Package 'QUBIC'

September 23, 2026

Type Package

Title An R Package for Qualitative Biclustering in Support of Gene Co-Expression Analyses

Description The core function of this R package is to provide the implementation of the well-cited and well-reviewed QUBIC algorithm, aiming to deliver an effective and efficient biclustering capability. This package also includes the following related functions: (i) a qualitative representation of the input gene expression data, through a well-designed discretization way considering the underlying data property, which can be directly used in other biclustering programs; (ii) visualization of identified biclusters using heatmap in support of overall expression pattern analysis; (iii) bicluster-based co-expression network elucidation and visualization, where different correlation coefficient scores between a pair of genes are provided; and (iv) a generalize output format of biclusters and corresponding network can be freely downloaded so that a user can easily do following comprehensive functional enrichment analysis (e.g. DAVID) and advanced network visualization (e.g. Cytoscape).

VignetteBuilder knitr

biocViews StatisticalMethod, Microarray, DifferentialExpression, MultipleComparison, Clustering, Visualization, GeneExpression, Network

Version 1.40.0

License CC BY-NC-ND 4.0 + file LICENSE

Encoding UTF-8

Depends R (>= 4.5.0)

Imports Rcpp (>= 0.11.0), methods, Matrix

LinkingTo Rcpp, RcppArmadillo

Suggests QUBICdata, qgraph, fields, knitr, rmarkdown

SystemRequirements C++11, Rtools (>= 3.1)

Enhances RColorBrewer

URL https://github.com/zy26/QUBIC

BugReports https://github.com/zy26/QUBIC/issues

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

git_branch RELEASE_3_23

git_last_commit 6627840

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-09-22

Author Yu Zhang [aut, cre],
Qin Ma [aut]

Maintainer Yu Zhang <zy26@jlu.edu.cn>

Contents

`BCQU-class` *Class BCQU.*

Description

Class `BCQU` define a QUalitative BIClustering calcuator.

Value

Class generator for objects of class `BCQU`.

See Also

`BCQU` `qudiscretize` `qunetwork` `qunet2xml`

QUBIC *QUBIC: A Qualitative Biclustering Algorithm for Analyses of Gene Expression Data*

Description

QUBIC is a biclustering package, with source code upgrading from C code to C++ code. The updated source code can avoid memory allocation error and is much efficient than the original one. Based on our preliminary analysis, it can save 40% running time on a plant microarray data. Whenever using this package, please cite as Yu Zhang, Juan Xie, Jinyu Yang, Anne Fennell, Chi Zhang, Qin Ma; QUBIC: a bioconductor package for qualitative biclustering analysis of gene co-expression data. *Bioinformatics*, 2017; 33 (3): 450-452. doi: 10.1093/bioinformatics/btw635

BCQUD performs a QUalitative BIClustering for a discret matrix.

Usage

```
BCQU(x = NULL,
        r = 1, q = 0.06,
        c = 0.95, o = 100, f = 1,
        k = max(ncol(x) %/% 20, 2),
        type = 'default', P = FALSE, C = FALSE, verbose = TRUE,
        weight = NULL, seedbicluster = NULL)

BCQUD(x = NULL,
        c = 0.95, o = 100, f = 1,
        k = max(ncol(x) %/% 20, 2),
        type = 'default', P = FALSE, C = FALSE, verbose = TRUE,
        weight = NULL, seedbicluster = NULL)

qubiclust_d(x, c = 0.95, o = 100, f = 1,
        k = max(ncol(x) %/% 20, 2),
        type = 'default', P = FALSE, C = FALSE, verbose = TRUE,
        weight = NULL, seedbicluster = NULL)

qubiclust(x, r = 1L, q = 0.06, c = 0.95, o = 100, f = 1,
        k = max(ncol(x) %/% 20, 2),
        type = 'default', P = FALSE, C = FALSE, verbose = TRUE,
        weight = NULL, seedbicluster = NULL)
```

Arguments

x	the input data matrix, which could be the normalized gene expression matrix or its qualitative representation from Qdiscretization or other discretization ways. (for example: a qualitative representation of gene expression data) For BCQU(), the data matrix should be real For BCQUD(), the data matrix should be discretized as integer. Zeros in the matrix will be treated as non-relevant value.

r	Affect the granularity of the biclusters. The range of possible ranks. A user can start with a small value of r (the default value is 1 so the corresponding data matrix consists of values '1', '-1' and '0'), evaluate the results, and then use larger values (should not be larger than half of the number of the columns) to look for fine structures within the identified biclusters.
q	Affect the granularity of the biclusters. The percentage of the regulating conditions for each gene. The choice of q's value depends on the specific application goals; that is if the goal is to find genes that are responsive to local regulators, we should use a relatively small q-value; otherwise we may want to consider larger q-values. The default value of q is 0.06 in QUBIC (this value is selected based on the optimal biclustering results on simulated data).
c	The required consistency level of a bicluster. The default value of c is 0.95
o	The number of output biclusters. o's default value is 100.
f	Control parameter, to control the level of overlaps between to-be-identified biclusters. The filter cut-off for data post-processing. For overlaps among to-be-identified biclusters. Its default value is set to 1 to ensure that no two reported biclusters overlap more than f.
k	The minimum column width of the block, minimum max(ncol(x) %/% 20, 2) columns.
type	The constrain type. If type is omitted or type='default', the original objective function in QUBIC will be used, which is to maximize the minimal value of numbers of rows and columns. If type='area', the program tries to identify the bicluster with the maximal value of number of rows multiplied by number of columns. Other types are reserved for future use.
P	The flag to enlarge current bicluster using a p-value contrain, which is defined based on its significance of expression consistency comparing to some simulated submatrix. Default: FALSE.
C	The flag to set the lower bound of the condition number in a bicluster as 5% of the total condition number in the input data. Only suggested to use when the input data has a few conditions (e.g. less than 20). Default: FALSE.
verbose	If 'TRUE', prints extra information on progress.
weight	Alternative weight matrix provided by user, will append to default weight. o, f, k, P, type, C will be ignored if using this parameter.
seedbicluster	Seed provided by user, normally should be a result of qubiclust.

Details

For a given representing matrix of a microarray data set, we construct a weighted graph G with genes represented as vertices, edges connecting every pair of genes, and the weight of each edge being the similarity level between the two corresponding (entire) rows. Clearly, the higher a weight, the more similar two corresponding rows are. Intuitively, genes in a bicluster should induce a heavier subgraph of G because under a subset of the conditions, these genes have highly similar expression patterns that should make the weight of each involved edge heavier, comparing to the edges in the background. But it should be noted that some heavy subgraph may not necessarily correspond to

a bicluster, i.e. genes from a heavy subgraph may not necessarily have similar expression patterns because different edges in a subgraph may have heavier weights under completely different subsets of conditions. It should also be noted that recognizing all heavy subgraphs in a weighted graph itself is computationally intractable because identification of maximum cliques in a graph is a special case of this, and the maximum clique problem is a well known intractable problem (NP-hard). So in our solution, we do not directly solve the problem of finding heavy subgraphs in a graph. Instead, we built our biclustering algorithm based on this graph representation of a microarray gene expression data, and tackle the biclustering problem as follows. We find all feasible biclusters (I,J) in the given data set such that min{|I|, |J|} is as large as possible, where I and J are subsets of genes and conditions, respectively.

Value

Returns a QUBIC biclustering result object (class `QUBICBiclustResult`).

Functions

- `BCQU()`: Performs a QUalitative BIClustering.
- `BCQUD()`: Performs a QUalitative BIClustering for a discret matrix.
- `qubiclust_d()`: Performs a QUalitative BIClustering for a discret matrix.
- `qubiclust()`: Performs a QUalitative BIClustering.

References

Yu Zhang, Juan Xie, Jinyu Yang, Anne Fennell, Chi Zhang, Qin Ma; QUBIC: a bioconductor package for qualitative biclustering analysis of gene co-expression data. *Bioinformatics*, 2017; 33 (3): 450-452.

See Also

`BCQU-class` `qudiscretize` `qunetwork` `qunet2xml`

Examples

```
# Random matrix with one embedded bicluster
test <- matrix(rnorm(5000), 100, 50)
test[11:20, 11:20] <- rnorm(100, 3, 0.3)
res <- qubiclust(test, verbose = FALSE)
summary(res)
show(res)

## Not run:
# Load example microarray matrix
data(ecoli, package = "QUBICdata")

# Biclustering
res <- qubiclust(ecoli, r = 1, q = 0.06, c = 0.95,
                 o = 100, f = 0.25, k = max(ncol(ecoli) %/% 20, 2),
                 verbose = FALSE)
```

```
# Visualize selected biclusters
quheatmap(x = ecoli, bicResult = res, number = 5, showlabel = TRUE)
quheatmap(x = ecoli, bicResult = res, number = c(4, 8), showlabel = TRUE)

# Build network and export XGMML
net <- qunetwork(ecoli, res, number = 5, groups = 5, method = "spearman")
outfile <- tempfile(fileext = ".gr")
sink(outfile)
qunet2xml(net, minimum = 0.6,
         color = cbind(grDevices::rainbow(length(net[[2]]) - 1), "gray"))
sink()
unlink(outfile)

## End(Not run)
# Biclustering of discretized yeast microarray data
mat <- matrix(rnorm(100), 10, 10)
disc <- qudiscretize(mat)
qubiclust_d(disc)
```

`qudiscretize` *Create a qualitative discrete matrix for a given gene expression matrix*

Description

`qudiscretize` delivers a discrete matrix. It is useful if we just want to get a discretized matrix.

Usage

```
qudiscretize(x, r = 1L, q = 0.06)
```

Arguments

`x`	the input data matrix, which could be the normalized gene expression matrix or its qualitative representation from Qdiscretization or other discretization ways. (for example: a qualitative representation of gene expression data) For `BCQU()`, the data matrix should be real For `BCQUD()`, the data matrix should be discretized as integer. Zeros in the matrix will be treated as non-relevant value.
`r`	Affect the granularity of the biclusters. The range of possible ranks. A user can start with a small value of `r` (the default value is `1` so the corresponding data matrix consists of values '`1`', '`-1`' and '`0`'), evaluate the results, and then use larger values (should not be larger than half of the number of the columns) to look for fine structures within the identified biclusters.
`q`	Affect the granularity of the biclusters. The percentage of the regulating conditions for each gene. The choice of `q`'s value depends on the specific application goals; that is if the goal is to find genes that are responsive to local regulators, we should use a relatively small q-value; otherwise we may want to consider larger q-values. The default value of `q` is `0.06` in QUBIC (this value is selected based on the optimal biclustering results on simulated data).

Details

qudiscretize convert a given gene expression matrix to a discrete matrix. It's implimented in C++, providing a increase in speed over the C equivalent.

Value

A qualitative discrete matrix

See Also

QUBIC

Examples

```
# Qualitative discretization on a synthetic matrix
mat <- matrix(rnorm(35), 7, 5)
qudiscretize(mat)
```

---

quheatmap *Visualization of identified biclusters*

Description

This function can visualize the identifed biclusters using heatmap in support of overall expression pattern analysis,either for a single bicluster or two biclusters.

Usage

```
quheatmap(
  x,
  bicResult,
  number = 1,
  showlabel = FALSE,
 col = c("#313695", "#4575B4", "#74ADD1", "#ABD9E9", "#E0F3F8", "#FFFFBF", "#FEE090",
    "#FDAE61", "#F46D43", "#D73027", "#A50026"),
  ...
)
```

Arguments

x	The data matrix
bicResult	Result object returned by qubiclust or qubiclust_d
number	which bicluster to be plotted
showlabel	If TRUE, show the xlabel and ylabel
col	default: c("#313695", "#4575B4", "#74ADD1", "#ABD9E9", "#E0F3F8", "#FFFFBF", "#FEE090", "#FDAE61", "#F46D43", "#D73027", "#A50026")
...	Additional options in fields::image.plot

Value

Invisibly returns NULL after drawing the heatmap.

See Also

qunet2xml QUBIC qunetwork

Examples

```
# Load microarray matrix
if (requireNamespace("QUBICdata", quietly = TRUE)) {
  data(ecoli, package = "QUBICdata")
  res <- qubiclust(ecoli, r = 1, q = 0.06, c = 0.95,
                 o = 100, f = 0.25, k = max(ncol(ecoli) %/% 20, 2),
                 verbose = FALSE)
  # Draw heatmap for the 5th identified bicluster
  par(mar = c(5, 4, 3, 5) + 0.1, mgp = c(0, 1, 0), cex.lab = 1.1,
      cex.axis = 0.5, cex.main = 1.1)
  quheatmap(x = ecoli, res, number = 5, showlabel = TRUE)
  # Draw heatmap for the 4th and 8th identified biclusters.
  par(mar = c(5, 5, 5, 5), cex.lab = 1.1, cex.axis = 0.5, cex.main = 1.1)
  quheatmap(x = ecoli, res, number = c(4, 8), showlabel = TRUE)
}
```

---

qunet2xml *Convert newwork to XGMML*

Description

This function can convert the constructed co-expression networks into XGMML format, which can be used to do further network analysis in Cytoscape, Biomax and JNets.

Usage

```
qunet2xml(
  net,
  minimum = 0.6,
  color = cbind(grDevices::rainbow(length(net[[2]]) - 1), "gray")
)
```

Arguments

net	Result of qunetwork
minimum	cutoff, default: 0.6
color	default: cbind(grDevices::rainbow(length(net[[2]]) - 1), 'gray')

Value

Text of XGMML

See Also

qunetwork QUBIC

Examples

```
# Load microarray matrix
if (requireNamespace("QUBICdata", quietly = TRUE)) {
  data(ecoli, package = "QUBICdata")
  res <- qubiclust(ecoli, r = 1, q = 0.06, c = 0.95,
                   o = 100, f = 0.25, k = max(ncol(ecoli) %/% 20, 2),
                   verbose = FALSE)
  # Get all biclusters
  net <- qunetwork(ecoli, res, number = 5, groups = 5, method = "spearman")
  # Save the network to a XGMML file
  outfile <- tempfile(fileext = ".gr")
  sink(outfile)
  qunet2xml(net, minimum = 0.6,
           color = cbind(grDevices::rainbow(length(net[[2]]) - 1), "gray"))
  sink()
  unlink(outfile)
}
```

---

qunetwork *Construction and visualization of co-expression network*

Description

This function can automatically create co-expression networks along with their visualization based on identified biclusters in QUBIC. Three correlation methods, Pearson, Kendall and Spearman, are available for a user, facilitating different preferences in practical usage.

Usage

```
qunetwork(
  x,
  BicRes,
  number = NULL,
  groups = NULL,
  method = c("pearson", "kendall", "spearman")
)
```

Arguments

x	The data matrix
BicRes	Result object returned by qubiclust or qubiclust_d
number	Which bicluster to be plotted
groups	An object that indicates which nodes belong together.

method — A character string indicating which correlation coefficient (or covariance) is to be computed. One of 'pearson' (default), 'kendall', or 'spearman', can be abbreviated.

Value

a list contains a weights matrix and groupinfo

See Also

qunet2xml QUBIC cor

Examples

```
# Load microarray matrix
if (requireNamespace("QUBICdata", quietly = TRUE)) {
  data(ecoli, package = "QUBICdata")
  res <- qubiclust(ecoli, r = 1, q = 0.06, c = 0.95,
                 o = 100, f = 0.25, k = max(ncol(ecoli) %/% 20, 2),
                 verbose = FALSE)
  # Constructing the network for the 5th identified bicluster.
  net <- qunetwork(ecoli, res, number = 5, groups = 5, method = "spearman")
}
## Not run:
if (requireNamespace('qgraph'))
    qgraph::qgraph(net[[1]], groups = net[[2]], layout = 'spring', minimum = 0.6,
       color = cbind(rainbow(length(net[[2]]) - 1),'gray'), edge.labels = FALSE)


## End(Not run)
## Not run:
# Load microarray matrix
data(ecoli, package = "QUBICdata")
res <- qubiclust(ecoli, r = 1, q = 0.06, c = 0.95,
                o = 100, f = 0.25, k = max(ncol(ecoli) %/% 20, 2),
                verbose = FALSE)
# Constructing the networks for the 4th and 8th identified biclusters,
#   using the whole network as a background.
net <- qunetwork(ecoli, res, number = c(4, 8), groups = c(4, 8), method = "spearman")
if (requireNamespace('qgraph'))
    qgraph::qgraph(net[[1]], groups = net[[2]], layout = 'spring', minimum = 0.6,
       color = c('red', 'blue', 'gold', 'gray'), edge.labels = FALSE)

## End(Not run)
```

---

showinfo *Show report of biclusters*

Description

This function can make a report for biclusters.

Usage

```
showinfo(matrix, bic)
```

Arguments

matrix	microarray matrix
bic	array of biclusters

Value

Text of report

See Also

QUBIC

Examples

```
# Load microarray matrix
if (requireNamespace("QUBICdata", quietly = TRUE)) {
  data(ecoli, package = "QUBICdata")
  matrix <- ecoli[1:50, , drop = FALSE]
  res1 <- qubiclust(matrix, verbose = FALSE)
  res2 <- qubiclust(matrix, c = 0.9, verbose = FALSE)
  res3 <- qubiclust(matrix, c = 0.85, verbose = FALSE)
  # Show the report
  showinfo(matrix, list(res1, res2, res3))
}
```

Index