Package 'samExploreR'

July 19, 2020

Type Package

Title samExploreR package: high-performance read summarisation to count vectors with avaliability of sequencing depth reduction simulation

Version 1.13.0

Depends ggplot2,Rsubread,RNAseqData.HNRNPC.bam.chr14,edgeR,R (>= 3.4.0)

Author Alexey Stupnikov, Shailesh Tripathi and Frank Emmert-Streib

Maintainer shailesh tripathi <shailesh.tripathy@gmail.com>

Description This R package is designed for subsampling procedure to simulate sequencing experiments with reduced sequencing depth. This package can be used to anlayze data generated from all major sequencing platforms such as Illumina GA, HiSeq, MiSeq, Roche GS-FLX, ABI SOLiD and LifeTech Ion PGM Proton sequencers. It supports multiple operating systems incluidng Linux, Mac OS X, FreeBSD and Solaris. Was developed with usage of Rsubread.

Imports grDevices, stats, graphics

License GPL-3

LazyLoad yes

Suggests BiocStyle,RUnit,BiocGenerics,Matrix

biocViews ImmunoOncology, Sequencing, SequenceMatching, RNASeq, ChIPSeq, DNASeq, WholeGenome, GeneTarget, Alignment, GeneExpression, GeneticVariability, GeneRegulation, Preprocessing, GenomeAnnotation, Software

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

git_branch master

git_last_commit 4f8e793

git_last_commit_date 2020-04-27

Date/Publication 2020-07-18

R topics documented:

`df_intersect` *Example data for plotting output of samExplore function.*

Description

Example data for plotting output of samExplore function. Dataframe consists three columns, first column contains names of new Genes or exons, second column provides sequence depth and third column provide total counts for the corresponding sequence coverage.

Usage

```
data("df_intersect")
```

Format

A data frame with 1125 observations on the following 3 variables.

`Label` a character vector

`Variable` a numeric vector

`Value` a numeric vector

Details

Example data for plotting output of samExplore function. Dataframe consists three columns, first column contains names of new Genes or exons, second column provides sequence depth and third column provide total counts for the corresponding sequence coverage.

Value

Example data for plotting results.

Examples

```
data(df_intersect)
```

df_sole *Example data for plotting output of samExplore function.*

Description

Example data for plotting output of samExplore function.Dataframe consists three columns, first column contains names of new Genes or exons, second column provides sequence depth and third column provide total counts for the corresponding sequence coverage.

Usage

```
data("df_sole")
```

Format

A data frame with 1125 observations on the following 3 variables.

`Label` a character vector

`Variable` a numeric vector

`Value` a numeric vector

Details

Example data for plotting output of samExplore function.Dataframe consists three columns, first column contains names of new Genes or exons, second column provides sequence depth and third column provide total counts for the corresponding sequence coverage.

Value

Example data for plotting results.

Examples

```
data(df_sole)
```

exploreRep *exploreRep: function to explore the reproducibility*

Description

This function explores the reproducibility of analysis with annotation altering

Usage

```
exploreRep(df_d, lbl_vect, f)
```

Arguments

df_d	a dataframe containing the dataset to explore with 3 columns: label, f ratio, value to compare (e.g. number of differentially expressed genes)
lbl_vect	a vector of character strings specifing the labels for which the analysis should be run
f	A numeric value of f for which the analysis should be run

Details

exploreRep function to explore the reproducibility of the analysis with altering of annotation. It runs ANOVA test for values to compare (e.g. number of differentially expressed genes) corresponding to different Annotation labels (i.e. analysis' run for different annotation types)

This function takes as input a dataframe containing the dataset to explore.

Here is the example of the dataframe

```
...
AnnotA 0.1 13
AnnotB 0.1 101
AnnotC 0.1 36
AnnotA 0.1 13
AnnotB 0.1 101
AnnotC 0.1 36
AnnotA 0.4 40
AnnotB 0.4 153
AnnotC 0.4 62
AnnotA 0.8 71
AnnotB 0.8 203
AnnotC 0.8 160
...
```

exploreRob Thired column gives the values to compare (here number of differentially expressed genes).

exploreRep function subsets the dataset to consider only valyes for one f and runs ANOVA test for groups corresponding to annotations of interest.

Value

An output of aov function

Author(s)

Alexey Stupnikov and Shailesh Tripathi

Examples

```
#library(samExploreR)
data("df_sole")
#run ANOVA for annotation types labeled 'New, Gene' and 'New, Exon' and
#f value 0.9
exploreRep(df_sole, lbl_vect = c('New, Gene', 'Old, Gene'), f = 0.9)
```

```
#run ANOVA for annotation type labeled 'Old' and 'New' and f value 0.5
exploreRep(df_sole, lbl_vect = c('New, Gene', 'Old, Gene'), f = 0.5)
```

---

exploreRob *exploreRob: function to explore the robustness*

Description

This function explores the robustness of analysis with sequencing depth altering

Usage

```
exploreRob(df_d, lbl, f_vect)
```

Arguments

df_d	a dataframe containing the dataset to explore with 3 columns : label, f ratio, value to compare (e.g. number of differentially expressed genes)
lbl	a character string specifing the label for which the analysis should be run
f_vect	A numeric vector containing the values of f for which the analysis should be run

Details

exploreRob function to explore the robustness of the analysis with altering of sequencing depth. It runs ANOVA test for values to compare (e.g. number of differentially expressed genes) corresponding to different f ratio values (i.e. values of sequencing depth)

This function takes as input a dataframe containing the dataset to explore.

Here is the example of the dataframe

```
...
AnnotA 0.1 13
AnnotB 0.1 101
AnnotC 0.1 36
AnnotA 0.1 13
AnnotB 0.1 101
AnnotC 0.1 36
AnnotA 0.4 40
AnnotB 0.4 153
AnnotC 0.4 62
AnnotA 0.8 71
AnnotB 0.8 203
AnnotC 0.8 160
...
```

exploreRob function subsets the dataset to consider only valyes for one type of annotation and runs ANOVA test for groups corresponding to f values of interest.

Value

An output of aov function

Author(s)

Alexey Stupnikov and Shailesh Tripathi

Examples

```
#library(samExploreR)
data("df_sole")
#run ANOVA for annotation type labeled 'New, Gene' and f values 0.9, 0.95
exploreRob(df_sole, lbl = 'New, Gene', f_vect = c(0.9, 0.95))


#run ANOVA for annotation type labeled 'Old' and f values 0.5, 0.95
exploreRob(df_sole, lbl = 'Old, Gene', f_vect = c(0.5, 0.95))
```

---

plotsamExplorer *Plots the results of output dataframe object.*

Description

Boxplot results between sequence-depth and number of differentially expressed genes.

Usage

```
plotsamExplorer(dat, save = FALSE, filename = NULL, p.depth = 0.9,
font.size = 3.5, anova = TRUE, x.lab=NULL, y.lab=NULL, leg.lab=NULL)
```

Arguments

dat	is a dataframe object, which consists three columns strictly labelled as: "Label", "Variable" and "Value".
save	is a logical value to save plot as a pdf.
filename	is a character to assign filename, if a user want to save the plot.
p.depth	is a numeric value for anova test to be performed for number differentially expressed genes of different sequence-depths.
font.size	is a numeric value to set font size of the plot.
anova	is a logical value for anova test to be performed for number differentially expressed genes of different sequence-depths.
x.lab	is a string value to assign a label for x-axis.
y.lab	is a string value to assign a label for y-axis.
leg.lab	is a string vector assigns lables for legends in the plot.

Value

Generates a plot in a pdf format.

Author(s)

Frank-Emmert Streib, Shailesh Tripathi, Aleksei sputnikov

Examples

```
data("df_sole")
data("df_intersect")

plotsamExplorer(df_sole,save=TRUE,filename="ss",p.depth=.9,
font.size=4, anova=TRUE)
plotsamExplorer(df_intersect,save=TRUE,filename="ss",p.depth=.9,
font.size=4, anova=FALSE)
```

`samExplore` *samExplore:*

Description

samExplore: This function assigns mapped sequencing reads to genomic features and simulates a sample with reduced sequencing depth

Usage

```
samExplore(..., subsample_d=1, N_boot=1,
    countboot=c("all","Assigned", "Unassigned_Ambiguity",
    "Unassigned_MultiMapping",   "Unassigned_NoFeatures",
    "Unassigned_Unmapped",       "Unassigned_MappingQuality",
    "Unassigned_FragmentLength", "Unassigned_Chimera",
    "Unassigned_Secondary",      "Unassigned_Nonjunction",
    "Unassigned_Duplicate" ))
```

Arguments

`...`	These are the same arguments of `featureCounts` function of `Rsubread` package, for more details check `featureCounts` function.
`subsample_d`	numeric value which describes fraction of reads to be remained in subsampling.
`N_boot`	integer value for number of resample procedures to be run.
`countboot`	is a character vector which contains following options: `all`,`Assigned`, `Unassigned_Ambiguity`, `Unassigned_MultiMapping`, `Unassigned_NoFeatures`, `Unassigned_Unmapped`, `Unassigned_MappingQuality`, `Unassigned_FragmentLength`, `Unassigned_Chimera`, `Unassigned_Secondary`, `Unassigned_Nonjunction`, `Unassigned_Duplicate` A user can select any of theses options for resampling if user selects `all` then the resampling procedure will consider all `assigned` and `unassigned` reads. If a user selects `Assigned` option then resampling procedure will consider `Assigned` reads only for resampling. If a user selects any other option it will consider those unmapped reads along with `Assigned` reads. A user can selects more than one choices and input as a vector

Details

samExplore See featureCounts for details. Output is a list objects which has three components. 1) "bootres": is a list object of size of input files, each list object contains a resampling matrix of features.

2) "target.size": it is a numeric vector contains total feature counts of a certain sequence depth for each input file.

3) "feature main": returns a list object which is the ouptput of 'featureCounts' function of Rsubread package.

Value

returns a list object.

Examples

```
# Simulate a sample with sequencing depth 80% of initial for SAM format
# single-end reads using built-in RefSeq annotation for hg19:
#### Consider all mapped and unmapped reads for resampling##
inpf <- RNAseqData.HNRNPC.bam.chr14_BAMFILES
res1 <- samExplore(files=inpf,annot.inbuilt="hg19", subsample_d = 0.8)

#### Consider Assigned and Unassigned Unmapped reads for resampling##
res2 <- samExplore(inpf, N_boot=10, subsample_d=.8,
    countboot=c("Assigned","Unassigned_Unmapped"))

#### Consider only Assigned reads for resampling##
res3 <- samExplore(inpf, N_boot=10, subsample_d=.8,
    countboot="Assigned")
```

Index