Package 'ERSSA'

November 11, 2026

Type Package

Title Empirical RNA-seq Sample Size Analysis

Version 1.30.0

Date 2020-01-21

Author Zixuan Shao [aut, cre]

Maintainer Zixuan Shao <Zixuanshao.zach@gmail.com>

Description The ERSSA package takes user supplied RNA-seq differential expression dataset and calculates the number of differentially expressed genes at varying biological replicate levels. This allows the user to determine, without relying on any a priori assumptions, whether sufficient differential detection has been acheived with their RNA-seq dataset.

Depends R (>= 4.0.0)

Imports edgeR (>= 3.23.3), DESeq2 (>= 1.21.16), ggplot2 (>= 3.0.0), RColorBrewer (>= 1.1-2), plyr (>= 1.8.4), BiocParallel (>= 1.15.8), apeglm (>= 1.4.2), grDevices, stats, utils

Suggests BiocStyle, knitr, rmarkdown

VignetteBuilder knitr

biocViews ImmunoOncology, GeneExpression, Transcription, DifferentialExpression, RNASeq, MultipleComparison, QualityControl

License GPL-3 | file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/zshao1/ERSSA

RoxygenNote 7.0.2

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

git_branch RELEASE_3_23

git_last_commit afda6cc

git_last_commit_date 2026-04-28

Repository Bioconductor 3.23

Date/Publication 2026-11-11

Contents

combinations.partial . . . 2
comb_gen . . . 2
condition_table.full . . . 4
condition_table.partial . . . 4
count_filter . . . 4
count_table.filtered.partial . . . 5
count_table.full . . . 5
count_table.partial . . . 6
deg.partial . . . 6
erssa . . . 7
erssa_deseq2 . . . 10
erssa_deseq2_parallel . . . 12
erssa_edger . . . 14
erssa_edger_parallel . . . 15
ggplot2_dotplot . . . 18
ggplot2_intersectPlot . . . 19
ggplot2_marginPlot . . . 20
ggplot2_TPR_FPRPlot . . . 21

Index **23**

`combinations.partial` *Example list of combinations generated by comb_gen function.*

Description

Generated with example sample set that contains 4 replicates each of heart and muscle from GTeX. 5 combinations generated per replicate level. Used for testing purposes and to shorten run time.

`comb_gen` *Calculate unique combinations of samples at varying replicate number*

Description

`comb_gen` function takes the list of samples in each condition and generate unique combination of sample names that allow subsampling at varying replicate numbers.

Usage

```
comb_gen(condition_table = NULL, n_repetition = 30)
```

Arguments

condition_table	A condition table with two columns and each sample as a row. Column 1 contains sample names and Column 2 contains sample condition (e.g. Control, Treatment).
n_repetition	The number of maximum unique combinations to generate at each replicate level. More tests will be performed with a bigger value, but run time also increases linearly. Default set to 30 unique combinations.

Details

At each replicate number (2 to N-1), the total number of unique combination of samples is computed. For example, 10 condition A samples subsampled at replicate number of 2 has 45 unique combinations.

When the total number of possible combinations at a particular replicate number is more than the specified x number of repetition (default=30), then only x unique combinations are selected.

When the total number of possible combination is smaller than the specified x number of repetitions, then only unique combinations are selected. For example, 10 samples subsample for 9 replicates has 10 unique combinations and only 10 combinations will be selected instead of 30.

This is repeated for both conditions and another level of combination is performed to combine samples from the two conditions. Again, only x number of unique combinations selected. When total unique combination is smaller than x, then all unique combinations are selected.

Selected combinations of samples at each replicate number are then returned.

Value

A list of character vectors containing sample combinations. Each element in the list corresponds to a replicate level. Each combination within the character vector is a single string with sample names separated by semicolon.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

Examples

```
# Use example condition_table
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(condition_table.partial, package = "ERSSA")

combinations.partial = comb_gen(condition_table.partial, n_repetition=5)
```

condition_table.full *Example table of sample names and conditions*

Description

Example condition table with two columns. Column 1 contains sample name, column 2 contains condition name. Use in conjuncture with count_table. Contains 10 GTEx samples each from heart and muscle.

condition_table.partial *Example table of sample names and conditions*

Description

Example condition table with two columns. Column 1 contains sample name, column 2 contains condition name. Use in conjuncture with count_table. A partial condition table containing 4 replicates each of heart and muscle. Used for testing purposes and to shorten run time.

count_filter *Filter RNA-seq expression table*

Description

`count_filter` function filters the RNA-seq count table to remove non- and low-expression genes at an defined average CPM cutoff (Default=1).

Usage

```
count_filter(count_table = NULL, cutoff = 1)
```

Arguments

`count_table`	A RNA-seq count matrix with genes on each row and samples on each column.
`cutoff`	Any gene with average CPM below this cutoff value will be filtered out.

Value

The filtered count table.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

Examples

```
# load simple GTeX example count table
# example dataset containing 1000 genes, 4 replicates
data(count_table.partial, package = "ERSSA")

#filter the counts
count_table.filtered.partial = count_filter(count_table.partial)
```

`count_table.filtered.partial` *Example count table of GTEx RNA-seq experiment, fitered by count_filter function*

Description

A filtered version of the example count_table.partial. Contains 974 genes that passed the filter. Used for testing purposes and to shorten run time.

References

https://jhubiostatistics.shinyapps.io/recount/

Melé, Marta, Pedro G. Ferreira, Ferran Reverter, David S. DeLuca, Jean Monlong, Michael Sammeth, Taylor R. Young, et al. "The Human Transcriptome across Tissues and Individuals." Science 348, no. 6235 (May 8, 2015): 660–65. https://doi.org/10.1126/science.aaa0355.

Collado-Torres, Leonardo, Abhinav Nellore, Kai Kammers, Shannon E. Ellis, Margaret A. Taub, Kasper D. Hansen, Andrew E. Jaffe, Ben Langmead, and Jeffrey T. Leek. "Reproducible RNA-Seq Analysis Using Recount2." Nature Biotechnology 35, no. 4 (April 2017): 319–21. https://doi.org/10.1038/nbt.3838.

`count_table.full` *Example count table of GTEx RNA-seq experiment*

Description

Count table generated using data downloaded from Recount2 website with processed Raw GTEx dataset. Table has samples on columns and genes on rows. Contains 10 samples each from heart and muscle.

References

https://jhubiostatistics.shinyapps.io/recount/

Melé, Marta, Pedro G. Ferreira, Ferran Reverter, David S. DeLuca, Jean Monlong, Michael Sammeth, Taylor R. Young, et al. "The Human Transcriptome across Tissues and Individuals." Science 348, no. 6235 (May 8, 2015): 660–65. https://doi.org/10.1126/science.aaa0355.

Collado-Torres, Leonardo, Abhinav Nellore, Kai Kammers, Shannon E. Ellis, Margaret A. Taub, Kasper D. Hansen, Andrew E. Jaffe, Ben Langmead, and Jeffrey T. Leek. "Reproducible RNA-Seq Analysis Using Recount2." Nature Biotechnology 35, no. 4 (April 2017): 319–21. https://doi.org/10.1038/nbt.3838.

---

count_table.partial *Example count table of GTEx RNA-seq experiment*

Description

Count table generated using data downloaded from Recount2 website that processed Raw GTEx dataset. Table has samples on columns and genes on rows. This is a partial count table containing 1000 genes and 4 replicates each from heart and muscle. Used for testing purposes and to shorten run time.

References

https://jhubiostatistics.shinyapps.io/recount/

Melé, Marta, Pedro G. Ferreira, Ferran Reverter, David S. DeLuca, Jean Monlong, Michael Sammeth, Taylor R. Young, et al. "The Human Transcriptome across Tissues and Individuals." Science 348, no. 6235 (May 8, 2015): 660–65. https://doi.org/10.1126/science.aaa0355.

Collado-Torres, Leonardo, Abhinav Nellore, Kai Kammers, Shannon E. Ellis, Margaret A. Taub, Kasper D. Hansen, Andrew E. Jaffe, Ben Langmead, and Jeffrey T. Leek. "Reproducible RNA-Seq Analysis Using Recount2." Nature Biotechnology 35, no. 4 (April 2017): 319–21. https://doi.org/10.1038/nbt.3838.

---

deg.partial *Example list of DE genes generated by edgeR*

Description

Generated using object count_table.filtered.partial and control='heart' as inputs to erssa_edger function. Default parameters used. Contains list of list of vectors. Each vector contains DE genes at a particular replicate level generated by edgeR in a given sample combination. Used for testing purposes and to shorten run time.

erssa *Empirical RNA-seq Sample Size Analysis*

Description

ERSSA is a package designed to test whether an currently available RNA-seq dataset has sufficient biological replicates to detect a majority of differentially expressed (DE) genes between two conditions. Base on the number of biological replicates available, the algorithm subsamples at step-wise replicate levels and uses existing differentially expression analysis softwares (e.g. edgeR and DESeq2) to identify the number of DE genes. This process is repeated for a given number of times with unique combinations of samples to generate a distribution of DE genes at each replicate level. Compare to existing RNA-seq sample size analysis algorithms, ERSSA does not rely on any a priori assumptions about the dataset, but rather uses an user-supplied pilot RNA-seq dataset to determine whether the current replicate level is sufficient to detect a majority of DE genes.

Usage

```
erssa(
  count_table = NULL,
  condition_table = NULL,
  DE_ctrl_cond = NULL,
  filter_cutoff = 1,
  counts_filtered = FALSE,
  comb_gen_repeat = 30,
  DE_software = "edgeR",
  DE_cutoff_stat = 0.05,
  DE_cutoff_Abs_logFC = 1,
  DE_save_table = FALSE,
  marginalPlot_stat = "median",
  TPR_FPR_stat = "mean",
  path = ".",
  num_workers = 1,
  save_log = FALSE,
  save_plot = TRUE
)
```

Arguments

count_table	A RNA-seq count matrix with genes on each row and samples on each column. If count_table has already been filtered to remove non- or low-expressing genes, then counts_filtered argument should be changed to TRUE.
condition_table	A condition table with two columns and each sample as a row. Column 1 contains sample names and Column 2 contains sample conditions (e.g. Control, Treatment).
DE_ctrl_cond	The name of control condition in the comparison. Must be one of the two conditions in the condition table.

`filter_cutoff` The average CPM threshold set for filtering genes. Default to 1.

`counts_filtered`
Boolean. Whether count table has already been filtered. Default = FALSE with the function run filtering by average CPM at the cutoff specified by filter_cutoff value.

`comb_gen_repeat`
The number of maximum unique combinations to generate at each replicate level. More tests will be performed with a bigger value, but run time also increases linearly. Default set to 30 unique combinations at maximum.

`DE_software` The name of DE analysis software to use. Current options include "edgeR" and "DESeq2". Default to "edgeR".

`DE_cutoff_stat` The cutoff in FDR or adjusted p-value used to determine whether a gene is differentially expressed. Genes with lower FDR or adjusted p-value pass the cutoff. Default = 0.05.

`DE_cutoff_Abs_logFC`
The cutoff in abs(log2FoldChange) for differential expression consideration. Genes with higher abs(log2FoldChange) pass the cutoff. Default = 1.

`DE_save_table` Boolean. The results of differential expression tests can be saved to the drive for further analysis. Default setting does not save the results to save drive space. Default = FALSE.

`marginalPlot_stat`
The statistic used for plotting of values in marginal plot function. Options include 'mean', 'median'. Default='median'.

`TPR_FPR_stat` The statistics used to summarize TPR and FPR at each replicate level in ggplot2_TPR_FPRPlot function. Options include 'mean', 'median'. Default = 'mean'.

`path` The path to which the plots and results will be saved. Default to current working directory.

`num_workers` Number of nodes to use for parallel computing the DE tests

`save_log` Boolean. Whether to save runtime parameters in log file. Defualt to false.

`save_plot` Boolean. Wehther to save ggplot2 plots to drive. Default to true.

Details

`erssa` function is a wrapper that calls several ERSSA functions in sequence. For additional description of the functions called, please see their respective manual.

For the majority of current RNA-seq analysis, RNA-seq samples are aligned to the reference, followed by running quantification packages to generate count tables. ERSSA can then take the unfiltered count table and remove non- to low-expressing genes with count_filter function. Alternatively, a filtered count table can be supplied and filtering will be skipped.

Next, unique combinations of samples at various biological replicate levels will be generated by comb_gen function and passed to a differential expression analysis software for statistical testing for DE genes. The pipeline currently supports edgeR and DESeq2, but additional software support can be easily added.

The generated differential expression results are then analyzed by several plotting functions briefly described here. ggplot2_dotplot function plots the trend in DE gene identification. ggplot2_marginPlot function plots the marginal change in the number of DE genes as replicate level increases. ggplot2_intersectPlot function plots the number of DE genes that is common across combinations. ggplot2_TPR_FPRPlot function plots the TPR and FPR of DE detection using the full dataset's list of DE gene as the ground truth. Base on insights from these plots, the user can determine whether a desirable level of DE gene discovery has been reached.

At the default setting, the results of statistical tests are not saved, only the list of DE genes is. However, all of the test results can be optionally saved for further analysis.

At default setting, only one CPU node is employed and depend on the number of tests that needs to be done, the calculations can take some time to complete. If additional nodes are available, additional nodes can be employed by specifying the num_workers argument. Parallel computing requires BiocParallel package.

The results including list of DE genes and ggplot2 objects are returned. All runtime parameters can optionally be saved in a log file named "erssa.log".

Value

A list of objects generated during the analysis is returned:

- count_table.filteredfiltered count table
- samp.name.combthe samples involved in each statistical test
- list.of.DE.geneslist of DE genes in each statistical test
- gg.dotplot.obja list of objects that can be used to recreate the dot plot. See function ggplot2_dotplot manual for more detail.
- gg.marinPlot.obja list of objects that can be used to recreate the marginal num. of DE genes plot. See function ggplot2_marginPlot manual for more detail.
- gg.intersectPlot.obja list of objects that can be used to recreate the num. of intersect genes plot. See function ggplot2_intersectPlot manual for more detail.
- gg.TPR_FPRPlot.obja list of objects that can be used to recreate the TPR vs. FPR plot. See function ggplot2_TPR_FPRPlot manual for more detail.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

Ching, Travers, Sijia Huang, and Lana X. Garmire. "Power Analysis and Sample Size Estimation for RNA-Seq Differential Expression." RNA, September 22, 2014. https://doi.org/10.1261/rna.046011.114.

Hoskins, Stephanie Page, Derek Shyr, and Yu Shyr. "Sample Size Calculation for Differential Expression Analysis of RNA-Seq Data." In Frontiers of Biostatistical Methods and Applications in Clinical Oncology, 359–79. Springer, Singapore, 2017. https://doi.org/10.1007/978-981-10-0126-0_22.

Examples

```
# load example dataset containing 1000 genes, 4 replicates and 5 comb. per
# rep. level
data(condition_table.partial, package = "ERSSA")
data(count_table.partial, package = "ERSSA")

# run erssa with the "partial" dataset, use default edgeR for DE
ssa = erssa(count_table.partial, condition_table.partial,
            DE_ctrl_cond='heart')

# run erssa with the "full" dataset containing 10 replicates per heart and
# muscle, all genes included.
# Remove comments to run
#   set.seed(1)
#   data(condition_table.full, package="ERSSA")
#   data(count_table.full, package="ERSSA")
#   ssa = erssa(count_table.full, condition_table.full, DE_ctrl_cond='heart')
```

`erssa_deseq2` *Run DESeq2 for computed sample combinations*

Description

`erssa_deseq2` function runs DESeq2 Wald test to identify differentially expressed (DE) genes for each sample combination computed by `comb_gen` function. A gene is considered to be differentially expressed by defined padj (Default=0.05) and log2FoldChange (Default=1) values. As an option, the function can also save the DESeq2 result tables as csv files to the drive.

Usage

```
erssa_deseq2(
  count_table.filtered = NULL,
  combinations = NULL,
  condition_table = NULL,
  control = NULL,
  cutoff_stat = 0.05,
  cutoff_Abs_logFC = 1,
  save_table = FALSE,
  path = "."
)
```

Arguments

`count_table.filtered`	Count table pre-filtered to remove non- to low- expressing genes. Can be the output of `count_filter` function.
`combinations`	List of combinations that is produced by `comb_gen` function.

`condition_table`
: A condition table with two columns and each sample as a row. Column 1 contains sample names and Column 2 contains sample condition (e.g. Control, Treatment).

`control`
: One of the condition names that will serve as control.

`cutoff_stat`
: The cutoff in padj for DE consideration. Genes with lower padj pass the cutoff. Default = 0.05.

`cutoff_Abs_logFC`
: The cutoff in abs(log2FoldChange) for differential expression consideration. Genes with higher abs(log2FoldChange) pass the cutoff. Default = 1.

`save_table`
: Boolean. When set to TRUE, function will, in addition, save the generated DESeq2 result table as csv files. The files are saved on the drive in the working directory in a new folder named "ERSSA_DESeq2_table". Tables are saved separately by the replicate level. Default = FALSE.

`path`
: Path to which the files will be saved. Default to current working directory.

Details

The main function calls DESeq2 functions to perform Wald test for each computed combinations generated by `comb_gen`. In all tests, the pair-wise test sets the condition defined in the object "control" as the control condition.

In typical usage, after each test, the list of differentially expressed genes are filtered by padj and log2FoldChange values and only the filtered gene names are saved for further analysis. However, it is also possible to save all of the generated result tables to the drive for additional analysis that is outside the scope of this package.

Value

A list of list of vectors. Top list contains elements corresponding to replicate levels. Each child list contains elements corresponding to each combination at the respective replicate level. The child vectors contain differentially expressed gene names.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

Love MI, Huber W, Anders S (2014). "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2." Genome Biology, 15, 550. doi: 10.1186/s13059-014-0550-8.

Examples

```
# load example filtered count_table, condition_table and combinations
# generated by comb_gen function
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(count_table.filtered.partial, package = "ERSSA")
data(combinations.partial, package = "ERSSA")
```

```
data(condition_table.partial, package = "ERSSA")

# run erssa_deseq2 with heart condition as control
deg.partial = erssa_deseq2(count_table.filtered.partial,
  combinations.partial, condition_table.partial, control='heart')
```

`erssa_deseq2_parallel` *Run DESeq2 for computed sample combinations with parallel computing*

Description

`erssa_deseq2_parallel` function performs the same calculation as `erssa_deseq2` except now employs BiocParallel to perform parallel DESeq2 calculations. This function runs DESeq2 Wald test to identify differentially expressed (DE) genes for each sample combination computed by `comb_gen` function. A gene is considered to be differentially expressed by defined padj (Default=0.05) and log2FoldChange (Default=1) values. As an option, the function can also save the DESeq2 result tables as csv files to the drive.

Usage

```
erssa_deseq2_parallel(
  count_table.filtered = NULL,
  combinations = NULL,
  condition_table = NULL,
  control = NULL,
  cutoff_stat = 0.05,
  cutoff_Abs_logFC = 1,
  save_table = FALSE,
  path = ".",
  num_workers = 1
)
```

Arguments

`count_table.filtered`	Count table pre-filtered to remove non- to low- expressing genes. Can be the output of `count_filter` function.
`combinations`	List of combinations that is produced by `comb_gen` function.
`condition_table`	A condition table with two columns and each sample as a row. Column 1 contains sample names and Column 2 contains sample condition (e.g. Control, Treatment).
`control`	One of the condition names that will serve as control.
`cutoff_stat`	The cutoff in padj for DE consideration. Genes with lower padj pass the cutoff. Default = 0.05.

`cutoff_Abs_logFC`
: The cutoff in abs(log2FoldChange) for differential expression consideration. Genes with higher abs(log2FoldChange) pass the cutoff. Default = 1.

`save_table`
: Boolean. When set to TRUE, function will, in addition, save the generated DESeq2 result table as csv files. The files are saved on the drive in the working directory in a new folder named "ERSSA_DESeq2_table". Tables are saved separately by the replicate level. Default = FALSE.

`path`
: Path to which the files will be saved. Default to current working directory.

`num_workers`
: Number of workers for parallel computing. Default=1.

Details

The main function calls DESeq2 functions to perform Wald test for each computed combinations generated by `comb_gen`. In all tests, the pair-wise test sets the condition defined in the object "control" as the control condition.

In typical usage, after each test, the list of differentially expressed genes are filtered by padj and log2FoldChange values and only the filtered gene names are saved for further analysis. However, it is also possible to save all of the generated result tables to the drive for additional analysis that is outside the scope of this package.

Value

A list of list of vectors. Top list contains elements corresponding to replicate levels. Each child list contains elements corresponding to each combination at the respective replicate level. The child vectors contain differentially expressed gene names.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

Morgan M, Obenchain V, Lang M, Thompson R, Turaga N (2018). BiocParallel: Bioconductor facilities for parallel evaluation. R package version 1.14.1, https://github.com/Bioconductor/BiocParallel.

Love MI, Huber W, Anders S (2014). "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2." Genome Biology, 15, 550. doi: 10.1186/s13059-014-0550-8.

Examples

```
# load example filtered count_table, condition_table and combinations
# generated by comb_gen function
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(count_table.filtered.partial, package = "ERSSA")
data(combinations.partial, package = "ERSSA")
data(condition_table.partial, package = "ERSSA")

# run erssa_deseq2_parallel with heart condition as control
deg.partial = erssa_deseq2_parallel(count_table.filtered.partial,
```

```
combinations.partial, condition_table.partial, control='heart',
num_workers=1)
```

---

erssa_edger *Run edgeR for computed sample combinations*

Description

erssa_edger function runs classic edgeR method to identify differentially expressed (DE) genes for each sample combination computed by comb_gen function. A gene is considered to be differentially expressed by defined FDR (Default=0.05) and logFC (Default=1) values. As an option, the function can also save the edgeR topTags tables as csv files to the drive.

Usage

```
erssa_edger(
  count_table.filtered = NULL,
  combinations = NULL,
  condition_table = NULL,
  control = NULL,
  cutoff_stat = 0.05,
  cutoff_Abs_logFC = 1,
  save_table = FALSE,
  path = "."
)
```

Arguments

count_table.filtered	Count table pre-filtered to remove non- to low- expressing genes. Can be the output of count_filter function.
combinations	List of combinations that is produced by comb_gen function.
condition_table	A condition table with two columns and each sample as a row. Column 1 contains sample names and Column 2 contains sample condition (e.g. Control, Treatment).
control	One of the condition names that will serve as control.
cutoff_stat	The cutoff in FDR for DE consideration. Genes with lower FDR pass the cutoff. Default = 0.05.
cutoff_Abs_logFC	The cutoff in abs(logFC) for differential expression consideration. Genes with higher abs(logFC) pass the cutoff. Default = 1.
save_table	Boolean. When set to TRUE, function will, in addition, save the generated edgeR TopTags table as csv files. The files are saved on the drive in the working directory in a new folder named "ERSSA_edgeR_table". Tables are saved separately by the replicate level. Default = FALSE.
path	Path to which the files will be saved. Default to current working directory.

Details

The main function calls edgeR functions to perform exact test for each computed combinations generated by `comb_gen`. In all tests, the pair-wise test sets the condition defined in the object "control" as the control condition.

In typical usage, after each test, the list of differentially expressed genes are filtered by FDR and log2FC values and only the filtered gene names are saved for further analysis. However, it is also possible to save all of the generated TopTags table to the drive for additional analysis that is outside the scope of this package.

Value

A list of list of vectors. Top list contains elements corresponding to replicate levels. Each child list contains elements corresponding to each combination at the respective replicate level. The child vectors contain differentially expressed gene names.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

Robinson MD, McCarthy DJ, Smyth GK (2010). "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data." Bioinformatics, 26(1), 139-140.

Examples

```
# load example filtered count_table, condition_table and combinations
# generated by comb_gen function
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(count_table.filtered.partial, package = "ERSSA")
data(combinations.partial, package = "ERSSA")
data(condition_table.partial, package = "ERSSA")

#run erssa_edger with heart condition as control
deg.partial = erssa_edger(count_table.filtered.partial, combinations.partial,
 condition_table.partial, control='heart')
```

`erssa_edger_parallel` *Run edgeR for computed sample combinations with parallel computing*

Description

erssa_edger_parallel function performs the same calculation as erssa_edger except now employs BiocParallel to perform parallel edgeR calculations. This function runs classic edgeR method to identify differentially expressed (DE) genes for each sample combination computed by comb_gen function. A gene is considered to be differentially expressed by defined FDR (Default=0.05) and logFC (Default=1) values. As an option, the function can also save the edgeR topTags tables as csv files to the drive.

Usage

```
erssa_edger_parallel(
  count_table.filtered = NULL,
  combinations = NULL,
  condition_table = NULL,
  control = NULL,
  cutoff_stat = 0.05,
  cutoff_Abs_logFC = 1,
  save_table = FALSE,
  path = ".",
  num_workers = 1
)
```

Arguments

count_table.filtered	Count table pre-filtered to remove non- to low- expressing genes. Can be the output of count_filter function.
combinations	List of combinations that is produced by comb_gen function.
condition_table	A condition table with two columns and each sample as a row. Column 1 contains sample names and Column 2 contains sample condition (e.g. Control, Treatment).
control	One of the condition names that will serve as control.
cutoff_stat	The cutoff in FDR for DE consideration. Genes with lower FDR pass the cutoff. Default = 0.05.
cutoff_Abs_logFC	The cutoff in abs(logFC) for differential expression consideration. Genes with higher abs(logFC) pass the cutoff. Default = 1.
save_table	Boolean. When set to TRUE, function will, in addition, save the generated edgeR TopTags table as csv files. The files are saved on the drive in the working directory in a new folder named "ERSSA_edgeR_table". Tables are saved separately by the replicate level. Default = FALSE.
path	Path to which the files will be saved. Default to current working directory.
num_workers	Number of workers for parallel computing. Default=1.

Details

The main function calls edgeR functions to perform exact test for each computed combinations generated by comb_gen. In all tests, the pair-wise test sets the condition defined in the object "control" as the control condition.

In typical usage, after each test, the list of differentially expressed genes are filtered by FDR and log2FC values and only the filtered gene names are saved for further analysis. However, it is also possible to save all of the generated TopTags table to the drive for additional analysis that is outside the scope of this package.

Value

A list of list of vectors. Top list contains elements corresponding to replicate levels. Each child list contains elements corresponding to each combination at the respective replicate level. The child vectors contain differentially expressed gene names.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

Morgan M, Obenchain V, Lang M, Thompson R, Turaga N (2018). BiocParallel: Bioconductor facilities for parallel evaluation. R package version 1.14.1, https://github.com/Bioconductor/BiocParallel.

Robinson MD, McCarthy DJ, Smyth GK (2010). "edgeR: a Bioconductor package for differential expression analysis of digital gene expression data." Bioinformatics, 26(1), 139-140.

Examples

```
# load example filtered count_table, condition_table and combinations
# generated by comb_gen function
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(count_table.filtered.partial, package = "ERSSA")
data(combinations.partial, package = "ERSSA")
data(condition_table.partial, package = "ERSSA")

# run erssa_edger_parallel with heart condition as control
deg.partial = erssa_edger_parallel(count_table.filtered.partial,
combinations.partial, condition_table.partial,
control='heart', num_workers=1)
```

ggplot2_dotplot *Plot number of DE genes*

Description

ggplot2_dotplot function plots the number of differentially expressed (DE) genes in each test.

Usage

```
ggplot2_dotplot(deg = NULL, path = ".", save_plot = TRUE)
```

Arguments

deg	The list of DE genes generated by one of ERSSA::DE_*.R scripts.
path	Path to which the plot will be saved. Default to current working directory.
save_plot	Boolean. Whether to save plot to drive. Default to TRUE.

Details

The number of DE genes are plotted as dots grouped by the associated replicate level. At each replicate level, a boxplot is drawn to mainly show the first and third quartiles as well as the median. Additionally, A red line is drawn representing the mean at each replicate level. A horizontal dashed blue line represents the number of DE genes found with all samples.

Value

A list is returned containing:

- gg_object the ggplot2 object, which can then be further customized.
- deg_dataframe the tidy table version of DEG numbers for plotting.
- full_num_DEG The number of DE genes with all samples included.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2009.

Examples

```
# load edgeR deg object generated by erssa_edger using example dataset
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(deg.partial, package = "ERSSA")

gg_dot = ggplot2_dotplot(deg.partial)
```

ggplot2_intersectPlot *Plot number of DE genes that is common across combinations*

Description

ggplot2_intersectPlot function generates and plots the list of differentially expressed (DE) genes that are found in all combinations at any particular replicate level. Often in small-scale RNA-seq experiments, the inclusion or exclusion of any paricular sample can result in a very different list of DE genes. To reduce the influence of any particular sample in the entire dataset analysis, it may be desirable to identify the list of DE genes that are enriched regardless of any specific sample(s) inclusion. This approach may be most useful analyzing the list of common DE genes at the greatest possible replicate to take advantage of the robust feature as well as employing typically the longest list of DE genes.

Usage

```
ggplot2_intersectPlot(deg = NULL, path = ".", save_plot = TRUE)
```

Arguments

deg	The list of DE genes generated by one of ERSSA::DE_*.R scripts.
path	Path to which the plot will be saved. Default to current working directory.
save_plot	Boolean. Whether to save plot to drive. Default to TRUE.

Details

Similar to how increasing number of detected DE genes can be found with more biological replicates, the list of common DE genes is expected to increase with more replicates. This eventually levels off as majority of DE genes have been found.

Value

A list is returned containing:

- gg_object the ggplot2 object, which can then be further customized.
- intersect.dataframe the tidy table version used for plotting.
- deg_dataframe the tidy table version of DEG numbers for plotting mean.
- intersect_genes list of vectors containing DE genes with vector name indicating the associated replicate level.
- full_num_DEG The number of DE genes with all samples included.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2009.

Examples

```
# load edgeR deg object generated by erssa_edger using example dataset
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(deg.partial, package = "ERSSA")

gg_intersect = ggplot2_intersectPlot(deg.partial)
```

`ggplot2_marginPlot` *Plot percent increase in detection of DE genes across replicate levels*

Description

`ggplot2_marginPlot` function plots the percent change in number of DE genes identified at each step-wise increase in replicate level.

Usage

```
ggplot2_marginPlot(deg = NULL, stat = "median", path = ".", save_plot = TRUE)
```

Arguments

`deg`	The list of DE genes generated by one of ERSSA::DE_*.R scripts.
`stat`	The statistic used for plotting. Options include 'mean', 'median'. Default='median'.
`path`	Path to which the plot will be saved. Default to current working directory.
`save_plot`	Boolean. Whether to save plot to drive. Default to TRUE.

Details

The percent change is calculated as (margin*100 lower replicate level). The results are visualized as bar plots. Either mean or median can be used for the calculation.

Value

A list is returned containing:

- gg_object the ggplot2 object, which can then be further customized.
- marg_diff.dataframe the tidy table version of percent changes for plotting.

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2009.

Examples

```
# load edgeR deg object generated by erssa_edger using example dataset
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(deg.partial, package = "ERSSA")

gg_margin = ggplot2_marginPlot(deg.partial)
```

---

ggplot2_TPR_FPRPlot *Plot TPR and FPR of each combination*

Description

ggplot2_TPR_FPR function uses the full dataset list of DE genes as the ground truth to calculate the True Positive Rate (TPR) and False Positive Rate (FPR) for each sample combinations tested. The TPR and FPR are then plotted with FPR on x-axis and TPR on y-axis similar to a ROC curve.

Usage

```
ggplot2_TPR_FPRPlot(
  deg = NULL,
  count_table.filtered = NULL,
  stat = "mean",
  path = ".",
  save_plot = TRUE
)
```

Arguments

deg	The list of DE genes generated by one of ERSSA::DE_*.R scripts.
count_table.filtered	The filtered count table with non- and low-expression genes removed. Used to identify the genes found to be non-DE.
stat	The statistics used to summarize TPR and FPR at each replicate level. Options include 'mean' and 'median'. Default = 'mean'.
path	Path to which the plot will be saved. Default to current working directory.
save_plot	Boolean. Whether to save plot to drive. Default to TRUE.

Details

Using the list of DE genes generated from the full dataset as the ground truth should be done with caution. Since the true list of DE genes is not known, this is the best alternative. This plot enables the visualization of the sensitivity and (1-specificity) of the DE gene detection at the tested replicate levels. At a sufficient replicate level, a relatively high TPR can be reached with reasonable low FPR. Such a replicate level is sufficient for most studies as additional replicates produce little improvement in TPR.

Value

A list is returned containing:

- gg_object the ggplot2 object, which can then be further customized.
- TPR_FPR.dataframe the tidy table used for plotting.
- list_TP_FP_genes lists of TP and FP genes. Follow the format of deg object with each comb_n now as a list contain two vectors, one for each of TP and FP list of DE genes

Author(s)

Zixuan Shao, <Zixuanshao.zach@gmail.com>

References

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2009.

Examples

```
# load edgeR deg object generated by erssa_edger using example dataset
# example dataset containing 1000 genes, 4 replicates and 5 comb. per rep.
# level
data(deg.partial, package = "ERSSA")
data(count_table.filtered.partial, package = "ERSSA")

gg_TPR_FPR = ggplot2_TPR_FPRPlot(deg.partial, count_table.filtered.partial)
```

Index

comb_gen, 2
combinations.partial, 2
condition_table.full, 4
condition_table.partial, 4
count_filter, 4
count_table.filtered.partial, 5
count_table.full, 5
count_table.partial, 6

deg.partial, 6

erssa, 7
erssa_deseq2, 10
erssa_deseq2_parallel, 12
erssa_edger, 14
erssa_edger_parallel, 15

ggplot2_dotplot, 18
ggplot2_intersectPlot, 19
ggplot2_marginPlot, 20
ggplot2_TPR_FPRPlot, 21