GeneSpring

April 19, 2009

`GSint` *Creation of GSint class object*

Description

The `GSint` class object contains the expression values and the annotations of the GeneSpring Experiment Interpretation object. Apart from the `new` function, the function `GSint` can be used to create a new `GSint` object.

Usage

```
GSint(expName = "GeneSpring experiment",
        a.nor = matrix(nc = 0, nr = 0),
        se.nor = matrix(nc = 0, nr = 0),
        sd.nor = matrix(nc = 0, nr = 0),
        n.nor = matrix(nc = 0, nr = 0),
        a.raw = matrix(nc = 0, nr = 0),
        se.raw = matrix(nc = 0, nr = 0),
        sd.raw = matrix(nc = 0, nr = 0),
        n.raw = matrix(nc = 0, nr = 0),
        a.ctr = matrix(nc = 0, nr = 0),
        se.ctr = matrix(nc = 0, nr = 0),
        sd.ctr = matrix(nc = 0, nr = 0),
        n.ctr = matrix(nc = 0, nr = 0),
        expparam = data.frame(),
        numConditions = 0,
        ngenes = 0,
        genenames = c("")
)
```

Arguments

`expName`	Experiment name.
`a.nor`	Matrix representing the AVERAGE expression values for the NORMALIZED expression values.
`se.nor`	Matrix representing the STANDARD ERROR for the NORMALIZED expression values.

`sd.nor`	Matrix representing the STANDARD DEVIATION for the NORMALIZED expression values.
`n.nor`	Matrix representing the NUMBER OF REPLICATES in this condition for the NORMALIZED values.
`a.raw`	Matrix representing the AVERAGE expression values for the RAW expression values.
`se.raw`	Matrix representing the STANDARD ERROR for the RAW expression values.
`sd.raw`	Matrix representing the STANDARD DEVIATION for the RAW expression values.
`n.raw`	Matrix representing the NUMBER OF REPLICATES in this condition for the RAW values.
`a.ctr`	Matrix representing the AVERAGE expression values for the CONTROL values.
`se.ctr`	Matrix representing the STANDARD ERROR for the CONTROL values.
`sd.ctr`	Matrix representing the STANDARD DEVIATION for the CONTROL values.
`n.ctr`	Matrix representing the NUMBER OF REPLICATES in this condition for the CONTROL values.
`expparam`	Data frame representing the sample attributes or experimental parameters.
`numConditions`	Numeric value representing the number of conditions (or samples) in this interpretation or experiment.
`ngenes`	Numeric values representing the number of genes in the experiment.
`genenames`	List representing the gene names in the experiment.

Details

The `GSint` class object is a representation of a GeneSpring Experiment Interpretation. The object consists of 17 slots:

1) The name of the experiment (slot: expName) 2) Twelve slots representing the various types of expression data as matrix objects 3) The experimental parameters (slot: expparam) 4) The number of samples or conditions in the experiment (slot: numConditions) 5) The number of genes (slot: ngenes) 5) A list of the gene names(slot: genenames)

All of the expression value slots are optional and are not guaranteed to contain data. When a GeneSpring Experiment is read in from file, only the a.nor slot and optionally the a.ctr slot will contain data. When a GeneSpring Experiment INTERPRETATION is read, any of the 12 expression values slots could contain data. Which data slots are filled is dependent on which choices were made in the construction of the external programs in GeneSpring.

For more information about GeneSpring external programs, see the GeneSpring manual. For more information about the differences between an Experiment and an Interpretation, see the R manual entries for `GSload.int` and `GSload.exp`.

Value

Both `GSint` and `new` return an object of class `GSint`.

Methods

show `signature(object = "GSint")`: Produces a short description of the GSint class. It lists all the objects contained in the class.

Note

This and any other functions in the GeneSpring package are only useful when using Agilent Technologies Gene Expression software GeneSpring.

Author(s)

Thon de Boer, Agilent Technologies, Santa Clara, CA, USA ⟨thon_deboer@agilent.com⟩

References

For more information on using GeneSpring with R see http://www.chem.agilent.com/scripts/generic.asp?lpage=34733

See Also

GSload.int, GSload.exp, GSsave.exp, GSint2BC, BC2GSint

Examples

```
#Create some simple sample annotation
ep <- t(data.frame(SampleName=1:10))
#Create a GSint object with random data as the RAW data object
gs.int <- GSint(expName="Random data",a.raw=matrix(rnorm(1000),nc=10),expparam=ep, genena

gs.int@expName
t(gs.int@expparam)
```

GSint2BC *Converters for GeneSpring GSint and BioConductor ExpressionSet objects*

Description

These functions convert GeneSpring Experiment (Interpretation) objects into BioConductor expression objects and vice versa.

Usage

```
expr.set <- GSint2BC(interpretation, what = "nor")
gs.int <- BC2GSint(exprSet)
```

Arguments

interpretation	GeneSpring Experiment Interpretation object.
what	For GSload.intBC() only. Indicates what data from the GeneSpring Experiment Interpretation should be used. Valid values for what can be nor and raw, representing Normalized or Raw expression values. The Control values cannot be used.
exprSet	GeneSpring Experiment Interpretation object.

Details

The GeneSpring Experiment Interpretation and the BioConductor objects are quite similar, but there are some differences. GeneSpring interpretations can contain information on both normalized and original raw values, and can include, Standard Error, Standard Deviation and number of replicates information, but the BioConductor objects can contain more sample attribute information in the form of the MIAME annotations.

These functions provide converters for both types of objects.

When converting a GeneSpring object into a BioConductor object, the user has the choice of either using the normalized or raw expression values for the BioConductor `ExpressionSet` object. Conversely, when a BioConductor object is converted into a GeneSpring object, the expression values are loaded as normalized values.

The phenoData objects describing the phenotypical data for the samples in a BioConductor object are converted into Experimental Parameter objects in the GeneSpring class object, and vice versa.

The experiment name and any of the other annotations for the BioConductor objects that are stored in the MIAME objects are currently not converted, although future versions of these converters may provide better mapping.

Value

The return value depends on which function was called. The `BC2GSint()` function returns an object of class `GSint`. The `GSint2BC()` function returns a BioConductor object of class `ExpressionSet`. See the documentation on `GSint` and `ExpressionSet` for more information.

Note

This and any other functions in the GeneSpring package are only useful when using Agilent Technologies Gene Expression software GeneSpring.

Author(s)

Thon de Boer, Agilent Technologies, Santa Clara, CA, USA ⟨thon_deboer@agilent.com⟩

References

For more information on using GeneSpring with R see http://www.chem.agilent.com/scripts/generic.asp?lpage=34733

See Also

`GSload.exp`, `GSload.int`, `GSsave.exp`

Examples

```
#Use the example data etc. from ExpressionSet
library(Biobase)
data(geneData)
data(geneCov)
covdesc<- list("Covariate 1", "Covariate 2", "Covariate 3")
names(covdesc) <- names(geneCov)
pdata <- new("AnnotatedDataFrame")
pData(pdata) <- geneCov
varLabels(pdata) <- covdesc
```

```
eset <- new("ExpressionSet", exprs=geneData, phenoData=pdata)

#Convert the objects into GeneSpring objects and back again
gs.int <- BC2GSint(eset)
expr.set <- GSint2BC(gs.int)
```

`GSload.exp` *Read GeneSpring Experiment from file*

Description

A GeneSpring Experiment is read in, usually from a file, but a connection object (like `stdin()`) can also be passed. The return value can either be an object of class `GSint` or BioConductor `ExpressionSet` representing the Experiment Interpretation from GeneSpring.

Usage

```
gs.int<-GSload.exp(filename = "GS_R_in.txt", EOF="///", chunk=1, append=FALSE)
expr.set<-GSload.expBC(filename = "GS_R_in.txt", EOF="///", chunk=1, append=TRUE
```

Arguments

`filename`	Name of the file containing the GeneSpring Experiment Interpretation. Can also be a connection object.
`EOF`	String containing the delimiter used to separate the chunks in the file to be read.
`chunk`	Number indicating the chunk the experiment should be read from.
`append`	Boolean indicating if the experiment should be appended to the output, allowing for multiple objects to be send back to GeneSpring.

Details

A GeneSpring Experiment is quite distinct from a GeneSpring Experiment INTERPRETATION. See `GSload.int` for more information on GeneSpring Experiment Interpretations.

A GeneSpring Experiment consists of the Normalized expression values for each sample and optionally the Control values for that normalized value. The control values for an experiment are the values that are used to create the normalized values by simply dividing the Raw expression value by the Control value. The GeneSpring Normalization routines will calculate the control values for each gene and these values can be used in subsequent analysis.

In a GeneSpring Experiment, each sample or hybridization has its own column (set) of expression values, where the expression values for an Experiment INTERPRETATION represent the mean value of the replicate values for that condition.

For example, if there are 4 replicates for samples of time 0, there will be one column of normalized data for time 0 in the Experiment Interpretation, and there will be four columns of average values for each of the replicates, in the GeneSpring Experiment.

The experimental parameters for the experiment, are stored in the header of the file, with each sample's parameter in columns above the expression values. When control values are present in the file, the parameter value is simply `control`.

The names of the experimental parameters can also indicate whether or not the parameter has any unit identifier (like hour, mM, etc.) and this will be a suffix of the parameter name in parenthesis.

The experimental parameters will also contain an indication whether or not the values are numeric or not, but the addition of an asterisk character for non-numeric values and the absence of the asterisk for numerical values.

The final character of the parameter name will indicate how the parameter was used in GeneSpring. When the parameter is used as a discontinuous parameter, the character is a 'C'. When the parameter is used as a color coding parameter, the character is a 'S'. When the parameter is not used to indicate a group or color coding the character is a 'R'. When a parameter was used as the continuous parameter, the last character is missing or empty.

When loading experimental parameters from file, the units and indication of use will be discarded.

GeneSpring will be able to send multiple objects and it sends all of the objects in one file, one object per "chunk", separated by a delimiter on its own line. The order in which the objects are defined (in the "Input" section of the External Program Interface definition) determines in which chunk the experiment can be found. Conversely, by using the "append" parameter in the `GSsave.exp` function, multiple experiments (or other objects) can be send to GeneSpring and each of them will be loaded as a separate object.

Value

The return value depends on which function was called. The `GSload.exp` function returns a object of class `GSint`. The `GSload.expBC` function returns a BioConductor object of class `ExpressionSet`. See the documentation on `GSint` and `ExpressionSet` for more information.

Note

This and any other functions in the GeneSpring package are only useful when using Agilent Technologies Gene Expression software GeneSpring.

Author(s)

Thon de Boer, Agilent Technologies, Santa Clara, CA, USA ⟨thon_deboer@agilent.com⟩

References

For more information on using GeneSpring with R see `http://www.chem.agilent.com/scripts/generic.asp?lpage=34733`

See Also

`GSload.int`, `GSsave.exp`, `GSint2BC`, `BC2GSint`

`GSload.genelist` *Load and save GeneSpring gene lists*

Description

These functions load and save list of genes from and to GeneSpring. Gene lists can consists of a simple list of genes or can contain one column of associated values containing either the classifications or some numeric value representing the result of an analysis (Like P values after a ANOVA for instance).

Usage

```
glist <- GSload.genelist(filename = "GS_R_in.txt", EOF="///", chunk=1)
GSsave.genelist(genelist, filename = "GS_R_out.txt", EOF="///", append=TRUE)
```

Arguments

filename	Name of the file of the genelist. Can be connection object like stdin().
EOF	String containing the delimiter used to separate the chunks in the file to be read.
chunk	Number indicating the chunk the gene list should be read from.
genelist	Gene list to be saved to the file. Can either be a list or a data frame/matrix.
append	Boolean indicating if the gene list should be appended to the output, allowing for multiple objects to be send back to GeneSpring.

Details

Gene lists are either simple lists or vectors of gene names or data.frames. When the gene list contains associated values, the gene list is stored as a data.frame, with the gene names as the first column of the data.frame and the values in the second column of the data.frame.

GeneSpring will be able to send multiple objects and it sends all of the objects in one file, one object per "chunk", separated by a delimiter on its own line. The order in which the objects are defined (in the "Input" section of the External Program Interface definition), determines in which chunk the gene list can be found. Conversely, by using the "append" parameter in the GSsave.genelist function, multiple gene lists (or other objects) can be send to GeneSpring and each of them will be loaded as a separate object.

There are two types of gene lists within GeneSpring, although only one is actually called a gene lists while the other one is called a classification.

Gene lists proper are list of genes in GeneSpring with an optional NUMERICAL associated value. This numerical values usually represents the result of one or more analyses, like the P value of an ANOVA, or the number of samples that passed the filter that was applied.

The other type of gene lists are those lists that are produced when a Classification is saved to a file. The classification objects are the result of the clustering methods in GeneSpring, like K-means clustering or the QT-clustering. When classifications are stored in file from GeneSpring, the classifications are simply gene lists with STRING associated values, where the values designate the cluster name that gene belongs to.

The two functions GSload.genelist and GSsave.genelist will not make a distinction between the two types of gene lists and it is left to the user to determine from the contents of the genelists which of the two types has been loaded.

When a classification is attempted to be loaded into GeneSpring as a proper gene list, the associated values will simply be missing if they are non-numerical.

Value

The GSload.genelist() function will return either a vector of genenames, if the input was a simple list of genes or a data.frame where the gene names are the first column, and the second column is the associated data.

Note

This and any other functions in the GeneSpring package are only useful when using Agilent Technologies Gene Expression software GeneSpring.

Author(s)

Thon de Boer, Agilent Technologies, Santa Clara, CA, USA ⟨thon_deboer@agilent.com⟩

References

For more information on using GeneSpring with R see http://www.chem.agilent.com/scripts/generic.asp?lpage=34733

Examples

```
#Create a simple GeneList
glist <- c("160640_at","101906_at", "104099_at")
#Save the genelist to a file
GSsave.genelist(glist, filename = "GS_R_out.txt")
#Load the genelist again
g2 <- GSload.genelist(filename = "GS_R_out.txt")
#Add some values to the genelist as associated values by making a data.frame out of it
g3 <- data.frame(genes = g2, value=rnorm(3))
#Save the genelist again to the default file ("GS_R_out.txt") and append it to the file
GSsave.genelist(g3, append=TRUE)
#Load the gene list and display
GSload.genelist(filename = "GS_R_out.txt", chunk=2)

#Create classification style gene list
g4 <- data.frame(genes = g2, classification=c("Cluster 1","Cluster 2", "Cluster 1"))
GSsave.genelist(g4, append=FALSE)
#Load the gene list and display
GSload.genelist(filename = "GS_R_out.txt")
```

GSload.int *Read GeneSpring Experiment Interpretation from file*

Description

A GeneSpring Experiment Interpretation is read in, usually from a file, but a connection object (like `stdin()`) can also be passed. The return value can either be an object of class `GSint` or BioConductor `ExpressionSet` representing the Experiment Interpretation from GeneSpring.

Usage

```
gs.int <- GSload.int(filename = "GS_R_in.txt")
expr.set <- GSload.intBC(filename = "GS_R_in.txt", what = "nor")
```

Arguments

`filename`	Name of the file containing the GeneSpring Experiment Interpretation. Can also be a connection object.
`what`	For `GSload.intBC()` only. Indicates what data from the GeneSpring Experiment Interpretation should be used. Valid values for `what` can be `nor` and `raw`, representing Normalized or Raw expression values. The Control values cannot be used.

Details

A GeneSpring Experiment INTERPRETATION is quite distinct from a GeneSpring Experiment. See `GSload.exp()` for more information on GeneSpring Experiments.

A GeneSpring Experiment Interpretation can consist of a maximum of three sets of six (eighteen total) columns for each condition. The sets can be the Normalized, Raw and Control sets and each set can consists of the Average, Min, Max, Standard Error, Standard Deviation and Number of samples per condition. The `GSint` class will be populated based on the names of the columns. The column names can be any of the following:

x.AVERAGE, x.MIN, x.MAX, x.STDERR, x.STDDEV, x.N

Where x can be either N, C or R, representing Normalized, Control and Raw data If there is more than one conditions, each consecutive set is numbered by adding a number to the end of the name. The first column does not have a number and the second will start numbering at 1.

If there is more than one type (i.e if there is both Normalized and Raw data), each value type (AVERAGE, MIN, STDEV etc) is listed first for each data type.

Examples:

N.AVERAGE, R.AVERAGE, N.MIN, R.MIN, N.AVERAGE.1, R.AVERAGE.1, N.MIN.1 etc.

Since the GeneSpring user can determine what data to send to the external program, it is up to the R programmer to determine which columns contains what type of data.

GeneSpring will be able to send multiple objects and it sends all of the objects in one file, one object per "chunk", separated by a delimiter on its own line. The order in which the objects are defined (in the "Input" section of the External Program Interface definition) determines in which chunk the interpretation can be found.

Value

The return value depends on which function was called. The `GSload.int` function returns a object of class `GSint`. The `GSload.intBC` function returns a BioConductor object of class `ExpressionSet`. See the documentation on `GSint` and `ExpressionSet` for more information.

Note

This and any other functions in the GeneSpring package are only useful when using Agilent Technologies Gene Expression software GeneSpring.

Author(s)

Thon de Boer, Agilent Technologies, Santa Clara, CA, USA ⟨thon_deboer@agilent.com⟩

References

For more information on using GeneSpring with R see http://www.chem.agilent.com/scripts/generic.asp?lpage=34733

See Also

`GSload.exp`, `GSsave.exp` `GSint2BC`, `BC2GSint`

GSsave.exp *Save a GeneSpring Interpretation to file*

Description

The normalized values of the GeneSpring Interpretation object will be written to file. If the Interpretation object contains Control values, they will also be used. The experimental parameters and the experiment name will also be included.

Usage

```
GSsave.exp(interpretation, filename = "GS_R_out.txt", EOF = "///", append = TRUE
```

Arguments

interpretation	The GeneSpring Experiment Interpretation object of class GSint.
filename	Name of the file. Can also be a connection object.
EOF	String containing the delimiter used to separate the chunks in the file to be read.
append	Boolean indicating if the experiment should be appended to the output, allowing for multiple objects to be send back to GeneSpring.

Details

A GeneSpring Experiment consists of the Normalized expression values for each sample and optionally the Control values for that normalized value. Each sample or hybridization has its own column (set) of expression values, where the expression values for an Experiment INTERPRETATION represent the mean value of the replicate values for that condition.

GeneSpring will be able to read multiple objects, and it requires the R program to store all objects in one file, one object per "chunk", separated by a delimiter on its own line. The order in which the objects are defined (in the "Output" section of the External Program Interface definition) determines in which chunk the interpretation should be read by GeneSpring.

For more information see the manual entry for GSload.exp.

Value

This function does not return a usable value upon return.

Note

This and any other functions in the GeneSpring package are only useful when using Agilent Technologies Gene Expression software GeneSpring.

Author(s)

Thon de Boer, Agilent Technologies, Santa Clara, CA, USA ⟨thon_deboer@agilent.com⟩

References

For more information on using GeneSpring with R see http://www.chem.agilent.com/scripts/generic.asp?lpage=34733

See Also

GSload.exp, GSload.int

Examples

```
#Use the example data etc. from ExpressionSet
library(Biobase)
data(geneData)
data(geneCov)
covdesc<- list("Covariate 1", "Covariate 2", "Covariate 3")
names(covdesc) <- names(geneCov)
pdata <- new("AnnotatedDataFrame")
pData(pdata) <- geneCov
varLabels(pdata) <- covdesc
eset <- new("ExpressionSet", exprs=geneData, phenoData=pdata)

#Convert the objects into GeneSpring objects
gs.int <- BC2GSint(eset)

#Save the Experiment in GeneSpring format
GSsave.exp(gs.int, filename = "GS_R_out.txt")
```

Index