Package ‘MODA’

December 21, 2016

**Type** Package

**Title** MODA: MOdule Differential Analysis for weighted gene co-expression network

**Version** 1.0.0

**Date** 2016-06-29

**Author** Dong Li, James B. Brown, Luisa Orsini, Zhisong Pan, Guyu Hu and Shan He

**Maintainer** Dong Li <dxl466@cs.bham.ac.uk>

**Description** MODA can be used to estimate and construct condition-specific gene co-expression networks, and identify differentially expressed subnetworks as conserved or condition specific modules which are potentially associated with relevant biological processes.

**License** GPL (>= 2)

**Depends** R (>= 3.1.0)

**Imports** WGCNA,dynamicTreeCut,igraph

**RoxygenNote** 5.0.1

**biocViews** GeneExpression, Microarray, DifferentialExpression, Network

**Suggests** BiocStyle, knitr

**VignetteBuilder** knitr

**NeedsCompilation** no

**R topics documented:**

- `CompareAllNets` ................................................................. 2
- `comparemodulestwonets` ................................................... 3
- `datExpr1` ........................................................................... 4
- `datExpr2` ........................................................................... 4
- `PartitionDensity` .............................................................. 5
- `PartitionModularity` ......................................................... 5
- `WeightedModulePartitionDensity` ...................................... 6

**Index** 8
CompareAllNets

Illustration of network comparison

Description

Compare the background network and a set of condition-specific network. Conserved or condition-specific modules are indicated by the plain files, based on the statistics

Usage

CompareAllNets(ResultFolder, intModules, speciesName, intConditionModules, conditionNames, specificTheta, conservedTheta)

Arguments

- `ResultFolder`: where to store results
- `intModules`: how many modules in the background network
- `speciesName`: identifier of current profile, served as a tag in name
- `intConditionModules`: a numeric vector, each of them is the number of modules in each condition-specific network. Or just single number
- `conditionNames`: a character vector, each of them is the name of condition. Or just single name
- `specificTheta`: the threshold to define min(s)+specificTheta, less than which is considered as condition specific module. s is the sums of rows in Jaccard index matrix. See supplementary file.
- `conservedTheta`: The threshold to define max(s)-conservedTheta, greater than which is considered as condition conserved module. s is the sums of rows in Jaccard index matrix. See supplementary file.

Value

None

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

See Also

WeightedModulePartitionDensity, comparemodulestwonets

Examples

data(synthetic)
ResultFolder = 'ForSynthetic' # where middle files are stored
CuttingCriterion = 'Density' # could be Density or Modularity
indicator1 = 'X' # indicator for data profile 1
indicator2 = 'Y' # indicator for data profile 2
specificTheta = 0.1 # threshold to define condition specific modules
conservedTheta = 0.1 # threshold to define conserved modules
intModules1 <- WeightedModulePartitionDensity(datExpr1,ResultFolder,
comparemodulestonets

indicator1,CuttingCriterion)
intModules2 <- WeightedModulePartitionDensity(datExpr2,ResultFolder,
indicator2,CuttingCriterion)
CompareAllNets(ResultFolder,intModules1,indicator1,intModules2,indicator2,
specificTheta,conservedTheta)

Description

Compare the background network and a condition-specific network. A Jaccard index is used to
measure the similarity of two sets, which represents the similarity of each module pairs from two
networks.

Usage

comparemodulestonets(sourcehead, nm1, nm2, ind1, ind2)

Arguments

sourcehead prefix of where to store results
nm1 how many modules in the background network
nm2 how many modules in the condition-specific network
ind1 indicator of the background network
ind2 indicator of the condition-specific network

Value

A matrix where each entry is the Jaccard index of corresponding modules from two networks

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

Examples

data(synthetic)
ResultFolder = 'ForSynthetic' # where middle files are stored
CuttingCriterion = 'Density' # could be Density or Modularity
indicator1 = 'X' # indicator for data profile 1
indicator2 = 'Y' # indicator for data profile 2
intModules1 <- WeightedModulePartitionDensity(datExpr1,ResultFolder,
indicator1,CuttingCriterion)
intModules2 <- WeightedModulePartitionDensity(datExpr2,ResultFolder,
indicator2,CuttingCriterion)
JaccardMatrix <- comparemodulestonets(ResultFolder,intModules1,intModules2,
paste('/DenseModuleGene_\',indicator1,sep=''),
paste('/DenseModuleGene_\',indicator2,sep=''))
Description

Synthetic gene expression profile with 20 samples and 500 genes.

Format

A matrix with 20 rows and 500 columns.

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

Examples

data(synthetic)
## plot the heatmap of the correlation matrix ...
## Not run: heatmap(cor(as.matrix(datExpr1)))

Description

Synthetic gene expression profile with 25 samples and 500 genes.

Format

A matrix with 25 rows and 500 columns.

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

Examples

data(synthetic)
## plot the heatmap of the correlation matrix ...
## Not run: heatmap(cor(as.matrix(datExpr2))}


**PartitionDensity**

**Illustration of partition density**

**Description**

Calculate the average density of all resulting modules from a partition. The density of each module is defined as the average adjacency of the module genes.

**Usage**

\[ \text{PartitionDensity}(\text{ADJ}, \text{PartitionSet}) \]

**Arguments**

- **ADJ**: gene similarity matrix
- **PartitionSet**: vector indicates the partition label for genes

**Value**

partition density, defined as average density of all modules

**Author(s)**

Dong Li, <dxl466@cs.bham.ac.uk>

**References**


**Examples**

```r
data(synthetic)
ADJ1 = abs(cor(datExpr1, use="p")) ^ 10
dissADJ1 = 1 - ADJ1
hierADJ1 = hclust(as.dist(dissADJ1), method="average")
groups <- cutree(hierADJ1, h = 0.8)
pDensity <- PartitionDensity(ADJ1, groups)
```

---

**PartitionModularity**

**Illustration of modularity density**

**Description**

Calculate the average modularity of a partition. The modularity of each module is defined from a natural generalization of unweighted case.

**Usage**

\[ \text{PartitionModularity}(\text{ADJ}, \text{PartitionSet}) \]

**Examples**

```r
data(synthetic)
ADJ1 = abs(cor(datExpr1, use="p")) ^ 10
dissADJ1 = 1 - ADJ1
hierADJ1 = hclust(as.dist(dissADJ1), method="average")
groups <- cutree(hierADJ1, h = 0.8)
pDensity <- PartitionDensity(ADJ1, groups)
```
WeightedModulePartitionDensity

Arguments

**ADJ**  
gene similarity matrix

**PartitionSet**  
vector indicates the partition label for genes

Value

partition modularity, defined as average modularity of all modules

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References


Examples

data(synthetic)  
ADJ1=abs(cor(datExpr1,use="p"){power}10  
dissADJ=1-ADJ1  
hierADJ=hclust(as.dist(dissADJ), method="average")  
groups <- cutree(hierADJ, h = 0.8)  
pDensity <- PartitionModularity(ADJ1,groups)

---

**WeightedModulePartitionDensity**

*Illustration of Modules detection*

Description

Module detection based on the optimal cutting height of dendrogram, which is selected to make the average density or modularity of resulting partition maximal. The clustering and visualization function are from WGCNA.

Usage

WeightedModulePartitionDensity(datExpr, foldername, indicatename,  
cutmethod = c("Density", "Modularity"), power = 10)

Arguments

**datExpr**  
gene expression profile, rows are samples and columns genes

**foldername**  
where to store the clusters

**indicatename**  
normally a specific tag of condition

**cutmethod**  
cutting the dendrogram based on maximal average Density or Modularity

**power**  
the power parameter of WGCNA, $W_{ij} = |\text{cor}(x_i, x_j)|^{\text{power}}$
WeightedModulePartitionDensity

Value

The number of clusters

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References


See Also

PartitionDensity
PartitionModularity

Examples

data(synthetic)
ResultFolder = 'ForSynthetic' # where middle files are stored
CuttingCriterion = 'Density' # could be Density or Modularity
indicator1 = 'X' # indicator for data profile 1
indicator2 = 'Y' # indicator for data profile 2
specificTheta = 0.1 #threshold to define condition specific modules
conservedTheta = 0.1#threshold to define conserved modules
intModules1 <- WeightedModulePartitionDensity(datExpr1,ResultFolder, indicator1,CuttingCriterion)
Index

*Topic comparison
  comparemodulestwonets, 3

*Topic cutting
  WeightedModulePartitionDensity, 6

*Topic data
  datExpr1, 4
  datExpr2, 4

*Topic dendrogram
  WeightedModulePartitionDensity, 6

*Topic density
  PartitionDensity, 5

*Topic differential
  CompareAllNets, 2

*Topic modularity
  PartitionModularity, 5

*Topic module
  CompareAllNets, 2
  comparemodulestwonets, 3
  CompareAllNets, 2
  comparemodulestwonets, 2, 3
  datExpr1, 4
  datExpr2, 4
  PartitionDensity, 5, 7
  PartitionModularity, 5, 7
  WeightedModulePartitionDensity, 2, 6