Package ‘BiocOncoTK’

April 10, 2024

Title Bioconductor components for general cancer genomics

Description Provide a central interface to various tools for genome-scale analysis of cancer studies.

Version 1.22.2

Author Vince Carey


Imports ComplexHeatmap, S4Vectors, bigrquery, shiny, stats, httr, rjson, dplyr, magrittr, grid, DT, GenomicRanges, IRanges, ggplot2, SummarizedExperiment, DBI, GenomicFeatures, curatedTCGAData, scales, ggpubr, plyr, car, graph, Rgraphviz, MASS, grDevices

Depends R (>= 3.6.0), methods, utils

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

License Artistic-2.0

LazyLoad yes

LazyData yes

LazyDataCompression xz

biocViews CopyNumberVariation, CpGIsland, DNAMethylation, GeneExpression, GeneticVariability, SNP, Transcription, ImmunoOncology

VignetteBuilder knitr

RoxygenNote 7.3.1

Encoding UTF-8

git_url https://git.bioconductor.org/packages/BiocOncoTK
git_branch RELEASE_3_18
R topics documented:

add_sym.................................................................................. 3
annotTabs.................................................................................. 4
bindMSI....................................................................................... 4
bipg_tests................................................................................... 5
buildPancanSE............................................................................ 6
CCLE_DRUG_BROAD.................................................................... 7
cell_70138.................................................................................. 8
clueDemos.................................................................................... 8
clueServiceNames....................................................................... 9
darmGBMcls............................................................................... 9
dingMSI....................................................................................... 10
featIDMapper................................................................................. 10
fireMSI......................................................................................... 11
get_plates.................................................................................... 11
ggFeatDens................................................................................ 12
ggFeatureSegs............................................................................ 13
ggMutDens.................................................................................... 14
icd10_c......................................................................................... 15
k23sig.......................................................................................... 15
kang_DNArepair........................................................................... 16
loadPatel...................................................................................... 17
load_ccleNRAS............................................................................ 17
log10p1l...................................................................................... 18
map_tegra_ncit............................................................................ 19
mc3toGR...................................................................................... 19
molpo_3utr.................................................................................. 20
molpo_5utr.................................................................................. 21
molpo_CDS................................................................................... 21
molpo_WGS................................................................................ 22
MSIsensor.10k.............................................................................. 23
multiviz......................................................................................... 23
oncoPrintISB............................................................................... 24
pancan.clin.varnames................................................................. 25
pancan_app.................................................................................. 25
pancan_BQ................................................................................... 26
pancan_clinicalTabVarnames...................................................... 26
pancan_longname....................................................................... 27
pancan_sampTypeMap............................................................... 27
pancan_tabulate.......................................................................... 28
patient_to_tumor_code.............................................................. 28
add_sym

Description

add symbols in rowData to a SummarizedExperiment that has Entrez IDs for rownames

Usage

add_sym(x)

Arguments

x SummarizedExperiment instance

Note

Will fail if 'symbol' is a column of rowData(x)

Examples

if (interactive()) {
  bq = pancan_BQ()
  rnse = try(buildPancanSE(bq, assay="RNASeqv2"))
  if (inherits(rnse, "try-error")) stop("probably need CGC_BILLING set in environment or with pancan_BQ")
  add_sym(rnse)
}
annotTabs  

**table names in Annotated pancancer data release**

**Description**

table names in Annotated pancancer data release

**Usage**

annotTabs

**Format**

character vector

**Source**

pancancer-atlas in BigQuery

**Examples**

BiocOncoTK::annotTabs

---

bindMSI  

**bind MSI data to a SummarizedExperiment**

**Description**

bind MSI data to a SummarizedExperiment

**Usage**

bindMSI(se, useDing = TRUE, onlyHL = TRUE)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>se</td>
<td>SummarizedExperiment instance</td>
</tr>
<tr>
<td>useDing</td>
<td>logical(1) if TRUE, use MSIsensor outputs from Ding et al. Cell 2018, otherwise use firehose labelings msi-h, msi-l</td>
</tr>
<tr>
<td>onlyHL</td>
<td>logical(1) if TRUE, retain only msi-h, msi-l records; ignored if useDing is TRUE</td>
</tr>
</tbody>
</table>

**Value**

SummarizedExperiment instance with expanded colData, samples limited to those with microsatellite instability values. The additional variable is called 'msiTest' and is numerical if useDing is TRUE and is character (msi-h,l,s) otherwise.
**Note**

This function adds the column `msiTest` to `colData(se)`. The contents of the column are given by `fireMSI`. Samples in `se` that do not correspond to a row of `fireMSI` are dropped. If there is already a column named `msiTest` in `colData(se)`, it is replaced and samples are filtered as described, and a message is given. If none of the samples in `se` have rows in `fireMSI`, an error is thrown. *OF NOTE:* The MSIsensor data from Ding’s cell paper (see help(dingMSI) for URL) provides the participant barcode. The participant barcode is a substring of the sample barcode. Be sure to filter the input SummarizedExperiment to include only tumor samples, using the `substr(colnames(se),14,15)` (values "10"..."14" correspond to normal, non-tumor samples.) Additionally, bindMSI will only work if the colnames of the (filtered) SummarizedExperiment have been truncated to the participant barcode, that is, the first 12 characters of the sample barcode.

**Examples**

```r
bindMSI
```

```r
tab <- data.frame(tumor = c("Lung", "Breast", "Colon"),
                  gene = c("EGFR", "HER2", "KRAS"),
                  tstat = c(3.2, 2.1, 4.5))
bindMSI(tab, NA, NA, 0.65, 0)
```

### Description

configure a bipartite graph relating tumor type to gene, using graphNEL

### Usage

```r
bipg_tests(stattab, genes_adverse = NA, genes_favorable = NA, gpar_cex = 0.65, gpar_lwd = 0)
```

### Arguments

- `stattab`: a data.frame with columns 'tumor', 'gene', and 'tstat'
- `genes_adverse`: a vector of genes whose increased expression is regarded as adverse
- `genes_favorable`: a vector of genes whose increased expression is regarded as favorable
- `gpar_cex`: tune size of graph labels
- `gpar_lwd`: tune appearance of node boundaries

### Value

a graphNEL instance (graph package)
Examples

bipg_tests(k23sig)

Description

helper for SummarizedExperiment construction from pancan

Usage

buildPancanSE(
  bq,
  acronym = "BLCA",
  assay = "meth450k",
  sampType = "TP",
  subjectIDName = "ParticipantBarcode",
  seTransform = force,
  bindMethRowranges = TRUE,
  featIDMap = featIDMapper()
)

Arguments

bq
  instance of BigQueryConnection for pancancer-atlas.Annotated Dataset
acronym
  character(1) 'cohort' label, e.g., 'BLCA'
assay
  character(1) element from names(BiocOncoTK::annotTabs), e.g., 'meth450k'.
  If `assay == "mc3_MAF"` an error is thrown as the mutation data are inconsistently annotated; the message produced directs the user to `mc3toGR`.
sampType
  character(1) element from BiocOncoTK::pancan_sampTypeMap$"SampleTypeLetterCode",
  e.g., 'TP' for Primary solid Tumor samples, or 'TB' for peripheral blood sample from primary blood derived cancer
subjectIDName
  character(1) field name for subject identifier
seTransform
  a function that accepts a SummarizedExperiment and returns a SummarizedExperiment; useful for feature name remapping, defaults to force (does nothing)
bindMethRowranges
  logical(1) if true and assay is meth27k
featIDMap
  a named character() vector defining, for each assay type, what field should be used to label features in rownames. or meth450k, annotation from FDb.InfiniumMethylation.hg19 and EnsDb.Hsapiens.v75 is obtained for available features and bound into the rowRanges component of returned object
Value

SummarizedExperiment, with metadata on acronym, assay, and sampleType propagated; if the assay is a methylation assay and bindMethRowranges is TRUE, a RangedSummarizedExperiment is returned.

Note

Note that pancancer-atlas is distinguished from TCGA by the presence of more sample types. The default type is 'TP' for primary solid tumor. Codes and their interpretations are available in BioconcoTK::pancan_sampTypeMap.

Examples

```r
if (interactive() && Biobase::testBioCConnection()) {
  billco = Sys.getenv("CGC_BILLING")
  if (nchar(billco)>0) {
    bq = pancan_BQ()
    methSE_BLCA = try(buildPancanSE(bq))
    methSE_BLCA
  }
}
```

## CCLE_DRUG_BROAD

CCLE_DRUG_BROAD: serialization of legacy CCLE 'Drug data' from Broad Institute

### Description

CCLE_DRUG_BROAD: serialization of legacy CCLE 'Drug data' from Broad Institute

### Usage

CCLE_DRUG_BROAD

### Format

S4Vectors DataFrame instance

### Source


### Examples

```r
data(CCLE_DRUG_BROAD)
requireNamespace("S4Vectors")
S4Vectors::metadata(CCLE_DRUG_BROAD) # imported using read.csv, stringsAsFactors=FALSE, coerced to DataFrame
head(CCLE_DRUG_BROAD)
```
cell_70138  

**Description**  

*cell_70138*: a table with cell-line information from LINCS

**Usage**

`cell_70138`

**Format**

`data.frame`

**Source**

GEO GSE70138 GSE70138_Broad_LINCS_cell_info_2017-04-28.txt.gz

**Examples**

```r
data(cell_70138)
```

---

cueDemos  

**Description**

generate lists to generate clue API queries

**Usage**

`clueDemos()`

**Value**

a list of lists of strings with 'where' and substructure as appropriate

**Note**

These are converted to JSON

**Examples**

```r
clueDemos()
```
**clueServiceNames**

Provide names of some clue.io services for which examples are available in this package.

**Description**

Provide names of some clue.io services for which examples are available in this package.

**Usage**

```r
clueServiceNames()
```

**Value**

a character vector of service names

**Note**

See [https://clue.io/api](https://clue.io/api).

**Examples**

```r
clueServiceNames()
```

---

**darmGBMcls**

Data in count_lstpm format from Darmanis 2017 (PMC 5810554) single cell RNA-seq in GBM

**Description**

Data in count_lstpm format from Darmanis 2017 (PMC 5810554) single cell RNA-seq in GBM

**Usage**

```r
darmGBMcls
```

**Format**

SummarizedExperiment with HDF Object store back end

**Note**

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5810554/ is the main source article.

**Source**

http://imlspenticton.uzh.ch/robinson_lab/conquer/data-mae/GSE84465.rds
Examples

BiocOncoTK::dirmgbmcls

dingMSI

Description

microsatellite instability data in TCGA, collected from Ding et al. Cell 173(2) 2018.

Usage

dingMSI

Format

DataFrame

Source


Examples

str(BiocOncoTK::dingMSI)

featIDMapper

Description

define assay-specific feature names in a character vector

Usage

featIDMapper()

Note

We may want to use Symbol instead of Entrez when retrieving expression data. The value of this function is supplied as a default for buildPancanSE's featIDMap parameter, and alternatives can be selected by passing similarly named vectors in featIDMap.

Examples

featIDMapper()
**Description**

microsatellite instability data in TCGA, collected from curatedTCGA-Data

**Usage**

fireMSI

**Format**

DataFrame

**Source**

firehose via curatedTCGAData; see metadata(BiocOncoTK::fireMSI)

**Examples**

```r
str(S4Vectors::metadata(BiocOncoTK::fireMSI))
```

---

**get_plates**

*use curatedTCGAData request to acquire plate codes for samples*

**Description**

use curatedTCGAData request to acquire plate codes for samples

**Usage**

```r
get_plates(
  tumcode = "BLCA",
  assay = "RNASeq2GeneNorm",
  samptypes = c("01", "02", "03", "04", "06", "09", "40")
)
```

**Arguments**

- **tumcode**
  - a TCGA tumor code, usually 3 or for characters
- **assay**
  - a curatedTCGAData assay code, run curatedTCGAData() to see a message with available options
- **samptypes**
  - a character vector with codes as defined at [https://gdc.cancer.gov/resources-tcga-users/tcga-code-tables/sample-type-codes](https://gdc.cancer.gov/resources-tcga-users/tcga-code-tables/sample-type-codes)
Value

a data.frame with a row for each TCGA contribution for the selected tumor type and assay type

Examples

```r
if (interactive()) {
    plts_blca_rnaseq = get_plates()
    dim(plts_blca_rnaseq)
    head(plts_blca_rnaseq)
}
```

ggFeatDens

create ggplot for density of starts of a GRanges in an interval

Description

create ggplot for density of starts of a GRanges in an interval

Usage

```r
ggFeatDens(
    gr,
    mcolvbl,
    chrname = "chr15",
    start = 20450000,
    end = 20730000,
    binwidth.in = 5000,
    basicfilt = function(data) dplyr::filter(data, Consequence == "non_coding_transcript_exon_variant"),
    ylab.in = "feature\ndensity",
    slstyle = "UCSC"
)
```

Arguments

- `gr` : GRanges instance of interest
- `mcolvbl` : character(1) mcols(gr) has this variable that will be used to specify different groups for computing/colouring the density traces
- `chrname` : character(1) chromosome/seqname
- `start` : numeric(1) start of interval
- `end` : numeric(1) end of interval
- `binwidth.in` : numeric(1) for geom_freqpoly binwidth setting
- `basicfilt` : a dplyr::filter operation, defaulting to select non-coding variants in mc3 MAF
- `ylab.in` : character(1) label for y axis
- `slstyle` : character(1) for GenomeInfoDb::seqlevelsStyle

```r"
ggFeatureSegs

Value

ggplot instance

Examples

ggFeatDens

------------------------------------------------------
ggFeatureSegs  generate a ggplot of segments of gene-like regions
------------------------------------------------------

Description

generate a ggplot of segments of gene-like regions

Usage

ggFeatureSegs(
  chrname = "chr15",
  start = 20450000,
  end = 20730000,
  db = EnsDb.Hsapiens.v75::EnsDb.Hsapiens.v75,
  slstyle = "UCSC",
  ylab.in = "ensembl\nnoncoding"
)

Arguments

  chrname  character(1) chromosome tag
  start    numeric(1) start of interval
  end      numeric(1) end of interval
  db       EnsDb instance for example
  slstyle  character(1) tag for seqlevelsStyle
  ylab.in  character(1) for use as y axis tag

Value

ggplot instance

Note

Most annotation is turned off with element_blank()

Examples

ggFeatureSegs
ggMutDens

make a ggplot with density traces of mutations per base pair, for 'most mutated' tumor types in a given interval

Description

make a ggplot with density traces of mutations per base pair, for 'most mutated' tumor types in a given interval

Usage

ggMutDens(
  bq,
  basicfilt = function(data) dplyr::filter(data, Consequence ==
    "non_coding_transcript_exon_variant"),
  chrname = "15",
  start = 20450000,
  end = 20730000,
  project_volume = 5,
  maxnrec = 50000,
  binwidth = 5000,
  xlab.in = ""
)

Arguments

bq bigquery BigQueryConnection instance
basicfilt a dplyr::filter operation, defaulting to select non-coding variants in mc3 MAF
chrname character(1) chromosome token in NCBI seqlevels style
start numeric(1) base coordinate to start
end numeric(1) base coordinate to end
project_volume numeric(1) tumor types will have different numbers of contributions; this parameter tells how many tumor types to represent, counting down from the most frequently represented
maxnrec numeric(1) for as.data.frame
binwidth numeric(1) passed to geom_freqpoly
xlab.in character(1) passed to ggplot2::xlab

Value

instance of ggplot
icd10_c

**Examples**

```r
if (interactive()) {
  if (!requireNamespace("ggplot2")) stop("install ggplot2 to run this function")
  bq = try(pancan_BQ())
  if (!inherits(bq, "try-error")) {
    ggMutDens(bq)
  }
}
```

---

### icd10_c

*helper for interpreting ICD-10 codes*

---

**Description**

helper for interpreting ICD-10 codes

**Usage**

```r
icd10_c
```

**Format**

data.frame

**Source**

ICD-10

**Examples**

```r
BiocOncoTK::icd10_c
```

---

### k23sig

*a table of 'significant' MSIsensor-score/expression relationships in TCGA*

---

**Description**

a table of 'significant' MSIsensor-score/expression relationships in TCGA

**Usage**

```r
k23sig
```

**Format**

data.frame
**Note**

provided to demonstrate bipartite graph construction

**Examples**

head(k23sig)

---

| kang_DNArepair | list of 151 genes annotated as DNA repair pathway members |

**Description**

list of 151 genes annotated as DNA repair pathway members

**Usage**

kang_DNArepair

**Format**

named list

**Note**

The zipped PDF was read using pdftools::pdf_text and then manually organized. All gene symbols present in curatedTCGADatRNASeq2GeneNorm rownames. The list elements are ATM, BER, FA.HR, MMR, NER, NHEJ, OTHER, TLS, RECQ, and XLR. These denote, respectively, ataxia-telangiectasia-mutated, base excision repair, Fanconi anemia/homologous recombination, mismatch repair, nucleotide excision repair, non-homologous end joining, other, translesion synthesis, recQ helicase pathway, and cross-link repair.

**Source**

https://academic.oup.com/jnci/article/104/9/670/872781#supplementary-data
**loadPatel**

*use BiocFileCache discipline to acquire patelGBMSC SummarizedExperiment*

---

**Description**

use BiocFileCache discipline to acquire patelGBMSC SummarizedExperiment

**Usage**

```r
loadPatel(
  remotePath = "https://s3.us-east-2.amazonaws.com/biocfound-scrna/patelGBMSC.rds",
  cache = BiocFileCache::BiocFileCache()
)
```

**Arguments**

- `remotePath` character(1) identifying remote RDS
- `cache` instance of BiocFileCache, defaults to BiocFileCache::BiocFileCache()

**Value**

a SummarizedExperiment instance

**Note**

The RDS for the SummarizedExperiment is in an AWS S3 bucket. This function will check local cache for the data and will download to cache if not found. That download is a one-time operation for any given value of cache.

**Examples**

```r
loadPatel
```

---

**load_ccleNRAS**

*utilities for mock data (not involving internet access for vignette)*

---

**Description**

utilities for mock data (not involving internet access for vignette)

**Usage**

```r
load_ccleNRAS()
load_NRAS_AHR()
load_nrasdf()
```
log10pl1

Description

log10(x+p) transformation for use with scales/ggplot2

Usage

log10pl1(p = 1)

Arguments

p value of shift before taking log10

Value

an instance of custom trans() for scales package
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>map_tcga_ncit</code></td>
<td>A manually constructed table mapping TCGA acronyms to NCIT thesaurus tags</td>
</tr>
</tbody>
</table>

### Description

A manually constructed table mapping TCGA acronyms to NCIT thesaurus tags.

### Usage

```r
map_tcga_ncit
```

### Format

data.frame

### Note

Constructed using ontoProc::getOncotreeOnto() result. See the vignette on Mapping TCGA tumor codes to NCIT for elaborating the mapping to aggregate tumors into NCIT organ systems.

---

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mc3toGR</code></td>
<td>Create a GRanges from the MC3 mutation data</td>
</tr>
</tbody>
</table>

### Description

Create a GRanges from the MC3 mutation data.

### Usage

```r
mc3toGR(
  bq,
  basicfilt = function(data) dplyr::filter(data, Consequence ==
    "non_coding_transcript_exon_variant"),
  maxnrec = 1e+05
)
```

### Arguments

- `bq` : bigquery BigQueryConnection instance
- `basicfilt` : a dplyr::filter instance or NULL to convert entire MAF
- `maxnrec` : numeric(1) used with dplyr::as.data.frame en route to GRanges

### Value

- A GRanges instance
```
if (interactive()) {
  con = try(pancan_BQ()) # need CGC_BILLING set
  if (!inherits(con, "try-error")) {
    aut = as.character(1:22) # some records in BQ have missing Chromosome
    chk = mc3toGR(con, basicfilt = function(data) dplyr::filter(data, project_short_name == "TCGA-BRCA", SYMBOL == "TP53", Chromosome %in% aut))
    print(chk[,1:5]) # lots of mcol fields
    table(chk$Variant_Classification)
  }
}
```

molpo_3utr

**representation of 3’UTR MSI events in TCGA from Cortes-Ciriano et al. 2017**

**Description**

representation of 3’UTR MSI events in TCGA from Cortes-Ciriano et al. 2017

**Usage**

molpo_3utr

**Format**

SummarizedExperiment

**Note**

Supplementary data 6 from publication noted in Source. See metadata() component of this SummarizedExperiment for more details.

**Source**

[https://www.nature.com/articles/ncomms15180#Sec22](https://www.nature.com/articles/ncomms15180#Sec22)

**Examples**

molpo_3utr
molpo_5utr

---

**molpo_5utr**  
*representation of 5’UTR MSI events in TCGA from Cortes-Ciriano et al. 2017*

---

**Description**

representation of 5’UTR MSI events in TCGA from Cortes-Ciriano et al. 2017

**Usage**

molpo_5utr

**Format**

SummarizedExperiment

**Note**

Supplementary data 7 from publication noted in Source. See metadata() component of this SummarizedExperiment for more details.

**Source**

https://www.nature.com/articles/ncomms15180#Sec22

**Examples**

molpo_5utr

---

**molpo_CDS**  
*representation of MSI events in coding regions TCGA from Cortes-Ciriano et al. 2017*

---

**Description**

representation of MSI events in coding regions TCGA from Cortes-Ciriano et al. 2017

**Usage**

molpo_CDS

**Format**

SummarizedExperiment
molpo_WGS

Note

Supplementary data 5 from publication noted in Source. See metadata() component of this SummarizedExperiment for more details.

Source

https://www.nature.com/articles/ncomms15180#Sec22

Examples

molpo_CDS

molpo_WGS

representation of events detected in 708 WGS experiments TCGA from Cortes-Ciriano et al. 2017

Description

representation of events detected in 708 WGS experiments TCGA from Cortes-Ciriano et al. 2017

Usage

molpo_WGS

Format

SummarizedExperiment

Note

Supplementary data 10 from publication noted in Source. See metadata() component of this SummarizedExperiment for more details.

Source

https://www.nature.com/articles/ncomms15180#Sec22

Examples

molpo_WGS
MSIsensor.10k  

MSIsensor microsatellite instability scores for TCGA, collected from Ding et al. Cell 173(2) 2018.

---

Description

MSIsensor microsatellite instability scores for TCGA, collected from Ding et al. Cell 173(2) 2018.

Usage

MSIsensor.10k

Format

DataFrame

Source


Examples

str(BiocOncoTK::dingMSI)

---

multiviz  

visualize aspects of MSIsensor/expression relationships

---

Description

visualize aspects of MSIsensor/expression relationships

Usage

multiviz(
    tum = "MESO",
    gene = "TYMS",
    intrans = log10pl1(p = 1),
    inmeth = "auto",
    topmsi = Inf,
    indata,
    nvar = 6
)
Arguments

- **tum**: a TCGA tumor code
- **gene**: a gene symbol used in the indata data.frame
- **intrns**: an instance of the trans() transformation method of scales package
- **inmeth**: a valid setting for method parameter for geom_smooth
- **topmsi**: maximum numeric value for x-axis when plotting against MSI value
- **indata**: a data.frame instance with values for acronym, gene, msival
- **nvar**: numeric() number of variables to show in biplot

**oncoPrintISB**

*interactive interface to ComplexHeatmap oncoPrint with inputs from ISB Cancer Genomics Cloud BigQuery back end*

Description

interactive interface to ComplexHeatmap oncoPrint with inputs from ISB Cancer Genomics Cloud BigQuery back end

Usage

```r
oncoPrintISB(bq)
```

Arguments

- **bq**: an instance of `BigQueryConnection-class` authenticated for ISB Cancer Genomics Cloud access

Value

only used for side effect of running shiny app

Note

This function will start a shiny app and will generate queries to Google BigQuery tables representing TCGA.

Examples

```r
if (interactive()) {
  bcode = Sys.getenv("CGC_BILLING")
  if (nchar(bcode)>0) {
    con <- DBI::dbConnect(bigrquery::bigquery(), project = "isb-cgc",
                          dataset = "tcga_201607_beta", billing = bcode)
    oncoPrintISB(con)
  }
}
```
**Description**

pancan.clin.varnames: a data.frame with a list of variable names for clinical patient data

**Usage**

```r
pancan.clin.varnames
```

**Format**

data.frame

**Source**

pancancer-atlas in BigQuery

**Examples**

```r
BiocOncoTK::pancan.clin.varnames[1:5,]
```

---

**pancan_app**

provide a shiny app to `glimpse` structure and content of pancan atlas

---

**Description**

provide a shiny app to `glimpse` structure and content of pancan atlas

**Usage**

```r
pancan_app(dataset = "Annotated", nrecs = 5)
```

**Arguments**

- **dataset** character(1) name of a BigQuery dataset in the pancan-atlas project
- **nrecs** numeric(1) number of records to request (limited through the n= parameter to as.data.table

**Value**

currently only as a side effect of starting app

**Examples**

```r
if (interactive()) pancan_app()
```
**pancan_BQ**

*provide bigquery connection to pancancer Annotated datasets*

**Description**

provide bigquery connection to pancancer Annotated datasets

**Usage**

```r
pancan_BQ(dataset = "Annotated", billing = Sys.getenv("CGC_BILLING"), ...)```

**Arguments**

- **dataset** character(1) dataset name
- **billing** character(1) Google cloud platform billing code; authentication will be attempted when using the resulting connection
- `...` passed to `dbConnect`, for example, `quiet=TRUE`

**Value**

BigQueryConnection instance

**Examples**

```r
pancan_BQ()
```

---

**pancan_clinicalTabVarnames**

*give an interface to tablenames*

**Description**

give an interface to tablenames

**Usage**

```r
pancan_clinicalTabVarnames()
```

**Value**

interactive datatable from DT

**Examples**

```r
if (interactive()) pancan_clinicalTabVarnames()
```
**pancan_longname**

utility to help find long table names

**Description**
utility to help find long table names

**Usage**

```r
pancan_longname(guess, ...)
```

**Arguments**

- `guess`: a regexp to match the table of interest
- `...`: passed to `agrep`

**Value**

character vector of matches

**Note**

Note that `ignore.case=TRUE` is set in the function.

**Examples**

```r
pancan_longname("rnaseq")
```

---

**pancan_sampTypeMap**

helper for interpreting pancan-atlas sample type codes

**Description**

helper for interpreting pancan-atlas sample type codes

**Usage**

```r
pancan_sampTypeMap
```

**Format**

data.frame

**Note**

The sample type codes are not straightforward to interpret. Primary solid tumor is denoted “TP”, and metastatic samples are denoted “TM”. This data frame pairs code and natural language terms.
### patient_to_tumor_code

**Dataframe mapping from TCGA patient_barcode to TCGA tumor code**

**Description**

Dataframe mapping from TCGA patient_barcode to TCGA tumor code

**Usage**

```r
patient_to_tumor_code
```
pertClasses

Format
data.frame

Note
Used IDs recorded in MSISensor.10k; one is unmatched at TCGA portal metadata() component of this SummarizedExperiment for more details.

Source
https://portal.gdc.cancer.gov/exploration?uploadCaseTab=matched

Examples
head(patient_to_tumor_code)

______________________________
pertClasses enumerate perturbagen classes
______________________________

Description
enumerate perturbagen classes

Usage
pertClasses(key = Sys.getenv("CLUE_KEY"))

Arguments
key character(1) API key provided by clue.io

Value
a character vector

Examples
if (nchar(Sys.getenv("CLUE_KEY"))>0) {
  pc = pertClasses()
  head(vapply(pc, "[[", character(1), 1))
}
pert_70138  

pert_70138: a table with perturbagen information from LINCS

Description

pert_70138: a table with perturbagen information from LINCS

Usage

pert_70138

Format

data.frame

Source

GEO GSE70138 GSE70138_Broad_LINCS_pert_info.txt.gz

Examples

data(pert_70138)

---

query_clue  

run the api.clue.io API to acquire information on LINCS experiments

Description

run the api.clue.io API to acquire information on LINCS experiments

Usage

query_clue(
  service = "profiles",
  filter = list(where = (list(pert_iname = "sirolimus", cell_id = "MCF7", assay = "L1000"))),
  key = Sys.getenv("CLUE_KEY")
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>service</td>
<td>a character(1) service name</td>
</tr>
<tr>
<td>filter</td>
<td>a list to be converted to JSON for submission as a GET request</td>
</tr>
<tr>
<td>key</td>
<td>character(1) API key provided by clue.io</td>
</tr>
</tbody>
</table>
Value

API return value processed by fromJSON

Examples

```r
if (nchar(Sys.getenv("CLUE_KEY"))>0) {
  demos = clueDemos()
  nd = length(demos)
  chk = lapply(seq_len(nd), function(x) query_clue( service=names(demos)[x],
    filter=demos[[x]])
  names(chk) = names(demos)
  sapply(chk,length)
}
```

reexports

Objects exported from other packages

Description

These objects are imported from other packages. Follow the links below to see their documentation.

**dplyr**  *filter, select, tbl*

**magrittr**  *%>%*

replaceRownames

map rownames of an SE to another vocabulary

Description

map rownames of an SE to another vocabulary

Usage

```r
replaceRownames(se, sourceVocab = "ENTREZID", targetVocab = "SYMBOL")
```

Arguments

- `se` SummarizedExperiment instance
- `sourceVocab` character(1) must be a keytype of org.Hs.eg.db, defaults to `ENTREZID`
- `targetVocab` character(1) must be a column of org.Hs.eg.db
small_msi  filtered MSI data for demonstrating exploratory app

Description
filtered MSI data for demonstrating exploratory app

Usage
small_msi

Format
DataFrame

Source
MSI values from dingMSI, expression from curatedTCGAD for three genes, two tumors

Examples
head(BiocOncoTK::small_msi)

TcgaMutCounts  obtain data frame with counts of mutation per gene symbol for selected tumor type

Description
obtain data frame with counts of mutation per gene symbol for selected tumor type

Usage
TcgaMutCounts(tumor, limit = NULL, db = "isb-cgc:tcga_201607_beta", project)

Arguments
tumor character(1) defaults to 'BRCA'
limit numeric(1) defaults to NULL, appended as limit to number of records returned if non-null
db character(1) BigQuery database name
project character(1) project code

Value
table as returned by bigquery::bq_project_query
**TcgaNIndWithAnyMut**

**Note**

This function returns overall mutation count, and many individuals have multiple mutations recorded per gene.

**Examples**

```r
if (interactive()) {
  requireNamespace("bigrquery")
  tt = TcgaMutCounts("BRCA", project="cgc-05-0009") # substitute your project name
  head(tt)
} # need authentication
```

---

**TcgaNIndWithAnyMut**  
*Give count of individuals with a mutation recorded for selected tumor*

**Description**

Give count of individuals with a mutation recorded for selected tumor

**Usage**

```r
TcgaNIndWithAnyMut(
  tumor = "BRCA",
  limit = NULL,
  db = "isb-cgc:tcga_201607_beta",
  project
)
```

**Arguments**

- **tumor**: character(1) defaults to 'BRCA'
- **limit**: numeric(1) defaults to NULL, appended as limit to number of records returned if non-null
- **db**: character(1) BigQuery database name
- **project**: character(1) project code

**Value**

numeric(1)

**Examples**

```r
if (interactive()) TcgaNIndWithAnyMut(project="cgc-05-0009")
```
tumNorSet

create list with SEs for tumor and normal for a tumor/assay pairing

Description

create list with SEs for tumor and normal for a tumor/assay pairing

Usage

tumNorSet(
  bq,  
  code = "PRAD",  
  assayDataTableName = pancan_longname("rnaseq"),  
  assayValueFieldName = "normalized_count",  
  assayFeatureName = "Entrez"
)

Arguments

  bq               a BigQuery connection
  code             character(1) a TCGA tumor code, defaults to "PRAD" for prostate tumor
  assayDataTableName
                   character(1) name of table in BigQuery
  assayValueFieldName
                   character(1) field from which assay quantifications are retrieved
  assayFeatureName
                   character(1) field from which assay feature names are retrieved

Examples

if (interactive()) {
  bqcon = try(pancan_BQ())
  if (!inherits(bqcon, "try-error")) {
    tn = tumNorSet(bqcon)
    tn
  }
}
viz_msi_raw

**Description**

small app to survey MSIsensor against expression

**Usage**

```r
viz_msi_raw(df, inmeth = MASS::rlm, nvar = 3)
```

**Arguments**

- `df`: a data.frame instance
- `inmeth`: a method for geom_smooth
- `nvar`: number of variables to show in biplot

**Note**

Use ask=FALSE if running example.

**Examples**

```r
if (interactive()) viz_msi_raw(BiocOncoTK::small_msi, nvar=3)
```
Index

* datasets
  - annotTabs, 4
  - CCLE_DRUG_BROAD, 7
  - cell_70138, 8
  - darmGBMcls, 9
  - dingMSI, 10
  - fireMSI, 11
  - icd10_c, 15
  - k23sig, 15
  - kang_DNArepair, 16
  - map_tcga_ncit, 19
  - molpo_3utr, 20
  - molpo_5utr, 21
  - molpo_CDS, 21
  - molpo_WGS, 22
  - MSIsensor.10k, 23
  - pancan.clin.varnames, 25
  - pancan_sampTypeMap, 27
  - patient_to_tumor_code, 28
  - pert_70138, 30
  - small_msi, 32

* internal
  - reexports, 31
  - %>% (reexports), 31
  - %>%, 31

  - add_sym, 3
  - agrep, 27
  - annotTabs, 4
  - bindMSI, 4
  - bipg_tests, 5
  - buildPancanSE, 6, 10
  - CCLE_DRUG_BROAD, 7
  - cell_70138, 8
  - clueDemos, 8
  - clueServiceNames, 9
  - darmGBMcls, 9

  - dbConnect, 26
  - dingMSI, 10
  - featIDMapper, 10
  - filter, 31
  - filter (reexports), 31
  - fireMSI, 5, 11
  - get_plates, 11
  - ggFeatDens, 12
  - ggFeatureSegs, 13
  - ggMutDens, 14
  - icd10_c, 15
  - k23sig, 15
  - kang_DNArepair, 16
  - load_ccleNRAS, 17
  - load_NRAS_AHR (load_ccleNRAS), 17
  - load_nrasdf (load_ccleNRAS), 17
  - loadPatel, 17
  - log10pl1, 18

  - map_tcga_ncit, 19
  - mc3toGR, 19
  - molpo_3utr, 20
  - molpo_5utr, 21
  - molpo_CDS, 21
  - molpo_WGS, 22
  - MSIsensor.10k, 23
  - multiviz, 23

  - oncoPrintISB, 24

  - pancan.clin.varnames, 25
  - pancan_app, 25
  - pancan_BQ, 26
  - pancan_clinicalTabVarnames, 26
  - pancan_longname, 27
  - pancan_sampTypeMap, 27
pancan_tabulate, 28
patient_to_tumor_code, 28
pert_70138, 30
pertClasses, 29
query_clue, 30
reexports, 31
replaceRownames, 31
select, 31
select (reexports), 31
small_msi, 32
tbl, 31
tbl (reexports), 31
tcgaMutCounts, 32
tcgaNIndWithAnyMut, 33
tumNorSet, 34
viz_msi_raw, 35