Package 'RTCGA'

July 3, 2025

Title The Cancer Genome Atlas Data Integration

Version 1.39.0

Date 2022-10-28

Maintainer Marcin Kosinski <m.p.kosinski@gmail.com>

Description The Cancer Genome Atlas (TCGA) Data Portal provides a platform for researchers to search, download, and analyze data sets generated by TCGA. It contains clinical information, genomic characterization data, and high level sequence analysis of the tumor genomes. The key is to understand genomics to improve cancer care. RTCGA package offers download and integration of the variety and volume of TCGA data using patient barcode key, what enables easier data possession. This may have an benefcial infuence on impact on development of science and improvement of patients' treatment. Furthermore, RTCGA package transforms TCGA data to tidy form which is convenient to use.

BugReports https://github.com/RTCGA/RTCGA/issues

URL https://rtcga.github.io/RTCGA

License GPL-2

LazyLoad yes

LazyData yes

Depends R (>= 3.3.0)

Imports XML, RCurl, assertthat, stringi, rvest, data.table, xml2, dplyr, purrr, survival, survminer, ggplot2, ggthemes, viridis, knitr, scales, rmarkdown, htmltools

Suggests devtools, testthat, pander, Biobase, GenomicRanges, IRanges, S4Vectors, RTCGA.rnaseq, RTCGA.clinical, RTCGA.mutations, RTCGA.RPPA, RTCGA.mRNA, RTCGA.miRNASeq, RTCGA.methylation, RTCGA.CNV, magrittr, tidyr

biocViews ImmunoOncology, Software, DataImport, DataRepresentation, Preprocessing, RNASeq, Survival, DNAMethylation, PrincipalComponent, Visualization

VignetteBuilder knitr

NeedsCompilation no

RoxygenNote 7.1.1

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

git_branch devel

git_last_commit 8a320c0

git_last_commit_date 2025-04-15

Repository Bioconductor 3.22

Date/Publication 2025-07-02

Author Marcin Kosinski [aut, cre],
Przemyslaw Biecek [ctb],
Witold Chodor [ctb]

Contents

`RTCGA-package` *The Caner Genome Atlas data integration*

Description

The Cancer Genome Atlas (TCGA) Data Portal provides a platform for researchers to search, download, and analyze data sets generated by TCGA. It contains clinical information, genomic characterization data, and high level sequence analysis of the tumor genomes. The key is to understand genomics to improve cancer care. RTCGA package offers download and integration of the variety and volume of TCGA data using patient barcode key, what enables easier data possession. This may have an benefcial infuence on impact on development of science and improvement of patients' treatment. Furthermore, RTCGA package transforms TCGA data to form which is convenient to use in R statistical package. Those data transformations can be a part of statistical analysis pipeline which can be more reproducible with RTCGA

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski [aut, cre] < m.p.kosinski@gmail.com >
Przemyslaw Biecek [aut] < przemyslaw.biecek@gmail.com >
Witold Chodor [ctb] <witoldchodor@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA.

Other RTCGA: boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
## Not run:
browseVignettes('RTCGA')

## End(Not run)
```

---

boxplotTCGA *Create Boxplots for TCGA Datasets*

Description

Function creates boxplots (geom_boxplot) for TCGA Datasets.

Usage

```
boxplotTCGA(
  data,
  x,
  y,
  fill = x,
  coord.flip = TRUE,
  facet.names = NULL,
  ylab = y,
  xlab = x,
  legend.title = xlab,
  legend = "top",
  ...,
  ggtheme = theme_RTCGA()
)
```

Arguments

data	A data.frame from TCGA study containing variables to be plotted.
x	A character name of variable containing groups.
y	A character name of continous variable to be plotted.
fill	A character names of fill variable. By default, the same as x.
coord.flip	Whether to flip coordinates.
facet.names	A character of length maximum 2 containing names of variables to produce facets. See examples.
ylab	The name of y label. Remember about coord.flip.
xlab	The name of x label. Remember about coord.flip.
legend.title	A character with legend's title.
legend	A character specifying legend position. Allowed values are one of c("top", "bottom", "left", "right", "none"). Default is "top" side position. to remove the legend use legend = "none".
...	Further arguments passed to geom_boxplot.
ggtheme	a ggtheme to be used (set to NULL, if using **ggthemr** package)

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Visualizations.html.

Other RTCGA: RTCGA-package, checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
library(RTCGA)
library(RTCGA.rnaseq)
# perfrom plot
library(dplyr)
expressionsTCGA(ACC.rnaseq, BLCA.rnaseq, BRCA.rnaseq, OV.rnaseq,
  extract.cols = "MET|4233") %>%
  rename(cohort = dataset,
  MET = `MET|4233`) %>%
  #cancer samples
  filter(substr(bcr_patient_barcode, 14, 15) == "01") -> ACC_BLCA_BRCA_OV.rnaseq


boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq, "cohort", "MET")
boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq, "cohort", "log1p(MET)")
boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq, "reorder(cohort,log1p(MET), median)", "log1p(MET)")
```

```
boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq, "reorder(cohort,log1p(MET), max)", "log1p(MET)")
boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq, "reorder(cohort,log1p(MET), median)", "log1p(MET)",
            xlab = "Cohort Type", ylab = "Logarithm of MET")
boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq, "reorder(cohort,log1p(MET), median)", "log1p(MET)",
            xlab = "Cohort Type", ylab = "Logarithm of MET", legend.title = "Cohorts")
boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq, "reorder(cohort,log1p(MET), median)", "log1p(MET)",
            xlab = "Cohort Type", ylab = "Logarithm of MET",
            legend.title = "Cohorts", legend = "bottom")

## facet example
library(RTCGA.mutations)
library(dplyr)
mutationsTCGA(BRCA.mutations, OV.mutations, ACC.mutations, BLCA.mutations) %>%
  filter(Hugo_Symbol == 'TP53') %>%
  filter(substr(bcr_patient_barcode, 14, 15) == "01") %>% # cancer tissue
  mutate(bcr_patient_barcode = substr(bcr_patient_barcode, 1, 12)) ->
  ACC_BLCA_BRCA_OV.mutations

mutationsTCGA(BRCA.mutations, OV.mutations, ACC.mutations, BLCA.mutations) ->
  ACC_BLCA_BRCA_OV.mutations_all

ACC_BLCA_BRCA_OV.rnaseq %>%
  mutate(bcr_patient_barcode = substr(bcr_patient_barcode, 1, 15)) %>%
  filter(bcr_patient_barcode %in%
  substr(ACC_BLCA_BRCA_OV.mutations_all$bcr_patient_barcode, 1, 15)) %>%
  # took patients for which we had any mutation information
  # so avoided patients without any information about mutations
  mutate(bcr_patient_barcode = substr(bcr_patient_barcode, 1, 12)) %>%
  # strin_length(ACC_BLCA_BRCA_OV.mutations$bcr_patient_barcode) == 12
  left_join(ACC_BLCA_BRCA_OV.mutations,
  by = "bcr_patient_barcode") %>% #joined only with tumor patients
  mutate(TP53 = ifelse(!is.na(Variant_Classification), "Mut", "WILD")) %>%
  select(cohort, MET, TP53) -> ACC_BLCA_BRCA_OV.rnaseq_TP53mutations

boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq_TP53mutations, "reorder(cohort,log1p(MET), median)",
            "log1p(MET)", xlab = "Cohort Type", ylab = "Logarithm of MET",
            legend.title = "Cohorts", legend = "bottom", facet.names = c("TP53"))

boxplotTCGA(ACC_BLCA_BRCA_OV.rnaseq_TP53mutations, "reorder(cohort,log1p(MET), median)",
            "log1p(MET)", xlab = "Cohort Type", ylab = "Logarithm of MET",
            legend.title = "Cohorts", legend = "bottom", fill = c("TP53"))
```

---

checkTCGA *Information About Datasets from TCGA Project*

Description

The checkTCGA function let's to check

- DataSets: TCGA datasets' names for current release date and cohort.
- Dates: TCGA datasets' dates of release.

Usage

```
checkTCGA(what, cancerType, date = NULL)
```

Arguments

what	One of DataSets or Dates.
cancerType	A character of length 1 containing abbreviation (Cohort code - https://gdac.broadinstitute.org/) of types of cancers to check for.
date	A NULL or character specifying from which date informations should be checked. By default (date = NULL) the newest available date is used. All available dates can be checked on https://gdac.broadinstitute.org/runs/ or by using checkTCGA('Dates') function. Required format 'YYYY-MM-DD'.

Details

- If what='DataSets' enables to check TCGA datasets' names for current release date and cohort.
- If what='Dates' enables to check dates of TCGA datasets' releases.

Value

- If what='DataSets' a data.frame of available datasets' names (to pass to the downloadTCGA function) and sizes.
- If what='Dates' a vector of available dates to pass to the downloadTCGA function.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website https://rtcga.github.io/RTCGA/.

Other RTCGA: RTCGA-package, boxplotTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
############################

# names for current release date and cohort
checkTCGA('DataSets', 'BRCA')
## Not run:
checkTCGA('DataSets', 'OV', tail(checkTCGA('Dates'))[3])
#checkTCGA('DataSets', 'OV', checkTCGA('Dates')[5]) # error

## End(Not run)
# dates of TCGA datasets' releases.
```

```
checkTCGA('Dates')

############################
## Not run:
# TCGA datasets' names availability for
# current release date and cancer type.

releaseDate <- '2015-08-21'
cancerTypes <- c('OV', 'BRCA')

cancerTypes %>% sapply(function(element){
  grep(x = checkTCGA('DataSets', element, releaseDate)[, 1],
       pattern = 'humanmethylation450', value = TRUE) %>%
        as.vector()
        })


## End(Not run)
```

`convertTCGA` *Convert Data from RTCGA Family to Bioconductor Classes*

Description

Functions use **Biobase** (http://bioconductor.org/packages/release/bioc/html/Biobase.html) package to transform data from packages from RTCGA data family to Bioconductor classes (**RTCGA.rnaseq**, **RTCGA.RPPA**, **RTCGA.PANCAN12**, **mRNA**, **RTCGA.methylation** to ExpressionSet and **RTCGA.CNV** to GRanges). For **RTCGA.PANCAN12** there is sense to convert `expression.cb1`, `expression.cb2`, `cnv.cb`.

Usage

```
convertTCGA(dataSet, dataType = "expression")

convertPANCAN12(dataSet)
```

Arguments

`dataSet`	A data.frame to be converted to ExpressionSet or GRanges.
`dataType`	One of `expression` or `CNV` (for **RTCGA.CNV** datasets).

Details

This functionality is motivated by that we were asked to offer the data in Bioconductor-friendly classes because many users already have their data in one of the core infrastructure classes. Data of the same type in compatible containers promotes interoperability and makes it easy to combine and organize.

Bioconductor classes were designed to capitalize on the biological structure of the data. If data have a range-based component it's natural, for Bioconductor users, to store and access these as a GRanges where they can extract position, strand etc. in the same way. Similarly for ExpressionSet. This class holds expression data along with experiment metadata and comes with built in accessors to extract and manipulate data. The idea is to offer a common API to the data; extracting the start

position in a GRanges is always start(). With a data.frame it is different each time (unless select() is implemented) as the column names and organization of data can be different.

AnnotationHub and the soon to come ExperimentHub will host many different types of data. A primary goal moving forward is to offer similar data in a consistent format. For example, CNV data in AnnotationHub is offered as a GRanges and as more CNV are added we will ask that they too are packaged as GRanges. The aim is that streamlined data on the back-end will make for a more intuitive experience on the front-end.

Value

Functions return an ExpressionSet or a GRanges for **RTCGA.CNV**

Biobase and GenomicRanges

This function use tools from the fantastic **Biobase** (and **GenomicRanges** for CNV) package, so you'll need to make sure to have it installed.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
########
########
# Expression data
########
########
library(RTCGA.rnaseq)
library(Biobase)
convertTCGA(BRCA.rnaseq) -> BRCA.rnaseq_ExpressionSet
## Not run:
library(RTCGA.PANCAN12)
convertPANCAN12(expression.cb1) -> PANCAN12_ExpressionSet
library(RTCGA.RPPA)
convertTCGA(BRCA.RPPA) -> BRCA.RPPA_ExpressionSet
library(RTCGA.methylation)
convertTCGA(BRCA.methylation) -> BRCA.methylation_ExpressionSet
library(RTCGA.mRNA)
convertTCGA(BRCA.mRNA) -> BRCA.mRNA_ExpressionSet
########
########
```

```
# CNV
########
########
library(RTCGA.CNV)
library(GRanges)
convertTCGA(BRCA.CNV, "CNV") -> BRCA.CNV_GRanges


## End(Not run)
```

`datasetsTCGA` *RTCGA - The Family of R Packages with Data from The Cancer Genome Atlas Study*

Description

Snapshots of the clinical, mutations, CNVs, rnaseq, RPPA, mRNA, miRNASeq and methylation datasets from the `2016-01-28` release date (check all dates of release with `checkTCGA('Dates')`) are included in the `RTCGA` family (factory) that contains below packages:

- **RTCGA.rnaseq.20160128** rnaseq.20160128
- **RTCGA.clinical.20160128** clinical.20160128
- **RTCGA.mutations.20160128** mutations.20160128
- **RTCGA.CNV.20160128** CNV.20160128
- **RTCGA.RPPA.20160128** RPPA.20160128
- **RTCGA.mRNA.20160128** mRNA.20160128
- **RTCGA.miRNASeq.20160128** miRNASeq.20160128
- **RTCGA.methylation.20160128** methylation.20160128

Snapshots of the clinical, mutations, CNVs, rnaseq, RPPA, mRNA, miRNASeq and methylation datasets from the `2015-11-01` release date (check all dates of release with `checkTCGA('Dates')`) are also included in the `RTCGA` family (factory).

- **RTCGA.rnaseq** rnaseq
- **RTCGA.clinical** clinical
- **RTCGA.mutations** mutations
- **RTCGA.CNV** CNV
- **RTCGA.RPPA** RPPA
- **RTCGA.mRNA** mRNA
- **RTCGA.miRNASeq** miRNASeq
- **RTCGA.methylation** methylation
- **RTCGA.PANCAN12** (not from TCGA)

Details

For more detailed information visit **RTCGA** family website https://rtcga.github.io/RTCGA. One can install all data packages with installTCGA.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski [aut, cre] < m.p.kosinski@gmail.com >
Przemyslaw Biecek [aut] < przemyslaw.biecek@gmail.com >
Witold Chodor [aut] < witoldchodor@gmail.com >

See Also

RTCGA website http://rtcga.github.io/RTCGA.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
# installation of packages containing snapshots
# of TCGA project's datasets

## Not run:

## RTCGA GitHub development newest versions
library(RTCGA)
?installTCGA

## Bioconductor releases for data from 2016-01-28 release
source('http://bioconductor.org/biocLite.R')
biocLite(RTCGA.clinical.20160128)
biocLite(RTCGA.mutations.20160128)
biocLite(RTCGA.rnaseq.20160128)
biocLite(RTCGA.CNV.20160128)
biocLite(RTCGA.RPPA.20160128)
biocLite(RTCGA.mRNA.20160128)
biocLite(RTCGA.miRNASeq.20160128)
biocLite(RTCGA.methylation.20160128)

## Bioconductor releases for data from 2015-11-01 release
source('http://bioconductor.org/biocLite.R')
biocLite(RTCGA.clinical)
biocLite(RTCGA.mutations)
biocLite(RTCGA.rnaseq)
biocLite(RTCGA.CNV)
biocLite(RTCGA.RPPA)
biocLite(RTCGA.mRNA)
biocLite(RTCGA.miRNASeq)
biocLite(RTCGA.methylation)

# use cases and examples + more data info
browseVignettes('RTCGA')

## End(Not run)
```

`downloadTCGA` *Download TCGA Data*

Description

Enables to download TCGA data from specified dates of releases of concrete Cohorts of cancer types. Pass a name of required dataset to the `dataSet` parameter. By default the Merged Clinical dataSet is downloaded (value `dataSet = 'Merge_Clinical.Level_1'`) from the newest available date of the release.

Usage

```
downloadTCGA(
  cancerTypes,
  dataSet = "Merge_Clinical.Level_1",
  destDir,
  date = NULL,
  untarFile = TRUE,
  removeTar = TRUE,
  allDataSets = FALSE
)
```

Arguments

`cancerTypes`	A character vector containing abbreviations (Cohort code) of types of cancers to download from https://gdac.broadinstitute.org/. For easy access from R check details below.
`dataSet`	A part of the name of dataSet to be downloaded from https://gdac.broadinstitute.org/runs/. By default the Merged Clinical dataSet is downloaded (value `dataSet = 'Merge_Clinical.Level_1`. Available datasets' names can be checked using checkTCGA function.
`destDir`	A character specifying a directory into which `dataSets` will be downloaded.
`date`	A `NULL` or character specifying from which date `dataSets` should be downloaded. By default (`date = NULL`) the newest available date is used. All available dates can be checked on https://gdac.broadinstitute.org/runs/ or by using checkTCGA function. Required format `'YYYY-MM-DD'`.
`untarFile`	Logical - should the downloaded file be untarred. Default is `TRUE`.
`removeTar`	Logical - should the downloaded `.tar` file be removed after untarring. Default is `TRUE`.
`allDataSets`	Logical - should download all datasets matching `dataSet` parameter or only the first one (without `FFPE` phrase if possible).

Details

All cohort names can be checked using: `sub( x = names( infoTCGA() ), '-counts', '' )`.

Value

No values. It only downloads files.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website https://rtcga.github.io/RTCGA/articles/Data_Download.html.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
dir.create('hre')

downloadTCGA(cancerTypes = 'ACC',
             dataSet = 'miR_gene_expression',
             destDir = 'hre',
             date = tail(checkTCGA('Dates'), 2)[1])

## Not run:
downloadTCGA(cancerTypes = c('BRCA', 'OV'),
             destDir = 'hre',
             date = tail(checkTCGA('Dates'), 2)[1])

## End(Not run)
```

---

expressionsTCGA *Gather Expressions for TCGA Datasets*

Description

Function gathers expressions over multiple TCGA datasets and extracts expressions for desired genes. See rnaseq, mRNA, RPPA, miRNASeq, methylation.

Usage

```
expressionsTCGA(..., extract.cols = NULL, extract.names = TRUE)
```

Arguments

...	A data.frame or data.frames from TCGA study containing expressions informations.
extract.cols	A character specifing the names of columns to be extracted with bcr_patient_barcode. If NULL (by default) all columns are returned.
extract.names	Logical, whether to extract names of passed data.frames in ....

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Note

Input data.frames should contain column `bcr_patient_barcode` if `extract.cols` is specified.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Visualizations.html.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
## for all examples
library(dplyr)
library(tidyr)
library(ggplot2)

## RNASeq expressions
library(RTCGA.rnaseq)
expressionsTCGA(BRCA.rnaseq, OV.rnaseq, HNSC.rnaseq,
                extract.cols = "VENTX|27287") %>%
  rename(cohort = dataset,
         VENTX = `VENTX|27287`) %>%
 filter(substr(bcr_patient_barcode, 14, 15) == "01") %>% #cancer samples
  ggplot(aes(y = log1p(VENTX),
             x = reorder(cohort, log1p(VENTX), median),
             fill = cohort)) +
  geom_boxplot() +
  theme_RTCGA() +
  scale_fill_brewer(palette = "Dark2")

## mRNA expressions
library(tidyr)
library(RTCGA.mRNA)
expressionsTCGA(BRCA.mRNA, COAD.mRNA, LUSC.mRNA, UCEC.mRNA,
                extract.cols = c("ARHGAP24", "TRAV20")) %>%
  rename(cohort = dataset) %>%
  select(-bcr_patient_barcode) %>%
  gather(key = "mRNA", value = "value", -cohort)  %>%
  ggplot(aes(y = value,
             x = reorder(cohort, value, mean),
             fill = cohort)) +
  geom_boxplot() +
  theme_RTCGA() +
  scale_fill_brewer(palette = "Set3") +
  facet_grid(mRNA~.) +
```

```
  theme(legend.position = "top")


## RPPA expressions
library(RTCGA.RPPA)
expressionsTCGA(ACC.RPPA, BLCA.RPPA, BRCA.RPPA,
    extract.cols = c("4E-BP1_pS65", "4E-BP1")) %>%
  rename(cohort = dataset) %>%
  select(-bcr_patient_barcode) %>%
  gather(key = "RPPA", value = "value", -cohort)  %>%
  ggplot(aes(fill = cohort,
             y = value,
             x = RPPA)) +
  geom_boxplot() +
  theme_dark(base_size = 15) +
  scale_fill_manual(values = c("#eb6420", "#207de5", "#fbca04")) +
  coord_flip() +
  theme(legend.position = "top") +
  geom_jitter(alpha = 0.5, col = "white", size = 0.6, width = 0.7)



## miRNASeq expressions
library(RTCGA.miRNASeq)
# miRNASeq has bcr_patienct_barcode in rownames...
mutate(ACC.miRNASeq,
   bcr_patient_barcode = substr(rownames(ACC.miRNASeq), 1, 25)) -> ACC.miRNASeq.bcr
mutate(CESC.miRNASeq,
   bcr_patient_barcode = substr(rownames(CESC.miRNASeq), 1, 25)) -> CESC.miRNASeq.bcr
mutate(CHOL.miRNASeq,
   bcr_patient_barcode = substr(rownames(CHOL.miRNASeq), 1, 25)) -> CHOL.miRNASeq.bcr
mutate(LAML.miRNASeq,
   bcr_patient_barcode = substr(rownames(LAML.miRNASeq), 1, 25)) -> LAML.miRNASeq.bcr
mutate(PAAD.miRNASeq,
   bcr_patient_barcode = substr(rownames(PAAD.miRNASeq), 1, 25)) -> PAAD.miRNASeq.bcr
mutate(THYM.miRNASeq,
   bcr_patient_barcode = substr(rownames(THYM.miRNASeq), 1, 25)) -> THYM.miRNASeq.bcr
mutate(LGG.miRNASeq,
   bcr_patient_barcode = substr(rownames(LGG.miRNASeq), 1, 25)) -> LGG.miRNASeq.bcr
mutate(STAD.miRNASeq,
   bcr_patient_barcode = substr(rownames(STAD.miRNASeq), 1, 25)) -> STAD.miRNASeq.bcr


expressionsTCGA(ACC.miRNASeq.bcr, CESC.miRNASeq.bcr, CHOL.miRNASeq.bcr,
             LAML.miRNASeq.bcr, PAAD.miRNASeq.bcr, THYM.miRNASeq.bcr,
             LGG.miRNASeq.bcr, STAD.miRNASeq.bcr,
  extract.cols = c("machine", "hsa-mir-101-1", "miRNA_ID")) %>%
                rename(cohort = dataset) %>%
   filter(miRNA_ID == "read_count") %>%
   select(-bcr_patient_barcode, -miRNA_ID) %>%
   gather(key = "key", value = "value", -cohort, -machine) %>%
   mutate(value = as.numeric(value)) %>%
   ggplot(aes(x = cohort,
              y = log1p(value),
              fill = as.factor(machine)) )+
   geom_boxplot() +
   theme_RTCGA(base_size = 13) +
```

```
  coord_flip() +
  theme(legend.position = "top") +
  scale_fill_brewer(palette = "Paired") +
  ggtitle("hsa-mir-101-1")
```

---

heatmapTCGA *Create Heatmaps for TCGA Datasets*

Description

Function creates heatmaps (geom_tile) for TCGA Datasets.

Usage

```
heatmapTCGA(
  data,
  x,
  y,
  fill,
  legend.title = "Expression",
  legend = "right",
  title = "Heatmap of expression",
  facet.names = NULL,
  tile.size = 0.1,
  tile.color = "white",
  ...
)
```

Arguments

data	A data.frame from TCGA study containing variables to be plotted.
x, y	A character name of variable containing groups.
fill	A character names of fill variable.
legend.title	A character with legend's title.
legend	A character specifying legend position. Allowed values are one of c("top", "bottom", "left", "right", "none"). Default is "top" side position. to remove the legend use legend = "none".
title	A character with plot title.
facet.names	A character of length maximum 2 containing names of variables to produce facets. See examples.
tile.size, tile.color	A size and color passed to geom_tile.
...	Further arguments passed to geom_tile.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Note

heatmapTCGA uses scale_fill_viridis from **viridis** package which is a port of the new matplotlib color maps (**viridis** - the default -, magma, plasma and inferno) to R. matplotlib https://matplotlib.org/ is a popular plotting library for python. These color maps are designed in such a way that they will analytically be perfectly perceptually-uniform, both in regular form and also when converted to black-and-white. They are also designed to be perceived by readers with the most common form of color blindness.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Visualizations.html.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
library(RTCGA.rnaseq)
# perfrom plot
library(dplyr)


expressionsTCGA(ACC.rnaseq, BLCA.rnaseq, BRCA.rnaseq, OV.rnaseq,
                extract.cols = c("MET|4233", "ZNF500|26048", "ZNF501|115560")) %>%
  rename(cohort = dataset,
         MET = `MET|4233`) %>%
  #cancer samples
  filter(substr(bcr_patient_barcode, 14, 15) == "01") %>%
  mutate(MET = cut(MET,
   round(quantile(MET, probs = seq(0,1,0.25)), -2),
   include.lowest = TRUE,
   dig.lab = 5)) -> ACC_BLCA_BRCA_OV.rnaseq

ACC_BLCA_BRCA_OV.rnaseq %>%
  select(-bcr_patient_barcode) %>%
  group_by(cohort, MET) %>%
  summarise_each(funs(median)) %>%
  mutate(ZNF500 = round(`ZNF500|26048`),
         ZNF501 = round(`ZNF501|115560`)) -> ACC_BLCA_BRCA_OV.rnaseq.medians
heatmapTCGA(ACC_BLCA_BRCA_OV.rnaseq.medians,
  "cohort", "MET", "ZNF500", title = "Heatmap of ZNF500 expression")

## facet example
library(RTCGA.mutations)
library(dplyr)
mutationsTCGA(BRCA.mutations, OV.mutations, ACC.mutations, BLCA.mutations) %>%
  filter(Hugo_Symbol == 'TP53') %>%
  filter(substr(bcr_patient_barcode, 14, 15) == "01") %>% # cancer tissue
  mutate(bcr_patient_barcode = substr(bcr_patient_barcode, 1, 12)) ->
   ACC_BLCA_BRCA_OV.mutations
```

```
mutationsTCGA(BRCA.mutations, OV.mutations, ACC.mutations, BLCA.mutations) ->
  ACC_BLCA_BRCA_OV.mutations_all

ACC_BLCA_BRCA_OV.rnaseq %>%
  mutate(bcr_patient_barcode = substr(bcr_patient_barcode, 1, 15)) %>%
  filter(bcr_patient_barcode %in%
  substr(ACC_BLCA_BRCA_OV.mutations_all$bcr_patient_barcode, 1, 15)) %>%
  # took patients for which we had any mutation information
  # so avoided patients without any information about mutations
  mutate(bcr_patient_barcode = substr(bcr_patient_barcode, 1, 12)) %>%
  # strin_length(ACC_BLCA_BRCA_OV.mutations$bcr_patient_barcode) == 12
  left_join(ACC_BLCA_BRCA_OV.mutations,
  by = "bcr_patient_barcode") %>% #joined only with tumor patients
  mutate(TP53 = ifelse(!is.na(Variant_Classification), "Mut", "WILD")) %>%
  select(-bcr_patient_barcode, -Variant_Classification, -dataset, -Hugo_Symbol) %>%
  group_by(cohort, MET, TP53) %>%
  summarise_each(funs(median)) %>%
  mutate(ZNF501 = round(`ZNF501|115560`)) ->
  ACC_BLCA_BRCA_OV.rnaseq_TP53mutations_ZNF501medians

heatmapTCGA(ACC_BLCA_BRCA_OV.rnaseq_TP53mutations_ZNF501medians, "cohort", "MET",
            fill = "ZNF501", facet.names = "TP53",
            title = "Heatmap of ZNF501 expression")
heatmapTCGA(ACC_BLCA_BRCA_OV.rnaseq_TP53mutations_ZNF501medians, "TP53", "MET",
            fill = "ZNF501", facet.names = "cohort",
            title = "Heatmap of ZNF501 expression")
heatmapTCGA(ACC_BLCA_BRCA_OV.rnaseq_TP53mutations_ZNF501medians, "TP53", "cohort",
            fill = "ZNF501", facet.names = "MET",
            title = "Heatmap of ZNF501 expression")
```

`infoTCGA` *Information About Cohorts from TCGA Project*

Description

Function restores codes and counts for each cohort from TCGA project.

Usage

```
infoTCGA()
```

Value

A list with a tabular information from https://gdac.broadinstitute.org/.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website https://rtcga.github.io/RTCGA/articles/Data_Download.html.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
infoTCGA()
library(magrittr)
(cohorts <- infoTCGA() %>%
rownames() %>%
   sub('-counts', '', x=.))

# in knitr chunk -> results='asis'
knitr::kable(infoTCGA())
```

---

installTCGA *Install Data Packages from RTCGA Family*

Description

Function installs data packages from https://github.com/RTCGA/. Packages are listed in datasetsTCGA.

Usage

```
installTCGA(
  packages = c("RTCGA.clinical.20160128", "RTCGA.mutations.20160128",
    "RTCGA.rnaseq.20160128", "RTCGA.RPPA.20160128", "RTCGA.mRNA.20160128",
    "RTCGA.CNV.20160128", "RTCGA.miRNASeq.20160128", "RTCGA.PANCAN12.20160128",
    "RTCGA.methylation.20160128"),
  build_vignettes = TRUE,
  ...
)
```

Arguments

packages	A character specifing the names of the data packages to be installed. By default installs all packages from .20160128 release.
build_vignettes	Should vignettes be build.
...	Further arguments passed to install_github.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
## Not run:
installTCGA() # it installs all!!! of them
installTCGA('RTCGA.clinical.20160128')

## End(Not run)
```

---

kmTCGA *Plot Kaplan-Meier Estimates of Survival Curves for Survival Data*

Description

Plots Kaplan-Meier estimates of survival curves for survival data.

Usage

```
kmTCGA(
  x,
  times = "times",
  status = "patient.vital_status",
  explanatory.names = "1",
  main = "Survival Curves",
  risk.table = TRUE,
  risk.table.y.text = FALSE,
  conf.int = TRUE,
  return.survfit = FALSE,
  pval = FALSE,
  ggtheme = theme_RTCGA(),
  ...
)
```

Arguments

x	A data.frame containing survival information. See survivalTCGA.
times	The name of time variable.
status	The name of status variable.
explanatory.names	Names of explanatory variables to use in survival curves plot.

main	Title of the plot.
risk.table	Whether to show risk tables.
risk.table.y.text	Whether to show long strata names in legend of the risk table.
conf.int	Whether to show confidence intervals.
return.survfit	Should return survfit object additionaly to survival plot?
pval	Whether to add p-value of the log-rank test to the plot?
ggtheme	a ggtheme to be used (set to NULL, if using **ggthemr** package)
...	Further arguments passed to ggsurvplot.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Visualizations.html.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
## Extracting Survival Data
library(RTCGA.clinical)
survivalTCGA(BRCA.clinical, OV.clinical, extract.cols = "admin.disease_code") -> BRCAOV.survInfo

## Kaplan-Meier Survival Curves
kmTCGA(BRCAOV.survInfo, explanatory.names = "admin.disease_code",  pval = TRUE)

kmTCGA(BRCAOV.survInfo, explanatory.names = "admin.disease_code", main = "",
       xlim = c(0,4000))

# first munge data, then extract survival info
library(dplyr)
BRCA.clinical %>%
    filter(patient.drugs.drug.therapy_types.therapy_type %in%
               c("chemotherapy", "hormone therapy")) %>%
    rename(therapy = patient.drugs.drug.therapy_types.therapy_type) %>%
    survivalTCGA(extract.cols = c("therapy"))  -> BRCA.survInfo.chemo

# first extract survival info, then munge data
    survivalTCGA(BRCA.clinical,
                 extract.cols = c("patient.drugs.drug.therapy_types.therapy_type"))  %>%
    filter(patient.drugs.drug.therapy_types.therapy_type %in%
               c("chemotherapy", "hormone therapy")) %>%
   rename(therapy = patient.drugs.drug.therapy_types.therapy_type) -> BRCA.survInfo.chemo
```

```
kmTCGA(BRCA.survInfo.chemo, explanatory.names = "therapy",
       xlim = c(0, 3000), conf.int = FALSE)
```

`mutationsTCGA` *Gather Mutations for TCGA Datasets*

Description

Function gathers mutations over multiple TCGA datasets and extracts mutations and further informations about them for desired genes. See mutations.

Usage

```
mutationsTCGA(
  ...,
 extract.cols = c("Hugo_Symbol", "Variant_Classification", "bcr_patient_barcode"),
  extract.names = TRUE,
  unique = TRUE
)
```

Arguments

`...`	A data.frame or data.frames from TCGA study containing mutations information (**RTCGA.mutations**).
`extract.cols`	A character specifing the names of columns to be extracted with `bcr_patient_barcode`. If `NULL` all columns are returned.
`extract.names`	Logical, whether to extract names of passed data.frames in `...`.
`unique`	Should the outputed data be unique. By default it's `TRUE`.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Note

Input data.frames should contain column `bcr_patient_barcode` if `extract.cols` is specified.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Visualizations.html.

Other RTCGA: `RTCGA-package`, `boxplotTCGA()`, `checkTCGA()`, `convertTCGA()`, `datasetsTCGA`, `downloadTCGA()`, `expressionsTCGA()`, `heatmapTCGA()`, `infoTCGA()`, `installTCGA()`, `kmTCGA()`, `pcaTCGA()`, `readTCGA()`, `survivalTCGA()`, `theme_RTCGA()`

Examples

```
library(RTCGA.mutations)
library(dplyr)
mutationsTCGA(BRCA.mutations, OV.mutations) %>%
  filter(Hugo_Symbol == 'TP53') %>%
  filter(substr(bcr_patient_barcode, 14, 15) == "01") %>% # cancer tissue
  mutate(bcr_patient_barcode = substr(bcr_patient_barcode, 1, 12)) -> BRCA_OV.mutations

library(RTCGA.clinical)
survivalTCGA(BRCA.clinical, OV.clinical, extract.cols = "admin.disease_code") %>%
  rename(disease = admin.disease_code)-> BRCA_OV.clinical

BRCA_OV.clinical %>%
  left_join(BRCA_OV.mutations,
  by = "bcr_patient_barcode") %>%
  mutate(TP53 = ifelse(!is.na(Variant_Classification), "Mut",
 "WILDorNOINFO")) -> BRCA_OV.clinical_mutations

BRCA_OV.clinical_mutations %>%
  select(times, patient.vital_status, disease, TP53) -> BRCA_OV.2plot
kmTCGA(BRCA_OV.2plot, explanatory.names = c("TP53", "disease"),
       break.time.by = 400, xlim = c(0,2000))
```

---

pcaTCGA *Plot Two Main Components of Principal Component Analysis*

Description

Plots Two Main Components of Principal Component Analysis

Usage

```
pcaTCGA(
  x,
  group.names,
  title = "",
  return.pca = FALSE,
  scale = TRUE,
  center = TRUE,
  var.scale = 1,
  obs.scale = 1,
  ellipse = TRUE,
  circle = TRUE,
  var.axes = FALSE,
  alpha = 0.8,
  add.lines = TRUE,
  ggtheme = theme_RTCGA(),
  ...
)
```

Arguments

x	A data.frame or matrix containing i.e. expressions information. See expressionsTCGA.
group.names	Names of group variable to use in labels of the plot.
title	The title of a plot.
return.pca	Should return pca object additionaly to pca plot?
scale	As in prcomp.
center	As in prcomp.
var.scale	As in ggbiplot.
obs.scale	As in ggbiplot.
ellipse	As in ggbiplot.
circle	As in ggbiplot.
var.axes	As in ggbiplot.
alpha	As in ggbiplot.
add.lines	Should axis lines be added to plot.
ggtheme	a ggtheme to be used (set to NULL, if using **ggthemr** package)
...	Further arguments passed to prcomp.

Value

If return.pca = TRUE then a list containing a PCA plot (of class ggplot) and a pca model, the result of prcomp function. If not, then only PCA plot is returned.

ggbiplot

This function is based on https://github.com/vqv/ggbiplot which had to be copied to **RTCGA** because Bioconductor does not support remote dependencies from GitHub.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Visualizations.html.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), readTCGA(), survivalTCGA(), theme_RTCGA()

Examples

```
## Not run:
library(dplyr)
## RNASeq expressions
library(RTCGA.rnaseq)
expressionsTCGA(BRCA.rnaseq, OV.rnaseq, HNSC.rnaseq) %>%
  rename(cohort = dataset) %>%
  filter(substr(bcr_patient_barcode, 14, 15) == "01") -> BRCA.OV.HNSC.rnaseq.cancer

pcaTCGA(BRCA.OV.HNSC.rnaseq.cancer, "cohort")
pcaTCGA(BRCA.OV.HNSC.rnaseq.cancer, "cohort", add.lines = FALSE)
pcaTCGA(BRCA.OV.HNSC.rnaseq.cancer, "cohort", return.pca = TRUE) -> pca.rnaseq
pca.rnaseq$plot
pca.rnaseq$pca

## End(Not run)
```

---

readTCGA *Read TCGA data to the tidy Format*

Description

`readTCGA` function allows to read unzipped files:

- clinical data - `Merge_Clinical.Level_1`
- rnaseq data (genes' expressions) - `rnaseqv2__illuminahiseq_rnaseqv2`
- genes' mutations data - `Mutation_Packager_Calls.Level`
- Reverse phase protein array data (RPPA) - `protein_normalization__data.Level_3`
- Merge transcriptome agilent data (mRNA) - `Merge_transcriptome__agilentg4502a_07_3__unc_edu__Level_3`
- miRNASeq data - `Merge_mirnaseq__illuminaga_mirnaseq__bcgsc_ca__Level_3__miR_gene_expression__` or `"Merge_mirnaseq__illuminahiseq_mirnaseq__bcgsc_ca__Level_3__miR_gene_expression__data.Leve`
- methylation data - `Merge_methylation__humanmethylation27`
- isoforms data - `Merge_rnaseqv2__illuminahiseq_rnaseqv2__unc_edu__Level_3__RSEM_isoforms_normali`
- CNV data - `segmented_scna_minus_germline_cnv_hg19`

from TCGA project. Those files can be easily downloded with downloadTCGA function. See examples.

Usage

```
readTCGA(path, dataType, ...)
```

Arguments

`path`	See details and examples.
`dataType`	One of `'clinical'`, `'rnaseq'`, `'mutations'`, `'RPPA'`, `'mRNA'`, `'miRNASeq'`, `'methylation'`, `'isoforms'`, `'CNV'` depending on which type of data user is trying to read in the tidy format.
`...`	Further arguments passed to the as.data.frame.

Details

All cohort names can be checked using: `sub( x = names( infoTCGA() ), '-counts', '')`.

Parameter path specification:

- If `dataType = 'clinical'` a path to a `cancerType.clin.merged.txt` file.
- If `dataType = 'mutations'` a path to the unzziped folder `Mutation_Packager_Calls.Level` containing `.maf` files.
- If `dataType = 'rnaseq'` a path to the uzziped file `rnaseqv2__illuminahiseq_rnaseqv2__unc_edu__Level_3__`
- If `dataType = 'RPPA'` a path to the unzipped file in folder `protein_normalization__data.Level_3`.
- If `dataType = 'mRNA'` a path to the unzipped file `cancerType.transcriptome__agilentg4502a_07_3__unc_edu`
- If `dataType = 'miRNASeq'` a path to unzipped files `cancerType.mirnaseq__illuminahiseq_mirnaseq__bcgsc_` or `cancerType.mirnaseq__illuminaga_mirnaseq__bcgsc_ca__Level_3__miR_gene_expression__data.da`
- If `dataType = 'methylation'` a path to unzipped files `cancerType.methylation__humanmethylation27__jhu_`
- If `dataType = 'isoforms'` a path to unzipped files `cancerType.rnaseqv2__illuminahiseq_rnaseqv2__unc_ed`
- If `dataType = 'CNV'` a path to unzipped files `cancerType.Merge_snp__genome_wide_snp_6__broad_mit_edu__`

Value

An output is a `data.frame` with `dataType` data.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

Witold Chodor, <witoldchodor@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Data_Download.html.

Other RTCGA: `RTCGA-package`, `boxplotTCGA()`, `checkTCGA()`, `convertTCGA()`, `datasetsTCGA`, `downloadTCGA()`, `expressionsTCGA()`, `heatmapTCGA()`, `infoTCGA()`, `installTCGA()`, `kmTCGA()`, `mutationsTCGA()`, `pcaTCGA()`, `survivalTCGA()`, `theme_RTCGA()`

Examples

```
## Not run:

##############
##### clinical
##############

dir.create('data')

# downloading clinical data
# dataset = "clinical" is default parameter so we may omit it
downloadTCGA(cancerTypes = c('BRCA', 'OV'),
             destDir = 'data' )
```

```
# shorten paths so that they are shorter than 256 signs - windows issue
 list.files("data", full.names = TRUE) %>%
   file.rename(to = substr(., start = 1, stop = 50))

# reading datasets
sapply(c('BRCA', 'OV'), function(element){
 path <- list.files('data', recursive = TRUE,
                    full.names = TRUE,
                    patten = "clin.merged.txt")
 assign(value = readTCGA( path, 'clinical' ),
        x = paste0(element, '.clin.data'),
        envir = .GlobalEnv)})

############
##### rnaseq
############

dir.create('data2')

# downloading rnaseq data
downloadTCGA(cancerTypes = 'BRCA',
             dataSet = 'Level_3__RSEM_genes_normalized',
             destDir = 'data2')

# shorten paths so that they are shorter than 256 signs - windows issue
list.files("data2", full.names = TRUE) %>%
   file.rename(to = substr(., start = 1, stop = 50))

path_rnaseq <- list.files('data2', recursive = TRUE,
                          full.names = TRUE,
                          patten = 'illuminahiseq')
readTCGA(path = pathRNA, dataType = 'rnaseq') -> rnaseq_data


###############
##### mutations
###############

# Example directory in which untarred data will be stored
dir.create('data3')


downloadTCGA(cancerTypes = 'OV',
             dataSet = 'Mutation_Packager_Calls.Level',
             destDir = 'data3')

# reading data
list.files('data3', recursive = TRUE) -> directory

readTCGA(directory, 'mutations') -> mut_file

#################
##### methylation
#################

# Example directory in which untarred data will be stored
dir.create('data4')
```

```
# Download KIRP methylation data and store it in data4 folder
cancerType = "KIRP"
downloadTCGA(cancerTypes = cancerType,
             dataSet = "Merge_methylation__humanmethylation27",
             destDir = "data4")

# Shorten path of subdirectory with KIRP methylation data
list.files(path = "data4", full.names = TRUE) %>%
    file.rename(to = file.path("data4", paste0(cancerType, ".methylation")))

# Remove manifest.txt file
list.files(path = "data4", full.names = TRUE,
           recursive = TRUE, pattern = "MANIFEST") %>%
           file.remove()

# Read KIRP methylation data
path <- list.files(path = "data4", full.names = TRUE, recursive = TRUE)
KIRP.methylation <- readTCGA(path, dataType = "methylation")


##########
##### RPPA
##########

# Directory in which untarred data will be stored
dir.create('data5')

# Download BRCA RPPA data and store it in data5 folder
cancerType = "BRCA"
downloadTCGA(cancerTypes = cancerType,
             dataSet = "protein_normalization__data.Level_3",
             destDir = "data5")

# Shorten path of subdirectory with BRCA RPPA data
list.files(path = "data5", full.names = TRUE) %>%
    file.rename(from = ., to = file.path("data5", paste0(cancerType, ".RPPA")))

# Remove manifest.txt file
list.files(path = "data5", full.names = TRUE,
           recursive = TRUE, pattern = "MANIFEST") %>%
           file.remove()

# Read BRCA RPPA data
path <- list.files(path = "data5", full.names = TRUE, recursive = TRUE)
BRCA.RPPA <- readTCGA(path, dataType = "RPPA")


##########
##### mRNA
##########

# Directory in which untarred data will be stored
dir.create('data6')

# Download UCEC mRNA data and store it in data6 folder
cancerType = "UCEC"
```

```
downloadTCGA(cancerTypes = cancerType,
             dataSet = "agilentg4502a_07_3__unc_edu__Level_3",
             destDir = "data6")

# Shorten path of subdirectory with UCEC mRNA data
list.files(path = "data6", full.names = TRUE) %>%
    file.rename(from = ., to = file.path("data6",paste0(cancerType, ".mRNA")))

# Remove manifest.txt file
list.files(path = "data6", full.names = TRUE,
           recursive = TRUE, pattern = "MANIFEST") %>%
           file.remove()

# Read UCEC mRNA data
path <- list.files(path = "data6", full.names = TRUE, recursive = TRUE)
UCEC.mRNA <- readTCGA(path, dataType = "mRNA")

###############
##### miRNASeq
###############

# Directory in which untarred data will be stored
dir.create('data7')

# Download BRCA miRNASeq data and store it in data7 folder
# Remember that miRNASeq data are produced by two machines:
# Illumina Genome Analyzer and Illumina HiSeq 2000 machines
cancerType <- "BRCA"
downloadTCGA(cancerTypes = cancerType,
dataSet = paste0("Merge_mirnaseq__illuminaga_mirnaseq__bcgsc",
                 "_ca__Level_3__miR_gene_expression__data.Level_3"),
             destDir = "data7")

downloadTCGA(cancerTypes = cancerType,
dataSet = paste0("Merge_mirnaseq__illuminahiseq_mirnaseq__",
                  "bcgsc_ca__Level_3__miR_gene_expression__data.Level_3"),
             destDir = "data7")

# Shorten path of subdirectory with BRCA miRNASeq data
list.files(path = "data7", full.names = TRUE) %>%
    sapply(function(path){
        if (grepl(pattern = "illuminaga", path)){
            file.rename(from = grep(pattern = "illuminaga", path, value = TRUE),
                     to = file.path("data7",paste0(cancerType, ".miRNASeq.illuminaga")))
        } else if (grepl(pattern = "illuminahiseq", path)){
            file.rename(from = grep(pattern = "illuminahiseq", path, value = TRUE),
                   to = file.path("data7",paste0(cancerType, ".miRNASeq.illuminahiseq")))
        }
    })

# Remove manifest.txt file
list.files(path = "data6", full.names = TRUE,
           recursive = TRUE, pattern = "MANIFEST") %>%
           file.remove()

# Read BRCA miRNASeq data
path <- list.files(path = "data7", full.names = TRUE, recursive = TRUE)
```

```
path_illuminaga <- grep("illuminaga", path, fixed = TRUE, value = TRUE)
path_illuminahiseq <- grep("illuminahiseq", path, fixed = TRUE, value = TRUE)

BRCA.miRNASeq.illuminaga <- readTCGA(path_illuminaga, dataType = "miRNASeq")
BRCA.miRNASeq.illuminahiseq <- readTCGA(path_illuminahiseq, dataType = "miRNASeq")

BRCA.miRNASeq.illuminaga <- cbind(machine = "Illumina Genome Analyzer",
                                  BRCA.miRNASeq.illuminaga)
BRCA.miRNASeq.illuminahiseq <- cbind(machine = "Illumina HiSeq 2000",
                                     BRCA.miRNASeq.illuminahiseq)

BRCA.miRNASeq <- rbind(BRCA.miRNASeq.illuminaga, BRCA.miRNASeq.illuminahiseq)

##############
##### isoforms
##############

# Directory in which untarred data will be stored
dir.create('data8')

# Download ACC isoforms data and store it in data8 folder
cancerType = "ACC"
downloadTCGA(cancerTypes = cancerType,
dataSet = paste0("Merge_rnaseqv2__illuminahiseq_rnaseqv2__unc",
                 "_edu__Level_3__RSEM_isoforms_normalized__data.Level_3"),
             destDir = "data8")

# Shorten path of subdirectory with ACC isoforms data
list.files(path = "data8", full.names = TRUE) %>%
    file.rename(from = ., to = file.path("data8",paste0(cancerType, ".isoforms")))

# Remove manifest.txt file
list.files(path = "data6", full.names = TRUE,
           recursive = TRUE, pattern = "MANIFEST") %>%
           file.remove()

# Read ACC isoforms data
path <- list.files(path = "data8", full.names = TRUE, recursive = TRUE)
ACC.isoforms <- readTCGA(path, dataType = "isoforms")


## End(Not run)
```

`survivalTCGA` *Extract Survival Information from Datasets Included in RTCGA.clinical and RTCGA.clinical.20160128 Packages*

Description

Extracts survival information from clicnial datasets from TCGA project.

Usage

```
survivalTCGA(
```

```
  ...,
  extract.cols = NULL,
  extract.names = FALSE,
  barcode.name = "patient.bcr_patient_barcode",
  event.name = "patient.vital_status",
  days.to.followup.name = "patient.days_to_last_followup",
  days.to.death.name = "patient.days_to_death"
)
```

Arguments

`...`	A data.frame or data.frames from TCGA study containing clinical informations. See clinical.
`extract.cols`	A character specifing the names of extra columns to be extracted with survival information.
`extract.names`	Logical, whether to extract names of passed data.frames in `...`.
`barcode.name`	A character with the name of `bcr_patient_barcode` which differs between TCGA releases. By default is the name from the newest release date `tail(checkTCGA('Dates'),1)`.
`event.name`	A character with the name of `patient.vital_status` which differs between TCGA releases. By default is the name from the newest release date `tail(checkTCGA('Dates'),1)`.
`days.to.followup.name`	A character with the name of `patient.days_to_last_followup` which differs between TCGA releases. By default is the name from the newest release date `tail(checkTCGA('Dates'),1)`.
`days.to.death.name`	A character with the name of `patient.days_to_death` which differs between TCGA releases. By default is the name from the newest release date `tail(checkTCGA('Dates'),1)`.

Value

A data.frame containing information about times and censoring for specific `bcr_patient_barcode`. The name passed in `barcode.name` is changed to `bcr_patient_barcode`.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Note

Input data.frames should contain columns `patient.bcr_patient_barcode`, `patient.vital_status`, `patient.days_to_last_followup`, `patient.days_to_death` or theyir previous equivalents. It is recommended to use datasets from clinical.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Visualizations.html.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), theme_RTCGA()

Examples

```
## Extracting Survival Data
library(RTCGA.clinical)
survivalTCGA(BRCA.clinical, OV.clinical, extract.cols = "admin.disease_code") -> BRCAOV.survInfo

## Kaplan-Meier Survival Curves
kmTCGA(BRCAOV.survInfo, explanatory.names = "admin.disease_code",  pval = TRUE)

kmTCGA(BRCAOV.survInfo, explanatory.names = "admin.disease_code", main = "",
       xlim = c(0,4000))

# first munge data, then extract survival info
library(dplyr)
BRCA.clinical %>%
    filter(patient.drugs.drug.therapy_types.therapy_type %in%
               c("chemotherapy", "hormone therapy")) %>%
    rename(therapy = patient.drugs.drug.therapy_types.therapy_type) %>%
    survivalTCGA(extract.cols = c("therapy"))  -> BRCA.survInfo.chemo

# first extract survival info, then munge data
    survivalTCGA(BRCA.clinical,
                 extract.cols = c("patient.drugs.drug.therapy_types.therapy_type"))  %>%
    filter(patient.drugs.drug.therapy_types.therapy_type %in%
               c("chemotherapy", "hormone therapy")) %>%
   rename(therapy = patient.drugs.drug.therapy_types.therapy_type) -> BRCA.survInfo.chemo


kmTCGA(BRCA.survInfo.chemo, explanatory.names = "therapy",
       xlim = c(0, 3000), conf.int = FALSE)
```

---

theme_RTCGA *RTCGA Theme for ggplot2*

Description

Additional **RTCGA** theme for ggtheme, based on theme_pander.

Usage

```
theme_RTCGA(base_size = 11, base_family = "", ...)
```

Arguments

base_size	base font size
base_family	base font family
...	Further arguments passed to theme_pander.

Issues

If you have any problems, issues or think that something is missing or is not clear please post an issue on https://github.com/RTCGA/RTCGA/issues.

Author(s)

Marcin Kosinski, <m.p.kosinski@gmail.com>

See Also

RTCGA website http://rtcga.github.io/RTCGA/articles/Visualizations.html.

Other RTCGA: RTCGA-package, boxplotTCGA(), checkTCGA(), convertTCGA(), datasetsTCGA, downloadTCGA(), expressionsTCGA(), heatmapTCGA(), infoTCGA(), installTCGA(), kmTCGA(), mutationsTCGA(), pcaTCGA(), readTCGA(), survivalTCGA()

Examples

```
library(RTCGA.clinical)
survivalTCGA(BRCA.clinical, OV.clinical, extract.cols = "admin.disease_code") -> BRCAOV.survInfo
kmTCGA(BRCAOV.survInfo, explanatory.names = "admin.disease_code",
    xlim = c(0,4000))
```

Index