Package 'ngstk'

November 22, 2018

Type Package

Title Next-Generation Sequencing (NGS) Data Analysis Toolkit

Version 0.2.3

Description Can be used to facilitate the analysis of NGS data, such as visualization, conversion of data format for WEB service input and other purpose.

Depends R (>= 3.3.0)

URL https://github.com/JhuangLab/ngstk

BugReports https://github.com/JhuangLab/ngstk/issues

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports stringr (>= 1.2.0), stringi, configr (>= 0.3.1), data.table, future, optparse, parallel

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, testthat, prettydoc

VignetteBuilder knitr

NeedsCompilation no

Author Jianfeng Li [aut, cre] (<https://orcid.org/0000-0003-2349-208X>)

Maintainer Jianfeng Li <lee_jianfeng@sjtu.edu.cn>

Repository CRAN

Date/Publication 2018-11-22 16:40:03 UTC

R topics documented:

`batch_file` *Process the input file a batch of one batch*

Description

Process the input file a batch of one batch

Usage

```
batch_file(filename = "", batch_lines = 1e+07, handler = NULL,
  param_names = c("x", "i"), extra_fread_params = list(sep = "\n",
  header = FALSE, return_1L = TRUE), extra_params = list(),
  start_index = 1)
```

Arguments

filename	Filename need to process
batch_lines	Batch lines to process the data, default 10000000
handler	The function to process the data
param_names	Hander function required parameter names
extra_fread_params	Extra fread parameters in read data step, default is list(sep = '\n', header = TRUE, return_1L = TRUE), return_1L to get x[[1L]]
extra_params	Extra paramemters pass to handler function
start_index	default is 1, control the skip rows, n = (i-1) * batch_lines

Examples

```
dat <- data.frame(a=1:100, b=1:100)
filename <- tempfile()
write.table(dat, filename, sep = '\t', row.names = FALSE, quote = FALSE)
handler_fun <- function(x, i = 1) {
  return(x[i])
}
batch_file(filename, 10, handler_fun)
```

---

format_filenames *Function to format filenames that can be used to unify the filenames style for more easily download or use*

Description

Function to format filenames that can be used to unify the filenames style for more easily download or use

Usage

```
format_filenames(input_files = NULL, files_dir = NULL,
  pattern = ".*.txt", do.rename = FALSE, suffix = "", prefix = "",
  replace = list(old = c("-", "__"), new = c("_", "_")),
  toupper = FALSE, tolower = FALSE)
```

Arguments

input_files	Basename of files that need to be format, default is NULL and use the regular expression pattern to select files
files_dir	Directory name of input files
pattern	Use regular expression to select files in files_dir
do.rename	If set TRUE, it will do rename step

suffix	Profix of filenames added in those without the same suffix
prefix	Prefix of filenames added in those without the same prefix
replace	Use str_replace to replace all old to new separately
toupper	Filenames toupper, default is FALSE
tolower	Filenames tolower, default is FALSE

Examples

```
files_dir <- system.file('extdata', 'demo/format', package = 'ngstk')
pattern <- '*.txt'
x <- format_filenames(files_dir = files_dir, pattern = pattern, suffix = 'hg38_')
```

---

format_pp_meta_age *To format ProteinPaint input meta data of age*

Description

To format ProteinPaint input meta data of age

Usage

```
format_pp_meta_age(raw_meta, outfn = NULL, age_group = "Age",
  adult_value = "Adult", child_value = "Pediatric",
  adult_color = "#c20b01", child_color = "#196abd", ...)
```

Arguments

raw_meta	A data.frame contain cols of 'sample', 'term', 'group', 'value', 'color' and 'legendorder'
outfn	Output file, default is NULL and not output to file
age_group	Name of age group, default is 'Age'
adult_value	Value of adult, default is 'Adult'
child_value	Value of child, default is 'Pediatric'
adult_color	Color of adult, default is '#c20b01'
child_color	Color of child, default is '#196abd'
...	Parameters pass to set_colors

Value

A data frame

Examples

```
meta_template <- system.file('extdata',
'demo/proteinpaint/heatmap_meta_template.txt',
package = 'ngstk')
raw_meta <- read.table(meta_template, sep = '\t', header = TRUE)
term <- group <- 'Age'
raw_meta$term <- term
raw_meta$group <- group
raw_meta$value <- c(rep(c('Adult', 'Pediatric'), 3),  'Male')
meta_age <- format_pp_meta_age(raw_meta)
```

`format_pp_meta_fusions` *To format ProteinPaint input meta data of gender*

Description

To format ProteinPaint input meta data of gender

Usage

```
format_pp_meta_fusions(raw_meta, outfn = NULL, fusions_color = NULL,
  ...)
```

Arguments

`raw_meta`	A data.frame contain cols of 'sample', 'term', 'group', 'value', 'color' and 'legendorder'
`outfn`	Output file, default is NULL and not output to file
`fusions_color`	In one group, different fusions show different colors, default is NULL and use the setted theme color
`...`	Parameters pass to `set_colors`

Value

A data frame

Examples

```
meta_template <- system.file('extdata',
'demo/proteinpaint/heatmap_meta_template.txt', package = 'ngstk')
raw_meta <- read.table(meta_template, sep = '\t', header = TRUE)
meta_test_1 <- raw_meta
term <- group <- 'Fusions'
meta_test_1$term <- term
```

```
meta_test_1$group <- group
meta_test_1$value <- c(rep(c('ZNF384-Fusions', 'MEF2D-Fusions'),
3), 'TCF3-PBX1')
meta_fusions <- format_pp_meta_fusions(meta_test_1)
meta_test_2 <- raw_meta
term <- group <- c(rep(c('MEF2D-Fusions', 'ZNF384-Fusions'),
3), 'DUX4-Fusions')
meta_test_2$term <- term
meta_test_2$group <- group
meta_test_2$value <- c('MEF2D-PA', 'EP300-ZNF384',
'MEF2D-PB', 'ABC-ZNF384', 'MEF2D-PB', 'ABD-ZNF384',
                       'DUX4-IGH')
meta_fusions <- format_pp_meta_fusions(meta_test_2)
```

`format_pp_meta_gender` *To format ProteinPaint input meta data of gender*

Description

To format ProteinPaint input meta data of gender

Usage

```
format_pp_meta_gender(raw_meta, outfn = NULL, gender_group = "Gender",
  male_value = "Male", female_value = "Female",
  male_color = "#c20b01", female_color = "#196abd", ...)
```

Arguments

`raw_meta`	A data.frame contain cols of 'sample', 'term', 'group', 'value', 'color' and 'legendorder'
`outfn`	Output file, default is NULL and not output to file
`gender_group`	Name of age group, default is 'Gender'
`male_value`	Value of male, default is 'Male'
`female_value`	Value of female, default is 'Female'
`male_color`	Color of male, default is '#c20b01'
`female_color`	Color of female, default is '#196abd'
`...`	Parameters pass to `set_colors`

Value

A data frame

Examples

```
meta_template <- system.file('extdata',
'demo/proteinpaint/heatmap_meta_template.txt', package = 'ngstk')
raw_meta <- read.table(meta_template, sep = '\t', header = TRUE)
term <- group <- 'Gender'
raw_meta$term <- term
raw_meta$group <- group
raw_meta$value <- c(rep(c('Male', 'Female'), 3), 'Male')
meta_gender <- format_pp_meta_gender(raw_meta)
```

`fusions2oncoprinter` *Function to convert fusion data to cbioportal Oncoprinter format.*

Description

Function to convert fusion data to cbioportal Oncoprinter format.

Usage

```
fusions2oncoprinter(input_data, input_type = "fusioncatcher",
  config_file = system.file("extdata", "config/cbioportal.toml", package
  = "ngstk"), config_list = NULL,
  handler_confg_file = system.file("extdata", "config/handler.toml",
  package = "ngstk"), mhandler_confg_file = system.file("extdata",
  "config/mhandler.toml", package = "ngstk"), handler_funs = NULL,
  mhandler_funs = NULL, handler_extra_params = NULL,
  mhandler_extra_params = NULL, outfn = NULL)
```

Arguments

`input_data`	A mutation data.frame need to be converted to ProteinPaint input.
`input_type`	Point the input data format (iseq or others)
`config_file`	ngstk ProteinPaint configuration file path, default is system.file('extdata', 'config/proteinpaint.toml', package = 'ngstk')
`config_list`	ngstk ProteinPaint configuration, default is NULL and read from config_file
`handler_confg_file`	ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')
`mhandler_confg_file`	ngstk handler configuration file path, default is system.file('extdata', 'config/mhandler.toml', package = 'ngstk')
`handler_funs`	handler function for single colnum, default is NULL and get value from config_file
`mhandler_funs`	handler function for mulitple colnums, #' default is NULL and get value from config_file

handler_extra_params
: Extra parameters pass to handler

mhandler_extra_params
: Extra parameters pass to mhandler

outfn
: Default is NULL and not output the result to file

Value

A data frame

Examples

```
demo_file <- system.file('extdata',
'demo/proteinpaint/muts2pp_iseq.txt', package = 'ngstk')
input_data <- read.table(demo_file, sep = '\t', header = TRUE, stringsAsFactors = FALSE)
disease <- 'T-ALL'
input_data <- data.frame(input_data, disease)
input_data$disease <- as.character(input_data$disease)
fusions2oncoprinter(input_data, input_type = 'fusioncatcher')
```

---

fusions2pp *Function to convert fusion data to ProteinPaint format.*

Description

Function to convert fusion data to ProteinPaint format.

Usage

```
fusions2pp(input_data, input_type = "fusioncatcher",
  config_file = system.file("extdata", "config/proteinpaint.toml",
  package = "ngstk"), config_list = NULL,
  handler_confg_file = system.file("extdata", "config/handler.toml",
  package = "ngstk"), mhandler_confg_file = system.file("extdata",
  "config/mhandler.toml", package = "ngstk"), handler_funs = NULL,
  mhandler_funs = NULL, handler_extra_params = NULL,
  mhandler_extra_params = NULL, outfn = NULL)
```

Arguments

input_data
: A gene fusions data.frame need to be converted to ProteinPaint input.

input_type
: Point the input data format (fusioncatcher or others)

config_file
: ngstk ProteinPaint configuration file path, default is system.file('extdata', 'config/proteinpaint.toml', package = 'ngstk')

config_list
: ngstk ProteinPaint configuration, default is NULL and read from config_file

handler_confg_file
: ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')

mhandler_confg_file
: ngstk handler configuration file path, default is system.file('extdata', 'config/mhandler.toml', package = 'ngstk')

handler_funs
: handler function for single colnum, default is NULL and get value from config_file

mhandler_funs
: handler function for mulitple colnums, #' default is NULL and get value from config_file

handler_extra_params
: Extra parameters pass to handler

mhandler_extra_params
: Extra parameters pass to mhandler

outfn
: Default is NULL and not output the result to file

Value

A data frame

Examples

```
demo_file <- system.file('extdata',
'demo/proteinpaint/fusions2pp_fusioncatcher.txt', package = 'ngstk')
input_data <- read.table(demo_file, sep = '\t', header = TRUE, stringsAsFactors = FALSE)
disease <- 'B-ALL'
sampletype <- 'diagnose'
input_data <- data.frame(input_data, disease, sampletype)
input_data$disease <- as.character(input_data$disease)
handler_data <- fusions2pp(input_data, input_type = 'fusioncatcher')
```

---

fusions2pp_meta *Function to convert fusion data to ProteinPaint heatmap meta rows format.*

Description

Function to convert fusion data to ProteinPaint heatmap meta rows format.

Usage

```
fusions2pp_meta(input_data, input_type = "fusioncatcher",
  config_file = system.file("extdata", "config/proteinpaint.toml",
  package = "ngstk"), config_list = NULL,
  handler_confg_file = system.file("extdata", "config/handler.toml",
  package = "ngstk"), mhandler_confg_file = system.file("extdata",
  "config/mhandler.toml", package = "ngstk"), handler_funs = NULL,
  mhandler_funs = NULL, handler_extra_params = NULL,
  mhandler_extra_params = NULL, outfn = NULL)
```

Arguments

input_data	A gene fusions data.frame need to be converted to ProteinPaint input.
input_type	Point the input data format (fusioncatcher or others)
config_file	ngstk ProteinPaint configuration file path, default is system.file('extdata', 'config/proteinpaint.toml', package = 'ngstk')
config_list	ngstk ProteinPaint configuration, default is NULL and read from config_file
handler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')
mhandler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/mhandler.toml', package = 'ngstk')
handler_funs	handler function for single colnum, default is NULL and get value from config_file
mhandler_funs	handler function for mulitple colnums, #' default is NULL and get value from config_file
handler_extra_params	Extra parameters pass to handler
mhandler_extra_params	Extra parameters pass to mhandler
outfn	Default is NULL and not output the result to file

Value

A data frame

Examples

```
demo_file <- system.file('extdata',
'demo/proteinpaint/fusions2pp_fusioncatcher.txt', package = 'ngstk')
input_data <- read.table(demo_file, sep = '\t', header = TRUE, stringsAsFactors = FALSE)
disease <- 'B-ALL'
sampletype <- 'diagnose'
input_data <- data.frame(input_data, disease, sampletype)
input_data$disease <- as.character(input_data$disease)
#handler_data <- fusions2pp_meta(input_data, input_type = 'fusioncatcher')
```

---

fusions_filter *Fusions handler_data filter that can be used to prepare the input data for downstream analysis*

Description

Fusions handler_data filter that can be used to prepare the input data for downstream analysis

Usage

```
fusions_filter(input_data, input_type = "common",
  config_file = system.file("extdata", "config/filter.toml", package =
  "ngstk"), config_list = NULL,
  handler_confg_file = system.file("extdata", "config/handler.toml",
  package = "ngstk"), mhandler_confg_file = system.file("extdata",
  "config/mhandler.toml", package = "ngstk"), handler_funs = NULL,
  mhandler_funs = NULL, handler_extra_params = NULL,
  mhandler_extra_params = NULL, outfn = NULL)
```

Arguments

input_data	A data frame containing the fusions cols (gene5, gene3, fusion_type)
input_type	Fusion filter type
config_file	ngstk filter configuration file path, default is system.file('extdata', 'config/filter.toml', package = 'ngstk')
config_list	ngstk filter configuration, default is NULL and read from config_file
handler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')
mhandler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/mhandler.toml', package = 'ngstk')
handler_funs	handler function for single colnum, default is NULL and get value from config_file
mhandler_funs	handler function for mulitple colnums, #' default is NULL and get value from config_file
handler_extra_params	Extra parameters pass to handler
mhandler_extra_params	Extra parameters pass to mhandler system.file('extdata', 'config/filter.toml', package = 'ngstk')
outfn	Default is NULL and not output the result to file

Value

A data frame

Examples

```
demo_file <- system.file('extdata',
'demo/proteinpaint/fusions2pp_fusioncatcher.txt', package = 'ngstk')
input_data <- read.table(demo_file, sep = '\t', header = TRUE, stringsAsFactors = FALSE)
result <- fusions_filter(input_data)
```

get_files_ctime *Function to check file create time and according the requirement to return check value*

Description

Function to check file create time and according the requirement to return check value

Usage

```
get_files_ctime(input_files = NULL, files_dir = NULL,
  pattern = ".*.txt", return_ctime = TRUE, return_check = TRUE,
  check_time_fun = function(files_ctime) {     all(files_ctime ==
  files_ctime[1]) })
```

Arguments

input_files	Basename of files that need to be check, default is NULL and use the regular expression pattern to select files
files_dir	Directory name of input files
pattern	Use regular expression to select files in files_dir
return_ctime	Logical indicating wheather to return files modification times
return_check	Logical indicating wheather to return file times check status
check_time_fun	Function to check files time, default is all equal

Examples

```
file_a <- tempfile()
file_b <- tempfile()
file.create(c(file_a, file_b))
get_files_ctime(input_files = c(file_a, file_b), return_ctime = TRUE)
```

get_files_mtime *Function to check file last change time and according the requirement to return check value*

Description

Function to check file last change time and according the requirement to return check value

Usage

```
get_files_mtime(input_files = NULL, files_dir = NULL,
  pattern = ".*.txt", return_mtime = TRUE, return_check = TRUE,
  check_time_fun = function(files_mtime) {     all(files_mtime ==
  files_mtime[1]) })
```

Arguments

input_files	Basename of files that need to be check, default is NULL and use the regular expression pattern to select files
files_dir	Directory name of input files
pattern	Use regular expression to select files in files_dir
return_mtime	Logical indicating wheather to return files modification times
return_check	Logical indicating wheather to return file times check status
check_time_fun	Function to check files time, default is all equal

Examples

```
file_a <- tempfile()
file_b <- tempfile()
file.create(c(file_a, file_b))
get_files_mtime(input_files = c(file_a, file_b), return_mtime = TRUE)
```

---

get_pp_samplegroup *Function to get samplegroup file that can be pass to Proteinpaint*

Description

Function to get samplegroup file that can be pass to Proteinpaint

Usage

```
get_pp_samplegroup(samples, group, outfn = NULL)
```

Arguments

samples	A vector indicating all samples
group	A vector indicating the group information
outfn	Default is NULL and not output the result to file

Value

A data frame

Examples

```
samples <- sprintf('A%s', 1:7)
group <- 'B-ALL'
samplegroup <- get_pp_samplegroup(samples, group)
outfn <- tempfile()
samplegroup <- get_pp_samplegroup(samples, group, outfn)
```

get_split_seqs *Function to calculate the split regions by sections and total numbers*

Description

Function to calculate the split regions by sections and total numbers

Usage

```
get_split_seqs(total_num, sections)
```

Arguments

total_num	Total numbers need to be divided into n sections
sections	Split section number (colnum)

Examples

```
total_num <- 1000
sections <- 3
get_split_seqs(total_num, sections)
```

make_option *Functions to enable our OptionParser to recognize specific command line options (optparse).*

Description

Functions to enable our OptionParser to recognize specific command line options (optparse).

Usage

```
make_option(...)
```

Arguments

...	Parameters pass to make_option

Examples

```
# example from vignette
option_list <- list(
  make_option(c('-v', '--verbose'), action='store_true', default=TRUE,
      help='Print extra output [default]'),
  make_option(c('-q', '--quietly'), action='store_false',
      dest='verbose', help='Print little output'),
  make_option(c('-c', '--count'), type='integer', default=5,
      help='Number of random normals to generate [default %default]',
      metavar='number'),
  make_option('--generator', default='rnorm',
      help = 'Function to generate random deviates [default \'%default\']'),
  make_option('--mean', default=0,
      help='Mean if generator == \'rnorm\' [default %default]'),
  make_option('--sd', default=1, metavar='standard deviation',
      help='Standard deviation if generator == \'rnorm\' [default %default]')
 )
```

---

merge_table_files *Util function to merge multiple table files.*

Description

Util function to merge multiple table files.

Usage

```
merge_table_files(input_files = NULL, files_dir = NULL,
  pattern = ".*.txt", outfn = NULL, add.filename = TRUE,
  read_fun = "read.table", read_params_file = "file",
  read_params = list(sep = "\t", header = TRUE),
  write_fun = "write.table", write_params_x = "x",
  write_params_file = "file", write_params = list(sep = "\t",
  row.names = FALSE), op = list(stringsAsFactors = FALSE))
```

Arguments

input_files	Basename of files that need to be merged, default is NULL and use the regular expression pattern to select files
files_dir	Directory name of input files
pattern	Use regular expression to select files in files_dir
outfn	Output file path, default is NULL and return the data.frame object
add.filename	Wheather to add the merged filename, default is TRUE
read_fun	Function to read data, default is read.table
read_params_file	Parameter name of input file in read_fun

read_params	Other parameters pass to read_fun
write_fun	Function to read data, default is read.table
write_params_x	Parameter name of output object in read.fun
write_params_file	Parameter name of input file in read.fun
write_params	Other parameters pass to write_fun
op,	Extra option that only take effect in merge process

Examples

```
a <- data.frame(col1=1:6, col2=2:7)
b <- data.frame(col1=6:11, col2=1:6)
file_a <- paste0(tempfile(), '_abcd')
file_b <- paste0(tempfile(), '_abcd')
write.table(a, file_a, sep = '\t', row.names = FALSE)
write.table(b, file_b, sep = '\t', row.names = FALSE)
input_files <- c(file_a, file_b)
x1 <- merge_table_files(input_files = input_files)
x2 <- merge_table_files(files_dir = tempdir(), pattern = '.*_abcd$')
outfn = tempfile()
```

---

muts2mutation_mapper *Function to convert mutation data to cbioportal MutationMapper format.*

Description

Function to convert mutation data to cbioportal MutationMapper format.

Usage

```
muts2mutation_mapper(input_data, input_type = "iseq",
  config_file = system.file("extdata", "config/cbioportal.toml", package
  = "ngstk"), config_list = NULL,
  handler_confg_file = system.file("extdata", "config/handler.toml",
  package = "ngstk"), mhandler_confg_file = system.file("extdata",
  "config/mhandler.toml", package = "ngstk"), handler_funs = NULL,
  mhandler_funs = NULL, handler_extra_params = NULL,
  mhandler_extra_params = NULL, outfn = NULL)
```

Arguments

input_data	A mutation data.frame need to be converted to ProteinPaint input.
input_type	Point the input data format (iseq or others)
config_file	ngstk ProteinPaint configuration file path, default is system.file('extdata', 'config/proteinpaint.toml', package = 'ngstk')

config_list ngstk ProteinPaint configuration, default is NULL and read from config_file

handler_confg_file
: ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')

mhandler_confg_file
: ngstk handler configuration file path, default is system.file('extdata', 'config/mhandler.toml', package = 'ngstk')

handler_funs handler function for single colnum, default is NULL and get value from config_file

mhandler_funs handler function for mulitple colnums, #' default is NULL and get value from config_file

handler_extra_params
: Extra parameters pass to handler

mhandler_extra_params
: Extra parameters pass to mhandler

outfn Default is NULL and not output the result to file

Value

A data frame

Examples

```
demo_file <- system.file('extdata',
'demo/proteinpaint/muts2pp_iseq.txt', package = 'ngstk')
input_data <- read.table(demo_file, sep = '\t', header = TRUE, stringsAsFactors = FALSE)
disease <- 'T-ALL'
input_data <- data.frame(input_data, disease)
input_data$disease <- as.character(input_data$disease)
muts2mutation_mapper(input_data, input_type = 'iseq')
```

---

muts2oncoprinter *Function to convert mutation data to cbioportal Oncoprinter format.*

Description

Function to convert mutation data to cbioportal Oncoprinter format.

Usage

```
muts2oncoprinter(input_data, input_type = "iseq",
  config_file = system.file("extdata", "config/cbioportal.toml", package
  = "ngstk"), config_list = NULL,
  handler_confg_file = system.file("extdata", "config/handler.toml",
  package = "ngstk"), mhandler_confg_file = system.file("extdata",
  "config/mhandler.toml", package = "ngstk"), handler_funs = NULL,
  mhandler_funs = NULL, handler_extra_params = NULL,
  mhandler_extra_params = NULL, outfn = NULL)
```

Arguments

input_data	A mutation data.frame need to be converted to ProteinPaint input.
input_type	Point the input data format (iseq or others)
config_file	ngstk ProteinPaint configuration file path, default is system.file('extdata', 'config/proteinpaint.toml', package = 'ngstk')
config_list	ngstk ProteinPaint configuration, default is NULL and read from config_file
handler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')
mhandler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/mhandler.toml', package = 'ngstk')
handler_funs	handler function for single colnum, default is NULL and get value from config_file
mhandler_funs	handler function for mulitple colnums, #' default is NULL and get value from config_file
handler_extra_params	Extra parameters pass to handler
mhandler_extra_params	Extra parameters pass to mhandler
outfn	Default is NULL and not output the result to file

Value

A data frame

Examples

```
demo_file <- system.file('extdata',
'demo/proteinpaint/muts2pp_iseq.txt', package = 'ngstk')
input_data <- read.table(demo_file, sep = '\t', header = TRUE, stringsAsFactors = FALSE)
disease <- 'T-ALL'
input_data <- data.frame(input_data, disease)
input_data$disease <- as.character(input_data$disease)
muts2oncoprinter(input_data, input_type = 'iseq')
```

---

muts2pp *Function to convert mutation data to ProteinPaint format.*

Description

Function to convert mutation data to ProteinPaint format.

Usage

```
muts2pp(input_data, input_type = "iseq",
  config_file = system.file("extdata", "config/proteinpaint.toml",
  package = "ngstk"), config_list = NULL,
  handler_confg_file = system.file("extdata", "config/handler.toml",
  package = "ngstk"), mhandler_confg_file = system.file("extdata",
  "config/mhandler.toml", package = "ngstk"), handler_funs = NULL,
  mhandler_funs = NULL, handler_extra_params = NULL,
  mhandler_extra_params = NULL, outfn = NULL)
```

Arguments

input_data	A mutation data.frame need to be converted to ProteinPaint input.
input_type	Point the input data format (iseq or others)
config_file	ngstk ProteinPaint configuration file path, default is system.file('extdata', 'config/proteinpaint.toml', package = 'ngstk')
config_list	ngstk ProteinPaint configuration, default is NULL and read from config_file
handler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')
mhandler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/mhandler.toml', package = 'ngstk')
handler_funs	handler function for single colnum, default is NULL and get value from config_file
mhandler_funs	handler function for mulitple colnums, #' default is NULL and get value from config_file
handler_extra_params	Extra parameters pass to handler
mhandler_extra_params	Extra parameters pass to mhandler
outfn	Default is NULL and not output the result to file

Value

A data frame

Examples

```
demo_file <- system.file('extdata',
'demo/proteinpaint/muts2pp_iseq.txt', package = 'ngstk')
input_data <- read.table(demo_file, sep = '\t', header = TRUE, stringsAsFactors = FALSE)
disease <- 'T-ALL'
input_data <- data.frame(input_data, disease)
input_data$disease <- as.character(input_data$disease)
muts2pp(input_data, input_type = 'iseq')
```

`ngstk` *ngstk can be used to facilitate the analysis of NGS data, such as visualization, conversion of data format for WEB service input and other purpose.*

Description

ngstk can be used to facilitate the analysis of NGS data, such as visualization, conversion of data format for WEB service input and other purpose.

Author(s)

Li Jianfeng lee_jianfeng@sjtu.edu.cn

See Also

Useful links:

https://github.com/JhuangLab/ngstk

Report bugs at https://github.com/JhuangLab/ngstk/issues

`opt_parser` *A function to create an instance of a parser object with subcommands.*

Description

Modifed from OptionParser

Usage

```
opt_parser(subcmds_list = list(), subcmds_sections = list(), ...)
```

Arguments

`subcmds_list`	A list setting the subcommands name and short description (e.g. list(subcmd1='Method1 to plot boxplot'))
`subcmds_sections`	A list setting the sections of subcommands (e.g. list(subcmd_group_1=c('subcmd1', 'subcmd2'), subcmd_group_2=c('subcmd2')))
`...`	Parameters pass to OptionParser

Examples

```
option_list <- list(
  make_option(c('-l', '--list-all-subcmds'), action = 'store_true',
              default = FALSE, help = 'Print all supported subcmds of ngsjs.')
)
subcmds_list <- list(subcmd1 = 'Use method 1 to plot boxplot',
                     subcmd2 = 'Use method 2 to plot boxplot')
description <- 'Method to plot boxplot'
usage <- 'usage: %prog [options] [params]'
opt_parser_obj <- opt_parser(subcmds_list = subcmds_list,
                             option_list = option_list,
                             description = description,
                             usage = usage)
```

`parse_args` *Parse command line options (optparse).*

Description

Parse command line options (optparse).

Usage

```
parse_args(...)
```

Arguments

`...`	Parameters pass to `parse_args`

Examples

```
# example from vignette
option_list <- list(
  make_option(c('-v', '--verbose'), action='store_true', default=TRUE,
      help='Print extra output [default]'),
  make_option(c('-q', '--quietly'), action='store_false',
      dest='verbose', help='Print little output'),
  make_option(c('-c', '--count'), type='integer', default=5,
      help='Number of random normals to generate [default %default]',
      metavar='number'),
  make_option('--generator', default='rnorm',
      help = 'Function to generate random deviates [default \'%default\']'),
  make_option('--mean', default=0,
      help='Mean if generator == \'rnorm\' [default %default]'),
  make_option('--sd', default=1, metavar='standard deviation',
      help='Standard deviation if generator == \'rnorm\' [default %default]')
  )
parse_args(opt_parser(option_list = option_list), args = c('--sd=3', '--quietly'))
```

parse_args2 *Parse command line options.*

Description

Parse command line options.

Usage

```
parse_args2(...)
```

Arguments

...	Parameters pass to parse_args2

Examples

```
#example from vignette using positional arguments
option_list2 <- list(
  make_option(c('-n', '--add-numbers'), action='store_true', default=FALSE,
      help='Print line number at the beginning of each line [default]')
  )
parser <- opt_parser(usage = '%prog [options] file', option_list=option_list2)

parse_args(parser, args = c('--add-numbers', 'example.txt'), positional_arguments = TRUE)

parse_args(parser, args = c('--add-numbers', 'example.txt'), positional_arguments = TRUE,
        convert_hyphens_to_underscores = TRUE)
parse_args2(parser, args = c('--add-numbers', 'example.txt'))
```

par_download *Function to download multiple file at the same time.*

Description

Function to download multiple file at the same time.

Usage

```
par_download(urls, destfiles = NULL, logdir = sprintf("%s/logs",
  tempdir()), ids_func = function(urls) {     stri_rand_strings(n =
  length(urls), length = 20) }, logfn_func = function(ids, logdir) {
  normalizePath(sprintf("%s/%s.log", logdir, ids), mustWork = FALSE) },
  parallel_method = "plan(multiprocess)", ...)
```

Arguments

urls	A character string naming the URLs of a resource to be downloaded.
destfiles	Default to use the basename of urls
logdir	Directory to store the download logs [tempdir()]
ids_func	Function to define the returned ids
logfn_func	Function to define the log files
parallel_method	Method for future parallel [plan(multiprocess)]
...	Other params pass to download.file

Value

A list

Examples

```
urls <- c(paste0('https://raw.githubusercontent.com/',
'Miachol/ftp/master/files/images/bioinstaller/maftools3.png'),
paste0('https://raw.githubusercontent.com/',
'Miachol/ftp/master/files/images/bioinstaller/maftools4.png'))
## Not run:
par_download(urls, sprintf('%s/%s', tempdir(), basename(urls)))

## End(Not run)
```

---

print_help *Printing an usage message from an OptionParser object*

Description

Modifed from print_help

Usage

```
print_help(object, help_order = c("description", "usage", "options",
  "subcmds", "epilogue"))
```

Arguments

object	A opt_parser instance.
help_order	The order to print the help message ['description', 'usage', 'options', 'subcmds', 'epilogue']

Examples

```
option_list <- list(
  make_option(c('-l', '--list-all-subcmds'), action = 'store_true',
  default = FALSE, help = 'Print all supported subcmds of ngsjs.')
)
subcmds_list <- list(subcmd1 = 'Use method 1 to plot boxplot',
                     subcmd2 = 'Use method 2 to plot boxplot')
description <- 'Method to plot boxplot'
usage <- 'usage: %prog [options] [params]'
opt_parser_obj <- opt_parser(subcmds_list = subcmds_list,
                             option_list = option_list,
                             description = description,
                             usage = usage)
print_help(opt_parser_obj)
```

---

rbin *Function to generate executable files for R package*

Description

Function to generate executable files for R package

Usage

```
rbin(pkgs, destdir = "~/.ngstk/bin")
```

Arguments

pkgs	Names f R packages to generate executable files
destdir	Destination directory to store the inst/bin files [~/.ngstk/bin]
...	Params pass to file.rename.

Examples

```
rbin('ngstk', tempdir())
```

---

set_colors *Function to get a series defined theme colors*

Description

Function to get a series defined theme colors

Usage

```
set_colors(theme = NULL, theme_config_file = NULL,
  show_all_themes = FALSE)
```

Arguments

theme	Colors theme, e.g. default, red_blue
theme_config_file	Theme configuration file, default is system.file('extdata', 'config/theme.toml', package = 'ngstk')
show_all_themes	Wheather show all avaliable colors theme, default is FALSE

Value

A character

Examples

```
red_blue <- set_colors('red_blue')
default <- set_colors('default')
colors <- set_colors(show_all_themes = TRUE)
```

---

set_tools *Function to generate tools path object*

Description

Function to generate tools path object

Usage

```
set_tools(config.file = "", config.list = list(), config.vec = c(),
  eval.params = list())
```

Arguments

config.file	Path of tools configuration file (json, ini, yaml and toml be supported)
config.list	List object of tools that all of tools path (exclude those without names).
config.vec	Vector object of tools that all of tools path (exclude those without names).
eval.params	Params pass to configr::eval.config

Value

List object contain the tools path that can be used by other function in ngstk package

Examples

```
config.file <- system.file('extdata', 'demo/tools_config.json', package = 'ngstk')
config.list <- list(gatk = '/path/gatk')
config.vec <- c('/path/samtools')
names(config.vec) <- 'samtools'
tools <- set_tools(config.file, config.list, config.vec,
                   eval.params = list(config = 'tools'))
```

show_handlers *Function to show all avaliabe handler function*

Description

Function to show all avaliabe handler function

Usage

```
show_handlers(handler_lib = "default_handlers", show_all_funs = TRUE,
  show_code = NULL, show_description = FALSE,
  handler_confg_file = system.file("extdata", "config/handler.toml",
  package = "ngstk"))
```

Arguments

handler_lib	handler lib name
show_all_funs	Default is TRUE and to show all functions in the handler_lib
show_code	Default is NULL, select handler you want to see its source code
show_description	Default is FALSE
handler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')

Examples

```
show_handlers(show_description = TRUE)
show_handlers(show_description = FALSE, show_all_funs = FALSE,
            show_code = 'handler_na_replace')
```

show_mhandlers *Function to show all avaliabe mhandler function*

Description

Function to show all avaliabe mhandler function

Usage

```
show_mhandlers(mhandler_lib = "default_mhandlers",
  show_all_funs = TRUE, show_code = NULL, show_description = FALSE,
  mhandler_confg_file = system.file("extdata", "config/mhandler.toml",
  package = "ngstk"))
```

Arguments

mhandler_lib	handler lib name
show_all_funs	Default is TRUE and to show all functions in the handler_lib
show_code	Default is NULL, select handler you want to see its source code
show_description	Default is FALSE
mhandler_confg_file	ngstk handler configuration file path, default is system.file('extdata', 'config/handler.toml', package = 'ngstk')

Examples

```
show_mhandlers(show_description = TRUE)
show_mhandlers(show_description = FALSE, show_all_funs = FALSE,
             show_code = 'mhandler_fusions_anyfull_match')
```

---

split_col_data *Data split function by colum*

Description

Data split function by colum

Usage

```
split_col_data(x, sections = 1)
```

Arguments

x	Data.frame or data.table object that need to be divided n sections by colum
sections	Split section number (colnum)

Examples

```
x1 <- data.frame(col1 = 1:10, col2 = 11:20)
x1.t <- t(x1)
x <- split_col_data(x1.t, sections = 3)
```

split_list *Function to split list*

Description

Function to split list

Usage

```
split_list(x, sections = 1)
```

Arguments

x	List object that need to be divided n sections
sections	Split section number (row)

Examples

```
x <- list(a=1:3, b=2:4, c=3, d=4)
split_list(x, 2)
```

split_row_data *Data split function by row*

Description

Data split function by row

Usage

```
split_row_data(x, sections = 1)
```

Arguments

x	Data.frame or data.table object that need to be divided n sections by row
sections	Split section number (row)

Examples

```
x1 <- data.frame(col1 = 1:10, col2 = 11:20)
x2 <- data.frame(col1 = 1:99, col2 = 101:199)
x <- split_row_data(x1, sections = 1)
x <- split_row_data(x1, sections = 2)
x <- split_row_data(x1, sections = 3)
x <- split_row_data(x1, sections = 4)
x <- split_row_data(x2, sections = 2)
x <- split_row_data(x2, sections = 3)
```

split_row_file *Function to split big file to a series small files (by row)*

Description

Function to split big file to a series small files (by row)

Usage

```
split_row_file(filename, each_file_lines = 100,
  use_system_split = FALSE, system_split_params = "_split",
  write_fun = "write.table", write_params_x = "x",
  write_params_file = "file", write_params = list(sep = "", row.names =
  FALSE, col.names = FALSE, quote = FALSE))
```

Arguments

filename	Filename that need to be split
each_file_lines	Each file row num
use_system_split	Wheather use system split commend
system_split_params	When use_system_split, provide the prefix and other params default is 'split'
write_fun	Function to read data, default is read.table
write_params_x	Parameter name of output object in read.fun
write_params_file	Parameter name of input file in read.fun
write_params	Other parameters pass to write_fun

Examples

```
dat <- data.frame(col1 = 1:1000)
outfn <- tempfile()
write.table(dat, outfn, sep = '\t', quote = FALSE, row.names = FALSE)
split_row_file(outfn)
```

supress_any_message *suppressWarnings(suppressMessages(...))*

Description

suppressWarnings(suppressMessages(...))

Usage

```
supress_any_message(...)
```

Arguments

...	Paramerters pass to downstream functions

Examples

```
supress_any_message(ls())
```

time_stamp *Function to generate time stamp in the files name or directories name.*

Description

Function to generate time stamp in the files name or directories name.

Usage

```
time_stamp(template = c("%Y_%m_%d_%H_%M_%S_",
  "%Y_%m_%d_%H_%M_", "%Y_%m_%d_%H_", "%Y_%m_%d_", "%Y_%m_",
  "%Y_"), extra_flag = c("-", "/", "@"))
```

Arguments

template	The template time format with '_'
extra_flag	Using extra flag to replace template '_' and get extra time stamp ['-', '-']

Value

A list

Examples

```
time_stamp()
```

Index