Package: geokit (via r-universe)

October 27, 2026

Title Tools for Accessing and Working with the Gene Expression Omnibus (GEO)

Version 0.0.2

Description Provides a tidy and fast R interface to the NCBI Gene Expression Omnibus (GEO) database. Functions are included to query, download, and parse GEO metadata and expression data, making it easier to integrate GEO datasets into downstream analyses and workflows.

License MIT + file LICENSE

URL https://github.com/WangLabCSU/geokit, https://WangLabCSU.github.io/geokit/

BugReports https://github.com/WangLabCSU/geokit/issues

Depends R (>= 3.5)

Imports methods, rlang (>= 0.4.11), cli (>= 3.6.0), data.table (>= 1.14.9), curl (>= 6.0.0), rentrez, stats, xml2, utils

Suggests BiocGenerics, Biobase, dplyr, ggplot2, stringr, tidyr, R.utils, duckdb, querychat (>= 0.3.0), knitr, rmarkdown, testthat (>= 3.0.0)

ByteCompile true

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

SystemRequirements Cargo (Rust's package manager), rustc >= 1.87.0

VignetteBuilder knitr

Config/roxygen2/version 8.1.1

Config/pak/sysreqs libxml2-dev libssl-dev libclang-dev

Repository https://wanglabcsu.r-universe.dev

Date/Publication 2026-10-27 13:00:06 UTC

RemoteUrl https://github.com/WangLabCSU/geokit

RemoteRef HEAD

RemoteSha 353fbf0f7953b80e2782aacecbfd67c5a8d76796

Contents

geo_chat 2
geo_gtype 4
geo_matrix 5
geo_meta 6
geo_search 7
geo_show 8
geo_soft 9
geo_suppl 11
geo_url 12
log_trans 15
parse_sample_data 16

Index **17**

`geo_chat` *Chat with GEO metadata using natural language*

Description

Create a `QueryChat` object for exploring GEO metadata with an LLM. Use `geo_chat()` to create the chat object, `geo_shiny()` to launch the Shiny app, and `geo_console()` to start a console chat.

Usage

```
geo_chat(client, data_source, table_name = NULL, ...)

geo_shiny(...)

geo_console(...)
```

Arguments

`client`	Optional chat client. Can be: • An ellmer::Chat object • A string to pass to `ellmer::chat()` (e.g., `"openai/gpt-4o"`) • `NULL` (default): Uses the `querychat.client` option, the `QUERYCHAT_CLIENT` environment variable, or defaults to `ellmer::chat_openai()`
`data_source`	A data.frame or a database connection containing GEO metadata, typically from `geo_meta()` or `geo_search()`.
`table_name`	A string specifying the table name to use in SQL queries. If `data_source` is a data.frame, this is the name to refer to it by in queries (typically the variable name). If not provided, will be inferred from the variable name for data.frame inputs. For database connections, this parameter is required.
`...`	Arguments passed on to `querychat::querychat`

`id` Optional module ID for the QueryChat instance. If not provided, will be auto-generated from `table_name`. The ID is used to namespace the Shiny module.

`greeting` Optional initial message to display to users. Can be a character string (in Markdown format) or a file path. If not provided, a greeting will be generated at the start of each conversation using the LLM, which adds latency and cost. Use `$generate_greeting()` to create a greeting to save and reuse.

`tools` Which querychat tools to include in the chat client, by default. `"filter"` includes the tools for filtering and resetting the dashboard and `"query"` includes the tool for executing SQL queries. Use `tools = "filter"` when you only want the dashboard filtering tools, or when you want to disable the querying tool entirely to prevent the LLM from seeing any of the data in your dataset. The legacy name `"update"` is still accepted as an alias for `"filter"`.

`data_description` Optional description of the data in plain text or Markdown. Can be a string or a file path. This provides context to the LLM about what the data represents.

`categorical_threshold` For text columns, the maximum number of unique values to consider as a categorical variable. Default is 20.

`extra_instructions` Optional additional instructions for the chat model in plain text or Markdown. Can be a string or a file path.

`cleanup` Whether or not to automatically run `$cleanup()` when the Shiny session/app stops. By default, cleanup only occurs if `QueryChat` is created within a Shiny app. Set to `TRUE` to always clean up, or `FALSE` to never clean up automatically.
In `querychat_app()`, in-memory databases created for data frames are always cleaned up.

`bookmark_store` The bookmarking storage method. Passed to `shiny::enableBookmarking()`. If `"url"` or `"server"`, the chat state (including current query) will be bookmarked. Default is `"url"`.

Details

`geo_chat()` intentionally does not serialize all rows or build a large data prompt. Instead, it delegates schema summarization, SQL querying, and dashboard filtering to `QueryChat`.

The three exported helpers differ only in how far they take the `QueryChat` workflow:

- `geo_chat()` creates and returns the `QueryChat` object. Use it when you want to inspect the generated prompt, customize the object, embed it in another Shiny app, or launch the app/console later with `qc$app()` or `qc$console()`.
- `geo_shiny()` creates the same `QueryChat` object and immediately launches its Shiny app. Use it for interactive browser-based filtering and exploration.
- `geo_console()` creates the same `QueryChat` object and immediately starts its console chat. Use it for command-line exploration without opening a Shiny app.

The default instructions guide the assistant to query, and filter GEO metadata, identify relevant studies.

Value

A QueryChat object configured with data_source, an LLM client, and GEO-specific instructions.

See Also

geo_meta(), geo_search(), QueryChat

Examples

```
if (requireNamespace("querychat", quietly = TRUE) &&
    requireNamespace("duckdb", quietly = TRUE)) {
    records <- data.frame(
        Accession = c("GSE1", "GSE2"),
        Title = c("human diabetes study", "mouse liver study"),
        Type = c("Expression profiling by array", "RNA-seq"),
        Samples = c(12L, 8L)
    )
    chat <- geo_chat(NULL, records, cleanup = TRUE)
    chat$cleanup()
}
```

---

geo_gtype *GEO accession type*

Description

Determine the type of a GEO accession ID (e.g. DataSet, Series, Sample, Platform). This function inspects the accession prefix and returns its corresponding GEO type, optionally in an abbreviated form.

Usage

```
geo_gtype(accession, abbre = FALSE)
```

Arguments

accession	A character of GEO accession IDs. Examples: • DataSets (GDS): "GDS505", "GDS606", "GDS1234", "GDS9999", etc. • Series (GSE): "GSE2", "GSE22", "GSE100", "GSE2000", etc. • Platforms (GPL): "GPL96", "GPL570", "GPL10558", etc. • Samples (GSM): "GSM12345", "GSM67890", "GSM112233", etc.
abbre	A logical scalar indicating whether to abbreviate the GEO type in the return value. If FALSE (default), the full type name is returned; if TRUE, a short abbreviation is used.

Value

A character of GEO accession type.

Examples

```
geo_gtype("GSE10")
geo_gtype("gpl98")
geo_gtype("GSM1")
geo_gtype("GDS10")

# use abbreviation
geo_gtype("GSE10", TRUE)
geo_gtype("gpl98", TRUE)
geo_gtype("GSM1", TRUE)
geo_gtype("GDS10", TRUE)
```

`geo_matrix` *Retrieve Series Matrix and Create ExpressionSet*

Description

The function downloads and parses the relevant Series Matrix files, optionally mapping platform IDs to Bioconductor annotation packages.

Usage

```
geo_matrix(
  accession,
  add_gpl = NULL,
  pdata_from_soft = FALSE,
  ftp_over_https = NULL,
  handle_opts = list(),
  odir = getwd()
)
```

Arguments

`accession` A character of GEO accession IDs. Examples:

- DataSets (GDS): `"GDS505"`, `"GDS606"`, `"GDS1234"`, `"GDS9999"`, etc.
- Series (GSE): `"GSE2"`, `"GSE22"`, `"GSE100"`, `"GSE2000"`, etc.
- Platforms (GPL): `"GPL96"`, `"GPL570"`, `"GPL10558"`, etc.
- Samples (GSM): `"GSM12345"`, `"GSM67890"`, `"GSM112233"`, etc.

`add_gpl` Logical or `NULL`. Whether to include platform information (the `featureData` slot). If `NULL` (default), the function attempts to map the GPL accession to a Bioconductor annotation package. If successful, the `annotation` slot is updated and `add_gpl` is set to `FALSE`; otherwise, `add_gpl` is set to `TRUE`.

`pdata_from_soft` Logical. Specifies whether to derive `phenoData` from the GSE series SOFT file. Defaults to `FALSE`, in which case `phenoData` is parsed directly from the series matrix file. Set to `TRUE` if you encounter issues parsing `characteristics_ch*` columns correctly, as it will attempt to retrieve the data from the SOFT file instead.

ftp_over_https	Logical scalar. If TRUE, connects to GEO FTP server via HTTPS (https://ftp.ncbi.nlm.nih.gov/geo); otherwise uses plain FTP (ftp://ftp.ncbi.nlm.nih.gov/geo). Only applicable to GEO FTP server access.
handle_opts	A list of named options / headers to be set in the multi_download.
odir	Destination directory for downloads. Defaults to the current working directory.

Value

An ExpressionSet or a list of ExpressionSets, one per Series Matrix file.

Examples

```
if (require("Biobase")) {
    eset <- geo_matrix("GSE10", odir = tempdir())
}
```

---

geo_meta *Get the metadata of multiple GEO identities*

Description

This function is useful for combining with geo_search() to filter results, as geo_search() cannot retrieve the full metadata for all GEO identifiers. By default, this function uses the soft format for GDS and GSE entities, and the full amount of text format data for GPL and GSM entities.

Usage

```
geo_meta(
  accession,
  famount = NULL,
  scope = NULL,
  ftp_over_https = NULL,
  handle_opts = list(),
  odir = getwd()
)
```

Arguments

accession	A character of GEO accession IDs. Examples: • DataSets (GDS): "GDS505", "GDS606", "GDS1234", "GDS9999", etc. • Series (GSE): "GSE2", "GSE22", "GSE100", "GSE2000", etc. • Platforms (GPL): "GPL96", "GPL570", "GPL10558", etc. • Samples (GSM): "GSM12345", "GSM67890", "GSM112233", etc.
famount	A character specifying either: • the file format on the GEO FTP server, or

- the amount of data in the GEO Accession Display Bar.

See geo_url() for details on the format and amount arguments.

scope — A character specifying which GEO accessions to include (Only applicable to Accession Display Bar access).

- "none": Applicable only to DataSets; for DataSets, this is also the sole valid option
- "self": the queried accession only.
- "gsm", "gpl", "gse": related samples, platforms, or series.
- "all": all accessions related to the query (family view).

ftp_over_https — Logical scalar. If TRUE, connects to GEO FTP server via HTTPS (https://ftp.ncbi.nlm.nih.gov/geo); otherwise uses plain FTP (ftp://ftp.ncbi.nlm.nih.gov/geo). Only applicable to GEO FTP server access.

handle_opts — A list of named options / headers to be set in the multi_download.

odir — Destination directory for downloads. Defaults to the current working directory.

Value

A data frame contains metadata of all ids.

Examples

```
geo_meta("GSE10", odir = tempdir())
```

---

geo_search *Search GEO database*

Description

Search the GDS database and return search results as a data frame.

Usage

```
geo_search(query, step = 500L, interval = NULL)
```

Arguments

query — A character string with the search term. The NCBI uses a fielded search syntax. For example, "Homo sapiens[ORGN]" searches the "Organism" field for *Homo sapiens*. See the GEO query tutorial for details. Searchable fields can be listed with rentrez::entrez_db_searchable("gds").

step — Integer. Number of records to fetch per request. Use a smaller value if requests fail.

interval — Numeric. Time interval (in seconds) between successive requests. Defaults to 0. Increase this value if requests fail due to rate limits.

Details

The NCBI allows higher request limits (10 per second) when using an API key. You can set this key for the current R session with `rentrez::set_entrez_key()`, or permanently by setting the `ENTREZ_KEY` environment variable via Sys.setenv(). Once set, `rentrez` will automatically use this key for all NCBI requests. See the rentrez tutorial for details.

Value

A data frame contains the search results

Examples

```
# Ensure you have an active internet connection before running the search.
# The `geo_search` function queries NCBI Entrez, which may have network
# restrictions and limited bandwidth usage for large queries.

out <- geo_search("diabetes[ALL] AND Homo sapiens[ORGN] AND GSE[ETYP]")
head(out)
```

`geo_show` *Open the GEO landing page in a browser*

Description

Construct a GEO landing page and open it directly in the system's default web browser (or a user-specified browser). By default, this function uses the `brief` amount of `html` format data for all entities.

Usage

```
geo_show(
  accession,
  famount = NULL,
  scope = NULL,
  ftp_over_https = NULL,
  browser = getOption("browser")
)
```

Arguments

`accession`	A character of GEO accession IDs. Examples: • DataSets (GDS): `"GDS505"`, `"GDS606"`, `"GDS1234"`, `"GDS9999"`, etc. • Series (GSE): `"GSE2"`, `"GSE22"`, `"GSE100"`, `"GSE2000"`, etc. • Platforms (GPL): `"GPL96"`, `"GPL570"`, `"GPL10558"`, etc. • Samples (GSM): `"GSM12345"`, `"GSM67890"`, `"GSM112233"`, etc.
`famount`	A character specifying either:

- the file `format` on the GEO FTP server, or
- the `amount` of data in the GEO Accession Display Bar.

See `geo_url()` for details on the `format` and `amount` arguments.

`scope` A character specifying which GEO accessions to include (Only applicable to Accession Display Bar access).

- `"none"`: Applicable only to DataSets; for DataSets, this is also the sole valid option
- `"self"`: the queried accession only.
- `"gsm"`, `"gpl"`, `"gse"`: related samples, platforms, or series.
- `"all"`: all accessions related to the query (family view).

`ftp_over_https` Logical scalar. If `TRUE`, connects to GEO FTP server via HTTPS (https://ftp.ncbi.nlm.nih.gov/geo); otherwise uses plain FTP (ftp://ftp.ncbi.nlm.nih.gov/geo). Only applicable to GEO FTP server access.

`browser` a non-empty character string giving the name of the program to be used as the HTML browser. It should be in the PATH, or a full path specified. Alternatively, an R function to be called to invoke the browser.

Under Windows `NULL` is also allowed (and is the default), and implies that the file association mechanism will be used.

Details

See `browseURL()`

References

- https://www.ncbi.nlm.nih.gov/geo/info/download.html
- https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi
- https://www.ncbi.nlm.nih.gov/geo/info/soft.html#format
- Programmatic access to GEO FTP site

Examples

```
if (interactive()) {
    geo_show("gpl96")
}
```

`geo_soft` *Retrieve GEO SOFT file from NCBI GEO*

Description

By default, this function uses the `soft` format for GDS and GSE entities, and the `full` amount of `text` format data for GPL and GSM entities.

Usage

```
geo_soft(
  accession,
  famount = NULL,
  scope = NULL,
  ftp_over_https = NULL,
  handle_opts = list(),
  odir = getwd()
)
```

Arguments

accession	A character of GEO accession IDs. Examples: • DataSets (GDS): "GDS505", "GDS606", "GDS1234", "GDS9999", etc. • Series (GSE): "GSE2", "GSE22", "GSE100", "GSE2000", etc. • Platforms (GPL): "GPL96", "GPL570", "GPL10558", etc. • Samples (GSM): "GSM12345", "GSM67890", "GSM112233", etc.
famount	A character specifying either: • the file format on the GEO FTP server, or • the amount of data in the GEO Accession Display Bar. See geo_url() for details on the format and amount arguments.
scope	A character specifying which GEO accessions to include (Only applicable to Accession Display Bar access). • "none": Applicable only to DataSets; for DataSets, this is also the sole valid option • "self": the queried accession only. • "gsm", "gpl", "gse": related samples, platforms, or series. • "all": all accessions related to the query (family view).
ftp_over_https	Logical scalar. If TRUE, connects to GEO FTP server via HTTPS (https://ftp.ncbi.nlm.nih.gov/geo); otherwise uses plain FTP (ftp://ftp.ncbi.nlm.nih.gov/geo). Only applicable to GEO FTP server access.
handle_opts	A list of named options / headers to be set in the multi_download.
odir	Destination directory for downloads. Defaults to the current working directory.

Details

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/Series 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

A GEOSoft object (or a list of them when multiple accessions are provided; subclasses `GEOPlatform` and `GEOSeries` are returned for GPL and GSE accessions respectively).

Examples

```
gse <- geo_soft("GSE10", odir = tempdir())
gpl <- geo_soft("gpl98", odir = tempdir())
gsm <- geo_soft("GSM1", odir = tempdir())
gds <- geo_soft("GDS10", odir = tempdir())
```

---

geo_suppl *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.

Usage

```
geo_suppl(
  accession,
  pattern = NULL,
  ftp_over_https = NULL,
  handle_opts = list(),
  odir = getwd()
)
```

Arguments

accession	A character of GEO accession IDs. Examples: • DataSets (GDS): "GDS505", "GDS606", "GDS1234", "GDS9999", etc. • Series (GSE): "GSE2", "GSE22", "GSE100", "GSE2000", etc. • Platforms (GPL): "GPL96", "GPL570", "GPL10558", etc. • Samples (GSM): "GSM12345", "GSM67890", "GSM112233", etc.
pattern	character string containing a regular expression to be matched in the supplementary file names.
ftp_over_https	Logical scalar. If TRUE, connects to GEO FTP server via HTTPS (https://ftp.ncbi.nlm.nih.gov/geo); otherwise uses plain FTP (ftp://ftp.ncbi.nlm.nih.gov/geo). Only applicable to GEO FTP server access.
handle_opts	A list of named options / headers to be set in the multi_download.
odir	Destination directory for downloads. Defaults to the current working directory.

Value

A list (or a character atomic vector if only one accession is provided) of the full file paths of the resulting downloaded files.

Examples

```
geo_suppl("GSM1137", odir = tempdir())
```

---

geo_url *GEO URL resolver*

Description

Construct and resolve URLs for GEO (Gene Expression Omnibus) resources. This function provides a unified interface for accessing GEO data either via Accession Display Bar of GEO database or directly from GEO FTP/HTTPS servers. Depending on the accession type or requested format and amount, it automatically generates the correct URL.

Usage

```
geo_url(
  accession,
  format = NULL,
  amount = NULL,
  scope = NULL,
  ftp_over_https = NULL
)
```

Arguments

`accession` A character of GEO accession IDs. Examples:

- DataSets (GDS): `"GDS505"`, `"GDS606"`, `"GDS1234"`, `"GDS9999"`, etc.
- Series (GSE): `"GSE2"`, `"GSE22"`, `"GSE100"`, `"GSE2000"`, etc.
- Platforms (GPL): `"GPL96"`, `"GPL570"`, `"GPL10558"`, etc.
- Samples (GSM): `"GSM12345"`, `"GSM67890"`, `"GSM112233"`, etc.

`format` A character specifying file format type requested. GEO data can be accessed through two sites:

- Direct FTP/HTTPS file retrieval from GEO FTP server (file type):
  - `"soft"`: SOFT (Simple Omnibus in Text Format) from GEO FTP site. When `accession` is `DataSets` or `Series`, this is the default.
  - `"soft_full"`: full SOFT (Simple Omnibus in Text Format) files from GEO FTP site by DataSet (GDS) containging additionally contains up-to-date gene annotation for the DataSet Platform.
  - `"miniml"`: MINiML (MIAME Notation in Markup Language, pronounced miniml) is an XML format that incorporates experimental data and metadata. MINiML is essentially an XML rendering of SOFT format.
  - `"matrix"`: Series matrix file.
  - `"annot"`: annotation files for Platforms.
  - `"suppl"`: supplementary files.
- For file retrieval from Accession Display Bar of GEO database:
  - `"text"`: machine-readable SOFT format (Simple Omnibus Format in Text).
  - `"xml"`: XML format.
  - `"html"`: human-readable format with hyperlinks (no downloadable entry available).

The following table summarizes the compatibility between GEO accession types and file format options:

format	GDS	GSE	GPL	GSM
SOFT (soft)	o	o	o	x
SOFTFULL (soft_full)	o	x	x	x
MINiML (miniml)	x	o	o	x
Matrix (matrix)	x	o	x	x

Annotation (annot)	x	x	o	x
Supplementaryfiles (suppl)	x	o	o	o
Html (html)	o	o	o	o
Text (text)	x	o	o	o
Xml (xml)	x	o	o	o

`amount` A character specifying the amount of data (Only applicable to Accession Display Bar access):

- `"none"`: Applicable only to DataSets; for DataSets, this is also the sole valid option.
- `"brief"`: accession attributes only.
- `"quick"`: accession attributes + first **20** rows of the data table.
- `"data"`: omits the accession's attributes, showing only links to other accessions and the full data table.
- `"full"`: accession attributes + complete data table.

`scope` A character specifying which GEO accessions to include (Only applicable to Accession Display Bar access).

- `"none"`: Applicable only to DataSets; for DataSets, this is also the sole valid option
- `"self"`: the queried accession only.
- `"gsm"`, `"gpl"`, `"gse"`: related samples, platforms, or series.
- `"all"`: all accessions related to the query (family view).

`ftp_over_https` Logical scalar. If `TRUE`, connects to GEO FTP server via HTTPS (https://ftp.ncbi.nlm.nih.gov/geo); otherwise uses plain FTP (ftp://ftp.ncbi.nlm.nih.gov/geo). Only applicable to GEO FTP server access.

Value

A character of GEO URL.

References

- https://www.ncbi.nlm.nih.gov/geo/info/download.html
- https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi
- https://www.ncbi.nlm.nih.gov/geo/info/soft.html#format
- Programmatic access to GEO FTP site

Examples

```
geo_url("GSE10")
geo_url("gpl98")
geo_url("GSM1")
geo_url("GDS10")
```

log_trans *Apply log2 Transformation to Expression Data*

Description

Checks whether the input data is already log-transformed; if not, applies a log2 transformation. This helps ensure comparability of expression values across datasets.

Usage

```
log_trans(data, pseudo = 1, ...)

## S3 method for class 'matrix'
log_trans(data, pseudo = 1, ...)

## S3 method for class 'ExpressionSet'
log_trans(data, pseudo = 1, ...)
```

Arguments

data	A matrix-like data object.
pseudo	A numeric value added before transformation to avoid taking log of zero. For example, log2(exprs + pseudo).
...	Additional arguments passed to methods.

Details

The function heuristically determines whether data has been log-transformed, following the methodology used in GEO2R. If not, it applies log2() with the specified pseudo offset.

Value

A matrix or an ExpressionSet with log2-transformed expression values.

References

NCBI GEO2R: https://www.ncbi.nlm.nih.gov/geo/geo2r/?acc=GSE1122

Examples

```
sample_means <- 2^runif(2, 2, 10)
sample_disp <- 100 / sample_means + 0.5
data <- matrix(
    rnbinom(4, mu = sample_means, size = 1 / sample_disp),
    nrow = 2
)
log_trans(data)
log_trans(log2(data))
```

parse_sample_data *Parse key-value pairs in the metadata of GEO Sample SOFT file*

Description

Lots of GSEs now use "characteristics_ch*" meta header data for key-value pairs of annotation. If that is the case, this simply cleans the **GEOSoft** @metadata slot up and transforms the keys to column names and the values to column values.

Usage

```
parse_sample_data(x, ...)

## S3 method for class 'GEOSeries'
parse_sample_data(x, ...)

## S3 method for class 'data.frame'
parse_sample_data(x, ..., fields = NULL, sep = ":")

## S3 method for class 'list'
parse_sample_data(x, ...)
```

Arguments

x	A GEOSeries object, a list of GEOSoft from the @gsm slot of a GEOSeries object, or a data frame from Series matrix file data table.
...	Additional arguments passed on to methods.
fields	A character vector which fields should be parsed.
sep	A single byte string defined the pairing separator.

Value

A data.frame whose rows are samples and columns are the sample infos

Examples

```
gse201530_soft <- geo_soft("GSE201530", odir = tempdir())
head(parse_sample_data(gse201530_soft))
```

Index

annotation, *5*

browseURL(), *9*

ellmer::Chat, *2*
ellmer::chat(), *2*
ellmer::chat_openai(), *2*
ExpressionSet, *6*, *15*

featureData, *5*

geo_chat, 2
geo_console *(*geo_chat*)*, 2
geo_gtype, 4
geo_matrix, 5
geo_meta, 6
geo_meta(), *2*, *4*
geo_search, 7
geo_search(), *2*, *4*, *6*
geo_shiny *(*geo_chat*)*, 2
geo_show, 8
geo_soft, 9
geo_suppl, 11
geo_url, 12
geo_url(), *7*, *9, 10*
GEOSeries, *16*
GEOSoft, *11*, *16*

log_trans, 15

multi_download, *6, 7*, *10*, *12*

parse_sample_data, 16

QueryChat, *2–4*
querychat::querychat, *2*

regular expression, *12*
rentrez::entrez_db_searchable(gds), *7*
rentrez::set_entrez_key(), *8*

shiny::enableBookmarking(), *3*
Sys.setenv(), *8*