Package 'iBBiG'

October 17, 2024

Type Package

Title Iterative Binary Biclustering of Genesets

Version 1.48.0

Date 2011-11-23

Author Daniel Gusenleitner, Aedin Culhane

Maintainer Aedin Culhane <aedin@jimmy.harvard.edu>

Depends biclust

Imports stats4,xtable,ade4

Suggests methods

Description iBBiG is a bi-clustering algorithm which is optimizes for binary data analysis. We apply it to meta-gene set analysis of large numbers of gene expression datasets. The iterative algorithm extracts groups of phenotypes from multiple studies that are associated with similar gene sets. iBBiG does not require prior knowledge of the number or scale of clusters and allows discovery of clusters with diverse sizes

Reference Gusenleitner D, Howe EA, Bentink S, Quackenbush J, Culhane AC. iBBiG: Iterative Binary Bi-clustering of Gene Sets. Bioinformatics. In Press

License Artistic-2.0

URL http://bcb.dfci.harvard.edu/~aedin/publications/

biocViews Clustering, Annotation, GeneSetEnrichment

git_url https://git.bioconductor.org/packages/iBBiG

git_branch RELEASE_3_19

git_last_commit f63ad3c

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-10-16

Contents

`iBBiG-package` *iBBiG performs bi-clustering of binary matrices*

Description

iBBiG is a bi-clustering algorithm, optimized for module discovery in sparse noisy binary genomics data. We designed iBBiG to have high specificity and thereby minimize the false positive rate when discovering new classes; the iterative approach employed in iBBiG is able to discover weak signals, even if they are potentially masked by stronger ones.

Details

Package:	iBBiG
Type:	Package
Version:	0.99.1
Date:	2012-03-15
License:	Free Artistic
LazyLoad:	yes
Depends:	methods

The main functions is iBBiG. This is the biclustering algorithm.

Author(s)

Aedin Culhane, Daniel Gusenleitner

Maintainer: Aedin <aedin@jimmy.harvard.edu>

References

Daniel Gusenleitner, Eleanor A Howe, Stefan Bentink, John Quackenbush and Aedin C Culhane iBBiG: Iterative Binary Bi-clustering of Gene Sets Bioinformatics. In review.

See Also

Also see `biclust` ~~

Examples

```
#create simulated datasets
binMat<-makeArtificial()
binMat
plot(binMat)
res<- try(iBBiG(binMat@Seeddata, nModules=10))
plot(res)
res

## Subset a cluster

res[4]
res[1:2]


## As iBBiG extends the class Biclust can use Biclust functions on it
## View the rows and columns of an iBBiG object

## Create a list of matrices, one for each cluster
Modules<-bicluster(res@Seeddata, res)
length(Modules)
lapply(Modules, dim)


# Or extract a list of a specific cluster
M1<-bicluster(res@Seeddata, res, 1)
dim(M1[[1]])
str(M1)
M1[[1]][1:5,1:3]
```

`iBBiG` *Iterative Binary Bi-Clustering for GeneSets*

Description

iBBiG is a bi-clustering algorithm which is optimized for clustering binary data resulting from discretized p-values of genomic analyses

Usage

```
iBBiG(binaryMatrix, nModules, alpha = 0.3, pop_size = 100, mutation = 0.08, stagnation = 50, selection_p
```

Arguments

`binaryMatrix`	Matrix. A binary or logical matrix.
`nModules`	Numeric. The number of expected modules. As iBBiG is optimized to find a miminal number, nModules can be a larger than expected value

alpha	Numeric, weighting factor, that will balances the tradeoff between specificity and sensitivity. Default 0.3. Simulated studies indicate range 0.3-0.5 is appropriate
pop_size	Numeric. Default 100. Population size establishes the genetic diversity of solutions in Genetic Algorithm. Simulated studies show that it has marginal effect on performance.
mutation	Numeric. Default 0.08. Mutation rate of GA. Simulated studies show that it has little effect on performance.
stagnation	Numeric. Default is stop criterion of 50 iterations of stagnation. Simulated studies show that it has little effect on performance.
selection_pressure	Numeric. Default is 1.2. Selection pressure for parent selection. Simulated studies show that it has little effect on performance
max_sp	Numeric. Default is 15. Simulated studies show that it has little effect on performance
success_ratio	Numeric. Deafult 0.6. Success ratio determines how many children have to outperform at least one of their parents. Simulated studies show that it has little effect on performance

Details

iBBiG is a bi-clustering algorithm, optimized for module discovery in sparse noisy binary genomics data. We designed iBBiG to have high specificity and thereby minimize the false positive rate when discovering new classes; the iterative approach employed in iBBiG is able to discover weak signals, even if they are potentially masked by stronger ones. For a compairions with global clustering approaches (K-means, hierarchical cluster analysis) and bi-clustering approaches (Bimax, FABIA, COALESCE) see our manuscript Gusenleitner et al., 2012. An advantage of iBBiG relative to other methods is that it does not require a priori knowledge of the true number of clusters. Following the application of iBBiG, the number of true clusters can be estimated from the weighted cluster scores and RowScorexNumber of the extracted modules. In some cases, we observed that a module may represent the residue or remaining signal of a stronger, previously extracted module. This residue remains because iBBiG only removes information from the data matrix that is actually used for the entropy based score in a module. However, we do not consider these residual modules to be a shortcoming of the method as their existence facilitates discovery of the true overlap between modules and, further, these modules can be easily detected by looking at the overlap of clinical covariates and gene sets.\ Although iBBiG includes several parameters, we have shown that most impact only computation time, and do not effect cluster discovery. The only parameter that had an impact on cluster discovery was alpha, which is a weighting factor that balances the cost of increasing cluster size (number of rows) against cluster homogeneity. In generating small homogeneous clusters, one might miss information. Conversely, large hetergeneous clusters may contain more false positives. Although alpha does not regulate the number of clusters, decreasing stringency, by increasing alpha values may produce greater numbers of clusters. As a results the alpha parameter is useful in adjusting the sensitivity-specificity ratio. Alpha has a range 0.1-1 where 0.1 will generate fewer, smaller homogeneous clusters whereas 0.9 is less stringent and results in more hetergeneous clusters (with greater potential for false positives). Increasing alpha will generate more clusters of greater size, with potentially greater specificity at the expense of decreased sensitivity. Following tests on simulated data we recommended alpha values between 0.3-0.5 (Gusenleitner et al., 2012). The default alpha is 0.3

Value

Returns an object with class iBBiG, which extents the class Biclust.

`Seeddata`	Input binaryMatrix
`RowScorexNumber`	Matrix. Score for each signature (row) in each cluster. Matrix with dimensions, Number of Rows in Seeddata x Number of clusters
`Clusterscores`	Vector. Score for each cluster. It has length equal to the number of clusters.
`Parameters`	List of Input Parameters (if provided)
`RowxNumber`	Binary or Logical Matrix with dimensions, Number of Rows in Seeddata x Number of clusters, where 1 represents cluster membership
`NumberxCol`	Binary or Logical Matrix with dimensions, Number of clusters x Number of Columns in Seeddata ,where 1 represents cluster membership
`Number`	Numeric. Number of modules(clusters)
`info`	list. which is a general contained for other information.

Author(s)

Aedin Culhane, Daniel Gusenleitner

References

Daniel Gusenleitner, Eleanor A Howe, Stefan Bentink, John Quackenbush and Aedin C Culhane iBBiG: Iterative Binary Bi-clustering of Gene Sets Bioinformatics. In review.

See Also

Further functions for viewing and clustering binaray data are available in the package biclust. We have written iBBiG and its classes so that it is compatible with biclust, and the class iBBiG inherits Biclust-class.

Examples

```
binMat<-makeArtificial()
plot(binMat)
res<- iBBiG(binMat@Seeddata, nModules=10)
plot(res)
res
analyzeClust(res,binMat)
```

`iBBiG-class` *Class* `"iBBiG"`

Description

Class to contain and describe result of iBBiG Anlaysis

Objects from the Class

Objects can be created by calls of the form `new("iBBiG", ...)`.

Slots

`Seeddata`**:** Input binaryMatrix

`RowScorexNumber`**:** Matrix. Score for each signature (row) in each cluster. Matrix with dimensions, Number of Rows in Seeddata x Number of clusters

`Clusterscores`**:** Vector. Score for each cluster. It has length equal to the number of clusters

`Parameters`**:** List of Input Parameters (if provided)

`RowxNumber`**:** Binary or Logical Matrix with dimensions, Number of Rows in Seeddata x Number of clusters, where 1 represents cluster membership

`NumberxCol`**:** Binary or Logical Matrix with dimensions, Number of clusters x Number of Columns in Seeddata ,where 1 represents cluster membership

`Number`**:** Numeric. Number of modules(clusters)

`info`**:** list. which is a general contained for other information.

Extends

Class `"Biclust"`, directly.

Methods

RowScorexNumber `signature(x = "iBBiG")`: Returns the row scores fore each cluster.

Clusterscores `signature(x = "iBBiG")`: Returns the overall score for each cluster.

Seeddata `signature(x = "iBBiG")`: Returns the original binary matrix, the clustering is based on.

Parameters `signature(x = "iBBiG")`: Returns parameter sets, inhereted from biclust.

RowxNumber `signature(x = "iBBiG")`: Returns a logical matrix indicating, which rows are included in each bicluster.

NumberxCol `signature(x = "iBBiG")`: Returns a logical matrix indicating, which columns are included in each bicluster.

Number `signature(x = "iBBiG")`: Returns the number of biclusters contained in the iBBiG object.

info `signature(x = "iBBiG")`: Returns additional information on the particular iBBiG object, inhereted from biclust.

plot `signature(x = "iBBiG")`: Plot the iBBiG clustering.

show `signature(object = "iBBiG")`: Shows the Biclusters.

summary `signature(object = "iBBiG")`: Summary of found bi-clusters.

[`signature(object = "iBBiG")`: ...

JIdist `signature(object = "iBBiG")`: ...

analyzeClust `signature(object = "iBBiG")`: ...

Author(s)

Aedin Culhane, Daniel Gusenleitner

References

Daniel Gusenleitner, Eleanor A Howe, Stefan Bentink, John Quackenbush and Aedin C Culhane iBBiG: Iterative Binary Bi-clustering of Gene Sets Bioinformatics. In review.

See Also

Further functions for viewing and clustering binary data are available in the package biclust. We have written iBBiG and its classes so that it is compatible with biclust, and the class iBBiG inherits Biclust-class.

Examples

```
showClass("iBBiG")


#create simulated datasets
binMat<-makeArtificial()
binMat

## Create a binary matrix of 400 rows v 400 cols
## Its created as a Biclust object, so its easier to visualize
plot(binMat)

## Perform biclustering analysis on the binary matrix
res<- iBBiG(binMat@Seeddata, nModules=8)
res
plot(res)

## Compare 2 iBBiG or Biclust results
analyzeClust(res, binMat)

## Subset a cluster

res[4]
res[1:2]
```

```
## As iBBiG extends the class Biclust can use Biclust functions on it
## View the rows and columns of an iBBiG object

## Create a list of matrices, one for each cluster
Modules<-bicluster(res@Seeddata, res)
length(Modules)
lapply(Modules, dim)

# Or extract a list of a specific cluster
M1<-bicluster(res@Seeddata, res, 1)
dim(M1[[1]])
str(M1)
M1[[1]][1:5,1:3]
```

`makeArtificial` *Create a 400x400 simulated binary matrix for testing iBBiG and other binary biclustering methods*

Description

Create a binary matrix of 400 rows x 400 columns, where 1 is a positive association. This matrix is seeded with 7 modules of various size and with various levels of noise as described by Gusenleitner et al.,

Usage

```
makeArtificial(nRow = 400, nCol = 400, noise = 0.1, verbose = TRUE, dM = makeSimDesignMat(verbose = verbo
```

Arguments

`nRow`	Numeric `nRow` number of rows
`nCol`	Numeric `nRow` number of columns
`noise`	Numeric. Value between 0-1. Default is 10 percent random noise (1) introduced into the spare binary matrix
`verbose`	Verbose output. Default is TRUE
`dM`	A design matrix specifying where the columns are. The function makeSimDesignMat create the matrix which specifies the design matrix
`seed`	Integer, passed to function set.seed() the random-number generator function, so that the articical simulated data is reproduced. If you wish to generate a random simulated data set use seed=NULL

Details

See Guesnleitner et al, for more information

Value

Output is a class of Biclust.

Author(s)

Aedin Culhane, Daniel Gusenleitner

References

Daniel Gusenleitner, Eleanor A Howe, Stefan Bentink, John Quackenbush and Aedin C Culhane iBBiG: Iterative Binary Bi-clustering of Gene Sets Bioinformatics. In review.

See Also

Further functions for viewing and clustering binaray data are available in the package biclust. We have written iBBiG and its classes so that it is compatible with biclust, and the class iBBiG inherits Biclust-class.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
arti<-makeArtificial()
plot(arti)
```

Index