Package 'eufmdis.adapt'

May 8, 2026

Type Package

Title Analyse 'EuFMDiS' Output Files via a Shiny App

Version 0.1.0

Author Ian Kopacka [aut, cre],
Tatiana Marschik [aut],
Elena Sassu [aut],
Annette Nigsch [aut],
Food and Agriculture Organization of the United Nations (FAO) [cph, fnd]

Maintainer Ian Kopacka <ian.kopacka@ages.at>

Description Analyses 'EuFMDiS' output files in a Shiny App. The distributions of relevant output parameters are described in form of tables (quantiles) and plots. The App is called using eufmdis.adapt::run_adapt().

License GPL (>= 3)

Encoding UTF-8

Imports graphics, stats, utils, rlang, grDevices, magrittr, dplyr, tibble, tidyselect, ggplot2, shiny, shinydashboard, shinyWidgets, htmltools, DT

RoxygenNote 7.2.3

NeedsCompilation no

Repository CRAN

Date/Publication 2023-09-12 06:10:09 UTC

Contents

`check_availability` *Check if list items are empty*

Description

The function argument is a list of data frames that are required for some subsequent analysis. If any of the data frames are empty (i.e. the data have not been uploaded to the app), a message is returned as HTML code listing the names of the required data frames (= names of list items).

Usage

```
check_availability(list_data)
```

Arguments

`list_data`	Named list of data frames

Details

In the ADAPT app, individual analyses can only be performed if the necessary output files are uploaded. If certain files are not uploaded, the app produces empty data frames. In the app, the function `check_availability()` is used with the necessary data frames to check if they have been uploaded and displays a message of the form "To generate this analysis, please upload the following reports: x, y" otherwise.

Value

Possibly empty HTML text, listing names of required data frames.

Author(s)

Ian Kopacka

cleanup_names *Clean up a vector of column names*

Description

The function takes a vector of column names as an argument and returns a cleaned up version of it.

Usage

```
cleanup_names(x)
```

Arguments

x	A character vector

Details

The following changes are made: - names are converted to lower case - dots are replaced by underscores - underscores in the beginning and end of a string are removed - multiple underscores are replaced by a single one

Value

A character vector

Author(s)

Ian Kopacka

compute_p_value *Compute p value for freedom from disease sample*

Description

Compute the probability of drawing no positives in a sample of n items from a Population of N containing n_dis positives.

Usage

```
compute_p_value(N, n, n_dis)
```

Arguments

N	Integer; size of the population
n	Integer; size of the sample
n_dis	Integer; number of positives in the population

Details

The probability is computed using the hypergeometric distribution. This function is used in `compute_sample_size`.

Value

Returns the probability of not finding any positives in the sample as a numeric between 0 and 1

Author(s)

Ian Kopacka

See Also

`compute_sample_size`

`compute_sample_size` *Compute sample size for freedom from disease*

Description

Compute sample size for a one stage freedom from disease survey for given Population size, design prevalence and accuracy, assuming a perfect diagnostic test.

Usage

```
compute_sample_size(N, prev, accuracy)
```

Arguments

`N`	Integer containing the Size of the population
`prev`	Numeric between 0 and 1; design prevalence
`accuracy`	Numeric between 0 and 1; accuracy of the survey (i.e. detection probability)

Details

The function finds the optimal sample size using a bisection method.

Value

Sample size (integer).

Author(s)

Ian Kopacka

`compute_sample_size_vectorised` *Compute sample size for freedom from disease (vectorised)*

Description

Compute sample size for a one stage freedom from disease survey for given Population size, design prevalence and accuracy, assuming a perfect diagnostic test. Vectorised version of `compute_sample_size`.

Usage

```
compute_sample_size_vectorised(N, prev, accuracy)
```

Arguments

`N`	Integer vector containing the population sizes
`prev`	Numeric between 0 and 1; design prevalence
`accuracy`	Numeric between 0 and 1; accuracy of the survey (i.e. detection probability)

Details

Uses vapply to vectorise `compute_sample_size` over the population size `N`. `prev` and `accuracy` must be scalars. For the sake of efficiency, the sample size is only computed once for every different value of `N`, even if they apper multiple times in the vector.

Value

Sample size (integer vector).

Author(s)

Ian Kopacka

See Also

`compute_sample_size`

`create_diag_control` *Data analysis for diagnostic samples during the control phase*

Description

Function to perform the data analysis, necessary for the analysis of the diagnostic samples during the control phase

Usage

```
create_diag_control(
  herd_summary,
  farm_summary,
  par_diag_control_ffd_prev,
  par_diag_control_ffd_certainty,
  par_diag_control_edta,
  par_diag_control_serum,
  par_diag_control_bulk_milk,
  par_diag_control_lesions_smrum,
  par_diag_control_lesions_pigs,
  par_diag_control_lesions_cattle,
  rel_cols_farm_summary_dc,
  rel_cols_herd_summary_dc,
  herd_types_dairy,
  herd_types_small_ruminants,
  herd_types_pigs,
  herd_types_cattle
)
```

Arguments

`herd_summary`	Data frame; EuFMDIS output file "Herd summary"
`farm_summary`	Data frame; EuFMDIS output file "Farm summary"
`par_diag_control_ffd_prev`	numeric between 0 and 100; design prevalence for the computation of the sample size according to freedom from disease
`par_diag_control_ffd_certainty`	numeric between 0 and 100; desired accuracy for the computation of the sample size according to freedom from disease
`par_diag_control_edta`	positive integer; Number of blood samples (EDTA) per symptomatic suspect holding
`par_diag_control_serum`	positive integer; Number of blood samples (serum) per symptomatic suspect holding

`par_diag_control_bulk_milk`
: positive integer; Number of bulk milk samples per dairy farm

`par_diag_control_lesions_smrum`
: positive integer; Number of acute lesion samples for small ruminants per farm

`par_diag_control_lesions_pigs`
: positive integer; Number of acute lesion samples for pigs per farm

`par_diag_control_lesions_cattle`
: positive integer; Number of acute lesion samples for cattle per farm

`rel_cols_farm_summary_dc`
: character vector of column names of the data frame `farm_summary` that are required for the analysis

`rel_cols_herd_summary_dc`
: character vector of column names of the data frame `herd_summary` that are required for the analysis

`herd_types_dairy`
: character vector listing the different herd types that are associated with dairy herds

`herd_types_small_ruminants`
: character vector listing the different herd types that are associated with small ruminant herds

`herd_types_pigs`
: character vector listing the different herd types that are associated with pig herds

`herd_types_cattle`
: character vector listing the different herd types that are associated with cattle herds

Details

This function is used internally to prepare the input data for the output (tables and plots) in the sub menu "Diagnostic tests control phase" of the ADAPT App.

Value

Returns an aggregated data frame with one line per simulation run. The data frame contains auxiliary variables needed to appriximate the number of diagnostic samples required during the control phase as well as the estimated values for number of bulk milk samples (n_bulk_milk), acute lesions (n_acute_lesion), swabs (n_swabs), blood samples for edta analysis (n_blood_edta) and serum analysis (n_blood_serum).

`create_long_data_frame` *Reshape wide data frame with combined column names*

Description

The function identifies columns whose name contains a combination of two categorical characteristics (e.g. farm type and output parameter), splits them up and reshapes the data to a long format.

Usage

```
create_long_data_frame(dat, categories, name_categories, starts_with = FALSE)
```

Arguments

dat	Data frame with combined column names (e.g. type_A_farms, type_B_farms, type_A_animals, type_B_animals)
categories	Character vector of possible values of categories in the column names (e.g. c("type_A", "type_B"))
name_categories	Character; name of the newly created column that contains the categories in the long data frame
starts_with	Logical; Flag indicating how the combined columns should be identified. starts_with = TRUE enforces a stricter search mode where only columns are considered whose name starts with the given string.

Details

The function looks for combined columns based on the category names provided in the argument categories. Two modes of searching are possible: starts_with = FALSE (=default) looks for all columns whose name contains the strings in categories, whereas starts_with = TRUE only includes columns whose name starts with the string. Relevant combinations of #' values that are not found in the wide data frame are filled with NA in the ling data frame.

Value

A long data frame where the combined columns have been split up

Author(s)

Ian Kopacka

---

discumulate_data *Inverse of Cumulative Sum*

Description

Computes the inverse of the cumsum function

Usage

```
discumulate_data(value_cum)
```

Arguments

value_cum	numeric vector; usually the result of cumulating values.

Value

A vector of the same length as `value_cum`

Author(s)

Ian Kopacka

`format_numbers_DT` *Safe wrapper for DT::formatCurrency Wrapper for* `DT::formatCurrency` *that returns NULL when the input table is* `NULL` *(instead of throwing an error).*

Description

Safe wrapper for DT::formatCurrency Wrapper for `DT::formatCurrency` that returns NULL when the input table is `NULL` (instead of throwing an error).

Usage

```
format_numbers_DT(x, ...)
```

Arguments

`x`	A table object created from `DT::datatable()`
`...`	other arguments passed to `DT::formatCurrency`

Value

Behaves the same output as `DT::formatCurrency` except when `x` is `NULL`. Then `NULL` is returned and no error is thrown.

Author(s)

Ian Kopacka

`import_data_file` *Import data from csv file in Shiny App*

Description

The function is used in the ADAPT app to import data from uploaded csv files into a data frame. Only relevant columns are returned, the column names are matched and unified, so that data produced by different versions of the EuFMDiS software can be used.

Usage

```
import_data_file(
  pattern,
  names_files,
  paths_files,
  def_columns,
  transpose = FALSE
)
```

Arguments

`pattern`	Character string containing a regular expression to identify the correct file by its name.
`names_files`	Character vector of file names as they were uploaded (= file name on the original file system from which they were uploaded)
`paths_files`	Character vector of file names + absolute paths of the files in the local hard drive to which they were uploaded. Each entry of `paths_files` corresponds to an entry of `names_files`. They must have the same length.
`def_columns`	Data frame of meta information containing the possible column names in the different versions of the EuFMDiS output files. The data frame must contain columns `Datensatz` (name of the relevant data frame; this corresponds to values used in `pattern`), `Name_Parameter` (the unified column name in the generated return value) and columns containing possible variations of the name in the different versions of EuFMDiS. Each column contains the notation in one version of EuFMDiS; the column names must begin with the string `Spalte`
`transpose`	Logical flag (default = FALSE). COntrols whether the data frame should be transposed prior to any data manipulation/extraction.

Value

Data frame containing the columns defined in `def_columns` for the relevant `Datensatz` according to the argument `pattern`.

plot_barchart *Plot bar chart with error bars*

Description

Creates a bar chart with error bars using `ggplot2::geom_col`.

Usage

```
plot_barchart(x)
```

Arguments

`x`	Data frame with columns `par` or `parameter` containing the name of the parameter, `q2.5` or `q2_5` for the 2.5 percentiles (i.e. the lower values for the error bars), `median` for the median values (i.e. the height of the bars) and `q97.5` or `q97_5` for the 97.5 percentiles (i.e. the upper values for the error bars).

Value

Returns an object of class `ggplot` and prints it to the graphics device.

plot_barchart_euros *Plot bar chart with error bars an Euro notation*

Description

Creates a bar chart with error bars using `ggplot2::geom_col`.

Usage

```
plot_barchart_euros(x, country)
```

Arguments

`x`	Data frame with columns `par` or `parameter` containing the name of the parameter, `q2.5` or `q2_5` for the 2.5 percentiles (i.e. the lower values for the error bars), `median` for the median values (i.e. the height of the bars) and `q97.5` or `q97_5` for the 97.5 percentiles (i.e. the upper values for the error bars).
`country`	Character to be displayed in the Plot title.

Value

Returns an object of class `ggplot` and prints it to the graphics device.

See Also

`plot_barchart`

plot_distribution *Plot the distribution of a variable*

Description

Creates a histogram of the value along with a horizontal boxplot above it to show the distribution of a variable.

Usage

```
plot_distribution(x, parameter, main = "")
```

Arguments

x	A numeric vector
parameter	Character to use as label of the x-axis
main	(optional) character to use as plot title

Value

No return value. Creates a plot.

Author(s)

Ian Kopacka

plot_time_series *Plot graph of a time series with daily error margin*

Description

Creates a line plot with a shaded polygon showing daily error margins (uncertainty ranges)

Usage

```
plot_time_series(x, parameter, main = "")
```

Arguments

x	Data frame with columns day containing the counter for the time steps (=days), q025 for the 2.5 percentiles (i.e. the lower values for the error margin), median for the median values (i.e. the values for the line plot) and q975 for the 97.5 percentiles (i.e. the upper values for the error margin).
parameter	Character to use as label of the y-axis
main	(optional) character to use as plot title

Value

No return value. Creates a plot.

Author(s)

Ian Kopacka

`run_adapt` *Run ADAPT Shiny App*

Description

This function runs the Shiny App "ADAPT" to analyse 'EuFMDiS' output files.

Usage

```
run_adapt()
```

Details

Upload the relevant csv output files via the "Upload files" dialog to trigger the analysis.

Value

no return value; starts a Shiny app

Author(s)

Ian Kopacka

Examples

```
if (interactive()) {
    run_adapt()
}
```

Index