Package 'rGREAT'

October 9, 2015

Type Package

Title Client for GREAT Analysis

Version 1.0.0

Date 2015-1-28

Author Zuguang Gu

Maintainer Zuguang Gu <z.gu@dkfz.de>

Depends R (>= 2.10.0), GenomicRanges, IRanges

Imports methods, rjson, GetoptLong (>= 0.0.9), RCurl

Suggests testthat (>= 0.3), knitr, circlize

VignetteBuilder knitr

biocViews GeneSetEnrichment, GO, Pathways, Software, Sequencing, WholeGenome, GenomeAnnotation

Description This package makes GREAT (Genomic Regions Enrichment of Annotations Tool) analysis automatic by constructing a HTTP POST request according to user's input and automatically retrieving results from GREAT web server.

URL https://github.com/jokergoo/rGREAT

License GPL (>= 2)

Repository Bioconductor

Date/Publication 2014-1-28 00:00:00

NeedsCompilation no

R topics documented:

availableCategories-GreatJob-method 2
availableOntologies-GreatJob-method 2
getEnrichmentTables-GreatJob-method 3
GreatJob-class 4
plotRegionGeneAssociationGraphs-GreatJob-method 5
submitGreatJob 7

Index **10**

availableCategories-GreatJob-method
Available ontology categories

Description

Available ontology categories

Usage

```
## S4 method for signature 'GreatJob'
availableCategories(job)
```

Arguments

job	a GreatJob instance

Details

The values of the supported categories sometime change. You should run the function to get the realtime values. The meaning of categories returned is quite self-explained by the name.

Value

The returned value is a vector of categories.

Author(s)

Zuguang gu <z.gu@dkfz.de>

Examples

```
job = readRDS(paste0(system.file("extdata", package = "rGREAT"), "/job.rds"))
availableCategories(job)
```

availableOntologies-GreatJob-method
All available ontology names

Description

All available ontology names

Usage

```
## S4 method for signature 'GreatJob'
availableOntologies(job, category = NULL)
```

Arguments

job	a GreatJob instance
category	one or multiple categories. All available categories can be get by availableCategories

Details

The values of the supported ontologies sometime change. You should run the function to get the realtime values. The meaning of ontology returned is quite self-explained by the name.

Value

The returned values is a vector of ontologies.

Author(s)

Zuguang gu <z.gu@dkfz.de>

Examples

```
job = readRDS(paste0(system.file("extdata", package = "rGREAT"), "/job.rds"))
availableOntologies(job)
availableOntologies(job, category = "Pathway Data")
```

---

getEnrichmentTables-GreatJob-method

Get enrichment tables from GREAT web server

Description

Get enrichment tables from GREAT web server

Usage

```
## S4 method for signature 'GreatJob'
getEnrichmentTables(job, ontology = NULL, category = "GO",
    request_interval = 30, max_tries = 100)
```

Arguments

job	a GreatJob instance
ontology	ontology names. Valid values are in availableOntologies. ontology is prior to category argument.
category	Pre-defined ontology categories. One category can contain more than one ontologies. Valid values are in availableCategories
request_interval	time interval for two requests. Default is 300 seconds.
max_tries	maximum tries

Details

The table contains statistics for the each term in each ontology catalogue.

Please note there is no FDR column in original tables. Users should calculate by themselves by functions such as `p.adjust`

Value

The returned value is a list of data frames in which each one corresponds to result for a single ontology. The structure of the data frames are same as the tables available on GREAT website.

See

`availableOntologies`, `availableCategories`

Author(s)

Zuguang gu <z.gu@dkfz.de>

Examples

```
job = readRDS(paste0(system.file("extdata", package = "rGREAT"), "/job.rds"))
tb = getEnrichmentTables(job)
names(tb)
head(tb[[1]])
job

tb = getEnrichmentTables(job, ontology = "GO Molecular Function")
tb = getEnrichmentTables(job, category = "GO")
```

GreatJob-class *Class to store and retrieve GREAT results*

Description

Class to store and retrieve GREAT results

Details

After submitting request to GREAT server, the generated results will be available on GREAT server for some time. The `GreatJob` class is defined to store parameters that user has set and result tables what were retrieved from GREAT server.

Constructor

Users don't need to construct by hand, `submitGreatJob` is used to generate a `GreatJob` instance.

Workflow

After submitting request to GREAT server, users can perform following steps:

- call `getEnrichmentTables` to get enrichment tables for selected ontologies catalogues.
- call `plotRegionGeneAssociationGraphs` to get associations between regions and genes as well as making plots.

Constructor

Users don't need to construct by hand, `submitGreatJob` is used to generate a `GreatJob` instance.

Workflow

After submitting request to GREAT server, users can perform following steps:

- call `getEnrichmentTables` to get enrichment tables for selected ontologies catalogues.
- call `plotRegionGeneAssociationGraphs` to get associations between regions and genes as well as making plots.

Author(s)

Zuguang gu <z.gu@dkfz.de>

Examples

```
# please refer to page of `submitGreatJob`
```

---

plotRegionGeneAssociationGraphs-GreatJob-method

Plot region-gene association figures

Description

Plot region-gene association figures

Usage

```
## S4 method for signature 'GreatJob'
plotRegionGeneAssociationGraphs(job, type = 1:3, ontology = NULL,
    termID = NULL, request_interval = 30, max_tries = 100)
```

Arguments

job	a GreatJob instance
type	type of plots, should be in 1, 2, 3. See details section for explanation
ontology	ontology name
termID	term id which corresponds to the selected ontology
request_interval	time interval for two requests. Default is 300 seconds.
max_tries	maximum tries

Details

Generated figures are:

- association between regions and genes
- distribution of distance to TSS
- distribution of absolute distance to TSS

If ontology and termID are set, only regions and genes corresponding to selected ontology term will be used. Valid value for ontology is in availableOntologies and valid value for termID is from 'id' column in the table which is returned by getEnrichmentTables.

Value

a GRanges object. Columns in metadata are:

gene genes that are associated with corresponding regions

distTSS distance from the regions to TSS of the associated gene

The returned values corresponds to whole input regions or only regions in specified ontology term, depending on user's setting.

If there is no gene associated with the region, corresponding gene and distTSS columns will be NA.

Author(s)

Zuguang gu <z.gu@dkfz.de>

Examples

```
job = readRDS(paste0(system.file("extdata", package = "rGREAT"), "/job.rds"))

op = par("mfrow")
par(mfrow = c(1, 3))
res = plotRegionGeneAssociationGraphs(job)
res

par(mfrow = c(1, 1))
plotRegionGeneAssociationGraphs(job, type = 1)
```

```
par(mfrow = c(1, 3))
res = plotRegionGeneAssociationGraphs(job, ontology = "GO Molecular Function",
    termID = "GO:0004984")
res

par(mfrow = op)
```

---

submitGreatJob *Send requests to GREAT web server*

Description

Send requests to GREAT web server

Usage

```
submitGreatJob(gr, bg = NULL,
    species               = c("hg19", "hg18", "mm9", "danRer7"),
    includeCuratedRegDoms = TRUE,
    bgChoice              = c("wholeGenome", "data"),
    rule                  = c("basalPlusExt", "twoClosest", "oneClosest"),
    adv_upstream          = 5.0,
    adv_downstream        = 1.0,
    adv_span              = 1000.0,
    adv_twoDistance       = 1000.0,
    adv_oneDistance       = 1000.0,
    request_interval = 300,
    max_tries = 10)
```

Arguments

gr	A GRanges object or a data frame which contains at least three columns (chr, start and end). Regions for test.
bg	A GRanges object or a data frame. Background regions if needed.
species	Species. Only four species ("hg19", "hg18", "mm9", "danRer7") are supported.
includeCuratedRegDoms	Whether to include curated regulatory domains.
bgChoice	How to define background. If it is set as data, bg should be set as well.
rule	How to associate genomic regions to genes. See 'details' section.
adv_upstream	Unit: kb, only used when rule is basalPlusExt
adv_downstream	Unit: kb, only used when rule is basalPlusExt
adv_span	Unit: kb, only used when rule is basalPlusExt
adv_twoDistance	Unit: kb, only used when rule is twoClosest

`adv_oneDistance`
: Unit: kb, only used when rule is `oneClosest`

`request_interval`
: Time interval for two requests. Default is 300 seconds.

`max_tries`
: Maximum times trying to connect to GREAT web server.

Details

Note it is not the standard GREAT API. This function directly send data to GREAT web server by HTTP POST.

Following text is copied from GREAT web site (http://bejerano-test.stanford.edu/great/public/html/index.php)

Explanation of `rule` and settings with names started with 'adv_' (advanced settings):

basalPlusExt Mode 'Basal plus extension'. Gene regulatory domain definition: Each gene is assigned a basal regulatory domain of a minimum distance upstream and downstream of the TSS (regardless of other nearby genes, controlled by `adv_upstream` and `adv_downstream` argument). The gene regulatory domain is extended in both directions to the nearest gene's basal domain but no more than the maximum extension in one direction (controlled by `adv_span`).

twoClosest Mode 'Two nearest genes'. Gene regulatory domain definition: Each gene is assigned a regulatory domain that extends in both directions to the nearest gene's TSS (controlled by `adv_twoDistance`) but no more than the maximum extension in one direction.

oneClosest Mode 'Single nearest gene'. Gene regulatory domain definition: Each gene is assigned a regulatory domain that extends in both directions to the midpoint between the gene's TSS and the nearest gene's TSS (controlled by `adv_oneDistance`) but no more than the maximum extension in one direction.

Value

A `GreatJob` class object which can be used to get results from GREAT server.

Author(s)

Zuguang gu <z.gu@dkfz.de>

See Also

`GreatJob-class`

Examples

```
set.seed(123)
bed = circlize::generateRandomBed(nr = 1000, nc = 0)
job = submitGreatJob(bed)

# more parameters can be set for the job
## Not run:
job = submitGreatJob(bed, species = "mm9")
job = submitGreatJob(bed, bg, species = "mm9", bgChoise = "data")
```

```
job = submitGreatJob(bed, adv_upstream = 10, adv_downstream = 2, adv_span = 2000)
job = submitGreatJob(bed, rule = "twoClosest", adv_twoDistance = 2000)
job = submitGreatJob(bed, rule = "oneClosest", adv_oneDistance = 2000)

## End(Not run)
```

Index

availableCategories, *3, 4*
availableCategories
(availableCategories-GreatJob-method), 2
availableCategories,GreatJob-method
(availableCategories-GreatJob-method), 2
availableCategories-GreatJob-method, 2
availableOntologies, *3, 4*, 6
availableOntologies
(availableOntologies-GreatJob-method), 2
availableOntologies,GreatJob-method
(availableOntologies-GreatJob-method), 2
availableOntologies-GreatJob-method, 2

getEnrichmentTables, *5, 6*
getEnrichmentTables
(getEnrichmentTables-GreatJob-method), 3
getEnrichmentTables,GreatJob-method
(getEnrichmentTables-GreatJob-method), 3
getEnrichmentTables-GreatJob-method, 3
GRanges, *6, 7*
GreatJob, *2, 3*, *6*, *8*
GreatJob *(*GreatJob-class*)*, 4
GreatJob-class, 4

p.adjust, *4*
plotRegionGeneAssociationGraphs, *5*
plotRegionGeneAssociationGraphs
(plotRegionGeneAssociationGraphs-GreatJob-method), 5
plotRegionGeneAssociationGraphs,GreatJob-method
(plotRegionGeneAssociationGraphs-GreatJob-method), 5
plotRegionGeneAssociationGraphs-GreatJob-method, 5

submitGreatJob, *4, 5*, 7