Package ‘GDSArray’

March 22, 2026

Title Representing GDS files as array-like objects

Version 1.30.0

Description GDS files are widely used to represent genotyping or sequence data. The GDSArray package implements the `GDSArray` class to represent nodes in GDS files in a matrix-like representation that allows easy manipulation (e.g., subsetting, mathematical transformation) in _R_. The data remains on disk until needed, so that very large files can be processed.

biocViews Infrastructure, DataRepresentation, Sequencing, GenotypingArray

Depends R (>= 3.5), gdsfmt, methods, BiocGenerics, DelayedArray (>= 0.5.32)

License GPL-3

Encoding UTF-8

URL https://github.com/Bioconductor/GDSArray

BugReports https://github.com/Bioconductor/GDSArray/issues

Imports tools, S4Vectors (>= 0.17.34), SNPRelate, SeqArray

RoxygenNote 7.3.1

VignetteBuilder knitr

Suggests testthat, knitr, markdown, rmarkdown, BiocStyle, BiocManager

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

git_branch RELEASE_3_22

git_last_commit 421dd04

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2026-03-22

Author Qian Liu [aut],
Martin Morgan [aut],
Hervé Pagès [aut],
Xiuwen Zheng [aut, cre]

Maintainer Xiuwen Zheng <zhengx@u.washington.edu>

Contents

acquireGDS 2
extract_array 3
GDSFile-class 4
seed,GDSArray-method 5

Index 7

`acquireGDS` *Acquire the GDS file connection in R in the* `gds.class` *class.*

Description

Acquire a (possibly cached) `gds.class` object given it's path.

Usage

```
acquireGDS(path, type = NULL, ...)

releaseGDS(path, type = NULL, ...)
```

Arguments

`path`	String containing a path to a GDS file.
`type`	String containing the GDS file type. Case insensitive. Can be "seqgds" for a GDS file with sequencing data, or "snpgds" for a GDS file with SNP data. This argument was added for the `VariantExperiment` package for certain functionalities. By default is NULL, which returns a regular `gds.class`.
`...`	arguments to be passed to `openfn.gds()` inside `acquireGDS`.

Details

`acquireConn` will cache the `gds.class` object in the current R session to avoid repeated initialization. This improves efficiency for repeated calls. The cached `gds.class` object for any given `path` can be deleted by calling `releaseGDS` for the same `path`.

Value

For `acquireGDS`, by default returns a regular `gds.class` object, which are identical to that returned by `gdsfmt::openfn.gds(path)`. If type is not NULL, a `SeqVarGDSClass` that is identical to `SeqArray::seqOPen(path)`, or `SNPGDSFileClass` that is identical to `SNPRelate::snpgdsOpen(path)`. Both are inherited from `gds.class` but with additional checking and methods.

For `releaseGDS`, any existing `gds.class` object for the `path` is disconnected and cleared from cache, and `NULL` is invisibly returned. This is equivalent to that returned by `gdsfmt::closefn.gds()` except it take `path` as input. If `path=NULL`, all cached connections are removed.

Author(s)

Qian Liu

Examples

```
fn <- gdsExampleFileName()
gdscon <- acquireGDS(fn)
acquireGDS(fn)  ## just re-uses the cache
acquireGDS(fn, type = "seqgds") ## construct a new GDS connection
releaseGDS(fn)  ## clears the cache
```

`extract_array` *GDSArray constructor and coercion methods.*

Description

`extract_array`: the function to extract data from a `GDS` file, by taking `GDSArraySeed` as input. This function is required by the `DelayedArray` for the seed contract.

`GDSArray`: The function to convert a gds file into the GDSArray data structure.

`GDSArray` example data

Usage

```
## S4 method for signature 'GDSArraySeed'
extract_array(x, index)

GDSArray(gdsfile, varname)

gdsExampleFileName(type = c("seqgds", "snpgds"))
```

Arguments

`x`	the GDSArraySeed object
`index`	An unnamed list of subscripts as positive integer vectors, one vector per dimension in `x`. Empty and missing subscripts (represented by `integer(0)` and `NULL` list elements, respectively) are allowed. The subscripts can contain duplicated indices. They cannot contain NAs or non-positive values.
`gdsfile`	Can be a GDSArraySeed, a character string of gds file name, or an "gds.class" R object.
`varname`	A character string specifying the gds array node to be read into GDSArray.
`type`	the type of gds file, available are "seqgds" for `SeqVarGDSClass` and "snpgds" for `SNPGDSFileClass`.

Value

`GDSArray` class object.

Examples

```
fn <- gdsExampleFileName("snpgds")
allnodes <- gdsnodes(fn)  ## print all available gds nodes in fn.
allnodes
GDSArray(fn, "genotype")
GDSArray(fn, "sample.annot/pop.group")

fn1 <- gdsExampleFileName("seqgds")
allnodes1 <- gdsnodes(fn1)  ## print all available gds nodes in fn1.
allnodes1
## GDSArray(fn1, "genotype/data")
GDSArray(fn1, "variant.id")
GDSArray(fn1, "sample.annotation/family")
GDSArray(fn1, "annotation/format/DP/data")
GDSArray(fn1, "annotation/info/DP")
gdsExampleFileName("snpgds")
gdsExampleFileName("seqgds")
```

`GDSFile-class` *GDSFile constructor and methods.*

Description

`GDSFile`: `GDSFile` is a light-weight class to represent a GDS file. It has the ‘$‘ completion method to complete any possible gds nodes. If the slot of ‘current_path‘ in ‘GDSFile‘ object represent a valid gds node, it will return the ‘GDSArray‘ of that node directly. Otherwise, it will return the ‘GDSFile‘ object with an updated ‘current_path‘.

`GDSFile`: the `GDSFile` class constructor.

`gdsfile`: `filename` slot getter for `GDSFile` object.

`gdsfile<-`: `filename` slot setter for `GDSFile` object.

`gdsnodes`: to get the available gds nodes from a gds file name or a `GDSFile` object.

Usage

```
GDSFile(file, current_path = "")

## S4 method for signature 'GDSFile'
gdsfile(object)

gdsfile(object) <- value

## S4 method for signature 'GDSFile'
x$name

## S4 method for signature 'ANY'
gdsnodes(x, node)
```

Arguments

`file`	the GDS file path.
`current_path`	the current path to the closest gds node.
`object`	`GDSFile` object.
`value`	the new gds file path
`x`	a character string for the GDS file name or a `GDSFile` object.
`name`	the name of gds node
`node`	the node name of a gds file or `GDSFile` object.

Value

`gdsfile`: the file path of corresponding `GDSfile` object.

`$`: a `GDSFile` with updated `@current_path`, or `GDSArray` object if the `current_path` is a valid gds node.

`gdsnodes`: a character vector of all available gds nodes within the related GDS file and the specified node.

Examples

```
fn <- gdsExampleFileName("seqgds")
gf <- GDSFile(fn)
gdsfile(gf)
fn <- gdsExampleFileName("seqgds")
gdsnodes(fn)
gdsnodes(fn, "annotation/info")
fn1 <- gdsExampleFileName("snpgds")
gdsnodes(fn1)
gdsnodes(fn1, "sample.annot")
gf <- GDSFile(fn)
gdsnodes(gf)
gdsnodes(gf, "genotype")
gdsfile(gf)
```

`seed,GDSArray-method` *GDSArraySeed or GDSArray related methods, slot getters and setters.*

Description

`dim`, `dimnames`: dimension and dimnames of object contained in the GDS file.

`seed`: the `GDSArraySeed` getter for `GDSArray` object.

`seed<-`: the `GDSArraySeed` setter for `GDSArray` object.

`gdsfile`: on-disk location of GDS file represented by this object.

Usage

```
## S4 method for signature 'GDSArray'
seed(x)

## S4 replacement method for signature 'GDSArray'
seed(x) <- value

gdsfile(object)

## S4 method for signature 'GDSArraySeed'
gdsfile(object)

## S4 method for signature 'GDSArray'
gdsfile(object)

## S4 method for signature 'DelayedArray'
gdsfile(object)
```

Arguments

`x`	the `GDSArray` and `GDSArraySeed` objects.
`value`	the new `GDSArraySeed` for the `GDSArray` object.
`object`	GDSArray, GDSMatrix, GDSArraySeed, GDSFile or SummarizedExperiment object.

Value

`dim`: the integer vector of dimensions for `GDSArray` or `GDSArraySeed` objects.

`dimnames`: the unnamed list of dimension names for `GDSArray` and `GDSArraySeed` objects.

`seed`: the `GDSArraySeed` of `GDSArray` object.

`gdsfile`: the character string for the gds file path.

Examples

```
fn <- gdsExampleFileName("snpgds")
ga <- GDSArray(fn, "sample.annot/pop.group")
dim(ga)
dimnames(ga)
type(ga)
seed(ga)
dim(seed(ga))
gdsfile(ga)
```

Index

$,GDSFile-method *(GDSFile-class)*, 4

acquireGDS, 2

coerce *(extract_array)*, 3
coerce,ANY,GDSMatrix-method *(extract_array)*, 3
coerce,GDSArray,GDSMatrix-method *(extract_array)*, 3
coerce,GDSMatrix,GDSArray-method *(extract_array)*, 3

extract_array, 3
extract_array,GDSArraySeed-method *(extract_array)*, 3

GDSArray *(extract_array)*, 3
GDSArray-class *(extract_array)*, 3
GDSArray-data *(extract_array)*, 3
GDSArray-method *(extract_array)*, 3
gdsExampleFileName *(extract_array)*, 3
GDSFile *(GDSFile-class)*, 4
gdsfile *(seed,GDSArray-method)*, 5
gdsfile,DelayedArray-method *(seed,GDSArray-method)*, 5
gdsfile,GDSArray-method *(seed,GDSArray-method)*, 5
gdsfile,GDSArraySeed-method *(seed,GDSArray-method)*, 5
GDSFile,gdsfile-method *(GDSFile-class)*, 4
gdsfile,GDSFile-method *(GDSFile-class)*, 4
GDSFile-class, 4
GDSFile-constructor *(GDSFile-class)*, 4
GDSFile-method *(GDSFile-class)*, 4
gdsfile<- *(GDSFile-class)*, 4
gdsfile<-,GDSFile-method *(GDSFile-class)*, 4
GDSMatrix *(extract_array)*, 3
GDSMatrix-class *(extract_array)*, 3
gdsnodes *(GDSFile-class)*, 4
gdsnodes,ANY-method *(GDSFile-class)*, 4
gdsnodes,GDSFile-method *(GDSFile-class)*, 4

matrixClass,GDSArray-method *(extract_array)*, 3

releaseGDS *(acquireGDS)*, 2

seed,GDSArray-method, 5
seed<-,GDSArray-method *(seed,GDSArray-method)*, 5