Package 'cnvGSA'

September 24, 2012

Type Package

Title Gene Set Analysis of (Rare) Copy Number Variants

Version 1.0.0

Date 2012-03-19

Author Daniele Merico <daniele.merico@gmail.com>; packaged by Robert Ziman <rziman@gmail.com>

Maintainer Robert Ziman <rziman@gmail.com>

Description
This package is intended to facilitate gene-set association with rare CNVs in case-control studies.

License LGPL

LazyLoad yes

biocViews MultipleComparisons

Depends methods

Suggests cnvGSAdata, org.Hs.eg.db

R topics documented:

cnvGSA-package *Gene-Set Analysis of (Rare) Copy Number Variants*

Description

cnvGSA is an R package meant to facilitate gene-set analysis of (rare) copy number variants (CNVs).

Details

Known gene-sets are tested for prevalence of rare variants in case vs. control subjects. Whenever a subject has at least one gene in a gene-set affected by a rare variant, a perturbation score of 1 is assigned to the (subject, gene-set) pair; for each gene-set, subject counts are tested vs. control counts using the Fisher Exact Test (FET). Significant gene-sets will have a significantly high count in cases compared to controls. Statistical reports on burden are also generated.

Author(s)

Daniele Merico <daniele.merico@gmail.com>; packaged by Robert Ziman <rziman@gmail.com>

cnvGSAexportBurdenStats *Export burden statistics*

Description

Exports burden statistics into a tab-delimited text file.

Usage

```
cnvGSAexportBurdenStats( output, filenamePrefix )
```

Arguments

output	Object of class CnvGSAOutput
filenamePrefix	Prefix to use in the names of the output files (see below)

Value

Given filenamePrefix="example", outputs "example_burdenGs_coverage.txt" and "example_burdenGs_pairs.txt".

Author(s)

Robert Ziman <rziman@gmail.com>

Examples

```
library( "cnvGSAdata" )
data( "cnvGSA_output_example" )
cnvGSAexportBurdenStats( output, "example" )
```

cnvGSAFisher *CNV/gene-set association using Fisher Exact Test*

Description

Performs the CNV/gene-set association using the Fisher Exact Test.

Usage

```
cnvGSAFisher( input )
```

Arguments

input	Object of class CnvGSAInput.

Details

Whenever a subject has at least one gene in a gene-set affected by a rare variant, a perturbation score of 1 is assigned to the (subject, gene-set) pair; for each gene-set, subject counts are tested vs. control counts using the Fisher Exact Test (FET). Significant gene-sets will have a significantly high count in cases compared to controls. Statistical reports on burden are also generated.

Value

Outputs an object of class CnvGSAOutput containing detailed enrichment results as well as gene-centric and global burden statistics.

Author(s)

Robert Ziman <rziman@gmail.com>

Examples

```
library("cnvGSAdata")
data("cnvGSA_input_example")
output <- cnvGSAFisher(input)
```

CnvGSAInput-class *Class "CnvGSAInput"*

Description

Container class for the input structures required by the main function (i.e. cnvGSA.Fisher()).

Slots

cnvData: Object of class "list" containing CNV data

gsData: Object of class "list" containing gene-set data

geneData: Object of class "list" containing gene annotations (symbols and descriptive names)

params: Object of class "list" containing the test parameters

Constructor

`CnvGSAInput( cnvData, gsData, geneData, params )`: Creates a CnvGSAInput object.

`cnvData` Structure containing CNV data along with sample-to-class information and filters

`gsData` Structure containing gene-set data

`geneData` Structure containing gene annotations (symbols and descriptive names)

`params` Structure containing main test parameters

See the vignette for complete details on each of these structures as well as a full example of how to load them.

Methods

cnvData `signature(obj = "CnvGSAInput")`: Gets `cnvData`.

cnvData<- `signature(obj = "CnvGSAInput")`: Sets `cnvData`.

geneData `signature(obj = "CnvGSAInput")`: Gets `geneData`.

geneData<- `signature(obj = "CnvGSAInput")`: Sets `geneData`.

gsData `signature(obj = "CnvGSAInput")`: Gets `gsData`.

gsData<- `signature(obj = "CnvGSAInput")`: Sets `gsData`.

params `signature(obj = "CnvGSAInput")`: Gets `params`.

params<- `signature(obj = "CnvGSAInput")`: Sets `params`.

Author(s)

Robert Ziman <rziman@gmail.com>

Examples

```
## See vignette for full details and worked example
```

CnvGSAOutput-class *Class* "CnvGSAOutput"

Description

Container class for the output structures produced by the main function (i.e. cnvGSA.Fisher()).

Slots

`cnvData`**:** Object of class `"list"` containing original and filtered CNV data

`burdenSample`**:** Object of class `"list"` containing burden analysis results for objects

`burdenGs`**:** Object of class `"list"` containing burden analysis results for gene-sets

`geneData`**:** Object of class `"list"` containing gene-centric statistics

`enrRes`**:** Object of class `"list"` containing the gene-set enrichment results

Methods

burdenGs `signature(obj = "CnvGSAOutput")`: Gets `burdenGs`.

burdenSample `signature(obj = "CnvGSAOutput")`: Gets `burdenSample`.

cnvData `signature(obj = "CnvGSAOutput")`: Gets `cnvData`.

enrRes `signature(obj = "CnvGSAOutput")`: Gets `enrRes`.

geneData `signature(obj = "CnvGSAOutput")`: Gets `geneData`.

summary `signature(object = "CnvGSAOutput")`: Prints out several summary statistics.

Author(s)

Robert Ziman <rziman@gmail.com>

Examples

```
## See vignette for full discussion of output elements
library("cnvGSAdata")
data("cnvGSA_output_example")
slotNames("output")
```

---

getCnvGenes *Determine genes hit by CNVs*

Description

Takes as input a data frame of CNVs and a data frame of gene coordinates and returns the genes hit by each CNV.

Usage

```
getCnvGenes( cnv, genemap, delim )
```

Arguments

`cnv`	Data frame containing the CNVs. It should contain at least the following columns: - Chr: Chromosome on which the CNV is found (e.g. "1", "12", "X") - Coord_i: Initial coordinate (aka start position) of the CNV on the chromosome - Coord_f: Final coordinate (aka end position) of the CNV on the chromosome
`genemap`	Data frame containing genes along with the chromosomes and coordinates for each. The columns should be similar to the `cnv` data frame: - Chr: Chromosome on which the gene is found (e.g. "1", "12", "X") - Coord_i: Initial coordinate (aka start position) of the gene on the chromosome - Coord_f: Final coordinate (aka end position) of the gene on the chromosome - GeneID: Gene ID
`delim`	Character to be used to separate genes in the output for each CNV.

Value

A vector in which each element contains a delimited string of the genes that fall within the range of the corresponding CNV in the input.

Author(s)

Robert Ziman <rziman@gmail.com>

Examples

```
library("cnvGSAdata")

## Read in the example gene map
genemapFile <- system.file(
"extdata",
"merge_00k_flank_hg18_refGene_jun_2011_exon.gff",
package = "cnvGSAdata"
)
fields <- read.table(
genemapFile,
sep = "\t",
comment.char = "",
quote = "\"",
header = FALSE,
stringsAsFactors = FALSE
)
genemap <- data.frame(
Chr = fields[,1],
Coord_i = fields[,4],
Coord_f = fields[,5],
GeneID = fields[,11],
stringsAsFactors = FALSE
)
genemap$Chr <- sub(genemap$Chr, pattern = "chr", replacement = "")

## Read in a few CNVs
cnvFile <- system.file("extdata", "cnv.gvf", package="cnvGSAdata")
cnv <- readGVF(cnvFile)
cnv <- cnv[1:5,]

## Get CNV genes
delim <- ";"
genes <- getCnvGenes(cnv, genemap, delim)
```

---

readGMT *Read gene-sets from a .gmt file*

Description

Reads gene-sets from a .gmt file.

Usage

```
readGMT(filename)
```

Arguments

filename	The name of a file in the Gene Matrix Transposed format.

Value

A list having the following two elements:

gs2gene - A list of character vectors where each vector contains the genes for a particular gene-set; the gene-set names ("GO:0030850" etc.) are stored as the names of the list elements. Since gene-sets can hold different numbers of genes, the vectors will typically have different lengths.

gs2name - A single character vector mapping each gene-set name to its description. The descriptions are stored as the vector elements and the gene-set names are stored as the names of the vector elements.

This list structure should be assigned to the gsData slot of a CnvGSAInput object.

Author(s)

Robert Ziman <rziman@gmail.com>

References

GSEA wiki: Data formats http://www.broadinstitute.org/cancer/software/gsea/wiki/index.php/Data_formats#GMT:_Gen

Examples

```
library("cnvGSAdata")
gsDataFile <- system.file("extdata", "gsData.gmt", package="cnvGSAdata")
gsData <- readGMT(gsDataFile)
```

---

readGVF *Read CNVs from a .gvf file*

Description

Reads CNV data from a .gvf file.

Usage

```
readGVF(filename)
```

Arguments

filename	The name of a Genome Variation Format file.

Value

A data frame containing the CNVs. Each row contains the following columns for one CNV:

`SampleID` ID assigned to the subject's DNA sample in which the CNV was found

`Chr` Chromosome on which the CNV is located

`Coord_i` Start position of the CNV on the chromosome

`Coord_f` End position of the CNV on the chromosome

`Type` CNV type (e.g. "DEL" or "DUP")

`Genes` (empty string)

CnvID ID assigned to the CNV

N.B. that the Genes column is *not* assigned. Normally it should contain genes affected by the CNV – stored in a delimited format inside a character string (e.g. "54777;255352;84435" for semicolon-delimited EntrezGene identifiers). To assign this column, use the getCnvGenes() function. The data frame can then be assigned to the cnv element of the cnvData list structure that comprises one of the slots in a CnvGSAInput object.

Author(s)

Robert Ziman <rziman@gmail.com>

References

http://www.sequenceontology.org/resources/gvf.html

Examples

```
library("cnvGSAdata")
cnvFile <- system.file("extdata", "cnv.gvf", package="cnvGSAdata")
cnv <- readGVF(cnvFile)
```

---

readParamsRFile *Read cnvGSA parameters from a file (R format)*

Description

Reads cnvGSA parameters from a text file that has them encoded using R syntax.

Usage

```
readParamsRFile(filename)
```

Arguments

filename	Name of the file.

Details

To make it easier to integrate the association test into a larger bioinformatics pipeline, it is convenient to read in the parameters from an external source such as a text file. One such implementation is to record each parameter on its own line using R syntax:

```
 grandtotals_mode <- "all" sample_classes <- c("case", "ctrl") fdr_iter <- 1000 extended_report
boxplot_PDFs <- FALSE limits_type <- "DEL"
```

The package provides readParamsRFile() to parse such a file (essentially just source()ing it and then handling the few possibilites around the cnvData\$filters parameters).

Value

Test parameters in a list structure that can be assigned to the params slot of a CnvGSAInput object.

Author(s)

Robert Ziman <rziman@gmail.com>

Examples

```
paramFile <- system.file("scripts", "params_example.R", package="cnvGSA")
params <- readParamsRFile(paramFile)
```

---

summary-methods *cnvGSA methods for function* summary

Description

cnvGSA methods for function summary

Methods

signature(object = "CnvGSAOutput") Outputs several summary statistics from the CnvGSAOutput object.

Examples

```
library("cnvGSAdata")
data("cnvGSA_output_example")
summary(output)
```

Index