Package 'normfluodbf'

September 28, 2024

Type Package

Title Cleans and Normalizes FLUOstar DBF and DAT Files from 'Liposome' Flux Assays

Version 2.0.0

Description Cleans and Normalizes FLUOstar DBF and DAT Files obtained from liposome flux assays. Users should verify extended usage of the package on files from other assay types.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Suggests knitr, learnr, rmarkdown

Config/testthat/edition 3

Imports shiny, data.table, foreign (>= 0.8.86), tidyr (>= 1.3.1), tibble, dplyr (>= 1.1.4), emojifont, stats, ggplot2 (>= 3.5.0), badger, stringr, magrittr (>= 2.0.3), assertthat, forcats, gridExtra, pkgsearch, plotly (>= 4.10.4), purrr (>= 1.0.2), rlang (>= 1.1.3), rstudioapi, wesanderson, hexSticker, testthat (>= 3.2.1)

Depends R (>= 4.3.0)

LazyData true

URL https://github.com/AlphaPrime7/normfluodbf, https://alphaprime7.github.io/normfluodbf/

BugReports https://github.com/AlphaPrime7/normfluodbf/issues

VignetteBuilder knitr

NeedsCompilation no

Author Tingwei Adeck [aut, cre, cph] (<https://orcid.org/0009-0009-7450-8863>), Tesla Adeck [cph], Amina Adeck [cph]

Maintainer Tingwei Adeck <awesome.tingwei@outlook.com>

Repository CRAN

Date/Publication 2024-09-27 23:10:03 UTC

Contents

`add_package_namespace` *Add Package Namespace*

Description

Add Package Namespace

Usage

```
add_package_namespace(dir, package)
```

Arguments

dir	dir
package	package

Value

modified file

Examples

```
## Not run: add_package_namespace(dir, package)
```

---

analyze *Analyze*

Description

Analyze

Usage

```
analyze_ready(plate)

## Default S3 method:
analyze_ready(plate)

## S3 method for class '`96well_plate`'
analyze_ready(plate)

## S3 method for class '`384well_plate`'
analyze_ready(plate)

## S3 method for class '`1536well_plate_t1`'
analyze_ready(plate)

## S3 method for class '`1536well_plate_t2`'
analyze_ready(plate)
```

Arguments

plate	plate

Value

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run: analyze_ready(plate)
```

`average_fluorescence_by_row_cycle` *Capitalize*

Description

Capitalize

Usage

```
average_fluorescence_by_row_cycle(plate)
```

Arguments

`plate`	plate

Value

capital letter

Examples

```
## Not run: average_fluorescence_by_row_cycle(plate)
```

capitalize *Capitalize*

Description

Capitalize

Usage

```
capitalize(x)
```

Arguments

x	well

Value

capital letter

Examples

```
## Not run: capitalize('a1')
```

check_broken_packages *Check Broken Packages*

Description

Check Broken Packages

Usage

```
check_broken_packages()
```

Value

broken packages

check_dirt *Check Dirt*

Description

Check Dirt

Usage

```
check_dirt(plate)
```

Arguments

plate	plate

Value

plate

See Also

Other plate_utils: `get_wells_used()`, `plate_data_summary()`, `platename`, `remove_leading_zero()`, `saveloadutils`, `set_plate_version()`, `type()`

check_package_usage *Check package or function Usage*

Description

Check package or function Usage

Usage

```
check_package_usage(directory, package_name)
```

Arguments

directory	directory
package_name	package or string

Value

use location

Examples

```
## Not run: check_package_usage('R','capitalize')
```

childtype *Child Type*

Description

Child Type

Usage

```
child_plate_type(plate, child_type = NULL)

## Default S3 method:
child_plate_type(plate, child_type = NULL)

## S3 method for class 'normfluodbf_plate'
child_plate_type(plate, child_type = NULL)
```

Arguments

plate	plate
child_type	child type

Value

plate

plate

Examples

```
## Not run: child_plate_type()
```

clean_commas *DAT file data frame cleaner.*

Description

The function takes the dirty data frame obtained from reading the FLUOstar DAT file and applies a function called comma_cleaner() to the dirty data frame, which automatically inserts NAs in place of the special characters, and rows with NAs only are removed.

Usage

```
clean_commas(df)
```

Arguments

df	A dirty data frame obtained from the FLUOstar DAT file.

Value

A clean data frame with clean NA values retained.

Author(s)

Tingwei Adeck

Examples

```
## Not run: clean_commas(df)
```

---

clean_odddat_optimus *DAT file wrangler.*

Description

The function takes the dirty data frame obtained from reading the FLUOstar DAT file, applies an original algorithm that inserts NAs in place of the special characters, and then applies a function called comma_cleaner() to the dirty data frame for the removal of commas, and finally, rows with NAs only are removed.

Usage

```
clean_odddat_optimus(df)

clean_even_dat(df)
```

Arguments

df	df

Value

A clean data frame with clean NA values retained.

df

Author(s)

Tingwei Adeck

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
partial_cleaned_dat <- clean_odddat_optimus(dat_df)
## End(Not run)
## Not run: clean_even_dat(df)
```

`comma_cleaner` *Comma Cleaner function.*

Description

This modular function, in the context of this package, is responsible for removing commas from attribute(s) values. Removal of commas facilitates the conversion of attributes into the numeric class.

Usage

```
comma_cleaner(comma_df)
```

Arguments

`comma_df`	A dirty data frame obtained from the FLUOstar DAT file.

Value

A clean data frame with numeric no-comma values for attribute(s).

Author(s)

Tingwei Adeck

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- comma_cleaner(dat_df)
## End(Not run)
```

comment_out_lines *Comment Out*

Description

Comment Out

Usage

```
comment_out_lines(input_file, output_file)
```

Arguments

input_file	file
output_file	file

Value

file

Examples

```
## Not run: comment_out_lines('tests/testthat/test_pipeline.R', 'tests/testthat/test_pipeline.R')
```

dat_1 *dat_1.*

Description

FLUOstar .dat files used for creation of the update and unusable for immediate data analysis.

Usage

```
dat_1
```

Format

An object of class `data.frame` with 320 rows and 12 columns.

dat_2 *dat_2.*

Description

FLUOstar .dat files used for creation of the update and unusable for immediate data analysis.

Usage

```
dat_2
```

Format

An object of class `data.frame` with 320 rows and 12 columns.

dat_3 *dat_3.*

Description

FLUOstar .dat files used for creation of the update and unusable for immediate data analysis. This file is unique because it validates a major bug fix to ensure that users get the right output.

Usage

```
dat_3
```

Format

An object of class `data.frame` with 320 rows and 12 columns.

dat_4 *dat_4.*

Description

FLUOstar .dat files used for creation of the update and unusable for immediate data analysis. This file is unique because it validates a major bug fix to ensure that users get the right output.

Usage

```
dat_4
```

Format

An object of class `data.frame` with 320 rows and 1 columns.

dat_5 *dat_5.*

Description

FLUOstar .dat files used for creation of the update and unusable for immediate data analysis. This file is unique because it validates a major bug fix to ensure that users get the right output.

Usage

```
dat_5
```

Format

An object of class `data.frame` with 105 rows and 1 columns.

dat_6 *dat_6.*

Description

FLUOstar .dat files used for creation of the update and unusable for immediate data analysis. This file is unique because it validates a major bug fix to ensure that users get the right output.

Usage

```
dat_6
```

Format

An object of class `data.frame` with 105 rows and 2 columns.

dat_7 *dat_7.*

Description

FLUOstar .dat files used for creation of the update and unusable for immediate data analysis. This file is unique because it validates a major bug fix to ensure that users get the right output.

Usage

```
dat_7
```

Format

An object of class `data.frame` with 105 rows and 3 columns.

dat_col_names_horizontal *Attribute(s) naming function.*

Description

This function is used to name attribute(s). Attribute(s) names, in this case, are equivalent to the well labels found on the microplate reader. An attribute for a sample loaded into row A - column 1 will be named A1. In short, the function takes a clean data frame and returns attribute names that match the FLUOstar plate layout often presented as an Excel file.

Usage

```
dat_col_names_horizontal(dat = NULL, df, rows_used = NULL, cols_used = NULL)
```

Arguments

dat	A string ("dat_1.dat") if the file is found within the present working directory (pwd) OR a path pointing directly to a ".dat" file.
df	A data frame that requires attribute labels.
rows_used	A character vector indicating the rows or tuples used on the microplate (usually a 96-well microplate). Initialized as NULL.
cols_used	A numeric vector indicating the plate columns or attributes used. Initialized as NULL.

Value

Returns a character or numeric vector of attribute(s) names for the normalized data frame.

Note

This function was designed to avoid the use of stringr. This function is designed to name attributes when the read direction is specified as horizontal.

Author(s)

Tingwei Adeck

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale(nocomma_dat, tnp=3, cycles=40)
n = c('A','B','C')
sample_col_names <- dat_col_names_horizontal(dat=fpath,resampled_scaled, n)
## End(Not run)
```

`dat_col_names_optimus` *Attribute(s) naming function.*

Description

This function is used to name attribute(s). Attribute(s) names, in this case, are equivalent to the well labels found on the microplate reader. An attribute for a sample loaded into row A - column 1 will be named A1. In short, the function takes a clean data frame and returns attribute names that match the FLUOstar plate layout often presented as an Excel file.

Usage

```
dat_col_names_optimus(
  dat = NULL,
  df,
  rows_used = NULL,
  cols_used = NULL,
  user_specific_labels = NULL,
  read_direction = NULL
)
```

Arguments

`dat`	A string ("dat_1.dat") if the file is found within the present working directory (pwd) OR a path pointing directly to a ".dat" file.
`df`	A data frame that requires attribute labels.
`rows_used`	A character vector indicating the rows or tuples used on the microplate (usually a 96-well microplate). Initialized as NULL.
`cols_used`	A numeric vector indicating the plate columns or attributes used. Initialized as NULL.
`user_specific_labels`	A character vector where the user manually enters the used microplate wells based on the FLUOstar plate layout.
`read_direction`	A string input with two choices, “vertical” or “horizontal.” The user indicates “vertical” if the user intends to have a final data frame with samples arranged as sample type triplets (A1, B1, C1, A1, B1, C1) OR “horizontal” if the user intends to have a final data frame with samples

Value

Returns a character or numeric vector of attribute(s) names for the normalized data frame.

Note

Users are advised to input rows used but won't be penalized for not doing so. If the user provides the rows used, then attribute names are generated for the user. The user must check to ensure that the names match the microplate layout. The user can leave the columns used as NULL if the user loaded samples from column 1 and did so in sequence. If the user fails to load in sequence from the first position, then the user must provide a numeric vector of columns used. For instance, where the user skips columns, the user will be prompted to interact with the program in order to ensure the final data frame has the correct attribute names. The user can bypass the rows used and columns used parameters if the user supplies a manually created character vector of the wells used in an experiment. The read direction parameter is used to determine the presentation of the samples in the final data frame.

Author(s)

Tingwei Adeck

See Also

normfluodat(), dat_col_names_rigid()

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale(nocomma_dat, tnp=3, cycles=40)
n = c('A','B','C')
sample_col_names <- dat_col_names_optimus(dat = fpath, resampled_scaled, n)
## End(Not run)
```

---

dat_col_names_prime *Attribute(s) naming function.*

Description

This function is used to name attribute(s). Attribute(s) names, in this case, are equivalent to the well labels found on the microplate reader. An attribute for a sample loaded into row A - column 1 will be named A1. In short, the function takes a clean data frame and returns attribute names that match the FLUOstar plate layout often presented as an Excel file.

Usage

```
dat_col_names_prime(
  dat = NULL,
  df,
  rows_used = NULL,
  cols_used = NULL,
```

```
  user_specific_labels = NULL
)
```

Arguments

dat	A string ("dat_1.dat") if the file is found within the present working directory (pwd) OR a path pointing directly to a ".dat" file.
df	A data frame that requires attribute labels.
rows_used	A character vector indicating the rows or tuples used on the microplate (usually a 96-well microplate). Initialized as NULL.
cols_used	A numeric vector indicating the plate columns or attributes used. Initialized as NULL.
user_specific_labels	A character vector where the user manually enters the used microplate wells based on the FLUOstar plate layout.

Value

Returns a character vector of attribute(s) names for the normalized data frame.

Author(s)

Tingwei Adeck

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale(nocomma_dat, tnp=3, cycles=40)
n = c('A','B','C')
sample_col_names <- dat_col_names_prime(dat = fpath, resampled_scaled, n)
## End(Not run)
```

---

dat_col_names_rigid *Attribute(s) naming function.*

Description

This function is used to name attribute(s). Attribute(s) names, in this case, are equivalent to the well labels found on the microplate reader. An attribute for a sample loaded into row A - column 1 will be named A1. In short, the function takes a clean data frame and returns attribute names that match the FLUOstar plate layout often presented as an Excel file.

Usage

```
dat_col_names_rigid(
  dat = NULL,
  df,
  rows_used = NULL,
  cols_used = NULL,
  user_specific_labels = NULL,
  read_direction = NULL
)
```

Arguments

dat	A string ("dat_1.dat") if the file is found within the present working directory (pwd) OR a path pointing directly to a ".dat" file.
df	A data frame that requires attribute labels.
rows_used	A character vector indicating the rows or tuples used on the microplate (usually a 96-well microplate). Initialized as NULL.
cols_used	A numeric vector indicating the plate columns or attributes used. Initialized as NULL.
user_specific_labels	A character vector where the user manually enters the used microplate wells based on the FLUOstar plate layout.
read_direction	A string input with two choices, “vertical” or “horizontal.” The user indicates “vertical” if the user intends to have a final data frame with samples arranged as sample type triplets (A1, B1, C1, A1, B1, C1) OR “horizontal” if the user intends to have a final data frame with samples arranged as clusters per sample type (A1, A2, A3, B1, B2, B3).

Value

Returns a character vector of attribute(s) names for the normalized data frame.

Note

Users are advised to input rows used but won’t be penalized for not doing so. If the user provides the rows used, then attribute names are generated for the user. The user must check to ensure that the names match the microplate layout. The user can leave the columns used as NULL if the user loaded samples from column 1 and did so in sequence. If the user fails to load in sequence from the first position, then the user must provide a numeric vector of columns used. For instance, where the user skips columns, the user will be prompted to interact with the program in order to ensure the final data frame has the correct attribute names. The user can bypass the rows used and columns used parameters if the user supplies a manually created character vector of the wells used in an experiment. The read direction parameter is used to determine the presentation of the samples in the final data frame.

This naming function only returns a character vector hence the rigid suffix.

Author(s)

Tingwei Adeck

See Also

dat_col_names_optimus()

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale(nocomma_dat, tnp=3, cycles=40)
n = c('A','B','C')
sample_col_names <- dat_col_names_rigid(dat = fpath, resampled_scaled, n)
## End(Not run)
```

---

defineparams *Define Plate Parameters*

Description

Define Plate Parameters

Usage

```
define_params(plate)

## Default S3 method:
define_params(plate)

## S3 method for class '`96well_plate`'
define_params(plate)

## S3 method for class '`384well_plate`'
define_params(plate)

## S3 method for class '`1536well_plate_t1`'
define_params(plate)

## S3 method for class '`1536well_plate_t2`'
define_params(plate)

set_default_params(plate)
```

Arguments

plate	plate

Value

default params

default params

default params

default params

default params

plate

Examples

```
## Not run: define_params(plate)
```

---

definestatus *Define Plate Status*

Description

Define Plate Status

Usage

```
define_status(plate)

## Default S3 method:
define_status(plate)

## S3 method for class '`96well_plate`'
define_status(plate)

## S3 method for class '`384well_plate`'
define_status(plate)

## S3 method for class '`1536well_plate_t1`'
define_status(plate)

## S3 method for class '`1536well_plate_t2`'
define_status(plate)

set_default_status(plate)

update_status_list(plate)

get_status_value(plate, index)
```

Arguments

plate	plate
index	index

Value

plate

status

status

status

status

status

status

plate

plate

plate

Examples

```
## Not run: define_steps(plate)
```

`definesteps` *Define Plate Steps*

Description

Define Plate Steps

Usage

```
define_steps(plate)

## Default S3 method:
define_steps(plate)

## S3 method for class 'normfluodbf_plate'
define_steps(plate)

## S3 method for class '`96well_plate`'
define_steps(plate)

## S3 method for class '`384well_plate`'
define_steps(plate)
```

```
## S3 method for class '`1536well_plate_t1`'
define_steps(plate)

## S3 method for class '`1536well_plate_t2`'
define_steps(plate)

set_default_steps(plate, ...)

update_steps_list(plate, new_key, new_value, index)
```

Arguments

plate	plate
...	custom steps
new_key	new_key
new_value	new_value
index	index

Value

steps

steps

steps

steps

steps

steps

plate

plate

Examples

```
## Not run: define_steps(plate)
## Not run: plate <- plate %>% update_steps_list('REMOVE_OUTLIER', 'remove_outlier', 3)
```

---

demo

Run the shiny App In addition to the functions provided in this package, the normfluodbf *package also provides an interactive tool that can be used to analyze liposome flux assay data more easily. The tool will be launched in a web browser.*

Description

Run the shiny App In addition to the functions provided in this package, the normfluodbf package also provides an interactive tool that can be used to analyze liposome flux assay data more easily. The tool will be launched in a web browser.

Usage

```
demo()
```

`detectoutliers` *Detect Outliers*

Description

Detect Outliers

Usage

```
detect_outliers_time_cn(plate, data)

detect_outliers_cn(plate, data)
```

Arguments

`plate`	plate
`data`	data

Value

data frame

data frame

data frame

Examples

```
## Not run: detect_outliers_time_cn(plate, data)
```

`dirutils` *Directory Utils*

Description

A function that facilitates a users' workflow by helping to check for DBFs in a directory.

A function that facilitates a users' workflow by helping to check for DATs in a directory.

Usage

```
list_dbfs(pathstring)

list_dats(pathstring)

is_file(pathstring)

is_dir(pathstring = NULL)

find_known_liposome_dat_file(fpath, fname)

find_known_liposome_dbf_file(fpath, fname)
```

Arguments

pathstring	path string
fpath	fpath
fname	fname

Value

directory utils

dbfs

dbfs

dbfs

dbfs

dbfs

dbfs

See Also

Other dirutils: normfluodbfcomms, sampledata

Examples

```
## Not run:
fpath <- system.file("extdata", package = "normfluodbf", mustWork = TRUE)
list_dbfs(fpath)
list_dats(fpath)
is_file(fpath)
find_known_liposome_dat_file(fpath, 'dat_1.dat')
find_known_liposome_dbf_file(fpath, 'liposomes_218')
## End(Not run)
```

formatplatedata *Format Plate Data*

Description

Format Plate Data

Usage

```
format_plate_data(plate)

## Default S3 method:
format_plate_data(plate)

## S3 method for class '`96well_plate`'
format_plate_data(plate)

## S3 method for class '`384well_plate`'
format_plate_data(plate)

## S3 method for class '`1536well_plate_t1`'
format_plate_data(plate)

## S3 method for class '`1536well_plate_t2`'
format_plate_data(plate)
```

Arguments

plate	plate

Value

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run: format_plate_data(plate)
```

getfilename *Get File Name(s)*

Description

Get File Name(s)

Usage

```
get_dbf_file_name(dbf_file)

get_dat_file_name(dat_file)

get_dat_common_name(dat_file)

get_common_dat_names(dat_files)
```

Arguments

dbf_file	DBF file
dat_file	DAT file
dat_files	DAT files

Value

file

name

name

name

name

Examples

```
## Not run:
get_dbf_file_name(dbf_file = "liposomes_218.dbf")
get_dat_file_name(dat_file = "dat_1.dat")
get_common_dat_names(dat_files = list.files(fpath, pattern = "\\.dat$"))
## End(Not run)
```

get_wells_used *Wells Used*

Description

Wells Used

Usage

```
get_wells_used(pl_data)
```

Arguments

pl_data	data

Value

wells used

See Also

Other plate_utils: check_dirt(), plate_data_summary(), platename, remove_leading_zero(), saveloadutils, set_plate_version(), type()

Examples

```
## Not run: get_wells_used(data)
```

globalcache *Globals Cache*

Description

Globals Cache

`isnormalized` *Is Normalized*

Description

Is Normalized

Usage

```
is_normalized(data, type = c("min-max", "z-score", "hundred"))
```

Arguments

`data`	type
`type`	type

Value

boolean

boolean

Examples

```
## Not run: is_normalized(data,type)
```

`launch` *Run the shiny App checking dependencies in R code ... WARNING In addition to the functions provided in this package, the* `normfluodbf` *package also provides an interactive tool that can be used to analyze liposome flux assay data more easily. The tool will be launched in a web browser.*

Description

Run the shiny App checking dependencies in R code ... WARNING In addition to the functions provided in this package, the `normfluodbf` package also provides an interactive tool that can be used to analyze liposome flux assay data more easily. The tool will be launched in a web browser.

Usage

```
launch()
```

liposomes_214 *liposomes_214.*

Description

FLUOstar .dbf file in wide format and unable to use for data analysis.

Usage

```
liposomes_214
```

Format

An object of class `data.frame` with 11 rows and 52 columns.

liposomes_215 *liposomes_215.*

Description

FLUOstar .dbf file in wide format and unable to use for data analysis.

Usage

```
liposomes_215
```

Format

An object of class `data.frame` with 11 rows and 52 columns.

liposomes_216 *liposomes_216.*

Description

FLUOstar .dbf file in wide format and unable to use for data analysis.

Usage

```
liposomes_216
```

Format

An object of class `data.frame` with 8 rows and 52 columns.

liposomes_218 *liposomes_218.*

Description

FLUOstar .dbf file in wide format and unable to use for data analysis.

Usage

```
liposomes_218
```

Format

An object of class `data.frame` with 11 rows and 52 columns.

liposomes_221 *liposomes_221.*

Description

FLUOstar .dbf file in wide format and unable to use for data analysis.

Usage

```
liposomes_221
```

Format

An object of class `data.frame` with 38 rows and 52 columns.

liposomes_227 *liposomes_227.*

Description

FLUOstar .dbf file in wide format and unable to use for data analysis.

Usage

```
liposomes_227
```

Format

An object of class `data.frame` with 29 rows and 52 columns.

liposome_fluor_dbfs *Cleans and Normalizes DBF files obtained from experiments using the FLUOstar Omega microplate reader (from BMG LABTECH).*

Description

The simplest function utilization scenario entails an input of the path to a DBF file obtained from the FLUOstar microplate (usually a 96-well microplate) reader; In a single step, this function will create a data frame, clean the data frame, normalize the data frame, append a "Cycle_Number" attribute, perform an adjustment to the “time” attribute and return a data frame that is ready for analysis. Since the initial publication of this package, several changes have been made to improve the user experience and to give the user more options to fine-tune the output from the package to meet the users’ aesthetic needs. Users who decide to move past the simplest utility scenario have been given more options to customize the output based on the users’ needs. Notably, several normalization sub-parameters have been provided in the package which yields different outputs based on what the user is used to seeing. Just as the FLUOstar instrument is built to handle an array of assays, this function is designed to be multi-dimensional (meaning it can handle data with the same DBF extension from other assay types), on the condition that the data from assay types other than liposome flux assays follow the same data format this package was designed to handle. Of course, users of this package are advised to pre-analyze DBF files from other assay types to ensure they are compliant with this package (compliance in this scenario is simple meaning DBF files from other assays should be like DBF files from liposome flux assays).

Usage

```
norm_tidy_dbf(
  file = NULL,
  norm_scale = NULL,
  transformed = NULL,
  fun = NA,
  ...
)

normfluordbf(file = NULL, norm_scale = NULL, transformed = NULL, fun = NA, ...)

normfluodbf(file = NULL, norm_scale, transformed = NULL, fun = NA, ...)
```

Arguments

file	A string ("liposomes_xxx.dbf") if the file is found within the present working directory (pwd) OR a path pointing directly to a ".dbf" file.
norm_scale	This parameter takes sub-parameters: ’raw’ , hundred’ , ’one’ , ’z-score’ , or ’decimal’ , which denotes the normalization type or scale; The parameter is initialized as NULL.
transformed	This parameter takes input ’log’, which denotes a logarithmic box-cox transformation; Initialized as NULL.

fun	A parameter defined as NA is used for Boolean expressions or manipulation.
...	An abstract placeholder or container parameter that can be used to capture extra variables if needed.

Value

A normalized data frame with an appended "Cycle_Number" attribute.

A normalized data frame with an appended "Cycle_Number" attribute.

A normalized data frame with an appended "Cycle_Number" attribute.

A normalized data frame with an appended "Cycle_Number" attribute.

Note

The default normalization sub-parameter outputs values in the 0-1 range. Unless a “norm_scale” level is specified by the user, the default output is in the 0-1 range. The “norm_scale” sub-parameter “decimal” is a machine-learning tool and should be avoided; it also provides no advantage for basic research analysis as its output operates on a sliding scale just like the raw data. Logarithmic transformation provides a minuscule advantage in data analysis and could/should be avoided. Backward compatibility is maintained in all updates, so there should be no issues with using the package the way the user was used to. The favorite "norm_scale" level is "z-score" since it divides the axis into negative and positive, thus facilitating interpretation.

The norm_scale must be provided if the user chooses to use this option.

Author(s)

Tingwei Adeck

See Also

normfluordbf(), normfluodat()

Examples

```
## Not run:
fpath <- system.file("extdata", "liposomes_214.dbf", package = "normfluodbf", mustWork = TRUE)
normalized_dbf <- norm_tidy_dbf(file=fpath, norm_scale = 'raw')
normalized_dbf <- normfluordbf(file=fpath, norm_scale = 'raw')
## End(Not run)
## Not run: wells = normfluodbf(lipsum_214, norm_scale = 'hundred')
```

loadplatedata *Load Plate Data*

Description

Load Plate Data

Usage

```
load_plate_data(plate)

load_plate_data(plate) <- value
```

Arguments

plate	plate
value	data

Value

plate

plate

plate

Examples

```
## Not run: load_plate_data(plate,value = data)
```

loadplatemeta *Load Plate Meta*

Description

Load Plate Meta

Usage

```
load_plate_meta(plate)

load_plate_meta(plate) <- value
```

Arguments

plate	plate
value	metadata

Value

plate

plate

plate

Examples

```
## Not run: load_plate_meta(plate, meta)
```

`modifyplatedata` *Modify Plate Data*

Description

Modify Plate Data

Usage

```
modify_plate_data(plate)

## Default S3 method:
modify_plate_data(plate)

## S3 method for class '`96well_plate`'
modify_plate_data(plate)

## S3 method for class '`384well_plate`'
modify_plate_data(plate)

## S3 method for class '`1536well_plate_t1`'
modify_plate_data(plate)

## S3 method for class '`1536well_plate_t2`'
modify_plate_data(plate)
```

Arguments

`plate`	plate

Value

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run: modify_plate_meta(plate)
```

`move_file` *Move File*

Description

Move File

Usage

```
move_file(source_path, destination_path)
```

Arguments

`source_path`	src
`destination_path`	dest

Value

kinetic file

Examples

```
## Not run:
source_file <- "path/to/source/file.txt"
destination_file <- "path/to/destination/file.txt"
move_file(source_file, destination_file)
move_file("~/Documents/Wip/R/PkgDev/pdf","~/Wip/R/PkgDev/R/pdf")
## End(Not run)
```

multiplot *Multiplot*

Description

Multiplot

Usage

```
multiplot(..., plotlist = NULL, file, cols = 1, layout = NULL)
```

Arguments

...	extra
plotlist	list
file	file
cols	cols
layout	layout

Value

grid plot

normalize *Normalize*

Description

Normalize

Usage

```
normalize(plate)

## Default S3 method:
normalize(plate)

## S3 method for class '`96well_plate`'
normalize(plate)

## S3 method for class '`384well_plate`'
normalize(plate)

## S3 method for class '`1536well_plate_t1`'
```

```
normalize(plate)

## S3 method for class '`1536well_plate_t2`'
normalize(plate)

normalize_dataframe(df)
```

Arguments

plate	plate
df	data frame

Value

plate

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run: normalize(plate)
normalize(plate)
## End(Not run)
```

`normalizebywell` *Normalize by Well*

Description

Normalize by Well

Usage

```
normalize_by_well(plate)

## Default S3 method:
normalize_by_well(plate)

## S3 method for class '`96well_plate`'
normalize_by_well(plate)
```

```
## S3 method for class '`384well_plate`'
normalize_by_well(plate)

## S3 method for class '`1536well_plate_t1`'
normalize_by_well(plate)

## S3 method for class '`1536well_plate_t2`'
normalize_by_well(plate)
```

Arguments

plate	plate

Value

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run: normalize_by_well(plate)
```

`normalizingagents` *Normalizing Agents*

Description

Normalizing Agents

Usage

```
min_max_norm(x)

min_max_norm_df(df)

min_max_norm_percent(x)

min_max_norm_percent_df(df)

norm_z(x)
```

`norm_z_df(df)`

`decimal_scaling(x)`

`decimal_scaling_df(df)`

`log_transformation(x)`

`roundfluor(x)`

Arguments

`x`	value(s)
`df`	data frame

Value

A normalized value when applied to a single value or a normalized attribute with values between the normalizing range.

normalized value (0-1)

normalized value (0-1)

normalized value (0-100)

normalized value (0-100)

normalized value (Z = N (0,1))

normalized value (Z = N (0,1))

normalized value

normalized value

log value

rounded value

See Also

Other normfluodbf_utils: `fluorthresholdcheck`

Examples

```
## Not run:
test_df <- as.data.frame(c(seq(40)))
colnames(test_df) <- "test"
test_df_norm <- lapply(test_df[1:ncol(test_df)], min_max_norm)
## End(Not run)
```

normfluodbfplots *Plot Plate - Favorite is Fluostar style*

Description

Plot Plate - Favorite is Fluostar style

Usage

```
## S3 method for class '`96well_plate`'
plot(
  x,
  whichplot = 1,
  fluorstarplot = 1 %in% whichplot,
  superimpose = 2 %in% whichplot,
  plate_layout = 3 %in% whichplot,
  plot_side_by_side = 4 %in% whichplot,
  legend_labels = NULL,
  plot_name = NULL,
  ...
)

## S3 method for class '`384well_plate`'
plot(
  x,
  whichplot = 1,
  fluorstarplot = 1 %in% whichplot,
  superimpose = 2 %in% whichplot,
  plate_layout = 3 %in% whichplot,
  plot_side_by_side = 4 %in% whichplot,
  legend_labels = NULL,
  plot_name = NULL,
  ...
)

## S3 method for class '`1536well_plate_t1`'
plot(
  x,
  whichplot = 1,
  fluorstarplot = 1 %in% whichplot,
  superimpose = 2 %in% whichplot,
  plate_layout = 3 %in% whichplot,
  plot_side_by_side = 4 %in% whichplot,
  legend_labels = NULL,
  plot_name = NULL,
  ...
)
```

```
## S3 method for class '`1536well_plate_t2`'
plot(
  x,
  whichplot = 1,
  fluorstarplot = 1 %in% whichplot,
  superimpose = 2 %in% whichplot,
  plate_layout = 3 %in% whichplot,
  plot_side_by_side = 4 %in% whichplot,
  legend_labels = NULL,
  plot_name = NULL,
  ...
)
```

Arguments

x	plot requirement
whichplot	int
fluorstarplot	whichplot = 1
superimpose	whichplot = 2
plate_layout	whichplot = 3
plot_side_by_side	whichplot = 4
legend_labels	labels whichplot = 2,4
plot_name	plot name
...	additional parameters

Value

plot object

print plot (return plate)

print plot (return plate)

print plot (return plate)

print plot (return plate)

Examples

```
## Not run: plot(plate, whichplot = 1)
```

`normfluordat` *Cleans and normalizes DAT files obtained from experiments using the FLUOstar Omega microplate reader (from BMG LABTECH).*

Description

The simplest case scenario entails inputting the name or directory of a DAT file as a string, the number of rows denoted by the tnp (test, negative, positive) parameter, and the number of cycles (selected by the user when running the FLUOstar instrument). The program takes these three baseline parameters, performs cleaning and normalization of the DAT file, and then appends an attribute called "Cycle_Number" to the normalized data frame.

The simplest case scenario entails inputting the name or directory of a DAT file as a string, the number of rows denoted by the tnp (test, negative, positive) parameter, and the number of cycles (selected by the user when running the FLUOstar instrument). The program takes these three baseline parameters, performs cleaning and normalization of the DAT file, and then appends an attribute called "Cycle_Number" to the normalized data frame.

The simplest case scenario entails inputting the name or directory of a DAT file as a string, the number of rows denoted by the tnp (test, negative, positive) parameter, and the number of cycles (selected by the user when running the FLUOstar instrument). The program takes these three baseline parameters, performs cleaning and normalization of the DAT file, and then appends an attribute called "Cycle_Number" to the normalized data frame.

The simplest case scenario entails inputting the name or directory of a DAT file as a string, the number of rows denoted by the tnp (test, negative, positive) parameter, and the number of cycles (selected by the user when running the FLUOstar instrument). The program takes these three baseline parameters, performs cleaning and normalization of the DAT file, and then appends an attribute called "Cycle_Number" to the normalized data frame.

Usage

```
normfluordat(
  dat,
  tnp,
  cycles,
  rows_used = NULL,
  cols_used = NULL,
  user_specific_labels = NULL,
  read_direction = NULL,
  na_omit = NULL
)

normfluodat(
  dat,
  tnp,
  cycles,
  rows_used = NULL,
  cols_used = NULL,
```

```
  user_specific_labels = NULL,
  read_direction = NULL,
  norm_scale = NULL,
  interval = NULL,
  first_end = NULL,
  pause_duration = NULL,
  end_time = NULL,
  normfluodbf.verbose = TRUE
)

normfluodatlite(
  dat,
  tnp,
  cycles,
  rows_used = NULL,
  cols_used = NULL,
  user_specific_labels = NULL,
  read_direction = NULL,
  norm_scale = NULL
)

normfluodatfull(
  dat,
  tnp,
  cycles,
  rows_used = NULL,
  cols_used = NULL,
  user_specific_labels = NULL,
  read_direction = NULL,
  norm_scale = NULL,
  na_omit = NULL
)
```

Arguments

dat	A string ("dat_1.dat") if the file is found within the present working directory (pwd) OR a path pointing directly to a ".dat" file.
tnp	A numeric value indicating the number of rows used. TNP is used as an acronym for Test, Negative, Positive.
cycles	A numeric value indicating the number of cycles selected by the user when running the FLUOstar instrument.
rows_used	A character vector of the rows used; ru = c('A','B','C').
cols_used	A numeric vector of the columns used; cu = c(1,2,3).
user_specific_labels	A character vector manually prepared by the user to denote the wells used on the microplate reader; usl = c('A1','B1','C1').
read_direction	A string input with two choices, "vertical" or "horizontal." The user indicates "vertical" if the user intends to have a final data frame with samples arranged

as sample type triplets (A1, B1, C1, A1, B1, C1) OR “horizontal” if the user intends to have a final data frame with samples arranged as clusters per sample type (A1, A2, A3, B1, B2, B3).

`na_omit` Takes a string "yes" OR "no".

`norm_scale` This parameter takes sub-parameters: ’raw’ , hundred’ , ’one’ , ’z-score’ , or ’decimal’ , which denotes the normalization type or scale; Initialized as NULL.

`interval` The time interval chosen for the assay often in seconds.

`first_end` The end time of the initial run, often the pause for the introduction of a new substance. This can be the cycle number chosen for the initial stop.

`pause_duration` The time between the first end (pause) and resumption of the assay.

`end_time` The final end time of the assay.

`normfluodbf.verbose`
verbose option

Value

A normalized data frame with an appended "Cycle_Number" attribute. The “Cycle_Number” attribute is the X-variable.

A normalized data frame with an appended "Cycle_Number" attribute. The “Cycle_Number” attribute is the X-variable.

A normalized data frame with an appended "Cycle_Number" attribute. The “Cycle_Number” attribute is the X-variable.

A normalized data frame with an appended "Cycle_Number" attribute. The “Cycle_Number” attribute is the X-variable.

Note

This function is a single-step function leveraging several subordinate functions. It is assumed that the user has the 3 baseline parameters to get this function working. Users must double-check attribute names to ensure they end up with accurate results.

This function is a single-step function leveraging several subordinate functions. It is assumed that the user has the 3 baseline parameters to get this function working. Users must double-check attribute names to ensure they end up with accurate results.

This function is a single-step function leveraging several subordinate functions. It is assumed that the user has the 3 baseline parameters to get this function working. Users must double-check attribute names to ensure they end up with accurate results.

Author(s)

Tingwei Adeck

See Also

`normfluodat()`

`normfluodatlite()`

normfluodat()

normfluodat()

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
normalized_fluo_dat <- normfluordat(dat=fpath, tnp = 3, cycles = 40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_4.dat", package = "normfluodbf", mustWork = TRUE)
normalized_fluo_dat <- normfluodat(dat=fpath, tnp = 3, cycles = 40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
normalized_fluo_dat <- normfluodatlite(dat=fpath, tnp = 3, cycles = 40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
normalized_fluo_dat <- normfluodatfull(dat=fpath, tnp = 3, cycles = 40)
## End(Not run)
```

---

parenttype *Parent Type*

Description

Parent Type

Usage

```
parent_plate_type(plate, type = NULL)

## S3 method for class 'normfluodbf_plate'
parent_plate_type(plate, type = NULL)

## Default S3 method:
parent_plate_type(plate, type = NULL)

## S3 method for class '`96well_plate`'
parent_plate_type(plate, type = NULL)

## S3 method for class '`384well_plate`'
parent_plate_type(plate, type = NULL)

## S3 method for class '`1536well_plate_t1`'
parent_plate_type(plate, type = NULL)

## S3 method for class '`1536well_plate_t2`'
parent_plate_type(plate, type = NULL)
```

Arguments

plate	plate
type	parent type

Value

plate

plate

plate

plate

plate

Examples

```
## Not run: parent_plate_type()
```

---

plate *Plate*

Description

Plate

Usage

```
empty_plate()

new_plate()

init_plate(plate = NULL, type = NULL, child_type = NULL)

setup_plate(plate, ...)

reset_plate(plate)

use_setup_plate()

use_initialized_plate()
```

Arguments

plate	plate
type	parent type
child_type	child type
...	dots

Value

plate

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run:
empty_plate()
new_plate()
init_plate()
setup_plate(plate)
reset_plate()
## End(Not run)
```

`platedata` *Plate Data*

Description

Plate Data

Usage

```
plate_data(file, tnp = NULL, cycles = NULL, rows_used = NULL, ...)
```

Arguments

`file`	file
`tnp`	tnp
`cycles`	cycles
`rows_used`	rows_used
`...`	dots

Value

plate data

plate data

Examples

```
## Not run: plate_data(file, tnp, cycles, rows_used = c(A,B,C), norm_scale = 'raw')
```

---

platemeta *Plate Meta*

Description

Plate Meta

Usage

```
plate_meta(plate, num_wells)

## Default S3 method:
plate_meta(plate, num_wells = 96L)

## S3 method for class '`96well_plate`'
plate_meta(plate, num_wells = 96L)

## S3 method for class '`384well_plate`'
plate_meta(plate, num_wells = 384L)

## S3 method for class '`1536well_plate_t1`'
plate_meta(plate, num_wells = 1536L)

## S3 method for class '`1536well_plate_t2`'
plate_meta(plate, num_wells)
```

Arguments

plate	plate
num_wells	number of wells

Value

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run: plate_meta(plate, num_wells = 96L)
```

`platename` *Plate Name*

Description

Plate Name

Usage

```
name(plate)

name(plate) <- value
```

Arguments

`plate`	plate
`value`	value

Value

plate

plate

See Also

Other plate_utils: `check_dirt()`, `get_wells_used()`, `plate_data_summary()`, `remove_leading_zero()`, `saveloadutils`, `set_plate_version()`, `type()`

`plate_data_summary` *Plate Data Summary*

Description

Plate Data Summary

Usage

```
plate_data_summary(plate)
```

Arguments

`plate`	plate

Value

sprintf string

See Also

Other plate_utils: `check_dirt()`, `get_wells_used()`, `platename`, `remove_leading_zero()`, `saveloadutils`, `set_plate_version()`, `type()`

`plate_types_tbl` *Plate Types Tibble*

Description

Plate Types Tibble

Plate Types List

Plate Types Vector

Plate Types Global

Usage

```
plate_types_tbl()

plate_types

plate_types_vector()
```

Format

An object of class `list` of length 5.

Details

The list equivalent of the tibble from `plate_types_tbl`.

The vector equivalent of the tibble from `plate_types_tbl`.

Value

A tibble

A list

A Vector

See Also

`plate_types()`

printer *Print*

Description

Print

Usage

```
## S3 method for class '`96well_plate`'
print(x, ...)

## S3 method for class '`384well_plate`'
print(x, ...)

## S3 method for class '`1536well_plate_t1`'
print(x, ...)

## S3 method for class '`1536well_plate_t2`'
print(x, ...)
```

Arguments

x	print requirement
...	placeholder

Value

plate

plate

plate

plate

Examples

```
## Not run: plate
```

quiet *Quiet*

Description

Quiet

Usage

```
quiet(
  expr,
  suppress_messages = FALSE,
  suppress_warnings = FALSE,
  suppress_output = FALSE,
  all = FALSE
)
```

Arguments

expr	expression
suppress_messages	logical
suppress_warnings	logical
suppress_output	logical
all	logical

Value

suppress comms

Examples

```
## Not run: quiet(expr)
```

removeoutliers *Outliers*

Description

Outliers

Usage

```
remove_outliers(plate)

## Default S3 method:
remove_outliers(plate)

## S3 method for class '`96well_plate`'
remove_outliers(plate)

## S3 method for class '`384well_plate`'
remove_outliers(plate)

## S3 method for class '`1536well_plate_t1`'
remove_outliers(plate)

## S3 method for class '`1536well_plate_t2`'
remove_outliers(plate)
```

Arguments

plate	plate

Value

plate

plate

plate

plate

plate

plate

plate

Note

Works on a data frame not in well format.

Works on a data frame not in well format.

Works on a data frame not in well format.

Works on a data frame not in well format.

Works on a data frame not in well format.

Examples

```
## Not run: remove_outliers(plate)
```

remove_leading_zero *Format Well Names*

Description

Format Well Names

Usage

```
remove_leading_zero(names_vector)
```

Arguments

names_vector	column names

Value

vector

See Also

Other plate_utils: check_dirt(), get_wells_used(), plate_data_summary(), platename, saveloadutils, set_plate_version(), type()

Examples

```
## Not run: remove_leading_zero(names)
```

replace_word_in_file *Replace Word*

Description

Replace Word

Usage

```
replace_word_in_file(file_path, old_word, new_word)
```

Arguments

file_path	path
old_word	old func name
new_word	new func name

Value

file

Note

Solves the inconvenient problem of renaming a function correctly and having to manually correct it.

Examples

```
## Not run: replace_word_in_file('R/plate_plot.R','plot_fluostar_style', 'plot_in_well')
```

---

resample_dat_scale *A function to create an attribute or column for each sample loaded into the microplate wells.*

Description

Creates a data frame where each sample loaded into the microplate wells has a separate attribute.

Creates a data frame where each sample loaded into the microplate wells has a separate attribute. NA values are retained for more control.

A function that takes tuples or rows consisting of several samples and perform a putative resampling to yield another data frame with a separate attribute for each sample.

A function that takes tuples or rows consisting of several samples and perform a putative resampling to yield another data frame with a separate attribute for each sample. NA values are retained.

A function that takes tuples or rows consisting of several samples and perform a putative resampling to yield another data frame with a separate attribute for each sample.

A function that takes tuples or rows consisting of several samples and perform a putative resampling to yield another data frame with a separate attribute for each sample.

Creates a data frame where each sample loaded into the microplate wells has a separate attribute.

Creates a data frame where each sample loaded into the microplate wells has a separate attribute. NA values are retained.

Creates a data frame where each sample loaded into the microplate wells has a separate attribute.

Creates a data frame where each sample loaded into the microplate wells has a separate attribute.

Usage

```
resample_dat_scale(df, tnp, cycles)

resample_dat_scale_naretainer(df, tnp, cycles)

resample_dat_scale_alt(df, tnp, cycles, na_omit = NULL)

resample_dat_scale_alt_na(df, tnp, cycles)
```

```
resample_dat_scale_alt_bf_na(df, tnp, cycles)

resample_dat_scale_alt_bfv(df, tnp, cycles)

resample_dat_scale_optimus(df, tnp, cycles)

resample_dat_scale_optimus_na(df, tnp, cycles)

resample_dat_scale_optimus_backend(df, tnp, cycles, na_omit = NULL)

resample_vect_scale(df, tnp, cycles, method = c("normal", "brute", "vector"))
```

Arguments

df	A clean data frame with attributes or tuples containing a mixture of samples.
tnp	A numeric value indicating the number of rows used. TNP is used as an acronym for Test, Negative, Positive.
cycles	A numeric value indicating the number of cycles selected by the user when running the FLUOstar instrument.
na_omit	Takes a string "yes" OR "no".
method	A string 'normal', 'brute' or 'vector' to specify the method of resampling.

Value

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

Note

This function builds on or scales-up @seealso `resample_dat()`, hence the suffix scale. This function is less optimized than @seealso `resample_dat_scale_optimus()`.

This function builds on or scales-up @seealso `resample_dat()`, hence the suffix scale. This function is less optimized than @seealso `resample_dat_scale_optimus()`.

This function builds on or scales-up @seealso `resample_dat()`, hence the suffix scale. This function is more optimized than @seealso `resample_dat_scale()`, hence the suffix scale_optimus.

This function builds on or scales-up @seealso resample_dat(), hence the suffix scale. This function is more optimized than @seealso resample_dat_scale(), hence the suffix scale_optimus.

This function builds on or scales-up @seealso resample_dat(), hence the suffix scale. This function is more optimized than @seealso resample_dat_scale(), hence the suffix scale_optimus.

This is the pseudo-vectorized approach and should be a more efficient function. This function will produce a vertical layout as defined in this package. This function inspired by the lapply approach pretty much applies the

Author(s)

Tingwei Adeck

See Also

resample_dat()

resample_dat()

resample_dat_alt()

resample_dat_alt()

resample_dat_alt(), resample_dat_scale_alt()

resample_dat_alt(), resample_dat_scale_alt()

resample_dat()

resample_dat()

resample_dat()

resample_dat_vect()

resample_dat_vect(). As a matter of fact, I took this approach to create compatibility with lapply and rapply but that failed.

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_4.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_4.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale_naretainer(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale_alt(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
```

```
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale_alt_na(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_4.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale_alt_bf_na(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_4.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale_alt_bfv(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale_optimus(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale_optimus_na(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
resampled_scaled <- resample_dat_scale_optimus_backend(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_3.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
alt_test_scale <- resample_vect_scale(nocomma_dat,3,40, method = 'brute')
alt_test_scale <- resample_vect_scale(nocomma_dat,3,40, method = 'normal')
alt_test_scale <- resample_vect_scale(nocomma_dat,3,40, method = 'vector')
alt_test_scale_norm <- lapply(alt_test_scale, min_max_norm)
## End(Not run)
```

`resample_dat_vect` *A function to create an attribute or column for each sample loaded into the microplate wells.*

Description

Designed as a prototype function to take a single attribute or column consisting of several samples and perform a putative resampling to yield another data frame with a separate attribute for each sample.

Designed as a prototype function to take a single attribute or column consisting of several samples and perform a putative resampling to yield another data frame with a separate attribute for each sample.

: Designed as a prototype function to take a single tuple or row consisting of several samples and perform a putative resampling to yield another data frame with a separate attribute for each sample.

Usage

```
resample_dat_vect(df, tnp, cycles, output = NULL)

resample_dat(df, tnp, cycles)

resample_dat_alt(df, tnp, cycles)
```

Arguments

df	A clean data frame with attributes or tuples containing a mixture of samples.
tnp	A numeric value indicating the number of rows used. TNP is used as an acronym for Test, Negative, Positive.
cycles	A numeric value indicating the number of cycles selected by the user when running the FLUOstar instrument.
output	A choice between "df' and 'vector' outputs. Leave NULL for a data frame.

Value

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

A new data frame where separated samples are assigned a separate attribute or column.

Note

This is the vectorized approach and should be a more efficient function when compared to say

Author(s)

Tingwei Adeck

See Also

`resample_vect_scale()`

`resample_dat()` or @seealso `resample_dat_alt()`. This function will produce a vertical layout as defined in this package.

resample_dat_scale(), resample_dat_scale_optimus()

resample_dat_scale_alt()

Examples

```
## Not run:
fpath <- system.file("extdata", "dat_4.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
samples_delineated <- resample_dat_vect(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_5.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
samples_delineated <- resample_dat(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
## Not run:
fpath <- system.file("extdata", "dat_5.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odddat_optimus(dat_df)
samples_delineated <- resample_dat_alt(nocomma_dat, tnp=3, cycles=40)
## End(Not run)
```

---

sampledata *Get Development Data*

Description

Get Development Data

Usage

```
sample_data_dir()

sample_data_file(gotofile = NULL)
```

Arguments

gotofile	file

Value

directory data

See Also

Other dirutils: dirutils, normfluodbfcomms

Examples

```
## Not run:
fpath <- system.file("extdata", package = "normfluodbf", mustWork = TRUE)
sample_data_dir()
sample_data_file(gotofile = NULL)
## End(Not run)
```

`saveloadutils` *Save and Load Plate*

Description

Save and Load Plate

Usage

```
save_plate(plate, suffix = NULL, interactive = F)

save_rds_plate(plate, save_name, use_tempfile = F)

load_rds_plate(plate, interactive = F)

var_str(var)
```

Arguments

`plate`	plate
`suffix`	suffix
`interactive`	boolean
`save_name`	name
`use_tempfile`	boolean
`var`	variable

Value

plate

plate

plate

plate

See Also

Other plate_utils: `check_dirt()`, `get_wells_used()`, `plate_data_summary()`, `platename`, `remove_leading_zero()`, `set_plate_version()`, `type()`

setsteps *Set Plate Steps*

Description

Set Plate Steps

Usage

```
steps(plate)

steps(plate) <- value
```

Arguments

plate	plate
value	value

Value

plate

plate

plate

See Also

Other steps: stepsutils

Examples

```
## Not run: steps(plate)
```

set_assiette_type *Set The Plate Types*

Description

Set The Plate Types

Usage

```
set_assiette_type(plate, type = NULL, child_type = NULL)
```

Arguments

plate	A parent plate
type	parent plate type
child_type	child plate type

Details

Non-recursive approach.

Value

plate

Examples

```
## Not run: set_assiette_type(parent_plate_type,96well_plate)
```

---

set_plate_type *Set The Child Plate Type*

Description

Set The Child Plate Type

Usage

```
set_plate_type(parent_plate, type)
```

Arguments

parent_plate	plate
type	child plate type

Details

This is a recursive method.

Value

plate

Examples

```
## Not run:
x = set_plate_type(parent_plate,"96well_plate")
print(class(x))
## End(Not run)
```

set_plate_version *Plate Version*

Description

Plate Version

Usage

```
set_plate_version(plate, pkg)
```

Arguments

plate	plate
pkg	package

Value

plate

See Also

Other plate_utils: check_dirt(), get_wells_used(), plate_data_summary(), platename, remove_leading_zero(), saveloadutils, type()

Examples

```
## Not run: set_plate_version(plate,pkg)
```

status *Status*

Description

Status

Usage

```
status(plate)

status(plate) <- value

dirty(plate)

dirty(plate) <- value

is_plate_dirty(plate)
```

Arguments

plate	plate
value	value

Value

plate

plate

plate

plate

plate

logical

Examples

```
## Not run: status(plate, status)
```

---

stepspipeline *Steps Pipeline*

Description

Steps Pipeline

Usage

```
.next_step(plate, n = 1)

next_step(plate, n = 1)

run_steps(plate, reset = FALSE, ...)
```

Arguments

plate	plate
n	n
reset	reset
...	dots

Value

plate

plate

plate

plate

Note

Recursive function to implement steps in the plate until all steps in the pipeline are complete

Recursive function to implement steps in the plate until all steps in the pipeline are complete

Examples

```
## Not run: next_step(plate, n=1)
```

`stepsutils` *Steps Utils*

Description

Steps Utils

Usage

```
step(plate, step)

step_name(plate, step)

.step_name(plate, step)

get_step_key_by_index(steps, index)

step_begin(...)

step_end(...)

has_step(plate, step)

check_step(plate, step)

steps_complete(plate)
```

Arguments

`plate`	plate
`step`	step
`steps`	steps
`index`	index
`...`	dots

Value

plate

step number

step name

step name

step name

boolean

boolean

boolean

Note

Step start time utilizing the cache

Step start time utilizing the cache

See Also

Other steps: `setsteps`

Examples

```
## Not run: step(plate, step)
```

`test_boilerplate` *Test Boilerplate*

Description

Test Boilerplate

Open Testfile

Usage

```
test_boilerplate(file_name = NULL)

open_testfile(testfile)
```

Arguments

`file_name`	file name
`testfile`	test file

Value

test boilerplate

open test file

Note

Solves the inconvenient process of navigating to the tests folder every time

Examples

```
## Not run: test_boilerplate(file_name = "test_remove_shit.R")
## Not run: open_testfile('test_pipeline.R')
```

`type` *Plate Type*

Description

Plate Type

Usage

```
type(plate, all = FALSE)
```

Arguments

`plate`	plate
`all`	Boolean

Value

class attribute

See Also

Other plate_utils: `check_dirt()`, `get_wells_used()`, `plate_data_summary()`, `platename`, `remove_leading_zero()`, `saveloadutils`, `set_plate_version()`

Examples

```
## Not run: type(plate)
```

uploadplatedata *Upload Plate Data*

Description

Upload Plate Data

Usage

```
upload_data(plate, file, ...)

## Default S3 method:
upload_data(plate, file, ...)

## S3 method for class '`96well_plate`'
upload_data(plate, file, ...)

## S3 method for class '`384well_plate`'
upload_data(plate, file, ...)

## S3 method for class '`1536well_plate_t1`'
upload_data(plate, file, ...)

## S3 method for class '`1536well_plate_t2`'
upload_data(plate, file, ...)
```

Arguments

plate	plate
file	file
...	dots

Value

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run: upload_data(plate, file, ...)
```

xycoordinates *Plot Coordinates*

Description

Plot Coordinates

Usage

```
x_var_one(plate)

x_var_one(plate) <- value

x_var_two(plate)

x_var_two(plate) <- value

y_var(plate)

y_var(plate) <- value

x_var_one_label(plate)

x_var_one_label(plate) <- value

x_var_two_label(plate)

x_var_two_label(plate) <- value
```

Arguments

plate	plate
value	value

Value

plate

plate

plate

plate

plate

plate

plate

plate

plate

plate

Examples

```
## Not run: x_var_one(plate,value)
```

`%there%` *The %there% operator*

Description

The %there% operator

Usage

```
dfile %there% dirpath
```

Arguments

`dfile`	file
`dirpath`	directory

Value

logical

Examples

```
## Not run:
fpath <- system.file("extdata", package = "normfluodbf", mustWork = TRUE)
"dat_1.dat" %there% fpath
## End(Not run)
```

Index