Package 'isomiRs'

October 18, 2017

Version 1.4.0

Date 2017-04-08

Type Package

Title Analyze isomiRs and miRNAs from small RNA-seq

Description Characterization of miRNAs and isomiRs, clustering and differential expression.

biocViews miRNA, RNASeq, DifferentialExpression, Clustering

Suggests knitr, RUnit, org.Mm.eg.db, cluster, clusterProfiler, BiocStyle

Depends R (>= 3.2), DiscriMiner, IRanges, S4Vectors (>= 0.9.25), GenomicRanges, SummarizedExperiment (>= 1.1.6)

Imports BiocGenerics (>= 0.7.5), DESeq2, reshape, tidyr, plyr, readr, dplyr, RColorBrewer, gplots, methods, ggplot2, gtools, gridExtra, grid, GGally

Author Lorena Pantano, Georgia Escaramis

Maintainer Lorena Pantano <lorena.pantano@gmail.com>

License MIT + file LICENSE

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

R topics documented:

`counts` *Accessors for the count matrix of a IsomirDataSeq object.*

Description

The counts slot holds the count data as a matrix of non-negative integer count values, one row for each isomiR, and one column for each sample. The normalized matrix can be obtained by using the parameter `norm=TRUE`.

Usage

```
counts.IsomirDataSeq(object, norm = FALSE)

## S4 method for signature 'IsomirDataSeq'
counts(object, norm = FALSE)

## S4 replacement method for signature 'IsomirDataSeq,matrix'
counts(object) <- value
```

Arguments

`object`	a `IsomirDataSeq` object
`norm`	TRUE return log2-normalized counts
`value`	an integer matrix

Value

`matrix` with raw or normalized count data.

Author(s)

Lorena Pantano

Examples

```
data(mirData)
head(counts(mirData))
```

design *Accessors for the 'design' slot of a IsomirDataSeq object.*

Description

The design holds the R `formula` which expresses how the counts depend on the variables in `colData`. See `IsomirDataSeq` for details.

Usage

```
## S4 method for signature 'IsomirDataSeq'
design(object)

## S4 replacement method for signature 'IsomirDataSeq,formula'
design(object) <- value
```

Arguments

`object`	a `IsomirDataSeq` object
`value`	a `formula` to pass to DESeq2

Examples

```
data(mirData)
design(mirData) <- formula(~ 1)
```

find_targets *Find miRNAs target using mRNA/miRNA expression*

Description

This function creates a matrix with rows (genes) and columns (mirnas) with values indicating if miRNA-gene pair is target according putative targets and negative correlation of the expression of both molecules.

Usage

```
find_targets(mirna_rse, gene_rse, target, summarize = "group",
  min_cor = -0.6)
```

Arguments

`mirna_rse`	SummarizedExperiment with miRNA information. See details.
`gene_rse`	SummarizedExperiment with gene information. See details.
`target`	matrix with miRNAs (columns) and genes (rows) target prediction values (1 if it is a target, 0 if not).

summarize	character column name in colData(rse) to use to group samples and compare betweem miRNA/gene expression.
min_cor	numeric cutoff for correlation value that will be use to consider a miRNA-gene pair as valid.

Examples

```
pairs <- as.matrix(data.frame(row.names=c("gene1", "gene2"),
                              mirna1=c(0,1), mirna2=c(1,0)))
mirna_matrix <- as.matrix(data.frame(row.names=c("mirna1", "mirna2"),
                                     time0_1=c(1,1),time0_2=c(1.2,0.9),
                                     time1_1=c(8,8),time1_2=c(8.2,7.9)))
gene_matrix <- as.matrix(data.frame(row.names=c("gene1", "gene2"),
                                     time0_1=c(8,8),time0_2=c(8.2,7.9),
                                     time1_1=c(1,1),time1_2=c(1.2,0.9)))
mirna_col <- data.frame(row.names=c("time0_1","time0_2","time1_1","time1_2"),
                        group=c("t0","t0","t1","t1"))
gene_col <- data.frame(row.names=c("time0_1","time0_2","time1_1","time1_2"),
                        group=c("t0","t0","t1","t1"))

mirna <- SummarizedExperiment(assays=SimpleList(norm=as.matrix(mirna_matrix)),
                              colData= mirna_col)
gene <- SummarizedExperiment(assays=SimpleList(norm=as.matrix(gene_matrix)),
                               colData= gene_col)
find_targets(mirna, gene, pairs)
```

---

isoCounts *Create count matrix with different summarizing options*

Description

This function collapses isomiRs into different groups. It is a similar concept than how to work with gene isoforms. With this function, different changes can be put together into a single miRNA variant. For instance all sequences with variants at 3' end can be considered as different elements in the table or analysis having the following naming `hsa-miR-124a-5p.iso.t3:AAA`.

Usage

```
isoCounts(ids, ref = FALSE, iso5 = FALSE, iso3 = FALSE, add = FALSE,
  subs = FALSE, seed = FALSE, minc = 1, mins = 1)
```

Arguments

ids	object of class IsomirDataSeq
ref	differentiate reference miRNA from rest
iso5	differentiate trimming at 5 miRNA from rest
iso3	differentiate trimming at 3 miRNA from rest
add	differentiate additions miRNA from rest
subs	differentiate nt substitution miRNA from rest
seed	differentiate changes in 2-7 nts from rest

minc	int minimum number of isomiR sequences to be included.
mins	int minimum number of samples with number of sequences bigger than minc counts.

Details

You can merge all isomiRs into miRNAs by calling the function only with the first parameter isoCounts(ids). You can get a table with isomiRs altogether and the reference miRBase sequences by calling the function with ref=TRUE. You can get a table with 5' trimming isomiRS, miRBase reference and the rest by calling with isoCounts(ids, ref=TRUE, iso5=TRUE). If you set up all parameters to TRUE, you will get a table for each different sequence mapping to a miRNA (i.e. all isomiRs).

Examples for the naming used for the isomiRs are at http://seqcluster.readthedocs.org/mirna_annotation.html#mirna-annotation.

Value

IsomirDataSeq object with new count table. The count matrix can be access with counts(ids).

Examples

```
data(mirData)
ids <- isoCounts(mirData, ref=TRUE)
head(counts(ids))
# taking into account isomiRs and reference sequence.
ids <- isoCounts(mirData, ref=TRUE, minc=10, mins=6)
head(counts(ids))
```

---

isoDE *Differential expression analysis with DESeq2*

Description

This function does differential expression analysis with DESeq2-package using the specific formula. It will return a DESeqDataSet object.

Usage

```
isoDE(ids, formula = NULL, ...)
```

Arguments

ids	object of class IsomirDataSeq
formula	used for DE analysis
...	options to pass to isoCounts including ref, iso5, iso3, add, subs and seed parameters.

Details

First, this function collapses all isomiRs in different types. Read more at isoCounts to know the different options available to collapse isomiRs.

After that, DESeq2-package is used to do differential expression analysis. It uses the count matrix and design experiment stored at (counts(ids) and colData(ids)) IsomirDataSeq object to construct a DESeqDataSet object.

Value

DESeqDataSet object. To get the differential expression isomiRs, use results from DESeq2 package. This allows to ask for different contrast without calling again isoDE. Read results manual to know how to access all the information.

Examples

```
data(mirData)
ids <- isoCounts(mirData, minc=10, mins=6)
dds <- isoDE(mirData, formula=~group)
```

IsomirDataSeq-class *Class that contains all isomiRs annotation for all samples*

Description

The IsomirDataSeq is a subclass of SummarizedExperiment used to store the raw data, intermediate calculations and results of an miRNA/isomiR analysis. This class stores all raw isomiRs data for each sample, processed information, summary for each isomiR type, raw counts, normalized counts, and table with experimental information for each sample.

Details

IsomirDataSeqFromFiles creates this object using seqbuster output files.

Methods for this objects are counts to get count matrix and isoSelect for miRNA/isomiR selection. Functions available for this object are isoCounts for count matrix creation, isoNorm for normalization, isoDE for differential expression and isoPLSDA for clustering. isoPlot helps with basic expression plot.

metadata contains two lists: rawList is a list with same length than number of samples and stores the input files for each sample; isoList is a list with same length than number of samples and stores information for each isomiR type summarizing the different changes for the different isomiRs (trimming at 3', trimming a 5', addition and substitution). For instance, you can get the data stored in isoList for sample 1 and 5' changes with this code metadata(ids)[['isoList']][[1]]$t5sum.

The naming of isomiRs follows these rules:

- miRNA name
- type:ref if the sequence is the same than the miRNA reference. iso if the sequence has variations.

- `t5 tag`:indicates variations at 5 position. The naming contains two words: `direction - nucleotides`, where direction can be UPPER CASE NT (changes upstream of the 5 reference position) or LOWER CASE NT (changes downstream of the 5 reference position). `0` indicates no variation, meaning the 5 position is the same than the reference. After `direction`, it follows the nucleotide/s that are added (for upstream changes) or deleted (for downstream changes).
- `t3 tag`:indicates variations at 3 position. The naming contains two words: `direction - nucleotides`, where direction can be LOWER CASE NT (upstream of the 3 reference position) or UPPER CASE NT (downstream of the 3 reference position). `0` indicates no variation, meaning the 3 position is the same than the reference. After `direction`, it follows the nucleotide/s that are added (for downstream changes) or deleted (for upstream chanes).
- `ad tag`:indicates nucleotides additions at 3 position. The naming contains two words: `direction - nucleotides`, where direction is UPPER CASE NT (upstream of the 5 reference position). `0` indicates no variation, meaning the 3 position has no additions. After `direction`, it follows the nucleotide/s that are added.
- `mm tag`: indicates nucleotides substitutions along the sequences. The naming contains three words: `position-nucleotideATsequence-nucleotideATreference`.
- `seed tag`: same than `mm` tag, but only if the change happens between nucleotide 2 and 8.

In general nucleotides in UPPER case mean insertions respect to the reference sequence, and nucleotides in LOWER case mean deletions respect to the reference sequence.

Examples

```
path <- system.file("extra", package="isomiRs")
fn_list <- list.files(path, full.names = TRUE)
de <- data.frame(row.names=c("f1" , "f2"), condition = c("newborn", "newborn"))
ids <- IsomirDataSeqFromFiles(fn_list, coldata=de)

head(counts(ids))
```

`IsomirDataSeqFromFiles`

`IsomirDataSeqFromFiles` *loads miRNA annotation from seqbuster tool*

Description

This function parses output of seqbuster tool to allow isomiRs/miRNAs analysis of samples in different groups such as characterization, differential expression and clustering. It creates an `IsomirDataSeq` object.

Usage

```
IsomirDataSeqFromFiles(files, coldata, rate = 0.2, canonicalAdd = TRUE,
  uniqueMism = TRUE, design = ~1, header = TRUE, skip = 0,
  quiet = TRUE, ...)
```

Arguments

files	files with the output of seqbuster tool
coldata	data frame containing groups for each sample
rate	minimum counts fraction to consider a mismatch a real mutation
canonicalAdd	boolean only keep A/T non-template addition. All non-template nucleotides at the 3' end will be removed if they contain C/G nts.
uniqueMism	boolean only keep mutations that have a unique hit to one miRNA molecule
design	a formula to pass to DESeqDataSet
header	boolean to indicate files contain headers
skip	skip first line when reading files
quiet	boolean indicating to print messages while reading files. Default FALSE.
...	arguments provided to SummarizedExperiment including rowData.

Details

This function parses the output of http://seqcluster.readthedocs.org/mirna_annotation.html for each sample to create a count matrix for isomiRs, miRNAs or isomiRs grouped in types (i.e all sequences with variations at 5' but ignoring any other type). It creates IsomirDataSeq object (see link to example usage of this class) to allow visualization, queries, differential expression analysis and clustering. To create the IsomirDataSeq, it parses the isomiRs files, and generates an initial matrix having all isomiRs detected among samples. As well, it creates a summary for each isomiR type (trimming, addition and substitution) to visualize general isomiRs distribution.

Value

IsomirDataSeq class object.

Examples

```
path <- system.file("extra", package="isomiRs")
fn_list <- list.files(path, full.names = TRUE)
de <- data.frame(row.names=c("f1" , "f2"), condition = c("newborn", "newborn"))
ids <- IsomirDataSeqFromFiles(fn_list, coldata=de)

head(counts(ids))
```

---

isoNetwork *Clustering miRNAs-genes pairs in similar pattern expression*

Description

Clustering miRNAs-genes pairs in similar pattern expression

Usage

```
isoNetwork(mirna_rse, gene_rse, target, summarize = "group", org,
  genename = "ENSEMBL", min_cor = -0.6)
```

Arguments

mirna_rse	SummarizedExperiment with miRNA information. See details.
gene_rse	SummarizedExperiment with gene information. See details.
target	matrix with miRNAs (columns) and genes (rows) target prediction (1 if it is a target, 0 if not).
summarize	character column name in colData(rse) to use to group samples and compare betweem miRNA/gene expression.
org	AnnotationDb. (org.Mm.eg.db)
genename	character keytype of the gene names in gene_rse object.
min_cor	numeric cutoff to consider a miRNA to regulate a target

Details

This function will correlate miRNA and gene expression data using a specific metadata variable to group samples and detect pattern of expression that will be annotated with GO terms. mirna_rse and gene_rse can be created using the following code:

```
mi_rse = SummarizedExperiment(assays=SimpleList(norm=mirna_matrix), colData, metadata=list(sign=m
```

where, `mirna_matrix` is the normalized counts expression, `colData` is the metadata information and `mirna_keep` the list of miRNAs to be used by this function.

Examples

```
library(org.Mm.eg.db)
library(clusterProfiler)
data(isoExample)
ego <- enrichGO(row.names(assay(gene_ex_rse, "norm")), org.Mm.eg.db, ont = "BP", keytype="ENSEMBL")
data = isoNetwork(mirna_ex_rse, gene_ex_rse, ma_ex, org=ego@result)
isoPlotNet(data)
```

---

isoNorm *Normalize count matrix*

Description

This function normalizes raw count matrix using `rlog` function from `DESeq2-package`.

Usage

```
isoNorm(ids, formula = NULL)
```

Arguments

ids	object of class IsomirDataSeq
formula	formula that will be used for normalization

Value

IsomirDataSeq object with the normalized count matrix in a slot. The normalized matrix can be access with `counts(ids, norm=TRUE)`.

Examples

```
data(mirData)
ids <- isoCounts(mirData, minc=10, mins=6)
ids <- isoNorm(mirData, formula=~group)
head(counts(ids, norm=TRUE))
```

`isoPlot` *Plot the amount of isomiRs in different samples*

Description

This function plot different isomiRs proportion for each sample. It can show trimming events at both side, additions and nucleotides changes.

Usage

```
isoPlot(ids, type = "iso5", column = "condition")
```

Arguments

`ids`	object of class `IsomirDataSeq`
`type`	string (iso5, iso3, add, subs) to indicate what isomiRs to use for the plot. See details for explanation.
`column`	string indicating the column in `colData` to color samples.

Details

There are four different values for `type` parameter. To plot trimming at 5' or 3' end, use `type="iso5"` or `type="iso3"`. In this case, it will plot 3 positions at both side of the reference position described at miRBase site. Each position refers to the number of sequences that start/end before or after the miRBase reference. The color indicates the sample group. The size of the point is proportional to the number of total counts. The position at `y` is the number of different sequences.

Same logic applies to `type="add"` and `type="subs"`. However, when `type="add"`, the plot will refer to addition events from the 3' end of the reference position. Note that this additions don't match to the precursor sequence, they are non-template additions. In this case, only 3 positions after the 3' end will appear in the plot. When `type="subs"`, it will appear one position for each nucleotide in the reference miRNA. Points will indicate isomiRs with nucleotide changes at the given position.

Value

ggplot object showing different isomiRs changes at different positions.

Examples

```
data(mirData)
isoPlot(mirData, column="group")
```

isoPlotNet *Functional miRNA / gene expression profile plot*

Description

Functional miRNA / gene expression profile plot

Usage

```
isoPlotNet(obj)
```

Arguments

obj	output from isoNetwork

isoPlotPosition *Plot nucleotides changes at a given position*

Description

This function plot different isomiRs proportion for each sample at a given position focused on the nucleotide change that happens there.

Usage

```
isoPlotPosition(ids, position = 1, column = "condition")
```

Arguments

ids	object of class IsomirDataSeq
position	integer indicating the position to show
column	string indicating the column in colData to color samples.

Details

It shows the nucleotides changes at the given position for each sample in each group. The color indicates the sample group. The size of the point is proportional to the number of total counts of isomiRs with changes. The position at y is the number of different sequences supporting the change.

Value

ggplot object showing nucleotide changes at a given position.

Examples

```
data(mirData)
isoPlotPosition(mirData, column="group")
```

isoPLSDA *Partial Least Squares Discriminant Analysis for* IsomirDataSeq

Description

Use PLS-DA method with the normalized count data to detect the most important features (miRNAs/isomiRs) that explain better the group of samples given by the experimental design. It is a supervised clustering method with permutations to calculate the significance of the analysis.

Usage

```
isoPLSDA(ids, group, validation = NULL, learn = NULL, test = NULL,
 tol = 0.001, nperm = 400, refinment = FALSE, vip = 1.2)
```

Arguments

ids	object of class IsomirDataSeq
group	column name in colData(ids) to use as variable to explain.
validation	type of validation, either NULL or "learntest". Default NULL
learn	optional vector of indexes for a learn-set. Only used when validation="learntest". Default NULL
test	optional vector of indices for a test-set. Only used when validation="learntest". Default NULL
tol	tolerance value based on maximum change of cumulative R-squared coefficient for each additional PLS component. Default tol=0.001
nperm	number of permutations to compute the PLD-DA p-value based on R2 magnitude. Default nperm=400
refinment	logical indicating whether a refined model, based on filtering out variables with low VIP values
vip	Variance Importance in Projection threshold value when a refinement process is considered. Default vip=1.2

Details

Partial Least Squares Discriminant Analysis (PLS-DA) is a technique specifically appropriate for analysis of high dimensionality data sets and multicollinearity (*Perez-Enciso, 2013*). PLS-DA is a supervised method (i.e. makes use of class labels) with the aim to provide a dimension reduction strategy in a situation where we want to relate a binary response variable (in our case young or old status) to a set of predictor variables. Dimensionality reduction procedure is based on orthogonal transformations of the original variables (miRNAs/isomiRs) into a set of linearly uncorrelated latent variables (usually termed as components) such that maximizes the separation between the different classes in the first few components (*Xia, 2011*). We used sum of squares captured by the model (R2) as a goodness of fit measure.

We implemented this method using the DiscriMiner-package into isoPLSDA function. The output p-value of this function will tell about the statistical significant of the group separation using miRNA/isomiR expression data.

Read more about the parameters related to the PLS-DA directly from plsDA function.

Value

A list with the following elements: R2Matrix (R-squared coefficients of the PLS model), components (of the PLS, similar to PCs in a PCA), vip (most important isomiRs/miRNAs), group (classification of the samples), p.value and R2PermutationVector obtained by the permutations.

If the option refinment is set to TRUE, then the following elements will appear: R2RefinedMatrix and componentsRefinedModel (R-squared coefficients of the PLS model only using the most important miRNAs/isomiRs). As well, p.valRefined and R2RefinedPermutationVector with p-value and R2 of the permutations where samples were randomized. And finally, p.valRefinedFixed and R2RefinedFixedPermutationVector with p-value and R2 of the permutations where miRNAs/isomiRs were randomized.

References

Perez-Enciso, Miguel and Tenenhaus, Michel. Prediction of clinical outcome with microarray data: a partial least squares discriminant analysis (PLS-DA) approach. Human Genetics. 2003.

Xia, Jianguo and Wishart, David S. Web-based inference of biological patterns, functions and pathways from metabolomic data using MetaboAnalyst. Nature Protocols. 2011.

Examples

```
data(mirData)
# Only miRNAs with > 10 reads in all samples.
ids <- isoCounts(mirData, minc=10, mins=6)
ids <- isoNorm(ids, formula=~group)
pls.ids = isoPLSDA(ids, "group", nperm = 2)
cat(paste0("pval:",pls.ids$p.val))
cat(paste0("components:",pls.ids$components))
```

---

isoPLSDAplot *Plot components from isoPLSDA analysis (pairs plot)*

Description

Plot the most significant components that come from isoPLSDA analysis together with the density of the samples scores along those components.

Usage

```
isoPLSDAplot(pls, n = 2)
```

Arguments

pls	output from isoPLSDA function.
n	integer number of components to plot

Details

The function isoPLSDAplot helps to visualize the results from isoPLSDA. It will plot the samples using the significant components (t1, t2, t3 ...) from the PLS-DA analysis and the samples score distribution along the components. It uses ggpairs for the plot.

Value

ggpairs plot showing the scores for each sample using isomiRs/miRNAs expression to explain variation.

data.frame object with a first column refering to the sample group, and the following columns refering to the score that each sample has for each component from the PLS-DA analysis.

Examples

```
data(mirData)
# Only miRNAs with > 10 reads in all samples.
ids <- isoCounts(mirData, minc=10, mins=6)
ids <- isoNorm(ids, formula=~group)
pls.ids <- isoPLSDA(ids, "group", nperm = 2)
isoPLSDAplot(pls.ids)
```

---

isoSelect *Method to select specific miRNAs from an IsomirDataSeq object.*

Description

This method allows to select a miRNA and all its isomiRs from the count matrix.

Usage

```
isoSelect.IsomirDataSeq(object, mirna, minc = 10)

## S4 method for signature 'IsomirDataSeq'
isoSelect(object, mirna, minc = 10)
```

Arguments

object	a IsomirDataSeq object.
mirna	string referring to the miRNA to show
minc	int minimum number of isomiR reads needed to be included in the table.

Value

DataFrame-class with count information. The row.names show the isomiR names, and each of the columns shows the counts for this isomiR in that sample. Mainly, it will return the count matrix only for isomiRs belonging to the miRNA family given by the mirna parameter. IsomiRs need to have counts bigger than minc parameter at least in one sample to be included in the output.

Author(s)

Lorena Pantano

Examples

```
data(mirData)
# To select isomiRs from let-7a-5p miRNA
# and with 10000 reads or more.
isoSelect(mirData, mirna="hsa-let-7a-5p", minc=10000)
```

isoTop — *Heatmap of the top expressed isomiRs*

Description

This function creates a heatmap with the top N isomiRs/miRNAs. It uses the matrix under `counts(ids)` to get the top expressed isomiRs/miRNAs using the average expression value and plot a heatmap with the raw counts for each sample.

Usage

```
isoTop(ids, top = 20)
```

Arguments

`ids`	object of class `IsomirDataSeq`
`top`	number of isomiRs/miRNAs used

Examples

```
data(mirData)
isoTop(mirData)
```

mirData — *Example of IsomirDataSeq with human brain miRNA counts data*

Description

This data set is the object return by `IsomirDataSeqFromFiles`. It contains miRNA count data from 6 samples: 3 newborns and 3 elderly human individuals (Somel et al, 2010). Use `colData` to see the experiment design.

Usage

```
data("mirData")
```

Format

a `IsomirDataSeq` class.

Author(s)

Lorena Pantano, 2016-04-07

Source

Data is available from GEO dataset under accession number GSE97285

Every sample was analyzed with seqbuster tool, see http://seqcluster.readthedocs.org/mirna_annotation.html for more details. You can get same files running the small RNA-seq pipeline from https://github.com/chapmanb/bcbio-nextgen.

bcbio_nextgen was used for the full analysis.

```
library(isomiRs) files = list.files(file.path(root_path),pattern = "mirbase-ready",recursive = T,f
metadata_fn =  list.files(file.path(root_path),pattern = "summary.csv$",recursive = T, full.names
metadata = read.csv(metadata_fn, row.names="sample_id") condition = names(metadata)[1]
mirData <- IsomirDataSeqFromFiles(files[rownames(design)], metadata)
```

References

Pantano L, Friedlander MR, Escaramis G, Lizano E et al. Specific small-RNA signatures in the amygdala at premotor and motor stages of Parkinson's disease revealed by deep sequencing analysis. Bioinformatics 2016 Mar 1;32(5):673-81. PMID: 26530722

Index