph being used (adjmatrix or adjmatrixlt).
gstar	A single columned data frame to be used as the g-star of the data set.

Value

The tau value for the data based on g star.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

braingstar <- estGStar(braingraphs)
braintau <- estTau(braingraphs, "adjMatrix", braingstar)
braintau
```

---

genAlg   *Find Edges Separating Two Groups using Genetic Algorithm (GA)*

Description

GA-Mantel is a fully multivariate method that uses a genetic algorithm to search over possible edge subsets using the Mantel correlation as the scoring measure for assessing the quality of any given edge subset.

Usage

```
genAlg(data, covars, iters = 50, popSize = 200, earlyStop = 0,
dataDist = "manhattan", covarDist = "gower", verbose = FALSE,
plot = TRUE, minSolLen = NULL, maxSolLen = NULL)
```

Arguments

`data`	A matrix of edges(rows) for each sample(columns).
`covars`	A matrix of covariates(columns) for each sample(rows).
`iters`	The number of times to run through the GA.
`popSize`	The number of solutions to test on each iteration.
`earlyStop`	The number of consecutive iterations without finding a better solution before stopping regardless of the number of iterations remaining. A value of '0' will prevent early stopping.
`dataDist`	The distance metric to use for the data. This can only be "manhattan" for now.
`covarDist`	The distance metric to use for the covariates. Either "euclidean" or "gower".
`verbose`	While 'TRUE' the current status of the GA will be printed periodically.
`plot`	A boolean to plot the progress of the scoring statistics by iteration.
`minSolLen`	The minimum number of columns to select.
`maxSolLen`	The maximum number of columns to select.

Details

Use a GA approach to find edges that separate subjects based on group membership or set of covariates.

The data and covariates should be normalized BEFORE use with this function because of distance functions.

This function uses modified code from the rbga function in the genalg package. rbga

Because the GA looks at combinations and uses the raw data, edges with a small difference may be selected and large differences may not be.

The distance calculations use the vegdist package. vegdist

Value

A list containing

`scoreSumm`	A matrix summarizing the score of the population. This can be used to figure out if the ga has come to a final solution or not. This data is also plotted if plot is 'TRUE'.
`solutions`	The final set of solutions, sorted with the highest scoring first.
`scores`	The scores for the final set of solutions.
`time`	How long in seconds the ga took to run.
`selected`	The selected edges by name.
`nonSelected`	The edges that were NOT selected by name.
`selectedIndex`	The selected edges by row number.

Author(s)

Sharina Carter, Elena Deych, Berkley Shands, William D. Shannon

Examples

```
## Not run:
data(braingraphs)

### Set covars to just be group membership
covars <- matrix(c(rep(0, 19), rep(1, 19)))

### We use low numbers for speed. The exact numbers to use depend
### on the data being used, but generally the higher iters and popSize
### the longer it will take to run.  earlyStop is then used to stop the
### run early if the results aren't improving.
iters <- 500
popSize <- 200
earlyStop <- 250

gaRes <- genAlg(braingraphs, covars, iters, popSize, earlyStop)

## End(Not run)
```

---

genAlgConsensus	*Find Edges Separating Two Groups using Multiple Genetic Algorithm's (GA) Consensus*

Description

GA-Mantel is a fully multivariate method that uses a genetic algorithm to search over possible edge subsets using the Mantel correlation as the scoring measure for assessing the quality of any given edge subset.

Usage

```
genAlgConsensus(data, covars, consensus = .5, numRuns = 10,
parallel = FALSE, cores = 3, ...)
```

Arguments

data	A matrix of edges(rows) for each sample(columns).
covars	A matrix of covariates(columns) for each sample(rows).
consensus	The required fraction (0, 1] of solutions containing an edge in order to keep it.
numRuns	Number of runs to do. In practice the number of runs needed varies based on data set size and the GA parameters set.
parallel	When this is 'TRUE' it allows for parallel calculation of the bootstraps. Requires the package doParallel.
cores	The number of parallel processes to run if parallel is 'TRUE'.
...	Other arguments for the GA function see genAlg

Details

Use a GA consensus approach to find edges that separate subjects based on group membership or set of covariates if you cannot run the GA long enough to get a final solution.

Value

A list containing

solutions	The best solution from each run.
consSol	The consensus solution.
selectedIndex	The selected edges by row number.

Author(s)

Sharina Carter, Elena Deych, Berkley Shands, William D. Shannon

Examples

```
## Not run:
data(braingraphs)

### Set covars to just be group membership
covars <- matrix(c(rep(0, 19), rep(1, 19)))

### We use low numbers for speed. The exact numbers to use depend
### on the data being used, but generally the higher iters and popSize
### the longer it will take to run.  earlyStop is then used to stop the
### run early if the results aren't improving.
iters <- 500
popSize <- 200
earlyStop <- 250
numRuns <- 3

gaRes <- genAlgConsensus(braingraphs, covars, .5, numRuns, FALSE, 3,
iters, popSize, earlyStop)

## End(Not run)
```

---

getGibbsMixture *Group Splitter*

Description

This function splits the data into groups based on the Gibbs criteria.

Usage

```
getGibbsMixture(data, type, desiredGroups, maxIter = 50, digits = 3)
```

Arguments

`data`	A data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
`type`	The type of graph being used (adjmatrix or adjmatrixlt).
`desiredGroups`	The number of groups to test for.
`maxIter`	The maximum number of iterations to run searching for an optimal split.
`digits`	The number of digits to round internal values to when checking the stop criteria.

Details

Generally this function is not used by itself but in conjunction with getLoglikeMixture.

Value

A list that contains information about the group splits. The list contains the final weights, gstars and taus for every group, a boolean indicating convergence, the number of iterations it took, and the group for each graph.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

braingm <- getGibbsMixture(braingraphs, "adjMatrix", 5)
```

---

getLoglikeMixture   *Group Finder*

Description

This function takes group splits and determines the likelihood of those groups.

Usage

```
getLoglikeMixture(data, mixture, numConst)
```

Arguments

`data`	A data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
`mixture`	The output of the `getGibbsMixture` function.
`numConst`	The numeric constant to multiply the loglikihood by.

Value

A list containing the BIC criteria and the log likelihood named bic and ll respectively.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

braingm <- getGibbsMixture(braingraphs, "adjMatrix", 5)
brainlm <- getLoglikeMixture(braingraphs, braingm)
brainlm

### By running the loglik mixture over several groups you can find which is the optimal
## Not run:
mixtures <- NULL
for(i in 1:5){
tempgm <- getGibbsMixture(braingraphs, "adjMatrix", i)
mixtures[i] <- getLoglikeMixture(braingraphs, tempgm)$bic
}

bestgroupnum <- which(min(mixtures) == mixtures)
bestgroupnum

## End(Not run)
```

---

getNumEdges   *Get the Number of Edges in a Graph*

Description

This function will return the number of edges for a given of graph.

Usage

```
getNumEdges(nodes, type)
```

Arguments

| nodes | The number of individual nodes in a given graph. |
| type | The type of graph being used (adjmatrix or adjmatrixlt). |

Value

The number of edges between individual nodes in the given graph.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

brainnodes <- getNumNodes(braingraphs, "adjMatrix")
brainedges <- getNumEdges(brainnodes, "adjMatrix")
brainedges
```

---

getNumNodes	*Get the Number of Nodes in a Graph*

Description

This function will return the number of nodes for a given of graph.

Usage

```
getNumNodes(data, type)
```

Arguments

data	A data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
type	The type of graph being used (adjmatrix or adjmatrixlt).

Value

The number of individual nodes in the given graph.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

brainnodes <- getNumNodes(braingraphs, "adjMatrix")
brainnodes
```

---

glrtPvalue   *GLRT Regression Results*

Description

This function returns the p-value of the significance of b1 in the regression model.

Usage

```
glrtPvalue(dataList, type, groups, numPerms = 10, parallel = FALSE,
cores = 3, data = NULL)
```

Arguments

dataList	A list where each element is a data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
type	The type of graph being used (adjmatrix or adjmatrixlt).
groups	Deprecated. Each data set should be an element in dataList.
numPerms	Number of permutations. In practice this should be at least 1,000.
parallel	TRUE or FALSE depending on whether the analysis will be parallelized for speed.
cores	The number of cores to use for parallelization. Ignored if parallel = FALSE.
data	Deprecated. Replaced with dataList for clarity.

Value

A list containing the p-value, tau, logliklihood value, glrt value, bob1, b0, b1 and the hamming errors.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

### Break our data into two groups
dataList <- list(braingraphs[,1:19], braingraphs[,20:38])

### We use 1 for speed, should be at least 1,000
numPerms <- 1

res <- glrtPvalue(dataList, "adjMatrix", numPerms=numPerms)
res$pvalue
```

---

graphNetworkPlot *Graph Network Plots*

Description

This function plots the connections between nodes in a single subject.

Usage

```
graphNetworkPlot(data, type, main = "Network Plot", labels, groupCounts, groupLabels)
```

Arguments

data	A vector of a single graph.
type	The type of graph being used (adjmatrix or adjmatrixlt).
main	The title for the plot.
labels	A vector which contains the names for each node.
groupCounts	A vector which contains the number of nodes in each group of nodes.
groupLabels	A vector which contains the names for each group of nodes.

Value

A plot displaying the connections between the nodes.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

main <- "Brain Connections"
gc <- c(5, 5, 4, 6)
gl <- c("Grp1", "Grp2", "Grp3", "Grp4")

graphNetworkPlot(braingraphs[,1], "adjMatrix", main, groupCounts=gc, groupLabels=gl)
```

---

lrtPvalue   *Likelihood Ratio Test*

Description

This function returns the p-value of the significance between two groups.

Usage

```
lrtPvalue(dataList, type, groups, numPerms = 10, parallel = FALSE, cores = 3, data = NULL)
```

Arguments

dataList	A list where each element is a data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
type	The type of graph being used (adjmatrix or adjmatrixlt).
groups	Deprecated. Each data set should be an element in dataList.
numPerms	Number of permutations. In practice this should be at least 1,000.
parallel	TRUE or FALSE depending on whether the analysis will be parallelized for speed.
cores	The number of cores to use for parallelization. Ignored if parallel = FALSE.
data	Deprecated. Replaced with dataList for clarity.

Value

The p-value for the difference between the two groups being tested.

Author(s)

Berkley Shands, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

### Break our data into two groups
dataList <- list(braingraphs[,1:19], braingraphs[,20:38])

### We use 1 for speed, should be at least 1,000
numPerms <- 1

lrt <- lrtPvalue(dataList, "adjMatrix", numPerms=numPerms)
lrt
```

---

pairedPvalue   *P-Value for Paired Data Results*

Description

This function returns the p-value of the significance of the difference in g-star values for paired data.

Usage

```
pairedPvalue(dataList, type, groups, numPerms = 10, parallel = FALSE,
cores = 3, data = NULL)
```

Arguments

dataList	A list where each element is a data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
type	The type of graph being used (adjmatrix or adjmatrixlt).
groups	Deprecated. Each data set should be an element in dataList.
numPerms	Number of permutations. In practice this should be at least 1,000.
parallel	TRUE or FALSE depending on whether the analysis will be parallelized for speed.
cores	The number of cores to use for parallelization. Ignored if parallel = FALSE.
data	Deprecated. Replaced with dataList for clarity.

Value

The p-value for the difference between the two groups being tested.

Author(s)

Berkley Shands, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

### Break our data into two groups
dataList <- list(braingraphs[,1:19], braingraphs[,20:38])

### We use 1 for speed, should be at least 1,000
numPerms <- 1

pval <- pairedPvalue(dataList, "adjMatrix", numPerms=numPerms)
pval
```

---

plotHeatmap *Plot Heat Map*

Description

This function plots the connections between nodes in a single subject as a heat map.

Usage

```
plotHeatmap(data, type, names, ...)
```

Arguments

data	A vector of a single graph.
type	The type of graph being used (adjmatrix or adjmatrixlt).
names	A vector of names for labeling the nodes on the plot.
...	Arguments to be passed to the plot method.

Value

A plot displaying the connections between the nodes as a heat map.

Author(s)

Berkley Shands, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

braingstar <- estGStar(braingraphs)
plotHeatmap(braingstar, "adjMatrix")
```

---

plotMDS *Plot MDS*

Description

This function plots all the data on an MDS plot.

Usage

```
plotMDS(dataList, groups, estGstar = TRUE, paired = FALSE,
returnCoords = FALSE, ..., data=NULL)
```

Arguments

`dataList`	A list where each element is a data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
`groups`	Deprecated. Each data set should be an element in dataList.
`estGstar`	When TRUE, the g star for every group is calculated and plotted.
`paired`	When TRUE, line segments between pairs will be drawn.
`returnCoords`	When TRUE, the MDS x-y coordinates will be returned.
`...`	Arguments to be passed to the plot method.
`data`	Deprecated. Replaced with dataList for clarity.

Value

An MDS plot and if returnCoords is TRUE, a 2 column data frame containing the x-y coordinates of the data points is also returned.

Author(s)

Berkley Shands, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

### Break our data into two groups
dataList <- list(braingraphs[,1:19], braingraphs[,20:38])

### Basic plot
plotMDS(dataList, main="MDS Plot")

### Paired Plot
plotMDS(dataList, paired=TRUE, main="Paired MDS Plot")
```

---

rGibbs	*Generate Random Data*

Description

Generate random data sampled from the Gibbs distribution.

Usage

```
rGibbs(gstar, tau, type, numGraphs = 1)
```

Arguments

gstar	G star vector.
tau	A single value that affects the dispersion of the generated data.
type	The type of graph being used (adjmatrix or adjmatrixlt).
numGraphs	The number of graphs to generate.

Value

A data frame containing all the graphs generated.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

braingstar <- estGStar(braingraphs)
braintau <- estTau(braingraphs, "adjMatrix", braingstar)
randombraingraphs <- rGibbs(braingstar, braintau, "adjMatrix", 3)
randombraingraphs[1:5,]
```

---

testGoF   *Test the Goodness of Fit*

Description

This function tests the goodness of fit for given a set of graphs.

Usage

```
testGoF(data, type, numSims = 10, plot = TRUE, main)
```

Arguments

data	A data frame in which the columns (subjects) contain a 0/1 value for row (Node or Edge).
type	The type of graph being used (adjmatrix or adjmatrixlt).
numSims	Number of simulations for Monte Carlo estimation of p-value(ideally, 1000 or more). Ignored if Chi-Square method is used.
plot	A boolean to create a plot of the results or not.
main	A title for the plot.

Value

A list containing information about the goodness of fit and potentially a plot. The list contains the Pearson statistics, degrees of freedom, and p-value, the G statistics and p-value, the Chi Squared statistics and p-value and finally the table with the observed and expected counts.

Author(s)

Terrence Brooks, Berkley Shands, Skye Buckner-Petty, Patricio S. La Rosa, Elena Deych, William D. Shannon

Examples

```
data(braingraphs)

numSims <- 1 ### This is set low for speed
braingof <- testGoF(braingraphs, "adjMatrix", numSims)
```

Index