Package ‘rmRNAseq’

November 8, 2019

Type Package

Title RNA-Seq Analysis for Repeated-Measures Data

Version 0.1.0

Description
A differential expression analysis method for RNA-seq data from repeated-measures design using general linear model framework and parametric bootstrap inference. The method ac-
counts for
the dependence of gene expression levels due to the repeated-
measures through continuous autoregressive
correlation structure. The method is described in Chapter 4 of
Nguyen (2018) <https://lib.dr.iastate.edu/cgi/viewcontent.cgi?article=7433&context=etd>.

Depends R (>= 3.3)

biocViews

Imports AUC (>= 0.3.0), Biobase, DESeq2, edgeR (>= 3.16.5), graphics,
limma (>= 3.30.13), MASS (>= 7.3-45), Matrix (>= 1.2-8),
methods, nlme (>= 3.1-131), parallel, reshape (>= 0.8.6),
rlang, splines, splineTimeR, statmod (>= 1.4.29), stats

License GPL (>= 2)

LazyData true

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, testthat

Encoding UTF-8

NeedsCompilation no

Author Yet Nguyen [aut, cre],
Dan Nettleton [aut],
Charity Gordon Law Smyth [ctb] (The authors of limma::voom, which was
modified slightly leading to my_voom),
Agata Michna [ctb] (The author of splineTimeR::splineDiffExprs, which
was modified slightly leading to my_splineDiffExprs)

Maintainer Yet Nguyen <tienyettoan@gmail.com>

Repository CRAN

Date/Publication 2019-11-08 13:10:02 UTC

R topics documented:

`covset` *Covariate Set Associated with RFI RNA-seq*

Description

This is a covariate set containing variables/measurements of the pigs that RNA-seq data were obtained.

Usage

```
covset
```

Format

`covset` is a dataframe with 32 rows and 11 columns, correspinding to 32 RNA-seq sample from 8 pigs at 4 times. 11 columns include

time a continuous vector of mapping time points

timef a factor of original time points

line a factor of original line of each pig

ear ear id of each pig

line2, time2, ... the columns corresponding to design matrix transformation such that each column is a corresponding test statistics.

Examples

```
data(covset)
dim(covset)
colnames(covset)
head(covset)
```

`dat` *RFI RNA-seq Data*

Description

This is the RFI RNA-seq data motivating our paper.

Usage

```
dat
```

Format

The dataset has 11911 rows and 32 columns corresponding to 32 RNA-seq samples from 8 pigs, each pig has 4 RNA-seq samples measured repeatedly at 0, 2, 6, 24 hours after LPS treatments.

Examples

```
data(dat)
dim(dat)
head(dat)
```

DESeq2Fit *Analysis of LPS RFI RNA-seq data Using DESeq2*

Description

This function analyzes the LPS RFI RNA-seq data and simulated datasets using DESeq, the Quasi-likelihood F-test in DESeq2 package.

Usage

```
DESeq2Fit(counts, design, Effect)
```

Arguments

counts	a matrix of count data.
design	a design matrix.
Effect	the effect used to simulate data, either line2, or time. This effect is considered as the main factor of interest where the status of DE and EE genes was specified. @param covset a data frame contain covariate set.

Value

a list of 4 components

fit	output of DESeq function.
pv	a vector of p-values of the test for significant of Effect.
qv	a vector of q-values corresponding to the pv above.

Examples

```
data(dat)
data(design)
counts <- dat[1:100,]
design <- design
Effect <- "line2"
DESeq2Fitout <- rmRNAseq:::DESeq2Fit(counts, design, Effect)
names(DESeq2Fitout)
```

design *Reparameterized Design Matrix*

Description

This is a reparameterized matrix from the original design matrix that includes `time,line,time x line` such that each column of the reparameterized design matrix corresponds to a test of interest such as line main effect, `time2 -time0`, `time6 -time0`, `time24 -time0`, and interaction effect of `timei -time0` with `line`. When using this reparameterized design matrix in the analysis, for example, regression coefficients corresponding to its second column `(line2)` represents the `Line` main effect.

Usage

```
design
```

Format

A design matrix with 8 columns `Intercept,line2,time2,time6,time24,linetime2,linetime6,linetime24`.

Examples

```
data(design)
colnames(design)
head(design)
```

edgeRFit *Analysis of the RFI RNA-seq data Using edgeR*

Description

This function analyzes the RFI RNA-seq data and simulated datasets using `glmQLFTest`, the Quasi-likelihood F-test in `edgeR` package.

Usage

```
edgeRFit(counts, design, Effect)
```

Arguments

`counts`	a matrix of count data.
`design`	a design matrix.
`Effect`	the effect used to simulate data, either line2, or time. This effect is considered as the main factor of interest where the status of DE and EE genes was specified.

Value

a list of 4 components

`fit`	output of `glmQLFit` function.
`pv`	a vector of p-values of the test for significant of `Effect`.
`qv`	a vector of q-values corresponding to the `pv` above.

Examples

```
data(dat)
data(design)
counts <- dat[1:100,]
design <- design
Effect <- "line2"
edgeRout <- rmRNAseq:::edgeRFit(counts, design, Effect)
names(edgeRout)
```

`estimate.m0` *Estimate Number of True Null Hypotheses Using Histogram-based Method*

Description

This function estimates the number of true null hypotheses given a vector of p-values using the method of Nettleton et al. (2006) JABES 11, 337-356. The estimate obtained is identical to the estimate obtained by the iterative procedure described by Mosig et al. Genetics 157:1683-1698. The number of p-values falling into B equally sized bins are counted. The count of each bin is compared to the average of all the bin counts associated with the current bins and all bins to its right. Working from left to right, the first bin that has a count less than or equal to the average is identified. That average is multiplied by the total number of bins to obtain an estimate of m0, the number of tests for which the null hypothesis is true.

Usage

```
estimate.m0(p, B = 20)
```

Arguments

`p`	a numerical vector of p-value
`B`	number of bin

Value

The function returns an estimate of m0, the number of tests for which the null hypothesis is true.

Author(s)

Dan Nettleton <dnett@iastate.edu>

References

1. Dan Nettleton, J. T. Gene Hwang, Rico A. Caldo and Roger P. Wise. Estimating the Number of True Null Hypotheses from a Histogram of p Values. Journal of Agricultural, Biological, and Environmental Statistics Vol. 11, No. 3 (Sep., 2006), pp. 337-356.

Examples

```
data(res)
p <- res$pqvalue$pv$line2
m0 <- rmRNAseq:::estimate.m0(p)
m0
```

`glsCAR1` *Fit General Linear Model with* `corCAR1` *Correlation Structure for One Gene*

Description

This function fits `gls` model with REML estimation method, `corCAR1` correlation structure for one gene in a RNA-seq repeated measures data, where data is log-transformed output from `voom`.

Usage

```
glsCAR1(d)
```

Arguments

`d`	a data frame containing several columns. The first 4 columns are `y:` a vector of log-counts (obtained by `voom`), `Subject:` a vector of subject/experimental units where repeated measures are obtained (can be either numeric or factor), `Time:` a vector of time points (continuous, since we fit `corCAR1`), `w:` weights to put in gls model, this is the inverse of weights obtained by `voom` The other columns are exactly the same as design matrix.

Value

Output is a vector including the following components

`aic`	AIC of the fitted model.
`s2`	estimate of error variance.
`rho`	correlation parameter in `corCAR1` correlation matrix.
`fixed`	fixed effects (estimates of regression parameters).
`varbeta`	the estimates of variance of fixed effects, just include lower part and diagonal part of the variance-covariance matrix.

Examples

```
data(res)
data(design)
data(covset)
d <- data.frame(cbind(y = res$ori.res$v$E[1,] ,Subject = covset$ear,
Time = covset$time, w = 1/res$ori.res$v$weights[1,], design))
glsout <- rmRNAseq:::glsCAR1(d)
glsout
```

`glsCAR1_loglik` *Calculate REML Log-Likelihood of glsCAR1 model for each gene*

Description

This function calculates log-likelihood value of glsCAR1 for each gene using `voom` data.

Usage

```
glsCAR1_loglik(d)
```

Arguments

`d`	a data frame containing several columns. The first 4 columns are `y:` a vector of log-counts (obtained by `voom`), `Subject:` a vector of subject/experimental units where repeated measures are obtained (can be either numeric or factor), `Time:` a vector of time points (continuous, since we fit `corCAR1`), `w:` weights to put in gls model, this is the inverse of weights obtained by `voom` The other columns are exactly the same as design matrix.

Value

reml log-likelihood value

Examples

```
data(res)
data(design)
data(covset)
d <- data.frame(cbind(y = res$ori.res$v$E[1,] ,Subject = covset$ear,
Time = covset$time, w = 1/res$ori.res$v$weights[1,], design))
glsloglikout <- rmRNAseq:::glsCAR1_loglik(d)
glsloglikout
```

glsSymm — *Fit General Linear Model with* `corSymm` *Correlation Structure for One Gene*

Description

This function fits `gls` model with REML estimation method, `corSymm` unstructured correlation for one gene in a RNA-seq repeated measures data, where data is the log-transformed counts obtained from `voom`.

Usage

```
glsSymm(d)
```

Arguments

`d`	a data frame containing several columns. The first 4 columns are `y:` a vector of log-counts (obtained by `voom`), `Subject:` a vector of subject/experimental units where repeated measures are obtained (can be either numeric or factor), `Time:` a vector of time points (continuous, since we fit `corSymm`), `w:` weights to put in gls model, this is the inverse of weights obtained by `voom` The other columns are exactly the same as design matrix.

Value

Output is a vector including the following components

`aic`	AIC of the fitted model.
`s2`	estimate of error variance.
`rho`	correlation parameter in `corSymm` correlation matrix.
`fixed`	fixed effects (estimates of regression parameters).
`varbeta`	the estimates of variance of fixed effects, just include lower part and diagonal part of the variance-covariance matrix.

Examples

```
data(res)
data(design)
data(covset)
d <- data.frame(cbind(y = res$ori.res$v$E[1,] ,Subject = covset$ear,
Time = covset$time, w = 1/res$ori.res$v$weights[1,], design))
glsout <- rmRNAseq:::glsSymm(d)
glsout
```

jabes.q *Q-value Using Histogram-based Method*

Description

This function computes q-values using the approach of Nettleton et al. (2006) JABES 11, 337-356.

Usage

```
jabes.q(p, B = 20)
```

Arguments

p	a numerical vector of p-value
B	number of bin

Value

The function returns a q-value vector of the input p-value vector.

Author(s)

Dan Nettleton <dnett@iastate.edu>

References

1. Dan Nettleton, J. T. Gene Hwang, Rico A. Caldo and Roger P. Wise. Estimating the Number of True Null Hypotheses from a Histogram of p Values. Journal of Agricultural, Biological, and Environmental Statistics Vol. 11, No. 3 (Sep., 2006), pp. 337-356.

Examples

```
data(res)
p <- res$pqvalue$pv$line2
q <- rmRNAseq:::jabes.q(p)
sum(q <= .05)
```

`myvoom` *myvoom function*

Description

This function modifies the original `voom` function to obtain the fitted function f of the lowess fit.

Usage

```
myvoom(counts, design = NULL, lib.size = NULL,
  normalize.method = "none", span = 0.5, plot = FALSE,
  save.plot = FALSE, ...)
```

Arguments

`counts`	a numeric `matrix` containing raw counts, or an `ExpressionSet` containing raw counts, or a `DGEList` object. Counts must be non-negative and NAs are not permitted.
`design`	design matrix with rows corresponding to samples and columns to coefficients to be estimated. Defaults to the unit vector meaning that samples are treated as replicates.
`lib.size`	numeric vector containing total library sizes for each sample. Defaults to the normalized (effective) library sizes in `counts` if `counts` is a `DGEList` or to the columnwise count totals if `counts` is a matrix.
`normalize.method`	the microarray-style normalization method to be applied to the logCPM values (if any). Choices are as for the `method` argument of `normalizeBetweenArrays` when the data is single-channel. Any normalization factors found in `counts` will still be used even if `normalize.method="none"`.
`span`	width of the lowess smoothing window as a proportion.
`plot`	logical, should a plot of the mean-variance trend be displayed?
`save.plot`	logical, should the coordinates and line of the plot be saved in the output?
`...`	other arguments are passed to `lmFit`.

Value

the same output as `voom` plus the fitted function f of the lowess fit of the `link[limma]{voom}` method.

my_splineDiffExprs *Differential expression analysis based on natural cubic spline regression models for time-course data*

Description

This function is a modified version of splineDiffExprs function that allows data input as log-transformed counts taking into account the voom-weights and 75th quantile as library size. The function compares time dependent behaviour of genes in two different groups. Applying empirical Bayes moderate F-statistic on differences in coefficients of fitted natural cubic spline regression models, differentially expressed in time genes are determined. The function is a wrapper of other R-functions to simplify differential expression analysis of time-course data.

Usage

```
my_splineDiffExprs(eSetObject, df, cutoff.adj.pVal = 1, reference,
  intercept = TRUE, voom_method = FALSE)
```

Arguments

`eSetObject`	`ExpressionSet` object of class `ExpressionSet` containing log-ratios or log-values of expression for a series of microarrays
`df`	number of degrees of freedom
`cutoff.adj.pVal`	Benjamini-Hochberg adjusted p-value cut-off
`reference`	character defining which treatment group should be considered as reference
`intercept`	if `TRUE`, F-test includes all parameters; if `FALSE`, F-test includes shape parameters only; default is `TRUE`
`voom_method`	logical value TRUE or FALSE. TRUE when using log-transformed counts, voom-weight, and 75th quantile library size

Value

a list of 4 components

`fit`	output of `lmFit` function.
`pv`	a vector of p-values of the test for line effect
`qv`	a vector of q-values corresponding to the pv above.
`diffExprs`	A data.frame with rows defining names/IDs of differentially expressed genes

Examples

```
library(rmRNAseq)
data(design)
data(covset)
data("resSymm")
EE <- 40
DE <- 10
n_idx <- sample(nrow(resSymm$ori.res$v), size = EE + DE)
v <- resSymm$ori.res$v[n_idx,]
newlm <- resSymm$ori.res$newlm[n_idx,]
BetaMat <- data.matrix(newlm[grep("fixed.", names(newlm))])
BetaMat[1:EE, 2] <- 0
BetaMat[(EE+1):(EE+DE), 2] <- BetaMat[(EE+1):(EE+DE), 2]*4
Sigma2Vec <- newlm$s2_shrunken
RhoVec <- data.matrix(newlm[grep("rho.", names(newlm))])
WeightMat <- v$weights
lib.size <- v$targets$lib.size
nrep <- 1
Subject <- covset$ear
Time <- covset$time
counts <- rmRNAseq:::sc_Symm(BetaMat, Sigma2Vec, RhoVec, WeightMat,
lib.size, design, Subject, Time,nrep)
inData <- counts
colnames(inData) <- paste(covset$line, Subject, covset$timef,sep="_")
design <- rmRNAseq::design
design1 <- data.frame(row.names=colnames(inData),
"SampleName"=colnames(inData),
"Time"=covset$time,"Treatment"=covset$line)
phenoData <- new("AnnotatedDataFrame",data=design1)
data <- Biobase::ExpressionSet(assayData=as.matrix(inData),phenoData=phenoData)
diffExprs <- rmRNAseq:::my_splineDiffExprs(eSetObject = data, df = 3,cutoff.adj.pVal = 1,
                            reference = "L",intercept = TRUE, voom_method = FALSE)
rmRNAseq:::pauc_out(diffExprs$pv, EE , DE)
diffExprs2 <- rmRNAseq:::my_splineDiffExprs(eSetObject = data, df = 3,cutoff.adj.pVal = 1,
reference = "L",intercept = TRUE, voom_method = TRUE)
rmRNAseq:::pauc_out(diffExprs2$pv, EE , DE)
```

`NewTimeEst` *Estimate New Time Points*

Description

This function estimate new time points to fit the glsCAR1 for the `voom` transformed data. Note that this function is very specific for this dataset, with only 4 time points. If there are more than 4 time points, the method needs to be updated.

Usage

```
NewTimeEst(v, Subject, Time, TimeMinOut, ncores)
```

Arguments

`v`	output of `voom` function.
`Subject`	a vector of subjects or experimental units.
`Time`	a vector of time points.
`TimeMinOut`	output from the `TimeMin` function
`ncores`	number of cores for embarrassingly parallel procedure. Default value of `ncores` is 1.

Value

New time points.

Examples

```
data(res)
data(covset)
v <- res$ori.res$v
v$E <- v$E[1:2,]
v$weights <- v$weights[1:2,]
Subject <- covset$ear
Time <- covset$time
ncores <- 1
NewTimeOut <- rmRNAseq:::NewTimeEst(v, Subject, Time, ncores)
NewTimeOut
```

`pauc_out` *Evaluation of Differential Expression Analysis Methods*

Description

This function calculates FDR (false discovery rate), pauc (partial area under ROC curve), when we know the vector of p-values obtained from a particular differential expression analysis method and the true status of each gene. The function requires that the first EE genes are true EE, and the last DE genes are true DE. This requirement can be fulfilled by reorder the rows of gene expression data set.

Usage

```
pauc_out(p, EE, DE)
```

Arguments

`p`	a vector of p-values.
`EE`	number of EE genes (the first EE genes in p-value vector p).
`DE`	number of DE genes (the last DE genes in p-value vector p).

Value

a vector including V (the number of false positives), R (the number of declared positives), FDR (the false discovery rate), pauc (the partial area under ROC curve with respect to false positive rate fpr less than or equal to a specified level), and auc. Here we consider 3 fpr: 0.05, 0.10, 0.20. So the output includes these 15 elements and the total auc.

Examples

```
set.seed(1)
EE <- 1000
DE <- 500
p1 <- runif(EE)
p2 <- rbeta(DE, shape1 = .5, shape2 = 1)
p <- c(p1, p2)
rmRNAseq:::pauc_out(p, EE, DE)
```

`res` *Data Containing Results of Our Proposed Method Applying to RFI RNA-seq data*

Description

This data set contains outputs of `voom`, `glsCAR1` and pqvalue of the tests for interested contrasts when analyzing the RFI RNA-seq data using our proposed method.

Usage

```
res
```

Format

A list with 3 components

ori.res a list consisting of 2 components: `v`, which is a voom output including several components; `newlm`, which is the output from `glsCAR1`

pqvalue a list of 2 components `pv` and `qv`. `pv` is a data frame of pvalues for each test such as `line2 time`; `qv` is a data frame of corresponding q-values of the above p-values, calculated using the method by Nettleton 2006.

Examples

```
data(res)
names(res)
dim(res$ori.res$v)
colnames(res$ori.res$v)
colnames(res$ori.res$newlm)
colnames(res$pqvalue$pv)
colnames(res$pqvalue$qv)
```

resSymm *Data Containing results when analyzing RFI using TC_Symm*

Description

This dataset contains the output of TC_Symm for 1000 genes when applying the TC_Symm to the RFI dataset.

Usage

```
resSymm
```

Format

An object of class `list` of length 2.

Examples

```
data(resSymm)
names(resSymm)
dim(resSymm[[1]]$v)
colnames(resSymm[[1]]$newlm)
colnames(resSymm[[2]]$pv)
colnames(resSymm[[2]]$qv)
```

sc_CAR1 *Simulating Count Data From The Output of Real Data Analysis*

Description

This function generates bootstrap samples using parametric bootstrap method.

Usage

```
sc_CAR1(BetaMat, Sigma2Vec, RhoVec, WeightMat, lib.size, design, Subject,
  Time, nrep)
```

Arguments

`BetaMat`	a matrix of estimates of regression coefficients.
`Sigma2Vec`	a vector of shrinkage estimates of error variances.
`RhoVec`	a vector of estimates of correlation.
`WeightMat`	a matrix of weights of all genes obtaining from voom.
`lib.size`	library size in voom method, we choose .75 quantile as library size.
`design`	a design matrix.

`Subject`	a vector of subjects/experimental units.
`Time`	a vector of time points.
`nrep`	simulation iteration.

Value

a matrix of count data that has nrow(BetaMat) rows and nrow(design) columns.

References

Yet Nguyen, Dan Nettleton, 2019. rmRNAseq: RNA-seq Analysis for Repeated-measures Data.

Examples

```
data(res)
v <- res$ori.res$v[1:50,]
newlm <- res$ori.res$newlm[1:50,]
BetaMat <- data.matrix(newlm[grep("fixed.", names(newlm))])
Sigma2Vec <- newlm$s2_shrunken
RhoVec <- newlm$rho
WeightMat <- v$weights
lib.size <- v$targets$lib.size
nrep <- 1
Subject <- covset$ear
Time <- covset$time
simcounts <- rmRNAseq:::sc_CAR1(BetaMat, Sigma2Vec, RhoVec, WeightMat,
lib.size, design, Subject, Time)
dim(simcounts)
```

sc_Symm *Simulating Count Data From The Output of Real Data Analysis (corSymm)*

Description

This function generates bootstrap samples using parametric bootstrap method.

Usage

```
sc_Symm(BetaMat, Sigma2Vec, RhoVec, WeightMat, lib.size, design, Subject,
  Time, nrep)
```

Arguments

`BetaMat`	a matrix of estimates of regression coefficients.
`Sigma2Vec`	a vector of shrinkage estimates of error variances.
`RhoVec`	a vector of estimates of correlation.
`WeightMat`	a matrix of weights of all genes obtaining from voom.
`lib.size`	library size in voom method, we choose .75 quantile as library size.
`design`	a design matrix.
`Subject`	a vector of subjects/experimental units.
`Time`	a vector of time points.
`nrep`	simulation iteration.

Value

a matrix of count data that has nrow(BetaMat) rows and nrow(design) columns.

Examples

```
data(resSymm)
v <- resSymm$ori.res$v[1:20,]
newlm <- resSymm$ori.res$newlm[1:20,]
BetaMat <- data.matrix(newlm[grep("fixed.", names(newlm))])
Sigma2Vec <- newlm$s2_shrunken
RhoVec <- data.matrix(newlm[grep("rho.", names(newlm))])
WeightMat <- v$weights
lib.size <- v$targets$lib.size
nrep <- 1
Subject <- covset$ear
Time <- covset$time
simcounts <- rmRNAseq:::sc_Symm(BetaMat, Sigma2Vec, RhoVec, WeightMat,
lib.size, design, Subject, Time,nrep)
dim(simcounts)
```

`shrink.phi` *Shrinkaged Estimates of Error Variance*

Description

This function implements Smyth's approach (empirical bayes estimate of error variance of genes, limma paper 2004).

Usage

```
shrink.phi(phi.hat, den.df)
```

Arguments

`phi.hat`	a numerical vector of the estimated of error variance of all genes.
`den.df`	denominator degree of freedom associated with the estimated variances phi.hat (=n sample -rank(design)).

Value

a list of 3 components

`phi.shrink`	vector of shrinkaged estimates of variance.
`d0`	estimated prior degree of freedom used inthe shrinkage procedure.
`phi0`	estimated prior variance used in the shrinkage procedure.

Examples

```
phi.hat <- rchisq(1000, 1)
den.df <- 2
shrinkout <- rmRNAseq:::shrink.phi(phi.hat, den.df)
hist(shrinkout$phi.shrink)
```

`TC_CAR1` *RNA-seq Analysis for Repeated-measures Data*

Description

This function implements our parametric bootstrap to analyze repeated measures RNA-seq data.

Usage

```
TC_CAR1(counts, design, Subject, Time, C.matrix, Nboot = 100,
  ncores = 1, print.progress = FALSE, saveboot = FALSE)
```

Arguments

`counts`	a matrix of RNA-seq counts.
`design`	a design matrix.
`Subject`	a vector of subjects or experimental units.
`Time`	a vector of time points.
`C.matrix`	is a list of matrix Ci in testing H0: Ci*beta = 0.
`Nboot`	number of bootstrap replicates, default is 100.
`ncores`	number of cores for embarrassingly parallel procedure. Default value of `ncores` is 1.
`print.progress`	logical indicator, TRUE or FALSE, to print the progress.
`saveboot`	TRUE or FALSE to save or not save bootstrap output

Value

a list of 3 components

`ori.res`	a list of 2 components `v`: voom's output, `newlm`: output from `voomgls_CAR1`.
`boot.res`	a list of `Nboot` components, each component is the output of `voomgls_CAR1` when apply this function to the corresponding bootstrap sample.
`pqvalue`	a list 2 components: `pv`: a matrix of p-values of the tests construted in C.matrix `qv`: matrix of q-values obtaining from using Nettleton 2006 paper approach, using `jabes.q` function.

References

Yet Nguyen, Dan Nettleton, 2019. rmRNAseq: RNA-seq Analysis for Repeated-measures Data.

Examples

```
# This example shows how to implement the method using LPS RFI data.
data(dat)
data(design)
data(covset)
Subject <- covset$ear
Time <- covset$time
Nboot <- 2  # for real data analysis, use Nboot at least 100
ncores <- 1 # for real data analysis and if the computer allows, increase ncores to save time
print.progress <- FALSE
saveboot <- FALSE
counts <- dat[1:3,]
C.matrix <- list()
# test for Line main effect
C.matrix[[1]] <- limma::makeContrasts(line2, levels = design)
# test for Time main effect
C.matrix[[2]] <- limma::makeContrasts(time2, time6, time24, levels = design)
names(C.matrix) <- c("line2", "time")
TCout <- rmRNAseq:::TC_CAR1(counts, design, Subject, Time, C.matrix,
Nboot, ncores, print.progress, saveboot)
names(TCout)
TCout$pqvalue$pv
TCout$pqvalue$qv
```

`TC_CAR1_sc` *A Wrap Function to analyze a Simulated Data - All Cases*

Description

This function does the following: 1. Generating a simulated counts data set consisting of EE genes, and DE genes with respect to some contrast from a ideal case, i.e., the counts are generated from corCAR1 structure, or misspecified case, i.e., the counts are generated from corSymm structure; analyzing this simulated data set using methods: `TC_CAR1`, `voomlimmaFit`, `edgeRFit` , `DESeq2Fit`.

Usage

```
TC_CAR1_sc(RFIanalysis, scenario, EE, DE, C.matrix, Subject, Time, Nboot,
  nrep, ncores, name_dir_sim = NULL, print.progress = FALSE,
  saveboot = FALSE)
```

Arguments

`RFIanalysis`	the output from RFI RNA-seq dataset. In the ideal simulation case, it is res from TC_CAR1, in the misspecified case, it is res from TC_Symm
`scenario`	either 2- 'Symm' or 1- 'CAR1'
`EE`	number of EE genes
`DE`	number of DE genes
`C.matrix`	is a list of matrix Ci in testing H0: Ci*beta = 0.
`Subject`	a vector of subjects or experimental units.
`Time`	a vector of time points.
`Nboot`	number of bootstrap replicates, default is 100.
`nrep`	index of sim replicate
`ncores`	number of cores for embarrassingly parallel procedure. Default value of `ncores` is 1.
`name_dir_sim`	name of directory to contain the output and result of this function
`print.progress`	`TRUE` or `FALSE`, printing the process or not.
`saveboot`	`TRUE` or `FALSE` to save or not save bootstrap output

Value

R, V, FDR, PAUC, and AUC of all 7 methods (2 oracles with unshrunken and shrunken) with FPR = 0.05, 0.10, 0.20 for each S, R, V, FDR, PAUC.

Examples

```
data(res)
data(resSymm)
data(design)
data(covset)
RFIanalysis <- list(CAR1 = res, Symm = resSymm)
C.matrix <- list()
# test for Line main effect
C.matrix[[1]] <- limma::makeContrasts(line2, levels = design)
names(C.matrix) <- c("line2")
scenario <- 1; EE <- 3; DE <- 2; ncores <- 1; Subject <- covset$ear;
Time <- covset$time; Nboot <- 2; nrep <- 1;
name_dir_sim <- NULL
print.progress <- FALSE; saveboot <- FALSE;
TC_Symm_scOut <- rmRNAseq:::TC_CAR1_sc(RFIanalysis, scenario,  EE, DE,  C.matrix,
Subject, Time,  Nboot,  nrep, ncores, name_dir_sim , print.progress, saveboot)
names(TC_Symm_scOut)
```

teststat — *Calculating F-Type Statistics To Test a General Linear Hypothesis*

Description

This function is to calculate F-type test statistics for a general linear hypothesis for each of G genes.

Usage

```
teststat(C.matrix, beta0 = NULL, regression.output, ncores = 1)
```

Arguments

`C.matrix`	is a list of matrix Ci in testing H0: Ci*beta = 0.
`beta0`	vector of the hypothesized value of beta, usually, beta0 is a 0 vector. The default option `beta0` = `NULL` means that `beta0` is a vector of 0.
`regression.output`	this is a data.frame containing the output of `glsCAR1` function for all G genes.
`ncores`	number of cores for embarrassingly parallel procedure. Default value of `ncores` is 1.

Value

A matrix of dimension G X length(C.matrix) of F-similar test statistics

Examples

```
data(design)
beta0 <- NULL
regression.output <- res$ori.res$newlm[1:50,]
ncores <- 1
C.matrix <- list()
C.matrix[[1]] <- limma::makeContrasts(line2, levels = design)
C.matrix[[2]] <- limma::makeContrasts(time2, time6, time24, levels = design)
names(C.matrix) <- c("line2","time")
teststatout <- rmRNAseq:::teststat(C.matrix, beta0, regression.output, ncores)
head(teststatout)
```

TimeMin *Identify Time Points Mapping to 0 and 1*

Description

This function is to identify which time points mapped to 0 and 1 based on the estimated correlations of observations between all pairs of time points. The correlation parameters are estimateed by fitting `voomgls_Symm` to the `voom` transformed data.

Usage

```
TimeMin(v, Subject, Time, ncores)
```

Arguments

`v`	output of `voom` function.
`Subject`	a vector of subjects or experimental units.
`Time`	a vector of time points.
`ncores`	number of cores for embarrassingly parallel procedure. Default value of `ncores` is 1.

Value

a vector of 2 components correspinding to the two time points that are mapping to 0, 1, respectively.

Examples

```
data(res)
data(covset)
v <- res$ori.res$v
v$E <- v$E[1:2,]
v$weights <- v$weights[1:2,]
Subject <- covset$ear
Time <- covset$time
ncores <- 1
TimeMinout <- rmRNAseq:::TimeMin(v, Subject, Time, ncores)
TimeMinout
```

varbeta *Recovering a Symmetric Matrix from Its Lower Triangular Matrix*

Description

This function recovers a symmetric matrix from the vector consisting of its lower triangular elements.

Usage

```
varbeta(lower.tri.vector, diag = TRUE)
```

Arguments

`lower.tri.vector`	a numerical vector containing the lower triangular elements of the symmetric matrix.
`diag`	a logical indicator, `TRUE` or `FALSE`, indicating if `lower.tri.vector` contains diagonal elements of the symmetric matrix or not.

Details

`diag = FALSE` is the case where a correlation matrix whose diagonal elements are 1, therefore there is no need in storing the diagonal elements.

Value

The function `varbeta` returns the original symmetric matrix. We only use `diag = FALSE` to recover the correlation matrix, because diagonal elements of a correlation matrix are all 1, so there is no reason to store these diagonal elements in this case.

Author(s)

Yet Nguyen <`ynguyen@odu.edu`>

References

Yet Nguyen, Dan Nettleton, 2019. rmRNAseq: RNA-seq Analysis for Repeated-measures Data

Examples

```
set.seed(1)
lower.tri.vector <- runif(10)
varout <- rmRNAseq:::varbeta(lower.tri.vector, diag=TRUE)
varout
```

voomgls_CAR1 *General Linear Model Using Voom Output*

Description

This function run general linear model with `corCAR1` correlation structure in function `gls` for all genes where the input data come from the output of `voom`.

Usage

```
voomgls_CAR1(v, Subject, Time, ncores = 1, C.matrix, beta0 = NULL,
  print.progress = FALSE)
```

Arguments

`v`	output of `voom` function.
`Subject`	a vector of subjects or experimental units.
`Time`	a vector of time points.
`ncores`	number of cores for embarrassingly parallel procedure. Default value of `ncores` is 1.
`C.matrix`	is a list of matrix Ci in testing H0: Ci*beta = 0.
`beta0`	vector of the hypothesized value of beta, usually, beta0 is a 0 vector. The default option `beta0 = NULL` means that `beta0` is a vector of 0.
`print.progress`	logical indicator, TRUE or FALSE, to print the progress.

Value

a data frame has G rows (= number of genes) containing all outputs from `glsCAR1` function, shrinkage estimates of error variances, and F-type test statistics calculated by `teststat` function.

Examples

```
data(res)
data(covset)
v <- res$ori.res$v
v$E <- v$E[1:50,]
v$weights <- v$weights[1:50,]
Subject <- covset$ear
Time <- covset$time
ncores <- 1
C.matrix <- list()
C.matrix[[1]] <- limma::makeContrasts(line2, levels = design)
C.matrix[[2]] <- limma::makeContrasts(time2, levels = design)
C.matrix[[3]] <- limma::makeContrasts(time6, levels = design)
C.matrix[[4]] <- limma::makeContrasts(time24, levels = design)
C.matrix[[5]] <- limma::makeContrasts(linetime2, levels = design)
```

```
C.matrix[[6]] <- limma::makeContrasts(linetime6, levels = design)
C.matrix[[7]] <- limma::makeContrasts(linetime24, levels = design)
C.matrix[[8]] <- limma::makeContrasts(time2, time6, time24, levels = design)
C.matrix[[9]] <- limma::makeContrasts(linetime2,linetime6, linetime24, levels = design)
names(C.matrix) <- c("line2", "time2", "time6", "time24",
                    "linetime2", "linetime6", "linetime24",
                    "time", "int")
beta0 <- NULL
print.progress <- FALSE
voomglsout <- rmRNAseq:::voomgls_CAR1(v, Subject, Time, ncores,
C.matrix, beta0, print.progress)
```

---

voomgls_Symm *General Linear Model Using Voom Output corSymm correlction structure*

Description

This function run general linear model with corSymm correlation structure in function gls for all genes where the input data come from the output of voom.

Usage

```
voomgls_Symm(v, Subject, Time, ncores = 1, C.matrix = NULL,
  beta0 = NULL, print.progress = FALSE)
```

Arguments

v	output of voom function.
Subject	a vector of subjects or experimental units.
Time	a vector of time points.
ncores	number of cores for embarrassingly parallel procedure. Default value of ncores is 1.
C.matrix	is a list of matrix Ci in testing H0: Ci*beta = 0.
beta0	vector of the hypothesized value of beta, usually, beta0 is a 0 vector. The default option beta0 = NULL means that beta0 is a vector of 0.
print.progress	logical indicator, T or F, to print the progress.

Value

a data frame has G rows (= number of genes) containing all outputs from glsSymm function, shrinkage estimates of error variances, and F-type test statistics calculated by teststat function.

Examples

```
data(res)
data(covset)
v <- res$ori.res$v
v$E <- v$E[1:20,]
v$weights <- v$weights[1:20,]
Subject <- covset$ear
Time <- covset$time
ncores <- 1
C.matrix <- list()
C.matrix[[1]] <- limma::makeContrasts(line2, levels = design)
C.matrix[[2]] <- limma::makeContrasts(time2, time6, time24, levels = design)
names(C.matrix) <- c("line2", "time")
beta0 <- NULL
print.progress <- FALSE
voomglsout <- rmRNAseq:::voomgls_Symm(v, Subject, Time, ncores, C.matrix, beta0, print.progress)
```

`voomlimmaFit` *Analysis of RFI RNA-seq data Using voom*

Description

This function analyzes RFI RNA-seq data and simulated datasets using `voom`, which uses precision weights and linear model pipeline for the analysis of log-transformed RNA-seq data.

Usage

```
voomlimmaFit(counts, design, Effect)
```

Arguments

`counts`	a matrix of count data.
`design`	a design matrix.
`Effect`	the effect used to simulate data, either line2, or time. This effect is considered as the main factor of interest where the status of DE and EE genes was specified.

Value

a list of 4 components

`fit`	output of voom-limma fit.
`pv`	a vector of p-values of the test for significant of `Effect`.
`qv`	a vector of q-values corresponding to the pv above.

References

1. Gordon K. Smyth, Matthew Ritchie, Natalie Thorne,James Wettenhall, Wei Shi and Yifang Hu. limma: Linear Models for Microarray and RNA-Seq Data. User's Guide. https://www.bioconductor.org/packages/devel/bioc/vignettes/limma/inst/doc/usersguide.pdf

2. Gordon K. Smyth. Linear models and empirical bayes methods for assessing differential expression in microarray experiments. Stat Appl Genet Mol Biol. 2004;3:Article3. Epub 2004 Feb 12.

Examples

```
data(dat)
data(design)
counts <- dat[1:50,]
design <- design
Effect <- "line2"
voomlimmaout <- rmRNAseq:::voomlimmaFit(counts, design, Effect)
names(voomlimmaout)
```

Index