Package 'ORdensity'

May 13, 2020

Type Package

Title Identification of Differentially Expressed Genes

Version 1.0

Author José María Martínez Otzeta, Concepción Arenas, Basilio Sierra, Itziar Irigoien

Maintainer José María Martínez Otzeta <josemaria.martinezo@ehu.eus>

Description Automated discovery of differentially expressed genes. The method (called ORdensity) is composed of two phases: discovering potential differentially expressed genes and recognizing differentially expressed genes. It makes use of a permutation resampling procedure to build outlying and density indexes. References: a) Irigoien, I. and Arenas, C. (2018). ``Identification of differentially expressed genes by means of outlier detection". <doi:10.1186/s12859-018-2318-8>. b) Martínez-Otzeta, J. M., Irigoien, I., Sierra, B., and Arenas, C. (2020). ``ORdensity: user-friendly R package to identify differentially expressed genes". <doi:10.1186/s12859-020-3463-4>.

Depends R (>= 2.10)

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Imports cluster, distances, Rfast, plyr, methods, foreach, doRNG, parallel, doParallel

NeedsCompilation no

Repository CRAN

Date/Publication 2020-05-13 14:40:02 UTC

R topics documented:

ORdensity-package *Automated discovery of differentially expressed genes*

Description

ORdensity is a package for the automated discovery of differentially expressed genes. It makes use of the ORdensity method and the associated FP and dFP values to detect the most likely differentially expressed (DE) genes. The details of the method are explained in (Martínez-Otzeta, J. M. et al. 2020; Irigoien, I., and Arenas, C. 2018).

Author(s)

José María Martínez Otzeta <josemaria.martinezo@ehu.eus>

Itziar Irigoien <itziar.irigoien@ehu.eus>

Concepción Arenas <carenas@ub.edu>

Basilio Sierra <b.sierra@ehu.eus>

References

Irigoien, I. and Arenas, C. (2018) Identification of differentially expressed genes by means of outlier detection. *BMC Bioinformatics*, 19:317

Martínez-Otzeta, J. M., Irigoien, I., Sierra, B., & Arenas, C. (2020). ORdensity: user-friendly R package to identify differentially expressed genes. *BMC Bioinformatics*, 21, 1-10.

Examples

```
# There is an example dataframe called simexpr shipped with the package. This data is the
# result of a simulation of 100 differentially expressed genes in a pool of 1000 genes. It
# contains 1000 observations of 62 variables. Each row correspond to a gene and contains 62 values:
# DEgen, gap and the values for the gene expression in 30 positive cases and in 30 negative cases.
# The DEgen field value is 1 for differentially expressed genes and 0 for those which are not.
#
# First, let us extract the samples from each experimental condition from the simexpr database.
# For the sake of brevity, we will work with a subset of the database
#
simexpr_reduced <- simexpr[c(1:15,101:235),]
x <- simexpr_reduced[, 3:32]
y <- simexpr_reduced[, 33:62]
EXC.1 <- as.matrix(x)
EXC.2 <- as.matrix(y)
#
# To create an S4 object to perform the analysis, follow this command
#
myORdensity <- new("ORdensity", Exp_cond_1 = EXC.1, Exp_cond_2 = EXC.2, B = 20)
#
# where B = 20 is the number of bootstraps replicates.
#
```

```
# A summary of the object can be generated with the summary function.
#
summary(myORdensity)
#
# The summary tells us the estimated optimal clustering of the data, and the number of genes in
# each cluster, along with their names. The clusters are ordered in decreasig order according to
# the value of the mean of the OR statistic. We see that the mean is higher in the first cluster
# than in the second one, which means that the first cluster is more likely composed of true
# differentially expressed genes, and the second one less likely. With any number of clusters, the
# last ones are likely false negatives.
#
# If the researcher just wants to extract the differentially expressed genes detected by the
# ORdensity method, a call to findDEgenes will return a list with the clusters found, along with
# the values of the OR statistic corresponding to each gene, and an indicator showing if the gene
# fulfil the strong and/or relaxed selection requirements. Following (Irigoien, I., and Arenas, C.
# 2018), two types of differentially expressed gene selection can be made:
#
# ORdensity strong selection: take as differentially expressed genes those with a large OR value
# and with FP and dFP equal to 0.
#
# ORdensity relaxed selection: take as differentially expressed genes those with a large OR
# value and with small FP and dFP values. As a reference to look for small values the expected
# number of false positive neighbours is computed.
#
# The motivation of the clustering is to distinguish those false positives that score high in OR
# and low in meanFP and density, but are similar to other known false positives obtained by
# bootstrapping. The procedure is detailed in (Irigoien, I., and Arenas, C. 2018) and it uses the
# PAM cluster procedure.
#
# After running this code
#
result <- findDEgenes(myORdensity)
#
# the method indicated the numbers of clusters in the optimal clustering, and then we could look
# the results
#
result
#
# As a rule of thumb, differentially expressed genes are expected to present high values of OR
# and low values of meanFP and density. We could also analyze each gene individually inside each
# cluster. The motivation of the clustering is to distinguish those false positives that score
# high in OR and low in meanFP and density, but are similar to other known false positives
# obtained by boostrapping. The procedure is detailed in (Irigoien, I., and Arenas, C. 2018).
#
# If the researcher is interested in a more thorough analysis, other functions are at their service.
#
# The data before being clustered can be obtained with the following function
#
preclusteredData(myORdensity)
#
# A plot with a representation of the potential genes based on OR (vertical axis), FP (horizontal
# axis) and dFP (size of the circle is inversely proportional to its value) can also be obtained.
# Genes that fulfil the relaxed criterion are drawn with triangles.
```

```
#
plot(myORdensity)
#
# By default, the number of clusters computed by the ORdensity method is used. Other values for
# the number of clusters can be specified.
#
plot(myORdensity, numclusters = 5)
```

`findDEgenes` *Clustering of the potential differentially expressed (DE) genes*

Description

This function clusters the potential differentially expressed (DE) genes among them so that the real DE genes can be distinguished from the not DE genes.

Usage

```
findDEgenes(object, numclusters = NULL)

## S4 method for signature 'ORdensity'
findDEgenes(object, numclusters = NULL)
```

Arguments

`object`	An object of `ORdensity` class.
`numclusters`	By default `NULL`, it inherits from the `object` parameter. Optionally, an integer number indicating number of clusters.

Value

A list composed by k lists where k is the best number of clusters found. The clusters are ordered based on their importance according to the mean OR values of the clusters (the greater the mean OR value of the cluster the more important are the genes in the cluster). The first one is the most important, the last one the less important. Each list has elements:

- `numberOfGenes`: Number of genes in the cluster.
- `CharacteristicsCluster`: Matrix with mean values and standard deviation of variables OR, FP and dFP for each cluster.
- `Genes`: Identification of the genes in the cluster.

Examples

```
# Read data from 2 experimental conditions
simexpr_reduced <- simexpr[c(1:15,101:235),]
x <- simexpr_reduced[, 3:32]
y <- simexpr_reduced[, 33:62]
EXC.1 <- as.matrix(x)
```

```
EXC.2 <- as.matrix(y)
myORdensity <- new("ORdensity", Exp_cond_1 = EXC.1, Exp_cond_2 = EXC.2, B = 20)
out <- findDEgenes(myORdensity)
# For instance, characteristics of cluster1, likely composed of true DE genes
out[[1]]
# It is also possible to choose the number of clusters
out <- findDEgenes(myORdensity, 5)
```

`ORdensity-class` *S4 class for representing potential differentially expressed genes*

Description

An object of class ORdensity includes all potential differentially expressed genes given microarray data measured in two experimental conditions.

Slots

`Exp_cond_1` Matrix including microarray data measured under experimental condition 1.

`Exp_cond_2` matrix including microarray data measured under experimental condition 2.

`labels` Vector of characters identifying the genes, by default rownames(Exp_cond_1) is inherited. If NULL, the genes are named 'Gene1', ..., 'Genen' according to the order given in `Exp_cond_1`.

`B` Numeric value indicating the number of permutations. By default, `B`=100.

`scale` Logical value to indicate whether the scaling of the difference of quantiles should be done.

`alpha` Numeric value used by the method to calculate the percentile $(1-\alpha)100$ of all the elements of the matrix with the permuted samples. By default 0.05.

`fold` Numeric value, by default `fold`=10. It controls the number of partitions.

`probs` Vector of numerics. It sets the quantiles to be considered. By default `probs = c(0.25,0.5,0.75)`.

`weights` Vector of numerics. It controls the weights given to the quantiles set in `probs`. By default `weights = c(1/4,1/2,1/4)`.

`numneighbours` Numeric value to set the number of nearest neighbours. By default `numneighbours=10`.

`numclustoseek` Numeric value to set the number of maximum clusters to consider. By default `numclustoseek=10`.

`out` List containing the potential DE genes and their characteristics.

`OR` Outlyingness index (See Martínez-Otzeta, J. M. et al. 2020; Irigoien, I., and Arenas, C. 2018).

`FP` Average number of false positive permuted cases in the neighbourhood (See Martínez-Otzeta, J. M. et al. 2020; Irigoien, I., and Arenas, C. 2018).

`dFP` Average density of false positive permuted cases in the neighbourhood (See Martínez-Otzeta, J. M. et al. 2020; Irigoien, I., and Arenas, C. 2018).

`char` Matrix holding internal computations. Non-developers should left this parameter as default.

`bestKclustering` Number of clusters for partitioning the data. It is advisable to let the object to automatically estimate the best partition.

`verbose` Boolean indicating if log messages are going to be printed.

`parallel` Boolean indicating if parallel process is used.

`nprocs` Integer indicating the number of processors to be used. If nprocs is 0 or negative, the number of processors detected in the machine is used.

`replicable` Boolean indicating if the same seed is used for the pseudorandom number generation.

`seed` Integer used as seed by the pseudorandom number generator.

Examples

```
# To create an instance of a class ORdensity given data from 2 experimental conditions
simexpr_reduced <- simexpr[c(1:15,101:235),]
x <- simexpr_reduced[, 3:32]
y <- simexpr_reduced[, 33:62]
EXC.1 <- as.matrix(x)
EXC.2 <- as.matrix(y)
myORdensity <- new("ORdensity", Exp_cond_1 = EXC.1, Exp_cond_2 = EXC.2, B = 20)
```

---

plot.ORdensity *Plot function implemented by ORdensity class*

Description

Plots a representation of the potential genes based on OR, FP and dFP.

Usage

```
## S3 method for class 'ORdensity'
plot(x, numclusters = x@bestKclustering, ...)
```

Arguments

`x`	Object of class `ORdensity`.
`numclusters`	By default `NULL`, it inherits from the `x`. Optionally, an integer number indicating number of clusters the genes are partitioned.
`...`	Optional arguments inherited from the generic `plot` method.

Value

Displays a plot with a representation of the potential genes based on OR (vertical axis), FP (horizontal axis) and dFP (size of the symbol is inversely proportional to its value). Moreover, genes identified as DE by the relaxed selection are represented by the symbol $\triangle$.

Examples

```
# Read data from 2 experimental conditions
simexpr_reduced <- simexpr[c(1:15,101:235),]
x <- simexpr_reduced[, 3:32]
y <- simexpr_reduced[, 33:62]
EXC.1 <- as.matrix(x)
EXC.2 <- as.matrix(y)
myORdensity <- new("ORdensity", Exp_cond_1 = EXC.1, Exp_cond_2 = EXC.2, B = 20)
plot(myORdensity)
```

---

preclusteredData *Preprocessed description of all the identified potential DE genes*

Description

This function returns the description of all the identified potential DE genes in terms of variables OR, FP, and dFP in one only table so that the interesed user can perform her own clustering analysis.

Usage

```
preclusteredData(object, verbose = TRUE)

## S4 method for signature 'ORdensity'
preclusteredData(object, verbose = TRUE)
```

Arguments

object	Object of class ORdensity.
verbose	Boolean indicating if log messages are going to be printed.

Value

data.frame with all potential DE genes.

Examples

```
# Read data from 2 experimental conditions
simexpr_reduced <- simexpr[c(1:15,101:235),]
x <- simexpr_reduced[, 3:32]
y <- simexpr_reduced[, 33:62]
EXC.1 <- as.matrix(x)
EXC.2 <- as.matrix(y)
myORdensity <- new("ORdensity", Exp_cond_1 = EXC.1, Exp_cond_2 = EXC.2, B = 20)
# dataframe with all potential DE genes:
preclusteredData(myORdensity)
```

simexpr *Simulated data with differentially expressed (DE) genes*

Description

Simulated data with 1000 genes measured under two different experimental conditions 1 and 2. 100 genes among the 1000 were generated as differentially expressed (DE) genes. The expression levels of all no DE genes were generated by $N(0,1)$ distribution in both conditions 1 and 2. The DE genes were generated using the $N(0,1)$ and $N(\mu_g,1)$ distributions for conditions 1 and 2, respectively, with $|\mu_g| = \Delta$. Parameter Δ sets the importance of gene g, where the bigger Δ is, the more important gene g is. We considered Δ in $\{1.5, 2, 3\}$. Each row g in `simexpr` corresponds to a simulated gene.

Usage

```
simexpr
```

Format

A dataframe with 1000 rows and 62 variables:

DEgen It indicates whether gene g is DE or not.

gap It contains Δ values.

A1-A30 These columns have the expression levels under experimental condition 1.

B1-B30 These columns have the expression levels under experimental condition 2.

summary.ORdensity *Summary function implemented by ORdensity class*

Description

This function clusters the potential differentially expressed (DE) genes among them so that the real DE genes can be distinguish from the not DE genes.

Usage

```
## S3 method for class 'ORdensity'
summary(object, numclusters = NULL, ...)
```

Arguments

`object`	An object of `ORdensity` class.
`numclusters`	By default `NULL`, it inherits from the ORdensity `object`. Optionally, an integer number indicating number of clusters.
`...`	Optional arguments inherited from the generic `summary` method.

Details

Once the potential DE genes are identified, the real DE genes and the not real DE genes or false positives must be distinguished. Since the real DE genes must have high OR values along with low FP and dFP values, and on the contrary, the false DE genes must have low OR values but high FP and dFP values, a clustering of all the potential DE genes is carried out. The clustering is based on build-on variables OR, FP and dFP (see class `ORdensity`) which are scaled. The clustering algorithm is `pam` and by default the number of clusters in the partition is obtained by `silhouette`. With parameter `numclusters` the number of clusters in the partition can be customized.

Value

A list with k lists where k is the best number of clusters found. The clusters are ordered based on their importance according to the mean OR values of the clusters (greater is the mean OR value of the cluster more important are the genes in the cluster). The first one is the most important, the last one the less important. Each list has elements:

- `numberOfGenes`: Number of genes in the cluster.
- `CharacteristicsCluster`: Matrix with mean values and standard deviation of variables OR, FP and dFP for each cluster.
- `Genes`: Identification of the genes in the cluster.

Examples

```
# Read data from 2 experimental conditions
simexpr_reduced <- simexpr[c(1:15,101:235),]
x <- simexpr_reduced[, 3:32]
y <- simexpr_reduced[, 33:62]
EXC.1 <- as.matrix(x)
EXC.2 <- as.matrix(y)
myORdensity <- new("ORdensity", Exp_cond_1 = EXC.1, Exp_cond_2 = EXC.2, B = 20)
summary(myORdensity)
```

Index