dRWRcontact

January 19, 2018

`dRWRcontact`	*Function to estimate RWR-based contact strength between samples from an input gene-sample data matrix, an input graph and its pre-computed affinity matrix*

Description

`dRWRcontact` is supposed to estimate sample relationships (ie. contact strength between samples) from an input gene-sample matrix, an input graph and its affinity matrix pre-computed according to random walk restart (RWR) of the input graph. It includes: 1) RWR-smoothed columns of input gene-sample matrix based on the pre-computed affinity matrix; 2) calculation of contact strength (inner products of RWR-smooth columns of input gene-sample matrix); 3) estimation of the contact signficance by a randomalisation procedure. Parallel computing is also supported for Linux or Mac operating systems.

Usage

```
dRWRcontact(data, g, Amatrix, permutation = c("random", "degree"),
num.permutation = 10, p.adjust.method = c("BH", "BY", "bonferroni",
"holm", "hochberg", "hommel"), adjp.cutoff = 0.05, parallel = TRUE,
multicores = NULL, verbose = T)
```

Arguments

`data`	an input gene-sample data matrix used for seeds. Each value in input gene-sample matrix does not necessarily have to be binary (non-zeros will be used as a weight, but should be non-negative for easy interpretation).
`g`	an object of class "igraph" or "graphNEL"
`Amatrix`	an affinity matrix pre-computed from the input graph. Notes: columns for starting nodes walking from, and rows for ending nodes walking to
`permutation`	how to do permutation. It can be 'degree' for degree-preserving permutation, 'random' for permutation purely in random
`num.permutation`	the number of permutations used to for generating the distribution of contact strength under randomalisation

`p.adjust.method`	the method used to adjust p-values. It can be one of "BH", "BY", "bonferroni", "holm", "hochberg" and "hommel". The first two methods "BH" (widely used) and "BY" control the false discovery rate (FDR: the expected proportion of false discoveries amongst the rejected hypotheses); the last four methods "bonferroni", "holm", "hochberg" and "hommel" are designed to give strong control of the family-wise error rate (FWER). Notes: FDR is a less stringent condition than FWER
`adjp.cutoff`	the cutoff of adjusted pvalue to construct the contact graph
`parallel`	logical to indicate whether parallel computation with multicores is used. By default, it sets to true, but not necessarily does so. It will depend on whether these two packages "foreach" and "doParallel" have been installed. It can be installed via: `source("http://bioconductor.org/biocLite.R"); biocLite(c("foreach","doParalle` If not yet installed, this option will be disabled
`multicores`	an integer to specify how many cores will be registered as the multicore parallel backend to the 'foreach' package. If NULL, it will use a half of cores available in a user's computer. This option only works when parallel computation is enabled
`verbose`	logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

Value

an object of class "dContact", a list with following components:

- `ratio`: a symmetric matrix storing ratio (the observed against the expected) between pairwise samples
- `zscore`: a symmetric matrix storing zscore between pairwise samples
- `pval`: a symmetric matrix storing pvalue between pairwise samples
- `adjpval`: a symmetric matrix storing adjusted pvalue between pairwise samples
- `cgraph`: the constructed contact graph (as a 'igraph' object) under the cutoff of adjusted value
- `call`: the call that produced this result

Note

none

See Also

`dRWR`, `dCheckParallel`

Examples

```
# 1) generate a random graph according to the ER model
g <- erdos.renyi.game(100, 1/100)

# 2) produce the induced subgraph only based on the nodes in query
subg <- dNetInduce(g, V(g), knn=0)
V(subg)$name <- 1:vcount(subg)

# 3) pre-compute affinity matrix from the input graph
Amatrix <- dRWR(g=subg, parallel=FALSE)
```

```
# 4) estimate RWR-based sample relationships
# define sets of seeds as data
# each seed with equal weight (i.e. all non-zero entries are '1')
aSeeds <- c(1,0,1,0,1)
bSeeds <- c(0,0,1,0,1)
data <- data.frame(aSeeds,bSeeds)
rownames(data) <- 1:5
# calcualte their two contacts
dContact <- dRWRcontact(data=data, g=subg, Amatrix=Amatrix,
parallel=FALSE)
dContact
```