Package 'BiocHail'

May 1, 2024

Date 2023-03-19

Title basilisk and hail

Version 1.4.0

Description Use hail via basilisk when appropriate, or via reticulate. This package can be used in terra.bio to interact with UK Biobank resources processed by hail.is.

License Artistic-2.0

Encoding UTF-8

Depends R (>= 4.3.0), graphics, stats, utils

Imports reticulate, basilisk, BiocFileCache, methods, dplyr, BiocGenerics

Suggests knitr, testthat, BiocStyle, ggplot2, DT

VignetteBuilder knitr

biocViews Infrastructure

RoxygenNote 7.2.3

URL https://github.com/vjcitn/BiocHail

BugReports https://github.com/vjcitn/BiocHail/issues

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

git_branch RELEASE_3_19

git_last_commit 9d40c00

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-01

Author Vincent Carey [aut, cre] (<https://orcid.org/0000-0003-4046-0063>)

Maintainer Vincent Carey <stvjc@channing.harvard.edu>

Contents

`as.data.frame` *S3 support*

Description

S3 support

Usage

```
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

`x`	entity coercible to data.frame
`row.names`	character or NULL
`optional`	logical
`...`	any args

Value

data.frame

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
as.data.frame(tab$head(3L))
```

---

as.data.frame.default *S3 support*

Description

S3 support

Usage

```
## Default S3 method:
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	entity coercible to data.frame
row.names	character or NULL
optional	logical
...	any args

Value

data.frame

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
as.data.frame(tab$head(3L))
```

as.data.frame.hail.table.Table

convert hail.table.Table to R data frame

Description

convert hail.table.Table to R data frame

Usage

```
## S3 method for class 'hail.table.Table'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)
```

Arguments

x	instance of "hail.table.Table"
row.names	not used
optional	not used
...	not used

Value

data.frame

Note

only use on small table because collect is used

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
as.data.frame(tab$head(3L))
```

bare_hail

bare interface to hail using reticulate

Description

bare interface to hail using reticulate

Usage

```
bare_hail()
```

Value

python reference to hail module

Note

'/home/jupyter/.local/share/r-miniconda/envs/r-reticulate/bin/pip3 install...' is used to ensure that reticulate's python ecosystem is what we want

Examples

```
# assumes terra
if (nchar(Sys.getenv("WORKSPACE_NAMESPACE"))>0) {
  hl = bare_hail()
  hl$init(idempotent=TRUE, spark_conf=list(
      'spark.hadoop.fs.gs.requester.pays.mode'= 'CUSTOM',
      'spark.hadoop.fs.gs.requester.pays.buckets'= 'ukb-diverse-pops-public',
      'spark.hadoop.fs.gs.requester.pays.project.id'= Sys.getenv("GOOGLE_PROJECT")))
 hl$read_matrix_table('gs://ukb-diverse-pops-public/sumstats_release/results_full.mt')$describe()
  ## Not run:
  # this is supposed to get us some LD data but xx.shape fails, issue filed
    hli = reticulate::import("hail.linalg")
    upa = reticulate::import("ukbb_pan_ancestry")
    xx = hli$BlockMatrix$read(upa$get_ld_matrix_path('AFR'))

## End(Not run)
  }
```

---

colnames,hail.table.Table-method

extract field names from hail.table.Table

Description

extract field names from hail.table.Table

Usage

```
## S4 method for signature 'hail.table.Table'
colnames(x, do.NULL = TRUE, prefix = "col")
```

Arguments

x	hail.table.Table instance
do.NULL	ignored
prefix	ignored

Value

character vector

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
colnames(tab)
```

`filter` *s3 support*

Description

s3 support

Usage

```
filter(.data, ..., .by = NULL, .preserve = FALSE)
```

Arguments

`.data`	instance of hail.table.Table
`...`	should include named components 'filter' which is a logical vector with same number of rows as '.data', 'hl', a reference to a hail environment (Module), and 'placeholder' an arbitrary character(1)
`.by`	not used
`.preserve`	not used

Value

filtered hail.table.Table reference

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
pick <- rep(FALSE, 3500)
pick[seq_len(10)] <- TRUE
ft <- filter(tab, filter = pick, hl = hl)
ft$count()
ft$head(2L)$collect()
```

`filter.hail.table.Table` *filter rows of a hail Table*

Description

filter rows of a hail Table

Usage

```
## S3 method for class 'hail.table.Table'
filter(.data, ..., .by = NULL, .preserve = FALSE)
```

Arguments

`.data`	instance of hail.table.Table
`...`	should include named components ‘filter‘ which is a logical vector with same number of rows as ‘.data‘, ‘hl‘, a reference to a hail environment (Module), and ‘placeholder‘ an arbitrary character(1)
`.by`	not used
`.preserve`	not used

Value

filtered hail.table.Table reference

Note

writes one line of table to disk to retrieve field names

FIXME: uses disk because I don’t know how to create a BooleanExpression except by importing.

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
pick <- rep(FALSE, 3500)
pick[seq_len(10)] <- TRUE
ft <- filter(tab, filter = pick, hl = hl)
ft$count()
ft$head(2L)$collect()
```

get_1kg *interface to 1kg import*

Description

interface to 1kg import

Usage

```
get_1kg(
  hl,
  retrieve_import_write = FALSE,
  path_1kg_zip = osn_1kg_path(),
  folder = tempdir(),
  cache = BiocFileCache::BiocFileCache()
)
```

Arguments

hl	hail object
retrieve_import_write	logical(1) if TRUE, use hl.utils.get_1kg to retrieve data, otherwise acquire a previously written zip file, either from a cache, or, if no file found in cache, from web, followed by caching
path_1kg_zip	character(1) path to zip of MatrixTable, defaults to ‘osn_1kg_path()‘.
folder	character(1) destination of 1kg.mt as retrieved using hl.utils.get_1kg, import_vcf, write
cache	a BiocFileCache-type cache

Value

"hail.matrixtable.MatrixTable" instance

Note

overwrite is permitted in the import_vcf.write event

Examples

```
hl <- hail_init()
mt <- get_1kg(hl)
mt
mt$rows()$select()$show(5L) # must use integer
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
tab$describe()
tab$show(width = 100L)
```

get_key *S3 generic for get_key*

Description

S3 generic for get_key

Usage

```
get_key(x)
```

Arguments

x	anything

Value

typically a list

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
get_key(tab)
```

get_key.hail.table.Table *S3 method for get_key*

Description

S3 method for get_key

Usage

```
## S3 method for class 'hail.table.Table'
get_key(x)
```

Arguments

x	instance of hail.table.Table

Value

a list with elements names (names of keys) and key_df (data.frame of key values, with column names)

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
get_key(tab)
```

`get_ukbb_sumstat_10kloci_mt` *interface to a small subset of UKBB summary stats in MatrixTable format*

Description

interface to a small subset of UKBB summary stats in MatrixTable format

Usage

```
get_ukbb_sumstat_10kloci_mt(
  hl,
  folder = tempdir(),
  cache = BiocFileCache::BiocFileCache(),
  timeout.ukbb = 3600
)
```

Arguments

`hl`	hail object
`folder`	character(1) destination of 1kg.mt as retrieved using hl.utils.get_1kg, import_vcf, write
`cache`	a BiocFileCache-type cache
`timeout.ukbb`	numeric(1) defaults to 3600 for timeout setting in 'options()'; option value is reset on exit

Value

"hail.matrixtable.MatrixTable" instance

Note

The loci were selected using a .000345 of over 28 million loci recorded in the UKBB pan-ancestry record. The sample is made available a) to assess some issues with data volume (the full resource is about 12.78 TB according to [this doc](https://pan-dev.ukbb.broadinstitute.org/docs/hail-format/index.html), and b) to provide full information on the scope of phenotypes and populations available.

This function will unzip 5GB of MatrixTable data. It may be desirable to cache the unzipped image to a persistent location. If this has been done and the environment variable 'HAIL_UKBB_SUMSTAT_10K_PATH' has been set to this location, this function will use the MatrixTable content found there.

Examples

```
hl <- hail_init()
ss <- get_ukbb_sumstat_10kloci_mt(hl)
# consider saving the unzipped image and recaching
ss$count()
```

`hail_init` *initialize hail, using more options*

Description

initialize hail, using more options

Usage

```
hail_init(
  quiet = FALSE,
  min_block_size = 0L,
  branching_factor = 50L,
  default_reference = "GRCh37",
  global_seed = 1234L,
  log = tempfile(),
  spark_conf = NULL,
  gcs_requester_pays_configuration = NULL
)
```

Arguments

`quiet`	logical(1) defaults to FALSE
`min_block_size`	integer(1) defaults to 0L
`branching_factor`	integer(1) defaults to 50L
`default_reference`	character(1) defaults to "GRCh37", for compatibility with earlier ‘hail_init‘
`global_seed`	integer(1) defaults to 1234L
`log`	character(1) target folder for logging, defaults to tempfile()
`spark_conf`	list, defaults to NULL
`gcs_requester_pays_configuration`	list, defaults to NULL

Value

python reference to hail module

Note

hail object may be passed around. See hail documentation for details on all args.

Examples

```
proj = Sys.getenv("GOOGLE_PROJECT")
buck = Sys.getenv("GCS_BUCKET")
if (nchar(buck)>0) {
  # conf = list(proj, c(buck)) doesn't seem to generate tuple[str,Sequence[str]]
  hl <- hail_init()   #gcs_requester_pays_configuration=conf)
  hl$default_reference()
}
```

---

hail_init_simple *initialize hail*

Description

initialize hail

Usage

```
hail_init_simple()
```

Value

python reference to hail module

Note

hail object may be passed around

Examples

```
hc <- hail_init_simple()
hc
```

hail_stop *stop hail*

Description

stop hail

Usage

```
hail_stop(hl)
```

Arguments

hl	a hail object produced by hail_init()

Value

result of stop() method for Hail module

Examples

```
hail_stop
```

kg_3202 *data.frame with metadata about 3202 samples genotyped against T2T reference*

Description

data.frame with metadata about 3202 samples genotyped against T2T reference

Usage

```
data("kg_3202")
```

Format

data.frame

Value

data.frame

Note

Source: index files described at ‘https://www.internationalgenome.org/data-portal/data-collection/30x-grch38‘

Examples

```
data(kg_3202)
dim(kg_3202)
```

`multipop_df` *pheno_data component harvesting from columns of summary stats MatrixTable allowing for info on multiple populations in the pheno_data component*

Description

pheno_data component harvesting from columns of summary stats MatrixTable allowing for info on multiple populations in the pheno_data component

Usage

```
multipop_df(
  x,
 top2get = c("trait_type", "phenocode", "description", "modifier", "coding_description",
    "coding"),
  pheno2get = c("n_cases", "n_controls", "heritability", "pop")
)
```

Arguments

x	Struct - a single element of the list returned by mt$cols()$collect()
top2get	character() vector of general fields to retrieve
pheno2get	character() vector of fields to be retrieved for each subpopulation

Value

data.frame

Examples

```
# following are too time-consuming but can be of interest
# if (nchar(Sys.getenv("HAIL_UKBB_SUMSTAT_10K_PATH"))>0) {
#  hl = hail_init()
#  ss = get_ukbb_sumstat_10kloci_mt(hl)
#  sscol = ss$cols()$collect() # may take a bit of time
#  print(length(sscol))
#  multipop_df(sscol[[1]])
# }
#

# if (nchar(Sys.getenv("HAIL_UKBB_SUMSTAT_10K_PATH"))>0) {
# # to get an overview of all phenotype-cohort combinations in a searchable table
# mmm = lapply(sscol, multipop_df )
```

```
# mymy = do.call(rbind, mmm) # over 16k rows
# DT::datatable(mymy)
# }
#

# this runs quickly and is demonstrative
hl <- hail_init()
litzip <- system.file("extdata", "myss2.zip", package = "BiocHail")
td <- tempdir()
unzip(litzip, exdir = td)
ntab <- hl$read_matrix_table(paste0(td, "/myss2.mt"))
ntab$describe()
nt2 <- ntab$col$collect()
multipop_df(nt2[[1]]) # must select one element
```

`osn_1kg_path` *Open Storage Network path to a zip of hail MatrixTable with some 1kg data for the Hail.is GWAS tutorial*

Description

Open Storage Network path to a zip of hail MatrixTable with some 1kg data for the Hail.is GWAS tutorial

Usage

```
osn_1kg_path()
```

Value

character(1) URL to zip

Examples

```
osn_1kg_path()
```

`osn_ukbb_sumst10k_path` *Open Storage Network path to a zip of hail MatrixTable with a small subset of UKBB summary statistics as of 12/25/2022*

Description

Open Storage Network path to a zip of hail MatrixTable with a small subset of UKBB summary statistics as of 12/25/2022

Usage

```
osn_ukbb_sumst10k_path()
```

Value

character(1) path to zip

Examples

```
osn_ukbb_sumst10k_path()
```

`path_1kg_annotations` *generate path to installed annotations file*

Description

generate path to installed annotations file

Usage

```
path_1kg_annotations()
```

Value

character(1) path to annotations

Note

.txt file retrieved from extraction on 'https://storage.googleapis.com/hail-1kg/tutorial_data.tar'

Examples

```
path_1kg_annotations()
```

pcs_191k *HWE-normalized PCA scores for 3202 thousand-genomes samples genotyped with the telomere-to-telomere reference*

Description

HWE-normalized PCA scores for 3202 thousand-genomes samples genotyped with the telomere-to-telomere reference

Usage

```
data("pcs_191k")
```

Format

data.frame

Value

data.frame

Note

The genotypes are from a 5

Examples

```
data(pcs_191k)
dim(pcs_191k)
```

pcs_38k *HWE-normalized PCA scores for 3202 thousand-genomes samples genotyped with the telomere-to-telomere reference*

Description

HWE-normalized PCA scores for 3202 thousand-genomes samples genotyped with the telomere-to-telomere reference

Usage

```
data("pcs_38k")
```

Format

data.frame

Value

data.frame

Note

The genotypes are from a 1

Examples

```
data(pcs_38k)
dim(pcs_38k)
```

`pheno_data_sec_2df` *pheno_data component harvesting from columns of summary stats MatrixTable*

Description

pheno_data component harvesting from columns of summary stats MatrixTable

Usage

```
pheno_data_sec_2df(
  m,
  section = 1,
  toget = c("n_cases", "n_controls", "heritability", "pop"),
  verbose = FALSE
)
```

Arguments

`m`	Struct returned from mt$cols()$collect()
`section`	numeric(1) element of pheno_data list to be transformed to data.frame
`toget`	character() vector of field names to retrieve
`verbose`	logical(1) if TRUE (NOT default) will message that there are multiple 'pheno_data' components returned

Value

1 row data.frame

Note

applies top2df to the pheno_data component of input

rg_update *update the reference genome for a hail instance*

Description

update the reference genome for a hail instance

Usage

```
rg_update(
  hc,
  init = "GRCh38",
  newjson = system.file("json/t2tAnVIL.json", package = "BiocHail")
)
```

Arguments

hc	hail context
init	character(1) valid name for a reference genome, defaults to "GRCh38"
newjson	character(1) path to a json spec of a reference genome [needs doc]

Value

a python list; the function is used for its side effect

Examples

```
hl <- hail_init()
rg_update(hl)
```

---

rownames,hail.table.Table-method
acquire row names of a Hail Table, assuming key has been set

Description

acquire row names of a Hail Table, assuming key has been set

Usage

```
## S4 method for signature 'hail.table.Table'
rownames(x, do.NULL = TRUE, prefix = "row")
```

Arguments

x	instance of hail.table.Table
do.NULL	not used
prefix	not used

Value

character()

character vector

Note

To try example, run ‘example("rownames,hail.table.Table-method")‘

Examples

```
hl <- hail_init()
annopath <- path_1kg_annotations()
tab <- hl$import_table(annopath, impute = TRUE)$key_by("Sample")
rt <- rownames(tab)
length(rt)
head(rt)
```

---

top2df *top-level annotation harvesting from columns of summary statistics MatrixTable*

Description

top-level annotation harvesting from columns of summary statistics MatrixTable

Usage

```
top2df(
  x,
 toget = c("trait_type", "phenocode", "description", "modifier", "coding_description")
)
```

Arguments

x	a Struct returned from mt$cols()$collect() – which can be slow
toget	character() vector of field names to retrieve

Value

1-row data.frame

Note

python None are transformed to NA

`ukbb_init` *initialize ukbb*

Description

initialize ukbb

Usage

```
ukbb_init()
```

Value

python module reference

Note

ukbb module may be passed around

Examples

```
ukbb <- ukbb_init()
names(ukbb)
```

Index