Package ‘normfluodbf’

October 28, 2023

Title Cleans and Normalizes 'FLUOstar' 'DBF' and 'DAT' Files

Version 1.4.3.9000

Description Converts a 'FLUOstar' 'DBF' File into a Normalized Data Frame, Ready for Analysis.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Suggests knitr,
learnr,
rmarkdown,
testthat (>= 3.0.0)

Config/testthat/edition 3

Imports data.table,
foreign,
tidyr,
tibble,
dplyr,
ggplot2,
cranlogs,
emojifont,
stats,
stringr

Depends R (>= 2.10)

LazyData true

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

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

VignetteBuilder knitr

R topics documented:

`clean_odddat_optimus` *Title: A function to partially clean dat files from FLuostar experiments*

Description

The function takes a dirty data frame and returns a clean data frame with the only exception being that most values in the returned data frame have commas at the end The data frame that goes into the function comes from reading dat files from FLUOstar experiments See the example code

Usage

```
clean_odddat_optimus(df)
```

Arguments

`df`	A data frame with n number of rows

Value

A new data frame with most impurities taken out; See the example

Note

This function should work on all types of Fluostar dat files (unlike the clean_evendat() function).

Author(s)

Tingwei Adeck

Examples

```
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)
```

`clean_odd_cc` *Title: A function to completely clean dat files from FLuostar experiments*

Description

The function takes a dirty data frame and returns a clean data frame. The function takes care of more QC concerns not seen in previous cleaning functions. The function accounts for possible machine data formatting when users skip wells by retaining NA values, which ensures that the user does not loose samples in data analysis.

Usage

```
clean_odd_cc(df)
```

Arguments

`df`	A data frame with n number of rows

Value

A new data frame with NAs retained.

Note

This function is superior to clean_odddat().

Author(s)

Tingwei Adeck

See Also

`normfluodat()`

Examples

```
fpath <- system.file("extdata", "dat_3.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
partial_cleaned_dat <- clean_odd_cc(dat_df)
```

comma_cleaner *Title: A function to remove unwanted commas in a partially cleaned data frame*

Description

The function takes a dirty data frame (comma_df) derived from either the clean_odddat or clean_evendat functions. The resulting data frame is a cleaned data frame with numeric values.

Usage

```
comma_cleaner(comma_df)
```

Arguments

`comma_df`	A dirty data frame (numeric values have commas at the end) with n number of rows

Value

A new data frame with commas at the end of numbers taken out

Note

This function is a subordinate function and follows a sequence of actions. In this package, this function cannot be used as a standalone.

Author(s)

Tingwei Adeck

Examples

```
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- comma_cleaner(dat_df)
```

dat_1 *dat_1.*

Description

FLUOstar .dat files used for creation of the update and unusable for data 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 data 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 data 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_col_names *Title: A function to obtain attribute names for experimental samples*

Description

The function takes a clean data frame, data on the experiment and returns the column names that match the FLUOstar plate reader.

Usage

```
dat_col_names(
  df,
  rows_used = NULL,
  cols_used = NULL,
  user_specific_labels = NULL
)
```

Arguments

`df`	A clean data frame obtained from the large scale delineation of samples.
`rows_used`	A character vector representing the plate rows used; eg ru <- c('A','B','C'). can be used in sequence or out of sequence.
`cols_used`	A numeric vector representing the plate columns used; eg cu <- c(1,2,3,4). keep as null if all the columns were used or columns are used in sequence.
`user_specific_labels`	A character vector with specific sample labels based on the plate setup

Value

Returns column names that will be added to the normalized data frame that contains all samples

Note

This function is a subordinate function and follows a sequence of actions. In this package, this function cannot be used as a standalone. Also, some work is needed here on the part of the user because i have no access to their setup file. A function that takes the setup excel file from the user should be part of the next update to prevent the user from doing much work. The program is always going to need rows_used. The user can choose to specify columns used but typically if things are in sequence then everything should be fine. The extreme case is an extreme unorthodox plate (hard to know when this will happen) and then the user must either specify rows used directly or the user is given a prompt by R to input rows used.

Author(s)

Tingwei Adeck

Examples

```
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odd_cc(dat_df)
resampled_scaled <- resample_dat_scale(nocomma_dat, tnp=3, cycles=40)
n = c('A','B','C')
sample_col_names <- dat_col_names(resampled_scaled, n)
```

`dat_col_names_optimus` *Title: A function to obtain attribute names for experimental samples*

Description

The function takes a clean data frame, data on the experiment and returns the column names that match the FLUOstar plate reader.

Usage

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

Arguments

`df`	A clean data frame obtained from the large scale delineation of samples.
`rows_used`	A character vector representing the plate rows used; eg ru <- c('A','B','C'). can be used in sequence or out of sequence.
`cols_used`	A numeric vector representing the plate columns used; eg cu <- c(1,2,3,4). keep as null if all the columns were used or columns are used in sequence.
`user_specific_labels`	A character vector with specific sample labels based on the plate setup
`read_direction`	User can leave null (vertical) for machine up-down read OR 'horizontal' for machine left-right read

Value

Returns column names that will be added to the normalized data frame that contains all samples

Note

This function is a subordinate function and follows a sequence of actions. In this package, this function cannot be used as a standalone. Also, some work is needed here on the part of the user because i have no access to their setup file. A function that takes the setup excel file from the user should be part of the next update to prevent the user from doing much work. The program is always going to need rows_used. The user can choose to specify columns used but typically if things are in sequence then everything should be fine. The extreme case is an extreme unorthodox plate (hard to know when this will happen) and then the user must either specify rows used directly or the user is given a prompt by R to input rows used.

Author(s)

Tingwei Adeck

Examples

```
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odd_cc(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(resampled_scaled, n)
```

`dat_col_names_prime` *Title: A function to obtain attribute names for experimental samples*

Description

The function takes a clean data frame, data on the experiment and returns the column names that match the FLUOstar plate reader.

Usage

```
dat_col_names_prime(
  df,
  rows_used = NULL,
  cols_used = NULL,
  user_specific_labels = NULL
)
```

Arguments

`df`	A clean data frame obtained from the large scale delineation of samples.
`rows_used`	A character vector representing the plate rows used; eg ru <- c('A','B','C'). can be used in sequence or out of sequence.
`cols_used`	A numeric vector representing the plate columns used; eg cu <- c(1,2,3,4). keep as null if all the columns were used or columns are used in sequence.
`user_specific_labels`	A character vector with specific sample labels based on the plate setup

Value

Returns column names that will be added to the normalized data frame that contains all samples

Note

This function is a subordinate function and follows a sequence of actions. In this package, this function cannot be used as a standalone. Also, some work is needed here on the part of the user because i have no access to their setup file. A function that takes the setup excel file from the user should be part of the next update to prevent the user from doing much work. The program is always going to need rows_used. The user can choose to specify columns used but typically if things are in sequence then everything should be fine. The extreme case is an extreme unorthodox plate (hard to know when this will happen) and then the user must either specify rows used directly or the user is given a prompt by R to input rows used.

Author(s)

Tingwei Adeck

Examples

```
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odd_cc(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(resampled_scaled, n)
```

`decimal_scaling` *Title: A function that performs decimal scaling (not normalization) of attributes*

Description

Title: A function that performs decimal scaling (not normalization) of attributes

Usage

```
decimal_scaling(x)
```

Arguments

`x`	Column or attribute values passed into the decimal scaling function

Value

Attribute(s) with values that have the decimal place moved to facilitate machine learning

Note

The lapply function is required to apply the function across several columns in a data set. This is NOT a normalization function because the data still exist on a sliding scale. This function is here for demonstration purposes and should be used for exercise as it is here for educational purposes for the inventor of the package.

Author(s)

Tingwei Adeck

References

https://www.statology.org/how-to-normalize-data-in-r/

Examples

```
test_df <- as.data.frame(c(seq(40)))
colnames(test_df) <- "test"
test_df_norm <- lapply(test_df[1:ncol(test_df)], decimal_scaling)
```

fluor_threshold_check *Title: Fluorescence experiment quality control function-Check fluorescence threshold*

Description

Check that fluorescence does not exceed 2^16. Experimental design issues should be investigated at fluorescence levels this high. The inventor of this program made a similar mistake when he began these experiments.

Usage

```
fluor_threshold_check(clean_df)
```

Arguments

`clean_df`	A cleaned dat or dbf file

Value

A polite warning message for the researchers next experimental design and the rows and columns with problem values.

Examples

```
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odd_cc(dat_df)
resampled_scaled <- resample_dat_scale(nocomma_dat, tnp=3, cycles=40)
resampled_scaled[1,1] <- 2^16
for(i in 1:ncol(resampled_scaled)){
for(j in 1:nrow(resampled_scaled)){
if(i ==2 && j ==2){
resampled_scaled[j,i] <- 2^16
}
}
}
fluor_threshold_check(resampled_scaled)
```

generic_identifier *Title: A generic identifier similar to unique identifier but end users supply a column name.*

Description

A function that creates a column 1:nrow(df) and steps by 1 but gives you the option to use any column name.

Usage

```
generic_identifier(numrows, col_name)
```

Arguments

`numrows`	The number of rows in a data frame of interest (nrows(df) can be used).
`col_name`	The desired column name for the column.

Value

A new single column data frame with the desired attribute added.

Author(s)

Tingwei Adeck

Examples

```
generic_identifier(40,col_name="Cycle_No")
```

`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.

`log_transformation` *Title: A function that performs log transformation of data*

Description

Title: A function that performs log transformation of data

Usage

```
log_transformation(x)
```

Arguments

`x`	Column or attribute values passed into the Z-standardization function

Value

Log transformed attributes

Note

The lapply function is required to apply the function across several columns in a data set.

Author(s)

Tingwei Adeck

References

https://www.statology.org/how-to-normalize-data-in-r/

Examples

```
test_df <- as.data.frame(c(seq(40)))
colnames(test_df) <- "test"
test_df_norm <- lapply(test_df[1:ncol(test_df)], log_transformation)
```

`min_max_norm`	*Title: A function that performs Min-Max normalization(0-1 range) of attributes that require normalization*

Description

Title: A function that performs Min-Max normalization(0-1 range) of attributes that require normalization

Usage

```
min_max_norm(x)
```

Arguments

`x`	Column or attribute values passed into the min-max normalization function

Value

A normalized value (between 0 and 1) when used as a standalone function, or a normalized attribute(s) when used with lapply.

Note

The lapply function is required to apply the function across several columns in a data set.

Author(s)

Tingwei Adeck (Adapted from Statology)

References

https://www.statology.org/how-to-normalize-data-in-r/

Examples

```
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)
```

`min_max_norm_percent` *Title: A function that performs Min-Max normalization (0-100 range) of attributes that require normalization*

Description

Title: A function that performs Min-Max normalization (0-100 range) of attributes that require normalization

Usage

```
min_max_norm_percent(x)
```

Arguments

`x`	Column or attribute values passed into the min-max normalization function

Value

A normalized value (between 0 and 100) when used as a standalone function, or a normalized attribute(s) when used with lapply.

Note

The lapply function is required to apply the function across several columns in a data set.

Author(s)

Tingwei Adeck

References

https://www.statology.org/how-to-normalize-data-in-r/

Examples

```
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_percent)
```

`nfd_tracker` *Title: Normfluodbf tracker*

Description

A simple tracker for the package. A fun addition to the package.

Usage

```
nfd_tracker(package, period = NULL, plot = NULL)
```

Arguments

`package`	package name as string (example: package = 'normfluodbf')
`period`	takes 'last-month' or 'last-week'; default is NULL
`plot`	takes 'cum' or 'daily'; default is null

Value

A data frame of cumulative downloads or a data frame of daily downloads

Note

The tracker is not modular so only works for normfluodbf hence in the package

Author(s)

Tingwei Adeck

Examples

```
nfd_tracker(package = 'normfluodbf', period = 'last-month', plot = 'cum')
```

`normfluodat` *Title: The root function that returns a normalized data frame with the Cycle No ready for analysis*

Description

Input a dat file (dat file directory) and required parameters and BOOM the researcher has a normalized data frame ripe and ready for clean analysis

Usage

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

Arguments

`dat`	directory to the users FLUOstar dat file
`tnp`	Stands for test,negative,positive (sample types); the number should match the number of sample types in the plate reader even if repeating a sample type
`cycles`	The number of cycles chosen by the researcher. In the case of this package 40 is the standard but ensure to have the right number of samples
`rows_used`	A character vector of the rows used, eg n = c('A','B','C')
`cols_used`	A numeric vector of the columns used, eg m = c(2,4,6)
`user_specific_labels`	A character vector with specific sample labels based on the plate setup
`read_direction`	User can leave null for machine up-down read OR 'horizontal' for machine left-right read

Value

A normalized data frame with the x-variable (Cycle_No), ready for analysis

Note

This is the MAIN function and stands alone but is dependent on the subordinate functions. If the user understands what they are doing this is all they need. The user should use the user_specific_labels parameter for naming variables if they have an extreme unorthodox experimental setup.

Author(s)

Tingwei Adeck

Examples

```
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
n <- c('A','B','C')
normalized_fluo_dat <- normfluodat(dat=fpath, tnp = 3, cycles = 40, n, read_direction = 'vertical')
```

normfluordbf — *Title: Cleans and Normalizes ".dbf" files obtained from experiments using the FLUOstar microplate reader.*

Description

Input the path to a ".dbf" file obtained from the FLUOstar microplate (usually a 96-well microplate) reader; this function will create a data frame, clean the data frame, normalize the data frame, append a "Cycle_Number" column and return a data frame that is ready for analysis. Most importantly, this function is a single_step function. Also, the function can be extended to other ".dbf" files if they follow the format for which this function was designed; this is totally at the users' discretion.

Usage

```
normfluordbf(file = NULL, norm_scale = NULL, 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, from FLUOstar experiments.
`norm_scale`	This parameter can taken in 'hundred', 'one', or 'z-score' which denotes the normalization type; Initialized as NULL.
`transformed`	This parameter can take in 'log' which denotes a logarithmic box-cox transformation; Initialized as NULL.
`fun`	A variable defined as NA, used for boolean expressions or manipulation.
`...`	A container object that can be used to capture extra variables if needed.

Value

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

Note

Re-nomenclature of norm_tidy_dbf to a more appropriate name that facilitates function utilization. Users can continue with the old name ("norm_tidy_dbf") but this is a better name in my opinion.

Examples

```
fpath <- system.file("extdata", "liposomes_214.dbf", package = "normfluodbf", mustWork = TRUE)
normalized_dbf <- normfluordbf(file=fpath)
```

norm_tidy_dbf — *Title: 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,
  ...
)
```

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.

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.

See Also

`normfluordbf()`, `normfluodat()`

Examples

```
fpath <- system.file("extdata", "liposomes_214.dbf", package = "normfluodbf", mustWork = TRUE)
normalized_dbf_default <- norm_tidy_dbf(file=fpath)
normalized_dbf_scale100 <- norm_tidy_dbf(file=fpath, norm_scale = 'hundred')
normalized_dbf_scalez <- norm_tidy_dbf(file=fpath, norm_scale = 'z-score')
```

`norm_z` *Title: A function that performs Z-score standardization (or normalization) of attributes*

Description

Title: A function that performs Z-score standardization (or normalization) of attributes

Usage

```
norm_z(x)
```

Arguments

x	Column or attribute values passed into the Z-standardization function

Value

Attribute(s) with values that have mean 0 and standard deviation 1

Note

The lapply function is required to apply the function across several columns in a data set.

Author(s)

Tingwei Adeck

References

https://www.statology.org/how-to-normalize-data-in-r/

Examples

```
test_df <- as.data.frame(c(seq(40)))
colnames(test_df) <- "test"
test_df_norm <- lapply(test_df[1:ncol(test_df)], norm_z)
```

`resample_dat` *Title: A function to extract sample types from the 120 tuples data cleaned data frame.*

Description

Designed as a prototype function to take a single column from the cleaned dat data frame and return the n=3 sample types (n can be any number based on the number of sample types used) as separate attributes with n=40 tuples (n can vary based on the number of cycles ran) The function is well thought and accounts for almost any scenario.

Usage

```
resample_dat(df, tnp, cycles, samples_per_tnp = NULL)
```

Arguments

`df`	A clean data frame with n number of rows
`tnp`	Stands for test,negative,positive (sample types); the number should match the number of sample types in the plate reader even if repeating a sample type
`cycles`	The number of cycles chosen by the researcher. In the case of this package 40 is the standard but ensure to have the right number of samples
`samples_per_tnp`	An optional parameter thought to be useful but had no use based on the approach taken to solve this problem; Will be useful in future functions

Value

A new data frame with attributes matching the number of sample types and tuples matching the number of cycles. In short it returns delineated samples.

Note

This function is a subordinate function and prototype that turned out very useful.

Author(s)

Tingwei Adeck

Examples

```
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odd_cc(dat_df)
col_1 <- nocomma_dat[,1]
col_1 <- as.data.frame(col_1)
samples_delineated <- resample_dat(col_1, tnp=3, cycles=40)
```

resample_dat_scale *Title: A scaling up of the resample_dat() function*

Description

Performs the role of the prototype resample_dat() function but it is scaled up to work on all columns in longer-form data frame.

Usage

```
resample_dat_scale(df, tnp, cycles)
```

Arguments

df	A clean data frame with n number of rows
tnp	Stands for test,negative,positive (sample types); the number should match the number of sample types in the plate reader even if repeating a sample type
cycles	The number of cycles chosen by the researcher. In the case of this package 40 is the standard but ensure to have the right number of samples

Value

A new data frame with attributes matching the number of sample types and tuples matching the number of cycles. In short it returns delineated samples.

Note

This function is the pre-requisite to the parent or main function of the update. As a matter of fact, this function is modified to produce the parent or main function.

Author(s)

Tingwei Adeck

See Also

resample_dat()

Examples

```
fpath <- system.file("extdata", "dat_1.dat", package = "normfluodbf", mustWork = TRUE)
dat_df <- read.table(file=fpath)
nocomma_dat <- clean_odd_cc(dat_df)
resampled_scaled <- resample_dat_scale(nocomma_dat, tnp=3, cycles=40)
```

roundfluor — *Title: Round attribute elements or attributes*

Description

Round an attribute element or attribute.

Usage

```
roundfluor(x)
```

Arguments

`x`	An attribute element that needs to be rounded up

Value

A rounded up element OR a rounded up attribute when assisted by lapply

Examples

```
test_df <- as.data.frame(c(seq(40)))
colnames(test_df) <- "test"
test_df_norm <- lapply(test_df[1:ncol(test_df)], roundfluor)
```

unique_identifier — *Title: A function to append a unique identifier attribute to any data frame within the normfluodbf package.*

Description

The function in the context of normfluodbf creates an attribute called Cycle_Number and appends this attribute to the cleaned or wrangled data frame derived from the dirty DBF file.

Usage

```
unique_identifier(df)
```

Arguments

`df`	A data frame with 1:n number of rows

Value

A data frame with the Cycle_Number attribute appended to the end of the data frame.

Note

The function operates in a closed system, meaning it is primarily designed to work with this package ONLY. Other use cases are simply a coincidence.

Author(s)

Tingwei Adeck

See Also

normfluodat(), norm_tidy_dbf(), normfluordbf(), generic_identifier()

Examples

```
test_df <- as.data.frame(c(seq(40)))
colnames(test_df) <- "test"
unique_identifier(test_df)
```

Index