

# Package ‘GSRI’

January 24, 2026

**Type** Package

**Title** Gene Set Regulation Index

**Version** 2.59.0

**Author** Julian Gehring, Kilian Bartholome, Clemens Kreutz, Jens Timmer

**Maintainer** Julian Gehring <jg-bioc@gmx.com>

**Imports** methods, graphics, stats, utils, genefilter, Biobase,  
GSEABase, les (>= 1.1.6)

**Depends** R (>= 2.14.2), fdrtool

**Suggests** limma, hgu95av2.db

**Enhances** parallel

**Description** The GSRI package estimates the number of differentially  
expressed genes in gene sets, utilizing the concept of the Gene  
Set Regulation Index (GSRI).

**License** GPL-3

**LazyLoad** yes

**biocViews** Microarray, Transcription, DifferentialExpression,  
GeneSetEnrichment, GeneRegulation

**git\_url** <https://git.bioconductor.org/packages/GSRI>

**git\_branch** devel

**git\_last\_commit** 6de90a8

**git\_last\_commit\_date** 2025-10-29

**Repository** Bioconductor 3.23

**Date/Publication** 2026-01-23

## Contents

GSRI-package . . . . .	2
export . . . . .	3
get-methods . . . . .	4
gsri . . . . .	6

Gsri-class	9
GSRI-internal	11
plot-methods	12
read-functions	14
sortGsri	15
test-functions	16
<b>Index</b>	<b>19</b>

---

**GSRI-package**
*Gene Set Regulation Index (GSRI) package*


---

**Description**

The **GSRI** package estimates the number of differentially expressed genes in gene sets, utilizing the concept of the Gene Set Regulation Index (GSRI).

**Details**

The GSRI approach estimates the number of differentially expressed genes in gene sets. It is independent of the underlying statistical test used for assessing the differential effect of genes and does not require any cut-off values for the distinction between regulated and unregulated genes. The approach is based on the fact that p-values obtained from a statistical test are uniformly distributed under the null hypothesis and are shifted towards zero in case of the alternative hypothesis.

Through non-parametric fitting of the uniform component of the p-value distribution, the fraction of regulated genes  $r$  in a gene set is estimated. The GSRI  $\eta$  is then defined as the  $\alpha * 100\%$ -quantile of the distribution of  $r$ , obtained from bootstrapping the samples within the groups. The index indicates that with a probability of  $(1 - \alpha)\%$  more than a fraction of  $\eta$  genes in the gene set is differentially expressed. It can also be employed to test the hypothesis whether at least one gene in a gene set is regulated. Further, different gene sets can be compared or ranked according to the estimated amount of regulation.

For details of the method, an application to experimental data, and a comparison with related approaches, see Bartholome et al., 2009.

The package is published under the GPL-3 license.

**Author(s)**

Julian Gehring, Kilian Bartholome, Clemens Kreutz, Jens Timmer

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

**References**

Kilian Bartholome, Clemens Kreutz, and Jens Timmer: Estimation of gene induction enables a relevance-based ranking of gene sets, *Journal of Computational Biology: A Journal of Computational Molecular Cell Biology* 16, no. 7 (July 2009): 959-967. <http://www.liebertonline.com/doi/abs/10.1089/cmb.2008.0226>

The **GSRI** package uses the functionality of the following packages:

Julian Gehring, Clemens Kreutz, Jens Timmer: les: Identifying Loci of Enhanced Significance in Tiling Microarray Data <http://bioconductor.org/help/bioc-views/release/bioc/html/les.html>

Korbinian Strimmer: fdrtool: Estimation and Control of (Local) False Discovery Rates. <http://CRAN.R-project.org/package=fdrtool>

Robert Gentleman, Vincent J. Carey, Wolfgang Huber, Florian Hahne: genefilter: methods for filtering genes from microarray experiments. <http://bioconductor.org/help/bioc-views/release/bioc/html/genefilter.html>

## See Also

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

## Examples

```
## Simulate expression data for a gene set of
## 100 genes, 20 samples (10 treatment, 10 control)
## and 30 regulated genes
set.seed(1)
exprs <- matrix(rnorm(100*20), 100)
exprs[1:30,1:10] <- rnorm(30*10, mean=2)
rownames(exprs) <- paste("g", 1:nrow(exprs), sep="")
groups <- factor(rep(1:2, each=10))

## Estimate the number of differentially expressed genes
res <- gsri(exprs, groups)
res

## Perform the analysis for different gene set
library(GSEABase)
gs1 <- GeneSet(paste("g", 25:40, sep=""), setName="set1")
gs2 <- GeneSet(paste("g", seq(1, nrow(exprs), by=5), sep=""), setName="set2")
gsc <- GeneSetCollection(gs1, gs2)

res2 <- gsri(exprs, groups, gs1)
res3 <- gsri(exprs, groups, gsc, verbose=TRUE)

summary(res2)
```

---

export

*Export results to file*

---

## Description

Export the results of a [Gsri](#) object to a file.

**Usage**

```
export(object, file, ...)
```

**Arguments**

object	An object of class <code>Gsri</code> whose results to export.
file	Character vector specifying the path of the file to be written.
...	Additional arguments, currently not used.

**Details**

The `export` method writes the results of the GSRI analysis, as obtained with `getGsri`, to a text file. The file is formatted with the standard parameters of the `write.table` function, while “`\t`” is used as field separator.

**Author(s)**

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

**See Also**

Package: [GSRI-package](#)

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

**Examples**

```
## Not run:
export(gsri, file)

## End(Not run)
```

**Description**

Access and extract results from an object of class `Gsri`.

**Usage**

```
getGsri(object, index, ...)
getCdf(object, index, ...)
getParms(object, ...)
```

## Arguments

object	An object of class <code>Gsri</code> whose results to access.
index	Optional argument used to subset the results of individual gene sets. For details, see the <code>plot</code> method.
...	Additional arguments, currently not used.

## Details

`getGsri` returns a data frame with the results of the GSRI analysis, equivalent to the display of the `print` and `show` method. For each gene set it contains the estimated fraction and total number of regulated genes, the standard deviation of the estimated fraction, the GSRI, and the total number of genes in the analysis.

`getCdf` returns a data frame with ECDF of the p-values for a gene set.

`getParms` returns a list with the parameters used in the analysis, including the weights (`weight`) for each gene in the gene set, the number (`nBoot`) of bootstraps for the calculation of the GSRI, the function (`test`) and its arguments (`testArgs`) used for assessing the differential effect between the groups, the confidence level for the GSRI, and the application of the Grenander estimator (`grenander`) in the calculation of the ECDF.

## Value

`getGsri` and `getCdf` return data frame with the results from the GSRI analysis and the ECDF, respectively. `getParms` returns a list with the parameters used in the analysis.

If more than one gene set is accessed, a list with one element per gene set is returned.

## Author(s)

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

## See Also

Package: [GSRI-package](#)

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

## Examples

```
## Not run:  
getGsri(object)  
getCdf(object)  
getParms(object)  
  
## End(Not run)
```

## Description

Estimate the number of differentially expressed genes in gene sets.

## Usage

```
gsri(exprs, groups, geneSet, names=NULL, weight=NULL, nBoot=100,
  test=rowt, testArgs=NULL, alpha=0.05, grenander=TRUE, verbose=FALSE,
  ...)
```

## Arguments

exprs	Matrix or object of class ExpressionSet containing the expression intensities of the microarray. If a matrix the rows represent the genes and the columns the samples, respectively.
groups	Factor with the assignments of the microarray samples to the groups, along which the differential effect should be estimated. Must have as many elements as exprs has samples.
geneSet	Optional object of class GeneSet or GeneSetCollection defining the gene set(s) used for the analysis. If missing all genes of exprs are considered to be part of the gene set. If an object of class GeneSet only these genes are considered to be part of the gene set. If an object of class GeneSetCollection the analysis is performed for each gene set of the collection individually.
names	Optional character vector with the names of the gene set(s). If missing the names are taken from the geneSet argument. Has to have as many unique elements as gene sets in the analysis.
weight	Optional numerical vector of weights specifying the certainty a gene is part of the gene set. If NULL all genes are assumed to have the same weight. Please note that the weights are defined in a relative way and thus any kind of positive weights is feasible. Must have as many elements as either the genes defined in geneSet or in exprs.
nBoot	Integer with the number of bootstrap samples to be drawn in the calculation of the GSRI (default: 100).
test	A function defining the statistical test used to assess the differential effect between the groups which are given by the groups argument. In this package, a t-test (rowt) and an F-test (rowF) are already supplied, with rowt being the default. Additionally, a custom test function can be used in order to be able to include any feasible statistical test in the analysis. For details, please see the 'details' section.
testArgs	List of optional arguments used by the test function. For details, please see the 'details' section and the help for test-functions.

alpha	Single numeric specifying the confidence level for the GSRI. The estimated GSRI is the lower bound of the $(1-\alpha)*100\%$ confidence interval obtained from bootstrapping.
grenander	Logical about whether the modified Grenander estimator for the cumulative density should be used instead of a centered ECDF. By default the modified Grenander estimator is used. For more information, please see the 'details' section.
verbose	Logical indicating whether the progress of the computation should be printed to the screen (default: FALSE). Most useful if geneSet is an object of class GeneSetCollection.
...	Additional arguments, including:
	<p><b>minSize:</b> Integer specifying the minimal number of genes in a gene set in order to perform an analysis. If the gene set has less than minSize in exprs, the gene set is ignored in the analysis.</p> <p><b>nCores:</b> Integer setting the number of cores used for the computation in combination with the <b>multicore</b> package for a GeneSetCollection. For details, please see the 'details' section.</p>

## Details

The `gsri` method estimates the degree of differential expression in gene sets. By assessing the part of the distribution of p-values consistent with the null hypothesis the number of differentially expressed genes is calculated.

Through non-parametric fitting of the uniform component of the p-value distribution, the fraction of regulated genes  $r$  in a gene set is estimated. The GSRI  $\eta$  is then defined as the  $\alpha * 100\%$ -quantile of the distribution of  $r$ , obtained from bootstrapping the samples within the groups. The index indicates that with a probability of  $(1 - \alpha)\%$  more than a fraction of  $\eta$  genes in the gene set is differentially expressed. It can also be employed to test the hypothesis whether at least one gene in a gene set is regulated. Further, different gene sets can be compared or ranked according to the estimated amount of regulation.

Assessing the differential effect is based on p-values obtained from statistical testing at the level of individual genes between the groups. The GSRI approach is independent of the underlying test and can be chosen according to the experimental design. With the t-test (`rowt`) and F-test (`rowF`) two widely used statistical test are already part of the package. Additional tests can easily be used which are passed with the `test` argument to the `gsri` method. For details on how to implement custom test functions, please refer to the help of `rowt` and `rowF` or the vignette of this package.

The GSRI approach further allows weighting the influence of individual genes in the estimation. This can be beneficial including for example the certainty that genes are part of a certain gene set derived from experimental findings or annotations.

Defining gene sets is available through the **GSEABase** package which provides the `GeneSet` and `GeneSetCollection` classes a single or multiple gene sets, respectively. This ensures a powerful approach for obtaining gene sets from data objects, data bases, and other bioconductor packages. For details on how to define or retrieve gene sets, please refer to the documentation of the **GSEABase** package, with a special focus on the `GeneSet` and `GeneSetCollection` classes.

The distribution of the p-values of a gene set is assessed in the cumulative density. In addition to a symmetrical empirical cumulative density function (ECDF), the modified Grenander estimator based on the assumption about the concave shape of the cumulative density is implemented and used

by default. While the modified Grenander estimator reduces the variance and makes the approach more stable especially for small gene set, it underestimates the number of regulated genes and thus leads to conservative estimates.

In the case that the computation is performed for several gene sets in the form of a `GeneSetCollection` object, it can be parallelized with the **multicore** package. Please note that this package is not available on all platforms. Using its capabilities requires attaching **multicore** prior to the calculation and specification of the `nCores` argument. For further details, please refer to the documentation of the **multicore** package. This may be especially relevant in the case that specific seed values for the bootstrapping are of interest.

## Value

An object of class `Gsri` with the slots:

**result:** Data frame containing the results of the GSRI estimation, with one row for each gene set.

**cdf:** List of data frames containing the ECDF of the p-values. Each data frame covers one gene set.

**parms:** List containing the parameter values used in the analysis, with the elements.

For details, please see the help for the `Gsri` class.

## Methods

Analysis for all genes of `exprs` part of the gene set:

```
signature(exprs="matrix", groups="factor", geneSet="missing")
signature(exprs="ExpressionSet", groups="factor", geneSet="missing")
```

Analysis for one gene set, defined as an object of class `GeneSet`:

```
signature(exprs="matrix", groups="factor", geneSet="GeneSet")
signature(exprs="ExpressionSet", groups="factor", geneSet="GeneSet")
```

Analysis for several gene sets, defined as an object of class `GeneSetCollection`:

```
signature(exprs="matrix", groups="factor", geneSet="GeneSetCollection")
signature(exprs="ExpressionSet", groups="factor", geneSet="GeneSetCollection")
```

In this case parallel computing capabilities provided by the **multicore** package may be available, depending on the platform.

## Note

The standard deviation of the estimated number of regulated genes as well as the GSRI are obtained through bootstrapping. Thus, the results for these two parameters may differ slightly for several realizations, especially for small numbers of bootstraps (`nBoot`). Setting the seed of the random number generator avoids this problem and yields exactly the same results for several realizations.

**Author(s)**

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

**See Also**

Package: [GSRI-package](#)

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

**Examples**

```
## Simulate expression data for a gene set of
## 100 genes, 20 samples (10 treatment, 10 control)
## and 30 regulated genes
set.seed(1)
exprs <- matrix(rnorm(100*20), 100)
exprs[1:30,1:10] <- rnorm(30*10, mean=2)
rownames(exprs) <- paste("g", 1:nrow(exprs), sep="")
groups <- factor(rep(1:2, each=10))

## Estimate the number of differentially expressed genes
res <- gsri(exprs, groups)
res

## Perform the analysis for different gene set
library(GSEABase)
gs1 <- GeneSet(paste("g", 25:40, sep=""), setName="set1")
gs2 <- GeneSet(paste("g", seq(1, nrow(exprs), by=5), sep=""), setName="set2")
gsc <- GeneSetCollection(gs1, gs2)

res2 <- gsri(exprs, groups, gs1)
res3 <- gsri(exprs, groups, gsc, verbose=TRUE)

summary(res2)
```

---

[Gsri-class](#)

*Class Gsri*

---

**Description**

Objects of the class `Gsri` contain the results of the GSRI analysis.

**Objects from the class**

Objects of class `Gsri` are returned by the `gsri` methods.

## Slots

**result:** Data frame containing the results of the GSRI estimation, with one row for each gene set and the columns:

**pRegGenes:** Fraction of regulated genes in the gene set

**pRegGenesSd:** Standard deviation of pRegGenes obtained from bootstrapping.

**nRegGenes:** Total number of regulated genes in the gene set.

**GSRI('alpha'%):** Gene Set Regulation Index, corresponding to the 'alpha'% quantile of the bootstrapped distribution.

**nGenes:** Total number of genes in the gene set.

**cdf:** List of data frames containing the ECDF of the p-values. Each data frame covers one gene set, with the columns:

**pval:** P-values obtained from the test function.

**cdf:** Empirical cumulative density.

**parms:** List containing the parameter values used in the analysis, with the elements:

- weight:** Weights for each gene in the gene set
- nBoot:** Number of bootstraps for the calculation of the GSRI
- test:** Statistical test function
- alpha:** Confidence level for the GSRI
- grenander:** Application of the Grenander estimator in the calculation of the ECDF
- testArgs:** Optional arguments for test function

## Methods

Analysis:

```
gsri: signature(exprs="matrix", groups="factor", geneSet="missing")
        signature(exprs="ExpressionSet", groups="factor", geneSet="missing")
        signature(exprs="matrix", groups="factor", geneSet="GeneSet")
        signature(exprs="ExpressionSet", groups="factor", geneSet="GeneSet")
        signature(exprs="matrix", groups="factor", geneSet="GeneSetCollection")
        signature(exprs="ExpressionSet", groups="factor", geneSet="GeneSetCollection")
Assess the degree of differential effect in the expression data.
```

Visualization:

```
plot: signature(x="Gsri", y=ANY)
```

Plot the empirical density of p-values and the corresponding estimated effect.

Export to file:

```
export: signature(object="Gsri", file="character")
```

Get methods:

```
getGsri: signature(object="Gsri")
```

```
getCdf: signature(object="Gsri")
```

**getParms:** `signature(object="Gsri")`

Show:

**show:** `signature(object="Gsri")`

**summary:** `signature(object="Gsri")`

### Author(s)

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

### See Also

Package: [GSRI-package](#)

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

### Examples

```
showClass("Gsri")
```

---

### Description

Internal functions of the **GSRI** package

### Usage

```
calcGsri(exprs, groups, name, id, weights, grenander=TRUE, nBoot=100,
test=NULL, testArgs=NULL, alpha=0.05, verbose=FALSE, ...)
multiStat(exprs, groups, id, index, test, testArgs)
gsriBoot(exprs, groups, weights, id, grenander, test, testArgs, nSamples)
bootInGroups(nSamples)
getArgs(name, first=NULL, last=NULL, ...)
```

### Details

Internal functions of the **GSRI** package. Users should not call them directly, but rather use the `gsri` methods.

### Author(s)

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

## References

The **GSRI** package uses the functionality of the following packages:

Julian Gehring, Clemens Kreutz, Jens Timmer: les: Identifying Loci of Enhanced Significance in Tiling Microarray Data <http://bioconductor.org/help/bioc-views/release/bioc/html/les.html>

Korbinian Strimmer: fdrtool: Estimation and Control of (Local) False Discovery Rates. <http://CRAN.R-project.org/package=fdrtool>

Robert Gentleman, Vincent J. Carey, Wolfgang Huber, Florian Hahne: genefilter: methods for filtering genes from microarray experiments. <http://bioconductor.org/help/bioc-views/release/bioc/html/genefilter.html>

## See Also

Package: [GSRI-package](#)

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

---

plot-methods

*Visualize GSRI results*

---

## Description

Plot the empirical cumulative density along with the estimated degree of regulation of the p-value distribution for a gene set.

## Usage

```
plot(x, y, ...)
```

## Arguments

<code>x</code>	An object of class <code>Gsri</code> containing the results to plot.
<code>y</code>	A single integer or character string specifying the results of which gene set to plot. This has to be given if <code>x</code> contains the results of several gene sets. An integer is interpreted as the index of the gene set (i.e. the row number), while a character is matched against the names of the gene sets.
<code>...</code>	Optional arguments used in order to customize the plot. See the ‘details’ section.

## Details

Plotting the p-value distribution and the estimation of the regularized component for a gene set allows to inspect the results in detail. The plot illustrates the empirical cumulative density function of the p-values obtained from testing for a differential effect between the groups. Additionally, the fit of the uniformly distributed component along with the estimated fraction of regularized genes and the GSRI is shown.

The `plot` method uses a special system in order to customize the graphical elements of the figure. It allows to refer to the different components with the name of the additional input argument; its value is a list containing named graphical parameters for the underlying plot function. The following list describes the possible names and their contribution.

`plot` Arguments for the axis and the labeling, passed to the `plot` function.

`fit` Arguments for the fit of the linear component of the ECDF, corresponding to the part without differential effect, passed to the `lines` function.

`ecdf` Arguments for the ECDF of the p-values, passed to the `lines` function.

`reg` Arguments for the horizontal line indicating the fraction of regulation, passed to the `lines` function.

`regText` Arguments for the text label of `reg`, passed to the `text` function.

`gsri` Arguments for the horizontal line indicating the GSRI, passed to the `lines` function.

`gsriText` Arguments for the text label of `gsri`, passed to the `text` function.

Thus, changing for example the limit of the y-axis, the plot type and color of the ECDF, and the label of the x-axis, you can use:

```
plot(x, plot=list(ylim=c(0, 0.8), xlab=expression(p)), ecdf=list(type="s", col="darkgreen"))
```

For more details, please see the ‘examples’ section.

## Author(s)

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

## See Also

Package: [GSRI-package](#)

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

## Examples

```
## Not run:
plot(x)
```

```
plot(x, plot=list(main="Example plot"), ecdf=list(pch=21),
fit=list(lty=2, lwd=0.5, col="black"), gsri=list(col="lightblue"))
```

```
plot(x2, 2)
```

```
plot(x2, "gs2")
## End(Not run)
```

---

read-functions

*Import of ‘.cls’ and ‘.gct’ files*

---

## Description

Import the groups from ‘.cls’ files and the expression data from ‘.gct’ files.

## Usage

```
readCls(file, ...)
readGct(file, ...)
```

## Arguments

file	Character vector specifying the path of the file to be read in.
...	Optinal arguments, currently not used.

## Details

With these methods the expression data and the assignment of the samples to groups can be read from ‘.cls’ (categorical class) and ‘.gct’ (gene cluster text) files, respectively. Details on the specific formats can be found at [http://www.broadinstitute.org/cancer/software/gsea/wiki/index.php/Data\\_formats](http://www.broadinstitute.org/cancer/software/gsea/wiki/index.php/Data_formats).

Please note that the `readCls` method reads only categorical class labels, not continuous ones.

## Value

For a ‘.cls’ file, a factor containing the groups.

For a ‘.gct’ file, a matrix containing the expression intensities, with rows corresponding to genes and columns to samples.

## Author(s)

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

## See Also

Package: [GSRI-package](#)

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

## Examples

```
## Not run:
exprs <- readGct(file)

groups <- readCls(file)

## End(Not run)
```

---

sortGsri

*Sort GSRI results*

---

## Description

Sort the results of an `Gsri` object.

## Usage

```
sortGsri(x, names, decreasing=TRUE, na.last=NA, ...)
```

## Arguments

<code>x</code>	Object of class <code>Gsri</code> whose results to sort.
<code>names</code>	Columns along which the results of <code>x</code> should be sorted, either a character vector with the names of the columns or an integer vector with the index of the columns. If the vector has several elements, sorting is performed along all of them, starting with the first and using subsequent ones to break existing ties. If <code>names</code> is not specified the results are sorted according to <code>pRegGenes</code> .
<code>decreasing</code>	Logical indicating whether the sorting should be in decreasing (default) or ascending order, see <code>sort</code> .
<code>na.last</code>	How NA values in the results should be treated, see <code>sort</code> .
<code>...</code>	Additional arguments, currently not used.

## Value

An object of class `Gsri`, with sorted slots `result` and `cdf`.

## Methods

```
signature(x="Gsri", names="ANY")
```

## Author(s)

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

**See Also**Package: [GSRI-package](#)Class: [Gsri](#)Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)**Examples**

```
## Not run:
sortGsri(object, c("pRegGenes", "nGenes"))
sortGsri(object, c(1, 5))

## End(Not run)
```

**Description**

Assess the differential effect in gene expression between groups of microarray replicates.

**Usage**

```
rowt(exprs, groups, id, index, testArgs)
rowF(exprs, groups, id, index, testArgs=list(var.equal=TRUE))
limmat(exprs, groups, id, index, testArgs)
```

**Arguments**

exprs	A matrix of expression values of size n x m, with rows representing the genes and columns representing the samples. The structure is the same as for the <code>exprs</code> argument of the <code>gsri</code> method.
groups	A factor with the length m, specifying the groups of the corresponding samples in <code>exprs</code> . The structure is the same as for the <code>exprs</code> argument of the <code>gsri</code> method.
id	Index vector for the rows of <code>exprs</code> which are part of the current gene set.
index	Index for the columns of <code>exprs</code> , such that <code>exprs[,index]</code> yields the bootstrapped expression matrix. Similar to the <code>index</code> arguments for <code>boot</code> of the <code>boot</code> package.
testArgs	Optional list with arguments passed to the <code>test</code> function. If 'NULL' or missing it is not passed to <code>test</code> and any existing default value of the function is used instead.
	<b>var.equal:</b> For the <code>rowF</code> function a logical indicating whether equal variances in the groups are assumed for the F-test (default: TRUE). For details, please see <code>rowFtests</code> in the <code>genefilter</code> package.

## Details

With the t-test and the F-test, two widely used statistical tests are available in this package. To allow a fast computation the implementations from the **genefilter** package is used.

It is also possible to use custom test statistics for assessing the differential effect between groups for each gene. In this case the function is passed as the `test` argument to the `gsri` method, while additional parameters for the function can be passed as a list via the `testArgs` argument. The defined function is required to be called as

```
function(exprs, groups, id, index, testArgs),
```

with `exprs` the matrix of expression intensities of the microarray and `groups` the factor of group labels, with the same structure as those passed initially to the `gsri` method. The vector `id` contains the indices of the genes part of the current gene set and is used to subset the expression intensities if necessary. The function has to return one p-value for each gene in the gene set indicating its differential effect. The vector `index` contains the indices of the samples for the bootstrapping. Applying `index` on the expression matrix in the form of `exprs[, index]` generates the bootstrapped data set.

For details on how to define and use your custom test functions, please refer to the ‘examples’ section or the vignette of this package.

## Value

A vector of p-values, indicating the significance of the differential effect between groups for each gene.

## Author(s)

Julian Gehring

Maintainer: Julian Gehring <julian.gehring@fdm.uni-freiburg.de>

## See Also

Package: [GSRI-package](#)

Class: [Gsri](#)

Methods: [gsri](#) [getGsri](#) [getCdf](#) [getParms](#) [export](#) [sortGsri](#) [plot](#) [show](#) [summary](#) [readCls](#) [readGct](#)

Statistical tests from the **genefilter** package: [rowFtests](#)

## Examples

```
## Not run:
## A simple example for a custom test function using a linear model.
## Note that for two groups this is equivalent to a t-test with equal variances.
testFcn <- function(exprs, groups, id, index, testArgs) {

  stat <- function(e, g, f) {
    m <- lm(f)
    pval <- summary(m)$coefficients[2,4]
  }

  pvals <- apply(exprs[id,index], 1, stat, groups, testArgs$f)
```

```
return(pvals)
}

## Pass the definition of the linear model through 'testArgs'
f <- formula(e ~ g)

res <- gsri(exprs, groups, test=testFcn, testArgs=list(f=f))

## End(Not run)
```

# Index

- \* **classes**
  - Gsri-class, 9
- \* **distribution**
  - gsri, 6
  - GSRI-package, 2
- \* **hplot**
  - plot-methods, 12
- \* **htest**
  - gsri, 6
  - GSRI-package, 2
  - test-functions, 16
- \* **internal**
  - GSRI-internal, 11
- \* **io**
  - export, 3
  - get-methods, 4
  - read-functions, 14
- \* **methods**
  - Gsri-class, 9
  - plot-methods, 12
- \* **package**
  - GSRI-package, 2
- \* **utilities**
  - plot-methods, 12
- \* **utils**
  - sortGsri, 15
- bootInGroups (GSRI-internal), 11
- calcGsri (GSRI-internal), 11
- export, 3, 3, 4, 5, 9, 11–14, 16, 17
- export, Gsri, character-method (export), 3
  - get-methods, 4
- getArgs (GSRI-internal), 11
- getCdf, 3–5, 9, 11–14, 16, 17
  - getCdf (get-methods), 4
  - getCdf, Gsri, ANY-method (get-methods), 4
  - getCdf, Gsri, missing-method (get-methods), 4
- getGsri, 3–5, 9, 11–14, 16, 17
  - getGsri (get-methods), 4
  - getGsri, Gsri, ANY-method (get-methods), 4
  - getGsri, Gsri, missing-method (get-methods), 4
  - getParms, 3–5, 9, 11–14, 16, 17
  - getParms (get-methods), 4
  - getParms, Gsri-method (get-methods), 4
  - GSRI (GSRI-package), 2
  - Gsri, 3–5, 8, 9, 11–14, 16, 17
  - Gsri (Gsri-class), 9
  - gsri, 3–5, 6, 9, 11–14, 16, 17
  - gsri, ExpressionSet, factor, GeneSet-method (gsri), 6
  - gsri, ExpressionSet, factor, GeneSetCollection-method (gsri), 6
  - gsri, ExpressionSet, factor, missing-method (gsri), 6
  - gsri, matrix, factor, GeneSet-method (gsri), 6
  - gsri, matrix, factor, GeneSetCollection-method (gsri), 6
  - gsri, matrix, factor, missing-method (gsri), 6
  - Gsri-class, 9
  - GSRI-internal, 11
  - gsri-methods (gsri), 6
  - GSRI-package, 2
  - gsriBoot (GSRI-internal), 11
  - limmat (test-functions), 16
  - multiStat (GSRI-internal), 11
  - plot, 3–5, 9, 11–14, 16, 17
    - plot (plot-methods), 12
    - plot, Gsri, ANY-method (plot-methods), 12
    - plot, Gsri, integer-method (plot-methods), 12
    - plot-methods, 12

read-functions, 14  
readCls, 3–5, 9, 11–14, 16, 17  
readCls (read-functions), 14  
readCls, character-method  
    (read-functions), 14  
readGct, 3–5, 9, 11–14, 16, 17  
readGct (read-functions), 14  
readGct, character-method  
    (read-functions), 14  
rowF (test-functions), 16  
rowFtests, 17  
rowt (test-functions), 16  
  
show, 3–5, 9, 11–14, 16, 17  
show (Gsri-class), 9  
show, Gsri-method (Gsri-class), 9  
sortGsri, 3–5, 9, 11–14, 15, 16, 17  
sortGsri, Gsri-method (sortGsri), 15  
sortGsri-methods (sortGsri), 15  
summary, 3–5, 9, 11–14, 16, 17  
summary (Gsri-class), 9  
summary, Gsri-method (Gsri-class), 9  
  
test-functions, 16