Package 'GEOquery'

November 17, 2026

Type Package

Title Get data from NCBI Gene Expression Omnibus (GEO)

Version 2.81.36

Date 2026-10-23

BugReports https://github.com/seandavi/GEOquery/issues/new

Depends methods, Biobase

Imports readr (>= 1.3.1), xml2, dplyr, data.table, tidyr, magrittr, curl, rentrez, R.utils, SummarizedExperiment, S4Vectors, httr2, jsonlite, rlang, BiocFileCache

Suggests knitr, rmarkdown, BiocGenerics, testthat, covr, markdown, quarto, SingleCellExperiment, TENxIO, anndataR, BiocIO, Seurat, limma

VignetteBuilder quarto

URL https://github.com/seandavi/GEOquery, http://seandavi.github.io/GEOquery, http://seandavi.github.io/GEOquery/

biocViews GeneExpression, Transcriptomics, Microarray, RNASeq, Sequencing, SingleCell, DataImport, ThirdPartyClient, OneChannel, TwoChannel, SAGE

Description The NCBI Gene Expression Omnibus (GEO) is a public repository of high-throughput functional genomics data, including microarray, RNA-seq, and single-cell experiments. GEOquery is the bridge between GEO and Bioconductor: it downloads and parses GEO Series (GSE), Sample (GSM), Platform (GPL), and DataSet (GDS) records. By default it parses GEO Series Matrix files into Bioconductor 'SummarizedExperiment' objects; it can also parse the full SOFT format into GEOquery's own S4 classes, retrieve NCBI-computed RNA-seq quantifications, download supplementary files, and search GEO.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.0.0

git_url https://git.bioconductor.org/packages/GEOquery

git_branch devel

git_last_commit b8125c8

git_last_commit_date 2026-10-23

Repository Bioconductor 3.24

Date/Publication 2026-11-16

Author Sean Davis [aut, cre] (ORCID: <https://orcid.org/0000-0002-8991-6458>)

Maintainer Sean Davis <seandavi@gmail.com>

Contents

as_SummarizedExperiment . . . 3
browseGEOAccession . . . 3
browseWebsiteRNASeqSearch . . . 4
clearGEOCache . . . 4
coercion . . . 5
extractFilenameFromDownloadURL . . . 6
GDS-class . . . 6
geoCache . . . 7
GEOData-accessors . . . 7
GEOData-class . . . 8
GEODataTable-class . . . 9
geoSingleCellManifest . . . 9
geoSingleCellUnits . . . 10
geoToSRA . . . 11
getDirListing . . . 12
getGEO . . . 12
getGEOfile . . . 15
getGEOSeriesFileListing . . . 16
getGEOSingleCell . . . 17
getGEOSuppFiles . . . 19
getGEOSuppFileURL . . . 20
getGSEDataTables . . . 21
getGSEDownloadPageURLs . . . 22
getRNAQuantAnnotationURL . . . 22
getRNAQuantRawCountsURL . . . 23
getRNASeqData . . . 23
getRNASeqQuantGenomeInfo . . . 24
getRNASeqQuantResults . . . 25
GPL-class . . . 25
GSE-class . . . 26
GSM-class . . . 26
hasRNASeqQuantifications . . . 27
parseGEO . . . 27
parseGSEMatrix . . . 28
readGEOSingleCell . . . 29
readRNAQuantAnnotation . . . 30
readRNAQuantRawCounts . . . 30
searchFieldsGEO . . . 31
searchGEO . . . 31
urlExtractRNASeqQuantGenomeInfo . . . 32
urlForAccession . . . 33

Index **34**

`as_SummarizedExperiment` *Coerce a GEOquery ExpressionSet to a SummarizedExperiment*

Description

A thin wrapper around `SummarizedExperiment::makeSummarizedExperimentFromExpressionSet()` used by `getGEO(..., returnType = "SummarizedExperiment")`, and available directly so existing ExpressionSet results can be modernized without re-downloading.

Usage

```
as_SummarizedExperiment(eset)
```

Arguments

`eset`	An `ExpressionSet`, e.g. an element returned by `getGEO()` for a GSE Series Matrix file.

Value

A `SummarizedExperiment`.

Examples

```
## Not run:
  gse <- getGEO("GSE2553")[[1]]
  se <- as_SummarizedExperiment(gse)

## End(Not run)
```

`browseGEOAccession` *Open the GEO page for a given accession*

Description

Sometimes, you just need to see the GEO website page for a GEO accession. This function opens the GEO page for a given accession number in the default browser.

Usage

```
browseGEOAccession(geo)
```

Arguments

`geo`	A GEO accession number

See Also

`urlForAccession`

Examples

```
## Not run:
browseGEOAccession("GSE262484")

## End(Not run)
```

`browseWebsiteRNASeqSearch` *Browse GEO search website for RNA-seq datasets*

Description

This function opens a browser window to the NCBI GEO website with a search for RNA-seq datasets. It is included as a convenience function to remind users of how to search for RNA-seq datasets using the NCBI GEO website and an "rnaseq counts" filter.

Usage

```
browseWebsiteRNASeqSearch()
```

Examples

```
## Not run:
browseWebsiteRNASeqSearch()

## End(Not run)
```

`clearGEOCache` *Clear the GEOquery download cache*

Description

Remove all entries from the persistent download cache (see geoCache).

Usage

```
clearGEOCache()
```

Value

NULL, invisibly.

See Also

geoCache

coercion *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.

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
getGPL	A boolean defaulting to TRUE as to whether or not to download and include GPL information when converting to ExpressionSet or MAList. You may want to set this to FALSE if you know that you are going to annotate your featureData using Bioconductor tools rather than relying on information provided through NCBI GEO. Download times can also be greatly reduced by specifying FALSE.

Details

GDS2MA() is **deprecated** and will be removed in a future release: the MAList return type is superseded by ExpressionSet/SummarizedExperiment, which can be coerced to whatever downstream analysis needs. Use GDS2eSet() instead (and convert to an MAList with limma if you still need one).

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)
```

`extractFilenameFromDownloadURL` *Extract filename from a GEO download URL*

Description

This function extracts the filename from a GEO download URL. The filename is expected to be a query parameter called "file". If the query parameter is not found, the function returns NULL.

Usage

```
extractFilenameFromDownloadURL(url)
```

Arguments

`url`	A GEO download URL

Details

The idea is to use this function to extract filenames that contain important metadata from the GEO RNA-seq quantification.

In particular, the filename is expected to contain the genome build and species information that we can attach to the SummarizedExperiment.

Value

A character vector with the filename

`GDS-class` *Class 'GDS'*

Description

A class describing a GEO GDS entity

Objects from the Class

Objects of this class are returned by `getGEO`; they are not normally constructed directly.

Author(s)

Sean Davis

See Also

GEOData-class

---

geoCache *GEOquery download cache*

Description

Return the BiocFileCache object that backs GEOquery's persistent download cache (see clearGEOCache). The cache is used by the download functions only when options(GEOquery.cache = TRUE) is set; its location defaults to tools::R_user_dir("GEOquery", "cache") and can be overridden with options(GEOquery.cache.path = ...).

Usage

```
geoCache()
```

Value

A BiocFileCache object.

See Also

clearGEOCache

GEOData-accessors *Accessors for GEOquery objects*

Description

Accessor generics for the S4 objects returned by getGEO when parsing SOFT-format records (GSE, GSM, GPL, GDS). Use these rather than reaching into slots directly.

Arguments

object	A GEOquery S4 object (GSE, GSM, GPL, GDS, or GEODataTable).

Details

Meta(object) The record metadata as a named list (title, submission dates, sample/platform attributes, and so on).

Accession(object) The GEO accession (the geo_accession metadata field).

Table(object) The data table as a data.frame – for example the measurement table of a GSM or the probe annotation of a GPL.

Columns(object) A data.frame describing the columns of Table(object).

dataTable(object) The underlying GEODataTable object, which holds both Table() and Columns().

GSMList(object) For a GSE, the list of its GSM sample objects.

GPLList(object) For a GSE, the list of its GPL platform objects.

Value

`Meta()` a list; `Accession()` a character string; `Table()` and `Columns()` data.frames; `dataTable()` a `GEODataTable`; `GSMList()` and `GPLList()` named lists.

Author(s)

Sean Davis

See Also

`GEOData-class`, `getGEO`

Examples

```
## Not run:
  gsm <- getGEO("GSM11805")
  Meta(gsm)$title
  head(Table(gsm))
  Columns(gsm)

  gse <- getGEO("GSE781", GSEMatrix = FALSE)
  names(GSMList(gse))

## End(Not run)
```

GEOData-class *Class 'GEOData'*

Description

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

Objects from the Class

Objects of this class are returned by `getGEO`; they are not normally constructed directly.

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 of this class are returned by `getGEO`; they are not normally constructed directly.

Author(s)

Sean Davis

geoSingleCellManifest *Inventory the single-cell supplementary files of a GEO Series or Sample*

Description

Lists the supplementary files attached to a GSE (or a single GSM) and classifies each by single-cell format (10x Matrix Market triplet, 10x HDF5, AnnData h5ad, loom, Seurat rds, tar archive, or other), extracting the GSM sample id where present. This lets you see what a single-cell study contains – and how 10x triplets group by sample – before downloading potentially many gigabytes.

Usage

```
geoSingleCellManifest(GEO, samples = NULL)
```

Arguments

`GEO`	A GEO Series (`"GSE..."`) or Sample (`"GSM..."`) accession, e.g. "GSE132771" or "GSM3891612".
`samples`	Optional character vector of GSM ids. For a GSE, restricts the inventory to these samples; when the series level has no loadable units this also avoids enumerating the whole series. Ignored when `GEO` is a GSM.

Details

For a GSE, the series-level suppl directory is inventoried first. Many single-cell studies ship only a `GSE..._RAW.tar` there, with the loadable per-sample files (10x triplets, h5, h5ad) living in each sample's own GSM suppl directory. When the series level yields no loadable units, the manifest falls back to enumerating the series' samples (via `getGEO`) and inventorying each GSM suppl directory. Pass a GSM accession to inventory just that one sample.

No files are downloaded. The result feeds the single-cell readers (see ADR-0004); reading itself uses Bioconductor importers (TENxIO, anndataR) that are optional dependencies.

The `platform` column (GPL accession per sample) is populated when the manifest is built from the GSM level – the common single-cell case, and the one that matters, since a GSE can span multiple

platforms (e.g. GSE132771 mixes mouse and human). It is the grouping used by `getGEOSingleCell(by = "platform")`. It is `NA` for a single GSM, and for whole-study files attached at the series level (which have no GSM).

Value

A data.frame with columns `fname`, `sample` (GSM id or NA), `platform` (GPL accession or NA), `format`, `role`, and `url`. Zero rows if nothing is found.

See Also

`getGEOSuppFiles`, `getGEOSingleCell`

Examples

```
## Not run:
  geoSingleCellManifest("GSE132771")        # GSE: falls back to GSM level
  geoSingleCellManifest("GSM3891612")       # a single sample

## End(Not run)
```

---

geoSingleCellUnits *Group a single-cell manifest into loadable units*

Description

Collapses a `geoSingleCellManifest` into one row per unit and reports completeness. A 10x Matrix Market unit groups a sample's matrix, barcodes, and features files and is "complete" only when all three are present; every other format is one unit per file. The `loadable` column flags units a built-in reader can consume – complete 10x Matrix Market, 10x HDF5, and AnnData h5ad. loom and Seurat `.rds` are reported but not loadable (read them with their native packages).

Usage

```
geoSingleCellUnits(manifest)
```

Arguments

`manifest`	A data.frame returned by `geoSingleCellManifest()`.

Value

A data.frame with columns `unit` (the grouping key), `sample`, `platform` (GPL, or NA), `format`, `n_files`, `status`, and `loadable`.

See Also

`geoSingleCellManifest`

Examples

```
## Not run:
  m <- geoSingleCellManifest("GSE161228")
  geoSingleCellUnits(m)

## End(Not run)
```

`geoToSRA` *Resolve a GEO accession to its SRA accessions*

Description

Maps a GEO Series (`GSE`) or Sample (`GSM`) to the underlying Sequence Read Archive (SRA) accessions — study (`SRP`), experiment (`SRX`), run (`SRR`), and sample (`SRS`) — by cross-referencing NCBI Entrez (`gds -> sra`). This is the accession-linking bridge to raw sequencing data; it downloads no sequence files.

Usage

```
geoToSRA(geo)
```

Arguments

`geo`	character(1), a GEO accession (`GSE...` or `GSM...`).

Details

Uses the NCBI Entrez API through **rentrez**. An `ENTREZ_KEY` raises the rate limit; see searchGEO for how to set one.

Value

A `data.frame` with one row per SRA run and columns `geo_accession`, `srp`, `srx`, `srr`, `srs`. Returns a zero-row data.frame (same columns) when the accession has no linked SRA records.

See Also

searchGEO

Examples

```
## Not run:
geoToSRA("GSE164073")

## End(Not run)
```

getDirListing *get a directory listing from NCBI GEO*

Description

This one makes some assumptions about the structure of the HTML response returned.

Usage

```
getDirListing(url)
```

Arguments

url	A URL, assumed to return an NCBI-formatted index page

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,
  getGPL = TRUE,
  parseCharacteristics = TRUE,
  returnType = c("SummarizedExperiment", "ExpressionSet"),
  encoding = NULL,
  token = NULL
)
```

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

`getGPL` A boolean defaulting to TRUE as to whether or not to download and include GPL information when getting a GSEMatrix file. You may want to set this to FALSE if you know that you are going to annotate your featureData using Bioconductor tools rather than relying on information provided through NCBI GEO. Download times can also be greatly reduced by specifying FALSE.

`parseCharacteristics` A boolean defaulting to TRUE as to whether or not to parse the characteristics information (if available) for a GSE Matrix file. Set this to FALSE if you experience trouble while parsing the characteristics.

`returnType` One of "SummarizedExperiment" (default) or "ExpressionSet". For GSE Series Matrix results, controls whether each entity is returned as a SummarizedExperiment or an ExpressionSet. SOFT-format results (GDS/GPL/GSM/GSE S4 objects) are unaffected. As of this release the default is "SummarizedExperiment"; pass returnType = "ExpressionSet" for the previous behavior.

`encoding` Optional character encoding for reading the downloaded GEO file, one of "unknown" (the default; let the reader auto-detect), "UTF-8", or "Latin-1". Most GEO files are UTF-8/ASCII, but a few carry Latin-1 bytes that are otherwise mis-decoded; set `encoding = "Latin-1"` for those. Applies for the duration of the call only. Equivalent to setting `options(GEOquery.encoding = ...)` globally.

`token` Optional NCBI GEO reviewer access token (character(1)) for fetching a private/embargoed record. Obtain it from the "Reviewer access" link on the private GSE's GEO page. Because private records are not published to the GEO FTP tree, supplying a token forces the SOFT (`acc.cgi`) path: for a GSE this returns a `GSE` S4 object (as with `GSEMatrix = FALSE`), not a `SummarizedExperiment` / `ExpressionSet`. See ADR-0007.

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 probesets, 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 manipulations 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 collection 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 SummarizedExperiment objects is returned by default (or ExpressionSet objects if `returnType = "ExpressionSet"`), one for each SeriesMatrix file associated with the GSE accession.

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

```
## Not run:

gds <- getGEO('GDS10')
gds

gse <- getGEO('GSE10')
# Returns a list, so look at first item

gse[[1]]


## End(Not run)
```

`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", "quick", "data"),
  token = NULL
)
```

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...
`token`	Optional NCBI GEO reviewer access token (character(1)) for a private/embargoed record. Private records are not published to the GEO FTP tree, so a token forces the CGI (`acc.cgi`) SOFT path and appends the token to the request. See `getGEO`.

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')
```

---

getGEOSeriesFileListing

GSE Supplemental file listing

Description

The GEO Series records often have one or more supplemental files. In most cases, those files are archived as '.tar' files, the contents of which are only available in a file listing file not present on the website for download.

Usage

```
getGEOSeriesFileListing(GSE)
```

Arguments

GSE	character(1) the GSE accession

Details

This function reads that file listing file and returns the results as a data.frame.

Value

A data.frame with 5 columns. See example.

Examples

```
## Not run:
getGEOSeriesFileListing('GSE288770')


## End(Not run)
```

`getGEOSingleCell` *Download and read the single-cell data of a GEO Series or Sample*

Description

High-level, best-effort convenience wrapper: inventories the GSE (or GSM) (`geoSingleCellManifest`), groups files into loadable units (`geoSingleCellUnits`), downloads each loadable unit, reads it with `readGEOSingleCell`, and returns the results. It reports which units it loads and which it skips.

Usage

```
getGEOSingleCell(
  GEO,
  samples = NULL,
  format = NULL,
  by = c("sample", "platform", "all"),
  as = c("SingleCellExperiment", "Seurat"),
  addSampleMeta = TRUE,
  destdir = tempdir()
)
```

Arguments

`GEO`	A GEO Series (`"GSE..."`) or Sample (`"GSM..."`) accession, e.g. "GSE132771" or "GSM3891612".
`samples`	Optional character vector of GSM ids to restrict to. Ignored when `GEO` is itself a GSM.
`format`	Optional format(s) to restrict to ("10x_mtx", "10x_h5", "h5ad", "rds").
`by`	One of `"sample"` (default), `"platform"`, or `"all"` – how to group the loaded samples into the return value. See Details.
`as`	Output class, one of "SingleCellExperiment" (default) or "Seurat" (coerced at the boundary via the Seurat package, an optional dependency).
`addSampleMeta`	Logical, default `TRUE`. Attach each sample's GEO metadata – the parsed `characteristics_*` fields, `title`, and source name – as constant per-cell `colData` columns, so a sample's annotation (age, sex, genotype, tissue, treatment, ...) travels with its cells (including after combining with `by = "platform"`/`"all"`). Added columns are prefixed `"sample."` to avoid colliding with the importer's own `colData`. The metadata comes from the Series Matrix pData (for a GSE) or the GSM SOFT record (for a lone GSM), so it costs one additional small download;

whole-study files with no GSM get nothing. Metadata lookup is best-effort: a failure warns and is skipped, never breaking the data load. Set FALSE to skip it.

destdir — Download destination directory.

Details

This handles common, well-structured layouts (clean per-sample 10x, h5ad, or a saved object in .rds), including the very common case where the series ships only a _RAW.tar and the per-sample files live in each GSM suppl directory (the manifest falls back to the GSM level automatically). You may also pass a single GSM accession to load just that sample. It does NOT handle every GSE: loom files, files available *only* inside a _RAW.tar archive, and idiosyncratic layouts (e.g. a single combined matrix for many samples) are out of scope – use the manifest plus readGEOSingleCell() directly for those.

Grouping (by). A GEO Series has two natural layers – it can span multiple platforms (GPLs), each holding many samples (GSMs) – and the platform is the feature-compatibility boundary (samples in one platform share a feature space; across platforms they generally do not). by chooses the return shape, and the shape is fixed by the argument (not the data):

"sample" **(default)** a named list with one object per sample (or per whole-study file).

"platform" a named list keyed by platform (GPL), each entry the samples of that platform combined into one object. The honest answer for a multi-platform study; a single-platform study yields a length-1 list. Samples with unknown platform are returned individually.

"all" a single object with every sample combined. Errors if the samples share no common features (e.g. a study mixing organisms) – use "platform" for those.

Combining (for "platform"/"all") restricts to the features common to the group and reconciles per-sample feature annotation so binding succeeds across CellRanger versions; whole-study single-file formats already hold one object, so grouping is effectively a no-op for them.

Value

Depends on by: a named list of objects per sample ("sample"); a named list of combined objects per platform ("platform"); or a single combined object ("all"). Each object is a SingleCellExperiment, or a Seurat object when as = "Seurat".

See Also

geoSingleCellManifest, readGEOSingleCell

Examples

```
## Not run:
  sce <- getGEOSingleCell("GSM3891612")                  # one sample
  per_sample <- getGEOSingleCell("GSE132771")            # list by GSM
  per_platform <- getGEOSingleCell("GSE132771", by = "platform")
  # -> list(GPL21103 = <mouse SCE>, GPL24676 = <human SCE>)

  # Each sample's GEO characteristics ride along in colData (default):
  sce <- getGEOSingleCell("GSE125708", by = "all")
  SummarizedExperiment::colData(sce)[, grep("^sample\\.", colnames(
    SummarizedExperiment::colData(sce)))]
  # sample.title, sample.age.ch1, sample.Sex.ch1, sample.tissue.ch1, ...

## End(Not run)
```

`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(),
  fetch_files = TRUE,
  filter_regex = NULL,
  quiet = getOption("GEOquery.quiet", FALSE)
)
```

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.
`fetch_files`	logical(1). If TRUE, then actually download the files. If FALSE, just return the filenames that would have been downloaded. Useful for testing and getting a list of files without actual download.
`filter_regex`	A character(1) regular expression that will be used to filter the filenames from GEO to limit those files that will be downloaded. This is useful to limit to, for example, bed files only.
`quiet`	logical(1). If TRUE, suppress informational messages such as "No supplemental files found" and "Using locally cached version". Defaults to the `GEOquery.quiet` option, or FALSE.

Details

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

Value

If fetch_files=TRUE, 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. If fetch_files=FALSE, a data.frame of URLs and filenames is returned.

Author(s)

Sean Davis sdavis2@mail.nih.gov

Examples

```
## Not run:

a <- getGEOSuppFiles('GSM1137', fetch_files = FALSE)
a

# with a set of single-cell RNA-seq data
a <- getGEOSuppFiles('GSE161228', fetch_files = FALSE)
a


## End(Not run)
```

`getGEOSuppFileURL` *Get GEO supplemental file URL for a given GEO accession*

Description

Get GEO supplemental file URL for a given GEO accession

Usage

```
getGEOSuppFileURL(GEO)
```

Arguments

`GEO`	A GEO accession, e.g. a Series ("GSE..."), Sample ("GSM..."), or Platform ("GPL...") id. The accession type determines which FTP `suppl/` directory URL is returned.

Value

A character(1) URL of the accession's supplementary-file directory on the NCBI GEO FTP site.

Examples

```
# an example of a GEO supplemental file URL
# with a set of single-cell RNA-seq data
url = getGEOSuppFileURL("GSE161228")
url

## Not run:
  browseURL(url)

## End(Not run)
```

`getGSEDataTables` *Get GSE data tables from GEO into R data structures.*

Description

In some cases, instead of individual sample records (GSM) containing information regarding sample phenotypes, the GEO Series contains that information in one or more data tables attached at the Series level. An example is given by GSE3494, where there are two data tables with important information contained within them. Series-level per-cell annotation tables from single-cell studies (the `!series_table` blocks in SOFT format, e.g. the "Listing of Individual Cells" table in GSE98638) are exposed here as well. Using getGEO with the standard parameters downloads the GSEMatrix file which, unfortunately, does not contain the information in the data tables. This function simply downloads the "header" information from the GSE record and parses out the data tables into R data.frames.

Usage

```
getGSEDataTables(GSE)
```

Arguments

`GSE`	The GSE identifier, such as "GSE3494".

Value

A list of data.frames, one per `<Data-Table>` in the Series record (a Series may carry zero, one, or several). Each data.frame's column names are taken from the table's column definitions.

Author(s)

Sean Davis sdavis2@mail.nih.gov

See Also

`getGEO`

Examples

```
## Not run:

dfl = getGSEDataTables('GSE3494')
lapply(dfl,head)



## End(Not run)
```

`getGSEDownloadPageURLs` *get all download links from a GEO accession*

Description

This function gets all download links from a GEO accession number.

Usage

```
getGSEDownloadPageURLs(gse)
```

Arguments

`gse`	GEO accession number

Value

A character vector with all download links

`getRNAQuantAnnotationURL` *Get the RNA-seq quantification annotation link*

Description

This function extracts the link to the RNA-seq quantification annotation file from a geoDownloadLinks object.

Usage

```
getRNAQuantAnnotationURL(links)
```

Arguments

`links`	A geoDownloadLinks object

Value

A character vector with the link to the annotation file

getRNAQuantRawCountsURL *Get the link to the raw counts file from GEO*

Description

This function extracts the link to the raw counts file from a geoDownloadLinks object.

Usage

```
getRNAQuantRawCountsURL(links)
```

Arguments

links	A geoDownloadLinks object

Value

A character vector with the link to the raw counts file

getRNASeqData *Get GEO RNA-seq quantifications as a SummarizedExperiment object*

Description

For human and mouse GEO datasets, NCBI GEO attempts to process the raw data and provide quantifications in the form of raw counts and an annotation file. This function downloads the raw counts and annotation files from GEO and merges that with the metadata from the GEO object to create a SummarizedExperiment.

Usage

```
getRNASeqData(accession)
```

Arguments

accession	GEO accession number

Details

A major barrier to fully exploiting and reanalyzing the massive volumes of public RNA-seq data archived by SRA is the cost and effort required to consistently process raw RNA-seq reads into concise formats that summarize the expression results. To help address this need, the NCBI SRA and GEO teams have built a pipeline that precomputes RNA-seq gene expression counts and delivers them as count matrices that may be incorporated into commonly used differential expression analysis and visualization software.

The pipeline processes RNA-seq data from SRA using the HISAT2 aligner and and then generates gene expression counts using the featureCounts program.

See the GEO documentation for more details.

Value

A SummarizedExperiment object with the raw counts as the counts assay, the annotation as the rowData, and the metadata from GEO as the colData.

Examples

```
## Not run:
se <- getRNASeqData("GSE164073")
se


## End(Not run)
```

`getRNASeqQuantGenomeInfo` *Extract genome build and species for GEO RNA-seq quantification*

Description

This function extracts the genome build and species information for a GEO RNA-seq quantification.

Usage

```
getRNASeqQuantGenomeInfo(gse)
```

Arguments

`gse`	GEO accession number

Value

A character vector with the genome build and species information

Examples

```
## Not run:
getRNASeqQuantGenomeInfo("GSE164073")


## End(Not run)
```

getRNASeqQuantResults *Get RNA-seq quantification and annotation from GEO*

Description

This function downloads the raw counts and annotation files from GEO for a given GEO accession number.

Usage

```
getRNASeqQuantResults(gse)
```

Arguments

gse	GEO accession number

Value

A list with two elements: quants (a matrix of raw counts) and annotation (a data frame of annotation information).

GPL-class *Class 'GPL'*

Description

Contains a full GEO Platform entity

Objects from the Class

Objects of this class are returned by getGEO; they are not normally constructed directly.

Author(s)

Sean Davis

See Also

GEOData-class

GSE-class *Class 'GSE'*

Description

Contains a GEO Series entity

Objects from the Class

Objects of this class are returned by `getGEO`; they are not normally constructed directly.

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 of this class are returned by `getGEO`; they are not normally constructed directly.

Author(s)

Sean Davis

See Also

`GEOData-class`

hasRNASeqQuantifications *Does a GEO accession have RNA-seq quantifications?*

Description

This function checks if a GEO accession number has RNA-seq quantifications available. It does this by checking if the GEO accession number has a "RNA-Seq raw counts" link available on the GEO download page.

Usage

```
hasRNASeqQuantifications(accession)
```

Arguments

accession	GEO accession number

Value

TRUE if the GEO accession number has RNA-seq quantifications available, FALSE otherwise.

Examples

```
hasRNASeqQuantifications("GSE164073")
```

parseGEO *Parse GEO text*

Description

Workhorse GEO parsers.

Usage

```
parseGEO(
  fname,
  GSElimits,
  destdir = tempdir(),
  AnnotGPL = FALSE,
  getGPL = TRUE,
  parseCharacteristics = TRUE
)
```

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.
destdir	The destination directory into which files will be saved (to be used for caching)
AnnotGPL	Fetch the annotation GPL if available
getGPL	Fetch the GPL associated with a GSEMatrix entity (should remain TRUE for all normal use cases)
parseCharacteristics	Whether or not to parse the characteristics information (if available) for a GSE Matrix file. Set to FALSE if you experience trouble parsing the characteristics.

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

---

parseGSEMatrix *Parse a GSE mstrix file*

Description

Not meant for user calling, but parses a single GSEMatrix file.

Usage

```
parseGSEMatrix(
  fname,
  AnnotGPL = FALSE,
  destdir = tempdir(),
  getGPL = TRUE,
  parseCharacteristics = TRUE
)
```

Arguments

fname	filename
AnnotGPL	set to TRUE to get the annotation GPL version
destdir	the destdination directory for download
getGPL	whether or not to get the GPL associated
parseCharacteristics	Whether or not to do full 'characteristic' parsing

---

readGEOSingleCell *Read a single-cell file (or 10x triplet) into a SingleCellExperiment*

Description

Low-level reader: given already-downloaded local file(s), dispatch on format to the appropriate Bioconductor importer and return a `SingleCellExperiment` (or a `Seurat` object with `as = "Seurat"`). Use this for full control; see `getGEOSingleCell` for the high-level convenience wrapper.

Usage

```
readGEOSingleCell(x, format = NULL, as = c("SingleCellExperiment", "Seurat"))
```

Arguments

x	A path to a single file (`.h5`/`.h5ad`/`.rds`), a directory containing a 10x triplet, or a character vector of the triplet files.
format	One of "10x_mtx", "10x_h5", "h5ad", "rds". If NULL (default), guessed from `x`.
as	Output class, one of "SingleCellExperiment" (default) or "Seurat" (coerced via the Seurat package, an optional dependency).

Details

Supported formats: `"10x_mtx"` (a directory, or the matrix/barcodes/ features files, read via TENxIO), `"10x_h5"` (CellRanger HDF5, TENxIO), `"h5ad"` (AnnData, anndataR), and `"rds"` (a saved `SingleCellExperiment` or `Seurat` object; detected by class). loom is not supported – read it with `LoomExperiment` directly.

Value

A `SingleCellExperiment`, or a `Seurat` object if `as = "Seurat"`.

See Also

`getGEOSingleCell`, `geoSingleCellManifest`

`readRNAQuantAnnotation` *Read RNA-seq quantification annotation from GEO*

Description

This function reads the annotation file from a GEO link. The annotation file is expected to be a tab-separated file with the first column containing the gene IDs and the remaining columns containing the annotation information.

Usage

```
readRNAQuantAnnotation(link)
```

Arguments

`link`	A link to the annotation file

Value

A data frame of annotation information with gene IDs as row names

`readRNAQuantRawCounts` *Read raw counts from GEO*

Description

This function reads the raw counts from a GEO link. The raw counts are expected to be in a tab-separated file with the first column containing the gene IDs and the remaining columns containing the raw counts.

Usage

```
readRNAQuantRawCounts(link)
```

Arguments

`link`	A link to the raw counts file

Details

This function reads the raw counts and returns a matrix with the gene IDs as the row names, ready for use in creating a SummarizedExperiment.

Value

A matrix of raw counts with gene IDs as row names

searchFieldsGEO *Provide a list of possible search fields for GEO search*

Description

Provide a list of possible search fields for GEO search

Usage

```
searchFieldsGEO()
```

Value

a data.frame with names of possible search fields for GEO search as well as descriptions, data types, etc. for each field. Fields are in rows and their properties are in columns.

See Also

searchGEO

Examples

```
searchFieldsGEO()
```

searchGEO *Search GEO database*

Description

This function searches the GDS database, and return a data.frame for all the search results.

Usage

```
searchGEO(query, step = 500L)
```

Arguments

query	character, the search term. The NCBI uses a search term syntax which can be associated with a specific search field with square brackets. So, for instance "Homo sapiens[ORGN]" denotes a search for Homo sapiens in the "Organism" field. Details see https://www.ncbi.nlm.nih.gov/geo/info/qqtutorial.html. The names and definitions of these fields can be identified using searchFieldsGEO.
step	the number of records to fetch from the database each time. You may choose a smaller value if failed.

Details

The NCBI allows users to access more records (10 per second) if they register for and use an API key. set_entrez_key function allows users to set this key for all calls to rentrez functions during a particular R session. You can also set an environment variable ENTREZ_KEY by Sys.setenv. Once this value is set to your key rentrez will use it for all requests to the NCBI. Details see https://docs.ropensci.org/rentrez/articles/rentrez_tutorial.html#rate-limiting-and-api-keys

Value

a data.frame with one row per matching GEO record and columns Accession, Title, Summary, Organism, Type, GPL, n_samples, PDAT, suppFile, FTPLink, SeriesTitle, entryType, and ID (the Entrez UID). An empty query returns a zero-row data.frame with those columns.

See Also

searchFieldsGEO

Examples

```
## Not run:
searchGEO("diabetes[ALL] AND Homo sapiens[ORGN] AND GSE[ETYP]")

## End(Not run)
```

---

urlExtractRNASeqQuantGenomeInfo

Extract genome build and species from a GEO download URL

Description

This function extracts the genome build and species information from a GEO download URL. The genome build and species information is expected to be in the filename of the download URL.

Usage

```
urlExtractRNASeqQuantGenomeInfo(url)
```

Arguments

url	A GEO annotation file download URL

Value

A character vector with the genome build and species information

`urlForAccession` *The URL for a GEO accession*

Description

Sometimes, you just need the URL for a GEO accession. This function returns the URL for a given GEO accession number that can be used to access the GEO page for that accession.

Usage

```
urlForAccession(geo)
```

Arguments

`geo`	A GEO accession number

Value

A character vector with the URL for the GEO accession

See Also

`browseGEOAccession`

Examples

```
urlForAccession("GSE262484")
```

Index

∗ **IO**
coercion, 5
GEOData-accessors, 7
getGEO, 12
getGEOfile, 15
getGEOSuppFiles, 19
getGSEDataTables, 21
parseGEO, 27
∗ **classes**
GDS-class, 6
GEOData-class, 8
GEODataTable-class, 9
GPL-class, 25
GSE-class, 26
GSM-class, 26
∗ **database**
getGEOSuppFiles, 19
∗ **internal**
extractFilenameFromDownloadURL, 6
getGSEDownloadPageURLs, 22
getRNAQuantAnnotationURL, 22
getRNAQuantRawCountsURL, 23
getRNASeqQuantResults, 25
parseGSEMatrix, 28
readRNAQuantAnnotation, 30
readRNAQuantRawCounts, 30
urlExtractRNASeqQuantGenomeInfo, 32

Accession *(GEOData-accessors)*, 7
Accession,GEOData-method *(GEOData-class)*, 8
Accession,GEODataTable-method *(GEODataTable-class)*, 9
as_SummarizedExperiment, 3

browseGEOAccession, 3, *33*
browseWebsiteRNASeqSearch, 4

clearGEOCache, 4, *7*
coercion, 5
Columns *(GEOData-accessors)*, 7
Columns,GEOData-method *(GEOData-class)*, 8
Columns,GEODataTable-method *(GEODataTable-class)*, 9

dataTable *(GEOData-accessors)*, 7
dataTable,GEOData-method *(GEOData-class)*, 8
dataTable,GEODataTable-method *(GEODataTable-class)*, 9

extractFilenameFromDownloadURL, 6

GDS-class, 6
GDS2eSet *(coercion)*, 5
GDS2MA *(coercion)*, 5
geoCache, *4*, 7
GEOData-accessors, 7
GEOData-class, 8
GEODataTable-class, 9
geoSingleCellManifest, 9, *10*, *17, 18*, *29*
geoSingleCellUnits, 10, *17*
geoToSRA, 11
getDirListing, 12
getGEO, *6–9*, 12, *15, 16*, *21*, *25, 26*, *28*
getGEOfile, *14*, 15
getGEOSeriesFileListing, 16
getGEOSingleCell, *10*, 17, *29*
getGEOSuppFiles, *10*, 19
getGEOSuppFileURL, 20
getGSEDataTables, 21
getGSEDownloadPageURLs, 22
getRNAQuantAnnotationURL, 22
getRNAQuantRawCountsURL, 23
getRNASeqData, 23
getRNASeqQuantGenomeInfo, 24
getRNASeqQuantResults, 25
GPL *(GPL-class)*, 25
GPL,GDS-method *(GPL-class)*, 25
GPL-class, 25
GPLList *(GEOData-accessors)*, 7
GPLList,GSE-method *(GSE-class)*, 26
GSE-class, 26
GSM-class, 26
GSMList *(GEOData-accessors)*, 7
GSMList,GSE-method *(GSE-class)*, 26

hasRNASeqQuantifications, 27

Meta *(*GEOData-accessors*)*, 7
Meta,GEOData-method *(*GEOData-class*)*, 8
Meta,GEODataTable-method
*(*GEODataTable-class*)*, 9
Meta,GSE-method *(*GSE-class*)*, 26

parseGDS *(*parseGEO*)*, 27
parseGEO, 27
parseGPL *(*parseGEO*)*, 27
parseGSE *(*parseGEO*)*, 27
parseGSEMatrix, 28
parseGSM *(*parseGEO*)*, 27

readGEOSingleCell, *17*, *18*, 29
readRNAQuantAnnotation, 30
readRNAQuantRawCounts, 30

searchFieldsGEO, *31*, 31, *32*
searchGEO, *11*, *31*, 31
set_entrez_key, *32*
show,GEOData-method *(*GEOData-class*)*, 8
show,GEODataTable-method
*(*GEODataTable-class*)*, 9
Sys.setenv, *32*

Table *(*GEOData-accessors*)*, 7
Table,GEOData-method *(*GEOData-class*)*, 8
Table,GEODataTable-method
*(*GEODataTable-class*)*, 9

urlExtractRNASeqQuantGenomeInfo, 32
urlForAccession, *3*, 33