Package ‘RIbench’

October 12, 2022

Type Package
Version 1.0.1
Date 2022-09-16
Title Benchmark Suite for Indirect Methods for RI Estimation
Author Tatjana Ammer [aut, cre],
    Christopher Rank [aut],
    Andre Schuetzenmeister [aut]
Maintainer Tatjana Ammer <tatjana.ammer@roche.com>
Depends R (>= 3.3.0)
Imports stats, optparse, digest, data.table, graphics, grDevices,
    RColorBrewer
Suggests knitr, rmarkdown
Description
The provided benchmark suite enables the automated evaluation and comparison of any exist-
ing and novel indirect method for reference interval ('RI') estimation in a systematic way.
Indirect methods take routine measurements of diagnostic tests, containing pathological and non-
pathological samples as input and use sophisticated
statistical methods to derive a model describing the distribution of the non-
pathological samples, which can then be
used to derive reference intervals. The benchmark suite contains 5,760 simu-
lated test sets with varying difficulty.
To include any indirect method, a custom wrapper function needs to be provided.
The package offers functions for generating the test sets, executing the indirect method and evalu-
ating the results.
See ?RIbench or vignette("RIbench_package") for a more comprehensive description of the fea-
tures.
A detailed description and application is described in Ammer T., Schuetzenmeis-
ter A., Prokosch H.-U., Zierk J., Rank C.M., Rauh M. `RIbench: A Proposed Bench-
mark for the Standardized Evaluation of Indirect Methods for Reference Interval Estima-
License GPL (>= 3)
VignetteBuilder knitr, rmarkdown
NeedsCompilation no
Repository CRAN
Date/Publication 2022-09-17 19:56:27 UTC

R topics documented:

- Rlbench-package
- addGrid
- as.rgb
- BoxCox
- computeDirect
- computePerfMeas
- computePerfMeasAll
- computeRIs
- computeRIsAll
- computeRuntimeAll
- computeSubResults
- defineSubset
- evaluateAlgorithmResults
- evaluateBiomarkerTestSets
- formatNumber
- generateBiomarkerTestSets
- generateBoxPlotOneAnalyte
- generateBoxplotsDistTypes
- generateBoxplotsMultipleCats
- generateDataFiles
- generateMD5
- generateScatterplotsAll
- getBenchmarkResults
- getRI
- getRIsAllwithoutModel
- getRuntime
- getSubset
- getSubsetForDefinedCats
- invBoxCox
- loadTestsetDefinition
- mergeAnalytes
- mergeSummaryErrors
- plotBarplot
- plotBoxplot
- plotScatterplot
- print.RWDRI
- progressInd
- readResultFiles
- readResultFilesAll
- readResultsAndComputeErrors
- restrictSet
- runDirectMethod
RIbench-package

RIbench: Benchmark Suite for the Standardized Evaluation of Indirect Methods for Reference Interval Estimation

Description

RIbench enables the automated evaluation and comparison of any existing and novel indirect method in a systematic way. Indirect methods take routine measurements of diagnostic tests, containing pathological and non-pathological samples as input and use sophisticated statistical methods to derive a model describing the distribution of the non-pathological samples, which can then be used to derive reference intervals. The benchmark suite contains 5,760 simulated data sets with varying difficulty. To include any indirect method, a custom wrapper function needs to be provided. The package offers functions for generating the test sets `generateBiomarkerTestSets`, executing the indirect method `evaluateBiomarkerTestSets` and evaluating the results `evaluateAlgorithmResults`.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>RIbench</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.0.1</td>
</tr>
<tr>
<td>Date</td>
<td>2022-09-16</td>
</tr>
<tr>
<td>License</td>
<td>GPL (&gt;=3)</td>
</tr>
<tr>
<td>LazyLoad</td>
<td>yes</td>
</tr>
</tbody>
</table>

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>, Christopher M Rank <christopher.rank@roche.com>, Andre Schuetzenmeister <andre.schuetzenmeister@roche.com>

References

addGrid

Add a grid to an existing plot.

Description

It is possible to use automatically determined grid lines (x=NULL, y=NULL) or specifying the number of cells x = 3, y = 4 as done by grid. Additionally, x- and y-locations of grid-lines can be specified, e.g. x = 1:10, y = seq(0,10,2).

Usage

addGrid(x = NULL, y = NULL, col = "lightgray", lwd = 1L, lty = 3L)

Arguments

- **x** (integer, numeric) single integer specifies number of cells, numeric vector specifies vertical grid-lines
- **y** (integer, numeric) single integer specifies number of cells, numeric vector specifies horizontal grid-lines
- **col** (character) color of grid-lines
- **lwd** (integer) line width of grid-lines
- **lty** (integer) line type of grid-lines

Author(s)

Andre Schuetzenmeister <andre.schuetzenmeister@roche.com>

as.rgb

Convert color-names or RGB-code to possibly semi-transparent RGB-code.

Description

Function takes the name of a color and converts it into the rgb space. Parameter "alpha" allows to specify the transparency within [0,1], 0 meaning completely transparent and 1 meaning completely opaque. If an RGB-code is provided and alpha != 1, the RGB-code of the transparency adapted color will be returned.

Usage

as.rgb(col = "black", alpha = 1)
BoxCox

Arguments

- col: (character) name of the color to be converted/transformed into RGB-space (code). Only those colors can be used which are part of the set returned by function colors(). Defaults to "black".
- alpha: (numeric) value specifying the transparency to be used, 0 = completely transparent, 1 = opaque.

Value

RGB-code

Author(s)

Andre Schuetzenmeister <andre.schuetzenmeister@roche.com>

---

**BoxCox**

*One-parameter Box-Cox transformation.*

Description

One-parameter Box-Cox transformation.

Usage

BoxCox(x, lambda)

Arguments

- x: (numeric) data to be transformed
- lambda: (numeric) Box-Cox transformation parameter

Value

(numeric) vector with Box-Cox transformation of x

Author(s)

Andre Schuetzenmeister <andre.schuetzenmeister@roche.com>
computeDirect  

Function to simulate the direct method

Description

Function to simulate the direct method

Usage

computeDirect(
  N = 120,
  analyte,
  params,
  seed = 123,
  NIter = 10000,
  RIperc = c(0.025, 0.975)
)

Arguments

N  (integer) specifying the number of samples used as sample size for the direct method, default: 120
analyte  (character) specifying the biomarker that is currently simulated
params  (list) of parameters for non-pathological distribution (nonp_mu, nonp_sigma, nonp_lambda, and nonp_shift)
seed  (integer) specifying the seed used for the simulation, default: 123
NIter  (integer) specifying the number of times N samples should be drawn out of the simulated non-pathological distribution (default: 10,000)
RIperc  (numeric) value specifying the percentiles, which define the reference interval

Value

(data frame) with the estimated reference intervals for NIter iterations

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
computePerfMeas

Function for computing performance measurements

Description

Function for computing performance measurements

Usage

computePerfMeas(
  analyte,  
  algo, 
  resRIs, 
  subTable, 
  RIperc = c(0.025, 0.975),
  cutoffZ = 5
)

Arguments

analyte (character) specifying current analyzed analyte
algo (character) specifying used algorithm
resRIs (data.frame) with all calculated reference intervals
subTable (data.frame) containing all information about the simulated test sets
RIperc (numeric) vector specifying the percentiles for the reference interval, default: 0.025 and 0.975
cutoffZ (numeric) specifying if a cutoff should be used to classify results as implausible and exclude from analysis

Value

updated data frame with computed performance measures

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
computePerfMeasAll

Function for computing reference intervals for all markers

Description

Function for computing reference intervals for all markers

Usage

computePerfMeasAll(analytes, algo, risIn, tableTCs, cutoffZ = 5)

Arguments

analytes (character) listing all analytes for which the result files should be parsed
algo (character) specifying used algorithm
risIn (list) with data frame of all calculated reference intervals
tableTCs (data.frame) containing all information about the simulated test sets
cutoffZ (integer) specifying if and if so which cutoff should be used to classify results as implausible (default: 5)

Value

list with the calculated errors as data frame for each marker

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

computeRIs

Function for computing reference intervals

Description

Function for computing reference intervals

Usage

computeRIs(
  analyte,
  algo,
  results,
  tableTCs,
  RIperc = c(0.025, 0.975),
  truncNormal = FALSE
)

computeRIsAll

Arguments

analyte (character) specifying analyte
algo (character) specifying used algorithm
results (list) with all calculated results as RWDRI objects
tableTCs (data frame) containing all information about the simulated test sets
RIPerc (numeric) vector specifying the percentiles for the reference interval, default: 0.025 and 0.975
truncNormal (logical) specifying if a normal distribution truncated at zero shall be assumed

Value
data frame with computed reference intervals

Author(s)
Tatjana Ammer <tatjana.ammer@roche.com>

data frame with computed reference intervals for each marker

computeRIsAll (character) listing all markers for which the result files should be parsed
algo (character) specifying used algorithm
results (list) with all calculated results for all markers as RWDRI objects
tableTCs (data frame) containing all information about the simulated test sets
truncNormal (logical) specifying if a normal distribution truncated at zero shall be assumed

Value
list with the calculated reference intervals as data frame for each marker

Author(s)
Tatjana Ammer <tatjana.ammer@roche.com>
computeRuntimeAll  
*Function to compute runtime statistics for all analytes*

**Description**
Function to compute runtime statistics for all analytes

**Usage**
computeRuntimeAll( analytes, algo, risIn, tableTCs )

**Arguments**
- analytes  
  (character) listing all analytes for which the result files should be parsed
- algo  
  (character) specifying used algorithm
- risIn  
  (list) with data frame of all calculated reference intervals and runtime
- tableTCs  
  (data.frame) containing all information about the simulated test cases

**Value**
(list) wit runtime statistics per analyte and data frames with raw runtime overall and per analyte

**Author(s)**
Tatjana Ammer <tatjana.ammer@roche.com>

computeSubResults  
*Helper function to compute the subscores for the distribution types and the mentioned categories*

**Description**
Helper function to compute the subscores for the distribution types and the mentioned categories

**Usage**
computeSubResults(  
  errorDf,  
  tableTCs,  
  distCat,  
  errorParam,  
  catList,  
  catLabels,  
  perfCombination = "mean"  
)
**defineSubset**

**Arguments**
- `errorDf` (data frame) containing the estimate reference intervals and all computed error measures
- `tableTCs` (data.frame) containing all information about the simulated test sets
- `distCat` (character) specifying the distribution category
- `errorParam` (character) specifying for which error parameter the data frame should be generated
- `catList` (character) vector containing the categories to split the dataset
- `catLabels` (character) vector containing the labels that will be used for the categories
- `perfCombination` (character) specifying if mean (default), median or sum should be computed

**Value**
(data frame) containing the computed subscores

**Author(s)**
Tatjana Ammer <tatjana.ammer@roche.com>

---

**defineSubset**  
*Function for defining a subset that is used for analyzing the computation time and can be used for other subanalyses.*

**Description**
Function for defining a subset that is used for analyzing the computation time and can be used for other subanalyses.

**Usage**
`defineSubset(tableTCs = NULL, N = 50, seed = 123)`

**Arguments**
- `tableTCs` (data frame) describing the pre-defined testcases
- `N` (integer) describing the number of testcases per biomarker contained in the subset (default: 50)
- `seed` (integer) specifying the seed used for defining the subset, default: 123

**Value**
(data frame) describing the updated table with all test case definitions.

**Author(s)**
Tatjana Ammer <tatjana.ammer@roche.com>
evaluateAlgorithmResults

*Convenience Function to generate all result plots and calculate the benchmark score*

**Description**

Convenience Function to generate all result plots and calculate the benchmark score

**Usage**

```r
evaluateAlgorithmResults(
  workingDir = "", 
  algoNames = NULL, 
  subset = "all", 
  evalFolder = "Evaluation", 
  withDirect = TRUE, 
  withMean = TRUE, 
  outline = TRUE, 
  errorParam = c("zzDevAbs_Ov", "AbsPercError_Ov", "AbsError_Ov"), 
  cutoffZ = 5, 
  cols = NULL, ... 
)
```

**Arguments**

- `workingDir` (character) specifying the working directory: Plots will be stored in `workingDir/evalFolder` and results will be used from `workingDir/Results/algoName/biomarker`
- `algoNames` (character) vector specifying all algorithms that should be part of the evaluation
- `subset` (character, numeric, or data.frame) to specify for which subset the algorithm should be evaluated. character options: 'all' (default) for all test sets, a distribution type: 'normal', 'skewed', 'heavilySkewed', 'shifted'; a biomarker: 'Hb', 'Ca', 'FT4', 'AST', 'LACT', 'GGT', 'TSH', 'IgE', 'CRP', 'LDH'; 'Runtime' for runtime analysis subset; numeric option: number of test sets per biomarker, e.g. 10; data.frame: customized subset of table with test set specifications
- `evalFolder` (character) specifying the name of the ouptut directory, Plots will be stored in `workingDir/evalFolder`, default: ‘Evaluation’
- `withDirect` (logical) indicating whether the direct method should be simulated for comparison (default:TRUE)
- `withMean` (logical) indicating whether the mean should be plotted as well (default: TRUE)
- `outline` (logical) indicating whether outliers should be drawn (TRUE, default), or not (FALSE)
- `errorParam` (character) specifying for which error parameter the data frame should be generated, choose between absolute z-score deviation ("zzDevAbs_Ov"), absolute percentage error ("AbsPercError_Ov"), and absolute error ("AbsError_Ov")
evaluateBiomarkerTestSets

Wrapper function to evaluate all test sets or a specified subset for a specified algorithm.

cutoffZ (integer) specifying if and if so which cutoff for the absolute z-score deviation should be used to classify results as implausible and exclude them from the overall benchmark score (default: 5)
cols (character) vector specifying the colors used for the different algorithms
additional arguments to be passed to the method, e.g. "truncNormal" (logical) vector specifying if a normal distribution truncated at zero shall be assumed, can be either TRUE/FALSE or a vector with TRUE/FALSE for each algorithm; "colDirect" (character) specifying the color used for the direct method, default: "grey" "ylab" (character) specifying the label for the y-axis

Value

(data frame) containing the computed benchmark results

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

Examples

## Not run:
# Ensure that 'generateBiomarkerTestSets()' and 'evaluateBiomarkerTestSets()' is called # with the same workingDir and for all mentioned algorithms before calling this function.

# first example, evaluation for several algorithms
benchmarkScore <- evaluateAlgorithmResults(workingDir=tempdir(), algoNames=c("Hoffmann", "TML", "kosmic", "TMc", "refineR"))
# The function will create several plots saved in workingDir/Evaluation.

# second example, evaluation for only one algorithm and a defined subset
benchmarkScore <- evaluateAlgorithmResults(workingDir = tempdir(), algoNames = "refineR", subset = "Ca")

# third example, saving the results in a different folder, and setting a different cutoff # for the absolute z-score deviation
benchmarkScore <- evaluateAlgorithmResults(workingDir = tempdir(), algoNames = "refineR", subset = 'Ca', cutoffZ = 4, evalFolder = "Eval_Test")

## End(Not run)
Description

Wrapper function to evaluate all test sets or a specified subset for a specified algorithm.

Usage

evaluateBiomarkerTestSets(
  workingDir = "",  # specifying the working directory: Results will be stored in 'workingDir/Results/algo/biomarker' and data will be used from 'workingDir/Data/biomarker'
  algoName = "refineR",  # specifying the name of the algorithm that is evaluated
  algoFunction = "findRI",  # specifying the name of the function needed for estimating RIs
  libs = "refineR",  # (list) containing all libraries needed for executing the algorithm
  sourceFiles = NULL,  # (list) containing all source files needed for executing the algorithm
  params = NULL,  # (list) with additional parameters needed for calling algoFunction
  requireDecimals = FALSE,  # (logical) indicating whether the algorithm needs the number of decimal places (TRUE) or not (FALSE, default)
  requirePercentiles = FALSE,  # (logical) indicating whether only percentiles and no model is estimated
  subset = "all",  # (character, numeric, or data.frame) to specify for which subset the algorithm should be executed. character options: 'all' (default) for all test sets; a distribution type: 'normal', 'skewed', 'heavilySkewed', 'shifted'; a biomarker: 'Hb', 'Ca', 'FT4', 'AST', 'LACT', 'GGT', 'TSH', 'IgE', 'CRP', 'LDH'; 'Runtime' for runtime analysis subset; numeric option: number of test sets per biomarker, e.g. 10; data.frame: customized subset of table with test set specifications
  timeLimit = 14400,  # (integer) specifying the maximum amount of time in seconds allowed to execute one single estimation (default: 14400 sec (4h))
  verbose = TRUE,  # (logical) indicating if the progress counter should be shown (default: TRUE)
  showWarnings = FALSE  # (logical) indicating whether warnings from the call to the indirect method/algorithm should be shown (default: FALSE)
)
evaluateBiomarkerTestSets

... additional arguments to be passed to the method, e.g. specified in- and output directory ('inputDir', 'outputDir')

Value

(data frame) containing information about the test sets where the algorithm terminated the R session or failed to report a result

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

Examples

## Not run:
# The evaluation of all test sets can take several hours depending on
# the computation time of the algorithm.
# Wrapper function for indirect method required, see vignette("RIBench_package")
# Ensure that 'generateBiomarkerTestSets()' is called with the same workingDir
# before calling this function.

# first generic example
evaluateBiomarkerTestSets(workingDir = tempdir(), algoName = 'myOwnAlgo', algoFunction = 'estimateModel', libs = c('myOwnAlgo'), sourceFiles = list("C:\Temp\MyAlgoWrapper.R"), requireDecimals = FALSE, requirePercentiles = FALSE, subset = 'all', timeLimit = 14400)

# second example, evaluation for only 'Calcium' test sets.
progress <- evaluateBiomarkerTestSets(workingDir = tempdir(), algoName = 'myOwnAlgo', algoFunction = 'estimateModel', libs = c('myOwnAlgo'), subset = 'Ca')

# third example, evaluation for only a subset testsets that follow a skewed distribution.
progress <- evaluateBiomarkerTestSets(workingDir = tempdir(), algoName = 'myOwnAlgo', algoFunction = 'estimateModel', libs = c('myOwnAlgo'), subset = 'skewed')

# forth example, evaluation for a subset of 3 testsets per biomarker.
progress <- evaluateBiomarkerTestSets(workingDir = tempdir(), algoName = 'myOwnAlgo', algoFunction = 'estimateModel', libs = c('myOwnAlgo'), subset = 3)

# fifth example, evaluation for a customized subset with all test sets that have
# a pathological fraction <= 30%.
testsets <- loadTestsetDefinition()
progress <- evaluateBiomarkerTestSets(workingDir = tempdir(), algoName = 'myOwnAlgo', algoFunction = 'estimateModel', libs = c('myOwnAlgo'), subset = testsets$testsets$fractionPathol <= 0.3, ] )
# sixth example, evaluation forwarding additional parameters to the 'algoFunction'
progress <- evaluateBiomarkerTestSets(workingDir = tempdir(), algoName = 'myOwnAlgo',
algoFunction = 'estimateModel', libs = c('myOwnAlgo'),
sourceFiles = list("Test_RIEst_2pBoxCox"), params = list("model='2pBoxCox'"))

# seventh example, evaluation for indirect method that requires the number of
# decimal points as input
evaluateBiomarkerTestSets(workingDir = tempdir(), algoName = 'myOwnAlgo',
algoFunction = 'estimateModelDec', libs = c('myOwnAlgo'),
sourceFiles = "C:\Temp\Test_RIEst_dec.R", requireDecimals = TRUE)

# eighth example, evaluation for indirect method that directly estimates the percentiles
evaluateBiomarkerTestSets(workingDir = tempdir(), algoName="myOwnAlgo",
algoFunction="estimateRIs", libs="myOwnAlgo",
sourceFiles = "C:\Temp\Test_RIEst.R", requirePercentiles=TRUE)

## End(Not run)

---

formatNumber  

**Rounding method with trailing zeros.**

### Description
Rounding method with trailing zeros.

### Usage

formatNumber(x, digits)

### Arguments

- **x**  
  (numeric) value that is rounded

- **digits**  
  (integer) indicating the number of decimal places to be used

### Value
Rounded value with trailing zeros

### Author(s)

Christopher Rank <christopher.rank@roche.com>
generateBiomarkerTestSets

Convenience function to generate simulated data and save each test set as a separate file

Description

Convenience function to generate simulated data and save each test set as a separate file

Usage

generateBiomarkerTestSets(
  workingDir = "",
  subset = "all",
  rounding = TRUE,
  verbose = TRUE
)

Arguments

workingDir  (character) specifying the working directory where `workingDir/Data/biomarker` folders will be generated containing the simulated data

subset  (character, numeric, or data.frame) to specify for which subset the data should be generated and the algorithms later applied to. character options: `all` (default) for all test sets; a distribution type: `normal`, `skewed`, `heavilySkewed`, `shifted`; a biomarker: `Hb`, `Ca`, `FT4`, `AST`, `LACT`, `GGT`, `TSH`, `IgE`, `CRP`, `LDH`; `Runtime` for runtime analysis subset; numeric option: number of test sets per biomarker, e.g. 10; data.frame: customized subset of table with test set specification

rounding  (logical) indicating whether decimal places stated in test set specification should be applied (default, TRUE), if FALSE, data will be rounded to 5 decimal places to mimic unrounded data

verbose  (logical) indicating if the progress counter should be shown (default: TRUE)

Value

No return value, instead the data files are generated and saved in the workingDir

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
Examples

```r
## Not run:
workingDir <- "C:\\Temp\\RIBench\\"
generateBiomarkerTestSets(workingDir = workingDir)

## End(Not run)

# example generating a subset of 2 test sets per biomarker
generateBiomarkerTestSets(workingDir = tempdir(), subset = 2)
```

---

**generateBoxPlotOneAnalyte**

*Wrapper function to generate one boxplot for a specified analyte*

**Description**

Wrapper function to generate one boxplot for a specified analyte

**Usage**

```r
generateBoxPlotOneAnalyte(
  errorListAll,
  colList,
  nameList,
  catList,
  catLabels,
  a,
  errorParam,
  outline = TRUE,
  withMean = TRUE,
  withCats = TRUE,
  withDirect = TRUE,
  titlePart = NULL,
  outputDir,
  filenamePart = NULL,
  ylim1 = c(0, 100),
  ylim2 = c(100, 1000),
  ...
)
```
Arguments

- `errorListAll` (list) containing the overall benchmark results per algorithm
- `colList` (character) vector specifying the colors used for the different algorithms (should correspond to columns of benchmark results)
- `nameList` (character) vector specifying the names used in the legend (should correspond to columns of benchmark results), if NULL, colnames will be used
- `catList` (character) vector specifying the categories for which the boxes should be drawn
- `catLabels` (character) vector specifying the labels to the associated categories used for the x-axis
- `a` (character) specifying the analyte for which the boxplot should be generated
- `errorParam` (character) specifying for which error measure the plot should be generated
- `outline` (logical) indicating whether outliers should be drawn (TRUE, default), or not (FALSE)
- `withMean` (logical) indicating whether the mean should be plotted as well (default: TRUE)
- `withCats` (logical) set to TRUE if categories (e.g. pathological fraction) should be plotted (default: FALSE)
- `withDirect` (logical) indicating whether the box of the direct method should be elongated to facilitate comparison (default: TRUE)
- `titlePart` (character) specifying the latter part of the title
- `outputDir` (character) specifying a output directory
- `filenamePart` (character) specifying a filename for the plot
- `ylim1` (numeric) vector specifying the limits in y-direction for the first granular scale
- `ylim2` (numeric) vector specifying the limits in y-direction for the second less detailed scale
- `...` additional arguments passed forward to other functions

Value

No return value. Instead, a plot is generated.

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
generateBoxplotsDistTypes

Wrapper function to generate all boxplots for the specified distribution types split by defined categories

Description

Wrapper function to generate all boxplots for the specified distribution types split by defined categories

Usage

```r
generateBoxplotsDistTypes(
  errorListAll,  
colList,  
nameList,  
catList,  
catLabels,  
errorParam = "zzDevAbs_Ov",  
outline = TRUE,  
withMean = TRUE,  
withDirect = TRUE,  
withCats = TRUE,  
titlePart = NULL,  
outputDir = NULL,  
filenamePart = NULL,  
ylim1Vec = NULL,  
ylim2Vec = NULL,  
yticks1Vec = NULL,  
yticks2Vec = NULL,  
...
)
```

Arguments

- `errorListAll` (list) containing the overall benchmark results per algorithm
- `colList` (character) vector specifying the colors used for the different algorithms (should correspond to columns of benchmark results)
- `nameList` (character) vector specifying the names used in the legend (should correspond to columns of benchmark results), if NULL, colnames will be used
- `catList` (character) vector specifying the categories for which the boxes should be drawn
- `catLabels` (character) vector specifying the labels to the associated categories used for the x-axis
- `errorParam` (character) specifying for which error measure the plot should be generated
generateBoxplotsMultipleCats

**Outline**

The `generateBoxplotsMultipleCats` function is a wrapper function to generate all boxplots for the specified analytes split by defined categories.

**Usage**

```r
generateBoxplotsMultipleCats(
  analytes,  # A character vector specifying the analytes.
  errorListAll,  # A numeric matrix specifying the error ranges.
  colList,  # A character vector specifying the colors.
  nameList,  # A character vector specifying the names.
  category = c("fractionPathol", "fractionPathol_cum", "N", "N_cum", "OvFreq", "OvFreq_cum"),
  ...  # Additional arguments passed forward to other functions.
)
```

**Value**

No return value. Instead, a plot is generated.

**Author(s)**

Tatjana Ammer <tatjana.ammer@roche.com>

---

**Description**

Wrapper function to generate all boxplots for the specified analytes split by defined categories.

---

**generateBoxplotsMultipleCats**

*Wrapper function to generate all boxplots for the specified analytes split by defined categories*

---

**Usage**

`generateBoxplotsMultipleCats` is used to generate boxplots for specified analytes split by defined categories. The function accepts various arguments for customization, such as colors, names, categories, and additional parameters.

---

**Value**

No return value. Instead, a plot is generated.

**Author(s)**

Tatjana Ammer <tatjana.ammer@roche.com>
generateBoxplotsMultipleCats

catList = NULL,
catLabels = NULL,
errorParam = "zzDevAbs_Ov",
outline = TRUE,
withMean = TRUE,
withDirect = TRUE,
withCats = TRUE,
titlePart = NULL,
outputDir = NULL,
filenamePart = NULL,
ylim1Vec = NULL,
ylim2Vec = NULL,
yticks1Vec = NULL,
yticks2Vec = NULL,
...
)

Arguments

analytes         (character) vector specifying for which analytes the plots should be generated
errorListAll     (named list) containing the overall benchmark results per algorithm (names of
                  list elements should be the names of the algorithms)
colList          (character) vector specifying the colors used for the different algorithms (should
                  correspond to columns of benchmark results)
nameList         (character) vector specifying the names used in the legend (should correspond
                  to columns of benchmark results), if NULL, colnames will be used
category         (character) defining the category used for creating the subsets. All defined sub-
                  features are used for the categorization. Choose from "fractionPathol" (default),
                  "N", or "OvFreq", individual or cumulative ("_cum"); if category is set this will
                  be used to define catList and catLabels
catList          (character) vector specifying the categories for which the boxes should be drawn
catLabels        (character) vector specifying the labels to the associated categories used for the
                  x-axis
errorParam       (character) specifying for which error measure the plot should be generated
outline          (logical) indicating whether outliers should be drawn (TRUE, default), or not
                  (FALSE)
withMean         (logical) indicating whether the mean should be plotted as well (default: TRUE)
withDirect       (logical) indicating whether the box of the direct method should be elongated to
                  facilitate comparison (default: TRUE)
withCats         (logical) set to TRUE if categories (e.g. pathological fraction) should be plotted
                  (default: FALSE)
titlePart        (character) specifying the latter part of the title
outputDir        (character) specifying an output directory
filenamePart     (character) specifying a filename for the plot
generateDataFiles

ylim1Vec (numeric) vector specifying the limits in y-direction for the first granular scale
ylim2Vec (numeric) vector specifying the limits in y-direction for the second less detailed scale
yticks1Vec (numeric) vector specifying the ticks in y-direction for the first granular scale
yticks2Vec (numeric) vector specifying the ticks in y-direction for the second less detailed scale
... additional arguments passed forward to other functions

Value
No return value. Instead, a plot is generated.

Author(s)
Tatjana Ammer <tatjana.ammer@roche.com>

---

**generateDataFiles**  
Generate simulated data with one start seed for each biomarker and save each test set as separate file

Description
Generate simulated data with one start seed for each biomarker and save each test set as separate file

Usage

```r
generateDataFiles(
  tableTCs = NULL,
  outputDir = NULL,
  rounding = TRUE,
  verbose = TRUE
)
```

Arguments

tableTCs (data.frame) containing all information about the simulated test cases
outputDir (character) specifying the output directory where the data files should be written to
rounding (logical) indicating whether decimal places stated in tableTCs should be applied (default, TRUE), if FALSE, data will be rounded to 5 decimal places to mimic unrounded data
verbose (logical) indicating if the progress counter should be shown (default: TRUE)
generateScatterplotsAll

Description
Wrapper function to generate scatterplots for the specified analytes

Value
No return value, instead the data files are generated

Author(s)
Tatjana Ammer <tatjana.ammer@roche.com>

---

generateMD5

Generate an MD5 hash sum for any R object.

Description
Generate an MD5 hash sum for any R object.

Usage
generateMD5(x)

Arguments
x (object) any R object.

Value
(character) MD5 hash sum of the input object.

Author(s)
Christopher Rank <christopher.rank@roche.com>

---

generateScatterplotsAll

Wrapper function to generate scatterplots for the specified analytes

Description
Wrapper function to generate scatterplots for the specified analytes
Usage

generateScatterplotsAll(
  analytes, errorListAll, colList = NULL, nameList, tableTCs,
  errorParam = "zzDevAbs", withColorCat = NULL, titlePart = NULL,
  outputDir = NULL, filenamePart = NULL, ylim = NULL,
  xlim = NULL, xlab = NULL, ylab = NULL,
  ...
)

Arguments

analytes (character) vector specifying for which analytes the scatterplot should be generated
errorListAll (list) containing the overall benchmark results per algorithm
collList (character) vector specifying the colors used for the different algorithms (should correspond to columns of benchmark results)
nameList (character) vector specifying the names used in the legend (should correspond to columns of benchmark results), if NULL, colnames will be used	tableTCs (data frame) containing all test case information
erroParam (character) specifying for which error measure the plot should be generated
withColorCat (character) indicating if plot should be colored according to the pathological fraction ("fractionPathol"), sample size ("N"), or "overlapPatholLeft", "overlapPatholRight"
titlePart (character) specifying the latter part of the title
outputDir (character) specifying a output directory
filenamePart (character) specifying a filename for the plot
ylim (numeric) vector specifying the limits in y-direction for the first granular scale
xlim (numeric) vector specifying the limits in y-direction for the second less detailed scale
xlab (character) specifying x-axis label
ylab (character) specifying y-axis label
... additional arguments passed forward to other functions
getBenchmarkResults

Value

No return value. Instead, a plot is generated.

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

Description

Computing benchmark table with the mean overall results.

Usage

getBenchmarkResults(
  errorList,
  nameVec,
  tableTCs,
  errorParam = "zzDevAbsCutoff_Ov",
  cutoffZ = 5,
  catList = c("fractionPathol <= 0.20 & N <= 5000",
    "fractionPathol <= 0.20 & N > 5000", "fractionPathol > 0.20 & N <= 5000",
    "fractionPathol > 0.20 & N > 5000"),
  catLabels = c("lowPlowN", "lowPhighN", "highPlowN", "highPhighN"),
  perfCombination = c("mean", "median", "sum")
)

Arguments

errorList (list) containing the the computed errors for the different (indirect) methods/algorithms
nameVec (character) vector specifying the names of the different (indirect) methods/algorithms
tableTCs (data.frame) containing all information about the simulated test sets
errorParam (character) specifying for which error parameter the data frame should be generated
cutoffZ (integer) specifying if and if so which cutoff for the absolute z-score deviation should be used to classify results as implausible and exclude them from the overall benchmark score (default: 5)
catList (character) vector containing the categories to split the dataset
catLabels (character) vector containing the labels that will be used for the categories
perfCombination (character) specifying which measure should be used to compute the overall benchmark score; choose from "mean" (default), "median", or "sum"
getRI

Value
(data frame) containing the computed benchmark results

Author(s)
Tatjana Ammer <tatjana.ammer@roche.com>

getRI
Method to calculate reference intervals (percentiles) for objects of class 'RWDRI'

Description
Method to calculate reference intervals (percentiles) for objects of class 'RWDRI'

Usage
getRI(
  x,  
  RIperc = c(0.025, 0.975), 
  CIprop = 0.95, 
  pointEst = c("fullDataEst", "medianBS", "meanBS"), 
  truncNormal = FALSE, 
  Scale = c("original", "transformed") 
)

Arguments
x (object) of class 'RWDRI'
RIperc (numeric) value specifying the percentiles, which define the reference interval
CIprop (numeric) value specifying the central region for estimation of confidence intervals
pointEst (character) specifying the point estimate determination: (1) using the full dataset ("fullDataEst"), (2) calculating the median from all bootstrap samples ("medianBS"), (2) works only if NBootstrap > 0 (3) calculating the mean from all bootstrap samples ("meanBS"), (3) works only if NBootstrap > 0
truncNormal (logical) specifying if a normal distribution truncated at zero shall be assumed
Scale (character) specifying if percentiles are calculated on the original scale ("Or") or the transformed scale ("Tr")

Value
(data.frame) with columns for percentile, point estimate and confidence intervals.

Author(s)
Christopher Rank <christopher.rank@roche.com>, Tatjana Ammer <tatjana.ammer@roche.com>
getRIsAllwithoutModel  Function for retrieving reference intervals if directly computed

Description
Function for retrieving reference intervals if directly computed

Usage
getRIsAllwithoutModel(analytes, algo, resIn, tableTCs)

Arguments
- **analytes** (character) listing all markers for which the result files should be parsed
- **algo** (character) specifying used algorithm
- **resIn** (list) with all calculated results for all markers as RWDRI objects
- **tableTCs** (data.frame) containing all information about the simulated test sets

Value
list with the calculated reference intervals as data frame for each marker

Author(s)
Tatjana Ammer <tatjana.ammer@roche.com>

getRuntime  Helper function to compute runtime statistics

Description
Helper function to compute runtime statistics

Usage
getRuntime(x, analyte)

Arguments
- **x** (data.frame) with one column specifying the Runtime
- **analyte** (character) specifying current analyzed marker

Value
(data.frame) containing runtime statistics (min, mean, median, max)
getSubset

**Author(s)**

Tatjana Ammer <tatjana.ammer@roche.com>

---

**getSubset**

*Function to group the data according to a specified feature.*

**Description**

The feature can either be the pathological fraction, the sample size or the overlap (category) individually or cumulative (_cum). For an individualized categorisation see `getSubsetForDefinedCats`.

**Usage**

```r
getSubset(
  subsetDef,
  distType = FALSE,
  tableTCs,
  errorList,
  category = c("fractionPathol", "fractionPathol_cum", "N", "N_cum", "OvFreq", "OvFreq_cum"),
  restrict = NULL
)
```

**Arguments**

- `subsetDef` (character) listing either the analytes or distribution types for which the result files should be parsed
- `distType` (logical) indicating if parameter `subsetDef` refers to analytes (FALSE, default) or distribution types (TRUE)
- `tableTCs` (data.frame) containing all information about the simulated test sets
- `errorList` (list) containing for each method the table with the computed error measurements
- `category` (character) defining the category used for creating the subsets. All defined sub-features are used for the categorization. Choose from "fractionPathol" (default), "N", or "OvFreq", individual or cumulative (_cum)
- `restrict` (character) indicating whether test sets should be filtered according to specified restriction, default NULL, e.g. `fractionPathol <= 0.30`

**Value**

(list) containing the performance measurements grouped according to specified subset definition and categories

**Author(s)**

Tatjana Ammer <tatjana.ammer@roche.com>
getSubsetForDefinedCats

Function to group the data according to a specified feature.

Description

Function to group the data according to a specified feature.

Usage

getSubsetForDefinedCats(
  subsetDef,
  distType = FALSE,
  tableTCs,
  errorList,
  catList = NULL,
  catLabels = NULL,
  restrict = NULL
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subsetDef</td>
<td>(character) listing either the analytes or distribution types for which the result files should be parsed</td>
</tr>
<tr>
<td>distType</td>
<td>(logical) indicating if <code>subsetDef</code> refers to analytes (FALSE, default) or distribution types (TRUE)</td>
</tr>
<tr>
<td>tableTCs</td>
<td>(data.frame) containing all information about the simulated test sets</td>
</tr>
<tr>
<td>errorList</td>
<td>(list) containing the table with the computed error measurements</td>
</tr>
<tr>
<td>catList</td>
<td>(list) containing the categories to split the dataset</td>
</tr>
<tr>
<td>catLabels</td>
<td>(list) containing the labels that will be used for the categories</td>
</tr>
<tr>
<td>restrict</td>
<td>(character) indicating whether testcases should be filtered according to specified restriction, default NULL, e.g. fractionPathol &lt;= 0.30</td>
</tr>
</tbody>
</table>

Value

(list) containing the performance measurements grouped according to specified subset definition and categories

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
**invBoxCox**

Inverse of the one-parameter Box-Cox transformation.

**Description**

Inverse of the one-parameter Box-Cox transformation.

**Usage**

```r
invBoxCox(x, lambda)
```

**Arguments**

- `x` (numeric) data to be transformed
- `lambda` (numeric) Box-Cox transformation parameter

**Value**

(numeric) vector with inverse Box-Cox transformation of `x`

**Author(s)**

Andre Schuetzenmeister <andre.schuetzenmeister@roche.com>

---

**loadTestsetDefinition**

Convenience function to load the table with the information about the pre-defined test sets

**Description**

Convenience function to load the table with the information about the pre-defined test sets

**Usage**

```r
loadTestsetDefinition()
```

**Value**

(data frame) containing the pre-defined parameter combinations to generate the simulations

**Author(s)**

Tatjana Ammer <tatjana.ammer@roche.com>
mergeAnalytes

Function to combine analytes for defined categories

Description

The feature can either be the pathological fraction, the sample size or the overlap (category) individually or cumulative (_cum). For a individualized categorisation see getSubsetForDefinedCats.

Usage

mergeAnalytes(
  tableTCs,
  errorList,
  catList = NULL,
  catLabels = NULL,
  distTypes = TRUE
)

Arguments

tableTCs (data.frame) containing all information about the simulated test sets

errorList (list) containing for each method the table with the computed error measurements

catList (list) containing the categories to split the dataset

catLabels (list) containing the labels that will be used for the categories

distTypes (logical) indicating if 'catList' refers to analytes (FALSE, default) or distribution types (TRUE)

Value

(list) containing the merged performance measurements grouped according to specified category

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
mergeSummaryErrors

Helper function to combine all computed summary errors

Description

Helper function to combine all computed summary errors

Usage

mergeSummaryErrors(
   errorList,  
   nameVec,  
   errorParam = "MedianAbsPercErrorOV",  
   cutoffZ = FALSE
)

Arguments

errorList (list) of the error lists for the different methods for which the summary errors should be combined
nameVec (character) vector specifying the names of the methods
errorParam (character) specifying for which error parameter the data frame should be generated
cutoffZ (logical) indicating if a cutoff was set, needed for CRP case

Value

(data frame) containing the summary errors per analyte per method

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

plotBarplot

Plot method for generating a barplot out of the benchmark results

Description

Plot method for generating a barplot out of the benchmark results
Usage

plotBarplot(
  benchmarkRes,
  perDistType = FALSE,
  colList,
  nameList = NULL,
  withLabels = FALSE,
  withHorizLines = FALSE,
  title = NULL,
  xlim = NULL,
  xlab = "Mean of Absolute Z-Score Deviations",
  outputDir = NULL,
  filename = NULL,
  ...
)

Arguments

  benchmarkRes  (data frame) containing the overall benchmark results
  perDistType   (logical) indicating if one overall plot should be generated or if it should be
                 separated by the distribution type
  colList       (character) vector specifying the colors used for the different algorithms (should
                 correspond to columns of benchmark results)
  nameList      (character) vector specifying the names used in the legend (should correspond
                 to columns of benchmark results), if NULL, colnames will be used
  withLabels    (logical) indicating whether the corresponding values should be plotted as well
                 (default: FALSE)
  withHorizLines (logical) indicating whether horizontal lines should be plotted for a better visual
                   separation of the different categories (default: FALSE)
  title         (character) specifying plot title
  xlim          (numeric) vector specifying the limits in x-direction
  xlab          (character) specifying the x-axis label
  outputDir     (character) specifying a output directory
  filename      (character) specifying a filename for the plot
  ...           additional arguments passed forward to other functions

Value

  No return value. Instead, a plot is generated.

Author(s)

  Tatjana Ammer <tatjana.ammer@roche.com>
plotBoxplot

Plot method for generating a boxplot of the benchmark results

Description

Plot method for generating a boxplot of the benchmark results

Usage

plotBoxplot(
  errorList,
  colList,
  nameList,
  outline = TRUE,
  withMean = TRUE,
  withCats = FALSE,
  withDirect = TRUE,
  title = "",
  outputDir = NULL,
  filename = NULL,
  ylim1 = c(0, 100),
  ylim2 = c(100, 1000),
  ...
)

Arguments

errorList containing the overall benchmark results
colList (character) vector specifying the colors used for the different algorithms (should correspond to columns of benchmark results)
nameList (character) vector specifying the names used in the legend (should correspond to columns of benchmark results), if NULL, colnames will be used
outline (logical) indicating whether outliers should be drawn (TRUE, default), or not (FALSE)
withMean (logical) indicating whether the mean should be plotted as well (default: TRUE)
withCats (logical) set to TRUE if categories (e.g. pathological fraction) should be plotted (default: FALSE)
withDirect (logical) indicating whether the box of the direct method should be elongated to facilitate comparison (default: TRUE)
title (character) specifying plot title
outputDir (character) specifying a output directory
filename (character) specifying a filename for the plot
ylim1 (numeric) vector specifying the limits in y-direction for the first granular scale
ylim2 (numeric) vector specifying the limits in y-direction for the second less detailed scale
...
  additional arguments passed forward to other functions
plotScatterplot

Plot method for generating a scatterplot

Description

Plot method for generating a scatterplot

Usage

plotScatterplot(
  errorList,
  colList,
  nameList,
  withColor = NULL,
  cats = NULL,
  title = "",
  outputDir = NULL,
  filename = NULL,
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  ...
)

Arguments

- **errorList** (data frame) containing the overall benchmark results
- **colList** (character) vector specifying the colors used for the different algorithms (should correspond to columns of benchmark results)
- **nameList** (character) vector specifying the names used in the legend (should correspond to columns of benchmark results), if NULL, colnames will be used
- **withColor** (character) indicating if plot should be colored according to pathological fraction, sample size or pathological overlap left / right
- **cats** (character) specifying the category labels
- **title** (character) specifying plot title
- **outputDir** (character) specifying a output directory
- **filename** (character) specifying a filename for the plot

Value

No return value. Instead, a plot is generated.

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
print.RWDRI  

xlim (numeric) vector specifying the limits in y-direction for the first granular scale  
ylim (numeric) vector specifying the limits in y-direction for the second less detailed scale  
xlab (character) specifying x-axis label  
ylab (character) specifying y-axis label  
... additional arguments passed forward to other functions  

Value  
No return value. Instead, a plot is generated.  

Author(s)  
Tatjana Ammer <tatjana.ammer@roche.com>  

print.RWDRI Standard print method for objects of class 'RWDRI'  

Description  
Standard print method for objects of class 'RWDRI'  

Usage  
## S3 method for class 'RWDRI'  
print(  
x,  
RIperc = c(0.025, 0.975),  
CIprop = 0.95,  
pointEst = c("fullDataEst", "medianBS", "meanBS"),  
truncNormal = FALSE,  
...  
)  

Arguments  
x (object) of class 'RWDRI'  
RIperc (numeric) value specifying the percentiles, which define the reference interval  
CIprop (numeric) value specifying the central region for estimation of confidence intervals  
pointEst (character) specifying the point estimate determination: (1) using the full dataset ("fullDataEst"), (2) calculating the median from the bootstrap samples ("medianBS"), (2) works only if NBootstrap > 0 (3) calculating the mean from the bootstrap samples ("meanBS"), (3) works only if NBootstrap > 0  
truncNormal (logical) specifying if a normal distribution truncated at zero shall be assumed  
... additional arguments passed forward to other functions.
Value

No return value. Instead, a summary is printed.

Author(s)

Christopher Rank <christopher.rank@roche.com>

progressInd Function for setting up the progress indicator.

Description

Function for setting up the progress indicator.

Usage

progressInd(value, maxValue, nCharMsg = 0)

Arguments

- value (integer) indicating the current number
- maxValue (integer) indicating the maximum number
- nCharMsg (integer) indicating the number of characters the message already has

Value

(character) returning generated progress message

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

readResultFiles Function for reading in the result files for one marker

Description

Function for reading in the result files for one marker

Usage

readResultFiles(analyte, algo, path = NULL, tableTCs = NULL)
readResultFilesAll

Arguments

analyte (character) specifying analyte
algo (character) specifying used algorithm
path (character) specifying path to Results directories
tableTCs (data frame) containing all information about the simulated test sets

Value

list with calculated results as RWDRI objects

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

readResultFilesAll Function for reading all results files.

Description

Function for reading all results files.

Usage

readResultFilesAll(analytes, algo, baseDir = NULL, inputDir = NULL, tableTCs)

Arguments

analytes (character) listing all analytes for which the result files should be parsed
algo (character) specifying used algorithm
baseDir (character) specifying the baseDir: Results will be used from baseDir/Results/algo/marker
if baseDir is set, inputDir will be ignored; if baseDir is NULL, the current working directory will be used
inputDir (character) specifying path directly to Results directories
tableTCs (data frame) containing all information about the simulated test sets

Value

list with all calculated results as RWDRI objects for each marker

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
readResultsAndComputeErrors

Function to read the result files and compute performance measures to create customized plots afterwards

Description

Function to read the result files and compute performance measures to create customized plots afterwards

Usage

readResultsAndComputeErrors(
  workingDir = getwd(),
  algoName = NULL,
  subset = "all",
  cutoffZ = 5,
  ...
)

Arguments

workingDir (character) specifying the working directory: Plots will be stored in workingDir/evalFolder and results will be used from workingDir/Results/algoName/biomarker;

algoName (character) vector specifying one algorithm for which the performance measures should be evaluated

subset (character, numeric, or data.frame) to specify for which subset the algorithm should be executed. character options: 'all' (default) for all test sets, a distribution type: 'normal', 'skewed', 'heavilySkewed', 'shifted'; a biomarker: 'Hb', 'Ca', 'FT4', 'AST', 'LACT', 'GGT', 'TSH', 'IgE', 'CRP', 'LDH'; 'runtime' for runtime analysis subset; numeric option: number of test sets per biomarker, e.g. 10; data.frame: customized subset of table with test set specifications

cutoffZ (integer) specifying if and if so which cutoff for the absolute z-score deviation should be used to classify results as implausible and exclude them from the overall benchmark score (default: 5)

... additional arguments to be passed to the method truncNormal (logical) specifying if a normal distribution truncated at zero shall be assumed

Value

(list) with (data frame) and a (list) with the computed performance measures

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
restrictSet

Function to get error subsets for defined category and restriction.

Description

Function to get error subsets for defined category and restriction.

Usage

restrictSet(overallCat, tableTCs, errorList, distType = TRUE, restrict = NULL)

Arguments

- `overallCat` (list) containing the categories to split the dataset
- `tableTCs` (data.frame) containing all information about the simulated test sets
- `errorList` (list) containing for each method the table with the computed error measurements
- `distType` (logical) indicating if 'overallCat' refers to analytes (FALSE, default) or distribution types (TRUE)
- `restrict` (character) indicating whether testcases should be filtered according to specified restriction, default NULL, e.g. fractionPathol <= 0.30

Value

(list) containing the merged performance measurements grouped according to specified category

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

runDirectMethod

Convenience function to simulate the direct method for the specified subset

Description

Convenience function to simulate the direct method for the specified subset

Usage

runDirectMethod(tableTCs = NULL, N = 120, cutoffZ = 5)
runTC_usingRscript

Function for running test sets per algorithm per marker with calling Rscript for each test set

Arguments

tableTCs (data frame) containing the pre-defined parameter combinations to generate the simulations

N (integer) specifying the number of samples used as sample size for the direct method, default: 120
cutoffZ (numeric) specifying if a cutoff should be used to classify results as implausible and exclude from analysis

Value

(data frame) with computed performance measures

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>

Examples

# example to run direct method only for test sets for hemoglobin (Hb)
testsets <- loadTestsetDefinition()
directRes <- runDirectMethod(tableTCs = testsets[testsets$Analyte == "Hb",], N = 120, cutoffZ = 5)
runTC_usingRscript

```r
outputDir = NULL,
inputDir = NULL,
timeLimit = 14400,
subsetDef = "all",
verbose = TRUE,
showWarnings = FALSE,
...
)
```

**Arguments**

- **biomarker** (character) specifying the biomarker for which the algorithm should calculate RIs
- **algoName** (character) specifying the name of the algorithm that is evaluated
- **algoFunction** (character) specifying the name of the function needed for estimating RIs
- **sourceFiles** (list) containing all source files needed for executing the algorithm
- **libs** (list) containing all libraries needed for executing the algorithm
- **params** (list) with additional parameters needed for calling algoFunction
- **decimals** (logical) indicating whether the algorithm needs the number of decimal places (TRUE) or not (FALSE, default)
- **ris** (logical) indicating whether only percentiles and no model is estimated
- **RIperc** (numeric) value specifying the percentiles, which define the reference interval
- **tableTCs** (data.frame) with the information about the simulated test sets
- **outputDir** (character) specifying the outputDir: Results will be stored in outputDir/Results/algo/biomarker
- **inputDir** (character) specifying the inputDir: Data files should be stored in inputDir/Data/biomarker
- **timeLimit** (integer) specifying the maximum amount of time in seconds allowed to execute one single estimation (default: 14400 sec (4h))
- **subsetDef** (character) describing the specified subset of all test sets the algorithm is applied to, used for naming the progress file
- **verbose** (logical) indicating if the progress counter should be shown (default: TRUE)
- **showWarnings** (logical) indicating whether warnings from the call to the indirect method/algorithm should be shown (default: FALSE)
- ... additional arguments to be passed to the method

**Value**

(data frame) containing information about the test sets where the algorithm terminated the R session or failed to report a result

**Author(s)**

Tatjana Ammer <tatjana.ammer@roche.com>
setupDirStructure

Convenience function to set up the directory structure used for storing data and results.

Description

Convenience function to set up the directory structure used for storing data and results.

Usage

setupDirStructure(
  outputDir = NULL,
  onlyData = FALSE,
  onlyResults = FALSE,
  tableTCs = NULL,
  algoName = NULL
)

Arguments

  outputDir  (character) specifying the base output directory. From here, Data/biomarker and Result/algoName/biomarker directories are generated
  onlyData   (logical) if set to TRUE, only the biomarker subdirectories are generated, name of output directory is used as it is (default:FALSE)
  onlyResults (logical) if set to TRUE, only the algoName/biomarker subdirectories are generated, name of output directory is used as it is (default:FALSE)
  tableTCs   (data frame) containing the pre-defined parameter combinations to generate the simulations
  algoName   (character) specifying the name of the algorithm used for creating the subdirectory

Value

No return value. Instead, the directory structure is set up.

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
writeResFile

Helper function to write result file when time out occurred or R session terminated

Description

Helper function to write result file when time out occurred or R session terminated.

Usage

```r
writeResFile(
  algoName,  
biomarker,  
N = 0,  
error = NULL,  
runtime = NULL,  
filename = NULL,  
outputDir = NULL
)
```

Arguments

- `algoName` (character): specifying the name of the algorithm that is evaluated.
- `biomarker` (character): specifying the biomarker for which the algorithm should calculate RIs.
- `N` (numeric): specifying the number of input data points.
- `error` (character): specifying the type of error (e.g. timeout, RSessionTerminated).
- `runtime` (numeric): specifying the computation time up until the error occurred.
- `filename` (character): specifying the filename for which the algorithm failed.
- `outputDir` (character): specifying the outputDir: Data files should be stored in outputDir/Data/biomarker and Results will be stored in outputDir/Results/algo/biomarker.

Author(s)

Tatjana Ammer <tatjana.ammer@roche.com>
Index

* package
  RIBench-package, 3

  addGrid, 4
  as.rgb, 4
  BoxCox, 5
  computeDirect, 6
  computePerfMeas, 7
  computePerfMeasAll, 8
  computeRIs, 8
  computeRIsAll, 9
  computeRuntimeAll, 10
  computeSubResults, 10
  defineSubset, 11
  evaluateAlgorithmResults, 3, 12
  evaluateBiomarkerTestSets, 3, 13
  formatNumber, 16
  generateBiomarkerTestSets, 3, 17
  generateBoxPlotOneAnalyte, 18
  generateBoxplotsDistTypes, 20
  generateBoxplotsMultipleCats, 21
  generateDataFiles, 23
  generateMDS, 24
  generateScatterplotsAll, 24
  getBenchmarkResults, 26
  getRI, 27
  getRIIsAllwithoutModel, 28
  getRuntime, 28
  getSubset, 29
  getSubsetForDefinedCats, 30
  invBoxCox, 31
  loadTestsetDefinition, 31
  mergeAnalytes, 32
  mergeSummaryErrors, 33
  plotBarplot, 33
  plotBoxplot, 35
  plotScatterplot, 36
  print.RWDRI, 37
  progressInd, 38
  readResultFiles, 38
  readResultFilesAll, 39
  readResultsAndComputeErrors, 40
  restrictSet, 41
  RIBench (RIBench-package), 3
  RIBench-package, 3
  runDirectMethod, 41
  runTC_usingRscript, 42
  setupDirStructure, 44
  writeResFile, 45