Package 'GSVA'

September 24, 2012

Version 1.4.4

Date 2012-09-18

Title Gene Set Variation Analysis

Author Justin Guinney <justin.guinney@sagebase.org> (with contributions from Robert Castelo <robert.castelo@upf.edu> and Sonja Haenzelmann <shanzelmann@imim.es)

Maintainer Justin Guinney <justin.guinney@sagebase.org>

Depends R (>= 2.13.0), methods, GSEABase (>= 1.18.0)

Imports methods, Biobase, GSEABase

Enhances snow, parallel

Suggests
limma, qpgraph, RBGL, graph, Rgraphviz, RColorBrewer,genefilter, mclust, edgeR, GSVAdata

Description Gene Set Variation Analysis (GSVA) is a non-parametric, unsupervised method for estimating variation of gene set enrichment through the samples of a expression data set. GSVA performs a change in coordinate systems, transforming the data from a gene by sample matrix to a gene-set by sample matrix, thereby allowing the evaluation of pathway enrichment for each sample. This new matrix of GSVA enrichment scores facilitates applying standard analytical methods like functional enrichment, survival analysis, clustering, CNV-pathway analysis or cross-tissue pathway analysis, in a pathway-centric manner.

License GPL (>= 2)

LazyLoad yes

biocViews Bioinformatics, Microarray, Pathways

URL http://www.sagebase.org

R topics documented:

`computeGeneSetsOverlap` *Compute gene-sets overlap*

Description

Calculates the overlap among every pair of gene-sets given as input.

Usage

```
## S4 method for signature 'list,character'
computeGeneSetsOverlap(gSets, uniqGenes, min.sz=1, max.sz=Inf)
## S4 method for signature 'list,ExpressionSet'
computeGeneSetsOverlap(gSets, uniqGenes, min.sz=1, max.sz=Inf)
## S4 method for signature 'GeneSetCollection,character'
computeGeneSetsOverlap(gSets, uniqGenes, min.sz=1, max.sz=Inf)
## S4 method for signature 'GeneSetCollection,ExpressionSet'
computeGeneSetsOverlap(gSets, uniqGenes, min.sz=1, max.sz=Inf)
```

Arguments

`gSets`	Gene sets given either as a `list` or a `GeneSetCollection` object.
`uniqGenes`	Vector of unique genes to be considered when calculating the overlaps.
`min.sz`	Minimum size.
`max.sz`	Maximum size.

Details

This function calculates the overlap between every pair of gene sets of the input argument `gSets`. Before this calculation takes place, the gene sets in `gSets` are firstly filtered to discard genes that do not match to the identifiers in `uniqGenes`. Secondly, they are further filtered to meet the minimum and/or maximum size specified with the arguments `min.sz` and `max.sz`. The overlap between two gene sets is calculated as the number of common genes between the two gene sets divided by the smallest size of the two gene sets.

Value

A gene-set by gene-set matrix of the overlap among every pair of gene sets.

Author(s)

J. Guinney

References

H\"anzelmann, S., Castelo, R. and Guinney, J. GSVA: Gene Set Variation Analysis, *submitted*

See Also

`filterGeneSets`

Examples

```
geneSets <- list(set1=as.character(1:4), set2=as.character(4:10))

computeGeneSetsOverlap(geneSets, unique(unlist(geneSets)))
```

---

filterGeneSets *Filter gene sets*

Description

Filters gene sets through a given minimum and maximum set size.

Usage

```
## S4 method for signature 'list'
filterGeneSets(gSets, min.sz=1, max.sz=Inf)
## S4 method for signature 'GeneSetCollection'
filterGeneSets(gSets, min.sz=1, max.sz=Inf)
```

Arguments

gSets	Gene sets given either as a list or a GeneSetCollection object.
min.sz	Minimum size.
max.sz	Maximum size.

Details

This function filters the input gene sets according to a given minimum and maximum set size.

Value

A collection of gene sets that meet the given minimum and maximum set size.

Author(s)

J. Guinney

References

H\"anzelmann, S., Castelo, R. and Guinney, J. GSVA: Gene Set Variation Analysis, *submitted*

See Also

computeGeneSetsOverlap

Examples

```
geneSets <- list(set1=as.character(1:4), set2=as.character(4:10))

filterGeneSets(geneSets, min.sz=5)
```

gsva *Gene Set Variation Analysis*

Description

Estimates GSVA enrichment scores.

Usage

```
## S4 method for signature 'ExpressionSet,list,missing'
gsva(expr, gset.idx.list, annotation=NULL,
    method=c("gsva", "ssgsea", "zscore", "plage"),
    rnaseq=FALSE,
    abs.ranking=FALSE,
    min.sz=1,
    max.sz=Inf,
    no.bootstraps=0,
    bootstrap.percent = .632,
    parallel.sz=0,
    parallel.type="SOCK",
    mx.diff=TRUE,
    tau=switch(method, gsva=1, ssgsea=0.25, NA),
    kernel=TRUE,
    verbose=TRUE)
## S4 method for signature 'ExpressionSet,GeneSetCollection,missing'
gsva(expr, gset.idx.list, annotation=NULL,
    method=c("gsva", "ssgsea", "zscore", "plage"),
    rnaseq=FALSE,
    abs.ranking=FALSE,
    min.sz=1,
    max.sz=Inf,
    no.bootstraps=0,
    bootstrap.percent = .632,
    parallel.sz=0,
    parallel.type="SOCK",
    mx.diff=TRUE,
    tau=switch(method, gsva=1, ssgsea=0.25, NA),
    kernel=TRUE,
    verbose=TRUE)
## S4 method for signature 'matrix,GeneSetCollection,character'
gsva(expr, gset.idx.list, annotation=NA,
    method=c("gsva", "ssgsea", "zscore", "plage"),
    rnaseq=FALSE,
    abs.ranking=FALSE,
    min.sz=1,
    max.sz=Inf,
    no.bootstraps=0,
    bootstrap.percent = .632,
    parallel.sz=0,
    parallel.type="SOCK",
    mx.diff=TRUE,
```

```
    tau=switch(method, gsva=1, ssgsea=0.25, NA),
    kernel=TRUE,
    verbose=TRUE)
## S4 method for signature 'matrix,list,missing'
gsva(expr, gset.idx.list, annotation=NULL,
    method=c("gsva", "ssgsea", "zscore", "plage"),
    rnaseq=FALSE,
    abs.ranking=FALSE,
    min.sz=1,
    max.sz=Inf,
    no.bootstraps=0,
    bootstrap.percent = .632,
    parallel.sz=0,
    parallel.type="SOCK",
    mx.diff=TRUE,
    tau=switch(method, gsva=1, ssgsea=0.25, NA),
    kernel=TRUE,
    verbose=TRUE)
```

Arguments

`expr`	Gene expression data which can be given either as an `ExpressionSet` object or as a matrix of expression values where rows correspond to genes and columns correspond to samples.
`gset.idx.list`	Gene sets provided either as a `list` object or as a `GeneSetCollection` object.
`annotation`	In the case of calling `gsva()` with expression data in a `matrix` and gene sets as a `GeneSetCollection` object, the `annotation` argument can be used to supply the name of the Bioconductor package that contains annotations for the class of gene identifiers occurring in the row names of the expression data matrix. By default `annotation=NULL` which implies that `gsva()` will try to match the identifiers in `expr` to the identifiers in `gset.idx.list` just as they are.
`method`	Method to employ in the estimation of gene-set enrichment scores per sample. By default this is set to `gsva` (H\"anzelmann et al, submitted) and other options are `ssgsea` (Barbie et al, 2009), `zscore` (Lee et al, 2008) or `plage` (Tomfohr et al, 2005). The latter two standardize first expression profiles into z-scores over the samples and, in the case of `zscore`, it combines them together as their sum divided by the square-root of the size of the gene set, while in the case of `plage` they are used to calculate the singular value decomposition (SVD) over the genes in the gene set and use the coefficients of the first right-singular vector as pathway activity profile.
`rnaseq`	Flag to inform whether the input gene expression data comes from microarray (`rnaseq=FALSE`, default) or RNA-Seq (`rnaseq=TRUE`) experiments.
`abs.ranking`	Flag to determine whether genes should be ranked according to their sign (`abs.ranking=FALSE`) or by absolute value (`abs.ranking=TRUE`). In the latter, pathways with genes enriched on either extreme (high or low) will be regarded as 'highly' activated.
`min.sz`	Minimum size of the resulting gene sets.
`max.sz`	Maximum size of the resulting gene sets.
`no.bootstraps`	Number of bootstrap iterations to perform.
`bootstrap.percent`	.632 is the ideal percent samples bootstrapped.

parallel.sz	Number of processors to use when doing the calculations in parallel. This requires to previously load either the parallel or the snow library. If parallel is loaded and this argument is left with its default value (parallel.sz=0) then it will use all available core processors unless we set this argument with a smaller number. If snow is loaded then we must set this argument to a positive integer number that specifies the number of processors to employ in the parallel calculation.
parallel.type	Type of cluster architecture when using snow.
mx.diff	Offers two approaches to calculate the enrichment statistic (ES) from the KS random walk statistic. mx.diff=FALSE: ES is calculated as the maximum distance of the random walk from 0. mx.diff=TRUE (default): ES is calculated as the magnitude difference between the largest positive and negative random walk deviations.
tau	Exponent defining the weight of the tail in the random walk performed by both the gsva (H\"anzelmann et al., submitted) and the ssgsea (Barbie et al., 2009) methods. By default, this tau=1 when method="gsva" and tau=0.25 when method="ssgsea" just as specified by Barbie et al. (2009) where this parameter is called alpha.
kernel	Logical, set to TRUE when the GSVA method employes a kernel non-parametric estimation of the empirical cumulative distribution function (default) and FALSE when this function is directly estimated from the observed data. This last option is justified in the limit of the size of the sample by the so-called Glivenko-Cantelli theorem.
verbose	Gives information about each calculation step. Default: FALSE.

Details

GSVA assesses the relative enrichment of gene sets across samples using a non-parametric approach. Conceptually, GSVA transforms a p-gene by n-sample gene expression matrix into a g-geneset by n-sample pathway enrichment matrix. This facilitates many forms of statistical analysis in the 'space' of pathways rather than genes, providing a higher level of interpretability.

The gsva() function first maps the identifiers in the gene sets to the identifiers in the input expression data leading to a filtered collection of gene sets. This collection can be further filtered to require a minimun and/or maximum size of the gene sets for which we want to calculate GSVA enrichment scores, by using the arguments min.sz and max.sz.

Value

A gene-set by sample matrix of GSVA enrichment scores.

Author(s)

J. Guinney and R. Castelo

References

Barbie, D.A. et al. Systematic RNA interference reveals that oncogenic KRAS-driven cancers require TBK1. *Nature*, 462(5):108-112, 2009.

H\"anzelmann, S., Castelo, R. and Guinney, J. GSVA: Gene Set Variation Analysis, *submitted*

Lee, E. et al. Inferring pathway activity toward precise disease classification. *PLoS Comp Biol*, 4(11):e1000217, 2008.

Tomfohr, J. et al. Pathway level analysis of gene expression using singular value decomposition. *BMC Bioinformatics*, 6:225, 2005.

See Also

filterGeneSets computeGeneSetsOverlap

Examples

```
library(limma)

p <- 10 ## number of genes
n <- 30 ## number of samples
nGrp1 <- 15 ## number of samples in group 1
nGrp2 <- n - nGrp1 ## number of samples in group 2

## consider three disjoint gene sets
geneSets <- list(set1=paste("g", 1:3, sep=""),
                 set2=paste("g", 4:6, sep=""),
                 set3=paste("g", 7:10, sep=""))

## sample data from a normal distribution with mean 0 and st.dev. 1
y <- matrix(rnorm(n*p), nrow=p, ncol=n,
            dimnames=list(paste("g", 1:p, sep="") , paste("s", 1:n, sep="")))

## genes in set1 are expressed at higher levels in the last 10 samples
y[geneSets$set1, (nGrp1+1):n] <- y[geneSets$set1, (nGrp1+1):n] + 2

## build design matrix
design <- cbind(sampleGroup1=1, sampleGroup2vs1=c(rep(0, nGrp1), rep(1, nGrp2)))

## fit linear model
fit <- lmFit(y, design)

## estimate moderated t-statistics
fit <- eBayes(fit)

## genes in set1 are differentially expressed
topTable(fit, coef="sampleGroup2vs1")

## estimate GSVA enrichment scores for the three sets
gsva_es <- gsva(y, geneSets, mx.diff=1)$es.obs

## fit the same linear model now to the GSVA enrichment scores
fit <- lmFit(gsva_es, design)

## estimate moderated t-statistics
fit <- eBayes(fit)

## set1 is differentially expressed
topTable(fit, coef="sampleGroup2vs1")
```

Index