GEOquery

April 19, 2010

Generic functions   *Generic functions for GEOquery*

Description

The main documentation is in the Class documentation

Author(s)

Sean Davis

See Also

[GEOData-class](#)

Converting   *Convert a GDS data structure to a BioConductor data structure*

Description

Functions to take a GDS data structure from getGEO and coerce it to limma MALists or ExpressionSets.

Usage

```
GDS2MA(GDS,do.log2=FALSE,GPL=NULL,AnnotGPL=TRUE)
GDS2eSet(GDS,do.log2=FALSE,GPL=NULL,AnnotGPL=TRUE)
```

Arguments

GDS	The GDS datastructure returned by getGEO
do.log2	Boolean, should the data in the GDS be log2 transformed before inserting into the new data structure
GPL	Either a GPL data structure (from a call to getGEO) or NULL. If NULL, this will cause a call to getGEO to produce a GPL. The gene information from the GPL is then used to construct the `genes` slot of the resulting limma `MAList` object or the `featureData` slot of the `ExpressionSet` instance.
AnnotGPL	In general, the annotation GPL files will be available for GDS records, so the default is to use these files over the user-submitted GPL files

Details

This function just rearranges one data structure into another. For GDS, it also deals appropriately with making the "targets" list item for the limma data structure and the phenoData slot of ExpressionSets.

Value

GDS2MA   A limma MAList

GDS2eSet   An ExpressionSet object

Author(s)

Sean Davis

References

See the limma and ExpressionSet help in the appropriate packages

Examples

```
## Not run: gds505 <- getGEO('GDS505')
## Not run: MA <- GDS2MA(gds505)
## Not run: eset <- GDS2eSet(gds505)
```

GDS-class   *Class "GDS"*

Description

A class describing a GEO GDS entity

Objects from the Class

Objects can be created by calls of the form new("GDS", ...)

Slots

gpl: Object of class "GPL"

dataTable: Object of class "GEODataTable" containing the data table for the GDS

header: Object of class "list" containing the metadata for the GDS; can be accessed via the Meta accessor

Extends

Class "GEOData", directly.

Methods

No methods defined with class "GDS" in the signature, but methods applying to GEOData are also applicable here.

Author(s)

Sean Davis

See Also

GEOData-class

GEOData-class   *Class "GEOData"*

Description

A virtual class for holding GEO samples, platforms, and datasets

Objects from the Class

Objects can be created by calls of the form `new("GEOData", ...)`.

Slots

`header`: Object of class `"list"` containing metadata

Methods

Accession `signature(object = "GEOData")`: returns the GEO acccession for the current object

Columns `signature(object = "GEOData")`: returns the column descriptions for the current object

Meta `signature(object = "GEOData")`: returns the metadata for the current object

Table `signature(object = "GEOData")`: returns the "Table" for the current object

dataTable `signature(object = "GEOData")`: returns the dataTable (column info and data) for the current object

show `signature(object = "GEOData")`: a convenience method for showing a GEO object

Author(s)

Sean Davis

See Also

GDS-class, GPL-class, GSM-class, GEODataTable-class,

GEODataTable-class *Class "GEODataTable"*

Description

Contains the column descriptions and data for the datatable part of a GEO object

Objects from the Class

Objects can be created by calls of the form `new("GEODataTable", ...)`.

Slots

`columns`: Object of class `"data.frame"` containing information about the columns in the datatable

`table`: Object of class `"data.frame"` containing the actual data

Methods

Columns `signature(object = "GEODataTable")`: get the column portion of the GEO-DataTable

Table `signature(object = "GEODataTable")`: get the table portion of the GEODataTable

show `signature(object = "GEODataTable")`: convenience show method

Author(s)

Sean Davis

---

getGEOfile *Download a file from GEO soft file to the local machine*

Description

This function simply downloads a SOFT format file associated with the GEO accession number given.

Usage

```
getGEOfile(GEO, destdir = tempdir(), AnnotGPL = FALSE, amount = c("full", "brief
```

Arguments

`GEO`	Character string, the GEO accession for download (eg., GDS84, GPL96, GSE2553, or GSM10)
`destdir`	Directory in which to store the resulting downloaded file. Defaults to tempdir()
`AnnotGPL`	A boolean defaulting to FALSE as to whether or not to use the Annotation GPL information. These files are nice to use because they contain up-to-date information remapped from Entrez Gene on a regular basis. However, they do not exist for all GPLs; in general, they are only available for GPLs referenced by a GDS
`amount`	Amount of information to pull from GEO. Only applies to GSE, GPL, or GSM. See details...

Details

This function downloads GEO SOFT files based on accession number. It does not do any parsing. The first two arguments should be fairly self-explanatory, but the last is based on the input to the acc.cgi url at the geo website. In the default "full" mode, the entire SOFT format file is downloaded. Both "brief" and "quick" offer shortened versions of the files, good for "peeking" at the file before a big download on a slow connection. Finally, "data" downloads only the data table part of the SOFT file and is good for downloading a simple EXCEL-like file for use with other programs (a convenience).

Value

Invisibly returns the full path of the downloaded file.

Author(s)

Sean Davis

References

http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi

See Also

getGEO

Examples

```
myfile <- getGEOfile('GDS10')
```

---

getGEO   *Get a GEO object from NCBI or file*

Description

This function is the main user-level function in the GEOquery package. It directs the download (if no filename is specified) and parsing of a GEO SOFT format file into an R data structure specifically designed to make access to each of the important parts of the GEO SOFT format easily accessible.

Usage

```
getGEO(GEO = NULL, filename = NULL, destdir = tempdir(), GSElimits=NULL,
GSEMatrix=TRUE,AnnotGPL=FALSE)
```

Arguments

GEO
: A character string representing a GEO object for download and parsing. (eg., 'GDS505','GSE2','GSM2','GPL96')

filename
: The filename of a previously downloaded GEO SOFT format file or its gzipped representation (in which case the filename must end in .gz). Either one of GEO or filename may be specified, not both. GEO series matrix files are also handled. Note that since a single file is being parsed, the return value is not a list of esets, but a single eset when GSE matrix files are parsed.

destdir
: The destination directory for any downloads. Defaults to the architecture-dependent tempdir. You may want to specify a different directory if you want to save the file for later use. Doing so is a good idea if you have a slow connection, as some of the GEO files are HUGE!

GSElimits
: This argument can be used to load only a contiguous subset of the GSMs from a GSE. It should be specified as a vector of length 2 specifying the start and end (inclusive) GSMs to load. This could be useful for splitting up large GSEs into more manageable parts, for example.

GSEMatrix
: A boolean telling GEOquery whether or not to use GSE Series Matrix files from GEO. The parsing of these files can be many orders-of-magnitude faster than parsing the GSE SOFT format files. Defaults to TRUE, meaning that the SOFT format parsing will not occur; set to FALSE if you for some reason need other columns from the GSE records.

AnnotGPL
: A boolean defaulting to FALSE as to whether or not to use the Annotation GPL information. These files are nice to use because they contain up-to-date information remapped from Entrez Gene on a regular basis. However, they do not exist for all GPLs; in general, they are only available for GPLs referenced by a GDS

Details

getGEO functions to download and parse information available from NCBI GEO (http://www.ncbi.nlm.nih.gov/geo). Here are some details about what is avaible from GEO. All entity types are handled by getGEO and essentially any information in the GEO SOFT format is reflected in the resulting data structure.

From the GEO website:

The Gene Expression Omnibus (GEO) from NCBI serves as a public repository for a wide range of high-throughput experimental data. These data include single and dual channel microarray-based experiments measuring mRNA, genomic DNA, and protein abundance, as well as non-array techniques such as serial analysis of gene expression (SAGE), and mass spectrometry proteomic data. At the most basic level of organization of GEO, there are three entity types that may be supplied by users: Platforms, Samples, and Series. Additionally, there is a curated entity called a GEO dataset.

A Platform record describes the list of elements on the array (e.g., cDNAs, oligonucleotide probe-sets, ORFs, antibodies) or the list of elements that may be detected and quantified in that experiment (e.g., SAGE tags, peptides). Each Platform record is assigned a unique and stable GEO accession number (GPLxxx). A Platform may reference many Samples that have been submitted by multiple submitters.

A Sample record describes the conditions under which an individual Sample was handled, the ma-nipulations it underwent, and the abundance measurement of each element derived from it. Each Sample record is assigned a unique and stable GEO accession number (GSMxxx). A Sample entity must reference only one Platform and may be included in multiple Series.

A Series record defines a set of related Samples considered to be part of a group, how the Samples are related, and if and how they are ordered. A Series provides a focal point and description of the experiment as a whole. Series records may also contain tables describing extracted data, summary conclusions, or analyses. Each Series record is assigned a unique and stable GEO accession number (GSExxx).

GEO DataSets (GDSxxx) are curated sets of GEO Sample data. A GDS record represents a col-lection of biologically and statistically comparable GEO Samples and forms the basis of GEO's suite of data display and analysis tools. Samples within a GDS refer to the same Platform, that is, they share a common set of probe elements. Value measurements for each Sample within a GDS are assumed to be calculated in an equivalent manner, that is, considerations such as background processing and normalization are consistent across the dataset. Information reflecting experimental design is provided through GDS subsets.

Value

An object of the appropriate class (GDS, GPL, GSM, or GSE) is returned. If the GSEMatrix option is used, then a list of ExpressionSet objects is returned, one for each SeriesMatrix file associated with the GSE accesion. If the filename argument is used in combination with a GSEMatrix file, then the return value is a single ExpressionSet.

Warning

Some of the files that are downloaded, particularly those associated with GSE entries from GEO are absolutely ENORMOUS and parsing them can take quite some time and memory. So, particularly when working with large GSE entries, expect that you may need a good chunk of memory and that coffee may be involved when parsing....

Author(s)

Sean Davis

See Also

getGEOfile

Examples

```
gds <- getGEO("GDS10")
gds
```

---

getGEOSuppFiles   *Get Supplemental Files from GEO*

Description

NCBI GEO allows supplemental files to be attached to GEO Series (GSE), GEO platforms (GPL), and GEO samples (GSM). This function "knows" how to get these files based on the GEO accession. No parsing of the downloaded files is attempted, since the file format is not generally knowable by the computer.

Usage

```
getGEOSuppFiles(GEO, makeDirectory = TRUE, baseDir = getwd())
```

Arguments

GEO   A GEO accession number such as GPL1073 or GSM1137

makeDirectory
                 Should a "subdirectory" for the downloaded files be created? Default is TRUE.
                 If FALSE, the files will be downloaded directly into the baseDir.

baseDir   The base directory for the downloads. Default is the current working directory.

Details

Again, just a note that the files are simply downloaded.

Value

A data frame is returned invisibly with rownames representing the full path of the resulting downloaded files and the records in the data.frame the output of file.info for each downloaded file.

Author(s)

Sean Davis <sdavis2@mail.nih.gov>

Examples

```
a <- getGEOSuppFiles('GSM1137')
a
```

`GPL-class`   *Class "GPL"*

Description

Contains a full GEO Platform entity

Objects from the Class

Objects can be created by calls of the form `new("GPL", ...)`.

Slots

`dataTable:` Object of class `"GEODataTable"`

`header:` Object of class `"list"` containing metadata associated with the GPL

Extends

Class `"GEOData"`, directly.

Methods

No methods defined with class "GPL" in the signature, but methods applicable to GEOData are also applicable here.

Author(s)

Sean Davis

See Also

[GEOData-class](#)

`GSE-class`   *Class "GSE"*

Description

Contains a GEO Series entity

Objects from the Class

Objects can be created by calls of the form `new("GSE", ...)`.

Slots

`header:` Object of class `"list"` containing metadata for the series

`gsms:` Object of class `"list"` containing a list of items of class `GSM` associated with the series

`gpls:` Object of class `"list"` containing a list of items of class `GPL` associate with the series

Methods

> **GPLList** `signature(object = "GSE")`: returns a list, each item of the list being a `GPL`
> object
>
> **GSMList** `signature(object = "GSE")`: returns a list, each item of the list being a `GSM`
> object
>
> **Meta** `signature(object = "GSE")`: returns a list, the metadata associated with the GSE

Author(s)

> Sean Davis

See Also

> `GPL-class`,`GSM-class`

`GSM-class`   *Class "GSM"*

Description

> A class containing a GEO Sample entity

Objects from the Class

> Objects can be created by calls of the form `new("GSM", ...)`.

Slots

> `dataTable`**:** Object of class `"GEODataTable"`
>
> `header`**:** Object of class `"list"` containing the metadata associated with the sample

Extends

> Class `"GEOData"`, directly.

Methods

> No methods defined with class "GSM" in the signature, but methods that apply to the `GEOData`
> also apply here.

Author(s)

> Sean Davis

See Also

> `GEOData-class`

---

gunzip   *Gunzip a file*

Description

gunzip a file

Usage

```
gunzip(filename, destname = gsub("[.]gz$", "", filename), overwrite = FALSE, rem
```

Arguments

filename	The filename to be unzipped
destname	The destination file
overwrite	Boolean indicating whether or not to overwrite a destfile of the same name
remove	Boolean indicating whether or not to remove the original file after completion
BFR.SIZE	The size of the read buffer....

Details

This function was stripped out of R.utils due to breaking some stuff on the bioconductor build machine.

Value

Invisibly, the number of bytes read.

Author(s)

Original author: Henrik Bengtsson

See Also

gzfile

---

parseGEO   *Parse GEO text*

Description

Workhorse GEO parsers.

Usage

```
parseGEO(fname, GSElimits)
parseGPL(fname)
parseGDS(fname)
parseGSE(fname, GSElimits)
parseGSM(fname)
```

Arguments

fname   The filename of a SOFT format file. If the filename ends in .gz, a gzfile() connection is used to read the file directly.

GSElimits   Used to limit the number of GSMs parsed into the GSE object; useful for memory management for large GSEs.

Details

These are probably not useful to the end-user. Use getGEO to access these functions. parseGEO simply delegates to the appropriate specific parser. There should be no reason to use the parseGPL, parseGDS, parseGSE, or parseGSM functions directly.

Value

parseGEO returns an object of the associated type. For example, if it is passed the text from a GDS entry, a GDS object is returned.

Author(s)

Sean Davis

See Also

getGEO

Index